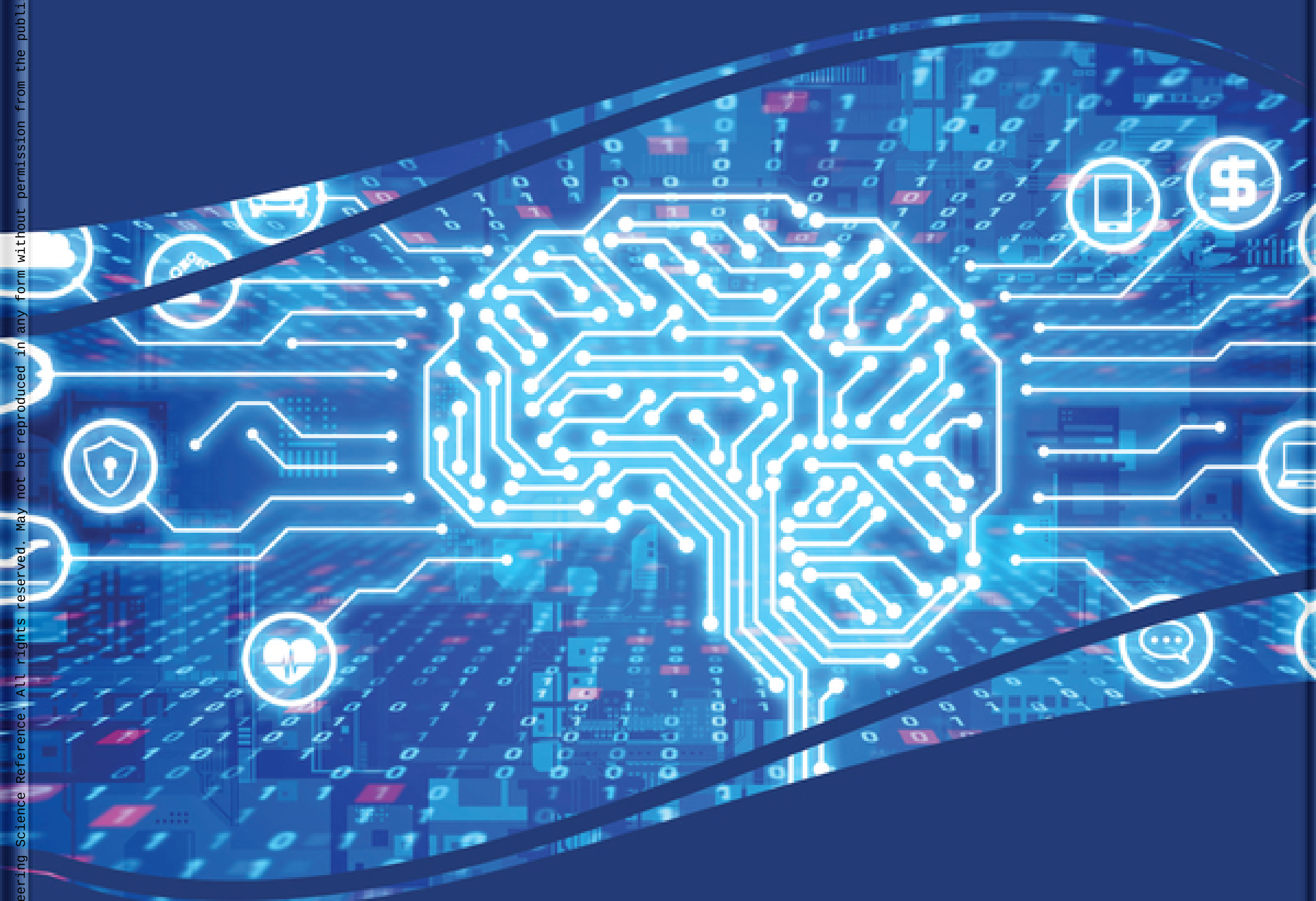
Premier Reference Source

Deep Learning Applications and Intelligent Decision Making in Engineering



Karthikrajan Senthilnathan, Balamurugan Shanmugam, Dinesh Goyal, Iyswarya Annapoorani, and Ravi Samikannu

Deep Learning Applications and Intelligent Decision Making in Engineering

Karthikrajan Senthilnathan
Revoltaxe India Pvt Ltd, Chennai, India

Balamurugan Shanmugam
Quants IS & CS, India

Dinesh Goyal
Poornima Institute of Engineering and Technology, India

Iyswarya Annapoorani
VIT University, India

Ravi Samikannu
Botswana International University of Science and Technology, Botswana

IGI Global
PUBLISHER of TIMELY KNOWLEDGE

A volume in the Advances in Computational Intelligence and Robotics (ACIR) Book Series

Published in the United States of America by
IGI Global
Engineering Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA, USA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com

Library of Congress Cataloging-in-Publication Data

Names: Senthilnathan, Karthikrajan, 1991- editor. | Shanmugam, Balamurugan, 1985- editor. | Goyal, Dinesh, 1976- editor. | Annapoorani, Iyswarya, 1976- editor. | Samikannu, Ravi, 1982- editor.
Title: Deep learning applications and intelligent decision making in engineering / Karthikrajan Senthilnathan, Balamurugan Shanmugam, Dinesh Goyal, Iyswarya Annapoorani and Ravi Samikannu, editors.
Description: Hershey, PA : Engineering Science Reference, 2020. | Includes bibliographical references. | Summary: "This book explores the application of deep learning in building a smart world, ranging from smart cities, smart agriculture to smart homes"-- Provided by publisher.

Identifiers: LCCN 2019036934 (print) | LCCN 2019036935 (ebook) | ISBN 9781799821083 (h/c) | ISBN 9781799821090 (s/c) | ISBN 9781799821106 (eISBN)
Subjects: LCSH: Engineering--Data processing. | Computational intelligence. | Machine learning. | Smart materials.
Classification: LCC TA347.A78 D44 2020 (print) | LCC TA347.A78 (ebook) | DDC 620.00285/631--dc23
LC record available at https://lccn.loc.gov/2019036934
LC ebook record available at https://lccn.loc.gov/2019036935

This book is published in the IGI Global book series Advances in Computational Intelligence and Robotics (ACIR) (ISSN: 2327-0411; eISSN: 2327-042X)

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

All work contributed to this book is new, previously-unpublished material. The views expressed in this book are those of the authors, but not necessarily of the publisher.

For electronic access to this publication, please contact: eresources@igi-global.com.

Advances in Computational Intelligence and Robotics (ACIR) Book Series

Ivan Giannoccaro
University of Salento, Italy

ISSN:2327-0411
EISSN:2327-042X

Mission

While intelligence is traditionally a term applied to humans and human cognition, technology has progressed in such a way to allow for the development of intelligent systems able to simulate many human traits. With this new era of simulated and artificial intelligence, much research is needed in order to continue to advance the field and also to evaluate the ethical and societal concerns of the existence of artificial life and machine learning.

The **Advances in Computational Intelligence and Robotics (ACIR) Book Series** encourages scholarly discourse on all topics pertaining to evolutionary computing, artificial life, computational intelligence, machine learning, and robotics. ACIR presents the latest research being conducted on diverse topics in intelligence technologies with the goal of advancing knowledge and applications in this rapidly evolving field.

Coverage

- Agent technologies
- Synthetic Emotions
- Brain Simulation
- Evolutionary Computing
- Heuristics
- Computer Vision
- Pattern Recognition
- Intelligent control
- Natural Language Processing
- Algorithmic Learning

IGI Global is currently accepting manuscripts for publication within this series. To submit a proposal for a volume in this series, please contact our Acquisition Editors at Acquisitions@igi-global.com or visit: http://www.igi-global.com/publish/.

The Advances in Computational Intelligence and Robotics (ACIR) Book Series (ISSN 2327-0411) is published by IGI Global, 701 E. Chocolate Avenue, Hershey, PA 17033-1240, USA, www.igi-global.com. This series is composed of titles available for purchase individually; each title is edited to be contextually exclusive from any other title within the series. For pricing and ordering information please visit http://www.igi-global.com/book-series/advances-computational-intelligence-robotics/73674. Postmaster: Send all address changes to above address.

Titles in this Series

For a list of additional titles in this series, please visit:
http://www.igi-global.com/book-series/advances-computational-intelligence-robotics/73674

Analyzing Future Applications of AI, Sensors, and Robotics in Society
Thomas Heinrich Musiolik (Berlin University of the Arts, Germany) and Adrian David Cheok (iUniversity, Tokyo, Japan)
Engineering Science Reference • © 2021 • 335pp • H/C (ISBN: 9781799834991) • US $225.00

AI-Based Services for Smart Cities and Urban Infrastructure
Kangjuan Lyu (SILC Business School, Shanghai University, China) Min Hu (SILC Business School, Shanghai University, China) Juan Du (SILC Business School, Shanghai University, China) and Vijayan Sugumaran (Oakland University, USA)
Engineering Science Reference • © 2021 • 330pp • H/C (ISBN: 9781799850243) • US $195.00

Intelligent Computations Applications for Solving Complex Problems
Naveen Dahiya (Maharaja Surajmal Institute of Technology, New Delhi, India) Zhongyu Lu (University of Huddersfield, UK) Vishal Bhatnagar (Ambedkar Institute of Advance Communication Technologies & Research, India) and Pardeep Sangwan (Maharaja Surajmal Institute of Technology, New Delhi, India)
Engineering Science Reference • © 2021 • 300pp • H/C (ISBN: 9781799847939) • US $225.00

Artificial Intelligence and the Journey to Software 2.0 Emerging Research and Opportunities
Divanshi Priyadarshni Wangoo (Indira Gandhi Delhi Technical University for Women, India)
Engineering Science Reference • © 2021 • 150pp • H/C (ISBN: 9781799843276) • US $165.00

Practical Applications and Use Cases of Computer Vision and Recognition Systems
Chiranji Lal Chowdhary (Vellore Institute of Technology, India) and B.D. Parameshachari (GSSS Institute of Engineering and Technology for Women, India)
Engineering Science Reference • © 2020 • 300pp • H/C (ISBN: 9781799849247) • US $195.00

Machine Learning Applications in Non-Conventional Machining Processes
Goutam Kumar Bose (Haldia Institute of Technology, Haldia, India) and Pritam Pain (Haldia Institute of Technology, India)
Engineering Science Reference • © 2020 • 300pp • H/C (ISBN: 9781799836247) • US $195.00

IGI Global
PUBLISHER of TIMELY KNOWLEDGE

701 East Chocolate Avenue, Hershey, PA 17033, USA
Tel: 717-533-8845 x100 • Fax: 717-533-8661
E-Mail: cust@igi-global.com • www.igi-global.com

Table of Contents

Detailed Table of Contents

The internet of things (IoT), big data analytics, and deep learning (DL) applications in the mechanical internet are expanding. The current digital era has various sensory devices for a wide range of fields and applications, which all generate various sensory data. DL is being applied for handling big data and has achieved great success in the IoT and other fields. The applications for data streams to discover new information, predict future insights, and make control decisions are crucial processes that make the IoT a worthy paradigm for businesses and a quality-of-life improving technology. This chapter provides a detailed account of the IoT domain, machine learning, and DL techniques and applications. The IoT that consists of DL with intelligence backgrounds is also discussed. Recent research on DL in the IoT within the big data domain is also discussed. Current challenges and potential areas for future research are discussed.

Deep learning is one of the popular machine learning strategies that learns in a supervised or unsupervised manner by forming a cascade of multiple layers of non-linear processing units. It is inspired by the way of information processing and communication pattern of the typical biological nervous system. The deep learning algorithms learn through multiple levels of abstractions and hierarchy of concepts; as a result, it is found to be more efficient than the conventional non-deep machine learning algorithms. This chapter explains the basics of deep learning by highlighting the necessity of deep learning over non-deep learning. It also covers discussion on several recently developed deep learning architectures and popular tools available in market for deep learning, which includes Tensorflow, PyTorch, Keras, Caffe, Deeplearning4j, Pylearn2, Theano, CuDDN, CUDA-Convnet, and Matlab.

The proposed IoT-based CDM system was experimented, and the results obtained encourage both the farmers and the researchers in this field.

Sarangam Kodati, Teegala Krishna Reddy Engineering College, India
Jeeva Selvaraj, Brilliant Institute of Engineering and Technology, India

Data mining is the most famous knowledge extraction approach for knowledge discovery from data (KDD). Machine learning is used to enable a program to analyze data, recognize correlations, and make usage on insights to solve issues and/or enrich data and because of prediction. The chapter highlights the need for more research within the usage of robust data mining methods in imitation of help healthcare specialists between the diagnosis regarding heart diseases and other debilitating disease conditions. Heart disease is the primary reason of death of people in the world. Nearly 47% of death is caused by heart disease. The authors use algorithms including random forest, naïve Bayes, support vector machine to analyze heart disease. Accuracy on the prediction stage is high when using a greater number of attributes. The goal is to function predictive evaluation using data mining, using data mining to analyze heart disease, and show which methods are effective and efficient.

Deepali R. Vora, Vidyalankar Institute of Technology, India
Kamatchi R. Iyer, Amity University, India

The goodness measure of any institute lies in minimising the dropouts and targeting good placements. So, predicting students' performance is very interesting and an important task for educational information systems. Machine learning and deep learning are the emerging areas that truly entice more research practices. This research focuses on applying the deep learning methods to educational data for classification and prediction. The educational data of students from engineering domain with cognitive and non-cognitive parameters is considered. The hybrid model with support vector machine (SVM) and deep belief network (DBN) is devised. The SVM predicts class labels from preprocessed data. These class labels and actual class labels act as input to the DBN to perform final classification. The hybrid model is further optimised using cuckoo search with levy flight. The results clearly show that the proposed model SVM-LCDBN gives better performance as compared to simple hybrid model and hybrid model with traditional cuckoo search.

Asha Gowda Karegowda, Siddaganga Institute of Technology, India
Devika G., Government Engineering College, India
Geetha M., Bharat Institute of Engineering and Technology, India

The continuously growing population throughout globe demands an ample food supply, which is one of foremost challenge of smart agriculture. Timely and precise identification of weeds, insects, and diseases in plants are necessary for increased crop yield to satisfy demand for sufficient food supply. With fewer experts in this field, there is a need to develop an automated system for predicting yield, detection of weeds, insects, and diseases in plants. In addition to plants, livestock such as cattle, pigs, and chickens also contribute as major food. Hence, livestock demands precision methods for reducing the mortality rate of livestock by identifying diseases in livestock. Deep learning is one of the upcoming technologies that when combined with image processing promises smart agriculture to be a reality. Various applications of DL for smart agriculture are covered.

Preface

Deep learning, a function of artificial intelligence works similar to human brain for decision making with data processing and data patterns. Deep learning includes a subset of machine learning for processing the unsupervised data with artificial neural network functions. As the development of deep learning in engineering applications made a great impact on digital era for decision making. The wide range of applications of deep learning includes, optimizing source and loads in smart grids, decision making in cyber physical systems, robotics, industry 4.0, optimized decision in cyber security, image and video processing, computer vision for human expressions, speech recognitions, natural learning process, e-commerce for money transactions. The major advantage of deep learning is to process the Big data analytics for better analysis and self-adaptive algorithms to handle more data.

The purpose of designing this book has been to portray certain practical applications of deep learning in building smart world. With a wide-spread application of smart factory, smart cities and smart home, the need of this book becomes important. This book is intended for research scholars, under-graduate and post-graduate researchers and practicing engineers working in the area of deep learning and automation. This book covers a spectrum of applications of deep learning in building smart world, ranging from smart cities, and smart agriculture to smart home. This book would serve as a handy reference guide for researchers and engineers working intensively in the area of deep learning and automation.

This comprises of 12 quality chapters to discuss deep learning and its applications. The detailed overview of the chapters is as follows:

CHAPTER 1: DEEP LEARNING IN IOT “INTRODUCTION, APPLICATIONS, AND PERSPECTIVE IN THE BIG DATA ERA

This chapter provides a detailed learning of IoT domain, machine learning and DL techniques, applications. The IOT devices consisting of DL with intelligence background is also discussed. A detailed analysis and summarization of recent and major research attempts on DL in IoT with Big Data domain is leveraged. Current challenges and potential research for future are discussed.

CHAPTER 2: DEEP LEARNING ARCHITECTURES AND TOOLS

This chapter explains the basics of deep learning by highlighting the necessity of deep learning over non-deep learning, also covers discussion on several recently developed deep learning architectures and popular tools available in market for deep learning.

CHAPTER 3: FACIAL EMOTION RECOGNITION SYSTEM USING ENTIRE FEATURE VECTORS AND SUPERVISED CLASSIFIER

This chapter proposes the facial expression system with the entire facial feature of geometric deformable model and classifier, in order to analyses the set of prototype expressions from frontal macro facial expression.

CHAPTER 4: KIDNEY TUMOUR DIAGNOSIS SYSTEM DESIGN USING DEEP CONVOLUTION NEURAL NETWORK

This Kidney Tumour Diagnosis System uses deep learning and convolution neural networks to classify CT images. A deep learning neural network models named KidNet has been implemented. They have been trained using labelled kidney CT images. To achieve acceleration during the training phase, GPUs have been used.

CHAPTER 5: LIVER DISEASE DETECTION USING GREY WOLF OPTIMIZATION AND RANDOM FOREST CLASSIFICATION

This work assesses diverse machine learning methods in expectation of cutting edge fibrosis by joining the serum bio-markers and clinical data to build up the order models. Recipient working trademark bend investigation has also performed to assess the execution of the proposed models

CHAPTER 6: DEEP NEURAL NETWORK-BASED ANDROID MALWARE DETECTION (D-AMD)

Mobile malware causes security incidents like monetary damages, steeling of personal information, etc., when it's deep-rooted into the target. In this chapter Deep Neural Network is used for the detection. Along with the introduction of Batch Normalization, the Deep Neural Network becomes effective and also the time taken by the training process is less.

CHAPTER 7: DEEP LEARNING WITH CONCEPTUAL VIEW IN META DATA FOR CONTENT CATEGORIZATION

This chapter covers the importance of Representations &Metadata Appendage and Feature Vector Construction for the Training Deep Models Optimization.

CHAPTER 8: A FULLY AUTOMATED CROP DISEASE MONITORING AND MANAGEMENT SYSTEM BASED ON IOT

In this chapter, an IoT-based crop disease management (CDM) system is proposed, which adopts statistical methods for identifying disease, recognizing a right pesticide, and recommending a right pesticide to farmers. The proposed IoT based CDM system monitors, analyzes, extracts & identifies the issues and take proper measures to ensure good quality the agricultural crops.

CHAPTER 9: ANALYSIS OF HEART DISORDER BY USING MACHINE LEARNING METHODS AND DATA MINING TECHNIQUES

This chapter uses algorithms including Random Forest, Naïve Bayes, and Support Vector Machine according to analysis of the heart diseases. Accuracy on the prediction stage is high when the usage of greater number regarding attributes. Objective of this work is to function predictive evaluation using it data mining, data mining using of heart diseases to analyze, Machine Learning algorithms used and end which methods are effective and efficient.

CHAPTER 10: IMPLEMENTING A DEEP LEARNING APPROACH FOR THE PERFORMANCE PREDICTION IN EDUCATIONAL INFORMATION SYSTEM

This research focuses on applying the deep learning methods to educational data for classification and prediction. The educational data of students from engineering domain with cognitive and non-cognitive parameters is considered. The hybrid model with Support Vector Machine (SVM) and Deep Belief Network (DBN) is devised.

CHAPTER 11: DEEP LEARNING SOLUTIONS FOR AGRICULTURAL AND FARMING ACTIVITIES

Deep Learning is one of the upcoming technologies when combined with image processing promises smart agriculture to be a reality. In this chapter Various applications of DL for smart agriculture: detection of weeds, insects, diseases in corps, crop identification, yield prediction using genotype information and counting are enclosed, livestock, and water requirement prediction are covered.

Chapter 1
Deep Learning in IoT:
Introduction, Applications, and Perspective in the Big Data Era

Devika G.
https://orcid.org/0000-0002-2509-2867
Government Engineering College, Mandya, India

Asha Gowda Karegowda
Siddaganga Institute of Technology, India

ABSTRACT

The internet of things (IoT), big data analytics, and deep learning (DL) applications in the mechanical internet are expanding. The current digital era has various sensory devices for a wide range of fields and applications, which all generate various sensory data. DL is being applied for handling big data and has achieved great success in the IoT and other fields. The applications for data streams to discover new information, predict future insights, and make control decisions are crucial processes that make the IoT a worthy paradigm for businesses and a quality-of-life improving technology. This chapter provides a detailed account of the IoT domain, machine learning, and DL techniques and applications. The IoT that consists of DL with intelligence backgrounds is also discussed. Recent research on DL in the IoT within the big data domain is also discussed. Current challenges and potential areas for future research are discussed.

INTRODUCTION

The era of the IoT and big data is creating and acquiring an interest in diverse research disciplines, which can be seen in the amount of recently published articles, surveys, and tutorials on the topic (Chen, 2014; Gheisari, 2017; Horidi, 2017; Najafabadi, 2015). According to McKinsey's report on the global economic impact of the IoT (Najafabadi, 2015), the annual economic impact of the IoT in 2025 will be in the range of $2.7 to $6.2 trillion. Healthcare applications contribute about 41% of the share in the IoT market, followed by industry and energy with 33% and 7%, respectively. Other fields make up about

DOI: 10.4018/978-1-7998-2108-3.ch001

15% of the IoT market. Deep learning (DL) implies a tremendous and sudden growth in the market in the coming years (Horidi, 2017) since DL is suitable for most of the real-time applications. This has led to increased demand for DL products and tasks, and experts are looking forward to designing these new products. McKinsey's report also mentions that advances in DL algorithms are the main enablers of knowledge work automation. Over a decade, IoT applications have become spread across many fields (health, transportation, smart home, smart city, agriculture, education, and others) that consider key design elements as intelligent learning behaviors such as prediction, mining data, and pattern recognition. DL, which is a part of the machine learning (ML) approach, is actively utilized in IoT applications compared to other methods. DL is getting prominence by its intrinsic behavior of analytics, as it is easier to integrate big data when compared to traditional ML. Based on the learning and hierarchical representation of deep architecture, DL consists of supervised and unsupervised learning techniques for the design structure of neural networks. The functioning of DL imitates human brain neurons in the transformation of data. The IoT and big data are inter-related. IoT directly or indirectly generates big data and the IoT is an important target for big data analytics, which mainly aims to improve IoT services (Goodfello, 2016). Searching trends in Google demonstrates the popularity of DL compared to the other

Table 1. Definitions of deep learning, IoT, and big data from different organizations

Deep Learning	
Wikipedia	Deep learning is a class of machine learning algorithms which - uses a cascade of many layers of nonlinear processing; are part of the broader machine learning field of learning representations of data facilitating end-to-end optimization; learn multiple levels of representations that correspond to hierarchies of concept abstraction
Techopedia	DL is a collection of algorithms used in machine learning, used to model high-level abstractions in data through the use of model architectures, which are composed of multiple nonlinear transformations
Data science	DL conjures up images of sentient robots staging a hostile takeover. DL is just another way to describe large neural networks
Microsoft	DL is a set of algorithms in machine learning that attempt to learn in multiple levels, corresponding to different levels of abstraction.
MIT	DL is a branch of machine learning based on a set of algorithms that attempt to model high-level abstractions in data.
IoT	
IBM	IoT is the concept of connecting any devices to the Internet and to other connected devices.
IEEE	A network of items, each embedded with sensors that are connected to the Internet.
CCSA	A network, which can collect information from the physical world or control the physical world objects through various deployed devices with the capability of perception, computation, execution and communication and support communication between things by transmission; classify and process information.
ITUT	A global infrastructure for the information society, enabling advanced services by interconnecting things based on existing and evolving interoperable information and communication technologies.
Big Data	
Oracle	Big data is data of greater variety arriving in increasing volumes and with ever-higher velocity
SAS	Big data is a term that describes both structured and unstructured that inundates a business on day-to-day basics.
Techopedia	Big data refers to a process that is used when trade handling techniques cannot uncover the insights an underlying data.
EMC	All data in any form that is used for gaining insights and generating value is considered big data

four ML techniques. The definitions of these recent technologies are given by different organizations and are summarized in Table 1.

The objective of this chapter is to provide a comprehensive review of the state of art of DL practices in IoT and Big Data domain in an aim to answer the following key questions.

1. Why is DL promising for solving IoT network problems?
2. How does DL influence big data?
3. Which techniques help connect the IoT and big data?
4. Why is it advantageous to apply DL to big data?
5. What relevant DL models are applicable for an IoT network?
6. What are the most recent techniques of DL that can be applied to an IoT network?
7. What are the powerful applications of DL?
8. What are the steps are needed to tailor DL to specific network problems in the IoT?
9. What are noteworthy, promising trends for future research related to IoT, DL, and big data?

The review of the literature reveals that the questions above are partially answered. This chapter goes beyond previous works and specifically focuses on the crossover between deep learning, IoT and big data networking. Overall, the survey distinguishes itself from earlier surveys in the following ways:

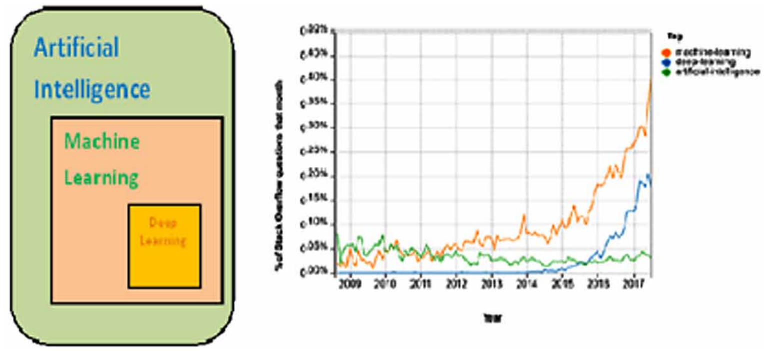
1. Focus on DL applications for mobile network analysis and management, broadly discuss DL methods (Goodfellow, 2016; Mnih, 2015), and focus on a single application domain (mobile big data analysis) with a specific platform (Mnih, 2015).
2. Discuss cutting-edge DL techniques from the perspective of the IoT networks (focusing on their applicability to big data area), while giving less attention to conventional DL models that may be out-of-date.
3. Exploit the characteristics of the IoT big data networks by conducting analysis that applies DL.
4. Provide insights into how to tailor DL to the IoT big data networking problems.

OVERVIEW OF DEEP LEARNING

DL is a collection of algorithms and is a subgroup of machine learning, which is also a subgroup of the broader artificial intelligence family, as depicted in Figure 1a (Gheisari, 2017). The DL imitates how the human brain processes tasks such as storing, analyzing, and creating patterns to use in decision making. DL algorithms try to minimize the difference between prediction and expected output among the given datasets of input and output pairs. With this process, DL tries to learn the pattern between the input and outputs, which allows the DL model to generalize the input. Deep learning is pertinent to many applications; one such example is fraud detection systems. These systems use a hierarchical level of artificial neural networks to carry out the process of deep learning. They learn from both unstructured and unlabeled data. The main differences between AI, machine learning, and DL in terms of inclusion are the number of neurons, more complex ways of connecting layers, automated feature extraction, and computing (Hinton, 2006; Lechan, 2016; Mnih, 2015;). DL is applied in various fields and Internet trends have compared the growth of DL to ML and AI from 2009 to 2017, as shown in Figure 1b.

DL Architecture

Figure 1. (a) The relationship between AI, ML and (b) DL Growth of DL, ML, and AI from 2009-2017



DL structure is similar to neural networks that consist of a large number of parameters and layers. DL design involves three major layers: input, hidden, and output. The hidden layer decides the depth of the network (Hinton, 2006). The fundamental types of DL network architecture include unsupervised pre-trained networks, supervised convolution neural networks (CNN), supervised recurrent neural networks, and reinforced recursive neural networks. The unsupervised pre-training works by initializing the discriminative neural net with an application using an unsupervised criterion such as a deep belief network or a deep autoencoder. CNN is a standard neural network that is extended across space using shared weights. A recurrent neural network is extended across time by having edges to feed the next layers rather than feeding into the same layer of the same step. A recursive neural network is like a hierarchical network where the input sequence will be processed in a tree structure. A few deep learning methods are based on these network architectures and are discussed in the following sections.

Unsupervised Pre-trained Networks

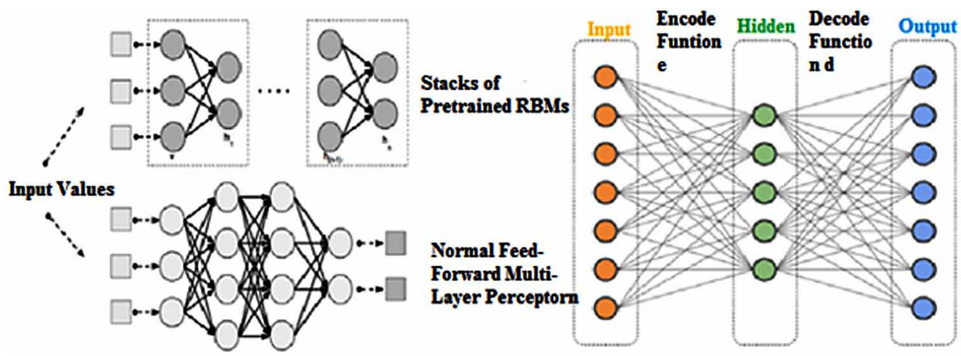
Deep belief networks (DBN) apply ML principles and is a generative graphical model, or an alternative class of deep neural network with multiple layers of hidden units by means of connections between layers but not between units within each layer (Hinton, 2006). Initially, the model obtains training and stacks several layers of Restricted Boltzmann Machines (RBM) in a greedy manner. Once this stack of RBMs is trained, it can be used to initialize a multi-layer neural network for classification. A typical DBN is represented in Figure 2a.

Multilayer perceptron (MLP) is designed based on artificial neural networks consisting of at least three layers of operations (Collobert, 2004). Each layer's components are heavily connected and require suitable weights to configure. An MLP containing more than one hidden layer is regarded as a deep learning structure.

Deep auto-encoders (DAE) improves the greedy DBN. It consists of two symmetric DBNs and four or five shallow layers representing the encoding half of the net and second set of four or five layers that make up the decoding half, as in Figure 2b (Kingma,2014).

Supervised Convolution Neural Networks

Figure 2. (a) Deep belief network and (b) deep autoencoders



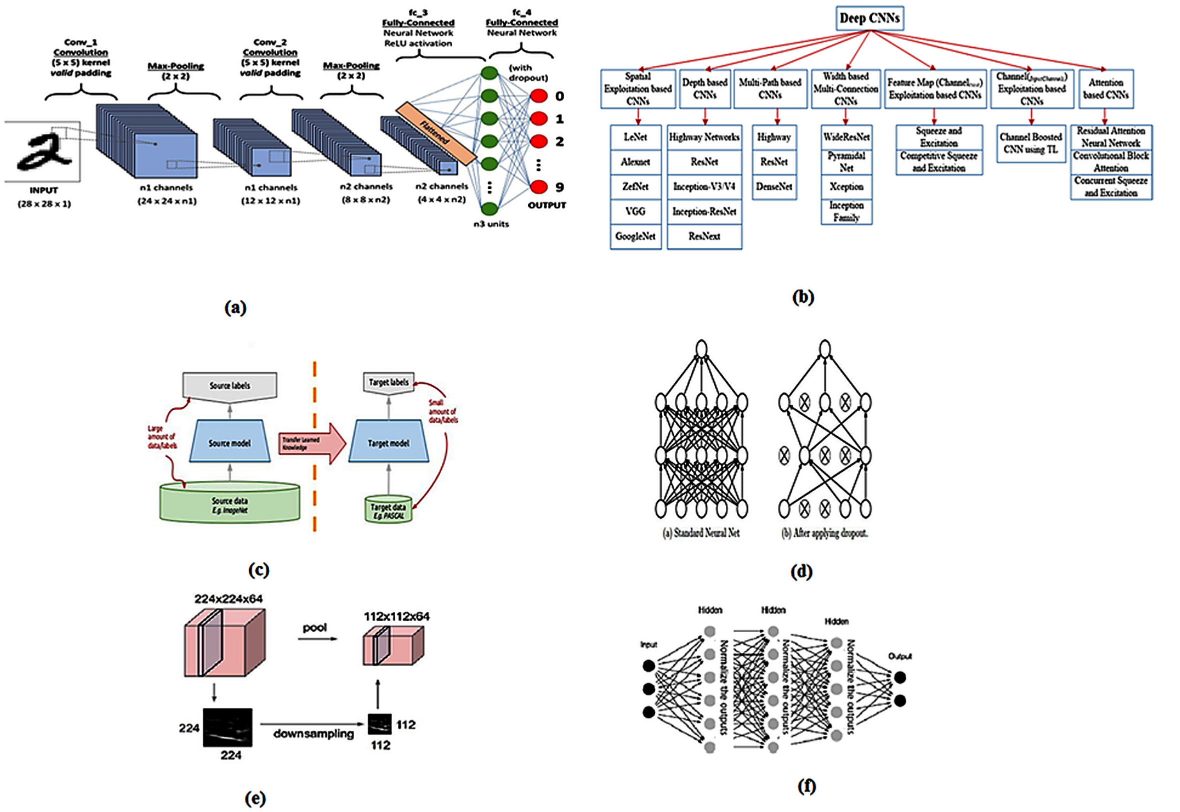
The typical CNN architecture includes an exchange layer that connects one or more fully connected layers at the end as the next of the convolution and pooling layers. For a specific application, these fully connected layers are replaced with a global average pooling layer. In some cases, in order to optimize performance, the CNN will also include batch normalization and dropout in addition to learning stages as regulatory layers. The performance of CNN is based on the arrangement of different components in its new architecture. The CNN sequencing classifies handwritten digits, which are shown in Figure 3a (http://towardsdatascience.com). The CNN can be categorized based on features such as spatial exploitation, depth, multi-path, width, feature map exploitation, channel boosting, and attention, which is depicted in Figure 3b (Khan, 2016).

The transfer learning method (TL) is used when pre-trained models require processing. In such a case, the model will be reused as a starting point for operation. For example, if CNN trained on one dataset, then according to training, it would remove or retain on the last layers. The training models are applied to recognize different higher-level features. The training time will be reduced in transfer learning. It is a helpful tool when there is enough data or when training takes too many resources. The transfer of learning is shown in Figure 3c (Ruder, 2017).

In the dropout address (DA), the problem of CNN is that neurons are dropped or ignored randomly during the training process. Larger networks are slower to use, making it difficult to deal with overfitting by combining predictions of many different large deep networks at the training time. In the dropout address, the randomly dropped units from the neural network prevent other units from adapting for training. This method normally improves the performance of neural networks on supervised learning. The improved dropout method is presented in Figure 3d.

The max-pooling (MP) method involves the discretization and is a sample-based process. Its main objective is to down-sample input representation by minimizing its dimensionality and features contained in sub-regions (Sutskever, 2017). This will provide an abstraction form of representation by reducing the computational cost by reducing the number of parameters needed to learn and provide basic conversions regarding its original representations. This method is suitable when applied using a maximum filter without overlapping the original subset regions. A max pool example is shown in Figure 3e.

Figure 3. (a) Typical CNN example for character identification; (b) Classification of CNN; (c) Transfer learning; (d) method and dropouts; (e) Max pooling example; and (f) batch normalization
Source: https://medium.com



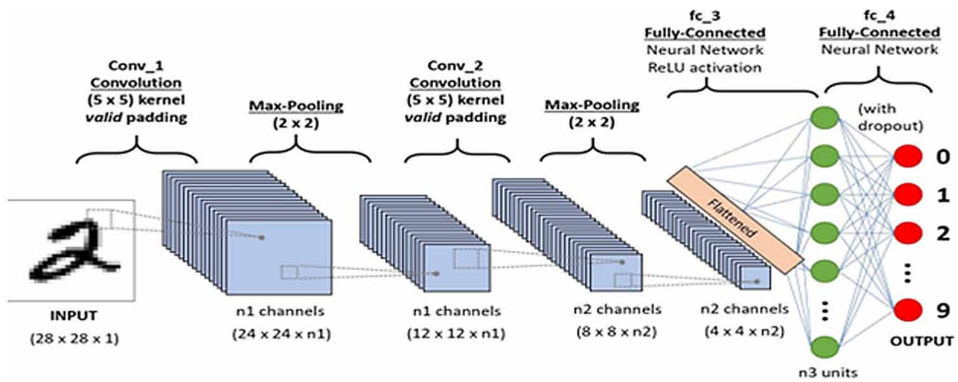
Batch normalization (BN) is a process used to routinely standardize (making the mean zero and the standard deviation one) the activations and gradients proliferating through a network. In other words, the batch normalization layer can be included in the network model to standardize the raw input variables or the outputs of a hidden layer with the objective of accelerating the training of deep learning neural networks to get better results. The sample batch normalization process is given in Figure 3f.

Supervised Recurrent Neural Networks

Recurrent neural networks (RNN) work on the principle of artificial neural networks to recognize patterns for a given sequence of data that are in form of text, genome format, handwritten, word, times series data, radiating from sensors data, or data from stock markets and government agencies. RNN will take the sequence and the time of the previously reported output, which are supported by a temporal dimension as an input for the next round. RNN supports internal memory to keep information from the previous input (Chung, 2014; Ienco, 2017). A simple RNN is shown in Figure 4a.

Long short-term memory (LSTM) is an extension of RNNs. Its architecture is based on the concept of gates for each of its units with value communication range values between zero and one. In addition to a store loop, each neuron has a multiplicative forget gate, read gate, and write gate. Gates are

Figure 4. (a) A simple recurrent neural network; (b) recursive neural network; and (c) long and short-term memory



introduced to support memory operations (Gers, 2016). The LSTM working process is shown in Figure 4b. The LSTM is further improved by reducing parameters in gated recurrent units (Cho, 2014). Its performance is very similar to LSTM and applicable to applications such as speech, signal, and music modeling. Continuous-time RNN uses differential equations to model neuron units, works in both directions from left-to-right as well as from right-to-left, and it is suitable and effective in evolutionary robotics applications (Graves, 2005).

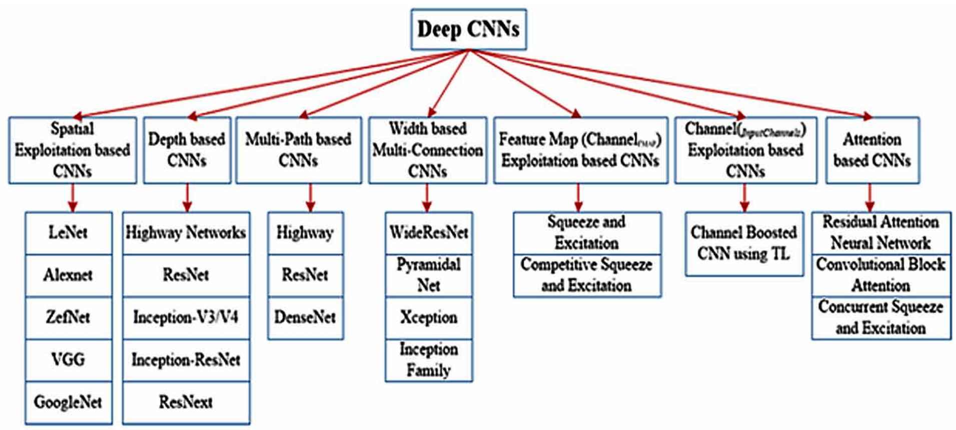
Supervised Recursive Neural Networks

Recursive neural networks are deep neural networks based on technique and are trained by automatic differentiation (Griewank, 2009) in reverse mode. It is created by applying the same weights to the structure input. The structure of a recursive neural network is shown in Figure 4c, and consists of a linear chained tree structure and is applicable for natural language processing in distributed structures. As shown in Figure 4c, the nodes are combined to form patterns using a weight matrix to share across the whole network and non-linearity. C1 and C2 are an n-dimensional representation of nodes. They are suitable for parsing scenes and for syntactic of natural languages and sentence applications. Recursive cascade correlation is an extension of the recursive neural network that is based on constructive NN and is commonly used in chemistry and acyclic graph-based applications (Sperduit, 1997). A variant of the recursive neural network is the recursive neural tensor network, which is developed using functions of tensor-based composition for every node on the network (Socher, 2013).

Generative adversarial network (GAN) trains follow the adversarial process. GAN uses two models: the generative model and the discriminative model. Both of these models are neural network-based. The GAN provides random numbers to generate a fake image. This fake image, along with the actual ground truth dataset is fed to the discriminator. The discriminator predicts whether the input image is fake or real (Goodfellow, 2014). The working of GAN is presented in Figure 5a. GAN can be used to improve the resolution of the image, predict the next frame in the video, and convert text to an image.

Deep reinforcement learning (DRL) is an amalgamation of reinforcement learning and DL. Reinforcement adopts trial and error to learn the best action by awarding the correct action and penalizing the wrong action. DRL has a sequential reinforcement learning process, in which DL regulates the action taken at every stage to get the best results (Schulman, 2017). The DRL computations are as shown in Figure 5b. The Soft Actor Critic algorithm by Google to train insect-like robots (using DRL) to learn day to day tasks with fewer tries, which keeps the robot from taking incorrect actions that would otherwise result in destruction. DRL in healthcare was used to envisage drug doses for sepsis patients and in deciding the optimal dose cycles for chemotherapy.

Figure 5. (a) A generative adversarial network and (b) a deep reinforcement learning network



The comparison of DL architectures considers parameters, learning scenarios, advantages, disadvantages, applicability of problem, applications, the type of structure, and kind of data model being accepting as input is briefed in Table 2. The hetero input type supports 2D, 3D, time series, voice, and image-based data.

Strengths and Weaknesses of Deep Learning

A strength of DL is that it is applicable in multiple domains and has the best-in-class performance. Further, the outcome can be automatically tuned according to the feature required. Existing data types are applicable for different applications as with possible usage of the same neural network approaches. Additionally, it allows for parallel computing, which delivers a higher performance of more data. Moreover, it allows for automated robustness to natural variations, reduces the need for feature engineering, and its architecture is applicable and changeable for new problems without difficulty.

The DL architecture is adaptable and applicable to new problems with ease. However, DL faces problems in training as it requires expense computations that require more corpora of data, not having a strong theoretical base and are not easy to comprehend what is learned.

The Need of Deep Learning

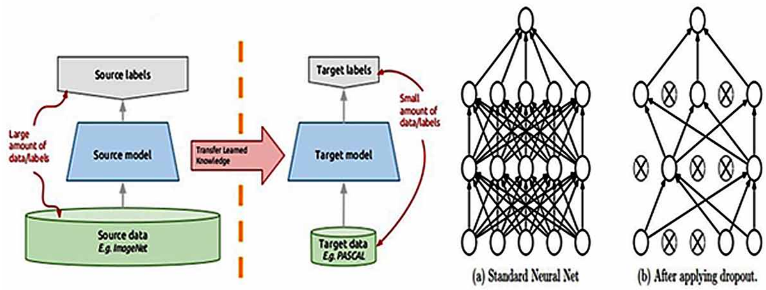
Deep learning performs better on a larger volume of data and adopts automated feature engineering, which is not the case in machine learning or with other algorithms, as represented in Figure 6.

Table 2. Summary of different deep learning architectures

DL Techniques	Learning Scenarios	Merits	Demerits	Suitable Problems	Latent Applications	Type	Input Data
MLP	Supervised, unsupervised, reinforcement ANN,	Naive structure and straight forward to build High complexity,	modest performance and slow convergence	Modeling data with simple correlations	Modeling multi-attribute mobile data, auxiliary or component of other deep architectures	Linear	Hetero
RBM	Unsupervised DBN [160], Convolutional DBN [161]	Can generate virtual samples	Difficult to train well	Extracting robust representations	Learning representations from unlabeled mobile Data, model weight initialization, network flow Prediction	linear	Hetero
AE	Unsupervised	Powerful and effective unsupervised learning	Expensive to pre-train with big data	Learning sparse and compact representations	model weight initialization, mobile data dimension reduction, mobile anomaly detection	linear	Hetero
CNN	Supervised, unsupervised, reinforcement	Weight sharing, affine invariance	High computational Cost, challenging to find optimal hyper-parameters requires deep structures for complex tasks	Spatial data modeling	Spatial mobile data analysis	Linear non-linear	2-D
RNN	Supervised, unsupervised, reinforcement	Expertise in capturing temporal dependencies	High model Complexity, gradient vanishing and exploding problems	Sequential data modeling	Individual traffic flow, Analysis, network-wide (spatio-) temporal data modeling	Linear non-linear	Time-series
GAN	unsupervised, reinforcement	Can produce lifelike artifacts from a target distribution	Training process is unstable	Data generation	Virtual mobile data generation, assisting supervised learning tasks in network data analysis	Linear non-linear	Hetero
DRL	unsupervised, reinforcement	Provides extra reward on agents considered	Temporal differences between consecutive predictions are more	State space-based	Resource management normally in clusters form	Linear non-linear	Hetero

OVERVIEW OF INTERNET OF THINGS

Figure 6. Deep Learning Performance



Every aspect of life is influenced and pervaded by the IoTs. IoT is all about data, devices, and connectivity (Chen, 2014). The IoT model can be defined as an interconnection of massive heterogeneous devices and systems that apply different communication patterns: thing-to-thing, human-to-human, or human-to-thing (Evans, 2011; Tan, 2017). The recent improvements in communication technologies transcended traditional sensing of the surrounding environment by enabling it with modernizations to improve the quality of life by collecting, quantifying, and analyzing surrounding environments (Chen, 2014). It is among the emerging field with an estimated billion devices by the end of 2020. In early 2000, Kevin Ashton laid the groundwork for designing the IoT at MIT's AutoID lab. Ashton envisioned that this design would improve its business by linking RFID information to the Internet (Abomhara, 2015). At that time, identifying each object and making objects communicate with each other was a major undertaking. In this vision connectivity, the IoT supports all objects under one umbrella including the computer (for controlling) and Internet infrastructure. After that, many other obstacles were resolved including reducing the size and cost of the wireless radio, the introduction of IPV6, and devices enabled with Wi-Fi and cellular connectivity. On the other hand, IoT knowledge is crucial for enhancing day-to-day, real-time application of specific areas like home, healthcare, transportation, and education. IoT devices are normally placed and used in unattended places and are normally interconnected through a wireless medium. So, the devices cannot support complex computational structures because of their limited storage and power resources (Tan, 2017). The IoT converts a physical object into a smart object by using communication technologies, Internet protocols and applications, sensors, and ubiquitous and pervasive computing (Gokhale, 2018). The realization of the IoTs system is crucial in all application fields because it enables the connection of billions of smart devices (Evans, 2011). The sensing devices consist of physical objects and are integrated into the communication network and are supported by computational equipment with smart devices. The architecture normally consists of three layers: application, network, and perception (Gokhale, 2018).

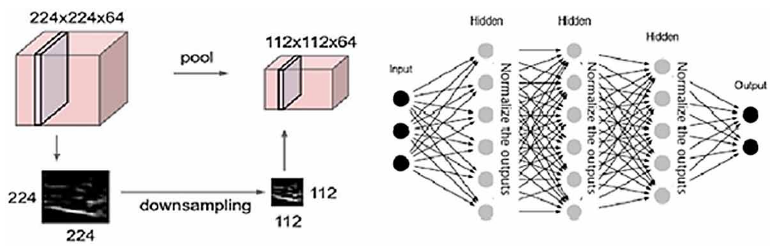
The IoT data is different from general data in terms of its characteristics (Mnih, 2015). These characteristics include large-scale data streaming, heterogeneity, time and space correlations, and high-noise data. Since the data generated by the IoTs is heterogeneous and huge, it demands an analysis that uses DL.

IoT Architecture

Improved and simplified architecture is shown in Figure 7. Each layer is described in the following sections.

Sensing/Perception Layer

Figure 7. IoT architecture



The sensing/perception layer involves physical objects, normally sensors, and actuators for temperature, humidity, motion, and acceleration with the main task of gathering data. The main function of the physical objects is to sense, collect, and process information. The plug-and-play is applicable at this level to configure heterogeneous sensors (Jin, 2016; Ray, 2016; Yan, 2014). The sensor devices are constrained by limited battery capacity and computational capability; it is a key step in achieving a context-aware IoT system (Abomhara, 2015). Because the growth of an object is connected to the IoT, the huge amount of data is collected, which demands big data analysis. Furthermore, the analysis of sensed huge data demands using DL.

Network Layer

The data sensed will be in the form of analog, which needs to be aggregated and converted into digital streams for further processing by downstream layers. The network layer process can be divided into connectivity and middleware tasks.

Connectivity

The main task of the IoT platform is to connect heterogeneous sensors to cooperate and subsequently provide smart services (Ray, 2016). The sensors deployed are normally resource-constrained, so they are required to work in a low-power, more lossed, and noisy communication environment (Gartner,2015). The following challenges are faced during connecting IoT devices: providing IPs to billions of devices connected to the internet, developing low-power communication for transmitting data generated by sensors, implementing effective routing protocols, and concentrating on integrity and security of transferred data.

Middleware

Middleware offers software level applications that intend to effectively represent the complexities of a system or hardware allowing designers to concentrate on problems on the system such as communication or computation issues without any interventions or interruptions (Zanella, 2014). In view of computation, it offers a level between the application and the system software (Agrawal, 2013). The tasks consist of facilitating cooperation between heterogeneous IoT objects, providing scalability for devices that interact in the IoT environment, providing device discovery and context awareness, and providing device security and privacy (Agrawal, 2013).

Data Processing Layer

This layer includes tasks of big data, cleansing, streaming, and storage of data. The collected or captured by IoT are valuable hence deep learning can play an analytical role in building intelligent IoT systems to deliver smart services (Zanella, 2014). The data produced by devices are analyzed in real-time to obtain insights by researchers by applying methods of integrating big data and IoT (Agrawal, 2013). The traditional methods, machine learning or deep learning will effectively obtain required data from big data and convert big data into useful information without or with minimum intervention of humans (IW, 2015; Zhang, 2014).

Application Layer

The application layer consists of various applications as demanded by the user and is supported by the IoT. A few of the most commonly used applications supported by the IoT include smart home, smart healthcare, smart transportation, wearables, smart building, smart cities, IoT agriculture, and others. These applications are briefed below.

- **Smart home**: Applying IoT components to home devices is easy and it is easy to observe and control them remotely (Smartpharma, 2015). Additionally, sensing can be provided internally and externally to manage objects during construction.
- **Smart healthcare**: IoT is gradually becoming a key player in healthcare (Tan, 2010). They are applied in health care systems to observe and collect patients' health status and send their health information to patients and to the doctor.
- **Smart transportation**: The transportation systems are becoming attainable with the IoTs. Smart transportation is used to control traffic in cities and for smart parking. Proper installation of devices will improve road safety and reduce the delivery time (Zhang, 2014).
- **Wearables**: These devices are incorporated to collect data about users, which will be pre-processed to extract essential parameters. Normally, such devices include fitness, health, and entertainment requirements (Zhang, 2014).
- **Smart governance**: The IoT will facilitate smart governance for integrating data of different governmental sectors by providing authorities with large scale information through wide sensor networks to replace conventional monitoring systems with knowledge-based systems.

- **Smart cities**: As a solution to problems faced by people in cities smart surveillance, automated transportation, smart energy management, water distribution, security, and environmental monitoring are applied (Informationweek,2015).
- **Smart grid**: The IoT grids are constructed between suppliers and consumers of electricity to handle and improve efficiency, safety, and provide real-time monitoring (Aggrwal, 2018). These are applied to increase reliability and decrease the cost of transmission.
- **Smart agriculture**: The sensors are deployed in agricultural fields to sense and collect information such as humidity, temperature, weather conditions, and moisture, which enables real-time monitoring. The collected data are analyzed in terms of irrigation, quality of water, soil condition, and others to get a higher yield. Livestock management is also supported smartly through the IoT.

Business Layer

In this layer, the IoT services are delivered to the user and data is captured from and analyzed at lower levels through integration. Confined data are effectively utilized to improve social and economic growth.

Merits and Demerits of the IoT

The prominent merits of the IoT are monitoring, as well as efficient utilization of resources, reduces human effort, increases data collection, minimizes time to gather data, and improves security. However, the problems that arise in IoT are data privacy, physical activities, and grid collapse.

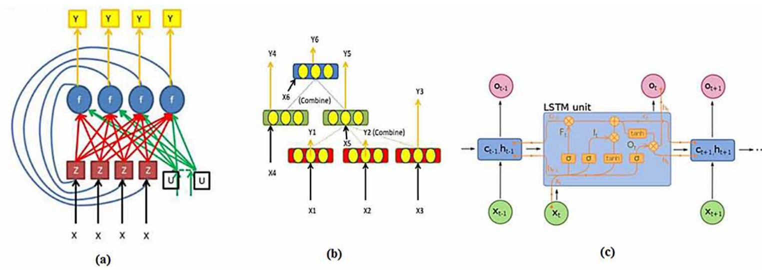
OVERVIEW OF BIG DATA

While big data offer great potential for revolutionizing all aspects of society, garnering valuable knowledge from big data is not a conventional task. The large and rapidly growing body of information hidden in the unprecedented volume of non-traditional data requires both the development of advanced technologies and interdisciplinary teams working in close collaboration. Big data describes voluminous amounts of structure, semi-structured, and unstructured data that has the potential to be mined for information. Big data include both heterogeneous data (text, images, videos) and frequently changing data (social network, geospatial, biometric, time series, etc.). Big data need to be preprocessed using data cleaning and formatting, and it needs to be stored and processed quickly. According to a Forbes 2019 documentary, big data is defined as “Extremely large data sets that may be analyzed computationally to reveal patterns, trends, and associations”. Today, machine learning techniques, together with advances in available computational power, have come to play a vital role in big data analytics and knowledge discovery (Brookes, 2015; Gubbi, 2013). They are employed widely to leverage the predictive power of big data in fields like search engines, medicine, and astronomy. As an extremely active subfield of machine learning, deep learning is considered along with big data as the “big deals and the bases for American innovation and economic revolution” (Gartner, 2015; Weinstein, 2016). Big data involves the task of capturing data, storing it, analyzing stored data, searching, sharing, transferring, visualizing, updating, querying, and securing data. A few of the vital characteristic aspects of big data are algorithm usage, processing clarity, tendency to collect all data, frequent processing of data, and possibly using new forms of data.

Big Data Architecture

As big data is designed to handle complex traditional database systems with the process of ingestion, processing, and analysis, it involves the following workload types: at rest batch processing, in motion real-time processing, interactive exploration, and predictive analytics and machine learning.

Figure 8. A Typical Big Data Architecture
Source: Microsoft



Big data architecture includes the following components:

1. Data source: the initial process begins by considering one or more data sources
2. Data storage: the data will be stored as a distributed file to hold a large volume of data
3. Batch processing: the stored data are usually processed in batches
4. Real-time message ingestion: the real-time message dropouts are processed using buffers
5. Stream processing: the messages are processed by filtering, aggregation, or through analysis
6. Analytical data store: as queried through analytical tools, the required structured solutions will be prepared
7. Analysis and reporting: big data will provide solutions to data through analysis and reporting
8. Orchestration: the data will be provided in an automated manner by providing repeated data processing, encapsulated data flows, and loading processed data

Need for Big Data

As the world is becoming filled with sensing devices, there is a need to discover patterns and correlations of all sensed data in real-time to positively impact businesses. Big data analytics is used to discover hidden patterns, marketing trends, and consumer preferences for the benefit of organizations' decision making.

Need for Deep Learning for Big Data

The concept of deep learning is to get information from a large amount of data to automatically identify patterns and extract features from complex, unsupervised data without involving humans, which makes it an important tool for big data analysis (Graves, 2005).

Advantages and Disadvantages of Big Data

Many companies and researchers have identified substantial benefits of big data, which include:

- Better decision making
- Increased productivity
- Reduce analytics cost
- Improved customer services
- Easy fraud detection
- Increased revenue
- Increased agility
- Capacity for innovation
- Faster speed to market

Big data has reported significant challenges, which include:

- Lack of big data skillset
- Need to address data quality issues
- Need for cultural change
- Cybersecurity risks
- Big data analytics efforts in complying with government regulations
- A rapid change in technologies
- Hardware needs to house store data
- Difficulty integrating legacy systems

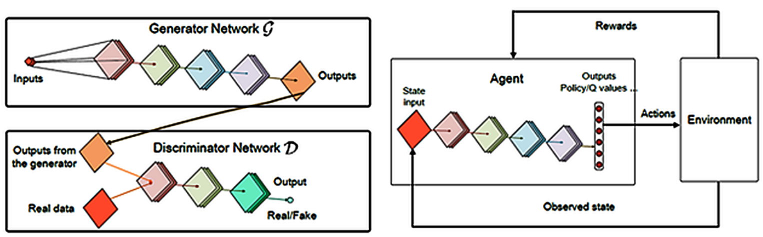
STATUS OF DEEP LEARNING

"Deep Learning is an amazing tool that is helping numerous groups create exciting AI applications," Andrew Ng says, Chief Scientist at Baidu and chairman/co-founder of Coursera. It is helping us build self-driving cars, accurate speech recognition, computers that can understand images, and much more. Many researchers consider this to be only the beginning of DL; new DL tools are emerging and are being explored for new applications and products. Few noticeable employable situations that demand DL are the ones where human experts are not reachable, where solutions are required for particular cases (e.g., personalization, fingerprint, where humans are unable to expertise), where solutions are not constant and change with time, and where humans cannot keep hold of the large data size when it comes to analysis. Currently, most of the applications fit with DL; hence, this approach is often termed as a universal learning approach. Infrastructure issues still exist that need solutions. Advances in DL are increasing as companies seek to use advanced computational intelligence techniques to find useful information hidden across large scale data, and DL outperforms humans in some tasks (e.g., classifying images, and automated car driving). Very few people in the world know how to use DL tools for different tools, so more research and employability are awaiting in the future. In the future, DL is getting its importance in speech-related applications, capsule networks, incorporate usage to know customers' choice, deep reinforcement learning, and decision-making processes.

Integrated Applications of Deep Learning With IoT

"In one go, not all issues, making whole nation smart are conceivable by IoT", was pointed out by Teradata's CTO. The guide for innovation may be progressively fretted over the personal satisfaction of the general population. The guide for innovation determination should be in accordance with the torment purposes of natives living in selected regions. DL and the IoT infrastructure and devices will improve performance, reduce the model size, inference time, network compression, approximate computing, and increase hardware device performance. The fog, edge, and cloud computing are based on DL, and the IoT is ideal in terms of security, and time constraints on devices because it allows for easy aggregation of several sources of IoT for data fog computing. Edge nodes with efficient DL algorithms can be localized for many complex analytical tasks that are currently performed in the cloud. DL will discover service protocols, model, aggregate and task distribution, dynamically join and leave the system, energy management, deploy CNN models on fog nodes for machine health prognosis, and find free nodes to delegate analytic tasks. The applications can be grouped into ten prominent categories as depicted in Figure 9.

Figure 9. Applications of DL for IoT



Smart Health

With this application, patients can be under continuous monitoring by doctors without having a physical presence. The current status data regarding a patient's health conditions can be obtained remotely. The measurements of various parametric features can be obtained remotely such as heart rate, blood sugar level, blood oxygen level, body temperature, and wound status. Yin (2015) characterized irregularities in tissue morphology such as tumors to determine if fibroids are present. Aminian, (2000) used an active learning approach to determine which genes and micro RNA features are present. DL approaches for predicting splicing code and understanding given expressions change by genetic variants (Sun, 2016). A DL based graph determines whether molecular structural features, physical properties, and activities are present by correlating the molecule compounds and target information (Junbo, 2015). Liu (2016) used a DL model to predict DNA methylation states from DNA sequences and incomplete profiles. Galloway (2016) and Li (2015) used a stacked AE method to detect microaneurysms in images as an instance of diabetic retinopathy strategy. Redy (2016), Verma (2013), and Guo (2016) applied DL methods to automate and extract relevant features with classification procedures Lu (2015) and Deutsch (2016) used an RBM for MRI data to discriminate deficit hyperactivity disorders. Others have automated and merged the extraction of relevant features using classification procedures (Mao, 2016; Thirukovalluru, 2016; Wang, 2016). DL models are applied to encode the parameters of deformable models to facilitate the segmentation of the left ventricle from short-axis cardiac MRI (Gan, 2016; Oh, 2016; Zhang, 2015). DL models are employed for computer-aided detection, segmentation and shape analysis in (Janssens, 2016; Zhang, 2016; Zhang, 2015). Babu (2016) and Ding (2017) used hybrid multi-layers to recognize the liver and spleen along with polynomial activation functions. Wang (2015) and Wulsin (2010) applied clustering to mammographic image data by employing DL architecture with standard CNN. Yan, (2017) used DL methodologies for recognizing the affective state of EEG. Ong (2015) used a combination of DL and varying features for the purpose of learning hierarchical representation with video input to recognize human activities. Futoma (2015) used DL models for pervasive sensings, such as food intake recognition, sign recognitions. DL modeled functions are devised to predict and classify clinical events (Phan, 2015). Felbo (2016) and Kendra (2015) applied DL for public health infrastructure for predicting demographic information, lifestyle diseases, and infectious disease epidemics.

Table 3. A brief summarization of the applications of smart health

Use case	DL Architecture	Purpose	Learning Type/Dataset	Precision %	Ref.
Smart Health	CNN	Food calorie estimation	10000 high-resolution food images	91%	Pouladzadeh, 2016
	CNN	Recognized human actions such as falling to floor and baby crawling; a three-stream convolution neural network was proposed	UCF101 and HMDB-51 datasets about elderly and children care	89%	Huang, 2016
	Hybrid	Hand gestures detection and classification; online system without noticeable lag; a recurrent 3D CNN was proposed	multimodal dynamic hand gesture dataset captured with depth, color, and stereo-IR sensors	99%	Molchanov, 2016
	DBN	EEG data classification and anomaly detection; applied convolutional DBN to learn features from high-dimensional and multichannel EEG signals.	EEG data classification and anomaly detection; applied convolutional DBN to learn features from high-dimensional and multichannel EEG signals.	90%	Ren, 2014
	SAE	Brain disease diagnosis; SAE for latent feature extraction on a large set of hand-crafted features.	Different types of neuroimaging modality data from ADNI dataset		Suk, 2015
	RNN	Multilabel classification of diagnoses; included drop out, target replication and auxiliary output	Anonymized clinical time series extracted from the EHR system at Children's Hospital LA	85%	Lipton, 2015
	CNN	Anomaly detection; Compared architectures for detecting interstitial disease and lymph nodes	Publicly available thoracoabdominal lymph node datasets and interstitial lung disease dataset	93%	Miotto, 2016
	CNN	Infectious disease epidemics	Geo-tagged images	86%	Felbo, 2016
	DBN	Lifestyle disease	Mobile phone metadata		Kendra, 2015
	DAE	Predicting demographic information	Social media		Ong, 2015
	RNN	Data mining	Blood/lab tests		Miotto, 2016
	CNN	Human behavior monitoring	Big medical dataset		Che, 2015
	DBN	Prediction of disease	Electronic records	87%	Ha, 2015
	DBN	Anomaly detection	EEG, ECG, implantable devices	88%	Wang, 2015
	CNN	Human activity recognition	Video, wearable device	90%	Huuang, 2015
	DBN	Human activity recognition	Video, wearable device	99%	Choi, 2013
	CNN	Hand gestures recognition	Depth camera	94%	Poggi, 2016
	DBN	Obstacle detection, Sign language recognition	RGB-D camera		Huang, 2015
	CNN	Food intake, energy expenditure	Wearable device, RGB image, mobile device		Pouladzadeh, 2016
	DAE	3D brain reconstruction	MRI/fMRI		Shan, 2016
	CNN	Neural cells classification	Funds images		Nie, 2016
	DBN	Brain tissues classification	PET scans		Suk, 2014
	DNN	Alzheimer/ MCI diagnosis	Funds images, PET scans		Kuang, 2015
	CDBN	Tissue classification	MRI/CT images		Li, 2015
	CNN	Organ selection	Endoscopy images		Cheng, 2016
	DAE	Cell clustering	Microscopy		Poulos, 2016
	DNN	Hemorrhage detection	Fundus images		Kondo, 2016
	CNN	Tumor detection	X-ray images		Zou, 2016
	DAE	Cancer diagnosis	Generic expression		Fakor, 204
	DBN	Gene selection DBN	Micro RNA		Ibrahim, 2014
	DNN	Geen variant DNNs	Microarray data		Quang, 2014
	DNN	Drug design	Molecule compounds		Sundar,2015
	DBN	Compound protein interaction	Protein & molecule structures		Tian, 2015
	DNN	RNA binding protein	Genes/RNA/DNA		Zhang, 2016
	DNN	DNA methylation	Genes/RNA/DNA sequence		Mansoor, 2016

Table 4. A brief summary of the applications of smart home and transportation

Use Case	DL Architecture	Purpose	Learning Type/Dataset	Precision %	Ref.
Smart Home	AE	To track measurement of areas, to learn deep and compact features for visual tracking	10 sequence of constructed videos	86%	Hoseini, 2013
	RNN	For sequential classification of multiple objects from an image input	when tested on Google street view house images		Bedingfield, 2018
	CNN	To discriminate object patches from their surrounding background	bike trip data		Shiraz, 2017
Smart transportation	DBN	Traffic flow prediction	Real-time traffic data in California (PeMS dataset) and entrance-exit station data of a highway	95%	Enache, 2015
	CNN	Forecasts the inflow and outflow of crowds in each region of a city;	GPS trajectories data, bike trip data, weather conditions and events for cloud	87%	Huang, 2014
	CNN	Parking lot occupancy detection	CNRPark and PKLot datasets contain images of parking lots and segmented parking spaces AlexNet	88%	Zhang, 2017
	CNN	To determine 12 of the most recent studies in the pavement distress detection	Parking lot occupancy detection; applied AlexNet and VGGNet,		Amato, 2016
	CNN	to detect a parking lot occupancy based on smart cameras	performed using a dataset created by the authors (CNRPark-EXT) and a (PKLot) mAlexNet	93%	Gopalakrishna, 2018
	DBN	to assist digital map creation by automatic street elements detection such as traffic lights, roundabouts, etc.	The input data for the system are obtained only by users GPS data	89%	Krizhevsky, 2012
	RNN	To determine traffic accident hotspots and their automated detection and classification show	road dataset obtained from the Swiss Road Authority (FEDRO)	30%	Organero, 2018
	RNN(LSTM)	To perform traffic congestion predictions.	600 million taxi trip data, contains GPS location and timestamp of pick-up and drop off event in NYC	96%	Ryder, 2017

Smart Homes

Smart homes have electronics that communicate with each other to monitor and manage devices in an energy-efficient manner. As part of it, devices can be operated without human interference remotely. Hoseini (2013) has presented an AE based on stacked denoising to learn deep compact to form real-time applications. Bedingfield (2018) used CNN based method to learn discriminative feature representations for visual tracking to discriminate object patches from their surrounding background. Bedingfield also adopted an LSTM to predict the next activities in data from large-scale American time data, rather than data from a smart home environment. Gill (2009) focused on the prediction of the movement of employees between rooms in an office setting using reinforcement feed forward neural networks to train for fixed prefixed length of input Shiraz (2017) and Enache (2015) used a DL method for sequential classification of multiple objects in Google street view house images using the Deep Recurrent Attention Model (DRAM).

Smart Transportation

Traffic can be easily monitored and an optimized route can be suggested which allows for easy transportation and parking reservations. Further, streetlights can be deployed and utilized economically, accidents can be prevented, and autonomous driving will be supported in cities. Enache (2015) and Huang (2014) performed efficient unsupervised features that learn the area of transportation. Mnih (2015) used CNN DL principles to forecast the inflow and outflow of crowds in each region of a city by applying

Table 5. A brief summary of the applications of communication

Use Case	DL Architecture	Purpose	Learning Type/Dataset	Precision %	Ref.
Communication	RNN(LSTM)	Uses mobile traffic forecasting to aid cloud radio access network optimization	Cloud RAN optimization	80%	Chen, 2018
	CNN	To predict traffic forecasting	Represents Spatio-temporal dependency via graphs and first work employing Graph neural networks for traffic forecasting		Wang, 2018
	Hybrid	To predict traffic forecasting	Mobile traffic forecasting-AE+LSTM, network type cloud	95%	Jing, 2017
	RNN(LSTM)	To fuse data gathered from multiple sensors and perform activity recognition	cloud		Liu, 2016
	CNN	To aid deaf people's awareness of emergencies	Edge		Mittal, 2016
	CNN	To investigate for garbage detection	Edge+cloud		Yao, 2017
	CNN	Car tracking, heterogeneous human activity recognition, and user identification	Edge		Lin, 2017
	RNN	To conduct a Generative model	Human activity chains generation for edge network		Ouyang, 2016
	CNN	Mobile user trajectory prediction	Online framework for data stream processing through prediction for cloud network		Wang, 2015
	RBN	Indoor localization	Works with calibrated phase information of CSI		Wang, 2017
	CNN	Uses more robust angle of arrival for estimation	CSI dataset of Indoor localization		Guan, 2017
	MLP	To determine light communications	Combining deep learning with genetic algorithms; visible light communication based on localization		Bernas, 2015
	MLP	To determine indoor locations	Resilient backpropagation		Lee, 2017
	CNN	Real-time query analysis	Query processing		Wang, 2018
	AE	Distributed WSN anomaly detection	Employs distributed anomaly detection techniques to offload computations from the cloud		Luo, 2018
	DNB(MLP)	To improve the energy efficiency in the aggregation process	Data aggregation		Khorasani, 2017
	AE	Use deep learning to enable fast data transfer and reduce energy consumption	WSN –SEA	87%	Heydari, 2017
	DBN	To recognize range-free WSN node localization	MLP, ELM		Banihashemian, 2018
	CNN	To select the nearest leakage location	Detection, localization		Kang, 2018
	AE	Rate-distortion balanced data compression for WSNs	WSN		Abu, 2016
	RNN	Caching and interference alignment	Deep Q		He, 2017
	RNN	Cellular network random access optimization	Deep Q		Chen, 2017
	RNN	To recognize Cellular network random access optimization	Deep gradient		Chen, 2018
	RNN	Unmanned aerial vehicles control	DQN		Liu, 2018
	RNN	Adaptive video bitrate	A3C		Mao, 2017
	RNN	Mobile actor node control	Deep Q		Oda, 2017
	RNN	Load balancing analysis based DBN	DBN		Kim, 2017
	RNN	Path planning for aerial vehicle networks			Challita, 2018
	RNN	Wireless online power control	Deep Q		Luo, 2018
	RNN	Dynamic orchestration of networking, caching, and computing	DQN		He, 2017
	RNN	Anti-jamming communications in dynamic and unknown environment	DQN		Xin Liu, 2018
	AE	Malware classification & Denial of service, probing, remote to user & user to root	SAE		Azar, 2017
	AE	Flooding, injection and impersonation attacks for attacks detection	MLP		Erza, 2016
	RNN	Radio transmitter settings selection in satellite communications			Paulo, 2018
	AE	To identify anomaly detection for avoiding sudden signal-to-noise ratio changes in the communication channels			Feng, 2016
	AE	Flooding attacks detection in wireless mesh networks	MLP		Khan, 2016
	AE	IoT distributed attacks recognition			Liu, 2018
	AE	Attacks detection in delayed feedback networks in smart grids using reservoir computing Supervised MLP	MLP		Shea, 2017
	CNN	Sparse linear inverse problem in MIMO CNN			Mark, 2017
	AE	Optimization of representations for encoding processes			
	AE	MIMO nonlinear equalization	MLP		Takuya, 2018
	AE	Super-resolution channel and direction of arrival estimation	MLP		Sreeraj, 2018
	CNN	Automatic modulation classification	LSTM		Shea, 2016
	CNN	Modulation recognition CNN	LSTM with ResNet		J. O. Shea, 2016
	CNN	Modulation recognition Radio transformer network			Ye, 2018
	CNN	Modulation classification CNN			Liang, 2016
	AE	Modulation classification in a software-defined radio testbed	MLP		Wei, 2018
	AE	Learning to communicate over an impaired channel	AE+ radio transformer network		Dörner, 2018

the ResNet model to spatial correlation. Zhang (2017) and Amato (2016) applied smart parking using CNN functions and used Alex Net for correlation. Valipour (2016) applied DL DCNN for pavement distresses detection for road anomalies detection. Gopalakrishnan (2018) used CNN functions to detect parking lot occupancy. Ouyang (2016) and Wang (2015) created a digital map to determine automatic street elements detection, like lights. Wang (2017) applied an RNN to analyze road data obtained from the road authority to demonstrate accident occurrence and the actual accident spot.

Communication

DL uses algorithms and models to facilitate wireless network analysis and is used for proper resource management. Sensors fusion with dynamic human signal recognition is being employed. This application area is still emerging; therefore, it is still pretty inefficient and immature. Data being communicated can be analyzed, DL driven fog, mobile, and cloud computing can be realized, and heterogeneous network architectures can be developed. The application of filters for fraudulent news and news aggregation is possible. In a DL based network-level mobile data analysis focuses on deep learning applications built on mobile big data collected within the network (Chen, 2018; Jing, 2017; Liu, 2016; Mittal, 2016; Wang, 2018). In addition, network predictions, traffic classification, and call details can record (CDR) mining. DL based mobile data analytics on edge devices are considered (Bernas, 2015; Guan, 2017; Lee, 2017). DL based user mobility analysis sheds light on the benefits of employing deep neural networks to understand the movement patterns of mobile users, either at the group or individual levels (Banihashemian, 2018; Heydari, 2017). DL based user localization reviews literature employs deep neural networks to localize users in indoor or outdoor environments (Abu, 2016; Kang, 2018). The developed environment is based on signals received from mobile devices or wireless channels. DL based wireless sensor network are implemented which discusses important work on deep learning applications in wireless sensor networks are discussed considering different perspectives such as centralized, decentralized, WSN data analysis, WSN localization and others (Challita, 2018; Chen, 2017; Chen, 2018; He, 2017; Kim, 2017; Liu, 2018; Luo, 2018; Mao, 2017; Oda, 2017). DL based network control investigations for deep reinforcement learning and deep imitation learning on network optimization, routing, scheduling, resource allocation, and radio control are also discussed (Azar, 2017; Erza, 2016; Feng, 2016; Khan, 2016; Liu, 2018; Paulo, 2018). Prior literature discusses DL based signal processing scrutinizes physical layer aspects that benefit from deep learning and reviews relevant work on signal processing (Liang, 2016; Shea, 2016; J O Shea, 2016; Ye, 2018; Wei, 2018). Dörner (2018) discusses DL based mobile network applications and discusses interesting DL applications in mobile networking.

Robotics

DL is an active research area because it is applicable for learning complex, high-dimensional, and novel dynamics, control policies in dynamic environments, knowing the dynamic kinematic structures, dynamic object recognition, and high-level task planning such as accepting human commands. A DL architecture approximates specific rectifiers to model a highly coupled dynamic radio-controlled helicopter (Punjani, 2015). It is a challenging analytical derivation and is a difficult system identification problem. The human expert obtained helicopter training data through aerobic maneuvers. It outperformed other methods for obtaining helicopter dynamics by about 60%. Others have modeled the time between drivers head movement and occurrences of maneuver varying from one vehicle speed to another using DL (Neverova, 2014). The predictions occur every 0.8 seconds based on the preceding five seconds of data with about 3.5 seconds of maneuvers. The model achieved an accuracy of 90.5%. A DL based function is subjected to identify the object and pose recognition for garments hanging from a single point picked by robotic gripper (Yu, 2013). The different clothes hanging from various grasp points are considered as training data and a test set of six objects, different from those used in training and obtained cent percent result. Deep CNN is used to recognize five known objects resting on a flat surface and categorizes their orientation into discretized categories (Mariolis, 2015). The DL based architecture is used to study and focus on recognition and poses estimation to plan with limited positioning, a parallel gripper at the object's center and aligned with an estimated angle. The result of grasping exceeded 90%. DL based can also recognize 48 common kitchen objects and classify of 88 cooking videos (Sun, 2016). The training of robots in mind exhibition of significant variation in background and scenery were implemented. Power and precision grasps and sub-classes of each were classified. The results were 79% achieved for object recognition accuracy and 91% grasp classification. Lu (2017) used one layer AE-based DL functions to

Table 6. A brief summary of the applications of robotics

Use Case	DL Architecture	Purpose	Learning Type/Dataset	Precision %	Ref.
Robotics	DNN	to model the highly coupled dynamics of a radio-controlled helicopter	IoT	60%	Punjani, 2015
		modeled the time between driver's head movement and occurrences of maneuver	Parking lot occupancy detection; applied AlexNet and VGGNet,	90.5%	Neverova, 2014
		to study and focus on recognition and pose estimation to grasp planning		90%	Yu, 2013
		to identify the object and pose recognition for garments hanging from a single point	GPS trajectories data, bike trip data,	100%	Yang, 2015
		To recognize and classify kitchen objects		91%	Mariolis, 2015
	AE	to classify induction motor faults			Sun, 2016
	AE	For fault diagnosis of rotary machinery components			Lu, 2017
	AE	frequency spectra are able to demonstrate how their constitutive components are distributed with discrete frequencies and may be more discriminative over the health conditions of rotating machinery.			Jia, 2016
	RBM	predicted RMS and the total time of the bearing's life			Deutsch, 2016
	CNN	used as input for spindle bearing fault diagnosis	Avg. pooling, a multiscale layer		Ding, 2017
	CNN	for machinery fault diagnosis	pre-process vibration data		Z. Chen, 2015
	RNN	for fault diagnosis and prognostics of aero engine	LSTM		Yuan, 2016
	RNN	Machine health monitoring system in the tool wear test	LSTM		Zhao, 2016
	AE	Channel estimation and signal detection in OFDM systems	MLP		Liao, 2018
	CNN	Channel decoding	LSTM		Huang, 2018
	CNN	NNs for channel decoding MLP, CNN and RNN			Huang, 2018
	AE	Over-the-air communications system			Roberto, 2017
	AE	Rayleigh fading channel prediction	MLP		Joan, 2018
	AE	Light-emitting diode (LED) visible light downlink error correction	MLP		Ahmed, 2018
	AE	Coordinated beamforming for highly mobile millimeter-wave systems	MLP		Ali, 2018
	CNN	Millimeter-wave positioning			Hitaj, 2017
	GAN	Channel agnostic end-to-end learning-based Communication system Conditional			Hao, 2018

classify induction motor faults. A stacked multiple AE is used by Jia (2016) to denoising operation for fault diagnosis of rotary machinery components. The SAE model is used for rotating machinery diagnosis is developed to consider the frequency spectra (Huang, 2018). Roberto (2017) applied an RBM based Dl function to predict remaining life. Joan (2018) diagnosed the spindle bearing fault using a deep convolution network. Hitaj (2017) adopted a CNN for gearbox fault diagnosis. Hao (2018) adopted a fault diagnosis and prognostics of aero-engine by applying RNN models. Ali (2018) used an empirical evaluation by applying the scheme of LSTM for machine health monitoring system in the tool wear test.

Image and Text Recognition

Automatic image caption generation, image recognition, handwritten generation and recognition, automatic text recognition and translation, and image colorization are being employed and implemented using DL. CNN is applied for face verification, recognition and clustering to directly optimize the embedding to prevent the intermediate bottleneck layer as in conventional CNNs (Kipf, 2017; Qi, 2017). CNN is applied for recognizing handwritten Bengali characters (Zeng, 2008), Bangla characters (Imtiaz, 2011), Persian characters (Kurian, 2012), and Devanagari characters (Bishop, 1992). Mori (1992) applied CNN to bridge a gap between understanding mathematical structure and computational implementation.

Smart Agriculture

Table 7. A brief summary of the applications of image recognition

Use Case	DL Architecture	Purpose	Learning Type/Dataset	Precision %	Ref.
Image and text recognition	CNN	Face verification: To face verification as deeply hidden identity features (DeepID).	Labeled Faces in the Wild (LFW)	100%	Qi, 2017
	CNN	Face verification: To optimize embedding with face verification, recognition, and clustering	Labeled Faces in the Wild (LFW) and YouTube Faces DB	98%	Kipf, 2017
		Character recognition: To recognize Persian handwritten characters	Persian handwritten dataset images with LeNet	97%	Zeng, 2008
	CNN	Character recognition: To recognize characters in a given document	Yi character documents	99.6%	Imtiaz, 2011
	CNN	Character recognition: To recognize Devanagari characters	92 thousand handwritten images with LeNet	98%	Kurian, 2012
	CNN	Character recognition: To recognize Bengali handwritten characters	Bengali handwritten dataset images		Bishop, 1992
	CNN	Character recognition: To recognize Bangla handwritten characters	Bangla lekha-Isolated dataset and CMARTER dataset with ResNet	96%	Mori, 1992

The agriculture field is getting smarter because of DL because it can easily monitor agricultural fields for plant protection, disease identification, fruit count, land classification, plant phenotyping, leaf area index, soil quality monitoring, water resource monitoring, and livestock. Kuwata (2015) uses AE methodology for leaf disease detection, and Rebetez (2016) used it for land cover classification. CNN is applied for plant disease detection (Pan, 2008; Pereira, 2018), crop classification (Pound, 2018), plant classification (Mnih, 2015), soil analysis (Song, 2018), crop yield prediction (Kuwata, 2015), and fruit

Table 8. A brief summary of the applications of smart agriculture

Use Case	DL Architecture	Purpose	Learning Type/Dataset	Precision %	Ref.
Smart Agriculture	AE	Leaf disease detection	Thirteen different types of plant diseases, plus healthy leaves CaffeNet	96%	Kuwata, 2015
	CNN	Plant disease detection, Identify 14 crop species and 26 diseases	Authors-created database containing 4483 images with Alexnet	99%	Sehal, 2017
	AE	Land cover classification, Identify 13 different land-cover classes in KSC and nine different classes in Pavia	A mixed vegetation site over Kennedy pace Center (KSC), FL, USA, and an urban site over the city of Pavia, Italy	98%	Rebetez, 2016
	CNN	Land classification, identify 21 land-use classes containing a variety of spatial patterns	UC Merced land-use data set	94%	Pan, 2010
	CNN	Crop type classification, nineteen multi-temporal scenes acquired by Landsat-8 and Sentinel-1A RS satellites from a test site in Ukraine	Classification of crops wheat, maize, soybean sunflower and sugar beet	94%	Pereira, 2018
	CNN	Plant classification, to recognize 44 different plant species	MalayaKew (MK) Leaf Data set which consists of 44 classes, collected at the Royal Botanic Gardens, Kew, England	98%	Pound, 2018
	CNN	Soil analysis, Segmentation of root and soil Identify roots from soils	Soil images coming from X-ray Tomography	56%	Song, 2018
	CNN	Crop yield prediction, Crop yield estimation Estimate maize yield at the county level in the USA	Maize yields from 2001 to 2010 in Illinois, USA, downloaded from Climate Research Unit (CRU), plus MODIS Enhanced Vegetation Index	63%	Kuwata, 2015
	CNN	Fruit count, Fruit counting to predict the number of tomatoes in images	24 000 synthetic images produced by the authors	91%	Moonfar, 2017
	CNN	Weed detection, Obstacle detection Identify ISO barrel-shaped obstacles in a row crops and grass mowing	A total of 437 images from authors' experiments and recordings	91%	Shao, 2015
	LSTM	Character recognition, Identification of weeds Classify 91 weed seed type	Data set of 3980 images containing 91 types of weed seeds	91%	Milioto, 2017
	DBN	Soil monitoring, Prediction of soil moisture content Predict the soil moisture content over an irrigated cornfield	Soil data collected from an irrigated cornfield (an area of 22 km2) in the Zhangye oasis, Northwest China	67%	Douarre, 2016
	CNN	Cattle race classification Practical and accurate cattle identification from five different races	A total of 1300 images created by the authors	94%	269

count (Moonfar, 2017). CNN is presented for weed detection (Shao, 2015; Milioto, 2017) and cattle race monitoring ((kasfi, 2016). Soil monitoring is achieved by applying LSTM (Douarre, 2016).

Table 9. A brief summary of the applications of environmental monitoring

Use Case	DL Architecture	Purpose	Learning Type/Dataset	Ref.
Environmental Conditions Monitoring	DNN	Air quality prediction; Hand-crafted spatial transformation component to address spatial correlation; fusion network to fuse different factors.	Hourly air pollutants, meteorological data and weather forecast data in China	Yi, 2018
	hybrid	Air quality, PM2.5 concentration prediction; pretraining with auto-encoder; use RNN to model time series.	PM2.5 prediction system called VENUS developed by the National Institute for Environmental Studies in Japan	Bun, 2016
	SAE	Air quality prediction; an SAE model for air quality feature selection, interpolation and prediction.	Hourly meteorological data (e.g. temperature, humidity, and etc.) and air quality data (e.g. PM2.5, PM10, and etc.) in Beijing	Qi, 2018
	SAE	Landslide recognition; a stacked denoising autoencoder and discrete wavelet transformation were used to extract features.	An optical remote sensing image set with 1200 samples from Google Earth	Liu, 2016
	CNN	Disaster damage detection; a multiple kernel-learning framework combines CNN features and 3D point cloud features.	Two groups of datasets based on multi-view oblique images from manned aircraft and UAVs	Vetrivel, 2018
	CNN	Fire detection in indoor and outdoor environments; a prioritization mechanism to change the priority of camera nodes; dynamic channel selection algorithm.	A dataset of 68457 images collected from different fire datasets of both images and Videos	Khan, 2018
	CNN	Crisis-related tweets classification with pre-trained word embeddings	Labeled twitter datasets: CrisisNLP, CrisisLex, and AIDR	Dat, 2018

Environmental Conditions Monitoring

Environmental conditions can be monitored with a variety of parametric components with barometers, humidity, temperature, pollution conditions. It is possible to monitor air, and water pollution levels in different areas. Accurate predictions of air quality are becoming a challenging task, as it is affected by multiple factors such as meteorology, social event, location, and traffic. A deep neural network with fusion component-based approaches can be applied to predict the air quality index (Yi, 2018). It is a widely used metric to indicate polluted air. Spatial transformations are used to address spatial correlation and a distributed fusion network is used to merge all affecting components (Bun, 2016). AE based hybrid functions are used to predict PM2.5 concentration for 52 cities in Japan (Qi, 2018). It uses historical PM2.5 concentration levels for hours. RNN is used to fine-tune time-series designs. A DL SAE based approach can unify air quality feature selection, prediction, and interpolation (Liu, 2016). A sparse layer on top of the input layer was introduced to perform feature selection and Spatio-temporal information. AE is applied to recognize landslide from optical remote sensing images (Nie, 2015). Wavelet transformation is applied to extract hidden and distinct features SoftMax classifier for prediction. CNN is designed to extract 3D point clouds from multiple kernel learning to detect building damages caused by a disaster like an earthquake (Khan, 2018). Oblique aerial images are taken by managing aircraft. CNN has also been applied for early fire detection systems in indoor as well as outdoor environments (Vetrivel, 2018). A pre-trained AlexNet is used for real-time fire detection based on images captured from surveillance

Table 10. A brief summary of the applications of security and surveillance

Use Case	DL Architecture	Purpose	Learning Type/Dataset	Precision %	Ref.
Security and Surveillance Systems	Supervised DL (SVM)	Adopt a new approach, deep learning, cybersecurity to enable the detection of attacks on the social internet of things.	Fog	80%	280
	CNN	To secure low resources IoT devices such as smart meters and sensors against any malicious behaviors.			281
	CNN	To review the advances on issues of security and privacy in IoT, including security and privacy requirements, attack types, and the relevant solutions, and discuss challenges and future trends in this area		90%	282
	CNN	Vehicle license plate recognition; Using two networks, one for recognition of digits and the other for letters; was the first time that a random CNN was used for vehicle identification.	The dataset was obtained from Brazilian license plate images captured in a real-world setting	93%	283
	hybrid	Vehicle license plate recognition; combined merits of both CNN and RNN to handle issues of poor quality, complex background, blur and noise	20105 images for experimentation in this work, 18270 images from UCSD dataset, 1835 images from MIMOS dataset	85%	284
	CNN	Vehicle detection and recognition; collected and annotated a new dataset BoxCars; showed that additional information easily obtainable in real-time from static surveillance cameras can boost the CNN verification performance greatly	Boxcars dataset with the 3D bounding boxes and contains 21250 vehicles (63750 images) of 27 different makes	90%	285
	CNN	Vehicle re-identification; collected the dataset VeRi-776; proposed deep learning-based, progressive vehicle re-identification approach for urban surveillance	VeRi-776 dataset contains about 40000 images of 619 vehicles captured by 20 surveillance cameras.		286

cameras. CNN based functions are adopted to capture the salient n-gram information using convolution and pooling operations (Dat, 2018).

Logistics and Supply Chain Management

Products can be easily tracked from production to the store, reducing cost and time significantly. The quality assurance, protection of products, and client information can be protected. Isolation cannot be the property of organizations as they depend on capabilities and resources embedded in suppliers, customers, and collaborators. Since 1980, importance is being given for supply chain management and how it can benefit from collaborative relationships within their boundaries. The supply chain has become an increasing concern for organizations of different sizes and across a wide range of industries. The reactive approach of responding to external pressure from different sections of organizations is considerable. This approach is complemented by the development and introduction of sustainable products. In the current competitive environment, professionals are struggling to handle large structured and unstructured data. The survey is conducted to produce, capture, organize, and analyze insights of industries. IoT is applied in the context of a network of physical objects that are digitally connected to sense, monitor, and interact within a company and between supply chain enabling agility, visibility, tracking, and sharing for timely coordination of logistical and supply chain processes (K. Perumal, 2017). The web database shows that from 1980-2018 around 308 papers were published based on topics related to the IoT and supply chain.

Security and Surveillance Systems

Security can provide networks, machines, and other components. The deployed cameras are capable of capturing videos within streets, which enables real-time visual object recognition to identify suspects and avoid serious situations. DL based intrusion detection techniques for IoT context are designed (Wong, 2019). The thousands of zero-day attacks appear because of additional protocols for the IoT and most small variants of previously known cyber-attacks. Such a situation indicated, even advanced mechanisms such as traditional DL systems face the difficulty of detecting small mutations of attacks over time. In (Amato,2017) employed a Nash equilibrium based lightweight anomaly detection technique is presented. The anomaly-based concept is of game theory. The method mainly predicted the equilibrium state that activates anomaly detection to detect new attack signatures. The results showed that data generated obtained excellent detection rates, low false positive alarm, and allow energy consumption.

INTEGRATED APPLICATIONS OF DEEP LEARNING WITH BIG DATA

Deep learning extracts meaningful information from raw data using a hierarchical multi-agent learning approach. At a high level, more abstract and complex representations are learned based on less abstract concepts and representation in the lower level of the learning hierarchy. Deep learning can be applied to learn from labeled data if it is available in sufficiently large amounts for extracting meaningful representations and patterns from big data.

The hierarchical data abstractions are learned from unsupervised data with DL. More conventional discriminative models can be trained with the aid of relatively fewer supervised data points. DL algorithms are shown to perform better at extracting non-local and global relationships and patterns in the data compared to relatively with shallow learning models (Salton, 1998). The characteristics of learned abstract representations by DL, relatively simple linear models can work effectively, with knowledge obtained from more complex and more abstract data representations. Increased automation of data representation extraction from unsupervised data, enables its broad application to different data types like image, text, audio, etc. Relational and semantic knowledge can be obtained at higher levels of abstractions. The major applications of deep learning to big data are described in the following sections.

The information retrieval is a key task associated with big data analytics. Efficient storage and retrieval of information is a growing problem in big data particularly since very large-scale quantities of data segregated and assigned with different domains such as fraud, defense, security, and marketing systems. The massive storage of data for information storage and retrieval is challenged by massive volumes of data and different data representations that are associated. The massive collection of data requires semantic indexing for storing data bit strings. It presents data in a more efficient manner and makes it useful as a source for knowledge comprehension and discovery. DL is used to generate high-level abstract data representations. It plays an important part in data indexing. The high-level abstract data representations need to be meaningful relational and semantic rules are followed to confer the input. This process can also be applied using DL. The representation in the form of vectors for data instances provides faster information retrieval and searching. This representation learning is much applicable for understanding complex semantic and relational information which are in the form of raw bits. The vector representation of complex high-level data abstractions for semantic indexing is feasible. The general idea of indexing is based on extracted data of DL. It is useful with DL has can be extended to other forms also. The goal of document representations is t63+925519o creates reduced unique aspects of documents. The document retrieval schemes such as TF-IDF (Robertson, 1994) and BM25 (Hinton, 2011) can be used. The document-representation schemes considered words, which are of different dimensions and independent. DL techniques are used to extract meaningful data representations from input data, to obtain different semantic features such as high-dimensional textual data that leads to dimensional reductions of documents data representations. A DL generative model is applied to learn codes for documents that are described (Salakhutdinov, 2009). The lowest layer of the DL network represents the word count vector of the document. It results in high-dimensional data with a high layer representing learned binary code of the document. The binary codes of documents of semantically similar lay relatively closer in the Hamming space. The binary code of the documents can then be used for information retrieval. The documents which are similar to binary codes are retrieved allowing relatively quicker searches by using a fast-bit counting algorithm to compute the Hamming distance of two binary codes. The work proves binary codes are faster than semantic-based analysis. DL models of generative type can also be used to produce shorter binary codes by forcing the highest layer in the learning hierarchy to use a relatively small number of variables. Memory addresses are simply considered for the shortest binary codes. The technique semantic hashing is being applied to describe each document considering one word of memory that is small Hamming ball around memory. Using strategy information retrieval on a very large document set with time independent of document set size can be performed. The semantic hashing is quite attractive for information retrieval since finding all of the similar information queried can be retrieved by finding how all of the memory addresses differ from one to another. The memory hashing is faster than locality-sensitive hashing; it is one of the fastest methods among available algorithms. In addition, the documents binary codes to algorithms TF-IDF instead of providing the entire document higher accuracy level can be achieved. The resulting knowledge yields fast interferences, as it is one of the major goals of big data Analytics. The production of binary code for a new document requires just a few vector-matrix computations including feed-forward pass through encoder component of DL network architecture. By applying supervised data in DL, better representation and abstractions can be learned. A study on the parameters of the DL model showed that it learned based on supervised and unsupervised data (Mikolov, 2013). There is no need to completely label a large collection of data. The data model to learn data representations to produce a good reconstruction of input and to provide good predictions of document class labels. The work shows for learning compact representations, DL models are better than shallow learning models. The compact representations are efficient as they require fewer computations if applied for indexing with less storage capacity. Google has developed a word2vex tool is a technique for the automated extraction of semantic representations from Big Data. Large-scale text input is supported in these tools and produces word vectors representation of word as it is used as a feature in many natural language processing and DL. It learns high-quality word vectors from the huge dataset with multiple hundreds of bits and distinct words in the vocabulary (Dean, 2012). They focus on artificial neural networks to learn distributed representations of words. The models are implemented on top of large-scale distributed representation of words framework. Mikolov (2013) discusses a disbelief framework. The word vectors are trained on a massive amount of data show a subtle semantic relationship between words such as a city and the country where it belongs. Word vector with such semantic relationships could be used to improve many existing natural language processing applications such as machine translation, information retrieval, and question response systems. In (Bordes, 2012), word2vec can be applied for natural language translation.

DL models are used to learn complex nonlinear representations between word occurrences which allow capturing high-level semantic aspects of documents. Capturing complex representations requires massive amounts of data for input and produces labeled data from massive input which is a difficult task. DL leverage unlabeled documents to have access to a much larger amount of input data using a smaller amount of supervised data to improve the representation of data to make more related to specific learning and inference tasks. The extracted data representations are effective for retrieving documents by search engines. DL can be used on other kinds of data to extract semantic representations from input amounts allowing semantic indexing of data. The relatively recent emergence of DL needs to be done using strategies from the hierarchical method for semantic indexing of big data. The remaining open question is used to define similar data representations for indexing purposes.

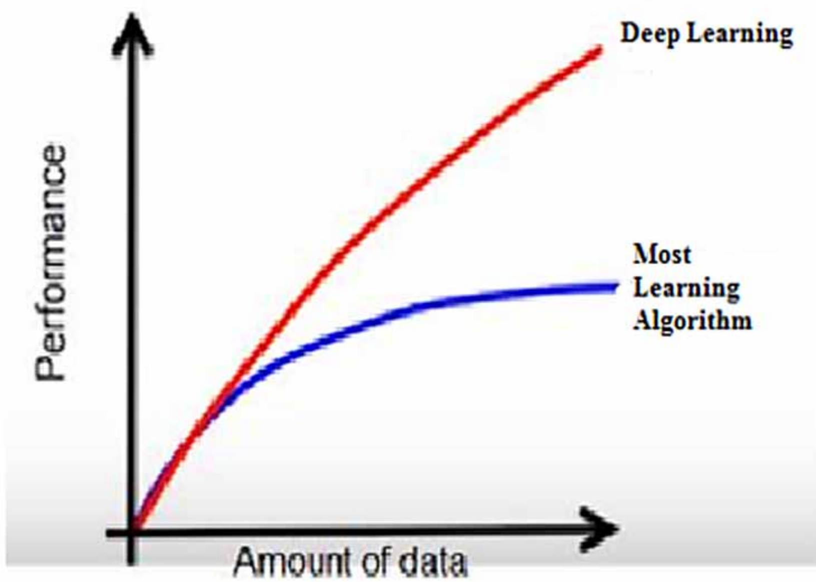
DL algorithms can be extracted complicated nonlinear features from the raw data and use linear models to perform discriminative tasks using the extracted features as input. The advantage of DL is it does more computation which is relatively simple and effective for the linear analytical model. The problem of developing efficient linear models for Big Data Analytics has been extensively investigated. The extraction of non-linear features from massive amounts of input data allows data analysts to benefit from knowledge available by applying simpler linear models for further analysis. The major benefit of using DL and big data to accomplish complicated tasks related to Artificial Intelligence such as image comprehension, object recognition in images, etc, constructing simple models. The Big Data aided with DL made relatively easier with discriminative tasks, which is a primary purpose of data analysis or it can perform better to conduct tagging on the data for searching. DL based speech recognition technology explored by Microsoft Research Audio Video Indexing applied to enable searching of audio and video files with speech (Daniel, 2013). The closed captions and keywords can increase accessibility and discovery of audio and video using MAVIS to convert digital audio and video signals. The explosion of online users in recent years is very rapid to increase in the size of digital image collections. The input data is being collected from social networks, global positioning satellites, image sharing, medical and security systems. The image file-based document content is explored by Google to consider image content itself. Artificial Intelligence provides improved image search. The textual representations of images are not always available in massive image collection repositories. The massive data collections are being collected from organizing and collecting experts through actual conduction, browsers, searching, and retrieval. The data collections are approached to consider automated process tagging images and extracting semantic information from images. DL presents a new primary model towards constructing complicated representations for image and video data as relatively high levels of abstractions used for image annotations and tagging useful for image indexing and retrieval. DL with the context of big data helps in the discriminative task of semantic tagging of data. In the semantic indexing, the focus is on DL abstract representations for data indexing purposes. The data considered as features for performing the task of data tagging is used for data indexing as a primary idea which makes possible to tag massive amount of data by applying simple linear modeling methods on complicated features to extract DL models. DL and CDD outperform existing approaches for image object recognition (Turang, 2010). IamgeNet is considered for image object recognition and searching for images. The same dataset is considered along with the DL model for large-scale software infrastructure to training Artificial Neural Networks (Tao, 2010). In (Chen, 2014), restricted Boltzmann Machines are presented for learning and extraction features from images. AE model is used in (Gheisari, 2017). The works mentioned only extracts low-level features from images. DL can be used to build high-level features for image detection. Google and Stanford formulated DNN to learn very high-level features such as face detection or cat detection from unlabeled data in (Chen, 2014). It is applied on large scale investigation on the feasibility of building high-level features with DL using unsupervised data considering ImageNet dataset. DL algorithms are used to extract features from a given dataset to successfully perform the discriminative task of Google's work involving questions to check whether there is the possibility of building a face feature detector(He, 2017). On the contrary, earlier works require a very large amount of labeled data in order to detect face features as data collections pose a challenging problem. In (FU, 2015), DL model based on recursive neural networks is used for predicting a tree structure for images, to extract multiple modalities from segmentation and annotation of complex image scenes is done. The results outperformed other methods based on conditional random fields or a combination of other methods. DL and RNN combination is applied to construct a meaningful designed based search (Hermato, 2015). DL models are used for the extraction of action scene recognition video data tagging. It is achieved using an independent variant analysis method to learn Spatio-temporal features of video data representations, which is an important research direction for generalized domains (Shalev, 2012). Overall, the DL models are adopted in extracting useful features for performing discriminative tasks on image and video data for the extraction of representations from all kinds of data. The results of discriminative data tagging and information retrieval is being applied by search engines. The high-level complex data representation can be obtained by DL applications for a computationally feasible simpler linear model for big data.

DEEP LEARNING, IOT AND BIG DATA COLLABORATION

The application of AI using DL is a key deciding technique to derive data streams from its insights. The IoT and big data along with DL are useful for real-time decision time decision-making applications. Earlier research of DL mostly ended in improving its architecture and algorithms, but as data sized grown, the data was deployed on cloud platforms, before to IoT era. But With the introduction of IoT, the problem was viewed in the form of resource-constrained devices for real-time analytics. The DL model will be used in IoT and resource-constrained devices as it involves most part with processors, battery energy and memory (Lioa, 2016). In conclusion, the technologies considered as IoT will sense big data as stimuli, and deep learning helps understand the future of a smart connected world. The digital technology is being used from years for processing or for different operations in a wide range of applications; even then there is a need to realize the power of sensors on a larger scale. The key aim of this chapter is to spotlight on the effects of integration of deep learning for IoT with big data usage. The series of these technologies will benefit in improved processing power.

Today's application technologies can have an amalgamation of robotics, computer and deep learning enable for various reasons including easy connectivity, improvements in sensor equipment of monitoring, measurement, and tracking of information from all regions. As IoT is generating voluminous data, it is preferred to store data in big data either on cloud or edge and consequently and the cost of storage is also decreasing. As a result, it helps to automate the specific process via IoT within an organization by storing and feeding an enormous amount of data via big data as input to deep learning. From this design of technologies, it is becoming easier for managers, executives, and data scientists to access and analyze. The data transfer will be done on a reliable connection and a number of connection standards will exist to cater application needs for sending data packets far long distances, for larger fields, or for maximum network infrastructure. The amalgamation of DL, IoT, and big data is represented in Figure 10. The integration of DL, big data and IoT will not happen over a day to take advantage of these technologies by bringing them through a holistic approach, will-built controlling power and flow of data. The main aim is to bind data collected from sensors and other contextual information to discover patterns and inter-relate into real-time to positively impact applications. Currently available big data technology requires improvement to store, manage, and extract data from a stream of sensed data in consumption and quickly deriving actionable events.

Figure 10. Combination of Deep learning, IoT, and Big Data



In the future, smart data analytical algorithms should be able to handle big data (i.e. IoT requires algorithms that can analyze data that comes from a variety of data sources). Incorporating deep learning will make easy computations using algorithms, as well as give a high accuracy rate with enough data in less time. It requires computing framework of processing data that involves prominent computing means such as fog computing, edge computing, and cloud computing and mobile backend as a service.

Advantages of a New Model

- Faster real-time decision making from collected data, such as identification of crop patterns through drone from a remote place, suspicious behavior of persons in any region.
- Support for digitalization in large scale

- Easier sensing and communication means. Example: toasters and coffee makers talking with each other.
- Helps to store, manage and derive unstructured data, deduce insights by deriving values for business
- Increases revenue, improve productivity, efficiency & lower cost to businesses.

FUTURE PERSPECTIVE OF DEEP LEARNING, IOT, AND BIG DATA

This section reviews several challenges, future directions for applications that are important from a DL point of view to implement and develop. The deep learning will be applicable for all domain capturing a greater variety of data and can excavate bigger data from any form of data whether it is structure or unstructured. The market for it is expected to grow by 41 percent from 2019-25 reaching $20 billion according to the market research future. It is not only applicable to large company's big data like Amazon, Facebook, and Google but is also needed for smaller companies. It also finds its roads in fields like natural language processing and computer vision. The deep learning helps in better consumer credit decisions faster. A DL powered health care IoT such as wearable devices can save lives. Retailers will provide more recommendations to drive sales.

In the future, its applications will involve maintaining the privacy of data as data collection processes can include personal or critical business data and the vast collection of devices that includes simply designed hardware in IoT, network security, and data encryption. Several challenges still exist.

The first challenge is the lack of a large IoT dataset. The main hurdle for incorporating DL models into IoT devices as more data is needed for DL to achieve more accuracy. The more data will prevent overfitting. For this, seek DL with big data analytics with empirical validations and evaluation of the system. Permissions level need to be modified to access copyrighted data sets or privacy considerations with common domains with human data such as health and education. The range of suitable data sets would be of a lot of help for developers and researchers.

Preprocessing, or the preparation of raw data in appropriate representation is another challenge for IoT applications. For good results, data must be preprocessed. The CNN, an image processing technique works better if DL models are applied for normalized, scaled or transformed pixels into standard representations. For IoT applications, preprocessing is more complex since the system deals with a variety of data.

Guaranteed data security and privacy are of major concern for IoT and big data applications. Since IoT big data will transfer through the Internet for analytics and can be viewed from the real-world. During randomization in many applications, the same technique is hacked and re-identified as anonymized data. The DL data can also be subject to malicious attack such as false data injection or adversarial sample inputs, by which many functional or non-functional requirements of IoT systems for systems which are in danger. DL has the ability to learn from features from raw data and each, therefore, learns from raw data and also from invalid data feed to it. Developing further techniques to defend and prevent the effect of this sort of attack on DL models is necessary for reliable IoT applications. The data structuring and time complexity is another challenge. The voluminous number of input data, their broad number of attributes and a high degree of the classification result in a complex Dl model and affect performances during runtime.

Further, running DL on a distributed or clustered framework of devices is a viable solution for parallel processing can also be a challenge. The DL model, if it is assured, cannot be guaranteed to work on noisy or unlabeled data. The variety of data from various sources pops up the challenge of managing conflicts from different data sources. If no conflicts, it has the ability to work on heterogeneous data.

The high generation of data with high velocity brings a challenge of high-speed processing and analysis of data. As a solution for this challenge, DL should be incorporated with online and sequential learning.

The IoT big data will not be useful if the input data is not coming from a trustworthy source, greater importance should be given in handling online streams of analytics. Likewise, it generated problems for immense streams of data flowing at once. Data sampling is the best solution for such scenarios. The persons in charge of data analytics to improve their business are not clear of adopting these technologies, so abstractions should be provided in order to smoothen tasks of DL. Developing DL on IoT devices requires handling DNNs in resources constrained places. As a solution, new algorithms need to be developed in support of it. Further, in a few cases, DL may result in building fooling examples totally unrecognizable by humans More investigations are required to clear aspects of regression with DL as frequently in classification applications opt for it.

The following are a few future directions for new models involving DL, IoT, and big data.

First, investigations are required for utilizing mobile data in DL approaches to come up with better services for IoT domains in smart data set scenarios.

Future research is also needed for integrating appropriate information. The environment's situation cannot be understood by the IoT sensor data alone, hence there is a requirement to fuse with other sources of data. This process will also help in a fast analytical and quick reasoning engine.

More research is also needed on online resource provisioning for IoT analytics. The installment of fast DL based data analytics on for and cloud would require online provisioning of their data resource to host the stream of data. As streaming is applied knowing the data in advance is not feasible because of its new algorithms that have to be developed which support online auctioning in support of cloud or fog.

Semi-supervised analytic frameworks are also needed. It requires a large amount of training labeled data that is either not available or comes at a great expense to prepare based on International Data Consortium's report to estimate. A combination of advanced DL algorithms should be designed for semi-supervised settings that fit well for smarter training data set for agent learning improvements for getting more accuracy.

Dependable and reliable IoT analytics are also needed. As most of the applications are adopting IoT in large scales there is a need for mechanisms to ensure the safety of the system against malicious attack and as well against failures so, DL approaches are beneficial in identifying and predicting weak points. This will help the system to come up against faults and consequently increase the level of dependability of Big data-enabled IoT devices.

Future research is also needed for self-organizing communication networks. With the improvements in M2M communications, the configuration and maintenance of networking and other operations are becoming harder. In this direction, DL architectures will prove their competency in providing a range of self-services such as self-configuration, self-healing, self-optimization and self- load balancing.

In an application like unmanned aerial vehicles: It is a promising application; it can improve service delivery in hard- reaching regions or in critical situations. It can also be used in image analysis in real-time operations such as surveillance tasks, search and rescue, and infrastructure inspection. DL will provide a great impact to get the best decision making in these applications.

Virtual /augmented reality is another application area that could benefit both IoT and DL. In turn, it also influences other domains such as education, museums, and others.

Another area for future research is mobile robotics. It is currently used in commercial and industrial settings for moving materials or performing tasks in hazardous environments. In providing online responses DL model is applicable.

Finally, a strong understanding will cause less dependence on ready to use DL models which are based on the type of brute force at times. Only IoT with DL requires offline training for DL models. It can export the optimized parameters to IoT enabled big data devices. Hence, this parameter reduction will maintain the performance of the model and is always challenging to find tradeoff for applications.

CONCLUSION

Should devices communicate to another device or with another person realizing different risks, are there risks? Certainly. As with any new technology, a promise to accept both the potential benefits and risks that come with mainstream adoption are keys that influence a person's decision to use the technology. The optimistic advantage of an intelligent, interconnected world is a device with an actual physical presence is optimized to bring safe, more efficient experience. Thus, instead of spending time waiting for various real-time applications is important. This work provides valuable references for researchers and computer science practitioners alike to consider the techniques, tools, and applications of DL and ignites interest into areas of DL, IoT, and big data for future consideration.

REFERENCES

Abomhara, M. (2015). Cyber security and the internet of things: Vulnerabilities, threats, intruders and attacks. *Journal of Cyber Security and Mobility*, *4*(1), 65–88. doi:10.13052/jcsm2245-1439.414

Abu Alsheikh, M., Lin, S., Niyato, D., & Tan, H.-P. (2016). Rate-distortion balanced data compression for wireless sensor networks. *IEEE Sensors Journal*, *16*(12), 5072–5083. doi:10.1109/JSEN.2016.2550599

Aggarwal, Ashish, & Sheth. (2013). *The Internet Of Things: A Survey From The Data-Centric Perespective, Managing And Mining Sensor Data*. Springer Science+Business Media.

Agrawal, S., & Vieira, D. (2013). A Survey On Internet Of Things. *Abakós, Belo Horizonte VOL.*, *1*(2), 78–95.

Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications, 60*, 19–31. Available: https://www.sciencedirect.com/science/article/pii/S1084804515002891

Alkhateeb, A., Alex, S., Varkey, P., Li, Y., Qu, Q., & Tujkovic, D. (2018). Deep Learning Coordinated Beamforming for Highly-Mobile Millimeter Wave Systems. *IEEE Access: Practical Innovations, Open Solutions*, *6*, 37328–37348. doi:10.1109/ACCESS.2018.2850226

Amato, G., Carrara, F., Falchi, F., Gennaro, C., Meghini, C., & Vairo, C. (2016). Deep learning for decentralized parking lot occupancy detection. *Expert Systems with Applications*, 72.

Amato, G., Carrara, F., Falchi, F., Gennaro, C., Meghini, C., & Vairo, C. (2017). Deep learning for decentralized parking lot occupancy detection. *Expert Systems with Applications*, *72*, 327–334. doi:10.1016/j.eswa.2016.10.055

Aminanto, M. E., & Kim, K. (2016). Detecting impersonation attack in WiFi networks using deep learning approach. *Proc. International Workshop on Information Security Applications*, 136–147.

Aminian, M., & Aminian, F. (2000). Neural-network based analog-circuit fault diagnosis using wavelet transform as preprocessor. *IEEE Transactions on Circuits and Systems. 2, Analog and Digital Signal Processing*, *47*(2), 151–156. doi:10.1109/82.823545

Angermueller, C., Lee, H., Reik, W., & Stegle, O. (2016). Accurate prediction of single-cell dna methylation states using deep learning. *bioRxiv*, 055715.

Anthimopoulos, M., Christodoulidis, S., Ebner, L., Christe, A., & Mougiakakou, S. (2016, May). Lung pattern classification for interstitial lung diseases using a deep convolutional neural network. *IEEE Transactions on Medical Imaging*, *35*(5), 1207–1216. doi:10.1109/TMI.2016.2535865

Avendi, M., Kheradvar, A., & Jafarkhani, H. (2016). Acombined deep-learning and deformable-model approach to fully automatic segmentation of the left ventricle in cardiac mri. *Medical Image Analysis*, *30*, 108–119. doi:10.1016/j.media.2016.01.005

Babu, G. S., Zhao, P., & Li, X.-L. (2016). Deep convolutional neural network based regression approach for estimation of remaining useful life. In *International Conference on Database Systems for Advanced Applications*. Springer. 10.1007/978-3-319-32025-0_14

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *International Conference on Learning Representations (ICLR 2015)*.

Banihashemian, S. S., Adibnia, F., & Mehdi, A. (2018). Sarram. A new range-free and storage-efficient localization algorithm using neural networks in wireless sensor networks. *Wireless Personal Communications*, *98*(1), 1547–1568. doi:10.100711277-017-4934-4

Bengio, Y., & LeCun, Y. (2007). Scaling learning algorithms towards, AI. In L. Bottou, O. Chapelle, D. DeCoste, & J. Weston (Eds.), *Large Scale Kernel Machines* (Vol. 34, pp. 321–360). Cambridge, MA: MIT Press. http://www.iro.umontreal.ca/~ lisa/pointeurs/bengio+lecun_chapter2007.pdf

Bernas, M., & Płaczek, B. (2015). Fully connected neural networks ensemble with signal strength clustering for indoor localization in wireless sensor networks. *International Journal of Distributed Sensor Networks*, *11*(12), 403242. doi:10.1155/2015/403242

Bertino, E., & Islam, N. (2017). Botnets and internet of things security. *Computer*, *50*(2), 76–79. doi:10.1109/MC.2017.62

Bidnessetc.com. (2014). *Internet of Things: Connecting (And Disrupting) Your Universe*. Available at: https://www.bidnessetc.com/business/internet-of-things-connecting-and-disrupting-your-universe/

Bishop, C. (1992). Pattern recognition and machine learning. Springer.

Bordes, A., Glorot, X., Weston, J., & Bengio, Y. (2012)/ Joint learning of words and meaning representations for open-text semantic parsing. *International Conference on Artificial Intelligence and Statistics*, 127–135.

Borgerding, M., Schniter, P., & Rangan, S. (2017). AMP-inspired deep networks for sparse linear inverse problems. *IEEE Transactions on Signal Processing*, *65*(16), 4293–4308. doi:10.1109/TSP.2017.2708040

Brosch, T. (2013). Manifold learning of brain mris by deep learning. *Proc. MICCAI*, 633–640. 10.1007/978-3-642-40763-5_78

Cambria, E., Huang, G. B., Kasun, L. L. C., Zhou, H., Vong, C. M., Lin, J., ... Liu, J. (2013, November). Extreme learning machines [trends controversies]. *IEEE Intelligent Systems*, *28*(6), 30–59. doi:10.1109/MIS.2013.140

Cao. (2016). Improving tuberculosis diagnostics using deep learning and mobile health technologies among resource-poor and marginalized communities. *IEEE Connected Health, Appl., Syst. Eng. Technol.*, 274–281.

Challita, U., Saad, W., & Bettstetter, C. (2018). Deep reinforcement learning for interference-aware path planning of cellular connected UAVs. *Proc. IEEE International Conference on Communications (ICC)*. 10.1109/ICC.2018.8422706

Che, Purushotham, Khemani, & Liu. (2015). *Distilling knowledge from deep networks with applications to healthcare domain*. ArXiv e-prints.

Chen, Lingys, Chen, & Liu. (2018). AuTO: scaling deep reinforcement learning for datacenter-scale automatic traffic optimization. *Proc. ACM Conference of the ACM Special Interest Group on Data Communication (SIGCOMM)*, 191–205.

Chen, H., Ni, D., Qin, J., Li, S., Yang, X., Wang, T., & Heng, P. A. (2015, September). Standard plane localization in fetal ultrasound via domain transferred deep neural networks. *IEEE Journal of Biomedical and Health Informatics*, *19*(5), 1627–1636. doi:10.1109/JBHI.2015.2425041

Chen, L., Yang, D., Zhang, D., Wang, C., Li, J., & Nguyen, T.-M.-T. (2018). Jonathan Li, and Thi-Mai-Trang Nguyen. Deep mobile traffic forecast and complementary base station clustering for C-RAN optimization. *Journal of Network and Computer Applications*, *121*, 59–69. doi:10.1016/j.jnca.2018.07.015

Chen, S., Xu, H., Liu, D., Hu, B., & Wang, H. (2014, August). A Vision of IoT: Applications, Challenges,and Opportunities With China Perspective. *IEEE Internet of Things Journal, 1*(4), 349–359. doi:10.1109/JIOT.2014.2337336

Chen, X.-W., & Lin, X. (2014). Big data deep learning: Challenges and perspectives. *IEEE Access: Practical Innovations, Open Solutions*, *2*, 514–525. doi:10.1109/ACCESS.2014.2325029

Chen, Z., & Smith. (2018). Heterogeneous machine-type communications in cellular networks: Random access optimization by deep reinforcement learning. *Proc. IEEE International Conference on Communications (ICC)*, 1–6.

Chen, Z., Li, C., & Sanchez, R.-V. (2015). *Gearbox fault identification and classification with convolutional neural networks* (Vol. 2015). Shock and Vibration.

Chen, Z., & Li, W. (2017). *Multi-sensor feature fusion for bearing fault diagnosis using sparse auto encoder and deep belief network. IEEE Transactions on IM*.

Cheng, J.-Z., Ni, D., Chou, Y.-H., Qin, J., Tiu, C.-M., Chang, Y.-C., ... Chen, C.-M. (2016). Computer-aided diagnosis with deep learning architecture: Applications to breast lesions in us images and pulmonary nodules in ct scans. *Scientific Reports*, *6*(1), 24454. doi:10.1038rep24454

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. arXiv:1406.1078

Choi, S., Kim, E., & Oh, S. (2013). Human behavior prediction for smart homes using deep learning. *Proc. IEEE RO-MAN*, 173–179.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). *Empirical evaluation of gated recurrent neural networks on sequence modeling*. arXiv preprint arXiv:1412.3555v1 [cs.NE]

Collobert, R., & Bengio, S. (2004). Links between perceptrons, MLPs and SVMs. *Proc. twenty-first ACM international conference on Machine learning*, 23. 10.1145/1015330.1015415

Costa, K. A. P., & Pereira, R. Y. M. (2015). *A nature-inspired approach to speed up optimum-path forest clustering and its application to intrusion detection in computer networks* (Vol. 294). Information Sciences. Available https://www.sciencedirect.com/science/article/pii/S0020025514009311

Cusumano, M.A. (2005). Google: What it is and what it is not. *Commun ACM - Med Image Moeling, 48*(2), 15–17. doi:10.1145/1042091.1042107

Daniel, E. (2013). O'Leary. Artificial intelligence and big data. *IEEE Intelligent Systems*, *28*(2), 96–99. doi:10.1109/MIS.2013.39

Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Le, Q., . . . Ng, A. (2012). Large scale distributed deep networks. In P. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems, (Vol. 25, pp. 1232–1240). http://books.nips.cc/papers/files/nips25/NIPS2012_0598.pdf

Deutsch & He. (2016). *Using deep learning based approaches for bearing remaining useful life prediction*. Academic Press.

Devi, S., & Neetha, T. (2017). Machine Learning based traffic congestion prediction in a IoT based Smart City. *Int. Res. J. Eng. Technol.*, *4*, 3442–3445.

Di Huang, C., Chien, Y. W., & Jia, C. W. (2016). Human action recognition system for elderly and children care using three stream ConvNet. *International Conference on Orange Technologies*, 5–9.

Ding, X., & He, Q. (2017). *Energy-fluctuated multiscale feature learning with deep convnet for intelligent spindle bearing fault diagnosis*. *IEEE Transactions on IM*. doi:10.1109/TIM.2017.2674738

Diro, A., & Chilamkurti, N. (2018). *Distributed attack detection scheme using deep learning approach for internet of things* (Vol. 82). Future Generation Computer Systems. Available https://www.sciencedirect.com/science/article/pii/S0167739X17308488

Dörner, S., Cammerer, S., Hoydis, J., & ten Brink, S. (2018). Deep learning based communication over the air. *IEEE Journal of Selected Topics in Signal Processing*, *12*(1), 132–143. doi:10.1109/JSTSP.2017.2784180

Douarre, C., Schielein, R., Frindel, C., Gerth, S., & Rousseau, D. (2016). Deep learning based root-soil segmentation from X-ray tomography. *bioRxiv*, 071662.

Elliss-Brookes, L. (2015, Nov. 10). Big data in action: the story behind Routes to Diagnosis. *Public health matters blog*.

Enache, A.-C., Sgarciu, V., & Petrescu-Nit¸a˘, A. (2015). Intelligent feature selection method rooted in binary bat algorithm for intrusion detection. *2015 IEEE 10th jubilee international symposium on applied computational intelligence and informatics (SACI)*, 517–521. 10.1109/SACI.2015.7208259

Erhan, D., Bengio, Y., Courville, A., & Vincent, P. (2009). *Visualizing higher-layer features of a deep network*. Technical Report 1341, Universit´e de Montr´eal.

Evans, D. (2011). *The internet of things: How the next evolution of the internet is changing everything.* CISCO White Paper.

Evans, D. (2011). *The internet of things: How the next evolution of the internet is changing everything*. CISCO White Paper.

Felbo, B., Sundsøy, P., Pentland, A., Lehmann, S., & de Montjoye, Y.-A. (2016). *Using deep learning to predict demographics from mobile phone metadata*. Available: https://arxiv.org/abs/1511.06660

Feng, Q., Dou, Z., Li, C., & Si, G. (2016). Anomaly detection of spectrum in wireless communication via deep autoencoder. *Proc. International Conference on Computer Science and its Applications*, 259–265.

Freytag, A., Rodner, E., Bodesheim, P., & Denzler, J. (2013). Labeling Examples that Matter: Relevance-Based Active Learning with Gaussian Processes. In *35th German Conference on Pattern Recognition (GCPR).* Saarland University an Max-Planck-Institute for Informatics.

Fritscher, K., Raudaschl, P., Zaffino, P., Spadea, M. F., Sharp, G. C., & Schubert, R. (2016). Deep neural networks for fast segmentation of 3d medical images. *Proc. MICCAI*, 158–165. 10.1007/978-3-319-46723-8_19

Fu, T., Han, Y., Li, X., Liu, Y., & Wu, X. (2015). Integrating prosodic information into recurrent neual network language model for speech recognition. *Proceedings of Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, 1194–1197.

Fujihashi, Koike-Akino, Watanabe, & Orlik. (2018). Nonlinear equalization with deep learning for multipurpose visual MIMO communications. *Proc. IEEE International Conference on Communications (ICC)*, 1–6.

Futoma, J., Morris, J., & Lucas, J. (2015). A comparison of models for predicting early hospital readmissions. *Journal of Biomedical Informatics*, *56*, 229–238. doi:10.1016/j.jbi.2015.05.016

Galloway, Catterson, Fay, Robb, & Love. (2016). *Diagnosis of tidal turbine vibration data through deep neural networks*. Academic Press.

Gan, M., Wang, C., & Zhu, C. (2016). Construction of hierarchical diagnosis network based on deep learning and its application in the fault pattern recognition of rolling element bearings. *Mechanical Systems and Signal Processing*, *72*, 92–104. doi:10.1016/j.ymssp.2015.11.014

Gante, J., Falcão, G., & Sousa, L. (2018). *Beamformed Fingerprint Learning for Accurate Millimeter Wave Positioning*. arXiv preprint arXiv:1804.04112

Garimella, V. R. K., Alfayad, A., & Weber, I. (2016). *Social media image analysis for public health. In Proc. CHIConf* (pp. 5543–5547). Human Factors Comput. Syst. Available http://doi.acm.org/10.1145/2858036.2858234

Gartner. (2015). *Big Data Management and Analytics*. Available: https://www.gartner.com/technology/topics/big-data.jsp

Geoffrey, E. (2006, July). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504–507. doi:10.1126cience.1127647

Gers, F., Schraudolph, N. N., & Schmidhuber, J. (2016). *Learning Precise Timing with LSTM Recurrent Networks*. ResearchGate.

GhaffarianHoseini, A. H., Dahlan, N. D., Berardi, U., GhaffarianHoseini, A., & Makaremi, N. (2013). The essence of future smart houses: From embedding ICT to adapting to sustainability principles. *Renewable & Sustainable Energy Reviews*, *24*, 593–607. doi:10.1016/j.rser.2013.02.032

Gheisari, M., Wang, G., & Md, Z. A. B. (2017). A survey on deep learning in big data. *Proc. IEEE International Conference on Computational Science and Engineering (CSE) and Embedded and Ubiquitous Computing (EUC), 2*, 173–180. 10.1109/CSE-EUC.2017.215

Gill, K., Yang, S.-H., Yao, F., & Lu, X. (2009). A zigbee-based home automation system. *IEEE Transactions on Consumer Electronics*, *5*(2), 422–430. doi:10.1109/TCE.2009.5174403

Gokhale, Bhat, & Bhat. (2007). Introduciton to IoT. *International Advanced Research Journal in Science, Engineering and Technology, 5*(1). 41 doi:10.17148/IARJSET.2018.517

Gomez, C., & Paradells, J. (2010). Wireless home automation networks: A survey of architectures and technologies. *IEEE Communications Magazine*, *48*(6), 92–101. doi:10.1109/MCOM.2010.5473869

Gonzalez, R., Garcia-Duran, A., Manco, F., Niepert, M., & Vallina, P. (2017). Network data monetization using Net2Vec. *Proc. ACM SIGCOMM Posters and Demos*, 37–39. 10.1145/3123878.3131978

Goodfello, I. (2016). *Deep learning*. MIT Press.

Goodfellow, I. (2016). *Deep learning*. MIT Press.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 2672–2680.

Gopalakrishnan, K. (2018). Deep Learning in Data-Driven Pavement Image Analysis and Automated Distress Detection: A Review. *Data*, *3*(3), 28. doi:10.3390/data3030028

Graves, A., & Schmidhuber, J. (2005). (Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *IJCNN*, *18*(5), 602–610.

Greenspan, H., van Ginneken, B., & Summers, R. M. (2016, May). Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. *IEEE Transactions on Medical Imaging*, *35*(5), 1153–1159. doi:10.1109/TMI.2016.2553401

Griewank, A., & Walther, A. (2008). *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation* (2nd ed.). SIAM. doi:10.1137/1.9780898717761

Guan, W., Wu, Y., Xie, C., Chen, H., Cai, Y., & Chen, Y. (2017). High-precision approach to localization scheme of visible light communication based on artificial neural networks and modified genetic algorithms. *Optical Engineering (Redondo Beach, Calif.)*, *56*(10), 106103. doi:10.1117/1.OE.56.10.106103

Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet Of Things (Iot): A Vision, Architectural Elements, And Future Directions. *Future Generation Computer Systems*, *29*(7), 1645–1660. doi:10.1016/j.future.2013.01.010

Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013, September). IoT: A vision, architectural elements, and future directions. *Future Generation Computer Systems*, *29*(7), 1645–1660. doi:10.1016/j.future.2013.01.010

Guo, L., Gao, H., Huang, H., He, X., & Li, S. (2016). *Multifeatures fusion and nonlinear dimension reduction for intelligent bearing condition monitoring* (Vol. 2016). Shock and Vibration.

Guo, X., Chen, L., & Shen, C. (2016). Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis. *Measurement*, *93*, 490–502. doi:10.1016/j.measurement.2016.07.054

Ha, S., Yun, J. M., & Choi, S. (2015). Multi-modal convolutional neural networks for activity recognition. *Proc. Int. Conf. Syst., Man, Cybern*., 3017–3022. 10.1109/SMC.2015.525

Hamedani, K., Liu, L., Atat, R., Wu, J., & Yi, Y. (2018). Reservoir computing meets smart grids: Attack detection using delayed feedback networks. *IEEE Transactions on Industrial Informatics*, *14*(2), 734–743. doi:10.1109/TII.2017.2769106

Havaei, M., Guizard, N., Larochelle, H., & Jodoin, P. (2016). *Deep learning trends for focal brain pathology segmentation in MRI*. Available: https://arxiv.org/abs/1607.05258

He, Y., Liang, C., Richard Yu, F., Zhao, N., & Yin, H. (2017). Optimization of cache-enabled opportunistic interference alignment wireless networks: A big data deep reinforcement learning approach. *Proc.* 2017 *IEEE International Conference on Communications (ICC)*, 1–6. 10.1109/ICC.2017.7996332

He, Y., Richard Yu, F., Zhao, N., & Victor, C. M. (2017). Software-defined networks with mobile edge computing and caching for smart cities: A big data deep reinforcement learning approach. *IEEE Communications Magazine*, *55*(12), 31–37. doi:10.1109/MCOM.2017.1700246

Hermanto, A., Adji, T. B., & Setiawan, N. A. (2015). Recurrent neural network language model for english-indonesian machine translation: Experimental study. *Proceedings of International Conference on Science in Information Technology*, 132–136. 10.1109/ICSITech.2015.7407791

Heydari & Minaei-Bidgoli. (2017). Reduce energy consumption and send secure data wireless multimedia sensor networks using a combination of techniques for multi-layer watermark and deep learning. *International Journal of Computer Science and Network Security*, *17*(2), 98–105.

Hinton, G., & Salakhutdinov, R. (2011). Discovering binary codes for documents by learning deep generative models. *Topics in Cognitive Science*, *3*(1), 74–79. doi:10.1111/j.1756-8765.2010.01109.x

Hitaj, Gasi, Ateniese, & Perez- Cruz. (2017). *PassGAN: A deep learning approach for password guessing*. arXiv preprint arXiv:1709.00440

Hordri, N. F., Samar, A., Yuhaniz, S. S., & Shamsuddin, S. M. (2017). A systematic literature review on features of deep learning in big data analytics. *International Journal of Advances in Soft Computing & Its Applications*, *9*(1).

Huang, C.-D., Wang, C.-Y., & Wang, J.-C. (2015). Human action recognition system for elderly and children care using three stream convnet. *Proc. Int. Conf. Orange Technol.*, 5–9. 10.1109/ICOT.2015.7498476

Huang, H., Jie, Y., Song, Y., Hao, H., & Guan, G. (2018). Deep Learning for Super-Resolution Channel Estimation and DOA Estimation based Massive MIMO System. *IEEE Transactions on Vehicular Technology*, *67*(9), 8549–8560. doi:10.1109/TVT.2018.2851783

Huang, J., Zhou, W., Li, H., & Li, W. (2015). Sign language recognition using real-sense. *Proc. IEEE ChinaSIP*, 166–170.

Huang, S., & Lin, H. (2018). Fully optical spacecraft communications: Implementing an omnidirectional PV-cell receiver and 8 Mb/s LED visible light downlink with deep learning error correction. *IEEE Aerospace and Electronic Systems Magazine*, *33*(4), 16–22. doi:10.1109/MAES.2018.170179

Huang, W., Song, G., Hong, H., & Xie, K. (2014). Deep architecture for traffic flow prediction: Deep belief networks with multitask learning. *IEEE Transactions on Intelligent Transportation Systems*, *15*(5), 2191–2201. doi:10.1109/TITS.2014.2311123

Huijie, Z., Ting, R., Xinqing, W., You, Z., & Husheng, F. (2015). Fault diagnosis of hydraulic pump based on stacked autoencoders. *2015 12th IEEE International Conference on Electronic Measurement Instruments (ICEMI), 1*, 58–62. 10.1109/ICEMI.2015.7494195

Ibrahim, Yousri, Ismail, & El-Makky. (2014). Multi-level gene/mirna feature selection using deep belief nets and active learning. *Proc. Eng. Med. Biol. Soc.*, 3957–3960.

Ienco, D., Gaetano, R., Dupaquier, C., & Maurel, P. (2017). *Land Cover Classification via Multi-temporal Spatial Data by Recurrent Neural Networks*. arXiv preprint arXiv:1704.04055

Imtiaz, H., & Fattah, S. A. (2011). A wavelet-domain local dominant feature selection scheme for face recognition. *International Journal of Computing and Business Research, 3*(2).

Innova.com.tr. (2015). *Innova deploys smart home and smart office services for TTNET | Innova*. Available at: http://www.innova.com.tr/en/newsdetail. asp?haber=777C888F-C922-8822-EEE2-BBB222FF333C

Janssens, O., Slavkovikj, V., Vervisch, B., Stockman, K., Loccufier, M., Verstockt, S., ... Van Hoecke, S. (2016). Convolutional neural network based fault detection for rotating machinery. *Journal of Sound and Vibration*, *377*, 331–345. doi:10.1016/j.jsv.2016.05.027

Jia, F., Lei, Y., Lin, J., Zhou, X., & Lu, N. (2016J). Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data. *Mechanical Systems and Signal Processing*, *72*, 303–315. doi:10.1016/j.ymssp.2015.10.025

Jia, X., Li, K., Li, X., & Zhang, A. (2014). A novel semi-supervised deep learning framework for affective state recognition on eeg signals. *Proc.Int. Conf. Bioinformat. Bioeng.*, 30–37. 10.1109/BIBE.2014.26

Jiang, Wang, Luo, Zhang, Xiong, & Pang. (2015). Convolutional neural networks in automatic recognition of trans-differentiated neural progenitor cells under bright-field microscopy. *Proc. Instrum. Meas., Comput., Commun. Control*, 122–126.

Jin, Y., & Raychowdhury, A. (2016). Internet of Things realize its potential. *Computer*, *49*(8), 112–116. doi:10.1109/MC.2016.245

Junbo, T., Weining, L., Juneng, A., & Xueqian, W. (2015). Fault diagnosis method study in roller bearing based on wavelet transform and stacked auto-encodeqr. In *The 27th Chinese Control and Decision Conference (2015 CCDC).* IEEE.

Kaed, C., Leida, B., & Gray, T. (2016). Building management insights driven by a multi-system semantic representation approach. *2016 IEEE 3rd world forum on internet of things (WF-IoT)*, 520–525. 10.1109/WF-IoT.2016.7845433

Kang, J., Park, Y.-J., Lee, J., Wang, S.-H., & Eom, D.-S. (2018). Novel Leakage Detection by Ensemble CNN-SVM and Graph-based Localization in Water Distribution Systems. *IEEE Transactions on Industrial Electronics*, *65*(5), 4279–4289. doi:10.1109/TIE.2017.2764861

Kasfi, K. T., Hellicar, A., & Rahman, A. (2016). Convolutional neural network for time series cattle behaviour classification. In *Proceedings of the Workshop on Time Series Analytics and Applications*. ACM. 10.1145/3014340.3014342

Kendra, R. L., Karki, S., Eickholt, J. L., & Gandy, L. (2015). Characterizing the discussion of antibiotics in the Twittersphere: What is the bigger picture? *J. Med. Internet Res., 17*(6).

Khademi, M., & Nedialkov, N. S. (2015). Probabilistic graphical models and deep belief networks for prognosis of breast cancer. *Proc. IEEE 14th Int. Conf. Mach. Learn. Appl.*, 727–732. 10.1109/ICMLA.2015.196

Khan, M. A., Khan, S., Shams, B., & Lloret, J. (2016). Distributed flood attack detection mechanism using artificial neural network in wireless mesh networks. *Security and Communication Networks*, *9*(15), 2715–2729. doi:10.1002ec.1204

Khorasani, F., & Naji, H. R. (2017). Energy efficient data aggregation in wireless sensor networks using neural networks. *International Journal of Sensor Networks*, *24*(1), 26–42. doi:10.1504/IJSNET.2017.084207

Kim, H.-Y., & Kim, J.-M. (2017). A load balancing scheme based on deep-learning in IoT. *Cluster Computing*, *20*(1), 873–878. doi:10.100710586-016-0667-5

Kingma, D. P., & Welling, M. (2014). Stochastic gradient VB and the variational auto-encoder. *Proceedings of the Second International Conference on Learning Representations (ICLR)*.

Kipf & Welling. (2107). Semi-supervised classification with graph convolutional networks. *Proc. International Conference on Learning Representations (ICLR)*.

Kleesiek, J., Urban, G., Hubert, A., Schwarz, D., Maier-Hein, K., Bendszus, M., & Biller, A. (2016). Deep MRI brain extraction: A 3D Convolutional neural network for skull stripping. *NeuroImage*, *129*, 460–469. doi:10.1016/j.neuroimage.2016.01.024

Kondo, T., Ueno, J., & Takao, S. (2016). Medical image diagnosis of lung cancer by deep feedback GMDH-type neural network. *Robot. Netw. Artif. Life*, *2*(4), 252–257. doi:10.2991/jrnal.2016.2.4.11

Kondo, T., Ueno, J., & Takao, S. (2014). Medical image recognition of abdominal multi-organs by hybrid multi-layered GMDH-type neural network using principal component-regression analysis. *Proc. 2nd Int. Symp. Comput. Netw.*, 157–163. 10.1109/CANDAR.2014.62

Krizhevsky, Sutskever, & Hinton. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 1097–1105.

Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems. Curran Associates, Inc.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Proceedings of the Advances in Neural Information Processing Systems (NIPS 2012)*.

Kuang & He. (2014). Classification on ADHD with deep learning. *Proc. Cloud Comput. Big Data*, 27–32.

Kuhad, P., Yassine, A., & Shimohammadi, S. (2015). Using distance estimation and deep learning to simplify calibration in food calorie measurement. *Proc. IEEE Int. Conf. Comput. Intell. Virtual Environ. Meas. Syst. Appl.*, 1–6. 10.1109/CIVEMSA.2015.7158594

Kumar, R., Talton, J. O., Ahmad, S., & Klemmer, S. R. (2012). Data-driven web design. *Proceedings of the 29th International.*

Kurian, C., & Balakriahnan, K. (2012). Cont inuous speech recognition system for Malayalam language using PLP cepstral coefficient. *Journal of Computing and Business Research, 3*(1).

Kuwata, K., & Shibasaki, R. (2015). Estimating crop yields with deep learning and remotely sensed data. *Geoscience and Remote Sensing Symposium*, 858-861. 10.1109/IGARSS.2015.7325900

Ladhak, Nazi, & Huber. (2013). Using deep learning to enhance cancer diagnosis and classification. *Proc. Int. Conf. Mach. Learn.*, 1–7.

Le, Q. V., Zou, W. Y., Yeung, S. Y., & Ng, A. Y. (2011) Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. *Computer Vision and Pattern Recognition (CVPR) 2011 IEEE Conference On*, 3361–3368. 10.1109/CVPR.2011.5995496

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. doi:10.1038/nature14539

Lee. (2017). Deep learning–based real-time query processing for wireless sensor network. *International Journal of Distributed Sensor Networks*, *13*(5).

Li, F., Tran, L., Thung, K. H., Ji, S., Shen, D., & Li, J. (2015, September). A robust deep model for improved classification of ad/mci patients. *IEEE Journal of Biomedical and Health Informatics*, *19*(5), 1610–1616. doi:10.1109/JBHI.2015.2429556

Li, G., Zhu, H., Cheng, G., Thambiratnam, K., Chitsaz, B., Yu, D., & Seide, F. (2012). Context-dependent deep neural networks for audio indexing of real-life data. In *Spoken Language Technology Workshop (SLT), 2012 IEEE* (pp. 143–148). IEEE. doi:10.1109/SLT.2012.6424212

Li, K., & Wang, Q. (2015). Study on signal recognition and diagnosis for spacecraft based on deep learning method. *Prognostics and System Health Management Conference (PHM)*, 1–5.

Liang, Shen, & Wu. (n.d.). *Exploiting noise correlation for channel decoding with convolutional neural networks*. Academic Press.

Liang, Z., Zhang, G., Huang, J. X., & Hu, Q. V. (2014). Deep learning for healthcare decision making with emrs. *Proc. Int. Conf. Bioinformat. Biomed.*, 556–559. 10.1109/BIBM.2014.6999219

Liao, L., Jin, W., & Pavel, R. (2016). Enhanced restricted boltzmann machine with prognosability regularization for prognostics and health assessment. *IEEE Transactions on Industrial Electronics*, *63*(11), 7076–7083. doi:10.1109/TIE.2016.2586442

Liao, R.-F., Wen, H., Wu, J., Song, H., Pan, F., & Dong, L. (2018). The Rayleigh Fading Channel Prediction via Deep Learning. *Wireless Communications and Mobile Computing*, *2018*, 1–11. doi:10.1155/2018/6497340

Lin, Z., & Yin, M. (2017). Deep generative models of urban mobility. *IEEE Transactions on Intelligent Transportation Systems*.

Lipton, Z. C., Kale, D. C., Elkan, C., & Wetzel, R. C. (2015). *Learning to diagnosewithLSTMrecurrent neural networks*. vailable: https://arxiv.org/abs/1511.03677

Lipton, Kale, Elkan, & Wetzell. (2015). Learning to diagnose with LSTM recurrent neural networks. *International Conference on Learning Representations (ICLR 2016).*

Liu, C. H., Chen, Z., Tang, J., Xu, J., & Piao, C. (2018). Energy-efficient UAV control for effective and fair communication coverage: A deep reinforcement learning approach. *IEEE Journal on Selected Areas in Communications*, *36*(9), 2059–2070. doi:10.1109/JSAC.2018.2864373

Liu, H., Li, L., & Ma, J. (2016). *Rolling bearing fault diagnosis based on stft-deep learning and sound signals* (Vol. 2016). Shock and Vibration.

Liu, S., & Du, J. (2016). Poster: Mobiear-building an environment independent acoustic sensing platform for the deaf using deep learning. *Proc. 14th ACM Annual International Conference on Mobile Systems, Applications, and Services Companion*, 50–50. 10.1145/2938559.2948831

Liu, X., Xu, Y., Jia, L., Wu, Q., & Anpalagan, A. (2018). Anti-jamming Communications Using Spectrum Waterfall: A Deep Reinforcement Learning Approach. *IEEE Communications Letters*, *22*(5), 998–1001. doi:10.1109/LCOMM.2018.2815018

Liu, Y., & Wu, L. (2016). Geological disaster recognition on optical remote sensing images using deep learning. *Procedia Computer Science*, *91*, 566–575. doi:10.1016/j.procs.2016.07.144

Lu, C., Wang, Z.-Y., Qin, W.-L., & Ma, J. (2017). Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification. *Signal Processing*, *130*, 377–388. doi:10.1016/j.sigpro.2016.07.028

Lu, W., Wang, X., Yang, C., & Zhang, T. (2015). A novel feature extraction method using deep neural network for rolling bearing fault diagnosis. *The 27th Chinese Control and Decision Conference (2015 CCDC)*, 2427–2431. 10.1109/CCDC.2015.7162328

Luo, C., Ji, J., Wang, Q., Yu, L., & Li, P. (2018). Online power control for 5G wireless communications: A deep Q-network approach. *Proc. IEEE International Conference on Communications (ICC)*, 1–6. 10.1109/ICC.2018.8422442

Luo, T., & Nagarajany. (2018). Distributed anomaly detection using autoencoder neural networks in WSN for IoT. *Proc. IEEE International Conference on Communications (ICC)*, 1–6.

Lyu, W., Zhang, Z., Jiao, C., Qin, K., & Zhang, H. (2018). Performance evaluation of channel decoding with deep neural networks. *Proc. IEEE International Conference on Communications (ICC)*, 1–6. 10.1109/ICC.2018.8422289

Mansoor, A., Cerrolaza, J. J., Idrees, R., Biggs, E., Alsharid, M. A., Avery, R. A., & Linguraru, M. G. (2016, August). Deep learning guided partitioned shape model for anterior visual pathway segmentation. *IEEE Transactions on Medical Imaging*, *35*(8), 1856–1865. doi:10.1109/TMI.2016.2535222

Mao, H., Netravali, R., & Alizadeh, M. (2017). Neural adaptive video streaming with pensieve. In *Proc. Conference of the ACM Special Interest Group on Data Communication (SIGCOMM)*, (pp. 197–210). ACM. 10.1145/3098822.3098843

Mao, W., He, J., Li, Y., & Yan, Y. (2016). Bearing fault diagnosis with autoencoder extreme learning machine: A comparative study. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 1–19.

Mariolis, I., Peleka, G., & Kargakos, A. (2015). Pose and category recognition of highly deformable objects using deep learning. *Advanced Robotics (ICAR), 2015 International Conference on*, 655-662. 10.1109/ICAR.2015.7251526

Maryam, M. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*, *2*(1), 1.

Mehrabi, S. (2015). Temporal pattern and association discovery of diagnosis codes using deep learning. *Proc. Int. Conf. Healthcare Informat.*, 408–416. 10.1109/ICHI.2015.58

Mikolov, T., Chen, K., & Dean, J. (2013). Efficient estimation of word representations in vector space. CoRR. *Computing*.

Mikolov, T., Le, Q.V., & Sutskever, I. (2013). *Exploiting similarities among languages for machine translation*. CoRR.

Milioto, A., Lottes, P., & Stachniss, C. (2017). Real-time blob-wise sugar beets vs weeds classification for monitoring fields using convolutional neural networks. *Proceedings of the International Conference on Unmanned Aerial Vehicles in Geomatics*. 10.5194/isprs-annals-IV-2-W3-41-2017

Miotto, R., Li, L., & Brian, A. (2016). Deep patient: An unsupervised representation to predict the future of patients from the electronic health records. *Scientific Reports*, *6*(1), 26094. doi:10.1038rep26094

Miotto, R., Li, L., Kidd, B. A., & Dudley, J. T. (2016). Deep patient: An unsupervised representation to predict the future of patients from the electronic health records. *Scientific Reports*, *6*(1), 1–10. doi:10.1038rep26094

Mittal, Yagnik, Garg, & Krishnan. (2016). Spotgarbage: smartphone app to detect garbage using deep learning. *Proc. ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 940–945.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533. doi:10.1038/nature14236

Molchanov, P., Yang, X., Gupta, S., Kim, K., Tyree, S., & Kautz, J. (2016). Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network. *Proceedings of the IEEEConference on Computer Vision and Pattern Recognition*, 4207–4215. 10.1109/CVPR.2016.456

Mori, S., Suen, C. Y., & Kamamoto, K. (1992, July). Historical review of OCR research and development. *Proceedings of the IEEE*, *80*(7), 1029–1058. doi:10.1109/5.156468

Muhammad, K., Ahmad, J., & Baik, S. W. (2018). Early fire detection using convolutional neural networks during surveillance for effective disaster management. *Neurocomputing*, *288*, 30–42. doi:10.1016/j.neucom.2017.04.083

Munoz-Organero, M., Ruiz-Blaquez, R., & Sánchez-Fernández, L. (2018). Automatic detection of traffic lights, street crossings and urban roundabouts combining outlier detection and deep learning classification techniques based on GPS traces while driving. *Computers, Environment and Urban Systems*, *68*, 1–8. doi:10.1016/j.compenvurbsys.2017.09.005

Naik, N. (2017). Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. *Proceedings of the 2017 IEEE international systems engineering symposium (ISSE)*, 1–7. 10.1109/SysEng.2017.8088251

Neverova, N., Wolf, C., & Taylor, G. W. (2014). Multi-scale deep learning for gesture detection and localization. Computer Vision-ECCV 2014 Workshops, 474-490.

Ng, J. R., Wong, J. S., Goh, V. T., Yap, W. J., Yap, T. T. V., & Ng, H. (2019). Identification of Road Surface Conditions using IoT Sensors and Machine Learning. In *Computational Science and Technology* (pp. 259–268). Singapore: Springer. doi:10.1007/978-981-13-2622-6_26

Nguyen, Al-Mannai, Joty, Sajjad, Imran, & Mitra. (2017). Robust classification of crisis-related data on social networks using convolutional neural networks. ICWSM, 632–635.

Nie, D., Zhang, H., Adeli, E., Liu, L., & Shen, D. (2016). 3d deep learning for multi-modal imaging-guided survival time prediction of brain tumor patients. *Proc. MICCAI*, 212–220. 10.1007/978-3-319-46723-8_25

Nie, L., Wang, M., Zhang, L., Yan, S., Zhang, B., & Chua, T. S. (2015, August). Disease inference from health-related questions via sparse deep learning. *IEEE Transactions on Knowledge and Data Engineering*, *27*(8), 2107–2119. doi:10.1109/TKDE.2015.2399298

O'Shea, Erpek, & Clancy. (2017). *Deep learning based MIMO communications*. arXiv preprint arXiv:1707.07980

O'Shea, Hitefield, & Corgan. (2016). End-to-end radio traffic sequence recognition with recurrent neural networks. *Proc. IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 277–281.

O'Shea, Karra, & Clancy. (2016). Learning to communicate: Channel auto-encoders, domain specific regularizers, and attention. *Proc. IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, 223–228.

Oda, T., Obukata, R., Ikeda, M., Barolli, L., & Takizawa, M. (2017). Design and implementation of a simulation system based on deep Q-network for mobile actor node control in wireless sensor and actor networks. *Proc. 31st IEEE International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, 195–200. 10.1109/WAINA.2017.67

Oh, Jeon, Jung, & Youn. (2016). *Smart diagnosis of journal bearing rotor systems: Unsupervised feature extraction scheme by deep learning*. Academic Press.

Ong, B. T., Sugiura, K., & Zettsu, K. (2015). Dynamically pre-trained deep recurrent neural networks using environmental monitoring data for predicting pm2. 5. *Neural Computing & Applications*, *27*, 1–14.

Ong, B. T., Sugiura, K., & Zettsu, K. (2016). Dynamically pre-trained deep recurrent neural networks using environmental monitoring data for predicting PM2. 5. *Neural Computing & Applications*, *27*(6), 1553–1566. doi:10.100700521-015-1955-3

Ortiz, A. M., Hussein, D., Park, S., Han, S. N., & Crespi, N. (2014, June). The cluster between Internet of Things and social networks: Review and research challenges. *IEEE Internet Things J.*, *1*(3), 206–215. doi:10.1109/JIOT.2014.2318835

Pan, S. J., & Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359. doi:10.1109/TKDE.2009.191

Paulo Victor, R. (2018). Multi-objective Reinforcement Learning for Cognitive Satellite Communications using Deep Neural Network Ensembles. *IEEE Journal on Selected Areas in Communications*.

Pereira, D. R., Papa, J. P., Saraiva, G. F. R., & Souza, G. M. (2018). Automatic classification of plant electrophysiological responses to environmental stimuli using machine learning and interval arithmetic. *Computers and Electronics in Agriculture*, *145*, 35–42. doi:10.1016/j.compag.2017.12.024

Perumal, K., & Manohar, M. (2017). *A Survey on Internet of Things: Case Studies, Applications, and Future Directions*. Cham: Springer International Publishing.

Phan, N., Dou, D., Piniewski, B., & Kil, D. (2015). Social restricted Boltzmann machine: Human behavior prediction in health social networks. *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, 424–431. 10.1145/2808797.2809307

Poggi, M., & Mattoccia, S. (2016). A wearable mobility aid for the visually impaired based on embedded 3d vision and deep learning. *Proceedings - IEEE Symposium on Computers and Communications*, 208–213. doi:10.1109/ISCC.2016.7543741

Pouladzadeh, P., Kuhad, P., Peddi, S. V. B., Yassine, A., & Shirmohammadi, S. (2016). Food calorie measurement using deep learning neural network. *Proc. IEEE Int. Instrum. Meas. Technol. Conf. Proc.*, 1–6. 10.1109/I2MTC.2016.7520547

Pouladzadeh, P., Kuhad, P., Sri, V. B. P., Yassine, A., & Shirmohammadi, S. (2016). Food calorie measurement using deep learning neural network. *IEEE International Instrumentation and Measurement Technology Conference Proceedings (I2MTC)*, 1–6. 10.1109/I2MTC.2016.7520547

Pound, M. P. (2016). Deep Machine Learning provides state-of-the-art performance in Punjani. *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, 3223-3230.

Putin, E. (2016). Deep biomarkers of human aging: Application of deep neural networks to biomarker development. *Aging (Albany NY)*, *8*(5), 1–021.

Qi, Z., Wang, T., Song, G., Hu, W., Li, X., & Zhang, Z. M. (2018). Deep air learning: Interpolation, prediction, and feature analysis of fine-grained air quality. *IEEE Transactions on Knowledge and Data Engineering*, *30*(12), 2285–2297. doi:10.1109/TKDE.2018.2823740

Qi, Yi, Su, & Guibas. (2017). PointNet++: Deep hierarchical feature learning on point sets in a metric space. Advances in Neural Information Processing Systems, 5099–5108.

Quang, D., Chen, Y., & Xie, X. (2014). Dann: A deep learning approach for annotating the pathogenicity of genetic variants. *Bioinformatics (Oxford, England)*, *31*(5), 761–763. doi:10.1093/bioinformatics/btu703

Rahnemoonfar, M., & Sheppard, C. (2017). Deep count: Fruit counting based on deep simulated learning. *Sensors (Basel)*, *17*(4), 905. doi:10.339017040905

Rajendran, S. (2018). *Deep learning models for wireless signal classification with distributed low-cost spectrum sensors*. *IEEE Transactions on Cognitive Communications and Networking*.

Ramsundar, Kearnes, Riley, Webster, Konerding, & Pande. (2015). *Massively multitask networks for drug discovery*. ArXiv e-prints.

Ravi, D., Wong, C., Lo, B., & Yang, G. Z. (2016). Deep learning for human activity recognition: A resource efficient implementation on low-power devices. *Proc. 13th Int.Conf.Wearable Implantable Body Sens.Netw.*, 71–76. 10.1109/BSN.2016.7516235

Ray, S., Jin, Y., & Raychowdhury, A. (2016). The changing computing paradigm with internet of things: A tutorial introduction. *IEEE Design & Test*, *33*(2), 76–96. doi:10.1109/MDAT.2016.2526612

Rebetez, J., Satiza'Bal, H. F., & Mota, M. (2016). Augmenting a convolutional neural network with local histograms - A case study in crop classification from high-resolution UAV imagery. *ESANN 2016 proceedings*, *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*.

Reddy, Sarkar, & Giering. (2016). *Anomaly detection and fault disambiguation in large flight data: A multi-modaldeep auto-encoder approach*. In Annual conference of the prognostics and health management society, Denver, CO.

Ren, Y., & Wu, Y. (2014). Convolutional deep belief networks for feature extraction of EEG signal. In *Neural Networks (IJCNN)*, *2014 International Joint Conference on*, (pp. 2850–2853). IEEE. 10.1109/IJCNN.2014.6889383

Robertson, S. E., & Walker, S. (1994). Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 232–241. 10.1007/978-1-4471-2099-5_24

Roth, H. R. (2015). Anatomy-specific classification of medical images using deep convolutional nets. *Proc. IEEE Int. Symp. Biomed. Imag.*, 101–104. 10.1109/ISBI.2015.7163826

Ryder, B., & Wortmann, F. (2017). Autonomously detecting and classifying traffic accident hotspots. *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and 2017 ACM International Symposium on Wearable Computers*, 365–370. 10.1145/3123024.3123199

Salakhutdinov, R., & Hinton, G. (2009) Semantic hashing. *Int J Approximate, Reasoning, 50*(7), 969–978. 10.1016/j.ijar.2008.11.006

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, *24*(5), 513–523. doi:10.1016/0306-4573(88)90021-0

Santoni, M. M., Sensuse, D. I., Arymurthy, A. M., & Fanany, M. I. (2015). Cattle race classification using gray level co-occurrence matrix convolutional neural networks. *Procedia Computer Science*, *59*, 493–502. doi:10.1016/j.procs.2015.07.525

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). *Proximal policy optimization algorithms.* arXiv preprint arXiv:1707.06347

Sedjelmaci, H., Senouci, S. M., & Al-Bahri, M. (2016). A lightweight anomaly detection technique for low-resource iot devices: A game-theoretic methodology. *IEEE International Conference on Communications (ICC)*, 1–6. 10.1109/ICC.2016.7510811

Sehal, Gupta, & Paneri. (2017). *Crop planning using stochastic visual optimization*. Academic Press.

Serpanos, D. (2018). The Cyber-Physical Systems Revolution. *Computer*, *51*(3), 70–73. doi:10.1109/MC.2018.1731058

Shalev-Shwartz, S. (2012). Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, *4*(2), 107–194. doi:10.1561/2200000018

Shamsabadi, A. S., Haddadi, H., & Cavallaro, A. (2018). Distributed one-class learning. *Proc. IEEE International Conference on Image Processing (ICIP).*

Shan, J., & Li, L. (2016). A deep learning method for microaneurysm detection in fundus images. *Proc. IEEE Connected Health, Appl., Syst. Eng. Technol.*, 357–358. 10.1109/CHASE.2016.12

Shan, J., & Li, L. (2016). A deep learning method for microaneurysm detection in fundus images. *Proc. IEEE Connected Health, Appl., Syst. Eng. Technol*., 357–358. 10.1109/CHASE.2016.12

Shen, Han, Yang, Yang, Wang, Li, & Wen. (2018). CS-CNN: Enabling robust and efficient Convolutional neural networks inference for internet-of-things applications. *IEEE Access, 6*(13), 439–448. doi:10.1109/access.2018.2810264

Shin, H., Lu, L., Kim, L., Seff, A., Yao, J., & Summers, R. M. (2015). *Interleaved text/image deep mining on a large-scale radiology database for automated image interpretation*. Available: https://arxiv.org/abs/1505.00670

Shin, H.-C., Roth, H. R., Gao, M., Lu, L., Xu, Z., Nogues, I., ... Summers, R. M. (2016, May). Deep convolutional neural networks for computeraided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE Transactions on Medical Imaging*, *35*(5), 1285–1298. doi:10.1109/TMI.2016.2528162

Shiraz, M., Gani, A., Khokhar, R., Rahman, A., Iftikhar, M., & Chilamkurti, N. (2017). A distributed and elastic application processing model for mobile cloud computing. *Wirel Pers Commun, 95*(4), 4403–4423.

Simonyan, K., & Zisserman, A. (2014). Very deep Convolutional networks for large-scale image recognition. *International Conference on Learning Representations (ICLR 2015).*

Smartpharma.co.za. (2015). *Welcome to Smart Pharmaceuticals*. Available at: http://www.smartpharma.co.za/

Socher, R., Lin, C. C., Ng, A., & Manning, C. (2011). Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning*. Omnipress.

Socher, R., & Perelygin, A. (2013). *Recursive deep models for semantic compositionality over a sentiment tree bank*. Emnlp.

Song, X., Zhang, G., Liu, F., Li, D., Zhao, Y., & Yang, J. (2016). Modeling spatio-temporal distribution of soil moisture by deep learning-based cellular automata model. *Journal of Arid Land*, *8*(5), 734–748. doi:10.100740333-016-0049-0

Sperduit, A., & Starita, A. (1997). Supervisedneurl networks for the classification of structures. *IEEE Trasctions on Neural Network, 8*(3).

Suk, H.-I., Lee, S.-W., & Shen, D. (2014). Hierarchical feature representation and multimodal fusion with deep learning for ad/mci diagnosis. *NeuroImage*, *101*, 569–582. doi:10.1016/j.neuroimage.2014.06.077

Suk, H.-I., Lee, S.-W., & Shen, D. (2015). Latent feature representation with stacked auto-encoder for AD/MCI diagnosis. *Brain Structure & Function*, *220*(2), 841–859. doi:10.100700429-013-0687-3

Sun, L., Jia, K., Chan, T.-H., Fang, Y., Wang, G., & Yan, S. (2014). DL-SFA: Deeply-learned slow feature analysis for action recognition. *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2625–2632. 10.1109/CVPR.2014.336

Sun, W., Shao, S., Zhao, R., Yan, R., Zhang, X., & Chen, X. (2016). A sparse auto-encoder-based deep neural network approach for induction motor faults classification. *Measurement*, *89*, 171–178. doi:10.1016/j.measurement.2016.04.007

Sun, Y., Wu, L., Wu, S., Li, S., Zhang, T., Zhang, L., ... Cui, X. (2017, June). Attacks and countermeasures in the internet of vehicles. *Annals of Telecommunications, 72*(5), 283–295 doi:10.100712243-016-0551-6

Sutskever, O. (2014). Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 3104–3112.

Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., & Liang, J. (2016, May). Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, *35*(5), 1299–1312. doi:10.1109/TMI.2016.2535302

Tan, J., & Koo, S. G. M. (2014). A Survey of Technologies in Internet of Things. *IEEE International Conference on Distributed Computing in Sensor Systems*, 269-274.

Tan, L., & Wang, N. (2010). Future Internet: The Internet of Things. *3rd International Conference On Advanced Computer Theory And Engineering (ICACTE),* 376-380.

Tan, Q. (2017). Emerging Technologies: IoT, Big Data, and CPS with Sensory Systems. *Hindawi Journal of Sensors*, *2018*, 3407542. doi:10.1155/2018/3407542

Tao, S., Zhang, T., Yang, J., Wang, X., & Lu, W. (2015). Bearing fault diagnosis method based on stacked autoencoder and softmax regression. *Control Conference (CCC)*, *2015 34th Chinese*, 6331–6335. 10.1109/ChiCC.2015.7260634

Tao, W. R. D. (2016). Non-local auto-encoder with collaborative stabilization for image restoration. *IEEE Transactions on Image Processing*, *25*(5), 2117–2129. doi:10.1109/TIP.2016.2541318

Thirukovalluru, R., Dixit, S., Sevakula, R. K., Verma, N. K., & Salour, A. (2016). Generating feature sets for fault diagnosis using denoising stacked auto-encoder. In *Prognostics and Health Management (ICPHM), 2016 IEEE International Conference on*. IEEE. https://towardsdatascience.com/a-comphrehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a5

Tian, K., Shao, M., Zhou, S., & Guan, J. (2015). Boosting compound-protein interaction prediction by deep learning. *Proc. IEEE Int. Conf. Bioinformat. Biomed.*, 29–34.

Turaga, S. C., Murray, J. F., Jain, V., Roth, F., Helmstaedter, M., Briggman, K., ... Seung, H. S. (2010). Convolutional networks can learn to generate affinity graphs for image segmentation. *Neural Computation*, *22*(2), 511–538. doi:10.1162/neco.2009.10-08-881

Valipour, S., Siam, M., Stroulia, E., & Jagersand, M. (2016). Parking-stall vacancy indicator system, based on deep convolutional neural networks. Internet of Things, 655–660. doi:10.1109/WF-IoT.2016.7845408 van Grinsven, M. J., van Ginneken, B., Hoyng, C. B., Theelen, T., & S'anchez, C. I. (2016, May). Fast convolutional neural network training using selective data sampling: Application to hemorrhage detection in color fundus images. *IEEE Transactions on Medical Imaging*, *35*(5), 1273–1284. doi:10.1109/TMI.2016.2526689

Verma, N. K., Gupta, V. K., Sharma, M., & Sevakula, R. K. (2013). Intelligent condition based monitoring of rotating machines using sparse autoencoders. *Prognostics and Health Management (PHM)*, *2013 IEEE Conference on*, 1–7.

Vetrivel, A., Gerke, M., Kerle, N., Nex, F., & Vosselman, G. (2018). Disaster damage detection through synergistic use of deep learning and 3D point cloud features derived from very high resolution oblique aerial images, and multiple-kernel-learning. *ISPRS Journal of Photogrammetry and Remote Sensing*, *140*, 45–59. doi:10.1016/j.isprsjprs.2017.03.001

Vlacheas, P., Giaffreda, R., Stavroulaki, V., Kelaidonis, D., Foteinos, V., Poulios, G., ... Moessner, K. (2013, June). Enabling smart cities through a cognitive management framework for the Internet of Things. *IEEE Communications Magazine*, *51*(6), 102–111. doi:10.1109/MCOM.2013.6525602

Wang, A., Song, C., Xu, X., Lin, F., Jin, Z., & Xu, W. (2015). Selective and compressive sensing for energy-efficient implantable neural decoding. *Proc. Biomed. Circuits Syst. Conf.*, 1–4. 10.1109/BioCAS.2015.7348375

Wang, B., Gu, X., Ma, L., & Yan, S. (2017). Temperature error correction based on BP neural network in meteorological wireless sensor network. *International Journal of Sensor Networks*, *23*(4), 265–278. doi:10.1504/IJSNET.2017.083532

Wang, J., Tang, J., Xu, Z., Wang, Y., Xue, G., Zhang, X., & Yang, D. (2017). Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach. *Proc. 36th Annual IEEE International Conference on Computer Communications (INFOCOM).* 10.1109/INFOCOM.2017.8057090

Wang, L., Zhao, X., Pei, J., & Tang, G. (2016). Transformer fault diagnosis using continuous sparse autoencoder. *SpringerPlus*, *5*(1), 1. doi:10.118640064-016-2107-7

Wang, X., Gao, L., & Mao, S. (2015). PhaseFi: Phase fingerprinting for indoor localization with a deep learning approach. *Proc. IEEE Global Communications Conference (GLOBECOM)*, 1–6. 10.1109/GLOCOM.2015.7417517

Wang, X., Wang, X., & Mao, S. (2017). CiFi: Deep Convolutional neural networks for indoor localization with 5 GHz Wi-Fi. *Proc. IEEE International Conference on Communications (ICC)*, 1–6. 10.1109/ICC.2017.7997235

Wang, X., Zhou, Z., Xiao, F., Xing, K., Yang, Z., Liu, Y., & Peng, C. (2018). Spatio-temporal analysis and prediction of cellular traffic in metropolis. *IEEE Transactions on Mobile Computing*.

West & O'Shea. (2017). Deep architectures for modulation recognition. *Proc. IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 1–6.

Wulsin, D., Blanco, J., Mani, R., & Litt, B. (2010). Semi-supervised anomaly detection for eeg waveforms using deep belief nets. *Proc. 9th Int. Conf. Mach. Learn. Appl*., 436–441. www.nytimes.com/2001/07/12/technology/news-watch-a-quick-way-to-search-for-images-on-the-web.html

Xi, O., Zhang, C., Zhou, P., & Jiang, H. (2016). *DeepSpace: An online deep learning framework for mobile big data to understand human mobility patterns*. arXiv preprint arXiv:1610.07009

Xin Shao, W., & Cheng, C. (2015). Weed seeds classification based on PCANet deep learning baseline. *IEEE Signal and Information Processing Association Annual Summit and Conference (APSIPA),* 408–415.

Xu, T., Zhang, H., Huang, X., Zhang, S., & Metaxas, D. N. (2016). Multimodal deep learning for cervical dysplasia diagnosis. *Proc. MICCAI*, 115–123. 10.1007/978-3-319-46723-8_14

Yalc¸ın, H. (2016). Human activity recognition using deep belief networks. *Proc. Signal Process. Commun. Appl. Conf.*, 1649–1652. 10.1109/SIU.2016.7496073

Yan, Y., Qin, X., Wu, Y., Zhang, N., Fan, J., & Wang, L. (2015). A restricted Boltzmann machine based two-lead electrocardiography classification. *Proc. 12th Int. Conf. Wearable Implantable Body Sens. Netw.*, 1–9.

Yan, Z., Zhan, Y., Peng, Z., Liao, S., Shinagawa, Y., Zhang, S., ... Zhou, X. S. (2016, May). Multi-instance deep learning: Discover discriminative local anatomies for bodypart recognition. *IEEE Transactions on Medical Imaging*, *35*(5), 1332–1343. doi:10.1109/TMI.2016.2524985

Yan, Z., Zhang, P., & Vasilakos, A. V. (2014). A survey on trust management for Internet of Things. *Journal of Network and Computer Applications*, *42*, 120–134. doi:10.1016/j.jnca.2014.01.014

Yang, K., & Zhang, Z. (2012). Summarize on IoT and exploration into technical system framework. *Proc. IEEE Symp. Robot. Appl. (ISRA),* 653–656.

Yang, Y., Li, Y., & Fermüller, C. (2015). Robot learning manipulation action plans by watching unconstrained videos from the world wide web. *29th AAAI Conference on Artificial Intelligence (AAAI-15)*, Austin, TX.

Yao, S., Hu, S., Zhao, Y., Zhang, A., & Abdelzaher, T. (2017). Deepsense: A unified deep learning framework for timeseries mobile sensing data processing. In *Proc. 26th International Conference on World Wide Web*, (pp. 351–360). International World Wide Web Conferences Steering Committee. 10.1145/3038912.3052577

Ye, Li, Juang, & Sivanesan. (2018). *Channel agnostic end-to-end learning based communication systems with conditional GAN*. arXiv preprint arXiv:1807.00447

Ye, H., Li, G. Y., & Juang, B.-H. (2018). Power of deep learning for channel estimation and signal detection in OFDM systems. *IEEE Wireless Communications Letters*, *7*(1), 114–117. doi:10.1109/LWC.2017.2757490

Yi, X., Zhang, J., Wang, Z., Li, T., & Zheng, Y. (2018). Deep distributed fusion network for air quality prediction. *Proceedings of the 24th SIGKDD conference on Knowledge Discovery and Data Mining*. 10.1145/3219819.3219822

Yin, S., Li, X., Gao, H., & Kaynak, O. (2015, January). Data-based techniques focused on modern industry: An overview. *IEEE Transactions on Industrial Electronics*, *62*(1), 657–667. doi:10.1109/TIE.2014.2308133

Yousefi-Azar, M., Varadharajan, V., Hamey, L., & Tupakula, U. (2017). Autoencoder-based feature learning for cyber security applications. *Proc. IEEE International Joint Conference on Neural Networks (IJCNN)*, 3854–3861. 10.1109/IJCNN.2017.7966342

Yu, J.-s., Chen, J., & Xiang, Z. (2015). A hybrid convolutional neural networks with extreme learning machine for WCE image classification. *Proc. IEEE Robot. Biomimetics*, 1822–1827.

Yu, J., Weng, K., & Liang, G. (2013). A vision-based robotic grasping system using deep learning for 3D object recognition and pose estimation. *Robotics and Biomimetics (ROBIO), 2013 IEEE International Conference on*, 1175-1180.

Yuan, M., Wu, Y., & Lin, L. (2016). Fault diagnosis and remaining useful life estimation of aero engine using lstm neural network. *2016 IEEE International Conference on Aircraft Utility Systems (AUS)*, 135–140. 10.1109/AUS.2016.7748035

Zanella, A., Bui, N., Castellani, A., Vangelista, L., & Zorzi, M. (2014, February). Internet of Things for smart cities. *IEEE Internet Things J.*, *1*(1), 22–32. doi:10.1109/JIOT.2014.2306328

Zantalis, Koulouras, Karabetsos, & Kandris. (2019). A Review of Machine Learning and IoT in Smart Transportation. *Future Internet, 11*(94). www.mdpi.com/journal/futureinterne

Zeng. (2014). Convolutional neural networks for human activity recognition using mobile sensors. *Proc. MobiCASE*, 197–205. .mobicase.2014.257786 doi:10.4108/icst

Zeng & Liu. (2008). Markov random field-based statistical character structure modeling for handwritten Chinese character recognition. *On pattern analysis and machine intelligence, IEEE Trans., 30*(5).

Zhang, C., Lim, P., Qin, A., & Tan, K. C. (2016). Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. *IEEE Transactions on Neural Networks and Learning Systems*.

Zhang, C., Sun, J. H., & Tan, K. C. (2015). Deep belief networks ensemble with multi-objective optimization for failure diagnosis. In *Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on*. IEEE. 10.1109/SMC.2015.19

Zhang, J., Zheng, Y., & Qi, D. (2017). Deep spatio-temporal residual networks for citywide crowd flows prediction. AAAI, 1655–1661.

Zhang, S., Zhou, J., Hu, H., Gong, H., Chen, L., Cheng, C., & Zeng, J. (2016). A deep learning framework for modeling structural features of rna-binding protein targets. *Nucleic Acids Research*, *44*(4), e32–e32. doi:10.1093/nar/gkv1025

Zhang, T., Antunes, H., & Aggarwal, S. (2014, February). Defending connected vehicles against malware: Challenges and a solution framework. *IEEE Internet Things J.*, *1*(1), 10–21. doi:10.1109/JIOT.2014.2302386

Zhao, L., Chen, J., Chen, F., Wang, W., Lu, C.-T., & Ramakrishnan, N. (2015). Simnest: Social media nested epidemic simulation via online semisupervised deep learning. *Proceedings - IEEE International Conference on Data Mining*, 639–648. doi:10.1109/ICDM.2015.39

Zhao, R., Wang, J., Yan, R., & Mao, K. (2016). Machine helath monitoring with lstm networks. In *2016 10th International Conference on Sensing Technology (ICST).* IEEE.

Zhen, X., Wang, Z., Islam, A., Bhaduri, M., Chan, I., & Li, S. (2016). Multi-scale deep networks and regression forests for direct bi-ventricular volume estimation. *Medical Image Analysis*, *30*, 120–129. doi:10.1016/j.media.2015.07.003

Zipern, A. (2001). A Quick Way to Search For Images on the Web. *The New York Times*.

Zou, B., Lampos, V., Gorton, R., & Cox, I. J. (2016). On infectious intestinal disease surveillance using social media content. *Proc.6th Int. Conf. Digit. Health Conf.*, 157–161. 10.1145/2896338.2896372

Chapter 2
Deep Learning Architectures and Tools:
A Comprehensive Survey

K. Bhargavi
Siddaganga Institute of Technology, India

ABSTRACT

Deep learning is one of the popular machine learning strategies that learns in a supervised or unsupervised manner by forming a cascade of multiple layers of non-linear processing units. It is inspired by the way of information processing and communication pattern of the typical biological nervous system. The deep learning algorithms learn through multiple levels of abstractions and hierarchy of concepts; as a result, it is found to be more efficient than the conventional non-deep machine learning algorithms. This chapter explains the basics of deep learning by highlighting the necessity of deep learning over non-deep learning. It also covers discussion on several recently developed deep learning architectures and popular tools available in market for deep learning, which includes Tensorflow, PyTorch, Keras, Caffe, Deeplearning4j, Pylearn2, Theano, CuDDN, CUDA-Convnet, and Matlab.

INTRODUCTION

Most of the companies are trending towards deep learning to solve most complex problems in an efficient way by training the network sufficient labeled or unlabelled data samples (Pulkit 2018; Angelov, and Sperduti 2016). It has become one of the important in digitalization era due to the following characteristics.

- Stronger synaptic connection among the neurons
- Lower dependencies on the input data samples
- Able to handle both labeled and unlabelled data samples.
- Used to train the machines to make cognitively intelligent.
- Explores the power of depth in machine learning.
- Fast training as it deals with fewer parameters.

DOI: 10.4018/978-1-7998-2108-3.ch002

- Distributed representation can handle higher dimensions easily.

Some of the applications of deep learning include coloring black and white images, generation of image captions, automatic game playing, audio recognition, designing drug, filtering social network data, inspection of composite materials, giving sound effects for silent movies, diagnosis of medical images, handwriting recognition and so on (Bhargavi, and Babu 2017; Deng, and Yu 2014).).

However deep learning involves several challenges (Chen, and Lin 2014; Falcini, and Lami 2017), some of them are listed below.

- The learning speed of the deep learning algorithms is very slow.
- The desired accuracy is achieved only when the algorithm is trained with more number of training samples.
- More number of input parameters is required for better tuning of the deep learning network.
- The efficiency of the model is trained by the ability of the model to perform better over the unseen data samples.
- Hyper parameters are sensitive towards the outliers.
- To train and test the deep learning algorithms the hardware devices must have high computational ability.
- The power consumption rate of the multi core GPUs used for training deep learning algorithms is high.
- The deep learning algorithms lacks flexibility as it does not provide precise solution to specific problems.
- The rate of retraining and retuning of the models is more even when small changes happen in the type of application.
- The noise in the training data causes overfitting problem which declines the performance of the network towards real world applications.
- The deep learning models cannot be generalized easily outside the training space of the problem.

In literature several works are available on deep learning tools and steps to install the tools for working is covered which includes hands-on sessions information about the deep learning tools, visualization tool available for machine learning, Python packages frequently used for deep learning, popular tools for modeling artificial intelligence components, and so on. However the works carried out have limitations as it does not discusses in-depth about the deep learning architectures and the deep learning tools. And the existing works concentrate more on the installation procedure to use the tools and advantages and disadvantages of each of the tools are not mentioned clearly. So in this chapter the basics of deep learning, comparison between deep learning and non-deep learning techniques, deep learning tools along with their advantages and disadvantages are discussed.

DEEP LEARNING VERSUS NON-DEEP LEARNING

A comparison between deep learning and non-deep learning is given in Table 1.

ARCHITECTURES OF DEEP LEARNING MODELS

Some of the potential architectures of deep learning include deep neural network, deep recurrent neural network, deep belief network, and deep convolutional neural network (Zhang, Yang, Chen, and Li 2018; Fei-Fei, Justin, and Serena 2017).

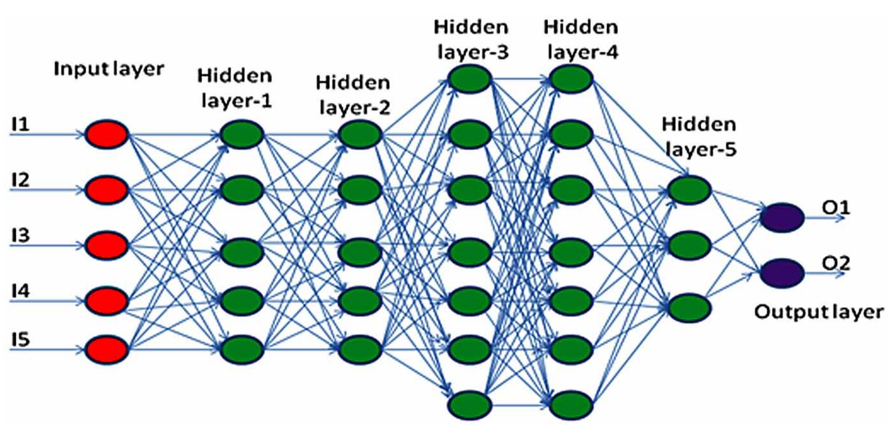
Table 1. Comparison between deep learning and non-deep learning techniques

Deep Learning	Non-Deep Learning
The deep learning technique achieves high accuracy while solving computation intensive applications like speech processing, video games, computer vision, natural language processing and so on.	Accuracy achieved by non-deep learning is poor especially in computation intensive applications like speech processing, video games, computer vision, natural language processing and so on.
Scale factor of deep learning algorithms is extremely high with the increase in the data.	The scale factor of the deep learning algorithms is extremely low with the increase in the data.
The deep learning does not demand for complex feature engineering for explorative analysis of the data.	The non-deep learning always demands for complex feature engineering for explorative analysis of the data.
The deep learning techniques quickly get adapted to varieties of applications.	The non-deep learning techniques cannot get adapted to varieties of applications.
The deep learning algorithms can be ported easily from one platform to another.	The non-deep learning algorithms cannot be easily ported from one platform to another.
The performance is better on both small and large data	The performance is better only on small data.
It has ability to produce high quality results.	It does not have ability to produce high quality results.

Deep Neural Network

It follows the architecture of neural network by adding more complexity as more number of layers is added between input and output layers. The network exhibit feature hierarchy as every neuron in the layers performs some form of ordering or sorting functions. The network basically works on trial and

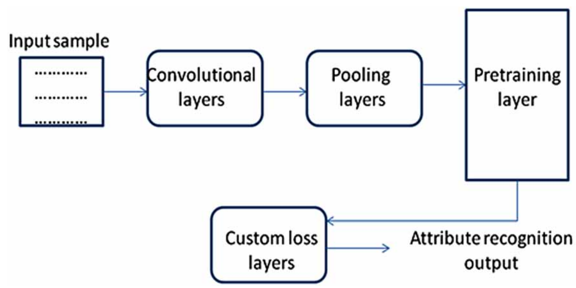
*Figure 1. 5*5*5*7*7*3*2 deep neural network*



error method as a result it needs sufficient datasets to train the network. Nowadays deep learning is being used in many fields like coloring black and white images, generation of image captions, automatic game playing, audio recognition, designing drug, filtering social network data, inspection of composite materials, giving sound effects for silent movies, diagnosis of medical images, handwriting recognition and so on. A sample model of deep neural network of size 5*5*5*7*7*3*2 is shown in figure 1.

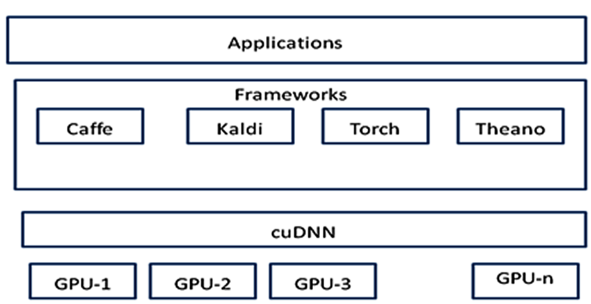
Deep Recurrent Neural Network

Figure 2. Three layered deep recurrent neural network



It is a powerful model that works on sequential form of data and has the ability to handle the data of any length. There are several types of deep recurrent network which includes bidirectional deep recurrent neural network (BRNN), Long-Short term memory (LSTM), and Gated Recurrent Unit (GRU). BRNN establishes connection between two opposite sided hidden layers to a single output unit as a result it can get information from past and present information simultaneously. LSTM contains a cell, input gate, output gate and a forget gate, the network tries to learn by remembering the values over the stipulated time intervals. The GRU is similar to LSTM but it lacks output gate and uses only few parameters compared to LSTM for training the network. A three layered deep recurrent neural network is given in figure 2.

Figure 3. Deep belief network

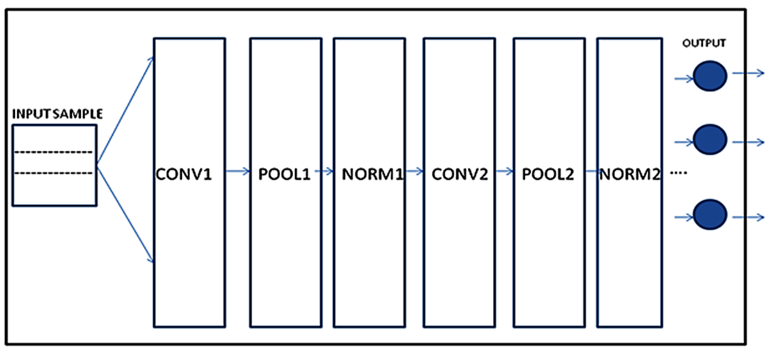


Deep Belief Network

It consists of several layers of hidden units which exhibits connection from one layer to another layer but not within the same layer. It is also referred as a stack of restricted Boltzmann machine in which neurons in each layer communicates with the neurons in the previous layers and subsequent layers. It basically uses unsupervised machine learning model to train and process the input and generate the output. It is commonly used in image processing for recognize the image, cluster the images, and meanwhile handle the images captured through motion capture data. A sample deep belief network is given in figure 3.

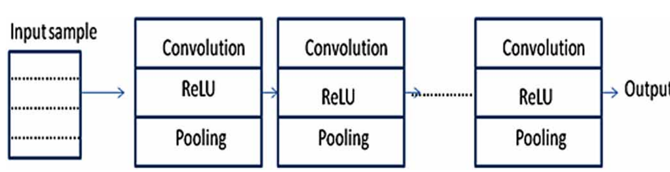
Deep Convolutional Neural Network

Figure 4. Deep convolutional neural network



It is also a form neural network used commonly for image analysis which inputs the image processes it by assigning priority to various aspects of the image to provide desired result. There are various applications of deep convolutional neural network like recognition of objects in the videos, image recognition, image classification, handwriting recognition, recommendation systems and so on. It is preferred over conventional feed forward neural network as it is able to capture the spatial and temporal dependencies in the input image samples with the help of filters. A sample deep convolutional neural network is given in figure 4.

Figure 5. Deep LSTM network



Deep Long Short Term Memory (LSTM) Architecture

The deep LSTM architecture consists of many hidden layers of LSTM compared to single LSTM architecture as it is composed of only one hidden layer of LSTM. It combines the learned representation of previous layers to output new representation with higher accuracy. By significantly scaling the number of hidden layers several alternate solutions can be obtained with very few neurons at a faster rate. Sample two layered LSTM architecture is given in figure 5.

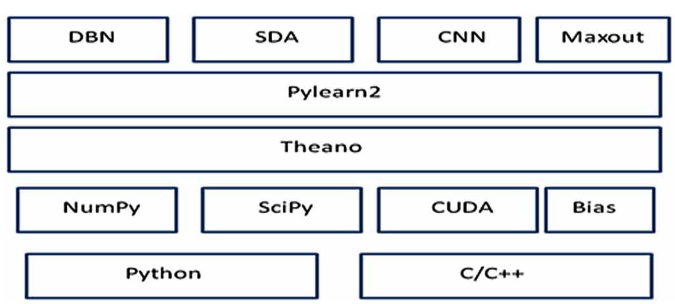
There are several tools available in market for deep learning that allows the developer to build intelligent machine learning based systems. Some of the promising tools for deep learning are discussed as follows.

GOOGLE TENSORFLOW

It is a free open source tool which offers APIs for languagues like Python, and C++ which is used for differential programming across varieties of incoming tasks. It is composed of two components one is graph protocol based buffer and the other is runtime distributed graph (Abadi, Barham, Chen, Chen, Davis, Dean, and et al. 2016; Edward 2017). The main characteristics of Google TensorFlow are given below.

- It provides an easy building model which can be trained easily using high-level APIs like Keras which supports easy debugging and early execution.
- Robust machine learning models is built easily anywhere as it is totally platform independent.
- Flexible architecture is supported to import code and carry out research experimentation work easily.
- It encourages the researchers to easily build and deploy machine learning powered applications.
- Tensors act as multidimensional arrays useful for mathematical computation.
- The TensorFlow toolkit consists of hierarchy of layers which consists of kernel in the base layer, on top of it reusable libraries and high-level object oriented API of Tensorflow is placed.

Figure 6. Google TensorFlow architecture



The architecture of the Google TensorFlow is given in Figure 6. The client determines the computation as a flow graph and initiates the session in terms graph execution period. The distributed master and dataflow executor partitions the graphs into sub-graphs and distributes it among different workers for services. The worker then schedules the execution of sub-graph on appropriate hardware like CPU, GPU, and so on.

Advantages:

- It works efficiently on research related to perceptual and language understanding tasks.
- Capable of running on both CPU and GPU and can be easily deployed on broad range Google products.
- It is able to perform numerical computations via data flow graphs.
- Easy to deploy applications as it is backed by Google community consisting of skilled employers.
- Scalability factor is good as it allows the libraries to be deployed on hardware machines starting from cellular devices to complex systems.
- It is parallel in nature, as it allows various software to run as daemon processes.

Disadvantages:

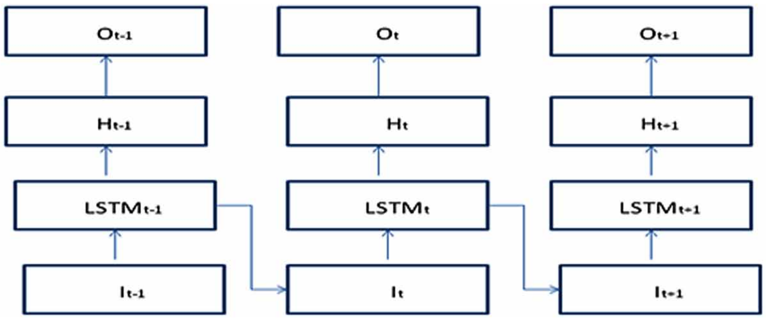
- It does not support symbolic loops to handle variable length sequences.
- The support is good for Linux environment but not good for Windows.
- Lacks behind in terms of speed and accuracy compared to other deep learning framework like CuDNN, Nervana, and so on.
- It is being supported only by NVIDIA GPUs, no other GPU support.
- In terms of computation speed, TensorFlow is lagging compared to other deep learning frameworks like Torch7, Caffee, and Theano.

PYTORCH

It is a open source Python based machine learning library developed by Facebook machine learning based research group and Ubers Pyro software, which is based on Torch usually used to extensively build natural processing based applications (Ketkar 2017; Paszke, Gross, Chintala, Chanan, Yang, DeVito, and et al. 2017). The main characteristics of PyTorch are given below.

- Supports Tensorflow based computation by exploiting the benefit of strong GPU acceleration.
- Deep neural network are constructed on top of autodiff system.
- It supports various types of Tensors like TensorFlow, Theano, Torch and ONIX.
- The PyTorch is composed of modules like optimum, autograd, and neural network.
- Prgramming in PyTorch can be started component wise, no need to wait for whole code to be written.
- The workflow of PyTorch is easy to understand as the developers can get support from Python scientific numpy library committee.
- Well built library is available as the interfaces can be specified in dataset, a sample or a data loader.
- Tensors in PyTorch act as NumPy array.

Figure 7. Internal architecture and element-wise filter of PyTorch



The internal architecture and internal architecture and element-wise filter of the PyTorch is given in figure 7. The input sample is fed as input which will be entered by initializing value to filters to provide desired output.

Advantages

- Ease of distribution of computational work among CPU or GPU cores.
- Speed is achieved via parallelism.
- Similar to NumPy to manage data intensive computation.
- Popular library as it is based on Python programming language.
- Deep integration towards graph computation provides ability to do in-depth computation.
- Training time is less compared to TensorFlow.
- Model consists of very few parameters so modeling is easy.
- Easy to develop applications as the developer need to create object oriented class and then encapsulate the data.
- Transparent for developer and data scientists.
- Easy to debug due to the availability of debugging tools like pdb, ipdb, and PyCharm.
- Provides support for implicit declarative data parallelism.
- It is developer friendly.

Disadvantages

- Visualization board is not awesome as TensorFlow.
- Lacks distributed training
- Deploying the models in PyTorch is easier.
- PyTorch developer community is nor bigger.
- Huge dataset is required to deal with complex deep learning architecture.
- Limited support for dynamic input samples.

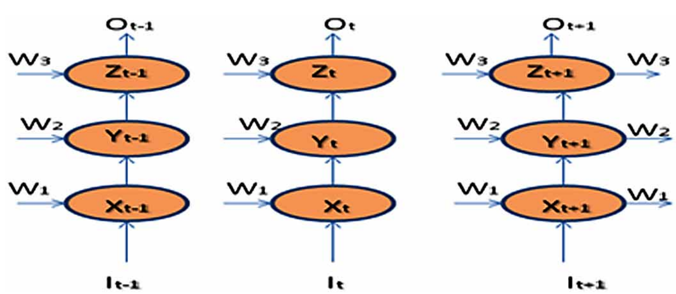
KERAS

Keras is a Python based library for neural network training and testing purpose. The library is able to run on top of TensorFlow and even Theano deep learning platforms. The library allows training the deep neural network in less time and is programmer friendly, modular and extensible in nature. Along with neural network, keras support numerous neural networks like convoluted neural network and recurrent neural network (Gulli, and Pal 2017; Martin 2018). The main characteristics of keras are given below.

- Supports distributed training of various deep learning models.
- Consists of implementations of neural network building blocks like layers, objectives, optimizers and activation functions.
- Offers high level abstraction of deep learning models.
- Capable of executing seamlessly on both CPUs and GPUs.
- It is easy to define, compile, fit and make predictions from the dataset.
- Allows quick data exploration.
- Provides clean and convenient way to develop deep learning models.

A sample keras based LSTM (Long Short Term Memory) model is given in figure 8. The LSTM networks are commonly used in prediction applications like stock price, electricity demand and so on. Mainly consists of input layer, layers of LSTM, layers of hidden layer and then the output layer. The feedback from one layer of LSTM is passed on to another layer of LSTM.

Figure 8. A sample Keras based LSTM deep learning model



Advantages

- Easy to construct multi layers neural network of various architecture.
- Critical decisions can be taken easily as it is easy to easy to add any number of hidden layers and fine tune the model.
- Provides better user experience compared to TensorFlow.
- Exhibits simplicity from the Python language.
- No need for large dataset for training.
- Less computation power is required for task completion.

- The performance of the neural network with Keras is good compared to TensorFlow.

Disadvantages

- Exhibits more dependency on lower level APIs like NumPy, Pandas, and Matplotlib.
- It is not flexible for customizable applications.
- Difficulty is involved in the deployment of the Keras model.
- Exhibits high levelness in programming.

DEEPLEARNING4J

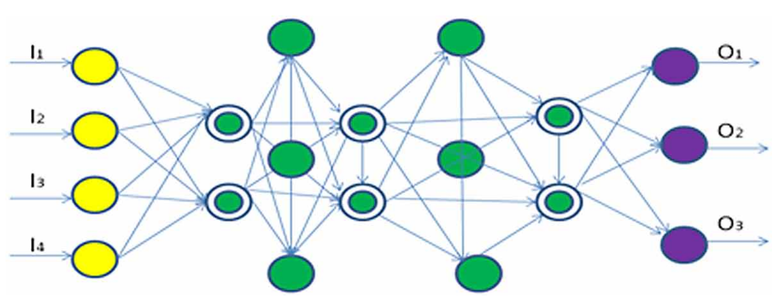
The deeplearning4j is one of the most widely java based deep learning framework. It consists of already predefined implementations of deep belief network, Boltzmann machine, and recursive neural network. The predefined implementations are supported by parallel distributed versions of Hadoop and Spark (Baeldung 2018; Erickson, Korfiatis, Akkus, Kline, and Philbrick 2017). The main characteristics of deeplearning4j are given below.

- Able to process the unstructured data samples.
- Supports the deep learning libraries written for Java and Scala programming language.
- Able to import neural network models from most frameworks like Keras, Caffe, and Theano.
- It can work on both CPU and GPU.
- Highly scalable on Amazon web services in cloud
- Computation happens in parallel to process massive amount of data.
- Ease of use since it follows micro service architecture.
- Supports reusability of packages on demand like SciPy, NumPy, scikit and Cython.

Sample deeplearning4j architecture is shown in figure 9 which mainly consists of ND4J, Canova and deeplearning4j functional components. On top of which the neural network model consisting of input layers, hidden layers and output layers is working. The ND4J consists of N-dimensional array and calculation library, Canova consists of vectorization library, and DeepLearning4j consists of neural network based configuration.

Advantages

Figure 9. deeplearning4j architecture



- The neural network computation is carried out with less overhead.
- It is easy to train the neural network with less time.
- Accelerated computation happens as it includes speed-up libraries like MKL, NVIDIA, and so on.
- GUI design is good as the performance can be viewed visually.
- Easy to interpret and debug network learning process.
- Maximum accuracy is achieves through prediction.
- Supports multi-GPU computation.
- High performance is achieved with respect to data consisting of images, video, audio, and animation related objects.
- Easily achievable multi-threaded programming.

Disadvantages

- To work with DeepLearning4j other prerequisites like JDK, Eclipse IDE, Git, and maven is required.
- Overfitting problem is high as the chances of setting high weights for training data is more.
- Lot of parameters need to be trained to adjust to the training pattern of deeplearning4j.
- Taking decisions with respect to hyper parameters setting is tedious.

CAFFE

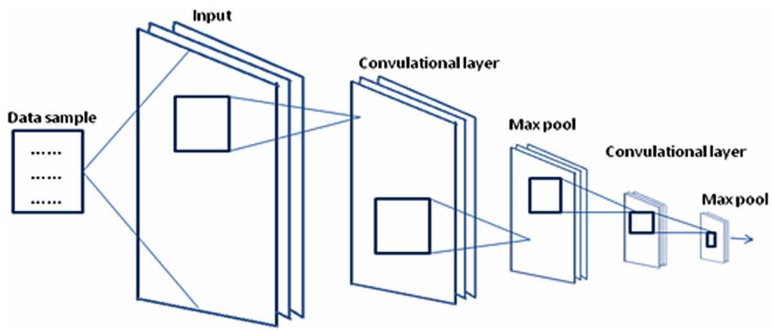
Caffe stands for Convolutional Architecture for Fast Feature Embedding, is a C++ oriented deep learning framework with Python interface developed by university of California, Berkeley. Caffe supports various types of neural network like CNN, RCNN, and LSTM, extensively used for image segmentation and classification (Peter 2017; Julien 2017). The main characteristics of Caffe are given below.

- Supports CPU-GPU based acceleration.
- Higher versions of Caffe are highly scalable and fast in computation.
- Consists of pretrained models with configuration files.
- Composed fast and well tested codes.
- Allows seamless transover between CPU and GPU.
- Command line Python based interfaces are available.
- Fundamental layers of Caffe are data access, convolutional, pooling, activation function, loss function, and dropout.
- Focuses more on reinforcement learning enabled applications.

Sample architecture of caffe for image classification is shown in figure 10, is mainly composed of convolutional layers, pooling layers, pretraining layers and custom loss layers. The input dataset is parsed and feed as input set of convolutional layers, the pooling layer acts progressively on the input sample to reduce the size of the image and in turn reduce the parameters and computation involved in image recognition. Then with the application of pretraining layer, the custom loss function the model is fine tuned to predict the output image.

Advantages

Figure 10. Internal architecture of Caffe



- Faster training compared to TensorFlow.
- Allows GPU and CPU based computation.
- Permission is provided to add more number of convolutional and pooling layers.
- Ease of use due to good GUI design.
- Supports more number of internal datatypes like NumPy, NumPy arrays, and so on.
- Achieves high performance even with the use of CPU and GPU.
- Adds out of box facility for image recognition and classification.
- Designed very well by keeping modularity and speed in mind.

Disadvantages

- Weak support for recurrent neural network.
- Lazy allocation of memory for high intensity tasks.
- Implementing recurrent neural network is difficult.
- Partial support for multi-GPU training.
- Poor support in terms of tutorials for beginners and documentation.

cuDDN

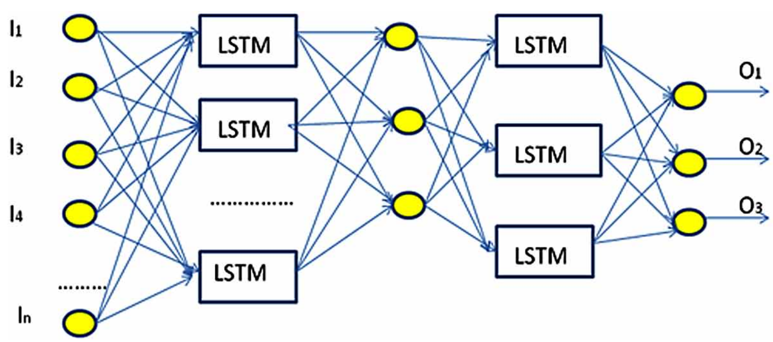
cuDDN (CUDA Deep Neural Network library) is GPU based library used tom implement deep neural network and does computation at accelerated speed. The cuDDN is highly tuned as it consists of in-built implementations of feed forward network, forward convolution, backward convolution, pooling layers and activation layers. It allows training of the neural network with accelerated speed, as it consists of learning frameworks like TensorFlow, PyTorch, Keras, Caffe, MATLAB, and Chainer. (Chetlur, Woolley, Vandermersch, Cohen, Tran, Catanzaro, and Shelhamer 2014; Oyama, Ben-Nun, Hoefler, and Matsuoka 2018) The main characteristics of cuDDN are listed below.

- Consists of depth-wise separable convolutional layers.
- Accelerated performance is achieved via TensorFlow models like DeepSpeech, GAN, and R-CNN.
- Supports mixed precision tensor operations like semantic segmentation, denoising, super resolution of images, and so on.

- Able to choose best recurrent neural network implementation using recurrent neural network APIs.
- Can be integrated easily into other neural network implementations.
- Tuning time is less on low level GPU performance.
- The cuDNN runs with the prerequisite of visual studio.
- Tensor core acceleration is achieved with FP32 inputs and outputs.
- Supports arbitrary dimension ordering, striding, and sub-regions operations.
- Achieves 3x faster training of CNN and RNN.

The architecture of GPU accelerated cuDNN is shown in figure 11, the base layer consists of set of GPU array, on top of which the cuDNN layer sits. The architecture supports frameworks including caffe, kaldi, torch and theano, on top which the applications using cuDNN will run. The cuDNN achieves high performance with the support of maximum flexibility offered by the high level programming languagues, easy acceleration through OpenAcc directives, and drop-in acceleration is achieved via libraries of cuDNN.

Figure 11. Architecture of GPU accelerated cuDNN



Advantages

- Allows normalization which supports for early convergence.
- Allows programmers to construct deep neural network without the need of any CUDA custom code.
- Support flexible data layout as it is capable of handling any data layouts.
- High performance is achieved with minimum usage of memory.
- High speed at execution as 80 to 90 percentage of the tasks are executed in convolution layers.
- The cuDNN routines can be implemented directly without any side effects.
- Strategies are available to select best convolution algorithm.
- Extends support for both 2D and 3D datasets
- Automatically pads zero in border of pooling layers.
- Parameter scaling is achieved easily with the support of built-in routines.
- Good GUI is available to monitor the working of neural network.

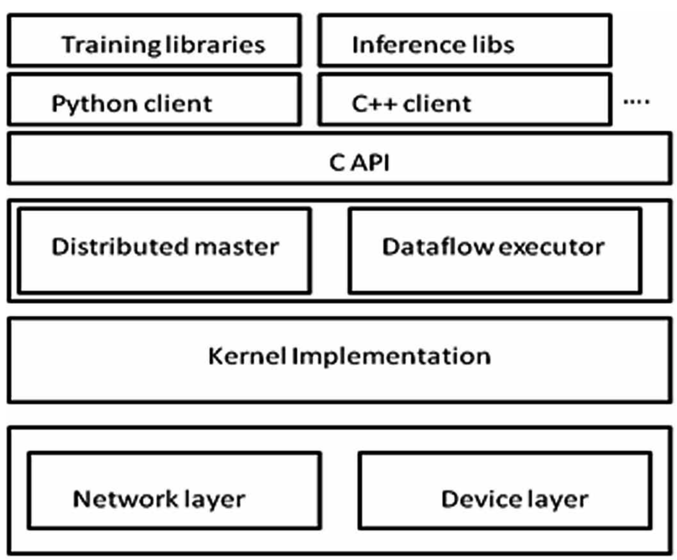
Disadvantages

- Less flexible compared to TensorFlow and Theano.
- Training time is lot despite of parallelization.
- Deep learning workloads are computationally intensive and it takes more time in cuDNN.
- Very less projects of cuDNN is available online.
- Not suitable for applications involving text, audio, video or any form of multimedia applications.
- Tedious to train complex and big networks like GoogleNet and ResNet.
- Community support for cuDNN training is not more.

CUDA-CONVNET

CUDA-convnet is C++ or CUDA based deep learning framework highly optimized for the given input sample and works on GPU accelerated systems. To launch any application in CUDA-Convnet first the model need to be developed, the solver command need to be written and run then finetuning of the convoluted neyral network is carried out (Tran, Ray, Shou, Chang, Paluri 2017; Xiong 2017).

Figure 12. Architecture of CUDA-convnet



The main characteristics of CUDA-Convnet are given below.

- The codebase of CUDA-convnet is not dynamic in nature.
- The CUDA-convnet are run only on GPU based systems.
- It is mainly composed of three modules i.e. convolution, pool, and response normalization.
- CUDA-Convnet achieves efficient implementation of convolution on CUDA.

- Allows implementation of sense and grappa algorithms in CUDA and C++ labguagues.
- Simple desktop applications can be developed easily.

The architecture of CUDA-convnet is given in figure 12; it mainly consists of convolutional layers, pooling layers, and normalization layers. The input sample composed of raw pixel values is fed as input, convolutional layer compute the output of every neuron as product of weight and small region. Pooling layer performs down sampling operation and finally the normalization layer normalizes and generates the output.

Advantages

- Achieves faster implementation of convolutional layers.
- Capable of running superfast on GPU based systems.
- Optimized performance is achieved through the use of CPU and GPU.
- High performance is achieved through small batch sizes.
- Easy to train deep convoluted neural networks.
- High computational throughput is achieved at lower cost.

Disadvantages

- Learning process gets halted frequently.
- Training time is more.
- Due to normalization lot of memory gets wasted.
- Space complexity is high.
- Poor in terms of documentation

MATLAB

Matlab provides a framework for designing the deep neural networks, in which the model is trained to do classification of images, audio, video, and texts. In matlab, simple command line arguments are used to create deep neural network and keeps adding layers to it (Kim 2017; Vedaldi, and Lenc 2015). The main characteristics of matlab towards deep learning are listed below.

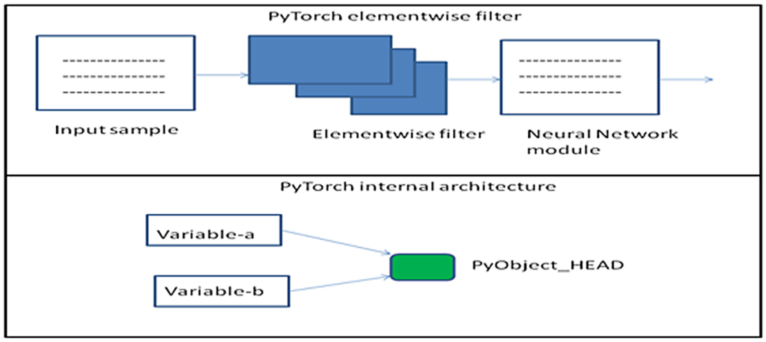
- Pretrained networks are available in matlab which makes it easy for implementation and extension.
- Easy to do advanced operations without the prerequisite knowledge of computer vision algorithms.
- Advanced driver assistance is provided for applications like autonomous car driving, detection of lane, automated parking system, identification of pedestrian on road, and so on.
- Ability to learn directly using the dataset without the need of any predefined models.

A sample deep learning application of image recognition using Matlab is given in figure 13, in which the input image is taken and fed as input to deep layered stack of convolution, ReLU, and pooling layers.

Advantages

- Supports transfer learning to fine tune the network.

Figure 13. Image classification using Matlab



- Easy to resize, rotate, and process the image for classification and regression purpose.
- Good GUI design as a result the features of the network can be viewed easily.
- Ability to create deep learning networks using time series data.
- Capable of working parallel on CPU, GPU, and multiple GPUs.
- Training train reduces even while dealing with huge datasets.
- Able to learn deep learning process without the need to create new network.

Disadvantages

- Frequently suffers from overfitting problem.
- Cost is more since it is licensed software.
- Cross compilation is a tedious activity.
- Deep learning usually deals with numerous errors.
- Sufficient deep learning libraries are not available.
- Exhibits steep deep learning curve.
- While doing deep learning lot of unnecessary computation happens which leads to memory wastage.
- Unfortunately more bugs occur during compilation and debugging becomes tough.
- Graph plotting ability is weak.
- Exhibits weaker integration with non-scientific based data processing applications.

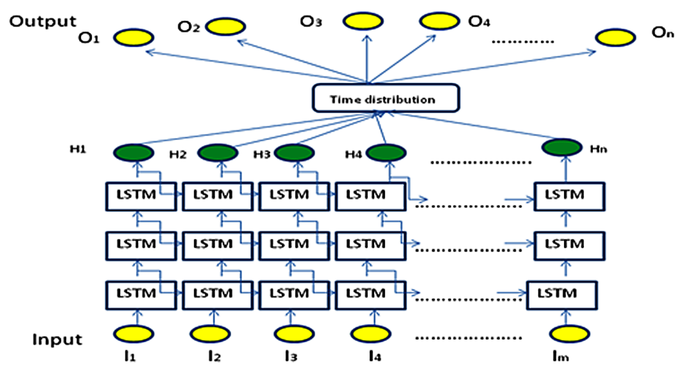
PYLEARN2

Pylearn2 is a machine learning framework constructed on top of Theano, new Pylearn2 plugins using mathematical expression, models and algorithms are built. It is capable of compiling and executing on both CPU and GPU (Goodfellow, Warde-Farley, Lamblin, Dumoulin, Mirza, Pascanu, and Bengio 2013; Robinson 2017).

The main characteristics of Pylearn2 are listed below.

- Popularly used toolbox for scientific computation.

Figure 14. Pylearn2 architecture



- The subcomponents of Pylearn2 can be reused easily.
- Optimized framework to build normal convolution experiments.
- Supports cross platform learning of neural network models.
- Provides dataset interface for multimedia data like audio, video, and images.
- Consists of rich set of plotting libraries.

The architecture of Pylearn2 is given in figure 14, Pylearn2 sits on top of Theano layer originally build to make machine learning easy. The architecture of Pylearn2 is built in such a way that is can reconfigured and supports implementation of wide range of machine learning experiments.

Advantages

- Implementation overhead is less.
- Capable of solving wide range of regression problems.
- Good support from Pylearn2 developer's community.
- Offers higher flexibility through scikit-learn libraries.
- The deep learning models can be constructed for classification and regression purpose.
- It is considered as best Python library for neural network implementation.
- Learning curve is steep.
- Good performance is achieved on GPU based systems.
- Various neural networks starting from multilayer perceptrons to restricted Boltzmann machine is predefined.
- Easily customizable sigmoid or tanh activation functions of neural networks.

Disadvantages

- Multiple GPU training is not supported.
- Data parallelism is not properly utilized.
- Does not consist of hooks for integration with other systems.
- Unable to achieve high performance with non GPY systems.
- Not suitable for generative adversarial network.
- Slower compared to other frameworks like Theano, Matlab, PyTorch, and TensorFlow.

- Inflexible prototyping in terms of large scale dataset processing.
- Challenging to load 3rd party dataset.

LASAGNE

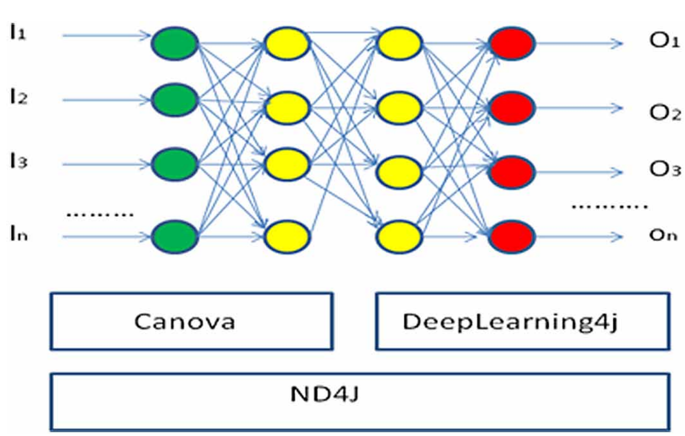
Lasagne is one of the lightweight machine learning library used for neural network implementation. The Lasagne library is still in construction phase which requires Python 2.7 or Phython 3.4 for execution purpose. The library consists of several layers for neural network implementation and provides support for parameter sharing and propagation of data through layers (Chen, Ju, Zhou, Zhang, Chen, Chang, and Wang 2018; Bahrampour, Ramakrishnan, Schott, and Shah 2016). The main characteristics of Lasagne are listed below.

- Consists of layers with multiple behaviors.
- Supports construction and customization of multi layered perceptrons.
- CNN, LSTM, and RNN can be constructed easily.
- Able to build architecture with multiple input and output.
- Several optimization methods are supported which includes Nesterov, RMSprop and ADAM.
- Support computation on both CPU and GPU.
- Allows modularity by adding various layers.
- Allows to directly processing the Theano mathematical expressions.

Advantages

- Simple and very well built library which is used to facilitate the research.
- Provides several features in terms of layers, optimizers and regularizers.
- Achieves transparency by hiding unnecessary details.
- Support common use cases of deep learning.
- Freely definable cost function of activation function of neural networks.

Figure 15. Lasagna LSTM model



Disadvantages

- No support for symbolic loops.
- Computation speed is less compared to other machine learning frameworks like TensorFlow and PyTorch.
- Memory usage is hard.
- It does not consist of several pretrained models.
- Compilation time of more for large data models.

Sample lasagna LSTM model is shown in figure 15, consists of a input layer followed by several hidden layers, a layer of LSTM, and a output layer.

The performance of the deep learning tools is summarized in Table 1 towards the performance metrics like convergence rate, precision, speed of operation, and scalability. From the Table 1 an inference is drawn that the performance of the Tensorflow and Pytorch is good towards implementation of deep learning algorithms.

Table 1. Comparison of deep learning tools

Deep Learning Tools	Convergence Rate	Precision	Speed of Operation	Scalability
TENSORFLOW	High	Medium	High	High
PYTORCH	High	High	Medium	High
KERAS	Low	Low	Medium	High
DEEPLEARNING4J	High	Low	Medium	High
CAFFE	Low	Medium	Medium	Low
cuDDN	High	Medium	Medium	Low
CUDA-CONVNET	High	High	High	Medium
MATLAB	Medium	Medium	Low	Low
PYLEARN2	Low	Medium	Low	Medium
LASAGNE	Low	Medium	Medium	Low

CONCLUSION

This chapter provides a brief discussion on deep learning highlights the differences between deep-learning and non-deep learning techniques and provides comprehensive discussion on the popular deep learning architectures and tools. Around ten popular deep learning tools are discussed along with their advantages and disadvantages and draw an inference that the tools like TensorFlow and PyTorch are very competitive and useful tools for deep learning applications.

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., & Dean, J. (2016). Tensorflow: A system for large-scale machine learning. *12th {USENIX} Symposium on Operating Systems Design and Implementation*, 265-283.

Angelov, P., & Sperduti, A. (2016). Challenges in deep learning. *24th European Symposium on Artificial Neural Networks*, 489-495.

Baeldung. (2018). *A Guide to Deeplearning4j*. Retrieved may 31, 2019 from https://www.baeldung.com/deeplearning4j

Bahrampour, S., Ramakrishnan, N., Schott, L., & Shah, M. (2016). Comparative study of caffe, neon, theano, and torch for deep learning. *Proceedings of the International Conference on Learning Representations*, 1-11.

Bhargavi, K., & Babu, B. S. (2017). Accelerating the Big Data Analytics by GPU-Based Machine Learning: A Survey. *International Symposium on Sensor Networks, Systems and Security*, 63-83.

Chen, M., Ju, C. J. T., Zhou, G., Zhang, T., Chen, X., Chang, K. W., & Wang, W. (2018). Lasagna: Multifaceted Protein-Protein Interaction Prediction Based on Siamese Residual RCNN. *bioRxiv*, 501791.

Chen, X. W., & Lin, X. (2014). Big data deep learning: Challenges and perspectives. *IEEE Access: Practical Innovations, Open Solutions*, *2*, 514–525. doi:10.1109/ACCESS.2014.2325029

Chetlur, S., Woolley, C., Vandermersch, P., Cohen, J., Tran, J., Catanzaro, B., & Shelhamer, E. (2014*). cudnn: Efficient primitives for deep learning*. arXiv preprint arXiv:1410.0759

Deng, L., & Yu, D. (2014). *Deep learning: methods and applications*. Foundations and Trends in Signal Processing.

Edward, D. (2017). *Intro to TensorFlow and Machine Learning*. Retrieved may 31, 2019 from https://meetings.internet2.edu/media/medialibrary/2017/10/18/20171018-doan-intro-tensorflow-machine-learning.pdf

Erickson, B. J., Korfiatis, P., Akkus, Z., Kline, T., & Philbrick, K. (2017). Toolkits and libraries for deep learning. *Journal of Digital Imaging*, *30*(4), 400–405. doi:10.100710278-017-9965-6 PMID:28315069

Falcini, F., & Lami, G. (2017). Deep Learning in Automotive: Challenges and Opportunities. *International Conference on Software Process Improvement and Capability Determination*, 279-288. 10.1007/978-3-319-67383-7_21

Fei-Fei, L., Justin, J., & Serena, Y. (2017). *Deep learning software*. Retrieved may 31, 2019 from http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture8.pdf

Goodfellow, I. J., Warde-Farley, D., Lamblin, P., Dumoulin, V., Mirza, M., Pascanu, R., & Bengio, Y. (2013). *Pylearn2: a machine learning research library*. arXiv preprint arXiv:1308.4214

Gulli, A., & Pal, S. (2017). *Deep Learning with Keras*. Packt Publishing Ltd.

Julien, D. (2017). *Getting Started with Caffe*. Retrieved may 31, 2019 from https://indico.cern.ch/event/395374/contributions/939872/attachments/1186808/1721069/Getting_started_with_caffe_v2.pdf

Ketkar, N. (2017). Introduction to pytorch. In *Deep learning with python* (pp. 195–208). Berkeley, CA: Apress. doi:10.1007/978-1-4842-2766-4_12

Kim, P. (2017). Matlab deep learning. In *With Machine Learning*, *Neural Networks and Artificial Intelligence.* Apress. doi:10.1007/978-1-4842-2845-6

Martin, H. (2018). *What is Keras? The deep neural network API explained*. Retrieved may 31, 2019 from https://www.infoworld.com/article/3336192/what-is-keras-the-deep-neural-network-api-explained.html

Oyama, Y., Ben-Nun, T., Hoefler, T., & Matsuoka, S. (2018). Accelerating deep learning frameworks with micro-batches. *IEEE International Conference on Cluster Computing (CLUSTER)*, 402-412. 10.1109/CLUSTER.2018.00058

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., & DeVito, Z. (2017). Automatic differentiation in pytorch. *31st Conference on Neural Information Processing Systems (NIPS 2017)*, 1-4.

Peter, A. (2017). *A Practical Introduction to Deep Learning with Caffe*. Retrieved may 31, 2019 from https://panderson.me/images/Caffe.pdf

Pulkit, S. (2018). *An Introductory Guide to Deep Learning and Neural Networks.* Retrieved may 31, 2019 from https://www.analyticsvidhya.com/blog/2018/10/

Robinson, S. (2017). *The Best Machine Learning Libraries in Python*. Stack Abuse.

Tran, D., Ray, J., Shou, Z., Chang, S. F., & Paluri, M. (2017). *Convnet architecture search for spatiotemporal feature learning*. arXiv preprint arXiv:1708.05038

Vedaldi, A., & Lenc, K. (2015). Matconvnet: Convolutional neural networks for matlab. *Proceedings of the 23rd ACM international conference on Multimedia*, 689-692. 10.1145/2733373.2807412

Xiong, Y. (2017). *Install and Run cuda-convnet on Windows OS with VS2010*. Retrieved May 31, 2019 from http://yjxiong.me/others/install_cudaconvnet.html

Zhang, Q., Yang, L. T., Chen, Z., & Li, P. (2018). A survey on deep learning for big data. *Information Fusion*, *42*, 146–157. doi:10.1016/j.inffus.2017.10.006

Chapter 3
Facial Emotion Recognition System Using Entire Feature Vectors and Supervised Classifier

Manoj Prabhakaran Kumar
https://orcid.org/0000-0003-4032-8433
VIT University, India

Manoj Kumar Rajagopal
VIT University, India

ABSTRACT

This chapter proposes the facial expression system with the entire facial feature of geometric deformable model and classifier in order to analyze the set of prototype expressions from frontal macro facial expression. In the training phase, the face detection and tracking are carried out by constrained local model (CLM) on a standardized database. Using the CLM grid node, the entire feature vector displacement is obtained by facial feature extraction, which has 66 feature points. The feature vector displacement is computed in bi-linear support vector machines (SVMs) classifier to evaluate the facial and develops the trained model. Similarly, the testing phase is carried out and the outcome is equated with the trained model for human emotion identifications. Two normalization techniques and hold-out validations are computed in both phases. Through this model, the overall validation performance is higher than existing models.

INTRODUCTION

Since the early 4th century (Aristotelian era), researchers have been interested in studying physiognomy and facial expression (Highfield, 2009). Physiognomy is the assessment of a person's personality, character, or behavior based on his/her outer appearance, especially his/her face. But over the years, the study of physiognomy has not been focused on human behavior; only the facial expressions of humans has

DOI: 10.4018/978-1-7998-2108-3.ch003

continuously been an active topic. The foundational studies that formed the basis of today's research on facial expressions of humans can be tracked to the 17^{th} century. In his book *Pathomyotmia*, gives details of muscle movements of the human head and various expressions in humans (Bulwer, 1649). In 1667, Le Brun's lectured on the physiognomy of the face; latterly revised his theories in the book of (Montagu., 1994). In the 18^{th} century, artists and actors referred to Le Burn's book to achieve "the perfect imitation of 'genuine' facial expression." However, in the 19^{th} century, a facial expression has a direct relationship of the automatic facial expression and analysis was given by Darwin's statements. In 1872 and 1965, Charles Darwin's book states the principles of basic emotion evolved in both human and animal, and also grouped the various kinds of expressions and cataloged the facial deformations (Darwin, 1904). In 1884, William James proposes the "James Lange theory" that the various kinds of expressions have derived from the presence of stimuli in the body, which is evoke by physiological responses (James, 1884).

Another important milestone for the study of facial expressions and the six basic emotions (i.e., surprise, happy, disgust, fear, anger, and sad) in humans was outlined by (Friesen., 1969). (Ekman, 1977) developed the automatic human-facial-expression recognizer, and analyses of facial expressions' muscular movements showed the different emotions in humans through photographic stimuli. Then, in 1978, Wallbott and Scherer determined an emotion from the body's muscular movements and speech signals (Wallbott, 1986). Suwa, Mase, and Pentland established the automatic recognition of facial expressions for analyzing facial expressions from the image sequence of tracking points, but they did not clearly see these tracking points until the 1990s (A.Pentland, 1991) and (N.Sugie, 1978). Later, in 1992, Samal and Iyengar established the facial feature and expression analysis of tracking points in movie frames and also established the robust model of automatic facial expression recognition system, which required facial feature detection and a facial tracking system (Samal, 1992). Since the 1990s, several researchers have been more interested in emotion recognition in humans for Human Computer Interaction (HCI), affective computing, and so forth. Emotion recognition in humans was established by (Karpouzis, 2005), and the various modes of extraction are: physiological (EEG, ECG, etc.,) and non-physiological signals (face, body, speech, text, etc.). (Karpouzis, 2005), states that facial expression recognition is best out of the various modes of extracting emotion methods. Since 1990, researchers have been mostly concentrating on the robust model of automatic facial emotion in humans through the face, which is compared to other modes of extracting emotion.

Developing an automatic facial expression recognition system which could be based on the detection of the human face using face detection and tracking, and extracting the feature/model. Finally, understanding the behavior of humans through their faces, using the facial expression classifier, with the help of the parameterized method. In the current chapter, the real-time facial expression recognition system is proposed for the detection of emotion in humans through the face. There are two novel methods of automatic facial expressions, one of which is based on the Constrained Local Method (CLM/AAM), is proposed for the face detection and tracking of the human face, and then the geometric deformable model of feature points is extracted (Simon, Jeffrey, & Jason, 2011). Second, the binary class of Support Vector Machines (SVMs) (Naumovich, 1998; Ventura, 2009) is proposed for the classification of emotions in the human face, with the help of Facial Animation Parameters (FAPs) (Zhang, Ji, Zhu, & Beifang Yi, 2008) on extracted features of the geometric deformable face model.

The rest of the chapter is as follows: the second section describes the background of automatic facial expression recognition in face modeling, describing the state-of-the-art Constrained Local Model (CLM/AAM) for face detection, tracking, and feature extraction. Then the binary class of Support Vector Machines (SVMs) with a parameterized model to attain the emotion classifiers in the human face is discussed. The third section describes novel methods for the proposed real-time facial expression recognition system. The fourth section describes the automatic facial expression of the experimental results of the six basic emotions and provides a discussion of it. The last section summarizes future research opportunities and provides a conclusion.

BACKGROUND

The background section details the important components for the emergent, real-time facial expression system of the facial imaging sequence. First, are the face modeling for good face detection and the tracking and extraction of feature. Then a parameters description is used for facial prototype expression in permanent face features. Lastly, the classification of facial expression recognition has more accuracy and reliability, which depends upon the choice of classifiers.

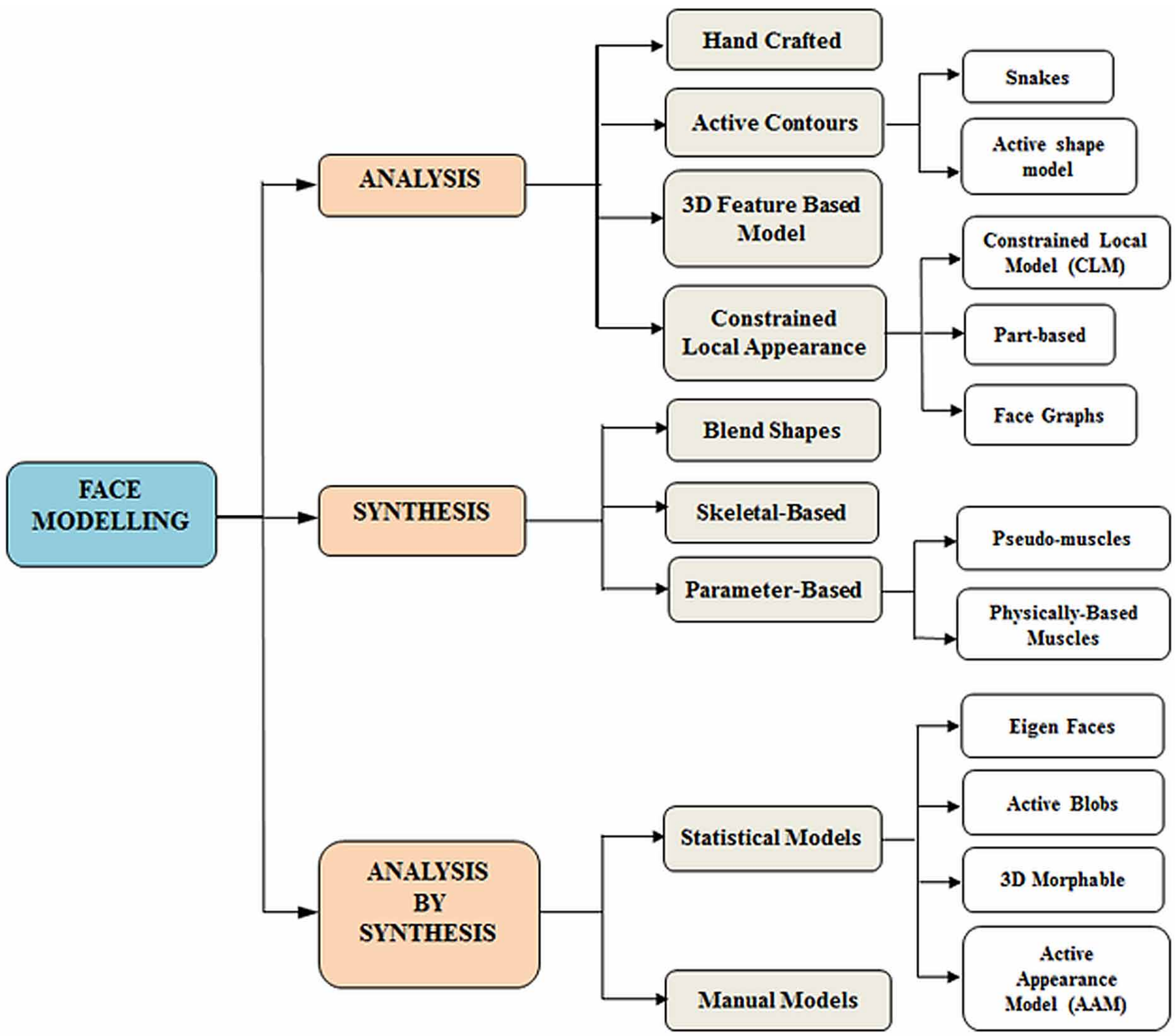
State-of-the-Art Face Modeling

In automatic facial expression recognition, first we have to place importance on the robust face identification of humans. Robust face detection and tracking are majorly important for automatic facial expression recognition, which was developed by face modeling. In the face detection and tracking of humans, some issues are varying backgrounds, head poses, occlusion, and lighting effects, can be overcome using state-of-the-art face modeling. Face modeling is used essentially for facial synthesis and animation, and computer vision is used for modeling human faces and their deformations for face analysis, and so on. The face model has explored that domain in Analysis, Synthesis, and Analysis-by-Synthesis.

Analysis

The face analysis, which has terms of combined shape and texture model in faces, for face detection and tracking by these methods, is shown in Figure 1. In Hand Crafted (Baizhen, 2006) has manually designed a feature of the face using a parameterized deformable template of face detection and tracking. The Active Contours of face model and snakes model have computational bridges between the high level and low level information of the extraction of a deformable face object (M. Kass, 1988). And in the Active Shape model, the face is defined by the alignment of the shape representation in the point distribution model and extracted by the Adaboost histogram classifier for face feature detection and tracking (Li, 2005). The 3D Feature-Based model has a 3D parametric model created by 2D facial feature tracking and extraction for analyzing the facial deformation (Cristinacce, 2007). In Constrained Local Appearance, the Constrained Local Model (CLM) has a combination of Eigen Faces and the Active Shape model with a parameterized face model for feature detection and tracking (Cristinacce, 2006). The Part-Based model has facial deformation by the collection of the object parts in the face by pictorial representation for face detection (Huttenlocher., 2005). And face graphs has designed a visual feature face graph by Elastic Bunch Graph Matching (Laurenz Wiskott, 1997).

Figure 1. State-of-the-Art Face Model



Synthesis

The Synthesis face model has defined the animation parameter face and representation of the expression on the face without feature detection and tracking of the face. From the synthesis, the Blend Shape model has a technique for creating a photorealistic 3D texture animation from the pictorial subject's face for the expression of the face (Joshi, 2005). The Skeleton model has a 3D animation skeleton parameter, which is created by a bones rig animation of facial expression (Popovic, 2007). The Parameter-based model has been created using the different deformation models corresponding to the parameter shape in two approaches (Salam, 2013). First, the Pseudo-Shape Muscles Geometric model has facial deformation in a geometric model that simulates muscles' visual effects. The geometric model was created by the Direct Parameterization model (Parke, 1982)) and Elementary Deformation Based model (FACS, FAPs, AMA, etc.) for face animation of expression. And the Physically Based Muscle model stimulates muscle action for facial deformation by physical approaches through parameters (Badler., 1981).

Analysis by Synthesis

The Analysis by Synthesis has a combination of Analysis and Synthesis models for good performance in facial feature detection, tracking, extraction for facial expression recognition, head pose variation, frontal animation of the face with the parameterized model, and profile facial images detection. The analysis by synthesis has two main approaches, as shown in Figure 1. In the statistical model, first the Eigen Faces has the aligned shape of a face to give the set of Eigenvectors for face recognition (P.Pentland, 1991). In Active Blob, 2D triangular mesh is textured–mapped deformable based (region-based approaches) to detect and track the non-rigid motion of the object (Isidoro., 1998). In the 3D Morphable model, the face deformable model has been scanned by the linear combination of 3D photograph or interface directly for face recognition (Vetter, 1999). The Active Appearance model (AAM) has the same Constrained Local appearance (CLM), which uses a combination of Eigen Faces and the Active Shape model (ASM) for face recognition (T.F. Cootes, 1998). It differs in its optimization model of deformable face model fitting. Second, the Manual model (i.e., the Candide model, Piecewise Bezier) has the manually designed the geometric deformable face model with the help of the Facial Action Coding System (FACS) and the Facial Animation Parameter (FAPs) for human face recognition and facial expression (Ahlberg, 2002).

From our review of the state-of-the-art face modeling:

- The Analysis model has the face terms of shape and texture representations for face feature detection and tracking, and Synthesis has the 3D animation of face-parameterized model for the facial expressions in humans.
- The Analysis by Synthesis has a combination of face feature detection, tracking, and the parameterized face model for facial expressions in humans.
- Analysis by Synthesis has been more important for facial feature detection, tracking, and expression classification in human faces, when compared to the Analysis and Synthesis model. In the Analysis by Synthesis model, the CLM/AAM and Manual model are good for facial expression system.

This chapter proposes the Constrained Local Model (CLM), which differs in the optimization strategies of facial expression, which attain a good performance in face detection and tracking, and also help to define the expression in human faces robustly compared to other face models.

Parameterized Models

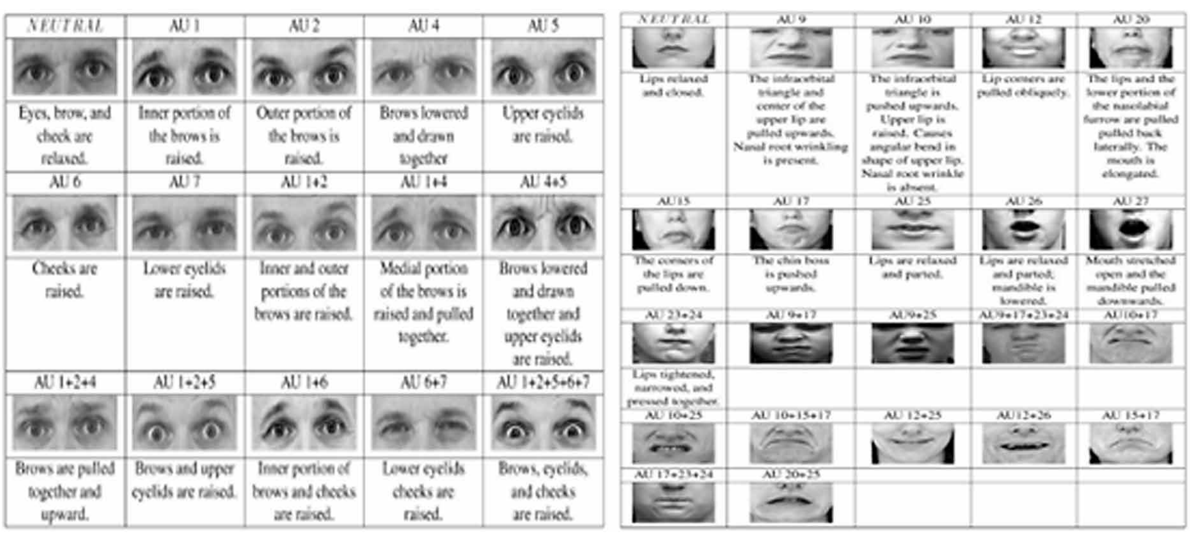
In the automatic facial expression recognition system, the parameterized model has definition based on the facial muscles' actions by set of face parameters for defining the various emotions in the human face. The parameter set of face has various types, but two main systems have been had major roles in the automatic facial expression, which include: the Facial Action Coding System (FACS), developed by Paul Ekman and Friesen (Ekman, 1977), and the Facial Animation Parameters (FAPs), which are part of the MPEG-4 Synthetic/Natural Hybrid Coding Standard (SNHCS) (Audio (MPEG Mtg), 1998).

Facial Action Coding System (FACS)

Ekman and Fresien (Ekman, 1977) developed the six basic emotions in humans from the visual stimuli through the Facial Action Coding System (FACS). The FACS defined the facial muscle action through a photographic or visually interface when there was the stimulation of a physiological response in a human. In the muscle-action approaches, which identify the various muscle actions as individual or combinational, they are said to be an Action Units (AUs). The FACS has 46 Action Units for facial muscle action, which described the emotion in human faces. The Facial Action Coding System is additive or non-additive, which is independent or non-independent of facial muscle action.

In automatic facial expression, the FACS has a huge list of muscle actions to describe the six emotions. The face feature is extracted, which defines the emotion with a combination of FACS and various combinations of classifiers. (Bartlett, Littlewort, Fasel, & Movellan, 2003) applied the feed-forward neural network with FACS to attain the AUs in the face. (Donato, 1999) applied the Gabor Wavelet filter and Independent Component analysis to obtain the emotion in head images. Bartlett et al. reported that the combination of the Gabor filter and support vector machines with FACS describe the Emotion in Frontal image sequence. Cohn and Kanade (Tian, 2001) developed the automatic face expression with Adaboost and Discriminant function analysis, given the AUs of muscles' actions. Kotsia and Pitas (2007) describe the emotion in human faces by the geometric deformation and support vector machine with AUs. Maja Pantic (Pantic & Patras, 2006) developed the automatic facial expression recognition system of FACS in frontal and profile image sequence.

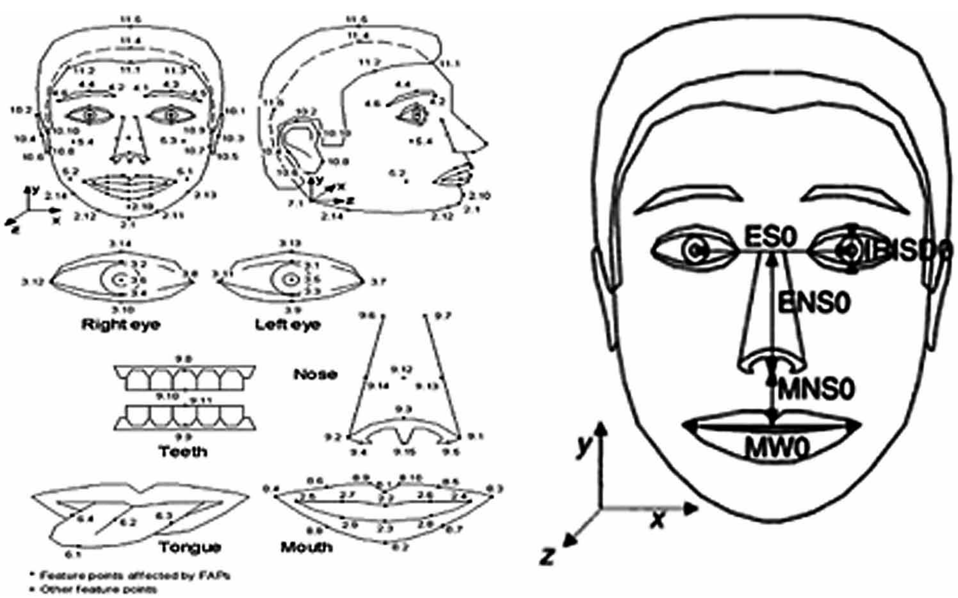
Figure 2. Facial Action Coding System (FACS)



Facial Animation Parameter (FAP)

In the 1990s and after, computer vision researchers were more interested in the facial animation expression analysis using pre-FACS. There is no defined and unifying standard for the facial animation analysis. So Pandzic and Forchheimer (Pandzic, 2002) were interested in developing the facial animation parameters for a set of facial parameters that define the Action Units of facial animation or face features. Later, the facial set of parameters had some issues in the combination of Action Units to describe the emotion. To address the issue, they standardized the parameter of face control using the Moving Pictures Expert Group, developing the facial animation parameter (FAPs) in MPEG standards. The MPEG is universal standard that specifies a Facial Animation Parameters (FAPS) and Facial Definition parameter (FDPs). Cowie (Cowie, 2008.) relates the main relationship between the MPEG-FAPs and FACS of AUs as the FAPs are the main contribution to the definition of the six basic emotions, which is related to Action Units. Abrantes and Pereira (Abrantes, 1999) Pandzic and Forchheimer (Pandzic, 2002) Raouzaiou (A. Raouzaiou, 2002), Tekalp (Ostermann, 2000) characterized the movement of a set of the facial feature parameters of FAPs with the related AUs.

Figure 3. Facial Animation Parameters (FAPs) with feature points and Action Units (AUs)



In FAPs, the set of face parameters that represent the movement of feature parameters defines the face expression in an MPEG4 group, as shown in Figure 3. With FAPs, there are 84 key feature points to define shape and movements of the face. From the feature points are developed the 64 FAPS, which are grouped into 10 Action Units that define the six basic emotions in humans and head pose variation in frontal and profile images. In recent approaches, the automatic facial recognition was developed from the FAPs with AUs. (A. Raouzaiou, 2002) attained the facial expression system of FAPs by Gradient Vector Flow (GVF), the Parabolic Template Algorithm, the combinational algorithm, and the Neurofuzzy network. (Zhang, Ji, Zhu, & Beifang Yi, 2008) attained the dynamic facial expression system of FAPs with the Dynamic Bayesian Network (DBN). (Shan, 2009) developed the facial expression recognition with FAPs' AUs, based on local binary patterns with support vector machines (SVM), Linear Discriminant Analysis (LDA), Principal Component Analysis (PCA), and a Gabor filter. (Petar S.Aleksic and Aggelos K. Katsaggelos, 2006) developed the automatic facial expression recognition system of FAPs with AUs by using an extracted feature through the Hidden Markov model.

From the parameterized model, FACS has 46 Action Units that define the 7,000 combinations of facial expressions in humans, but FAPs only 10 groups is defines the six basic emotions and its categories into 136 emotion in human faces (W.G. Parrott, 2000) In addition, the FAPs define the permanent facial feature movement (eyebrow, mouth, etc.) of each feature. This chapter proposes the Facial Animation Parameter is a description for the Real Time Facial Expressions of humans.

Facial Expression Recognition

The automatic facial expression system needs good facial feature detection and tracking and a well-defined face emotion from the Parameterized model. A robust facial expression recognition system develops the extraction of facial features in a static face or video with a suitable classifier, which are mainly divided into two approaches: 1) Feature-Based Approaches (appearance) and 2) Model-Based Approaches (shape-based).

Feature-Based Approaches

In order to obtain the extraction of facial features, one must extract the certain face region pixels intensities or whole region of face feature pixel value, but that must be good face feature detection and tracking. These approaches are simple and very fast, but it has a higher dimensionality and affects the reliability of facial expression recognition. For improving the system, apply the extraction of facial features in classifications to the exact definition of the emotion in a human face. (Bartlett, Littlewort, Fasel, & Movellan, 2003) have applied the Dynamic Bayesian Networks (DBN) to classify the emotion in static face images. For dynamic approaches of the facial expression system, Cohn (Tian, 2001) applied the Multilevel Hidden Markov Model (HMM). Irene Kotsia (Pitas, 2007) developed the automatic facial recognition system, which defines the six basic facial expressions using the classifier of support vector machines (SVM). Zhang implemented the Neural Network for automatic facial expression recognition (Karpouzis, 2005), developed the Neurofuzzy system for analysis of FAPs variation of emotion in the human face. Gao (Yongsheng.Gao & al, 2003) implemented a Linear Edge Mapping (LEM) classifier for the identification of emotion in facial images.

Model-Based Approaches

The Model-based feature describes the facial feature model as having variations in parameterized facial features that classify the emotions in humans. Facial feature model has lower dimensionality feature extraction, which provides a good performance in a robust face feature expression in a static image and also more detailed information in a video sequence. (Padgett, 1997) and (Zhao J, 1996) implemented the Artificial Neural Network for classifying the expression in static face images. (Littlewort, 2006) developed the Support Vector Machine (SVM) for a dynamic facial expression system in an image face sequence.

Zhang and Ji (Zhang, Ji, Zhu, & Beifang Yi, 2008) implemented the Dynamic Bayesian Network to classify the facial expression in 2D and 3D facial images. Maja Pantic (Pantic & Patras, 2006) developed the Hidden Markov model (HMM) for identifying the emotion in frontal and profile face images. Aleksic (Petar S.Aleksic and Aggelos K. Katsaggelos, 2006) implemented the Hidden Markov Model (HMM) to classify the automatic facial expression recognition system in 3D facial images robustly.

Based on these two approaches, the Support Vector Machines have more accuracy and reliability for the classification of emotions in the human face. This chapter proposes the Support Vector Machines for the classification of the real time facial expression of image sequence.

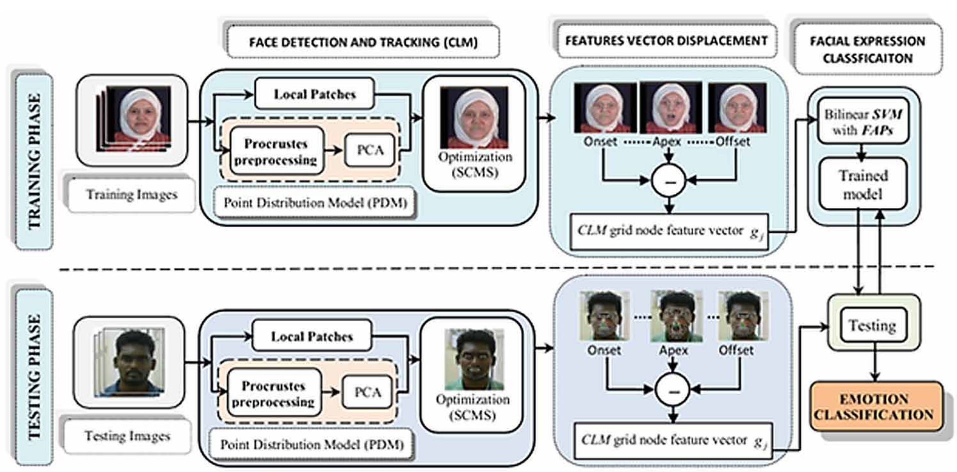
In this chapter, proposes the

- Constrained Local Model (CLM) (Simon, Jeffrey, & Jason, 2011) for face detection and tracking and feature extraction are also evaluated in this model.
- For the description of each prototype expression using the Facial Animation Parameters (FAPs), which defines of the features vector displacements.
- The normalized entire feature vector is a concern in the facial expression system of the facial image sequence using two classes of Support Vector Machines.

SYSTEM DESCRIPTION

The robust facial expression recognition system is composed of two categories: one for face detection, tracking, and extraction information through a grid node; and the second for the extraction of information out of the grid node on classifications to define the six basic emotions in human faces. The grid node extraction information is done by Constrained Local Model (CLM) and two classes of Support Vector Machines (SVM). The system architecture of proposed automatic facial expression recognition is shown in Figure 4.

Figure 4. Real-time facial expressions recognition systems



Information on Grid Node Extraction

Linear Grid Shape Model

In deformable model fitting, before searching for the local detector of face feature, the Discriminative model (CLM) (Simon, Jeffrey, & Jason, 2011) builds the 2D vertex-based grid shape node. The shape model of a non-rigid object *s* of the Point Distribution Model (PDM) (Cristinacce, 2007) and (Cristinacce, 2006) is represented by 2D vertex location mesh with 2*v* dimensionality vectors of shape $s = (x_1, y_1, \ldots\ldots, x_v, y_v)^T$; *v* is the 2D landmark point of the PDM. In building according to the Point Distribution Model (PDM), the Principal Component Analysis (PCA) gets an aligned shape. For the aligned shape of the 2D vector mesh to have different shape variation *s*, rotation *R*, and translation t_x, t_y, one must remove these parameters and mark the feature points on images applying the Procrustes analysis preprocessing before the PCA.

The linear shaped model of the PDM is a non-rigid shape variation, and it is composed of a global transformation of rigid shapes represented as

$$\tilde{x}_i = \vec{x}_i + P_s b_s \; or \; \tilde{x}_i = \vec{x}_i + \Phi_i b_s \tag{1}$$

$$x_i = sR\left(\tilde{x}_i\right) + T_{t_x,t_y} \Leftarrow T_{s,R,t_x,t_y}\left(\tilde{x}_i\right) \tag{2}$$

Where $\vec{x}_i$ denotes a mean shape or base mesh of *i*[th] landmark of 2D location of the PDMs, P_s is the n Eigen vector of the non-rigid shape subspace matrix, b_s is the n dimensional vector of set of the non-rigid shape parameters, and $p = \left(s, R, t, q\right)$ represents as holding the four pose parameters of PDM's.

In CLM, the PDM is the 2D vector of the aligned shape model, but need as good detectors of local (patch) search for CLM landmark fitting and optimization strategies for tracking of local search. Holistic approaches have some drawbacks of generative methods and are overcome in the Discriminative model (CLM) as reviewed in the section on Analysis by Synthesis. This current CLM has mainly two goals: i) exhaustive local detectors for each 2D PDM landmark to get the response map using some feature detector algorithm and ii) the detection of response maps and overall landmark of PDM parameters are jointly maximized using by optimization strategies. Figure 4 is shown as the Constrained Local Model (CLM) components.

Local Images Search

In the first step of CLM landmark fitting, the appearance model of CLM has likelihood maps that are generated by local (patch) detectors for each landmark to get the response maps. For a local response search (and Wang, 2008.), a number of feature detectors have been implemented by various algorithms. For *v* as local detector is evaluated the 2D pixel location $x_j = \left(x_j, y_j\right)$ of local images with correlation of the ith landmark parameter is given by Equation (3)

$$C_i\left(I\left(x_i\right)\right) = w_i^T\left[I\left(x_1\right);\ldots\ldots;I\left(x_m\right)\right] + b_i \tag{3}$$

C_i denotes as a linear classifier of local detector with $\left\{x_i\right\}_{i=1}^m \in \Omega_{x_i}$ denotes as an image patch of a landmark, and w_i^T denotes as a linear detectors. *I* is the image, x denotes as a 2D location in the local images, and b_i denote as shape vector parameters. The 2D landmark deals with a local maps detector by including the wrap normalization step, particularly similarity transformation into the base grid node. At this stage, local detectors of the 2D landmark converted the exhaustive local maps search into a match score or similarity measures (Gaussian likelihood)/probabilistic output function (Simon, Jeffrey, & Jason, 2011) to get the response patch maps $\left\{p\left(l_i = aligned \mid I,x\right)\right\}_{i=1}^n$ and also other good local detectors for CLM 2D framework such as Haar-based boosted classifier (C.J.Taylor, 1992). In our work, the simplest solution of a local maps search is the Linear Logistic Regressor (Simon, Jeffrey, & Jason, 2011), which gives the exhaustive local feature detectors to get response images $\left\{p\left(l_i = 1 \mid I(x_i)\right)\right\}_{i=1}^n$ of the i^th^ landmark PDM. The probabilistic output function of local detectors is

$$p\left(l_i = aligned \mid I,x\right) = \frac{1}{1+\exp\left\{\alpha C_i\left(I;x\right)+\beta\right\}} \tag{4}$$

$$p\left(l_i = 1 \mid I(x_i)\right) = \frac{1}{1+e^{-\alpha_i\beta_1 C_i\left(I(x_i)\right)+\beta_0}} \tag{5}$$

Where α_i denotes the correct landmark of local maps, l_i is a discrete random variable that denotes whether the ith landmark of 2D PDM is correctly aligned or not, and β_0, β_1 are the regression coefficient respectively of a local images search. Only a proper probability function is always negative, and

$p\left(l_i = 1 \mid I(x_i)\right) + p\left(l_i = -1 \mid I(x_i)\right) = 1$.

Existing Global Optimization Strategies

Once the response maps of each landmark of local search have been found, the conditional independence between the probabilistic function of local response images detection for each landmark is maximized by optimization strategies:

$$p\left(\left\{l_i = aligned\right\}_{i=1}^n \mid p\right) = \prod_{i=1}^n p\left(l_i = aligned \mid x_i\right) \tag{6}$$

With respect to **p** PDM parameters, x_i is parameterized of response images. The summation of local responses of CLM pose is minimizing by Equation (6). From the formulation (6), (Simon, Jeffrey, & Jason, 2011) when assuming the uniform prior over the 2D PDM parameters, which leads to a Maximum Likelihood (ML) estimate or Maximum *a-posterior* (MAP) estimate. For a linear shape model to attain the set of aligned shapes applied by the PCA, and non-rigid shape parameters to be attained by Gaussian distribution, assume that non-primitive and primitive prior are all rigid transformations and are equally in optimization. In our work, optimization strategies for maximizing a constrained PDM landmark by Subspace Constrained Mean-Shift (SCMS) only jointly maximizes the all non-parametric representation of a response image at once by employing the Kernel Density Estimator.

Active Shape Model

The method is replacing the simpler parametric of true response map $\left\{p\left(l_i \mid x\right)\right\}_{i=1}^{n}$, and performing the optimization strategies instead of original response. The Active Shape model (C.J.Taylor, 1992) entails first finding the location of the maximum attained within each response map $\mu = \left[\mu_1;\ldots;\mu_n\right]$. The optimization procedure is the difference between the PDM landmark and the coordinate of the maximum peak response with minimizing the weighted least square.

$$Q\left(p\right) = \sum_{i=1}^{n} w_i \left\|x_i - \mu_i\right\|^2 \tag{7}$$

Where $\left\{w_i\right\}_{i=1}^{n}$ is the weights that reflect the maximum peak response μ_i coordinate and linear shape model $\left\{x_i\right\}_{i=1}^{n}$ for a high matching score of optimization. This is typically set to $\left\{x_i\right\}_{i=1}^{n}$ of response at $\left\{\mu_i\right\}_{i=1}^{n}$, making it more resistant and more weakly weighted for partial landmark occlusion. For minimizing the optimization *prior* by taking the first order Taylor expansion of 2D PDM landmark, Equation (7) is:

$$x_i \approx x_i^c + J_i \Delta p \tag{8}$$

and the parameter update solving:

$$\Delta p = \left(\sum_{i=1}^{n} w_i J_i^T J_i\right)^{-1} \sum_{i=1}^{n} w_i J_i^T \left(\mu_i - x_i^c\right) \tag{9}$$

The current parameters are $p \leftarrow p + \Delta p$ applied the parameter update Δp to estimate the current pose and shape. Here is the Jacobian $J = \left[J_1;\ldots;J_n\right]$ and current shape estimate $x = \left[x_1^c;\ldots;x_n^c\right]$ of the current parameter update. Using a Isotropic Gaussian estimator for obtaining the approximation the response maps is equivalent to, the ASM optimization is:

$$p\left(l_i = aligned \mid x\right) \approx N\left(x; \mu_i, \sigma_i^2 I\right) \tag{10}$$

Where $w_i = \sigma_i^{-2}$ with this approximation of ASM optimization, only taking the likelihood negative log in Equation (6) results in the objective in Equation (7).

Subspace Constrained Mean Shift

For each PDM landmark response maps approximation using the parametric model, here we consider the non-parametric model of response map approximation using the Subspace Constrained Mean Shift (SCMS) (Simon, Jeffrey, & Jason, 2011). In this SCMS, we propose using the homoscedastic Kernel Density Estimate (KDE) (Silverman, 1986) with an isotropic Gaussian kernel:

$$p\left(l_i = aligned \mid x\right) \approx \sum_{\mu_i \in \psi_{x_i^c}} \alpha_{\mu_i}^i N\left(x; \mu_i, \sigma^2 I\right) \tag{11}$$

Equation (12) $\alpha_{\mu_i}^i$ defines the normalized true response detector through $\psi_{x_i^c}$ is 2D kernel grid centers are fixed at current estimate x_i^c, which, mixing weights obtained from the true response maps σ^2, is a Gaussian kernel variance that regulates the approximation smoothness. The advantage of KDE (Simon, Jeffrey, & Jason, 2011) is not required to learn the parameter representation by nonlinear optimization, and the GMM based representation of a mixture response is complex in computational and fitting the suboptimal nature of a mixture model, then KDE is more stable and efficient, when σ^2 is set a-priori.

$$\alpha_x^i = \frac{p\left(l_i = aligned \mid x\right)}{\sum_{y \in \psi_{x_i^c}} p\left(l_i = aligned \mid y\right)} \tag{12}$$

The maximizing of KDE optimization is nontrivial representation using the Mean-Shift algorithm (Cheng, 1995), which consists of a fixed-point iteration and is:

$$x_i^{(\tau+1)} \leftarrow \sum_{\mu_i \in \psi_{x_i^c}} \frac{\alpha_{\mu_i}^i N\left(x_i^{(\tau)}; \mu_i, \sigma^2 I\right)}{\sum_{y \in \psi_{x_i^c}} \alpha_y^i N\left(x_i^{(\tau)}; y, \sigma^2 I\right)} \mu_i \tag{13}$$

Where $J^\dagger$ denotes the step-time iteration process. The fixed-point iteration is finding the mode of KDE for the improvement of step-time response iteration. In Equation (13), where applied iteratively for some convergence Δp is met until. The two-step strategy to comprise the constrained shape model into optimization is as follows: i) compute the mean-shift update for each deformable fitting landmark (PDM) and ii) using Least Square fit is Equation (14) of PDM parameterization is hold from the updated of constrained Mean Shift landmark.

$$Q\left(p\right)=\sum_{i=1}^{n}\left\|x_{i}-x_{i}^{\left(\tau+1\right)}\right\|^{2} \tag{14}$$

The Q-function of the M-step is in Equation (15) form by using the linear shape model in Equation (8) and global objective maximizing of PDM landmark in Equation (6) using the expectation-maximization (EM) algorithm (Kanade L. G., 2008) is:

$$\Delta p=J^{\dagger}\left[x_{1}^{\left(\tau+1\right)}-x_{1}^{c};\cdots\cdots;x_{n}^{\left(\tau+1\right)}-x_{n}^{c}\right] \tag{15}$$

Where $J^{\dagger}$ denotes the pseudo-inverse of J, and $x_{i}^{\left(\tau+1\right)}$ is denoted as the i^{th} landmark of mean shift, and update parameters are given in Equation (13). This is simply the constrained model fitting of Subspace Constrained Mean Shift (SCMS) by using the updated least squares mean shift of PDM landmarks and sharing the EM algorithm properties in Equation (15) with the KDE representation of maximizing optimization that is in Equation (6), to attain a CLM fitting of the face that is improved and converged. The procedure of completing a Subspace Constrained Mean Shift (SCMS) is summarized in Algorithm (Simon, Jeffrey, & Jason, 2011).

Expression Classification

Binary Class of Support Vector Machines (SVM) with FAPs Detection

For facial expression classification, we formulate the Support Vector Machine with Facial animation parameters of extracted feature points. Support Vector Machine (SVM) is a linear separating a maximum margin of hyperplane in a higher dimensionality space (Naumovich, 1998). In our work, we formulate the modified two classes of SVM for the six basic emotions. Let

$$g_{j}=\left\{(\vec{x}_{i},\vec{y}_{i})\right\};i=1\ldots\ldots k;\vec{x}\in\Re^{n};y_{i}\in\left\{-1,+1\right\}$$

are the training data of extracted feature points of facial expressions. Then separating the hyperplane of linear data of the form is:

$$\vec{w}^{T}\cdot\vec{x}_{i}+b=0 \tag{16}$$

separating the grid deformation feature vector $g_{j}=\Re^{L}$ of positive and negative class data in a form,

$$\vec{w}^T.\vec{x}_i + b \geq 0 \, for \left(y_i = +1\right)$$
$$\vec{w}^T.\vec{x}_i + b < 0 \, for \left(y_i = -1\right) \tag{17}$$

Wherever $\vec{w}^T$ is weight vector, where normal to the separating hyperplane and $\vec{w}^T$ is a bias, which is the perpendicular distance between the origin from the hyperplane. The decision functions of separating hyperplane:

$$f(\vec{x}) = \vec{w}^T \cdot \vec{x} + b \tag{18}$$

The training data have more than one possibility for separating hyperplane. To find out the optimal hyperplane, applying the concept of maximum margin of separating hyperplane on training data, which form in Equation (19):

$$\vec{w}^T \cdot \vec{x} + b \geq +1 \, for \left(y_i = +1\right)$$
$$\vec{w}^T \cdot \vec{x} + b \leq -1 \, for \left(y_i = -1\right) \tag{19}$$

and, subject to constraint inequalities, is $y_i \left(\vec{w}^T \cdot \vec{x}_i + b\right) - 1 \geq 0$, $i = 1, ...N$

Linear Case

From Equation (19), assume that the possible training data and negative training data are linearly separable. To find the optimal hyperplane, where apply the quadratic optimization of linearly separable of training data. In this quadratic cost function of linearly data (Ventura, 2009), hyperplane is separated by minimum weight vector of positive class data and maximum weight vector of the negative class data.

$$\vec{w} = \sum_{i=1} \alpha_i \cdot \tilde{S}_i \tag{22}$$

Equation (22) gets the values (where weight vector of hyperplane and bias are, respectively), from our augmented weighted vector with a bias, for the linear optimal hyperplane separating. The two classes of linear SVM of the decision surface are:

$$f\left(x\right) = \sigma \left(\sum_{i=1} \alpha_i \Phi\left(\tilde{S}_i\right) \cdot \Phi\left(x\right) \right) or \, y = \vec{w} \cdot x + b \tag{23}$$

From the figure shown, optimal hyperplane with a maximum margin and Equation (23) give the discriminating hyperplane of a separating cluster in a decision surface. For *Non Linear case SVM*, the training data are changed into linear data by using normalization and the transformation of Φ mapping function (Devi, 2011) and (Ventura, 2009).

$$\begin{aligned} &Linear: K(x_i, x_j) = \left(x_i \cdot x_j\right) \\ &Polynomial: K(x_i, x_j) = (x_i \cdot x_j + \sigma)^d \\ &RBF: K(x_i, x_j) = \exp\left(-\frac{\left\|x_i - x_j\right\|^2}{2\sigma^2}\right) \end{aligned} \tag{24}$$

Equation (23) gives the result of classifying the six emotions in the human face by the test image sequence as detailed in the fourth section.

Data Scaling

In Happy, the real time data of minimal feature vector displacements of the corner lip mouth region (Group 2) was implemented in the decision surface of the happy train model. From the results of the deformable feature vector being applied, Min-Max and Z-form Normalization (25) are the best approaches in data scaling. Then the values were substituted in various multiclass SVM approaches for data validation.

$$\begin{aligned} &Min\text{-}Max_{(-1,1)}: f(x) = 2\left(\frac{x - Max(x)}{Max(x) - Min(x)}\right) - 1 \\ &Z\text{-}Norm: f(x) = \left(\frac{x - \mu}{\sigma}\right) \end{aligned} \tag{25}$$

EXPERIMENTAL AND RESULTS OF AUTOMATIC FACIAL EXPRESSION

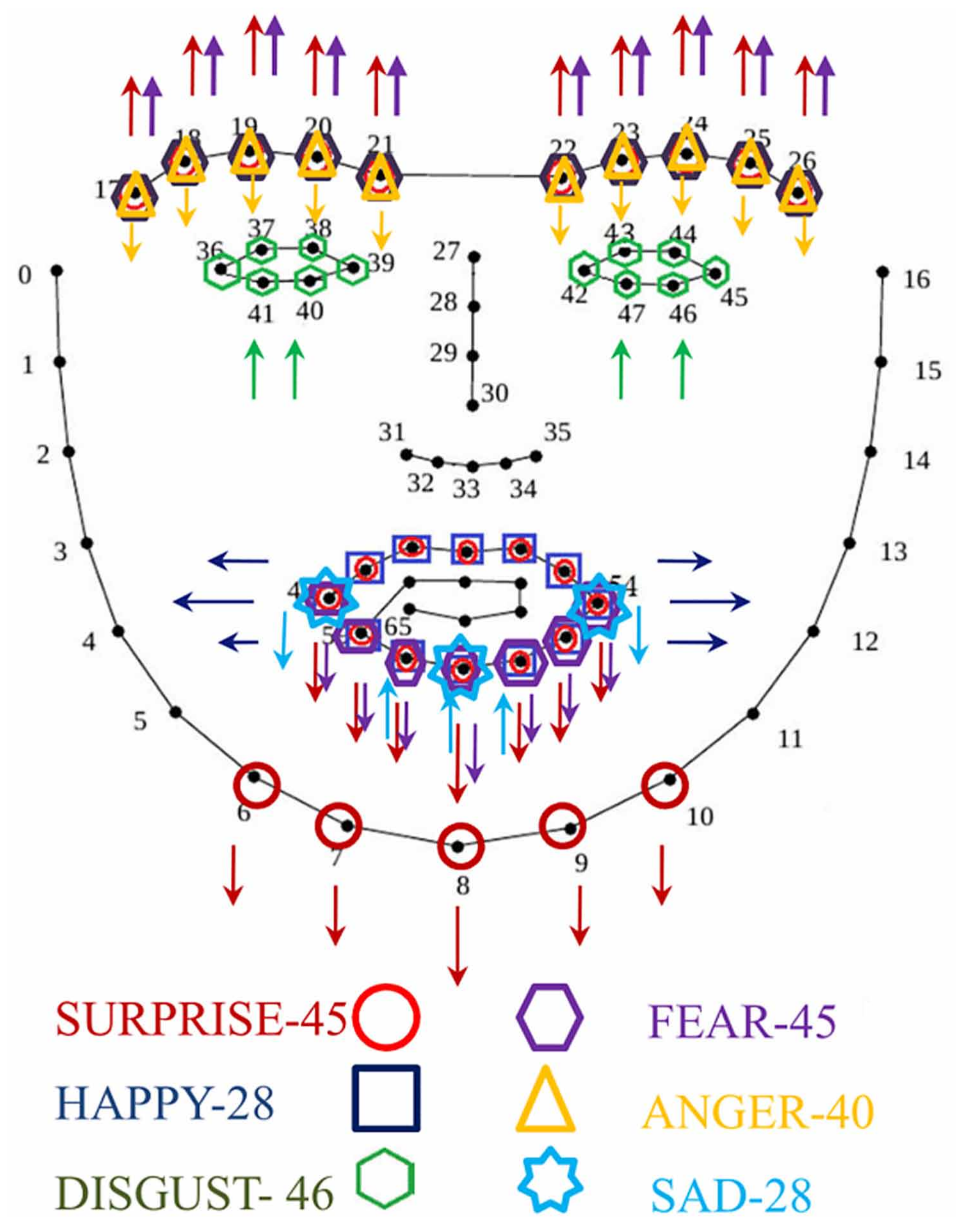
Extraction of Entire Feature Vector Displacement

From facial expressions, datasets are captured by the deformable grid node, and the feature displacement information is extracted. The geometric information of feature displacement is one node displacement, $d_{i,j}$, defined as the consecutive, frame-by-frame difference between the grid node displacements of the first to i^th node coordinates and fully formed the facial expression video frames.

$$d_{i,j} = \begin{bmatrix} \Delta x_{i,j} \\ \Delta y_{i,j} \end{bmatrix} = \begin{pmatrix} a_{11} - a_{12} & a_{12} - a_{13} & \cdots & a_{1,j} - a_{1,j+1} \\ a_{21} - a_{22} & a_{22} - a_{23} & \cdots & a_{2,j} - a_{2,j+1} \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ a_{i,j} - a_{i,j+1} & \cdots & \cdots & a_{n,m+1} - a_{n,m+2} \end{pmatrix} \tag{24}$$

$i = 1,.......F, j = 1,.......N$ Where $\Delta x_{i,j}, \Delta y_{i,j}$ are x and y axis coordinates of grid node displacement of the i^th node in the j^th frame image, respectively. *F* is the number of grid node (*F*=66 nodes of Constrained Local Model) and *N* is the number of the extracted facial images from the training facial expression database image sequence. In the facial expression database, the image sequence starts at neutral to one of emotion and turns to the neutral face state. For every sequence of the facial expression images in the training data set, an *extracted feature vector* g_j is created from the displacements of the geometric of every grid node $d_{i,j}$, g_j is called the *grid deformation feature vector*.

Figure 5. Constrained Local Model (CLM) of extracted features in single frame



$$g_j = \begin{bmatrix} d_{1,j} & d_{2,j} & \cdots & d_{E,j} \end{bmatrix}^T ; j = 1, ...N \tag{25}$$

In the geometric deformable grid node, CLM has L = 66*2 = 132 dimensions. In the feature vector g_j of image sequence, the $d_{i,j}$ displacements of the grid form neutral face to expressed state face (i.e., initial frame to peak response of frame) are computed, and the expressed face to neutral state are neglected. The feature vector g_j is now employed for the classification of the six basic emotions in humans, using the two classes of Support Vector Machines (SVM).

The feature vector g_j of the image sequence database is defined in each expression of facial movement, which parameterized the model of Facial Animation Parameters description (Ostermann, 2000), as shown in Table 1.

Table 1. Description of facial expression by facial animation parameters

Expression	Textual Description
Surprise	The eyebrows are raised. The upper eyelids are wide open, the lower relaxed. The jaw is opened.
Happy	The eyebrows are relaxed. The mouth is open and the mouth's corners are pulled back toward the ears.
Disgust	The eyebrows and eyelids are relaxed. The upper lip is raised and curled, often asymmetrically.
Fear	The eyebrows are raised and pulled together. The inner eyebrows are bent upward. The eyes are tense and alert.
Anger	The inner eyebrows are pulled downward and together. The eyes are wide open. The lips are pressed against each other or opened to expose the teeth.
Sad	The inner eyebrows are bent upward. The eyes are slightly closed. The mouth is relaxed.

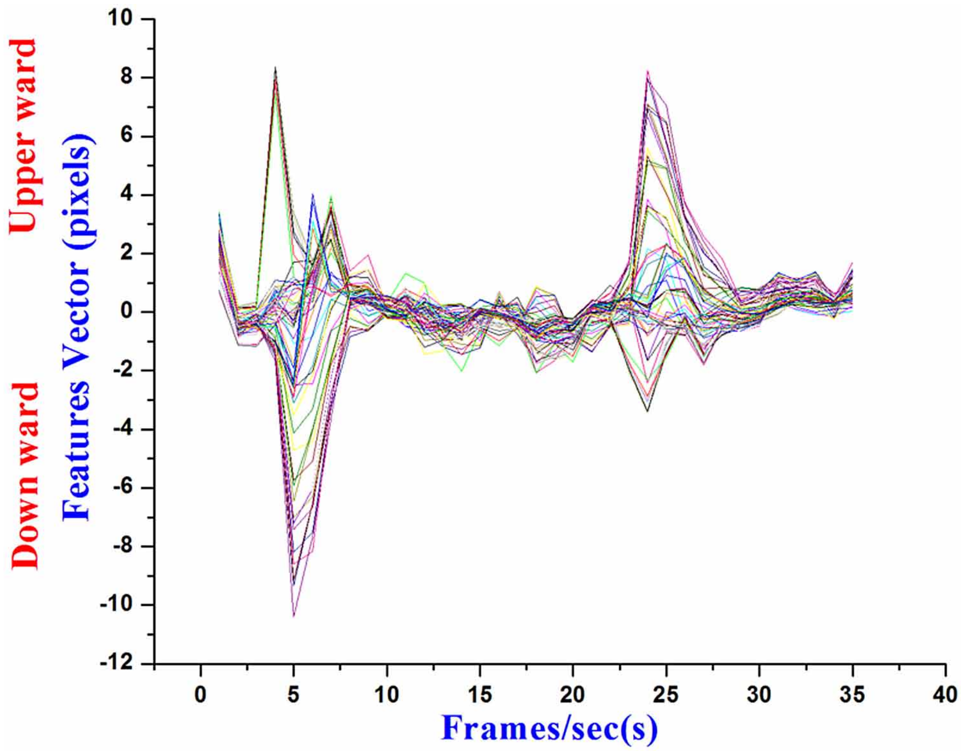
The FAPs descriptions employ the six basic emotions (surprise, happy, disgust, fear, anger, and sad) of the automatic facial expression system.

Surprise

In surprise, the displacements of the feature vector were formed by the captured image sequence. The major facial movements of surprise were in Group 4 (eyebrows) and Group 8 (outer lip/mouth) described from the FAPs. Figure 6a shows the expressive episode time of all the entire feature vectors of surprise emotions. In Group 4 (eyebrow) there are 10 features points and Group 8 (outer lip/mouth) has 12 feature points. First, the displacements of feature vectors of the eyebrow region (10 feature points) are demonstrated in Figure 6b. Figure 6b shows the face expression start at neutral to peak and return to neutral state of feature vector displacement, which is called expression episode time.

Next, the major role of facial movement in surprise was Group 8 (i.e., the outer lip/mouth region has 12 features points) and its entire feature vector displacement as shown in Figure (6c). From the entire feature vector displacment of the mouth region in surprise, six feature vectors (i.e., three features in the inner lip and three features in the outer lip) have a high peak response.

Figure 6. a) 66-feature vector of surprise emotions; b) Entire feature vector of surprise eyebrow (10 fps); c) surprise outer lip/mouth (18 fps)



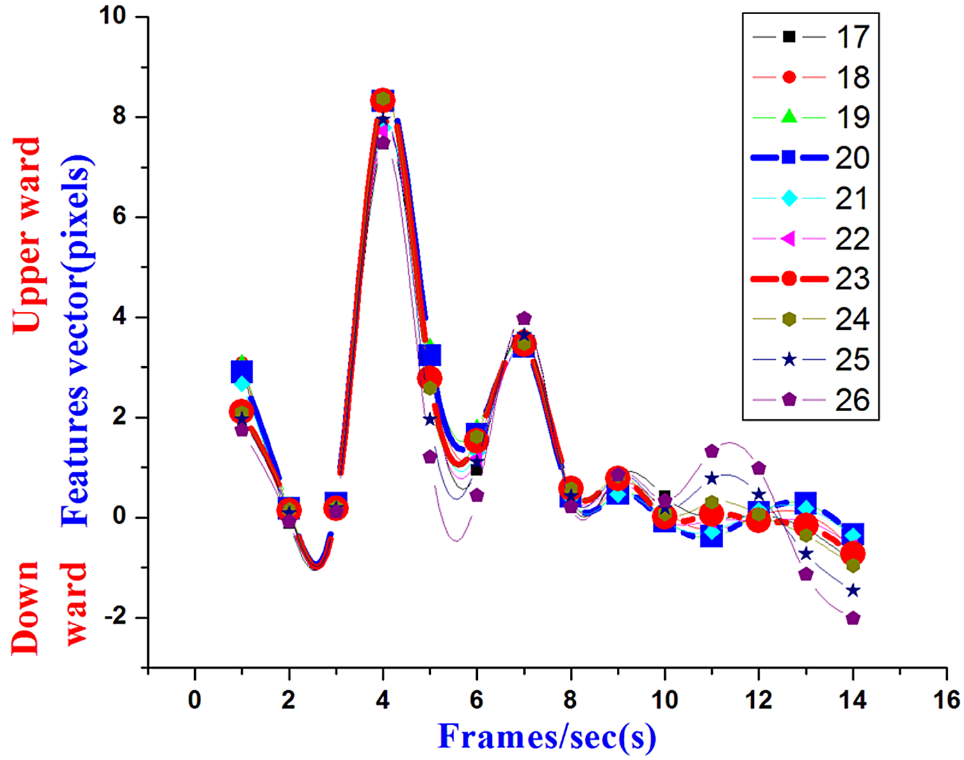
Happy

In happy, the major facial muscle movement in Group 8 (outer lip/mouth region) and Group 2 (corner lip region) of temporal segments of the entire feature vector displacment is shown in Figures 7a and 7b. Figure 7a shows the expressive episode time of all entire feature vectors of happy emotions. In the entire feature vector displacement of happy, the outer lip region (12 features points) was horizantally expanded, which had a high peak response in Figure 7b.

Disgust

In disgust, the facial muscle movement was observed in Group 3 (eyelid region) of temporal segments, as shown in Figures 8a and 8b. Figure 8a shows the expressive episode time of all entire feature vectors of disgust emotions. The entire feature vector displacements of disgust in the eyelid have 12 features points, as shown in Figure 8b. From the Figure 8b, a high peak response (i.e., expressive episode time) at initial denotes a disgust expression, and a spike response at offset indicates the eye blink region, which has to be neglected.

Figure 7. a) 66-feature vector of happy emotion; b) Entire feature vector of happy mouth (18 fps)



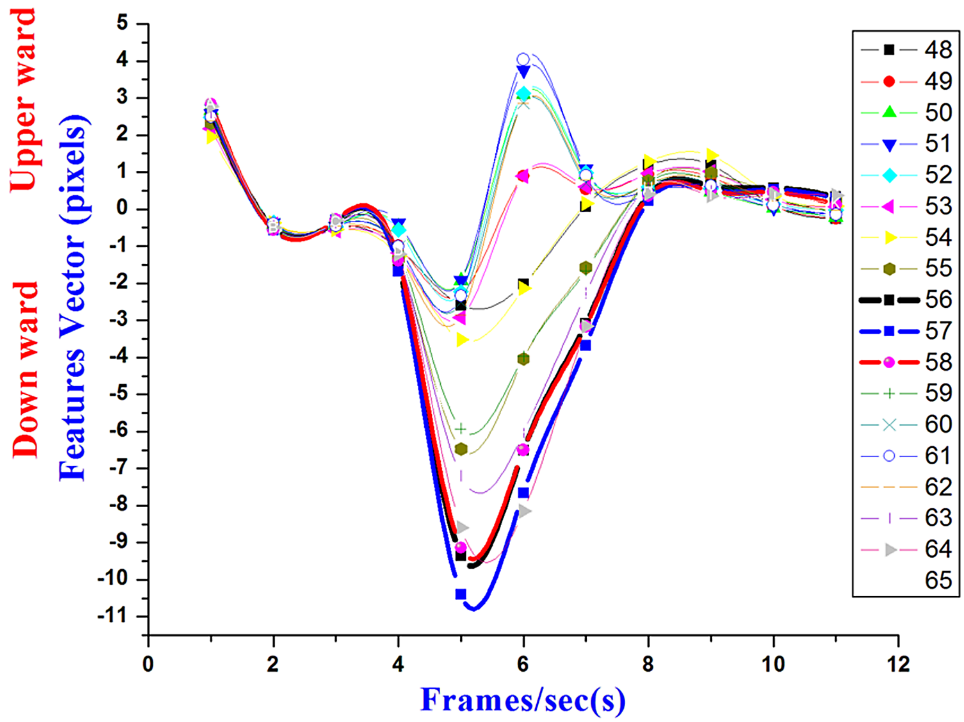
Fear

In fear, the displacements of the feature vector were formed by the captured image sequence. The major facial movements of fear in Group 4 (eyebrows) and Group 8 (outer lip/mouth) were described in the FAPs. Figure 9a shows the expressive episode time of all entire feature vectors of disgust emotions. In Group 4 (eyebrow), there are 10 features points, and Group 8 (outer lip/mouth) has 12 feature points. First, the displacements of the feature vectors of the eyebrow region (10 feature points) are demonstrated in Figure 9b. In fear, mouth is in Group 8 (i.e., outer lip/mouth region has 12 features points), and its entire feature vector displacement is shown in Figure 9c.

Anger

In anger, the displacements of the feature vector were formed by the captured image sequence. The major facial movements of fear in Group 4 (eyebrows) were described from the FAPs. Figure 10a shows the expressive episode time of all the entire feature vectors of anger emotions. In Group 4 (eyebrow), there are 10 features points. The displacements of the entire feature vectors of the eyebrow region (10 feature points) are demonstrated in Figure 10b.

Figure 8. a) 66-feature vector of disgust emotion; b) Entire feature vector of disgust in eyelids region (12 fps)



Sad

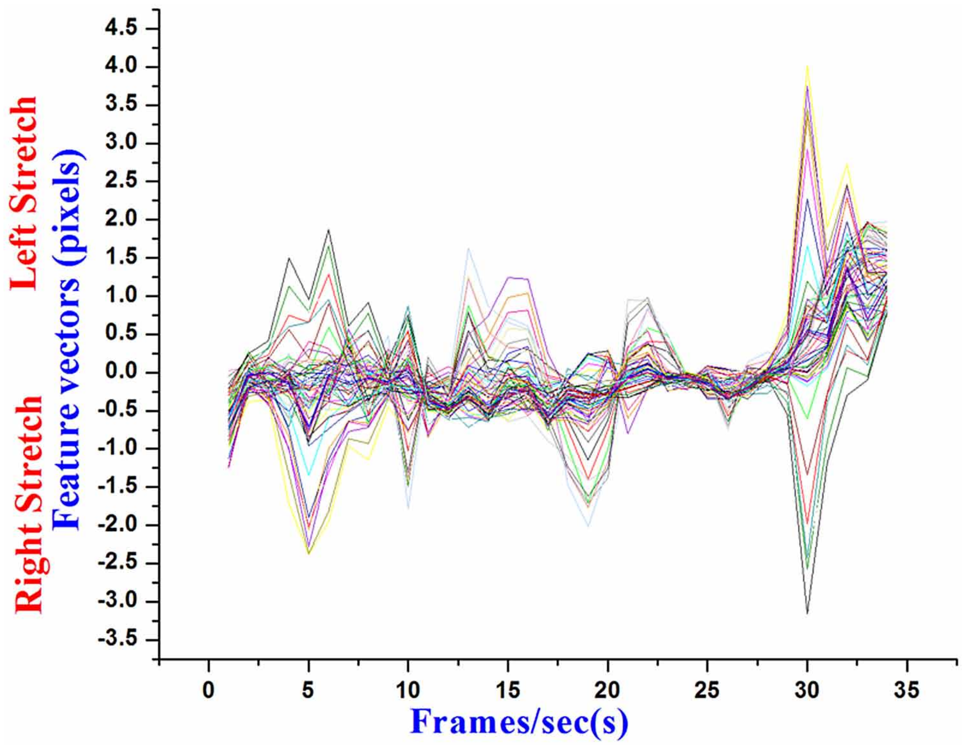
In sad, the displacements of the feature vector were formed by the captured image sequence. In sad, the major facial muscle movement in Group 8 (outer lip/mouth region) and Group 2 (corner lip region) of temporal segments of the entire feature vector displacment is shown in Figure 11a and 11b. Figure 11a shows the expressive episode time of all the entire feature vectors of sad emotions. In the entire feature vector displacement of sad, the outer lip region (12 features points) was horizantally expanded, which had a high peak response in Figure 11b.

Experimental Analysis of Facial Expression System

Experimental Setup

This proposed system is performed based on the displacement of the geometrical information. The geometrical deformable model is extracted from the displacement of feature points using logistic regression. The displacement of the geometric model (CLM) of feature points is used for the classification of facial expression by two classes of Support Vector Machines (SVM) with the FAPs parameterized model. For classification, the extracted displacement of geometric information is employing a set of rules to record them as six emotions in the human face. The geometric deformable model (CLM) (Simon, Jeffrey, & Jason, 2011) is developed in C++ with open framework tools. A Support Vector Machine by LIBSVM

Figure 9. a) 66-feature vector of fear emotions; b) Entire feature vector of fear eyebrow (10 fps); c) Fear outer lip/mouth (18 fps)

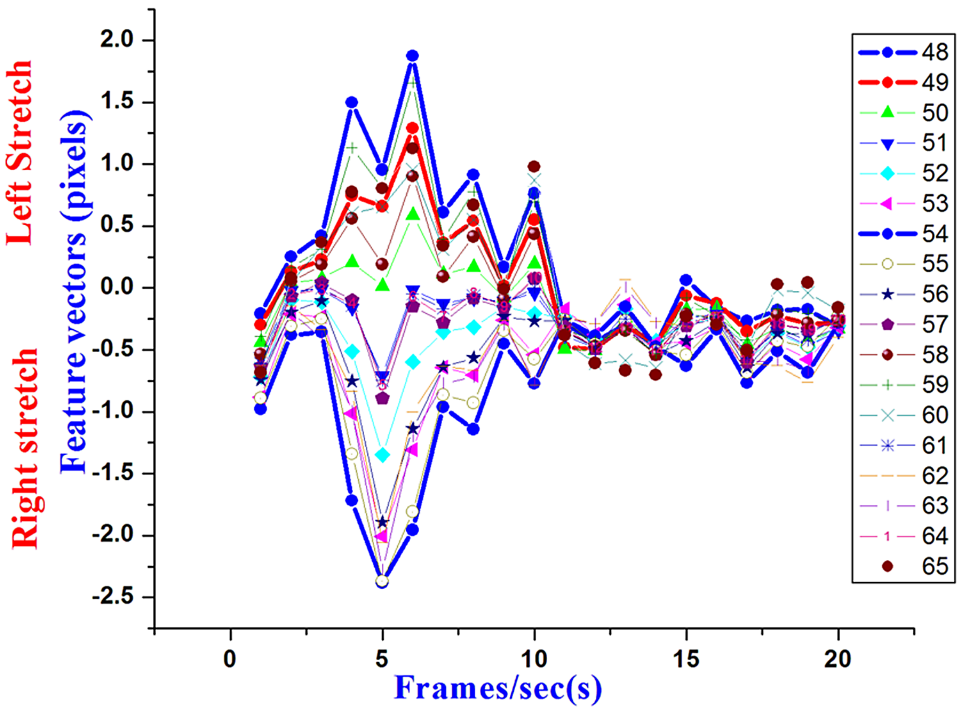


binary was implemented in an Intel i5 processor with the matlab tool. In this proposed model, we used the MMI facial expression database (Pantic., 2010.), the Oulu Database (Zhao, 2011), a CK (Kanade T. J., 2000), an Extended CK+ database (Lucey, 2010), and the Mahnob Laughter database (Petridis, 2013) for our training data of an automatic facial expression system. It consists of six basic emotion faces (*surprise, happy, disgust, fear, angry, and sad*) of various genders, ages, and countries. The video rate is 30 frames/s and only frontal face image sequences are captured, as shown in Figure 12.

In our training process, g_j is the feature vector of the extracted facial image sequence, as $d_{i,j}$ g_j emotions in the face. The feature vector was defined based on the FAPs description of face emotions. In this proposed model, details about the descriptions of the facial expression movement of each emotion on feature extraction are as follows:

- MMI database (Pantic., 2010.)
- Cohn-Kanade database (Kanade T. J., 2000)
- Extended-CK data*b*ase (Lucey, 2010)
- Oulu database (Zhao, 2011)
- MAHNOB-Laughter (Petridis, 2013)
- Real-Time data (Kumar & Rajagopal, 2019)

Figure 10. a) 66-feature vector of anger emotion; b) Entire feature vector of anger in eyebrow region (10 fps)

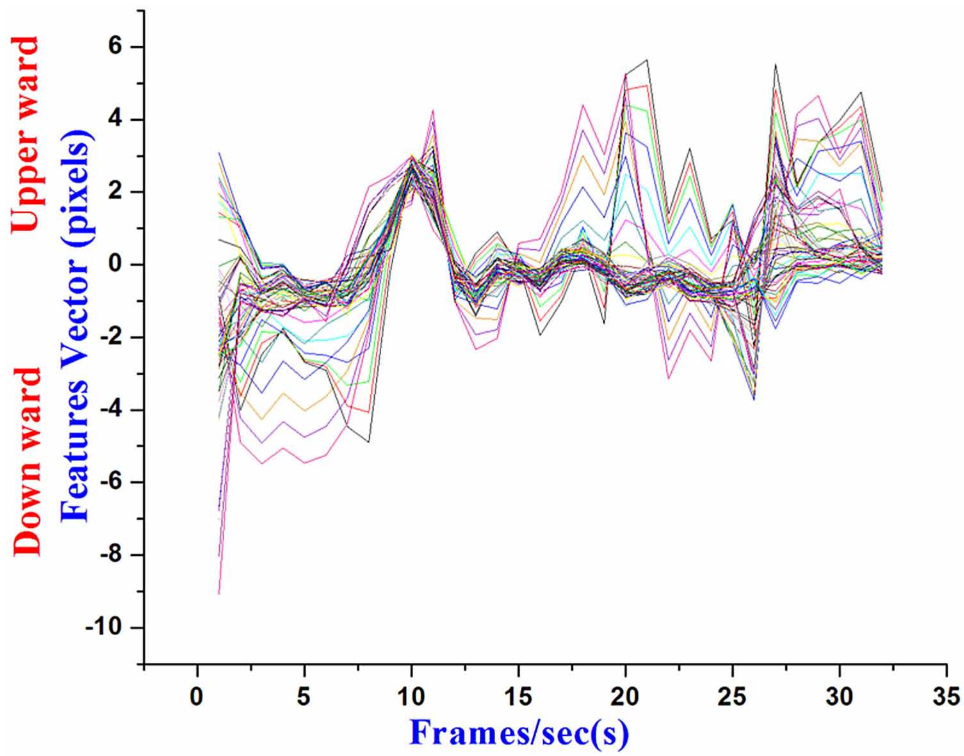


The training and testing process used four standard databases and one real-time database (Kumar & Rajagopal, 2019). This system used 1,602 subjects of different emotions. Both the training and testing phase applied hold-out cross validations, and the data were randomly split to the training model and to testing.

Training and Testing Hold-out Validation

The hold-out cross-validation was applied in both the training and testing phases. A new database was formed by fusing existing standard databases, such as MMI, Oulu, CK, CK+, and Mahnob, and real-time datasets. The global, normalized, minimal feature vector of the new database was given to hold-out validation. In hold-out validation, 80% of normalized datasets are given to the training phase and the remaining 20% of the datasets is taken for the testing phase for validating the facial emotion classifier. In validation, the multi-class SVM of three kernel cases (linear, polynomial, and RBF) with both normalizations (Max & Min and Z-norm) are evaluated. In hold-out validation, three kernel cases of penalty and kernel parameters values such as c1, c2, and $\sigma = 0.5$ are applied. The Confusion Matrix of "One vs. One" and "One vs. All" is employed in both phases. From the results of the Confusion Matrix, validation parameters such as True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN), Accuracy (Acc), and so forth, are calculated. For all emotions, 18 trained models of multi-classes ("One vs. One" and "One vs. All," with three kernels), with one normalization method, are calculated. Then 36 multi-class, trained models are calculated using TWSVM with two normalization techniques

Figure 11. a) 66-feature vector of sad emotion; b) Entire feature vector of sad in the mouth region (18 fps)



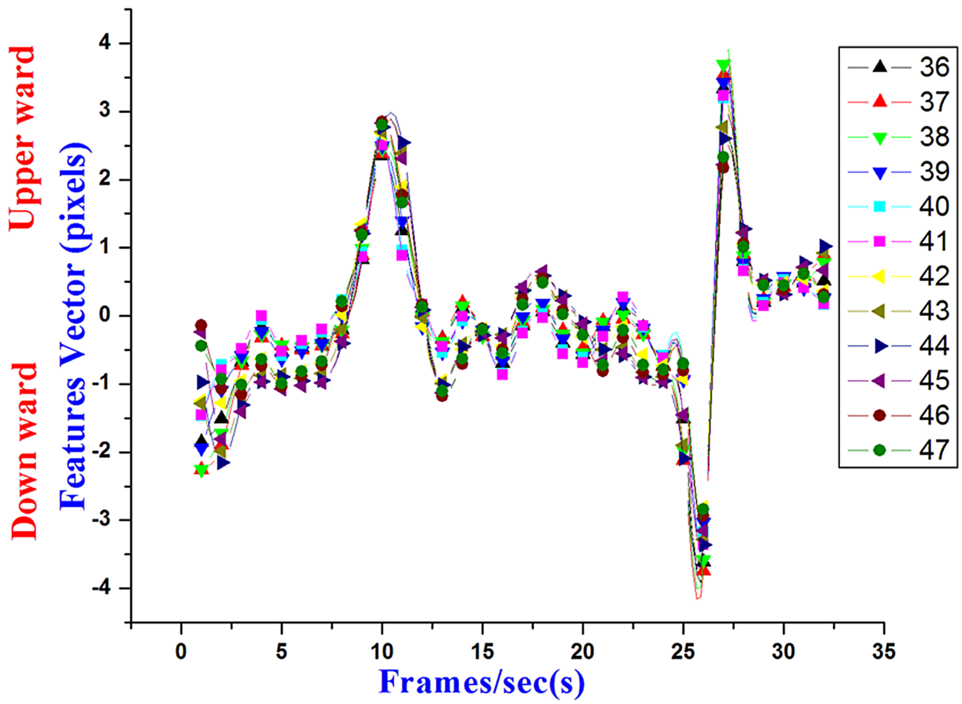
and compared, to determine the performance. Also, the computation time of the training and testing phases of all basic emotions are calculated. In this work, a supervised SVM multi classifier is used by the LIBSVM library (Chang, 2011) for detecting all emotion.

From the 36 trained models, all entire features vectors of the six basic emotions are developed. The test data given in the 36 trained models, and evaluated, are shown in Tables 2a–2h. In these tables, all validation parameters are calculated and shown. In Tables 2a–2h, "One vs. One" has lower performance than the "One vs. All." From these, it is inferred that the RBF kernel in "One vs All" multi-class of the Z-normalized entire feature vector has achieved a higher validation performance than the other kernel in "One vs All" and is calculated as shown in Table 3. In Table 3, RBF Kernel has high accuracy at 88.45±2.7%, high F1-score at 0.57±0.23, less error rate at 0.11±0.05, and less computational time at 0.31±0.17 sec, compared with other kernels and normalizations.

CONCLUSION

In human–computer interaction, researchers are more interested in developing the communication between humans and computers and determining which emotion has a more significant role among their applications. For HCI applications, there is a need to develop automatic facial expression recognition systems that are robust, less computational, highly accurate, and have good reliability. This chapter proposed the real-time facial expression recognition system of the entire face feature in the facial image sequence. In

Figure 12. Training model of MMI, CK, CK+, Oulu, Mahnob, and few real-time data from facial expression databases



this proposed system, the training phase accomplished the face identification and tracking in the frame by frame of a standard database video, with the help of CLM, and attained good tracking of the frontal face. Consequently, the facial features of CLM were extracted and the entire and minimal feature vectors were developed. The features vector displacement is employed in Support Vector Machines with Facial Animation Parameters (FAPs) for accomplishing the trained model of the Facial Expression Classification system. Although, in testing the process is similar, the real-time facial video of face detection and tracking and feature extraction are implemented by CLM, and entire feature vectors are developed.

In this proposed system, the entire facial feature vector has a RBF Kernel with high accuracy, at 88.45±2.7%; a high F1-score, at 0.57±0.23; a lower error rate, at 0.11±0.05; and less computational time, at 0.31±0.17 sec; all are compared with other kernels and normalizations. And, to conclude, our future work will include head pose variations, micro facial emotion recognition, and also a profile face image sequence (i.e., side view of face image) for developing, which will be helpful in the Human–Computer Interaction application.

Table 2a. Validation result of all emotions: Surprise eyebrow

	Norm	One vs One	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
One vs One	Max & Min	Sur vs Ang	Lin	31.91	0.15	0.03	0.06	0.97	0.073
		Sur vs Dis	Lin	36.18	0.37	0.08	0.14	0.92	0.125
		Sur vs Fea	Lin	37.69	0.41	0.09	0.15	0.91	0.053
		Sur vs Hap	Lin	36.68	0.54	0.11	0.19	0.89	0.061
		Sur vs Sad	Lin	29.15	0.39	0.08	0.13	0.92	0.07
		Sur vs Ang	Poly	39.95	0.5	0.11	0.18	0.89	0.061
		Sur vs Dis	Poly	54.77	0.46	0.14	0.22	0.86	0.039
		Sur vs Fea	Poly	57.79	0.44	0.15	0.22	0.85	0.038
		Sur vs Hap	Poly	36.18	0.67	0.13	0.22	0.87	0.037
		Sur vs Sad	Poly	40.95	0.59	0.13	0.21	0.87	0.04
		Sur vs Ang	RBF	43.97	0.15	0.04	0.07	0.96	0.052
		Sur vs Dis	RBF	30.9	0.33	0.07	0.12	0.93	0.035
		Sur vs Fea	RBF	35.93	0.35	0.08	0.13	0.92	0.038
		Sur vs Hap	RBF	68.84	0.43	0.2	0.27	0.8	0.033
		Sur vs Sad	RBF	53.77	0.28	0.09	0.14	0.91	0.039
	Z-norm	Sur vs Ang	Lin	91.46	0.59	0.73	0.65	0.27	0.042
		Sur vs Dis	Lin	88.44	0.72	0.56	0.63	0.44	0.05
		Sur vs Fea	Lin	87.19	0.74	0.52	0.61	0.48	0.056
		Sur vs Hap	Lin	90.7	0.57	0.69	0.63	0.31	0.107
		Sur vs Sad	Lin	89.95	0.67	0.62	0.64	0.38	0.047
		Sur vs Ang	Poly	89.95	0.35	0.79	0.49	0.21	0.048
		Sur vs Dis	Poly	90.7	0.54	0.71	0.61	0.29	0.036
		Sur vs Fea	Poly	90.7	0.54	0.71	0.61	0.29	0.033
		Sur vs Hap	Poly	67.09	0.69	0.25	0.36	0.75	9.706
		Sur vs Sad	Poly	90.45	0.46	0.74	0.57	0.26	0.036
		Sur vs Ang	RBF	88.44	0.72	0.56	0.63	0.44	0.064
		Sur vs Dis	RBF	82.16	0.87	0.42	0.57	0.58	0.045
		Sur vs Fea	RBF	76.88	0.91	0.36	0.52	0.64	0.044
		Sur vs Hap	RBF	88.44	0.7	0.56	0.62	0.44	0.041
		Sur vs Sad	RBF	85.93	0.76	0.49	0.59	0.51	0.047
One vs All	Norm	One vs All	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
	Max & Min	Sur vs All	Lin	89.95	0.56	0.71	0.63	0.29	0.154
		Sur vs All	Poly	87.19	0.13	0.64	0.22	0.36	0.085
		Sur vs All	RBF	89.95	0.55	0.7	0.62	0.3	0.087
	Z-norm	Sur vs All	Lin	90.45	0.59	0.71	0.58	0.26	0.088
		Sur vs All	Poly	87.19	0.13	0.64	0.48	0.14	10.46
		Sur vs All	RBF	90.45	0.59	0.73	0.65	0.3	0.109

Table 2b. Validation result of all emotions: Surprise mouth

	Norm	One vs One	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
One vs One	Max & Min	Sur vs Ang	Lin	75.32	0.72	0.32	0.45	0.68	0.1
		Sur vs Dis	Lin	74.81	0.72	0.32	0.44	0.68	0.36
		Sur vs Fea	Lin	72.52	0.78	0.3	0.44	0.7	0.07
		Sur vs Hap	Lin	83.21	0.67	0.43	0.52	0.57	0.08
		Sur vs Sad	Lin	71.25	0.72	0.28	0.41	0.72	0.08
		Sur vs Ang	Poly	69.47	0.78	0.28	0.41	0.72	0.08
		Sur vs Dis	Poly	71.25	0.74	0.29	0.41	0.71	0.06
		Sur vs Fea	Poly	73.28	0.76	0.31	0.44	0.69	0.05
		Sur vs Hap	Poly	80.92	0.65	0.38	0.48	0.62	0.05
		Sur vs Sad	Poly	70.99	0.76	0.29	0.42	0.71	0.06
		Sur vs Ang	RBF	40.46	0.48	0.11	0.18	0.89	0.07
		Sur vs Dis	RBF	47.07	0.5	0.13	0.21	0.87	0.11
		Sur vs Fea	RBF	38.42	0.56	0.12	0.2	0.88	0.05
		Sur vs Hap	RBF	39.44	0.57	0.13	0.21	0.87	0.05
		Sur vs Sad	RBF	44.78	0.5	0.12	0.2	0.88	0.05
	Z-norm	Sur vs Ang	Lin	83.46	0.51	0.41	0.45	0.59	0.31
		Sur vs Dis	Lin	76.84	0.74	0.34	0.46	0.66	0.32
		Sur vs Fea	Lin	75.83	0.75	0.33	0.46	0.67	0.12
		Sur vs Hap	Lin	84.99	0.77	0.47	0.58	0.53	0.26
		Sur vs Sad	Lin	84.73	0.64	0.45	0.53	0.55	0.6
		Sur vs Ang	Poly	73.79	0.72	0.3	0.42	0.7	0.11
		Sur vs Dis	Poly	73.79	0.7	0.3	0.42	0.7	0.07
		Sur vs Fea	Poly	79.39	0.62	0.35	0.45	0.65	0.06
		Sur vs Hap	Poly	82.44	0.6	0.4	0.48	0.6	0.06
		Sur vs Sad	Poly	74.3	0.7	0.3	0.42	0.7	0.07
		Sur vs Ang	RBF	41.48	0.45	0.11	0.17	0.89	0.07
		Sur vs Dis	RBF	47.33	0.34	0.09	0.15	0.91	0.05
		Sur vs Fea	RBF	47.84	0.36	0.1	0.16	0.9	0.06
		Sur vs Hap	RBF	53.18	0.45	0.13	0.21	0.87	0.05
		Sur vs Sad	RBF	41.48	0.36	0.09	0.14	0.91	0.05
One vs All	Norm	One vs All	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
	Max & Min	Sur vs All	Lin	91.35	0.41	0.92	0.56	0.08	0.12
		Sur vs All	Poly	89.57	0.24	1	0.39	0	0.12
		Sur vs All	RBF	91.35	0.41	0.92	0.56	0.08	0.12
	Z-norm	Sur vs All	Lin	90.84	0.63	0.94	0.75	0.06	0.14
		Sur vs All	Poly	91.35	0.55	0.88	0.67	0.12	9.26
		Sur vs All	RBF	91.86	0.67	0.97	0.79	0.03	0.16

Table 2c. Validation result of all emotions: Happy mouth

	Norm	One vs One	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
One vs One	Max & Min	Hap vs Ang	Lin	48.48	0.38	0.22	0.28	0.78	0.15
		Hap vs Dis	Lin	46.19	0.44	0.23	0.3	0.77	0.16
		Hap vs Fea	Lin	44.67	0.5	0.24	0.32	0.76	0.15
		Hap vs Sad	Lin	48.22	0.39	0.23	0.29	0.77	0.16
		Hap vs Sur	Lin	48.73	0.44	0.24	0.31	0.76	0.16
		Hap vs Ang	Poly	44.67	0.3	0.18	0.23	0.82	0.15
		Hap vs Dis	Poly	43.91	0.32	0.18	0.24	0.82	0.17
		Hap vs Fea	Poly	43.4	0.34	0.19	0.24	0.81	0.16
		Hap vs Sad	Poly	44.67	0.31	0.18	0.23	0.82	0.17
		Hap vs Sur	Poly	46.7	0.31	0.19	0.24	0.81	0.16
		Hap vs Ang	RBF	51.27	0.56	0.29	0.38	0.71	0.18
		Hap vs Dis	RBF	53.3	0.6	0.31	0.41	0.69	0.16
		Hap vs Fea	RBF	50.76	0.61	0.29	0.4	0.71	0.14
		Hap vs Sad	RBF	50.51	0.57	0.29	0.38	0.71	0.17
		Hap vs Sur	RBF	45.94	0.64	0.28	0.39	0.72	0.16
	Z-norm	Hap vs Ang	Lin	24.11	0.9	0.25	0.39	0.75	0.08
		Hap vs Dis	Lin	66.5	0.19	0.3	0.23	0.7	0.06
		Hap vs Fea	Lin	62.69	0.11	0.18	0.14	0.82	0.07
		Hap vs Sad	Lin	13.2	0.48	0.15	0.23	0.85	0.07
		Hap vs Sur	Lin	81.22	0.31	0.97	0.47	0.03	0.06
		Hap vs Ang	Poly	26.65	0.8	0.24	0.37	0.76	0.07
		Hap vs Dis	Poly	70.81	0.22	0.42	0.29	0.58	0.07
		Hap vs Fea	Poly	72.34	0.04	0.36	0.07	0.64	0.06
		Hap vs Sad	Poly	25.89	0.94	0.26	0.41	0.74	0.07
		Hap vs Sur	Poly	25.13	0.91	0.25	0.39	0.75	0.06
		Hap vs Ang	RBF	58.12	0.48	0.32	0.38	0.68	0.07
		Hap vs Dis	RBF	58.63	0.49	0.32	0.39	0.68	0.07
		Hap vs Fea	RBF	58.88	0.51	0.33	0.4	0.67	0.06
		Hap vs Sad	RBF	58.88	0.47	0.32	0.38	0.68	0.07
One vs All	Norm	One vs All	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
	Max & Min	Hap vs All	Lin	88.32	0.59	0.95	0.73	0.05	0.36
		Hap vs All	Poly	75.13	0.07	1	0.13	0	0.38
		Hap vs All	RBF	87.82	0.57	0.95	0.71	0.05	0.38
	Z-norm	Hap vs All	Lin	87.56	0.69	0.93	0.79	0.07	0.37
		Hap vs All	Poly	85.28	0.54	0.93	0.68	0.07	1.77
		Hap vs All	RBF	88.58	0.71	0.94	0.81	0.06	0.49

Table 2d. Validation result of all emotions: Disgust eyelids

	Norm	One vs One	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
One vs One	Max & Min	Dis vs Hap	Lin	55.58	0.79	0.22	0.34	0.78	0.07
		Dis vs Ang	Lin	58.63	0.67	0.21	0.32	0.79	0.06
		Dis vs Fea	Lin	57.87	0.67	0.21	0.31	0.79	0.06
		Dis vs Sad	Lin	68.27	0.46	0.22	0.29	0.78	0.05
		Dis vs Sur	Lin	56.09	0.68	0.2	0.31	0.8	0.06
		Dis vs Hap	Poly	54.57	0.79	0.21	0.33	0.79	0.04
		Dis vs Ang	Poly	57.36	0.65	0.2	0.31	0.8	0.04
		Dis vs Fea	Poly	58.38	0.65	0.2	0.31	0.8	0.05
		Dis vs Sad	Poly	67.01	0.46	0.21	0.29	0.79	0.04
		Dis vs Sur	Poly	55.58	0.67	0.2	0.3	0.8	0.04
		Dis vs Hap	RBF	56.85	0.4	0.14	0.21	0.86	0.04
		Dis vs Ang	RBF	53.3	0.54	0.16	0.25	0.84	0.04
		Dis vs Fea	RBF	51.78	0.46	0.14	0.21	0.86	0.06
		Dis vs Sad	RBF	43.65	0.51	0.13	0.21	0.87	0.04
		Dis vs Sur	RBF	55.33	0.47	0.16	0.23	0.84	0.04
	Z-norm	Dis vs Hap	Lin	71.83	0.67	0.29	0.41	0.71	0.21
		Dis vs Ang	Lin	83.5	0.56	0.44	0.5	0.56	0.17
		Dis vs Fea	Lin	84.01	0.65	0.46	0.54	0.54	0.19
		Dis vs Sad	Lin	78.43	0.47	0.33	0.39	0.67	0.26
		Dis vs Sur	Lin	72.34	0.72	0.31	0.43	0.69	0.17
		Dis vs Hap	Poly	59.9	0.72	0.22	0.34	0.78	0.05
		Dis vs Ang	Poly	59.64	0.65	0.21	0.32	0.79	0.05
		Dis vs Fea	Poly	60.15	0.7	0.22	0.34	0.78	0.08
		Dis vs Sad	Poly	69.29	0.61	0.26	0.37	0.74	0.05
		Dis vs Sur	Poly	60.15	0.7	0.22	0.34	0.78	0.06
		Dis vs Hap	RBF	58.63	0.35	0.14	0.2	0.86	0.04
		Dis vs Ang	RBF	53.81	0.46	0.15	0.22	0.85	0.04
		Dis vs Fea	RBF	51.27	0.37	0.12	0.18	0.88	0.06
		Dis vs Sad	RBF	51.02	0.44	0.13	0.21	0.87	0.04
		Dis vs Sur	RBF	56.6	0.35	0.13	0.19	0.87	0.04
One vs All	Norm	One vs All	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
	Max & Min	Dis vs All	Lin	87.06	0.11	1	0.19	0	0.19
		Dis vs All	Poly	87.31	0.12	1	0.22	0	0.13
		Dis vs All	RBF	87.06	0.11	1	0.19	0	0.12
	Z-norm	Dis vs All	Lin	86.8	0.21	0.63	2.53	0.37	0.12
		Dis vs All	Poly	86.8	0.3	0.71	2.83	0.29	0.99
		Dis vs All	RBF	87.06	0.23	0.68	2.74	0.32	0.14

Table 2e. Validation result of all emotions: Fear eyebrow

	Norm	One vs One	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
One vs One	Max & Min	Fea vs Hap	Lin	45.48	0.27	0.07	0.11	0.93	0.04
		Fea vs Ang	Lin	52.76	0.24	0.07	0.11	0.93	0.05
		Fea vs Dis	Lin	37.44	0.35	0.07	0.12	0.93	0.07
		Fea vs Sad	Lin	46.48	0.31	0.08	0.12	0.92	0.06
		Fea vs Sur	Lin	37.44	0.35	0.07	0.12	0.93	0.05
		Fea vs Hap	Poly	50	0.27	0.07	0.12	0.93	0.03
		Fea vs Ang	Poly	55.28	0.18	0.06	0.09	0.94	0.03
		Fea vs Dis	Poly	40.45	0.33	0.07	0.12	0.93	0.05
		Fea vs Sad	Poly	34.17	0.37	0.07	0.12	0.93	0.04
		Fea vs Sur	Poly	35.18	0.37	0.07	0.12	0.93	0.04
		Fea vs Hap	RBF	50	0.27	0.07	0.12	0.93	0.04
		Fea vs Ang	RBF	55.28	0.18	0.06	0.09	0.94	0.03
		Fea vs Dis	RBF	40.45	0.33	0.07	0.12	0.93	0.05
		Fea vs Sad	RBF	34.17	0.37	0.07	0.12	0.93	0.04
		Fea vs Sur	RBF	35.18	0.37	0.07	0.12	0.93	0.04
	Z-norm	Fea vs Hap	Lin	48.99	0.53	0.13	0.2	0.87	0.23
		Fea vs Ang	Lin	50.75	0.57	0.14	0.22	0.86	0.16
		Fea vs Dis	Lin	68.84	0.33	0.15	0.21	0.85	0.14
		Fea vs Sad	Lin	77.64	0.22	0.18	0.2	0.82	0.05
		Fea vs Sur	Lin	58.04	0.39	0.12	0.19	0.88	0.2
		Fea vs Hap	Poly	47.99	0.47	0.11	0.18	0.89	0.13
		Fea vs Ang	Poly	47.99	0.45	0.11	0.18	0.89	0.1
		Fea vs Dis	Poly	37.94	0.41	0.08	0.14	0.92	0.43
		Fea vs Sad	Poly	74.62	0.2	0.14	0.17	0.86	0.04
		Fea vs Sur	Poly	41.71	0.55	0.11	0.19	0.89	0.13
		Fea vs Hap	RBF	56.53	0.41	0.12	0.19	0.88	0.04
		Fea vs Ang	RBF	58.79	0.45	0.14	0.21	0.86	0.04
		Fea vs Dis	RBF	58.54	0.47	0.14	0.22	0.86	0.05
		Fea vs Sad	RBF	58.04	0.53	0.15	0.24	0.85	0.04
		Fea vs Sur	RBF	62.81	0.43	0.15	0.22	0.85	0.06
One vs All	Norm	One vs All	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
	Max & Min	Fea vs All	Lin	85.38	0.65	0.45	1.8	0.55	0.08
		Fea vs All	Poly	86.42	0.53	0.47	1.89	0.53	0.13
		Fea vs All	RBF	86.95	0.49	0.49	1.96	0.51	0.11
	Z-norm	Fea vs All	Lin	85.9	0.61	0.37	1.48	0.63	0.16
		Fea vs All	Poly	86.42	0.59	0.38	1.53	0.62	29.1
		Fea vs All	RBF	86.16	0.41	0.34	1.36	0.66	0.19

Table 2f. Validation result of all emotions: Fear mouth

	Norm	One vs One	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
One vs One	Max & Min	Fea vs Hap	Lin	59.8	0.62	0.24	0.35	0.76	0.42
		Fea vs Ang	Lin	55.98	0.48	0.13	0.21	0.87	0.13
		Fea vs Dis	Lin	61.32	0.46	0.15	0.22	0.85	0.12
		Fea vs Sad	Lin	69.47	0.44	0.18	0.26	0.82	0.12
		Fea vs Sur	Lin	54.71	0.48	0.13	0.21	0.87	0.12
		Fea vs Hap	Poly	55.47	0.56	0.15	0.24	0.85	0.06
		Fea vs Ang	Poly	49.62	0.73	0.16	0.26	0.84	0.05
		Fea vs Dis	Poly	55.22	0.67	0.17	0.27	0.83	0.07
		Fea vs Sad	Poly	70.48	0.4	0.18	0.25	0.82	0.05
		Fea vs Sur	Poly	55.98	0.56	0.15	0.24	0.85	0.06
		Fea vs Hap	RBF	53.69	0.38	0.11	0.17	0.89	0.05
		Fea vs Ang	RBF	48.09	0.4	0.1	0.16	0.9	0.05
		Fea vs Dis	RBF	45.04	0.5	0.11	0.18	0.89	0.07
		Fea vs Sad	RBF	41.98	0.58	0.12	0.2	0.88	0.05
		Fea vs Sur	RBF	57	0.4	0.12	0.18	0.88	0.05
	Z-norm	Fea vs Hap	Lin	55.73	0.42	0.12	0.19	0.88	0.23
		Fea vs Ang	Lin	54.2	0.33	0.1	0.15	0.9	0.34
		Fea vs Dis	Lin	64.38	0.25	0.1	0.15	0.9	0.53
		Fea vs Sad	Lin	70.23	0.17	0.09	0.12	0.91	0.2
		Fea vs Sur	Lin	59.54	0.44	0.14	0.21	0.86	0.22
		Fea vs Hap	Poly	56.74	0.6	0.16	0.25	0.84	0.1
		Fea vs Ang	Poly	57.51	0.58	0.16	0.25	0.84	0.06
		Fea vs Dis	Poly	58.02	0.5	0.15	0.23	0.85	0.2
		Fea vs Sad	Poly	70.74	0.27	0.14	0.18	0.86	0.06
		Fea vs Sur	Poly	49.62	0.63	0.14	0.23	0.86	0.09
		Fea vs Hap	RBF	56.23	0.31	0.1	0.15	0.9	0.05
		Fea vs Ang	RBF	57.51	0.31	0.1	0.15	0.9	0.05
		Fea vs Dis	RBF	51.4	0.52	0.13	0.21	0.87	0.07
		Fea vs Sad	RBF	59.03	0.58	0.17	0.26	0.83	0.05
		Fea vs Sur	RBF	54.96	0.35	0.1	0.16	0.9	0.06
One vs All	Norm	One vs All	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
	Max & Min	Fea vs All	Lin	81.93	0.25	0.26	1.02	0.74	0.26
		Fea vs All	Poly	84.73	0.25	0.33	1.33	0.67	0.71
		Fea vs All	RBF	82.7	0.42	0.33	1.33	0.67	0.25
	Z-norm	Fea vs All	Lin	82.7	0.21	0.25	1	0.75	0.51
		Fea vs All	Poly	84.73	0.04	0.13	0.5	0.88	74.5
		Fea vs All	RBF	85.24	0.42	0.4	1.6	0.6	0.25

Table 2g. Validation result of all emotions: Anger eyebrow

	Norm	One vs One	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
One vs One	Max & Min	Ang vs Hap	Lin	79.65	0.48	0.37	0.42	0.63	0.13
		Ang vs Dis	Lin	74.12	0.66	0.33	0.44	0.67	0.14
		Ang vs Fea	Lin	62.81	0.66	0.24	0.35	0.76	0.1
		Ang Vs Sad	Lin	67.34	0.59	0.26	0.36	0.74	0.1
		Ang vs Sur	Lin	64.07	0.64	0.24	0.35	0.76	0.14
		Ang vs Hap	Poly	79.4	0.48	0.37	0.41	0.63	0.04
		Ang vs Dis	Poly	71.86	0.62	0.3	0.4	0.7	0.04
		Ang vs Fea	Poly	62.31	0.66	0.24	0.35	0.76	0.05
		Ang Vs Sad	Poly	71.86	0.59	0.29	0.39	0.71	0.05
		Ang vs Sur	Poly	64.07	0.61	0.24	0.34	0.76	0.04
		Ang vs Hap	RBF	74.12	0.38	0.26	0.31	0.74	0.04
		Ang vs Dis	RBF	49.25	0.54	0.16	0.25	0.84	0.04
		Ang vs Fea	RBF	57.54	0.51	0.18	0.27	0.82	0.06
		Ang Vs Sad	RBF	37.94	0.57	0.14	0.22	0.86	0.04
		Ang vs Sur	RBF	59.8	0.46	0.18	0.26	0.82	0.04
	Z-norm	Ang vs Hap	Lin	81.41	0.59	0.42	0.49	0.58	0.12
		Ang vs Dis	Lin	64.82	0.52	0.22	0.31	0.78	0.1
		Ang vs Fea	Lin	65.33	0.75	0.27	0.4	0.73	0.09
		Ang Vs Sad	Lin	66.58	0.34	0.18	0.24	0.82	0.09
		Ang vs Sur	Lin	63.07	0.7	0.25	0.37	0.75	0.1
		Ang vs Hap	Poly	74.37	0.59	0.32	0.41	0.68	0.04
		Ang vs Dis	Poly	67.84	0.62	0.27	0.37	0.73	0.05
		Ang vs Fea	Poly	57.54	0.67	0.22	0.33	0.78	0.08
		Ang Vs Sad	Poly	76.63	0.33	0.28	0.3	0.72	0.04
		Ang vs Sur	Poly	57.29	0.75	0.23	0.35	0.77	0.05
		Ang vs Hap	RBF	72.61	0.48	0.27	0.35	0.73	0.09
		Ang vs Dis	RBF	66.08	0.52	0.23	0.32	0.77	0.04
		Ang vs Fea	RBF	54.02	0.46	0.16	0.23	0.84	0.06
		Ang Vs Sad	RBF	61.56	0.56	0.21	0.31	0.79	0.04
		Ang vs Sur	RBF	58.54	0.49	0.18	0.27	0.82	0.04
	Norm	One vs All	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
One vs All	Max & Min	Ang vs All	Lin	83.21	0.52	0.5	2	0.5	0.14
		Ang vs All	Poly	84.22	0.51	0.46	1.82	0.54	0.1
		Ang vs All	RBF	85.24	0.42	0.48	1.92	0.52	0.12
	Z-norm	Ang vs All	Lin	83.97	0.49	0.45	1.79	0.55	0.17
		Ang vs All	Poly	85.5	0.46	0.49	1.96	0.51	0.81
		Ang vs All	RBF	84.48	0.38	0.47	1.88	0.53	0.17

Table 2h. Validation result of all emotions: Sad mouth

	Norm	One vs One	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
One vs One	Max & Min	Sad vs Hap	Lin	59.8	0.62	0.24	0.35	0.76	0.42
		Sad vs Ang	Lin	58	0.6	0.23	0.33	0.77	0.14
		Sad vs Dis	Lin	57.3	0.68	0.24	0.35	0.76	0.08
		Sad vs Fea	Lin	52.4	0.65	0.21	0.32	0.79	0.12
		Sad vs Sur	Lin	63.4	0.6	0.26	0.36	0.74	0.07
		Sad vs Hap	Poly	64.6	0.59	0.26	0.37	0.74	0.06
		Sad vs Ang	Poly	55.7	0.62	0.22	0.33	0.78	0.06
		Sad vs Dis	Poly	60.6	0.62	0.25	0.35	0.75	0.07
		Sad vs Fea	Poly	59	0.62	0.24	0.34	0.76	0.08
		Sad vs Sur	Poly	61.1	0.59	0.24	0.34	0.76	0.06
		Sad vs Hap	RBF	44.8	0.4	0.13	0.2	0.87	0.06
		Sad vs Ang	RBF	43.8	0.38	0.13	0.19	0.87	0.06
		Sad vs Dis	RBF	44.3	0.4	0.13	0.2	0.87	0.05
		Sad vs Fea	RBF	46.3	0.34	0.12	0.18	0.88	0.08
		Sad vs Sur	RBF	44.8	0.43	0.14	0.21	0.86	0.06
	Z-norm	Sad vs Hap	Lin	76.3	0.59	0.38	0.46	0.62	0.49
		Sad vs Ang	Lin	77.1	0.53	0.38	0.44	0.62	0.17
		Sad vs Dis	Lin	64.1	0.51	0.24	0.33	0.76	0.4
		Sad vs Fea	Lin	73.3	0.49	0.32	0.39	0.68	1.62
		Sad vs Sur	Lin	77.9	0.4	0.37	0.38	0.63	0.21
		Sad vs Hap	Poly	63.1	0.5	0.23	0.32	0.77	12.5
		Sad vs Ang	Poly	68.2	0.51	0.28	0.36	0.72	2.63
		Sad vs Dis	Poly	58.5	0.44	0.19	0.27	0.81	9.78
		Sad vs Fea	Poly	35.9	0.9	0.2	0.33	0.8	35.5
		Sad vs Sur	Poly	70.2	0.35	0.25	0.29	0.75	0.72
		Sad vs Hap	RBF	68.4	0.37	0.24	0.29	0.76	0.06
		Sad vs Ang	RBF	63.6	0.47	0.23	0.31	0.77	0.06
		Sad vs Dis	RBF	57.5	0.49	0.2	0.28	0.8	0.06
		Sad vs Fea	RBF	55	0.5	0.19	0.28	0.81	0.08
		Sad vs Sur	RBF	55.5	0.57	0.21	0.31	0.79	0.06
One vs All	Norm	One vs All	Kernel	Acc(%)	Pre	Rec	F1-sco	Err.rte	Comp.Time of both phase (sec)
	Max & Min	Sad vs All	Lin	79.6	0.04	0.17	0.07	0.83	0.28
		Sad vs All	Poly	80.2	0.07	0.25	0.11	0.75	0.16
		Sad vs All	RBF	81.7	0.12	0.4	0.18	0.6	0.18
	Z-norm	Sad vs All	Lin	85	0.29	0.67	0.41	0.33	0.33
		Sad vs All	Poly	84.2	0.25	0.63	0.36	0.37	39.3
		Sad vs All	RBF	85.8	0.34	0.7	0.46	0.3	0.23

Table 3. Overall validation performance of all six basic emotions in "One vs. All"

Validation Parameters	Hold-Out Cross Validations—Kernels With Normalizations					
	Max& Min-Linear	Max& Min-Polynomial	Max& Min-RBF	Z-Linear	Z-Polynomial	Z-RBF
Accuracy (%)	85.14±5.50	81.75±6.62	86.16±4.48	87.21±3.43	86.74±2.52	88.45±2.7
F1-score	0.39±0.32	0.29±0.18	0.44±0.26	0.55±0.23	0.46±0.22	0.57±0.23
Error Rate	0.14±0.05	0.18±0.06	0.13±0.04	0.12±0.05	0.12±0.04	0.11±0.05
Computational Time (sec)	0.24±0.10	0.26±0.16	0.24±0.13	0.24±0.13	26.31±25.50	0.31±0.17

REFERENCES

Abrantes, G. A. (1999). MPEG-4 facial animation technology: Survey, implementation, and results. *Circuits and Systems for Video Technology*. *IEEE Transactions on*, *9*(2), 290–305.

Ahlberg, J. (2002). An active model for facial feature tracking. *EURASIP Journal on Applied Signal Processing*, ▪▪▪, 566–571.

Aleksic, P. S., & Katsaggelos, A. K. (2006). Automatic facial expression recognition using facial animation parameters and multistream HMMs. *Information Forensics and Security*. *IEEE Transactions on*, *1*(1), 3–11.

Audio (MPEG Mtg). M. V. (1998). ISO/MPEG N2503. Atlantic City.

Badler, S. P. (1981). Animating facial expressions. *Computer Graphics*, *15*(3), 245–252. doi:10.1145/965161.806812

Baizhen, Z. a. (2006). Facial feature extraction using improved deformable templates. *Signal Processing, 2006 8th International Conference on, 4.*

Bartlett, M. S., Littlewort, G., Fasel, I., & Movellan, J. R. (2003, June). Real Time Face Detection and Facial Expression Recognition: Development and Applications to Human Computer Interaction. *Computer Vision and Pattern Recognition Workshop,2003. CVPRW '03. Conference on, 5*(53), 16-22.

BulwerJ. (1649). Retrieved from https://www.library.northwestern.edu/spec/hogarth/physiognomics1.html

Chang, C.-C. J., & Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*(3), 1–27. doi:10.1145/1961189.1961199

Cheng, Y. (1995). Mean shift, mode seeking, and clustering. *Pattern Analysis and Machine Intelligence*. *IEEE Transactions on*, *17*(8), 790–799.

Cootes, T. F., G. E. (1998). A comparative evaluation of active appearance model algorithms. *British Machine Vision Conference*, 2, 680–689. 10.5244/C.12.68

Cowie, R. E. (2008). Recognition of emotional states in natural human-computer interaction. In *Multimodal user interfaces.* Springer Berlin Heidelberg.

Cristinacce, D. A. (2006). Feature Detection and Tracking with Constrained Local Models. *British Machine Vision Conference*, *2*(5). 10.5244/C.20.95

Cristinacce, D. A. (2007). Boosted Regression Active Shape Models. *British Machine Vision Conference*.

Darwin, C. (1904). The Expression of the Emotions in Man and Animals (2nd ed.; J. Murray, Ed.). London.

Devi, M. N. (2011). *Pattern Recognition: An Algorithmic Approach* (1st ed.). Springer-Verlag London.

Donato, G. e. (1999). Classifying facial actions. *Pattern Analysis and Machine Intelligence*. *IEEE Transactions on*, *21*(10), 974–989.

Ekman, W. a. (1977). *Facial action coding system*. Palo Alto: Consulting Psychologists Press, Stanford University.

Friesen, P. E. (1969). Pan-Cultural Elements In Facial Display Of Emotion. *Science*, 86–88.

Highfield, R. (2009). How your looks betray your personality. *New Scientist*.

Huttenlocher, P. F. (2005). Pictorial structures for object recognition. *International Journal of Computer Vision*, *61*(1), 55–79. doi:10.1023/B:VISI.0000042934.15159.49

Ioannou, S. V., Raouzaiou, A. T., Tzouvaras, V. A., Mailis, T. P., Karpouzis, K. C., & Kollias, S. D. (2005). Emotion recognition through facial expression analysis based on a neurofuzzy network. *Neural Networks*, *18*(4), 423–435. doi:10.1016/j.neunet.2005.03.004 PMID:15963691

Isidoro, S. S. (1998). Active blobs. In *Computer Vision, 1998. Sixth International Conference*, (pp. 1146–1153). Academic Press.

James, W. (1884). What is Emotion. *Mind*, *9*(34), 188–205. doi:10.1093/mind/os-IX.34.188

Joshi, W. T. (2005). Learning controls for blend shape based realistic facial animation. ACM SIGGRAPH 2005 Courses, 8.

Kanade, L. G. (2008). A generative shape regularization model for robust face alignment. In *Computer Vision–ECCV.* Springer Berlin Heidelberg.

Kanade, T. J. (2000). Comprehensive database for facial expression analysis. In *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition* (Cat. No. PR00580). IEEE. 10.1109/AFGR.2000.840611

Karpouzis, S. K. (2005). Multimodal Emotion Recognition And Expressivity Analysis. *IEEE International Conference on Mulitimedia and Expo*, 779-783.

Kass, M., Witkin, A., & Terzopoulos, D. (1988). Snakes: Active contour models. *International Journal of Computer Vision*, *1*(4), 321–331. doi:10.1007/BF00133570

Kumar, M. P., & Rajagopal, M. K. (2019). *Detecting facial emotions using normalized minimal feature vectors and semi-supervised twin support vector machines classifier*. Academic Press.

Laurenz Wiskott, J.-M. F. (1997). Face recognition by elastic bunch graph matching. *Pattern Analysis and Machine Intelligence*. *IEEE Transactions*, *19*(7), 775–779.

Li, Y. A. (2005). Shape parameter optimization for adaboosted active shape model. *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, 1*.

Littlewort, G., Bartlett, M. S., Fasel, I., Susskind, J., & Movellan, J. (2006). Dynamics of facial expression extracted automatically from video. *Image and Vision Computing*, *24*(6), 615–625. doi:10.1016/j.imavis.2005.09.011

Lucey, P. J. (2010). The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*, (pp. 94-101). IEEE. 10.1109/CVPRW.2010.5543262

Ludo Maat, M. P. (n.d.). Retrieved from MMI Facial Expression Database: https://mmifacedb.eu/

Montagu, J. (1994). *The Expression of the Passions: The Origin and Influence of Charles Le Brun's 'Conférence sur l'expression générale et particulière*. Yale University press. doi:10.1111/j.1467-8357.1994.tb00386.x

Naumovich, V. V. (1998). *Statistical learning theory*. New York: Wiley.

Ostermann, A. M. (2000). Face and 2-D mesh animation in MPEG-4. *Signal Processing Image Communication*, *15*(4), 387–421.

Padgett, C. a. (1997). Representing face images for emotion classification. *Advances in Neural Information Processing Systems*, 894–900.

Pandzic, I. S. (2002). *MPEG-4 Facial Animation: The Standard, Implementation and Applications*. Chichester, UK: John Wiley&Sons. doi:10.1002/0470854626

Pantic, M., & Patras, I. (2006). Dynamics of facial expression: recognition of facial actions and their temporal segments from face profile image sequences. *Systems, Man, and Cybernetics, Part B: Cybernetics IEEE Transactions on*, *36*(2), 433–449. PMID:16602602

Pantic, M. A. (2010.). Induced disgust, happiness and surprise: an addition to the mmi facial expression database. *Proc. 3rd Intern. Workshop on EMOTION (satellite of LREC): Corpora for Research on Emotion and Affect.*

Parke, F. I. (1982). Parameterized models for facial animation. *IEEE Computer Graphics and Applications*, *2*(9), 61–68. doi:10.1109/MCG.1982.1674492

Parrott. (2000, October). *Emotions in Social Psychology.* Psychology Press.

Pentland, K. A. (1991). Recognition of facial expression from optical flow. *IEICE Transactions on Information and Systems*, *74*(10), 3474–3483.

Pentland, P. M. A. (1991). Face recognition using eigenfaces. *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91., IEEE Computer Society Conference on*, 586-591.

Petridis, S. B. (2013). *The MAHNOB laughter database*. Academic Press.

Pitas, I. K. (2007). Facial expression recognition in image sequences using geometric deformation features and support vector machines. *Image Processing*. *IEEE Transactions on*, *16*(1), 172–187.

Popovic, B. (2007). Automatic rigging and animation of 3d characters. *ACM Transactions on Graphics*, *26*(3), 72. doi:10.1145/1276377.1276467

Raouzaiou, A., Tsapatsoulis, N., Karpouzis, K., & Kollias, S. (2002). Parameterized facial expression synthesis based on MPEG-4. *EURASIP Journal on Advances in Signal Processing*, *10*(10), 1–18. doi:10.1155/S1110865702206149

Roger Highfield, R. W. (2009, February). How your looks betray your personality. *New Scientist*.

Salam, H. (2013). *Multi-Object modeling of the face. Signal and Image processing* (Thesis). Université Rennes 1.

Samal, A., & Iyengar, P. A. (1992). Automatic recognition and analysis of human faces and facial expressions: A survey. *Pattern Recognition*, *1*(25), 65–77. doi:10.1016/0031-3203(92)90007-6

Shan, C. S., Gong, S., & McOwan, P. W. (2009). Facial expression recognition based on local binary patterns: A comprehensive study. *Image and Vision Computing*, *27*(6), 803–816. doi:10.1016/j.imavis.2008.08.005

Silverman, B. (1986). *Density estmaation for statistics and data analysis.* Chapman & Hall/CRC Press.

Simon, L., Jeffrey, C., & Jason, M. S. (2011). Deformable model fitting by regularized landmark mean-shift. *International Journal of Computer Vision*, *91*(2), 200–215. doi:10.100711263-010-0380-4

Sugie, N., M. K. (1978). A Perliminary note on pattern recognition of human emotional expression. *Proc. 4th Int.Joint Conf. on Pattern Recognition*, 408-410.

Taylor, C. J. T. (1992). Active Shape Models - 'Smart Snakes'. *British Machine Vision Conference*, 266-275.

Tian, Y.-T. (2001). Recognizing action units for facial expression analysis. *Pattern Analysis and Machine Intelligence*. *IEEE Transactions on*, *23*(2), 97–115.

Ventura, D. (2009). *SVM example.* Lecture notes.

Vetter, V. B. (1999). A morphable model for the synthesis of 3d faces. *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, 187–194.

Wallbott, H. G. (1986). Cues and channels in emotion recognition. *Journal of Personality and Social Psychology*, 69–70.

Wang, Y. S. (2008). Enforcing convexity for improved alignment with constrained local models. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE. doi:10.1155/S1110865702203078

Yongsheng & Gao. (2003). Facial expression recognition from line-based caricatures. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on, 33*(3), 407-412.

Zhang, Y., Ji, Q., Zhu, Z., & Yi, B. (2008, October). Dynamic Facial Expression Analysis and Synthesis With MPEG-4 Facial Animation Parameters. *IEEE Transactions on Circuits and Systems for Video Technology*, *18*(10), 1383–1396. doi:10.1109/TCSVT.2008.928887

Zhao, G. E. (2011). *Facial expression recognition from near-infrared videos*. Academic Press.

Zhao, J. K. G. (1996). Classifying Facial Emotions by Backpropagation Neural Networks with Fuzzy Inputs. *Proceedings of Conference on Neural Information Processing*, *1*, 454–457.

Chapter 4
KidNet:
Kidney Tumour Diagnosis System Design Using Deep Convolutional Neural Network

Umamaheswari S.
https://orcid.org/0000-0002-9931-9243
Anna University, MIT Campus, India

Sangeetha D.
Anna University, MIT Campus, India

C. Mouliganth
Anna University, India

Vignesh E. M.
Anna University, India

ABSTRACT

Kidney cancer is one of the 10 most common cancers in both men and women. The lifetime risk for one developing kidney cancer is about 1.6%. The rate of kidney cancer diagnosis has been rising since the 1990s due to the use of newer imaging tests such as CT scans. The kidneys are deep inside the body and hence small kidney tumours cannot be seen or felt during a physical examination. Existing work on kidney tumour diagnosis uses traditional machine learning and image processing techniques to find and classify the images. Deep learning systems do not require this domain-specific knowledge. The kidney tumour diagnosis system uses deep learning and convolutional neural networks to classify CT images. A deep learning neural network model named KidNet has been implemented. It has been trained using labelled kidney CT images. To achieve acceleration during the training phase, GPUs have been used. The network when trained with abdominal CT images achieved 86.1% accuracy, and the one trained with cropped portion of kidney images achieved 89.6% accuracy.

DOI: 10.4018/978-1-7998-2108-3.ch004

INTRODUCTION

Medical imaging technologies utilize Computerized Tomography (CT) scans or Magnetic Resonance Imaging(MRI) scans to assess the state of an organ or tissue and can observe a patient's condition for diagnosis and recommending treatment. It is because of these techniques we are able to diagnose cancers that might never have been diagnosed otherwise. Medical imaging is one of the applications that took advantage of Graphics Processing Unit (GPU) to get acceleration. The use of GPUs in this field has come to the point that there are several medical modalities shipping with GPUs.

The kidneys are internal organs that are so deep in the human body that tinytumors cannot be identified during physical examinations. Kidney tumor diagnosis is normally carried out by trained professionals manually, but it requires a very tedious process and is greatly dependent on the individual. So there is a need for an automated process.

Recently, there has been much important advancement in machine learning and artificial intelligence. They have played a major role in fields like medical image processing, computer-aided diagnosis, and so forth. Machine-learning techniques are used to minethe information from images and represent it effectively and efficiently. Machine learning and artificial intelligence techniques help doctors make a diagnosis, predict diseases accurately, and prevent them.

Existing work on kidney tumor diagnosis uses traditional machine learning and image-processing techniques to find and classify the images. The choice of descriptive features derived from the image greatly affects the performance of traditional machine learning neural networks. Also, identifying the required features is not a straightforward process, as it requires domain-specific knowledge.

Deep learning models do not require this domain-specific knowledge, as the network automatically learns high-level features from the input images. Deep learning methods have helped industries and researchers achieve very accurate results in various fields like speech recognition, computer vision, and natural language processing. Results produced by deep learning systems are on par with and sometimes better than those from human experts. Deep learning techniques provide state-of-the-art accuracy. It had created opportunities also in precise medical image analysis.

Deep learning using convolutional neural networks is becoming more important in image analysis as it is able to tackle complex problems successfully. The advantage of using a convolutional neural network (CNN), popularly used in image analysis, is that it requires reasonably fewer pre-processing steps compared to other image classification models.

Health care is a major domain in which deep learning provides solutions to a wide range of issues like cancer diagnosing, disease monitoring, and specific treatment suggestions. There is an enormous amount of data available at hospitals which can be used to train the diagnosing system. Image acquisition devices have improved so much that we are able to get X-ray, CT, and MRI scan images using radiology with better resolution.

GPUs are used in sophisticated systems likesmart phones, PDAs, workstations, and play stations, because they are very proficient in handling video, image, and graphics processing. This makes them a convincing choice over conventional processors in super computing systems. Deep Neural Network needs to perform thousands of identical operations on large data. The parallel processing capabilities of GPUs makes them capable of processing larger blocks of data in parallel, using a divide-and-conquer strategy (Owens, Houston, Lubeke, Green, & Stone, 2008).

A CPU may have limited cores optimized for sequential computing; a GPU's structural design has thousands of tiny but efficient processing elements that operate extremely in parallel for carrying out multiple processes concurrently (Asaduzzaman, Martinez, &Sepehri, 2015). General-purpose programming on GPUs has become popular due to its capability to handle massive computation-intensive applications like deep learning.

After the success of deep learning and convolutional neural networks in other real-world applications, like classification and object detection, the combination is also offering good solutions with high accuracy in classification based on medical imaging. It has also become an important method for diagnosing applications in the health sector. The application of GPU technology to deep learning using CNNs offers great prospects in achieving acceleration and greatly improving the system.

The proposed system, KidNet, uses deep learning and convolutional neural networks to classify CT images. Three convolutional neural networks were implemented, and they used kidney CT images for training. The results obtained are comparable to those from the current state-of-the-art systems; however, to achieve high accuracy and accelerate the training phase, GPUs were used to parallelize the model. The parallelized KidNet model performed better than CPUs and the results obtained remain stable.

KidNet was designed using CNNs from scratch, and contains eight layers: ðve convolutional, one fully connected, and two pooling layers. This model was then trained using two different approaches. In the first approach, the abdominal CT images for training KidNet were directly given. The network learned most of the features of the abdominal part of the images and classified whether the image is of the affected or the healthy category. Those abdominal CT scans had not only kidneys, but also other parts of the body like the liver and spleen. The network when trained with abdominal CT images achieved 86.1% accuracy. Next, since the other parts are not necessary, the kidney portion was cropped and that cropped portion was given as input to KidNet in the second approach. Now, the network did not learn unnecessary features, and using the cropped portion of the kidney images, achieved 89.6% accuracy.

The image dataset used consists of about 700 abdominal CT images. Out of these,568 images (401 affected and 167 healthy) were chosen for training and 115 images(75 affected and 40 healthy) were taken for testing. Image-augmentation techniques like translation, rotation were used to randomly alter the image data to extend to 1,800 images.

Neural Networks

The brain has billions of neurons that communicate with each other and share information. Artificial neurons were created with the same idea in mind, but were created for machines to make them think and act like our brain.

Neural networks are algorithms based on the human brain, and they are mode led to recognize patterns. They recognize input data through supervised learning or unsupervised learning, andthe patterns they recognize are mostly numerical data. Therefore, all kinds of digital data like sounds, images, or any other real-world data must be converted to numerical form and given to neural networks for it to process them.

Deep Learning

Deep learning neural networks are basically artificial intelligence functions that imitate the human brain. It is a set of models and techniques that develop deep neural networks to learn to extract complex features from data and to classify the data. Deep learning is a subset of machine learning, but has neural networks that are able to learn from unstructured or unlabeled data. These methods have helped industries and researchers achieve very accurate end results in different domains incomputational intelligence, such as speech processing, image and video analytics, natural-language processing (NLP), and natural-language generation (NLG).

According to Forbes, we generate around 2.6 quintillion bytes of data every day. Since deep learning requires huge amounts of data to learn, the increase in the daily generation of data has been one of the reasons deep learning technologies have grown considerably in the last few years.

Deep neural networks differ from conventional neural networks by their deepness, which means more hidden layers to process the data for pattern recognition. Feature extraction is a complex task that can take a long time to accomplish for data scientists. The advantage of deep learning networks is they automatically extract features without human intervention.

Deep learning methods using convolutional neural networks are becoming more important in image analysis due to their success at tackling complex problems. Traditional machine learning models in image analysis require human experts to derive descriptive features from the image. Deep learning models do not require this domain-specific knowledge, as the network automatically learns high-level features from the images. Results produced by deep learning systems are on par with or sometimes even better than those from human experts.

Convolutional Neural Networks

A convolutional neural network is a category of feed-forward, deep learning neural networks that is popularly used and applied in image analysis and video analytics. In traditional image classification algorithms, the filters are actually hand-engineered. Conversely, a convolutional neural network automatically fine-tunes the filters. The advantage of using CNNs is that they require reasonably less pre-processing compared to traditional image classification algorithms.

In the convolutional layer, a mathematical convolution operation is performed to produce a single value in the resulting feature map. Pooling is a sample-based discretization process; it decimates the input data and reduces their dimensionality. For this, a max pooling algorithm is generally used; although, average pooling can be used in some cases. In the dense layer or fully connected layer, each node in the current layer is connected to each node in the previous hidden layer. In the output layer, the probability for different classes is computed.

Convolutional neural networks have been used in a wide variety of computer vision applications. Krizhevsky et al (2012) as shown a record-breaking performance in the ImageNet Visual Recognition Challenge using CNNs. Most papers on medical image classification use CNNs for classification.

Medical Images

Medical imaging is mainly used to diagnose abnormalities and treat disease. Different kinds of imaging techniques include computed tomographic scans, magnetic resonance imaging, radiographs, ultrasounds, and electrography. Since these techniques provide a very large amount of data, the DICOM format (global standard) is used to store the medical images.

General-Purpose GPU Computing

GPUs typically handle computations meant only for computer graphics because GPUs analyze data as if they were an image. GPU computing is the use of GPUs to perform computations in applications that are traditionally handled using a CPU.

General-purpose GPUs have become popular due to their ability to handle massive computationally intensive applications. The purpose of using a GPU as an alternative computational platform is to achieve acceleration for computationally intensive tasks, beyond the domain of graphical applications.

BACKGROUND

Making a cancer diagnosis is usually done manually by experts like medical doctors. But it is a very tedious process and is very much dependent on the practitioner's expertise. So, there is a need for an automated process. Many works have been published on providing a medical diagnosis through the use of image-processing and machine learning techniques.

Kalaivani, Chatterjee, Juyal, and Gupta (2017) proposed a method for detecting cancer through a computerized procedure to reduce human error and make accurate predictions. Kalaivani et al. used traditional image-processing techniques for pre-processing and extracting features. In their work, artificial neural networks were used to predict lung cancer at the early stage. Image-processing steps include conversion into grayscale, histogram equalization, thresholding, and feature extraction. Their choice of neural network was a back propagation network.

Many algorithms and techniques have been proposed for improving the accuracy of the classification of medical data (Mredhula & Dorairangaswamy, 2015;Rao, Pereira, & Srinivasan, 2016). Shanmugapriya, Nehemiah, Bhuvaneswaran, Kannan, and Sweetlin(2017)suggested data discretization as a data pre-processing technique to enhance the effectiveness of feature extraction on medical data. The performance of their proposed crisp and fuzzy discretization methods is measured with the help of some classification methods like tree-based classification, the probabilistic induction method, the rule-based method, the network learning approach, kernel and distance-based techniques, and a rule-based fuzzy inference system. The classification accuracy remains stable,with less deviation.

Mohsenaet al.(2017) designed a classifier in which they combined discrete wavelet transform (DWT) and principal components analysis (PCA). The evaluation of the performance was quite good over many performance measures. They used this network to classify a dataset of 66 brain MRIs into 4 classes: glioblastoma, sarcoma, normal, and metastatic bronchogenic carcinoma tumors. Their methodology resembles a CNN but requires less hardware specifications and time to process large-sized images.

Feature extraction is a challenging task in image analysis due to the diversity of the features. Vu, Mousavi, Monga, Rao, and Rao(2016) proposed a framework that uses class-specific dictionaries for automatic feature discovery. The paper presented a low-complexity technique for classifying and ranking the disease, called discriminative feature-oriented dictionary learning (DFDL). The classes are designed in such a way that they cannot represent samples from other classes. The sparse representation method is used for learning the dictionaries, and then this information is used to identify new images. Experiments were done on real-world datasets: histopathological images of intraductal breast lesions; mammalian kidney, lung, and spleen images; and brain tumor images.

Kim and Park(2004) implemented a computer-aided kidney tumor diagnosing system from abdominal CT scans using kidney segmentation. Their computer-aided detection process utilized the following methods: digitizing the CT image, analyzing the image, segmenting the kidney from the abdominal scan image, analyzing the texture, seed pixel selection of the kidney tumor, segmenting the kidney tumor. The detection scheme uses thresholding for kidney segmentation and region growing for the delineation of the kidney tumor boundary. Lee, Hong, and Kim (2017) proposed a segmentation method to detect small renal masses in contrast-enhanced CT images and a classification technique using texture and context feature.

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is a competition that allows researchers to compare progress in image detection across millions of objects. It evaluates detection and classification algorithms. The most important neural network models that won the ImageNet challenge were AlexNet and GoogLeNet.

Krizhevsky, Sutskever, and Hinton (2012) trained a neural network, AlexNet, that consisted of 5 convolutional layers and 3 fully connected layers with a 1,000-way soft-max. The neural network had 60 million parameters and 650,000 neurons. They also used an efficient GPU implementation to speed up the training process. Additionally, a regularization method called dropout was employed to reduce the problem of overfitting. They achieved top-1 and top-5 error rates of 37.5% and 17.0%, respectively, which are better than the results of the previous state-of-the-art model. In the ILSVRC 12 competition, they achieved a winning top-5 test error rate of 15.3%.

Szegedy et al.(2015) proposed a deep convolutional neural network architecture and set a new state-of-the-art threshold in ILSVRC 14. The architecture improved the utilization of the resources, due to the increased depth and width of the network. The decisions were based on the Hebbian principle. The model used in their submission for ILSVRC 14, called GoogLeNet, was 22 layers deep.

After the advent of deep learning and convolutional neural networks, several simple and powerful models based on AlexNet were proposed to classify medical images like MRI, CT, and X-ray images. Li et al. (2014) designed a shallow CNN to categorize lung image patches with an interstitial lung disease (ILD) data set. Feature descriptors are complex and domain-specific. The tailor-made CNN can learn the intrinsic image features, automatically and effectively, from lung image patches. Generalized architecture can be used to classify other medical images.

The choice of descriptive features derived from the image significantly affects the performance of traditional machine learning neural networks. Also, identifying the required features is not a straightforward process, as it requires domain-specific knowledge.

Deep learning models do not require this domain-specific knowledge, as the network automatically learns high-level features from the images. Results produced by deep learning systems are at an equal or higher level of accuracy as those from manual diagnoses done by experts. Yang, Chen, Ding, and Pang (2016) studied the feasibility of using deep learning neural networks for classifying tumor tissues. The paper also discussed several practical ways to tune the parameters for the CNN.

It takes great effort to train a neural network from scratch, as it requires a large amount of labeled data. It also takes a great deal of expertise to ensure proper convergence of the network. One good alternative is to transfer learning in a CNN. Transfer learning in machine learning is the technique of using a pre-trained model for an application used for another application. Tajbakhsh et al.(2016) proved that the use of pre-trained deep CNNs with little tuning could perform better than a deep CNN that is trained from scratch.

Deep learning algorithms and CNNs have become a methodology of choice for analyzing medical images. Litjens et al. (2017) reviewed the most important deep learning theoriesrelevant to classification based on medical imaging. They surveyed the use of deep learning techniques for image classification, object detection, segmentation, registration, and other tasks. They also discussed open concerns and future research directions in medical image analysis.

Nowadays, deep learning–based embedded applications have become products. Xu, Wang, and Li(2017) introduced image recognition based on deep learning. They also introduced the application of deep learning in video analysis and provided optimized algorithms for deep learning–embedded applications.

Mo, Kim, Kim, Mohaisen, and Lee (2017) observed the performance enhancements in deep neural network (DNN) computation on a GPU-enabled, multi-core parallel computing platform. To assess the performance of a DNN, they varied the numbers of hidden layers and implemented it using GPUs. The computation time is reduced by about three times when compared to using only CPUs.

GPU-accelerated computing is the use of a GPU to speed-up analytics, deep learning, and engineering applications. GPU technology has been offering great prospects in deep learning and parallelization. Maceina and Manduchi (2017) stated the reasons for the limited penetration of GPU technology. One of the reasons is that highly parallel programming of real-time applications is practically not available to efficiently utilize GPUs. Another reason is that the performance of GPU is pulled down by memory transfers.

In summary, feature extraction is still a challenging process in medical image analysis because of the diversity of the features. As far as deep learning is concerned, high-end computing machines are needed to achieve state-of-the-art performance levels. Also, deep learning requires a very large training dataset, since classification accuracy of the classifier greatly depends on the quality and size of the dataset. A real-time challenge that exists in any deep learning neural network is to train using a large set of images in minimal time.

Challenges

Many algorithms and techniques have been proposed for improving the accuracy of the classification of medical data. Feature extraction is a challenging task in medical image analysis because of the large number of features and their diversity in nature. The choice of descriptive features derived from the image greatly affects the performance of traditional machine learning neural networks (Razzak, Naz,&Zaib,2017). Also, identifying the required features in the medical image is itself a great challenge, since it requires medical knowledge.

That's one reason why deep learning is being chosen over the machine learning technique, as it provides a better performance than previous techniques. As far as deep learning is concerned, high-end computing machines are needed to achieve a state-of-the-art performance. Also, deep learning needs a very large amount of training input data, since the accuracy of the classification greatly depends on the quality and volume of the input data. However, the scarcity of input medical image data is another big challenge of using deep learning. Even though data are available, it is hard to obtain health records because of privacy issues. Data analytics researchers find anonymizing patient information is a big challenge, as it is difficult to prevent its use or disclosure.

Machine learning algorithms use huge amounts of data as inputs; they identify patterns and build a prediction model. But understanding how the model worked is an issue. It is still, at least for now, difficult to efficiently tune the hyper-parameters of a neural network.

SOLUTION

A convolutional neural network is a sub-section of deep, feed-forward ANNs, which can be used to analyze visual imagery. CNNs use only a few pre-processing steps as compared to traditional classification methods, based on medical images. This means that the CNN learns the filter weights automatically, in contrast to traditional algorithms, for which they are hand-engineered. Additionally, CNNs are basic models of deep learning, where a complex model drives the development of computational intelligence by contributing the systems that mimic human brain activities.

A convolutional neural network combines three architectural ideas: local receptive fields, shared weights (or weight replication), and spatial or temporal sub-sampling. There are three basic components:

1. Convolutional layer
2. Pooling layer
3. Fully connected layer

Convolutional Layer

In each level, the layer carries out a set of mathematical operations, like convolution, to generate a distinct value in the output of that layer.

conv2d () - Constructs a two-dimensional convolutional layer.

It takes the following as arguments:

- Number of filters
- Filter kernel size
- Padding
- Activation function

Pooling Layer

A commonly used pooling algorithm is max pooling, which extracts sub-regions of the feature map, keeps their maximum value, and discards all other values.

max_pooling2d () - Constructs a two-dimensional pooling layer using the max pooling algorithm.

It takes the following as arguments:

- Pooling filter size
- Stride

Fully Connected Layer

In a dense layer, every node in the layer is connected to every node in the preceding layer.

dense () - Constructs a dense (fully connected) layer.

It takes the following as arguments:

- Number of neurons
- Activation function

Softmax Classifier

The softmax function takes a vector of arbitrary, real-valued scores and squashes it into a vector of values between zero and one that sum to one. A softmax classifier uses the softmax function and gives the probabilities for each class label.

Overfitting

Overfitting occurs when the network fits too well to the training data set. But for new images, the model fails to fit. This means that the model classifies the trained data accurately, but fails to classify new data accurately. In case of overfitting, the model recognizes specific images in a training set instead of a general pattern. The following steps are generally used to reduce overfitting:

1. Adding more data (increase the size of dataset)
2. Using data augmentation techniques
3. Adding regularization techniques
4. Reducing architecture complexity

Dropout

Dropout consists of setting to zero the output of selected hidden neurons. The dropped-out neurons could not participate in the forward path computation and also inregression. Hence, an input data point is given each time to a different architecture of the neural network, but the same weights are shared bythese architectures.In this arrangement, the dropping of neurons is randomized; therefore, a neuron doesnot depend on the existence of other, specificneurons. Hence, the network iscompelled to learn the features in a robust manner, and overfitting of the network is minimized.

KIDNEY TUMOR DIAGNOSING APPROACH

Overfitting occurs when the model has over-learned the training data set, and it is problematic for the model to generalize to a new test image data set not used during training. A new model, which is free from the overfitting problem, is proposed and shown in Figure 1.

It consists of five convolutional layers, two pooling layers, one fully connected layer, and an output layer. This model is proposed to reduce the overfitting problem and increase the accuracy of classification.

This convolution layer uses 32 10×10 convolutions with a stride of 1 and 5×5 padding to convolve the features followed by a Rectiðed Linear Unit (ReLU) layer to set all negative elements to zero. The pooling layer calculates the maximum value of the feature over a region of the image. This max pooling layer has a ðlter size of 3 × 3. The next layer has a 10% dropout to reduce the overðtting. The next convolution layer uses 64 5 × 5 convolutions with a stride of 1. This is followed by a ReLU. The next layer has a 10% dropout. The next convolution layer uses 128 5 × 5 convolutions with a stride of 1. The next two convolution layer uses 3 × 3 convolutions with a stride of 1. The pooling layer calculates the average value of the feature over a region of the image. This max pooling layer has a ðlter size of 3 × 3. The next layer has a 50% dropout. Every individual neuron in the last but previous layer is connected to each neuron in its preceding layer, and hence it is named as "fully connected layer." The final layer of the KidNet is the prediction layer, which outputs the estimated significance of each class for the inputdata.

Method 1: With Abdominal CT Images

In method 1, the abdominal CT images for training KidNet are directly given. The network learned most of the features of the abdominal-part images and classified whether the image is of the affected or the healthy category.

Method 2: With Cropped Kidney Portions

KidNet is trained using abdominal CT scans. The abdominal CT scans had not only kidneys, but other parts of the body, like the liver and spleen, as well. Since other parts are not necessary, the kidney portion is cropped and that cropped portion is given as input to KidNet. Now, the network does not learn unnecessary features.

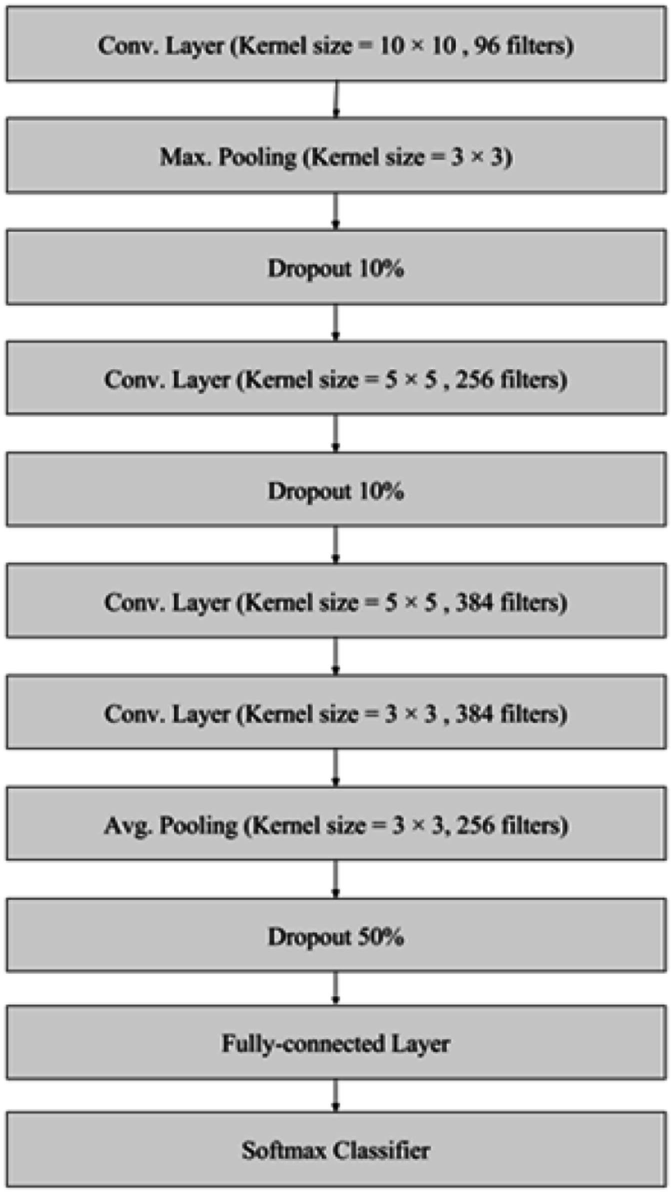
IMPLEMENTATION

The neural networks are implemented using TensorFlow on NVIDIA DIGITS. The weight initialization technique used is Xavier Initialization, created by Zelier and Fergus (2014). The rectified linear unit is the choice of activation function, and the solver type is Stochastic Gradient Descent. For the classifier, a two-way softmax is used.

TensorFlow is an open-source machine learning framework. The flexibility of the architecture allows us to use single or many core CPUs with/without GPUs for implementing the system. It was initiallyestablished for researchin machine learning and deep learning, but the system is very generic and can be applicable to other domains as well. TensorFlow provides APIs for Python, C++, Haskell, Java, Go, and Rust. In addition, third-party packages are available for C#, Julia, R, and OCaml.

Keras is a neural networks library, written in Python. It runs on top of TensorFlow, CNTK, or Theano. Keras allows for easy and fast prototyping. It allows for the development of both convolutional neural networks and recurrent neural networks and runs perfectly on CPU with/without GPU. It puts user experience first, by reducing the number of user actions and offering clear and actionable comment on user errors. There are built-in modules for different kinds of layers, objective functions, optimizers, initialization, activation functions, and regularization. Not only all these modules are all standalone modules that can be combined to create new models, but it is also simple to add modules. Models are compact, easier to debug, and allow for ease of extensibility.

Figure 1. KidNet architecture



DIGITS is a wrapper for NVCaffe, Torch, and TensorFlow. It gives a GUI solution to the above-referenced tools like TensorFlow instead of handling them from the terminal. DIGITS is a built-in DNN model, and it can be used for image and video analytic applications. It makes things easier for deep learning modules like data augmentation, devising and training the networks on multi-GPU systems, and evaluating the performance using sophisticated visualizations. It also helps in choosing the most efficient model from the performance results for final implementation. DIGITS is totally supportive of DNN designers and hence is helpful in devising and training networks instead of programming from scratch and sorting out the issues.

GPUs are used in intricate systems like mobile devices phones, play stations, and so forth. They are very efficient in handling video, image, and graphics processing, and the parallel processing capabilities of GPUs makes them capable of processing larger blocks of data in parallel, using a divide-and conquer-strategy. In a desktop system, a GPU is used on a VGA card, or it is used on the motherboard or in the processor chip itself. The GPUs used in this work are NVIDIA's GeForce 940M and GTX 1050.

In a Gradient Descent (GD) optimization scheme, after every iteration, the weights are adjusted. If the input dataset is huge in size, the use of GD may be expensive, because, in every iteration, the computation is done. Therefore, for the huge input dataset, the GD adjusts the weights at a slow pace; hence, convergence may be delayed.

In Stochastic Gradient Descent (SGD), the weights are updated after each training sample.SGD is faster because the algorithm uses a single or a few training examples to calculate the parameters (Bottou, 2010). In SGD, the learning rate is much smaller, compared to the standard gradient descent, due to higher variance, and random shuffle is required because the order dataset can bias the gradient.

The Adam method(Kingma& Ba, 2015) is an efficient stochastic optimization that only requires first-order gradients, with little memory requirement. The values of weight updates are independent of changes in the gradient. The step levels are roughly limited by the step-level hyper parameter. It does not need a fixed objective; it performs even for sparse gradients, and it indeed implements a process of step-size annealing.

In deep neural networks, choosing the initial network parameters plays a vital role in converging the network in a considerable time duration and optimizing the loss function, even after many iterations (Glorot&Bengio, 2010). In the case of very small coefficients, the variance of the input value decreases as it drifts through every level of the network. The value finally becomes insignificant and does not make any impact in decision making. On the other hand, if they are large, then the variance of input value grows fast while passing through every level. This leads to flat output from a non-linear activation function, like sigmoid function, and it has no role in decision making.

It is essential to initialize the coefficients of the neural network appropriately to make it to perform efficiently. In the training phase, the coefficients are to be initialized in rational manner. Xavier initialization helps in this step. The coefficients are initialized in a manner that the variance becomes constant. In that way, it supports the signal not to rise largely or die out to zero.The input values are transformed into output signals by the activation function in every node of ANN, and the output of a node in a layer is given as the input to the succeeding layer in the network.

CNNs are basically back-propagation neural networks. Back-propagation networks suffer from a fundamental problem known as the vanishing gradient. Vanishing gradients make the network only remember recent events and forget the more distant past. During training, the gradient decreases in value back through the net. When the gradient is large, the network trains quickly; when it is small, it trains slowly. Deep convolutional neural networks with ReLU as an activation function train much quicker than similar networks with sigmoid or tanh as activation functions because ReLU reduces the likelihood of vanishing gradients. The ReLU is defined as the positive part of its argument, $f(x) = \max(x, 0)$.

The input is a set of abdominal CT scans of affected and healthy kidneys, and the output is the classification of whether the CT scan is that of an affected kidney or a healthy one.The CT scan images in a DICOM format are converted into JPEG images. The images in the datasets are augmented by rotating, translating, and adding noise to the images. Then both the original and augmented images are used as inputs to the system. The parameters like learning rate, number of epochs, weights, and the solver types are initialized, then the images are converted into a NumPy array to easily process them. These arrays are given to the model; the training dataset is evaluated, and the model is tested. The softmax classifier computes the probabilities of each type.

The input dataset consists of about 700 abdominal CT images collected from Bharath Scans, Chennai, and The Cancer Imaging Archive (TCIA), in which 500 are affected and 200 are healthy. These labeled images are trained in KDTS using KidNet. The networks were trained using 568 kidney CT scan images, in which 401 were affected and 167 were healthy. The trained networks were tested using 115 kidney CT scan images, in which 75 were affected and 40 were healthy.

KidNet Implementation

Method 1: With CT Images

KidNet uses a combination of max and average pooling layers and has been fully trained using kidney CT images. The activation function used is ReLU and the solver type is Adam. The learning rate was initialized as 0.001, and the weights were initialized based on Xavier initialization. The results of method 1 are given in Table 1.

Table 1. Results using CT images

	Classified as Affected	Classified as Healthy
Affected	69	6
Healthy	13	27

Precision = True Positive/(True Positive + False Positive) = 68 / (68 + 9) =88.3%
Recall = True Positive/(True Positive + False Negative) = 68/ (68+7) = 90.6%
Accuracy = (True Positive + True Negative)/Total images = (68+31)/115=86.1%

Method 2: With cropped kidney portion

In method 1, the model was trained using CT images of the abdomen. For method 2, the approach was adjusted, and the unnecessary regions in the CT images were blacked out and then given as input to KidNet for training. The results of method 2 are shown in Table 2.

Table 2. Results using cropped kidney portion

	Classified as Affected	Classified as Healthy
Affected	70	5
Healthy	8	32

Precision = True Positive/(True Positive + False Positive) = 70/ (70+7) = 90.9%
Recall = True Positive/(True Positive + False Negative) = 70/ (70+5) = 93.3%
Accuracy = (True Positive + True Negative)/Total images = (70+33)/115=89.6%

Comparison

Various parameters like layers, activation function, pooling function, training time, accuracy, precision, and recall are compared and tabulated in Table 3.

Table 3. Comparison of parameters

	Method 1	Method 2
Accuracy	86.1%	89.6%
Precision	88.3%	90.9%
Recall	90.6%	93.3%

Since KidNet overcomes the problem of overfitting and the cropped images of kidneys are given for training, method 2produces better results than method 1 and the other networks.

CONCLUSION

A CNN model has been proposed to detect tumors in kidneys. For the training process, about 600 images were used. Then image augmentation techniques were used, and the dataset was extended to 1,800 images. KidNet trained with images of only the cropped portion, with solely the kidneys, achieved 89.6% accuracy, and training took 150 minutes.

FUTURE WORK

Augmentation techniques were used to increase the dataset, to increase the accuracy, and to reduce overfitting. But the input images are never reused exactly, so the networks could be trained using more images instead of augmentation. The utility of the model could be extended to classify different types of cancers on the kidney, and KidNet could be generalized and fine-tuned to be used as a generic neural network for all kinds of medical images.

REFERENCES

Asaduzzaman, A., Martinez, A., & Sepehri, A. (2015). *A time efficient image processing algorithm for multicore/many core parallel computing*. IEEE Southeast Conference Fort Lauderdale.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In Y. Lechevallier, & G. Saporta (Eds.), *Proceedings of COMPSTAT'2010, Physica-Verlag HD*. 10.1007/978-3-7908-2604-3_16

Glorot, X., & Bengio, Y. (2010). *Understanding the difficulty of training deep feed-forward neural networks*. International Conference on Artificial Intelligence and Statistics, Chia Laguna Resort, Sardinia, Italy.

Kalaivani, S., Chatterjee, P., Juyal, S., & Gupta, R. (2017). *Lung cancer detection using digital image processing and artificial neural networks*. International conference of Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India. 10.1109/ICECA.2017.8212773

Kim, D. Y., & Park, J. W. (2004). Computer-aided detection of kidney tumor on abdominal computed tomography scans. *Acta Radiologica*, *45*(7), 791–795. doi:10.1080/02841850410001312 PMID:15624525

Kingma, D. P., & Ba, J. (2015).*Adam: A method for stochastic optimization*. Paper presented at ICLR, San Diego, CA.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Proceedings of the conference Neural Information Processing Systems*.

Lee, H. S., Hong, H., & Kim, J. (2017). Detection and segmentation of small renal masses in contrast-enhanced CT images using texture and context feature classification. *Proceedings of IEEE International Symposium on Biomedical Imaging*. 10.1109/ISBI.2017.7950588

Li, Q., Cai, W., Wang, X., Zhou, Y., Feng, D. D., & Chen, M. (2014). Medical image classification with convolutional neural network. *IEEE International Conference on Control Automation Robotics & Vision (ICARCV)*. 10.1109/ICARCV.2014.7064414

Litjens, G., Kooi, T., Bejnordi, B. E., Ciompi, F., Ghafoorian, M., Jeroen, A. W. M., ... Sanchez, C. I. (2017). A survey on deep learning in medical image analysis. *Medical Image Analysis*, *42*, 60–88. doi:10.1016/j.media.2017.07.005 PMID:28778026

Maceina, T. J., & Manduchi, G. (2017). Assessment of general purpose GPU systems in real-time control. *IEEE Transactions on Nuclear Science*, *64*(6), 1455–1460. doi:10.1109/TNS.2017.2691061

Mo, Y. J., Kim, J., Kim, J. K., Mohaisen, A., & Lee, W. (2017). *Performance of deep learning computation with TensorFlowsoftware library in GPU-capable multi-core computing platforms*. International Conference on Ubiquitous and Future Networks (ICUFN), Milan, Italy.

Mohsen, H., El-Dahshan, E. S. A., El-Horbaty, E. S. M., & Salem, A. B. M. (2017). Classification using deep learning neural networks for brain tumors. *Future Computing and Informatics Journal*, *3*(1), 68–71. doi:10.1016/j.fcij.2017.12.001

Mredhula, L., & Dorairangaswamy, M. A. (2015). Detection and classification of tumours in CT images. *Indian Journal of Computer Science and Engineering*, *6*(2), 52–59.

Owens, J. D., Houston, M., Lubeke, D., Green, S., & Stone, J. E. (2008). GPU computing. *Proceedings of the IEEE*, *96*(5), 879–899. doi:10.1109/JPROC.2008.917757

Rao, P., Pereira, N. A., & Srinivasan, R. (2016). *Convolutional neural networks for lung cancer screening in computed tomography scans*. IEEE International Conference on Contemporary Computing and Informatics (IC3I), Noida, India. 10.1109/IC3I.2016.7918014

Razzak, M. I., Naz, S., & Zaib, A. (2017). *Deep learning for medical image processing: Overview, challenges and future. Classification in BioApps, Lecture Notes in Computational Vision and Biomechanics, 26.*

Shanmugapriya, M., Nehemiah, K. H., Bhuvaneswaran, R. S., Kannan, A., & Sweetlin, J. D. (2017). Fuzzy discretization based classification of medical data. *Research Journal of Applied Sciences, Engineering and Technology*, *8*(8), 291–298. doi:10.19026/rjaset.14.4953

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., …Rabinovich, A. (2015). *Going deeper with convolutions*. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA.

Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., & Liang, J. (2016). Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, *35*(5), 1299–1312. doi:10.1109/TMI.2016.2535302 PMID:26978662

Vu, T. H., Mousavi, H. S., Monga, V., Rao, A., & Rao, G. (2016). Histopathological image classification using discriminative feature oriented dictionary learning. *IEEE Transactions on Medical Imaging*, *35*(1), 738–751. doi:10.1109/TMI.2015.2493530 PMID:26513781

Xu, J., Wang, B., & Li, J. (2017). *Deep learning application based on embedded GPU. IEEE International Conference on Electronics Instrumentation & Information Systems (EIIS)*, Harbin, China. 10.1109/EIIS.2017.8298723

Yang, X., Chen, S., Ding, V., & Pang, B. (2016). *A deep learning approach for tumor tissue image classification*. International Association of Science and Technology for Development (IASTED) International Conference on Biomedical Engineering, Innsbruck, Austria. 10.2316/P.2016.832-025

Zelier, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. *Springer LNCS*, *8689*, 818–833.

Chapter 5
Liver Disease Detection Using Grey Wolf Optimization and Random Forest Classification

Singaravelan Shanmugasundaram
PSR Engineering College, India

Parameswari M.
FX Engineering College, India

ABSTRACT

Utilizing machine learning approaches as non-obtrusive strategies is an elective technique in organizing perpetual liver infections for staying away from the downsides of biopsy. This chapter assesses diverse machine learning methods in expectation of cutting-edge fibrosis by joining the serum bio-markers and clinical data to build up the order models. An imminent accomplice of patients with incessant hepatitis C was separated into two sets—one classified as gentle to direct fibrosis (F0-F2) and the other ordered as cutting-edge fibrosis (F3-F4) as per METAVIR score. Grey wolf optimization, random forest classifier, and decision tree procedure models for cutting-edge fibrosis chance expectation were created. Recipient working trademark bend investigation was performed to assess the execution of the proposed models.

MACHINE LEARNING

Over the past two decades machine learning has become one of the mainstays of information technology and, with that, a rather central, albeit usually hidden, part of people's life. With the ever increasing amounts of data becoming available, there is good reason to believe that smart data analysis will become even more pervasive as a necessary ingredient for technological progress.

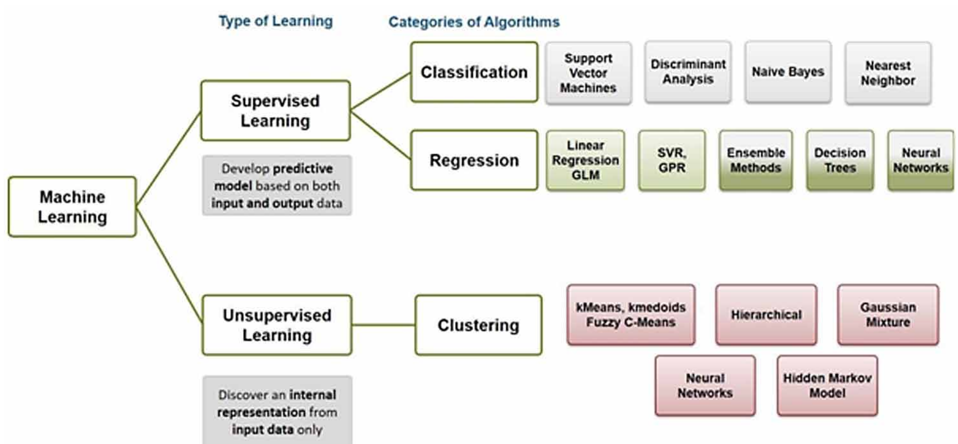
Machine learning can appear in many guises. Presently talk about various applications, the sorts of information manage, lastly, formalize the issues in a to some degree progressively adapted style. The latter is key to avoid reinventing the wheel for every new application. Instead, much of the art of machine learning is to reduce a range of fairly disparate problems to a set fairly narrow prototypes. Much

DOI: 10.4018/978-1-7998-2108-3.ch005

of the science of machine learning is then solving those problems and providing good guarantees for the solutions.

MACHINE LEARNING: ALGORITHMS TYPES

Figure 1. Machine learning techniques include both unsupervised and supervised learning



Machine learning algorithms are organized into a taxonomy, based on the desired outcome of the algorithm. Common algorithm types include:

- **Supervised Learning:** The algorithm generates a function that maps inputs to desired outputs. One standard formulation of the supervised learning task is the classification problem: The learner is required to learn (to approximate the behaviour of) a function which maps a vector into one of several classes by looking at several input-output examples of the function.
- **Unsupervised Learning:** It models a set of inputs; labelled examples are not available.
- **Semi-Supervised Learning:** It combines both labelled and unlabelled examples to generate an appropriate function or classifier.
- **Reinforcement Learning:** The algorithm learns a policy of how to act given an observation of the world. Every action has some impact in the environment, and the environment provides feedback that guides the learning algorithm.
- **Transduction:** Similar to supervised learning, but does not explicitly construct a function. Instead, it tries to predict new outputs based on training inputs, training outputs, and new inputs.
- **Learning to Learn:** The algorithm learns its own inductive bias based on previous experience.

The performance and computational analysis of machine learning algorithms is a branch of statistics known as computational learning theory.

Machine learning is about designing algorithms that allow a computer to learn. Learning is not necessarily involves consciousness but learning is a matter of finding statistical regularities or other patterns in the data. Thus, many machine learning algorithms will barely resemble how human might approach a learning task. However, learning algorithms can give insight into the relative difficulty of learning in different environments.

Supervised Learning Approach

Supervised machine learning builds a model that makes predictions based on evidence in the presence of uncertainty. A supervised learning algorithm takes a known set of input data and known responses to the data (output), and trains a model to generate reasonable predictions for the response to new data. Use supervised learning if have known data for the output are trying to predict.

Supervised learning is fairly common in classification problems, because the goal is often to get the computer to learn a classification system that have created. Digit recognition, once again, is a common example of classification learning. More generally, classification learning is appropriate for any problem where deducing a classification is useful and the classification is easy to determine. In some cases, it might not even be necessary to give predetermined classifications to every instance of a problem if the agent can work out the classifications for itself. This would be an example of unsupervised learning in a classification context.

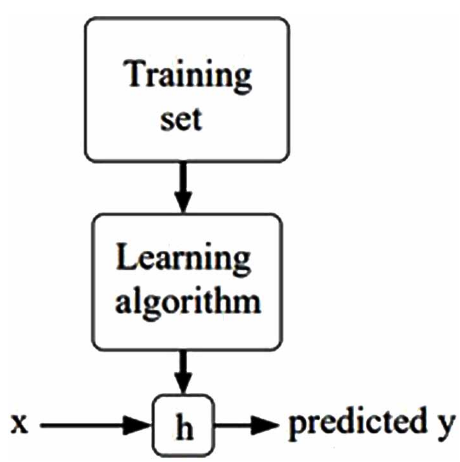
Supervised learning often leaves the probability for inputs undefined. This model is not needed as long as the inputs are available, but, if some of the input values are missing, it is not possible to infer anything about the outputs. As to unsupervised learning, all the observations are assumed to be caused by latent variables, that is, the observations are assumed to be at the end of the causal chain.

Unsupervised Learning

Unsupervised learning finds hidden patterns or intrinsic structures in data. It is used to draw inferences from datasets consisting of input data without labelled responses.

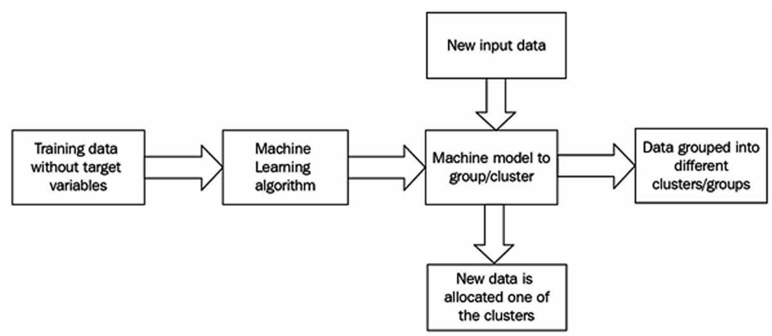
Unsupervised learning seems much harder: The goal is to have the computer learn how to do something that don't tell it how to do! There are actually two approaches to unsupervised learning. The first

Figure 2. Supervised learning techniques



approach is to teach the agent not by giving explicit categorizations, but by using some sort of reward system to indicate success. This type of training will generally fit into the decision problem framework because the goal is not to produce a classification but to make decisions that maximize rewards. This approach nicely generalizes to the real world, where agents might be rewarded for doing certain actions and punished for doing others. A second type of unsupervised learning is called clustering. In this type of learning, the goal is not to maximize a utility function, but simply to find similarities in the training data. The assumption is often that the clusters discovered will match reasonably well with an intuitive classification.

Figure 3. Unsupervised learning techniques



WHY USING MACHINE LEARNING

See Figure 4.

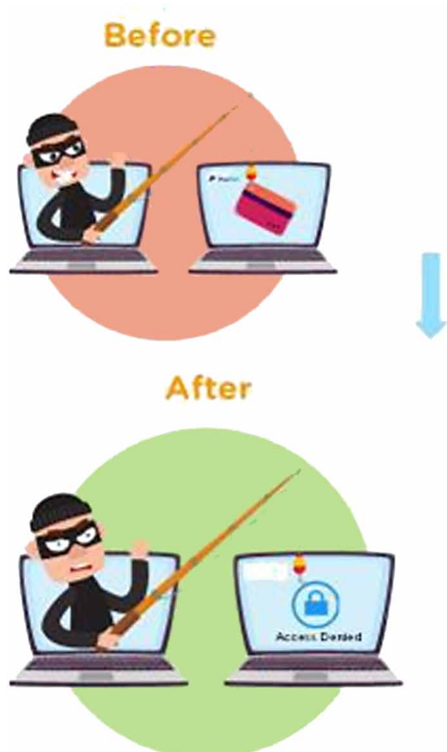
HEPATITIS C - LIVER FIBROSIS

Hepatitis C is a blood-borne virus that infects the cells of the liver (Alodini, 2012). Most cases occur in people who share needles or injecting equipment contaminated with traces of blood to inject “street drugs”. Some people clear the infection naturally. Some people with persistent infection remain free of symptoms, although others have symptoms. Persistent infection can lead to “scarring” of the liver (cirrhosis) and may lead to liver cancer. Treatment can clear the infection in over half of cases.

Liver fibrosis occurs when the healthy tissue of a person’s liver becomes scarred and therefore cannot work as well. Fibrosis is the first stage of liver scarring. Later, if more of the liver becomes scarred, it is known as liver cirrhosis.

While some animal studies have shown the potential for the liver to regenerate or heal itself, once liver damage is done in humans, the liver does not usually heal. However, medications and lifestyle changes can help to keep fibrosis from getting worse.

Figure 4. Why use machine learning

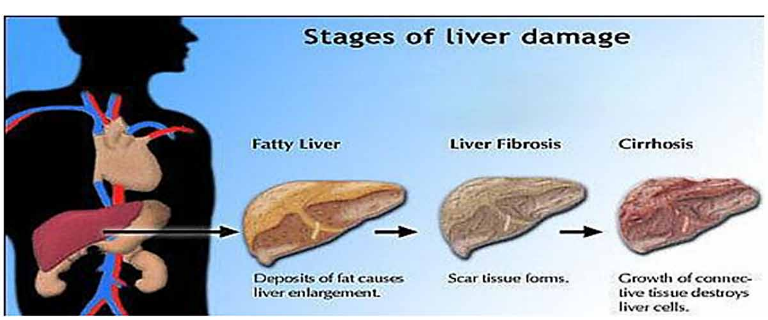


Stages of Liver Fibrosis

There are several different scales of liver fibrosis staging, where a doctor determines the degree of liver damage. Since staging can be subjective, each scale has its own limitations. One doctor may think a liver is slightly more scarred than another. However, doctors will usually assign a stage to liver fibrosis because it helps the patient and other doctors understand the degree to which a person's liver is affected.

One of the more popular scoring systems is the METAVIR scoring system. This system assigns a score for "activity" or the prediction of how fibrosis is progressing, and for the fibrosis level itself. Doc-

Figure 5. Stages of liver damage



tors can usually assign this score only after taking a biopsy (Bedossa & Carrat, 2009) or tissue sample of a piece of the liver. The activity grades range from A0 to A3:

- A0: No activity.
- A1: Mild activity.
- A2: Moderate activity.
- A3: Severe activity.

The fibrosis stages range from F0 to F4:

- F0: No fibrosis.
- F1: Portal fibrosis without septa.
- F2: Portal fibrosis with few septa.
- F3: Numerous septa without cirrhosis.
- F4: Cirrhosis.

Therefore, a person with the most severe disease form would have an A3, F4 METAVIR score. Another scoring system is Batts and Ludwig, which grades fibrosis on a scale of grade 1 to grade 4, with grade 4 being the most severe. The International Association of the Study of the Liver (IASL) also has a scoring system with four categories that range from minimal chronic hepatitis to severe chronic hepatitis.

Symptoms of Liver Fibrosis

Doctors do not often diagnose liver fibrosis (Bonacini, Hadi, Govindarajan, & Lindsay, 1997) in its mild to moderate stages. This is because liver fibrosis does not usually cause symptoms until more of the liver is damaged.

When a person does progress in their liver disease, he/she may experience symptoms that include:

- Appetite loss.
- Difficulty in thinking clearly.
- Fluid build-up in the legs or stomach.
- Jaundice (where the skin and eyes appear yellow).
- Nausea.
- Unexplained weight loss.
- Weakness.

According to a study, an estimated 6 to 7% of the world's population has liver fibrosis and does not know it because they do not have symptoms.

Causes of Liver Fibrosis

Liver fibrosis occurs after a person experiences injury or inflammation in the liver. The liver's cells stimulate wound healing. During this wound healing, excess proteins such as collagen and glycoproteins build up in the liver. Eventually, after many instances of repair, the liver cells (known as hepatocytes) can no longer repair themselves. The excess of proteins form scar tissue or fibrosis.

Several types of liver diseases exist that can cause fibrosis. These include:

- Autoimmune hepatitis.
- Biliary obstruction.
- Iron overload.
- Nonalcoholic fatty liver disease, which includes nonalcoholic fatty liver (NAFL) and nonalcoholic steatohepatitis (NASH)
- Viral hepatitis B And C.
- Alcoholic liver disease.

According to The Lancet (Crisan et al., 2012) the most common cause of liver fibrosis is NAFLD, while the second is alcoholic liver disease due to long-term excesses of drinking alcohol.

Treatment Options

Treatment options for liver fibrosis usually depend upon the underlying cause of the fibrosis. A doctor will treat the underlying illness, if possible, to reduce the effects of liver disease. For example, if a person drinks alcohol excessively, a doctor may recommend a treatment program to help them stop drinking. If a person has NAFLD (Castera, 2012), a doctor may recommend making dietary changes to lose weight and taking medications to promote better blood sugar control. Exercising and losing weight may also help to reduce the disease's progression.

A doctor may also prescribe medications known as antifibrotics, which have been shown to reduce the likelihood that liver scarring will occur. The antifibrotic prescribed usually depends on the underlying medical condition. Examples of these treatments include:

- **Chronic Liver Disease**: ACE inhibitors, such as benazepril, Lisinopril, and Ramipril.
- **Hepatitis C Virus**: a-Tocopherol or interferon-alpha.
- **Nonalcoholic Steatohepatitis**: PPAR-alpha agonist.

While researchers are conducting many tests to try to find medications that can reverse the effects of liver fibrosis, there are no medications that can accomplish this currently.

If a person's liver fibrosis advances to where their liver is very scarred and does not work (ElHefnawi et al., 2012), a person's only treatment is often to receive a liver transplant. However, the waiting list is long for these transplant types and not every person is a surgical candidate.

ADVANTAGES OF MACHINE LEARNING

- Machine learning has many wide applications (e.g., banking and financial sector, healthcare, retail, and publishing).
- Google and Facebook are using machine learning to push relevant advertisements. That advertisements are based on users past search behavior.

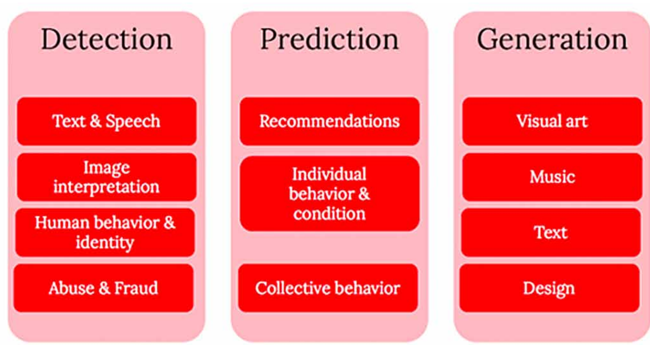
- Machine learning is used to handle multidimensional and multivariety data in dynamic environments.
- Machine learning allows time cycle reduction and efficient utilization of resources.

As there are too many things that come under practical benefit of machine learning. Also, involve the development of autonomous computers, software programs. Hence, it includes processes that can lead to automation of tasks.

THREE MAIN CATEGORIES OF MACHINE LEARNING APPLICATIONS AND THEIR USE CASES

The power of learning algorithms comes down to two major applications: Detection and prediction. Detection is about interpreting the present, and prediction is the way of the future. Interestingly, machines can also do generative or "creative" tasks. However, these are still a marginal application.

Figure 6. Three main categories of machine learning applications and their use cases



LITERATURE SURVEY

Overview of Literature Survey

Literature survey is the most important step in the process of software development . Before developing the tool, it is necessary to determine the time factor, economy, and company strength. Once these things are satisfied, then next step is determinign which operating system and language can be used for developing the tool. Once the programmers start building the tool, the programmers need much external support. This support can be obtained from senior programmers, from books or from Web sites. Before building the system, the above consideration are taken into account for developing the proposed system.

Title: *Accurate Prediction of Advanced Liver Fibrosis Using the Decision Tree Learning Algorithm in Chronic Hepatitis C Egyptian Patients*

Authors: S. Hashem et al.
Year: 2016

Respectively with the prevalence of chronic hepatitis C in the world, using noninvasive methods as an alternative in staging chronic liver diseases for avoiding the drawbacks of biopsy is significantly increasing. The aim of this study is to combine the serum biomarkers and clinical information to develop a classification model that can predict advanced liver fibrosis. 39,567 patients with chronic hepatitis C were included and randomly divided into two separate sets. Liver fibrosis was assessed via METAVIR score; patients were categorized as mild to moderate (F0–F2) or advanced (F3-F4) fibrosis stages. Two models were developed using alternating decision tree algorithm. Model 1 uses six parameters, while model 2 uses four, which are similar to FIB-4 features except alpha-fetoprotein instead of alanine aminotransferase. Sensitivity and receiver operating characteristic curve were performed to evaluate the performance of the proposed models. The risk of advanced liver fibrosis, due to chronic hepatitis C, could be predicted with high accuracy using decision tree learning algorithm that could be used to reduce the need to assess the liver biopsy.

Title: *A Simple Multi-Linear Regression Model for Predicting Fibrosis Scores in Chronic Egyptian Hepatitis C Virus Patients*
Authors: S. Hashem et al.
Year: 2014

Hepatitis C is considered as a common infection in Egypt, especially in genotype.

The prognosis of hepatitis C and the risk of developing cirrhosis are related to the stage of fibrosis. Liver biopsy is the best indicator for identifying the extent of liver fibrosis, but it has many drawbacks. Furthermore, it is costly and susceptible to sampling error. Noninvasive methods for the assessment of liver fibrosis are alternative in staging chronic liver diseases. The aim of this paper is to develop a simple multilinear model to predict the levels of risk for liver fibrosis based on standard laboratory tests. In this proposed model, liver fibrosis was assessed via Metavir score; patients were categorized as mild (F0-F1), moderate (F2), or advanced (F3-F4) fibrosis stages. Statistical analysis was performed using Med Calc software. The relationship between serum markers and the presence of significant fibrosis was assessed. The P-value and the correlation coefficients revealed that, age, AST, AFP, Albumin, platelet count, Glucose, Postprandial Glucose test and BMI, were significantly associated with fibrosis. Multilinear regression analysis is performed to develop a model for prediction of liver fibrosis scores based on serum markers. Sensitivity and receiver operating characteristic (ROC) curve analysis were performed to evaluate the proposed model. It has been concluded that, multi-linear regression model can predict fibrosis stages in chronic hepatitis C with accepted accuracy that could be used to reduce the need to assess the liver biopsy.

Title: *Invasive and NonInvasive Diagnosis of Cirrhosis and Portal Hypertension*
Authors: M. Y. Kim and W. K. Y. Jeong,
Year: 2014

With advances in the management and treatment of advanced liver disease, including the use of antiviral therapy, a simple, one stage description for advanced fibrotic liver disease has become inadequate. Although refining the diagnosis of cirrhosis to reflect disease heterogeneity is essential, current diagnostic tests have not kept pace with the progression of this new paradigm. Liver biopsy and hepatic venous pressure gradient measurement are the gold standards for the estimation of hepatic fibrosis and portal hypertension (PHT), respectively, and have diagnostic and prognostic value. However, are invasive and, as such, cannot be used repeatedly in clinical practice. The ideal noninvasive test should be safe, easy to perform, inexpensive, reproducible as well as to give numerical and accurate results in real time. It should be predictive of long term outcomes related with fibrosis and PHT to allow prognostic stratification. Recently, many types of noninvasive alternative tests have been developed and are under investigation. In particular, imaging and ultrasound based tests, such as transient elastography, have shown promising results. Although most of these noninvasive tests effectively identify severe fibrosis and PHT, the methods available for diagnosing moderate disease status are still insufficient, and further investigation is essential to predict outcomes and individualize therapy in this field.

Title: *Liver Fibrosis Recognition Using Multicompression Elastography Technique*
Authors: A. Wahba, N. Mohammed, and A. Seddik,
Year: 2013

Liver fibrosis recognition is an important issue in diagnostic imaging. The accurate estimation of liver fibrosis stages is important to establish prognosis and to guide appropriate treatment decisions. Liver biopsy has been for many years the reference procedure to assess histological definition for liver diseases. But biopsy measurement is an invasive method besides it takes large time. So, fast and improved methods are needed. Using elastography technology, a correlation technique can be used to calculate the displacement of liver tissue after it has suffered a compression force. This displacement is related to tissue stiffness, and liver fibrosis can be classified into stages according to that displacement. The value of compression force affects the displacement of tissue and so affects the results of the liver fibrosis diagnosing. By using finite element method, liver fibrosis can be recognized directly within a short time.

Title: *Prospective Non-Invasive Follow-Up of Liver Fibrosis in Patients with Chronich Hepatitis C*
Authors: D. Crisan et al.
Year: 2012

Noninvasive methods for the assessment of liver fibrosis are accurate in staging chronic liver diseases before treatment. To prospectively assess liver fibrosis in chronic hepatitis C (CHC) in patients treated vs untreated, using non-invasive methods. 224 patients with CHC were included in the study: 179 received antiviral treatment for 48 weeks, and 45 patients received no antiviral therapy. All patients underwent liver biopsy at baseline and were also evaluated by simple biological scores (APRI, HAPRI, Forns, Bonacini, Lok) and transient elastography (TE). The progression of fibrosis was non-invasively assessed over a period of 72 weeks. The prospective follow-up of liver fibrosis assessed by simple biological scores and TE in patients with CHC revealed a down staging of fibrosis in treated patients and especially in those who gained SVR.

Title: *Noninvasive Methods to Assess Liver Disease in Patients with Hepatitis B or C*
Author: L. Castera
Year: 2012

The prognosis and management of patients with chronic viral hepatitis B and C depend on the amount and progression of liver fibrosis and the risk for cirrhosis. Liver biopsy, traditionally considered to be the reference standard for staging of fibrosis, has been challenged over the past decade by the development of noninvasive methodologies. These methods rely on distinct but complementary approaches: a biologic approach, which quantifies serum levels of biomarkers of fibrosis, and a physical approach, which measures liver stiffness by ultrasound or magnetic resonance elastography. Noninvasive methods were initially studied and validated in patients with chronic hepatitis C but are now used increasingly for patients with hepatitis B, reducing the need for liver biopsy analysis. review the advantages and limitations of the noninvasive methods used to manage patients with chronic viral hepatitis B or C infection.

Title: *Artificial Neural Network Aided Noninvasive Grading Evaluation of Hepatic Fibrosis by Duplex Ultrasonography*
Authors: L. Zhang, Q.-Y. Li, and Y.-Y.Duan
Year: 2012

Artificial neural networks (ANNs) are widely studied for evaluating diseases. This paper discusses the intelligence mode of an ANN in grading the diagnosis of liver fibrosis by duplex ultrasonogaphy. 239 patients who were confirmed as having liver fibrosis or cirrhosis by ultrasound guided liver biopsy were investigated in this study. Quantified ultrasonographic parameters as significant parameters using a data optimization procedure applied to an ANN. 179 patients were typed at random as the training group; 60 additional patients were consequently enrolled as the validating group. Performance of the ANN was evaluated according to accuracy, sensitivity, specificity, Youden's index and receiver operating characteristic (ROC) analysis. The established ANN model had good sensitivity and specificity in quantitative diagnosis of hepatic fibrosis or liver cirrhosis. Our study suggests that the ANN model based on duplex ultrasound may help noninvasive grading diagnosis of liver fibrosis in clinical practice.

Title: *Accurate Prediction of Response to Interferon-Based Therapy in Egyptian Patients with Chronic Hepatitis C Using Machine-Learning Approaches*
Authors: M. ElHefnawi et al.
Year: 2012

Hepatitis C virus' patients with genotypes 1 & 4 have break-even response rates to Pegylated-Interferon (Peg-IFN) and Ribavirin (RBV) treatment. Furthermore, the incompliance to the treatment because of its high cost and related unfavorable effects makes its prediction of paramount importance. By using machine-learning techniques, a significantly accurate predictive model constructed to predict Egyptian patients' response based on their clinical and biochemical data. The model uses Artificial Neural Networks (ANN) and Decision Trees (DT) to achieve this goal. Two-hundred patients treated with Peg-IFN and RBV; 83 responders (41%), and 117 non-responders (59%) retrospectively studied to extract informative features and train the Neural Networks and Decision Trees. Optimization done by using six different Neural Network architectures, starting with an input layer of 12 neurons, a hidden layer of 70 to 180 neurons and an output layer containing a single neuron. For decision Trees (DTs), the CART classification algorithm was used. conclude that decision trees gave a higher accuracy in predicting response and would help in proper therapy options for patients.

Title: *Prevalence of Hepatitis B Virus (HBV) and Hepatitis C Virus (HCV) Infections among Blood Donors at Al-Thawra Hospital Sana'a City-Yemen*
Author: A. Q. Alodini
Year: 2012

Hepatitis is a disease of the liver caused by the infectious and non-infectious agents. Hepatitis B and C are major public health problems worldwide. The aim of the study was to estimate the prevalence of hepatitis B and hepatitis C viruses among voluntary of healthy blood donors at Al-Thawra Hospital Sana'a City – Yemen, during February to April 2010. The data from Blood bank in Al-Thawra Hospital were collected and analyzed. All samples were tested by Enzymes Linked Immunosorbent Assay (ELISA) test. Blood donors at Al-Thawra Hospital Sana'a have a lower prevalence for infection with HBV and HCV compared to other Arabic countries.

Title: *Clinical Practice, Chronic Hepatitis C Infection*
Author: H. Rosen
Year: 2011

A 45-year-old man undergoing a routine examination for life insurance is noted to have an aspartate aminotransferase level of 80 U per milliliter (normal range, 9 to 40)and an alanine aminotransferase level of 110 U per millilitre (normal range, 7 to 52) reports a remote history of intravenous drug use. Tests for hepatitis C antibody and hepatitis B surface antibody are positive, and tests for hepatitis A and human immunodeficiency virus (HIV) antibodies are negative. Genotyping of the hepatitis C virus (HCV) reveals genotype 1b; the viral load is 2,460,000 IU per milliliter. The complete blood count is normal; the platelet count is 220×109 per liter. An abdominal ultra-sonogram is normal.

Title: *Lipid Profile Among Chronic Hepatitis C Egyptian Patients and Its Levels Pre and Post Treatment*
Author: E. Nashaat
Year: 2010

Hepatitis C is a common infection in the Egyptian population, specially genotype 4. It is well recognized in many studies that hepatitis C chronic infection is associated with hypolipidemia, so in this study the author compares the lipid profile between 150 patients with chronic hepatitis C and 150 normal persons with comparable age, sex, and body mass index (BMI). The fasting cholesterol, low density lipoprotein (LDL), high density lipoprotein (HDL), and triglyceride were compared. Then, 36 patients of them received treatment in the form of pegylated interferon and ribavirin and then the patients who achieved viral clearance were reevaluated as regard the lipid profile versus the patients who did not achieve viral clearance and the relpsers.

Title: *Diagnosis of Cirrhosis by Transient Elastography (FibroScan): A Prospective Study*
Authors: J. Foucher et al.
Year: 2006

Transient elastography (FibroScan®) is a new non-invasive rapid and reproducible method, allowing evaluating liver fibrosis by measurement of liver stiffness. In cirrhotic patients, liver stiffness measurements range from 12.5 to 75.5 kPa. However, the clinical relevance of these values is unknown. The aim of this prospective study was to evaluate the accuracy of liver stiffness measurement for the detection of cirrhosis in patients with chronic liver disease. 711 patients with chronic liver disease were studied. Aetiologies of chronic liver diseases were HCV or HBV infection, alcohol, NASH, other or combination of above aetiologies. Liver fibrosis was evaluated according to METAVIR. Transient elastography is a promising noninvasive method for the detection of cirrhosis in patients with chronic liver disease. Its use for the follow-up and management of these patients could be of great interest and should be further evaluated.

Title: *Performance of Serum Marker Panels for Liver Fibrosis in Chronic Hepatitis C*
Authors: J. Parkes, I. N. Guha, and P. Roderick
Year: 2006

Chronic hepatitis C (CHC) is characterised by hepatic fibrosis, used as a proxy measure of prognosis. Liver biopsy is a flawed reference standard and serum markers of fibrosis offer an attractive alternative. A systematic review was conducted to assess the performance of panels of serum markers of hepatic fibrosis in CHC, incorporating analyses placing markers in a clinical context. Serum markers can rule-in or rule-out fibrosis in up to 35% of patients, but cannot differentiate stages of fibrosis reliably.

Improvement of index and reference test in needed including evaluation of clinical outcomes as reference. Improved test reporting is needed to derive LR and DOR as performance indicators.

Title: *Sampling Error and Intraobserver Variation in Liver Biopsy in Patients with Chronic HCV Infection*
Authors: A. Regev et al.
Year: 2002

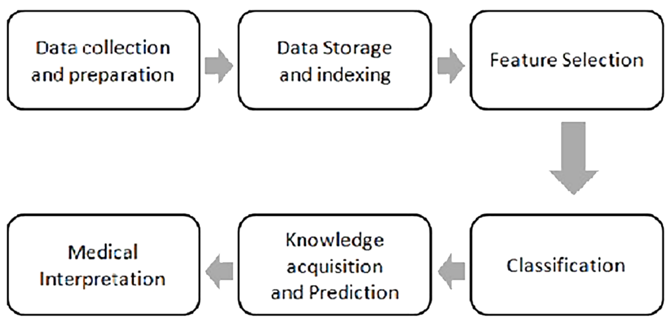
Needle liver biopsy has been shown to have a high rate of sampling error in patients with diffuse parenchymal liver diseases. In these cases, the sample of liver tissue does not reflect the true degree of inflammation, fibrosis, or cirrhosis, despite an adequate sample size. The aim of this study was to determine the rate and extent of sampling error in patients with chronic hepatitis C virus infection, and to assess the intraobserver variation with the commonly used scoring system proposed. A total of 124 patients with chronic hepatitis C virus infection underwent simultaneous laparoscopy-guided biopsies of the right and left hepatic lobes. Formalin-fixed paraffin-embedded sections were stained with hematoxylin and eosin and with trichrome. The slides were blindly coded and randomly divided among two hepatopathologists. Inflammation and fibrosis were scored according to the standard grading (inflammation) and staging (fibrosis) method based on the modified Scheuer system. Following the interpretation, the slides were uncoded to compare the results of the right and left lobes. Fifty of the samples were blindly resubmitted to each of the pathologists to determine the intra observer variation.

SYSTEM ANALYSIS

Existing System

Existing system (Gravitz, 2011) use of noninvasive methods as alternative in staging chronic liver diseases have significantly increased, in attempt to avoid the drawbacks of biopsy. Serum markers of liver fibrosis offer an attractive alternative to liver biopsy. less invasive than biopsy, with no risk of complications, eliminate sampling and observer variability, easy to perform, and can be performed repeatedly. Noninvasive methods in detection of fibrosis even based on indexes derived from serum markers (Hashem et al., 2014), such as FIB-4 score and the aspartate aminotransferase (AST)-to-platelet ratio index (APRI), or based on imaging techniques, such as using transient elastography (TE), which used ultrasound and vibratory waves for estimating the extent of liver fibrosis.

Figure 7. Framework of machine learning model to predict the individual stages of hepatic fibrosis in patients with HCV



Disadvantages of Existing System

- This technique is expensive and cannot yet be regularly used in all medical institutions
- This work to avoid complicated calculations and shortcoming cannot be eliminated.
- Complex serum panels does not contain direct biomarkers
- It is required to enhance the diagnostic reliability.
- Long-term process and it is given the low prevalence of cirrhosis in the general population.

Proposed System

The propose system aims to compare and evaluate the usefulness of different machine learning techniques in prediction of advanced fibrosis by combining the serum biomarkers and clinical information to develop the classification models. Grey wolf optimization, particle swarm optimization, decision tree, multilinear regression, genetic algorithm models, and random forest classification models for advanced fibrosis risk prediction were developed. The proposed models should be easy to perform, inexpensive, and give numerical and accurate results in real time. These models predicted the presence of advanced liver fibrosis with high accuracy and correlation coefficient especially with alternating decision tree and particle swarm optimization algorithms.

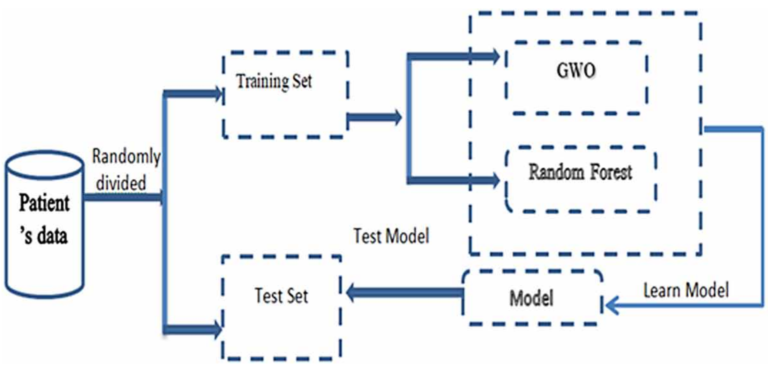
Advantages of Proposed System

- High accuracy.
- High speed.
- Easy to perform, inexpensive, and gives numerical and accurate results in real time.
- Reduced error-pruning tree (REP).
- Acceptable, safe, and low-cost, alternating for predict fibrosis rather than relatively risky alternative tools in chronic hepatitis C virus patients.

Block Diagram

See Figure 8.

Figure 8. Block diagram of proposed system



Description

- Preprocessing.
- Statistical feature extraction.
- GWO-based feature selection.
- Random forest classification.

Preprocessing

In these study patients records are collected from machine learning repository. Database contains 1230 male patients and 540 female patients. The lab tests also conducted at a same time of liver biopsy. The datasets are randomly divided into testing and training sets. In this time, 80% of the data are trained and 20% of the data should be tested based on the following Table. Table 1 explains the liver function test.

The patient's data are divided into three categories that is normal, mild-moderate and advanced stage of liver fibrosis.

Statistical Feature Extraction

Table 1. Liver function test

Liver Function Test			
	Normal	(Mild – Moderate)	Advanced
Bilirubin	0.3 – 1.0	1.0 - 2.5	> 2.5
Tot_Proteins	6 - 8.3	8.3 - 12	> 12
Albumin	3.5 - 4.5	3.0 - 3.5	< 3.0
A/g_Ratio	0.8 - 2.0	< 0.8	> 2.0
Sgpt	7- 56	56 – 200	> 200
Sgot	5 – 40	40 – 200	> 200
Alkphos	37 - 116	< 37	> 116

The data were statistically analyzed using MedCalc software and Microsoft Excel, while Matlab and WekaSoftwares performed the GWO and RFC learning algorithms. Data were reported as mean value ± standard deviation (SD). The relationship between variables and the presence of significant fibrosis has been assessed (P-value). The Kruskal-Wallis test has been used for continuous variables with non-normal distribution.

The Chi-square test has been used for categorical variables. Pearson correlation coefficients between fibrosis and each variable have been assessed.

Implemented several types of Machine learning techniques such as particle swarm optimization, genetic algorithm, multi-linear regression and decision tree learning algorithms. Decision tree algorithms; such as classification and regression tree (CART), reduced error-pruning tree (REP). Hashem et al. (2016) Evaluated the performance of each of them on the datasets. The test set represents an external data set that was not used for training. The receiver operating curves (ROCs), sensitivities, specificities, predictive values and accuracies were applied to evaluate the performance of each model or technique on both the training and test sets.

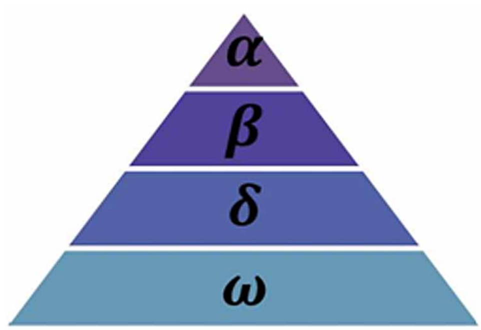
Grey Wolf Optimization

Grey wolves are considered to be at the top of food chain and prefer to live in a pack. [10] Four types of grey wolves such as alpha (α), beta (β), delta (δ), and omega (ω) are employed for simulating the leadership hierarchy. In order to mathematically model the social hierarchy of wolves while designing GWO, consider the fittest solution as the alpha (α). Consequently, the second and third best solutions are named as beta (β) and delta (δ), respectively. The rest of the candidate solutions are assumed to be omega (ω). Fig. 3 shows three main steps of GWO algorithm, namely hunting, chasing and tracking for prey, encircling prey, and attacking prey which is implemented to design GWO for performing optimization. In this section the inspiration of the proposed method is first discussed. Then, the mathematical model is provided.

Inspiration

Grey wolf (Canis lupus) belongs to Canidae family. Grey wolves are considered as apex predators, meaning that are at the top of the food chain. Grey wolves mostly prefer to live in a pack. The group size is 5-12 on average. Of particular interest is that have a very strict social dominant hierarchy (Figure 9).The leaders are a male and a female, called alphas. The alpha is mostly responsible for making decisions about hunting, sleeping place, time to wake, and so on. The alpha's decisions are dictated to the pack.

Figure 9. Hierarchy of grey wolf (dominance decreases from top down)



However, some kind of democratic behavior has also been observed, in which an alpha follows the other wolves in the pack. In gatherings, the entire pack acknowledges the alpha by holding their tails down (Nashaat, 2010). The alpha wolf is also called the dominant wolf since his/her orders should be followed by the pack . The alpha wolves are only allowed to mate in the pack. Interestingly, the alpha is not necessarily the strongest member of the pack but the best in terms of managing the pack. This shows that the organization and discipline of a pack is much more important than its strength.

The second level in the hierarchy of grey wolves is beta. The betas are subordinate wolves that help the alpha in decision-making or other pack activities. The beta wolf can be either male or female, and he/she is probably the best candidate to be the alpha in case one of the alpha wolves passes away or becomes very old. The beta wolf should respect the alpha, but commands the other lower-level wolves as well. It plays the role of an advisor to the alpha and discipliner for the pack. The beta reinforces the alpha's commands throughout the pack and gives feedback to the alpha.

The lowest ranking grey wolf is omega. The omega plays the role of scapegoat. Omega wolves always have to submit to all the other dominant wolves. The last wolves that are allowed to eat. It may seem the omega is not an important individual in the pack, but it has been observed that the whole pack face internal fighting and problems in case of losing the omega. This is due to the venting of violence and frustration of all wolves by the omega(s). This assists satisfying the entire pack and maintaining the dominance structure. In some cases the omega is also the baby sitters in the pack.

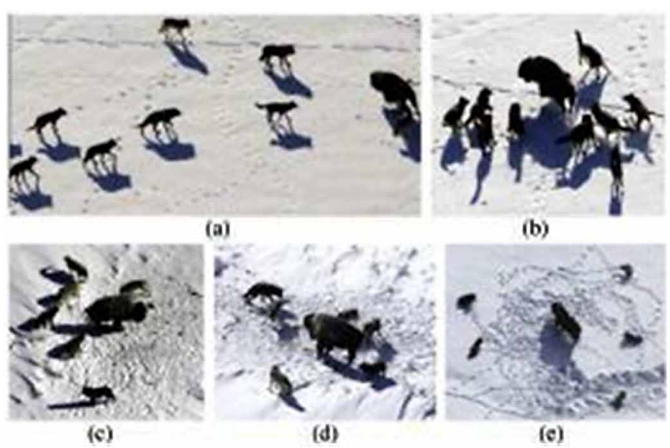
If a wolf is not an alpha, beta, or omega, he/she is called subordinate (or delta in some references) (Parkes, Guha, Roderick, & Rosenberg, 2006). Delta wolves have to submit to alphas and betas, but dominate the omega. Scouts, sentinels, elders, hunters, and caretakers belong to this category. Scouts are responsible for watching the boundaries of the territory and warning the pack in case of any danger. Sentinels protect and guarantee the safety of the pack. Elders are the experienced wolves who used to be alpha or beta. Hunters help the alphas and betas when hunting prey and providing food for the pack.

Finally, the caretakers are responsible for caring for the weak, ill, and wounded wolves in the pack. In addition to the social hierarchy of wolves, group hunting is another interesting social behavior of grey wolves. According to Muro et al. The main phases of grey wolf hunting are as follows:

1. Tracking, chasing, and approaching the prey.
2. Pursuing, encircling, and harassing the prey until it stops moving.
3. Attack towards the prey.

These steps are shown in Figure 10 in this work this hunting technique and the social hierarchy of grey wolves are mathematically modelled in order to design GWO and perform optimization.

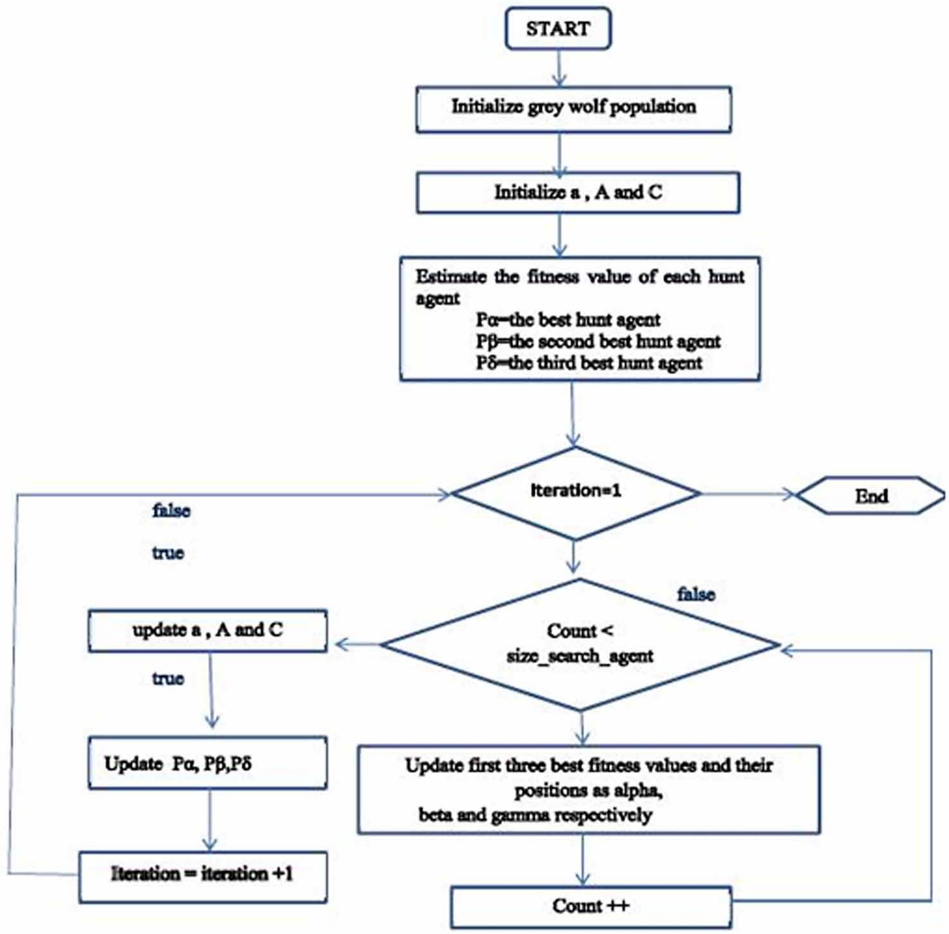
Figure 10. Hunting behavior of grey wolves: (a)–(c) chasing and tracking prey; (d) encircling prey; (e) attacking prey



Mathematical Equations

In order to mathematically model the social hierarchy of wolves when designing GWO, Regev et al. (2002)consider the fittest solution as the alpha (a). Consequently, the second and third best solutions are named beta (b) and delta (d) respectively. The rest of the candidate solutions are assumed to be omega (x). In the GWO algorithm, the hunting (optimization) is guided by a, b, and d. The x wolves follow these three wolves.

Figure 11. Flowchart of grey wolf optimization



Grey wolves encircle a prey during the hunt and the encircling behavior can be modeled by the following equations:

$$\vec{D} = \left| \vec{C} \cdot \overrightarrow{X_p}(t) - \overrightarrow{X_p}(t) \right| \tag{1}$$

$$\vec{X}(t+1) = \vec{X}(t) - \vec{A} \cdot \vec{D} \tag{2}$$

where t denotes the current iteration, D, A, and C denote coefficient vectors, Xp is the position vector of the prey, and X indicates the position vector of grey wolf.

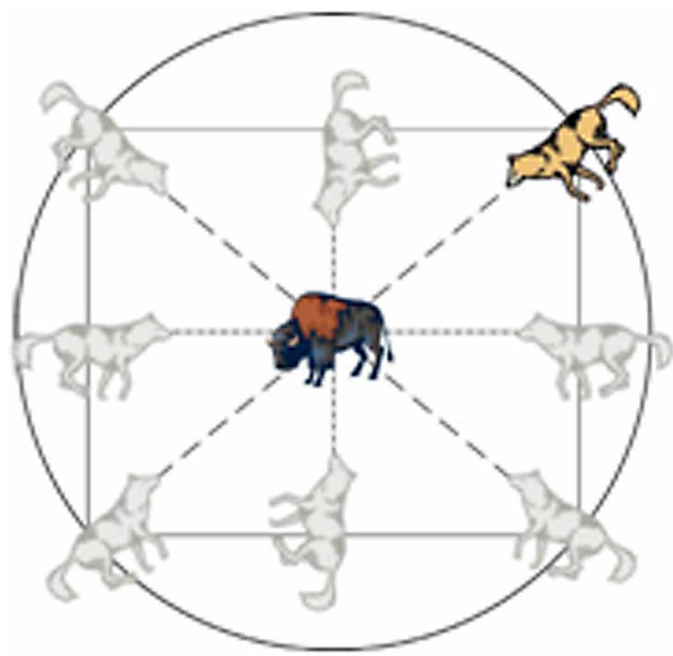
The vectors A and C are calculated as follows:

$$\vec{A} = 2\vec{a} \cdot \vec{r}_1 - \vec{a} \tag{3}$$

$$\vec{C} = 2 \cdot \overrightarrow{r_2} \tag{4}$$

where components of a linearly decreases from 2 to 0 during the course of iterations and r1, r2 are random vectors in [0, 1]. The hunt is usually guided by alpha called leaders followed by beta and delta which might also participate in hunting occasionally.

Figure 12. 2D Position vectors and their possible next locations



The pseudo code of the GWO algorithm is presented in Figure 12 To see how GWO is theoretically able to solve optimization problems, some points may be noted:

- The proposed social hierarchy assists GWO to save the best solutions obtained so far over the course of iteration.
- The proposed encircling mechanism defines a circle-shaped neighbourhood around the solutions which can be extended to higher dimensions as a hypersphere.
- The random parameters A and C assist candidate solutions to have hyperspheres with different random radii.
- The proposed hunting method allows candidate solutions to locate the probable position of the prey.
- Exploration and exploitation are guaranteed by the adaptive values of a and A.

- The adaptive values of parameters a and A allow GWO to smoothly transition between exploration and exploitation.
- With decreasing A, half of the iterations are devoted to exploration (|A|P1) and the other half are dedicated to exploitation (|A| < 1).
- The GWO has only two main parameters to be adjusted (a and C).

Pseudo Code of the GWO Algorithm

There are possibilities to integrate mutation and other evolutionary operators to mimic the whole life cycle of grey wolves. However, the authors have kept the GWO algorithm as simple as possible with the fewest operators to be adjusted. Such mechanisms are recommended for future work.

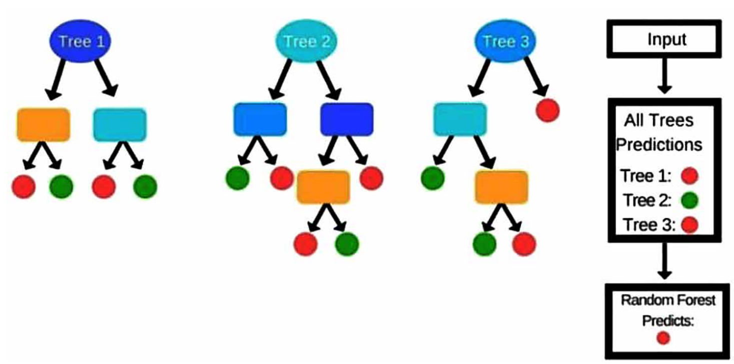
Applications of GWO

1. Machine learning applications.
2. Engineering applications.
3. Wireless sensor network applications.
4. Environmental modelling applications.
5. Medical and bioinformatics application.
6. Image processing applications.

Random Forest Classification

Random Forest is an adaptable, simple to utilize Machine learning calculation that produces, even without hyper-parameter tuning, an extraordinary outcome more often than not. It is likewise a standout amongst the most utilized calculations since it's straightforwardness and the way that it very well may be utilized for both characterization and relapse assignments.

Figure 13. How random forest model works



How Random Forest Model Works

Random forest is a supervised learning algorithm. Like can already see from its name, it creates a forest and makes it somehow random. The “forest” it builds, is an ensemble of Decision Trees, most of the time trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result.

Therefore, in random forest, only a random subset of the features is taken into consideration by the algorithm for splitting a node (Rosen, 2011). Can even make trees more random, by additionally using random thresholds for each feature rather than searching for the best possible thresholds (as a normal decision tree does). Figure 13 explains how the random forest classification performed.

Another great quality of the random forest algorithm is that it is very easy to measure the relative importance of each feature on the prediction (Sterling et al., 2006). Sklearn provides a great tool for this, that measures a features importance by looking at how much the tree nodes, which use that feature, reduce impurity across all trees in the forest. It computes this score automatically for each feature after training and scales the results, so that the sum of all importance is equal to 1.

The pseudo code for random forest algorithm can split into two stages:

1. Random forest creation pseudo code.
2. Pseudo code to perform prediction from the created random forest classifier. with the creation of the random forest pseudo code implies the following steps:
    a. Randomly select “k” features from total “m” features.
        i. Where k << m.
    b. Among the “k” features, calculate the node “d” using the best split point.
    c. Split the node into daughter nodes using the best split.
    d. Repeat a to c steps until “l” number of nodes has been reached.
    e. Build forest by repeating steps 1 to 4 for “n” number times to create “n” number of trees.

The beginning of random forest algorithm starts with randomly selecting “k” features out of total “m” features. Figure 13 highlights that they are randomly taking features and observations. The next stage consists in using the randomly selected “k” features to find the root node by using the best split approach. The next stage will be calculating the daughter nodes using the same best split approach. The first 3 stages are necessary to form the tree with a root node and have the target as the leaf node (Zhang et al., 2012).

Finally, the repetition of stages 1 to 4 is necessary to create “n” randomly created trees. These randomly created trees form the random forest.

Then, to perform prediction using the trained random forest algorithm, it is necessary to use the pseudocode below:

1. Take the test features and use the rules of each randomly created decision tree to predict the outcome, then store the predicted outcome (target).
2. Calculate the votes for each predicted target.
3. Consider the high voted predicted target as the final prediction from the random forest algorithm.

In order to perform the prediction using the trained random forest algorithm, it is necessary to pass the test features through the rules of each randomly created trees. If, for example, 100 random decision trees are required to form the random forest, each random forest will predict a different target (outcome) for the same test feature. Then, by considering each predicted target, votes will be calculated. If the 100 random decision trees are prediction some 3 unique targets x, y, z, then the votes of x is nothing but out of 100 random decision tree how many trees prediction is x.

Likewise for other 2 targets (y, z). If x is getting high votes, for example, out of 100 random decision tree 60 trees are predicting the target will be x. Then, the final random forest returns the x as the predicted target. This concept of voting is known as majority voting.

Advantages of Random Forest Classification

1. The over fitting problem will never come when the random forest algorithm is used in any classification problem.
2. The same random forest algorithm can be used for both classification and regression task.
3. The random forest algorithm can be used for feature engineering, which means identifying the most important features out of the available features from the training dataset.

Application of Random Forest Classification

Some of the fields in which random forest algorithm is widely used are: Banking, medicine, stock market, and e-commerce.

SYSTEM REQUIREMENTS

Software Description

The name MATLAB stands for MATrixLABoratory. The tutorials are independent of the rest of the document. The primary objective is to help learn quickly the first steps. The emphasis here is on learning by doing. Therefore, the best way to learn is by trying it yourself. Working through the examples will give a feel for the way that MATLAB operates. This section will describe how MATLAB handles simple numerical expressions and mathematical formulas. MATLAB was written originally to provide easy access to matrix software developed by the LINPACK (linear system package) and EISPACK (Eigen system package) projects. MATLAB is a high-performance language for technical computing. It integrates a computation, visualization, and programming environment. Furthermore, MATLAB is a modern programming language environment: I has sophisticated data structures, contains built-in editing and debugging tools, and supports object-oriented programming. These factors make MATLAB an excellent tool for teaching and research.

MATLAB has many advantages, compared to conventional computer languages (e.g., C, and FORTRAN) for solving technical problems. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. The software package has been commercially available since 1984 and is now considered as a standard tool at most universities and industries worldwide. It has powerful built-in routines that enable a very wide variety of computations. It also has easy to use graphics commands that make the visualization of results immediately available. Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering.

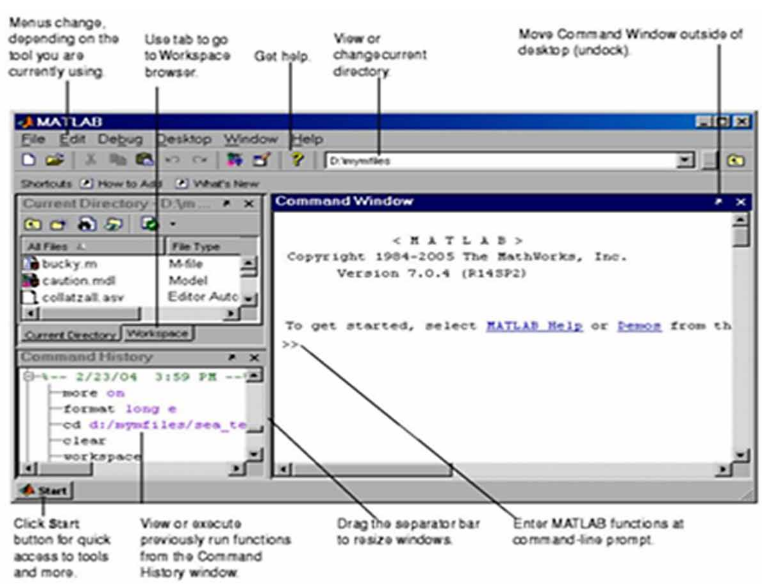
Starting Matlab

After logging into his/her account, the user can enter MATLAB by double-clicking on the MATLAB shortcut icon (MATLAB 7.0.4) on his/her Windows desktop. When the user starts MATLAB, a special window called the MATLAB desktop appears. The desktop is a window that contains other windows.

The major tools within or accessible from the desktop are:

- The Command Window.
- The Command History.
- The Workspace.
- The Current Directory.
- The Help Browser.
- The Start button.

Figure 14. The graphical interface to the MATLAB workspace



When MATLAB is started for the first time, the screen looks like the one in Figure 14. This illustration also shows the default configuration of the MATLAB desktop. The user can customize the arrangement of tools and documents to suit his/her needs. An example can be that the user wants to do some simple calculations and he/she has sufficient understanding of the computer under which MATLAB is being run.

Now faced with the MATLAB desktop on his/her computer, which contains the prompt (>>) in the Command Window. Usually, there are 2 types of prompt:

1. >>for full version.
2. EDU> for educational version.

In order to simplify the notation, the user will use the prompt >> as a standard prompt sign, though the authors' MATLAB version is for educational purpose.

Matlab Programming

So far in these lab sessions, all the commands were executed in the Command Window. The problem is that the commands entered in the Command Window cannot be saved and executed again for several times. Therefore, a different way of executing repeatedly commands with MATLAB is:

1. Creating a file with a list of commands.
2. Saving the file.
3. Running the file.

If needed, corrections or changes can be made to the commands in the file. The files that are used for this purpose are called "script ¯les" or "scripts" for short.

This section covers the following topics: M-file scripts and M-file functions.

A script file is an external file that contains a sequence of MATLAB statements. Script files have a filename extension .m and are often called M-files. M-files can be scripts that simply execute a series of MATLAB statements, or can be functions that can accept arguments and can produce one or more outputs. All variables created in a script file are added to the workspace. This may have undesirable effects because:

- Variables already existing in the workspace may be overwritten.
- The execution of the script can be affected by the state variables in the workspace.

As a result, because scripts have some undesirable effects, it is better to code any complicated applications using rather M-file functions.

Debugging Process

M-files can be debugged using the Editor/Debugger as well as using debugging functions from the Command Window. The debugging process consists of:

- Preparing for debugging.
- Setting breakpoints.
- Running an M-file with breakpoints.
- Stepping through an M-file.
- Examining values.

- Correcting problems.
- Ending debugging.

Preparing for Debugging

- Here the authors use the Editor/Debugger for debugging. They suggest to do the following to prepare for debugging:Open the file
- Save changes
- Be sure the file run and any ¯les it calls are in the directories that are on the search path.

Setting Breakpoints

1. Set breakpoints to pause execution of the function, so to examine where the problem might be. There are three basic types of breakpoints:
    a. A standard breakpoint, which stops at a specified line.
    b. A conditional breakpoint, which stops at a specified line and under specified conditions.
    c. An error breakpoint that stops when it produces the specified type of warning, error,
2. Nan, or infinite value cannot set breakpoints while MATLAB is busy, for example, running an M-file.

Running With Breakpoints

After setting breakpoints, run the M-¯les from the Editor/Debugger or from the Command Window. Running the M-file results in the following:

- The prompt in the Command Window changes to K>> indicating that MATLAB is in debug mode.
- The program pauses at the first breakpoint. This means that line will be executed when continue. The pause is indicated by the green arrow.
- In the breakpoint, the user can examine the variable, step through programs, and run other calling functions.

Examining Values

While the program is paused, the user can view the value of any variable currently in the workspace. He/she can examine values when he/she wants to see whether a line of code has produced the expected result or not. If the result is as expected, the user can step to the next line and continue running. If the result is not as expected, then that line, or the previous line, contains an error. When running a program, the current workspace is shown in the Stack field. It is possible to use who or whos to list the variables in the current workspace.

- Viewing values as data tips.
- First, position the cursor to the left of a variable on that line. Its current value appears.
- This is called a “datatip,” which is like a tooltip for data. In case of trouble getting the datatip to appear, the user can click in the line and then move the cursor next to the variable.

Figure 15. Home page of the liver disease detection using soft computing

Correcting and Ending Debugging

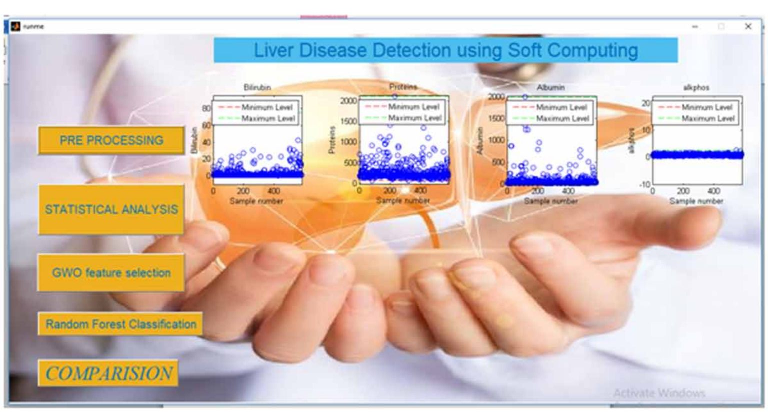
While debugging, the user can change the value of a variable to see if the new value produces the expected results. While the program is paused, the user has to assign a new value to the variable in the Command Window, Workspace browser or Array Editor. Then, he/she can continue running and stepping through the program.

RESULTS

Screen Shots

Given a dataset containing various attributes of 583 Indian patients, use the features available in the dataset and define a supervised classification algorithm which can identify whether a person is suffering from liver disease or not. The dataset for this problem is the ILPD (Indian Liver Patient Dataset) taken from the UCI Machine Learning Repository. Number of instances are 583. It is a multivariate data set,

Figure 16. Preprocessing of data such as bilirubin, proteins, albumin and alphos.



contain 10 variables that are age, gender, total Bilirubin, direct Bilirubin, total proteins, albumin, A/G ratio, SGPT, SGOT and Alkphos. Performance metrics on which the models would be evaluated were decided. The dataset was then split into training and testing set.

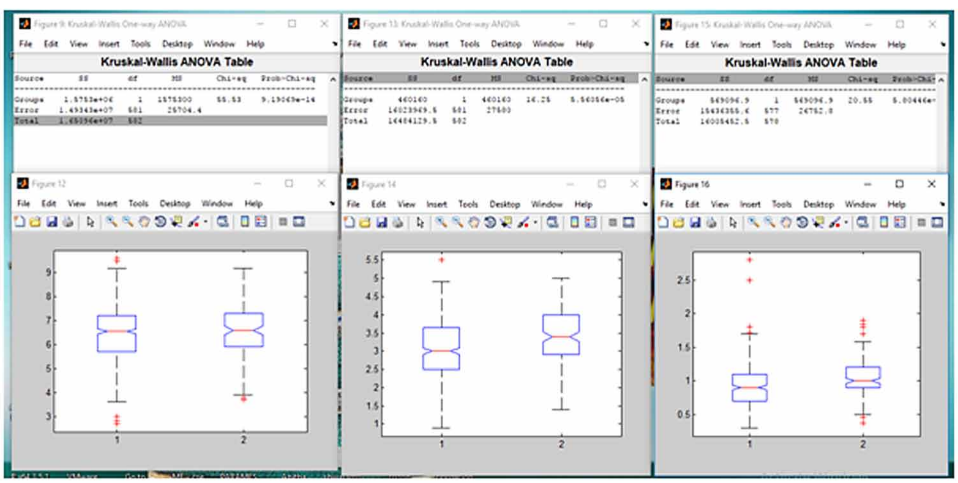
In this page, display the all modules and home page of the liver disease using soft computing. In this page contains following modules:

- Pre-processing.
- Statistical analysis.
- GWO feature selection.
- Random forest classification.

This module illustrates only preprocessing. First of all, it is important to explore all data (i.e., load data into this system). Every datum should be normalized and the gender column removed. In normalization find mean and variance of the given data for normalization.

This module illustrates statistical feature extraction by using Kruskal-Wallis test and one way ANOVA table. Kruskal-Wallis test is used to find the difference between two large groups of data.

Figure 17. Statistical feature extractions by using Kruskal-Wallis test and one way ANOVA table.

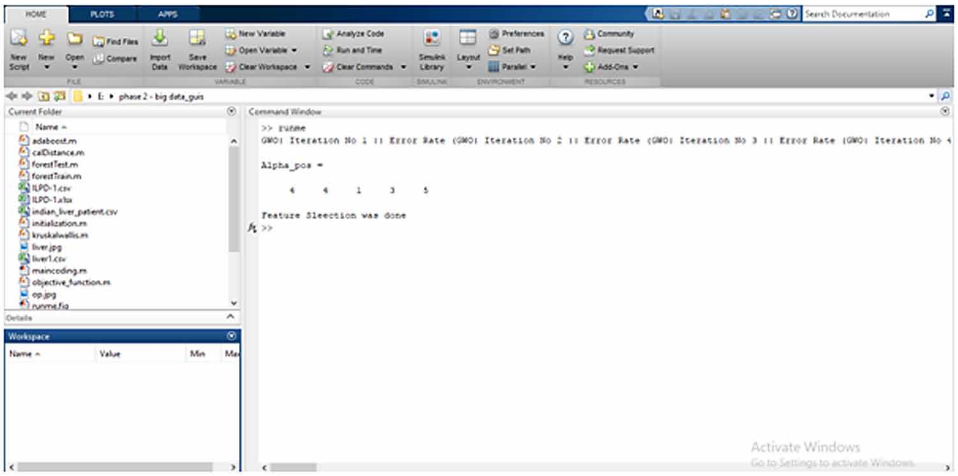


In statistical feature extraction, data were analysed using Mat-Lab software and Excel, while the software performed the optimization and classification learning. These data should be reported as mean and standard deviation values. The relationship between the two larger variables and the presence of significance of fibrosis has been analysed by using the Kruskal-Wallis test and one way ANOVA table.

Implemented several optimization and classification techniques, such as GWO, ANN, regression, SVM, and random forest. The ROCs accuracy and predictive values were evaluating the performance of both training and test data sets.

This module illustrates the GWO-based feature selection method. It is an effective feature selection method. First, it eliminates the redundant and irrelevant features by searching for the best features in

Figure 18. GWO-based feature selection



the patient's liver data set. GWO produces the initial positions of the population, and then updates the current positions of the population in the discrete searching space.

FUTURE RESEARCH DIRECTIONS

The future aim of this study is to test the patient's data by using the threshold values to create the user interface module by using artificial neural networks. In the future, further optimization of this technique can also be done. The future aim of this project is to test the data to analyze the patient's data features and give the results if the patients have chronic hepatitis C infection or not. If the patients have chronic hepatitis C infection, the authors' model shows how much the liver is affected and also gives suggestions.

CONCLUSION

In this project, the authors conducted a comparison between different machine learning approaches on prediction of advanced liver fibrosis in chronic hepatitis C patients. The rearchers developed grey wolf optimization models for classify the data. The authors concluded that this model could predict advanced fibrosis stage for chronic HCV patients using different machine learning approaches with high accuracy. They found the four parameters (i.e., age, AST, albumin, and platelets count) to be the most important features in the prediction of the advanced fibrosis as statistically they have significant relationship (P-value < 0.0001) and accepted correlation coefficients (j r j>0.1) with presence of advanced fibrosis, as the results showed. The proposed models could be used as an acceptable, safe, and low-cost alternative for predicting advanced fibrosis, rather than relatively risky alternative tools (e.g., the liver biopsy) in chronic hepatitis C virus patients.

REFERENCES

Alodini, Q. (2012). Prevalence of Hepatitis B Virus (HBV) and Hepatitis C Virus (HCV) infections among blood donors at Al-Thawra Hospital Sana'a City-Yemen. *Yemeni Journal for Medical Sciences*, *6*, 16–20.

Bedossa, B., & Carrat, F. (2009). Liver biopsy: The best, not the gold standard. *Journal of Hepatology*, *50*(1), 1–3. doi:10.1016/j.jhep.2008.10.014 PMID:19017551

Bonacini, M., Hadi, G., Govindarajan, S., & Lindsay, K. L. (1997). Utility of a discriminant score for diagnosing advanced fibrosis or cirrhosis in patients with chronic hepatitis C virus infection. *The American Journal of Gastroenterology*, *92*, 1302–1304. PMID:9260794

Castera, L. (2012). Noninvasive methods to assess liver disease in patients with hepatitis B or C. *Gastroenterology*, *142*(6), 1293–1302. doi:10.1053/j.gastro.2012.02.017 PMID:22537436

Crisan, D. (2012). Prospective noninvasive follow-up of liver fibrosis in patients with chronic hepatitis C. *Journal of Gastrointestinal and Liver Diseases; JGLD*, *21*, 375–382. PMID:23256120

ElHefnawi, M. (2012). Accurate prediction of response to Interferon-based therapy in Egyptian patients with Chronic Hepatitis C using machine-learning approaches. In *Proceedings of IEEE/ACM Int. Conf. Advances Social Netw. Anal. Mining* (pp. 771-778). 10.1109/ASONAM.2012.140

Gravitz, L. (2011). A smouldering public-health crisis. *Nature*, *474*(7350), S2–S4. doi:10.1038/474S2a PMID:21666731

Hashem, S. (2014). A simple multi-linear regression model for predicting fibrosis scores in chronic Egyptian hepatitis C virus patients. *International Journal of Biological Technology*, *4*(3), 37–46.

Hashem, S., Esmat, G., Elakel, W., Habashy, S., Abdel Raouf, S., Darweesh, S., ... ElHefnawi, M. (2016). Accurate prediction of advanced liver fibrosis using the decision tree learning algorithm in chronic Hepatitis C Egyptian patients. *Gastroenterology Research and Practice*, *2016*, 2636390. doi:10.1155/2016/2636390 PMID:26880886

Kim, M. Y., Jeong, W. K. Y., & Baik, S. K. (2014). Invasive and noninvasive diagnosis of cirrhosis and portal hypertension. *World Journal of Gastroenterology*, *20*(15), 4300–4315. doi:10.3748/wjg.v20.i15.4300 PMID:24764667

Nashaat, E. (2010). Lipid profile among chronic hepatitis C Egyptian patients and its levels pre and post treatment. *Nature and Science*, *8*(7), 83–89.

Parkes, J., Guha, I. N., Roderick, P., & Rosenberg, W. (2006). Performance of serum marker panels for liver fibrosis in chronic hepatitis C. *Journal of Hepatology*, *44*(3), 462–474. doi:10.1016/j.jhep.2005.10.019 PMID:16427156

Regev, A., Berho, M., Jeffers, L. J., Milikowski, C., Molina, E. G., Pyrsopoulos, N. T., ... Schiff, E. R. (2002). Sampling error and intra-observer variation in liver biopsy in patients with chronic HCV infection. *The American Journal of Gastroenterology*, *97*(10), 2614–2618. doi:10.1111/j.1572-0241.2002.06038.x PMID:12385448

Rosen, H. (2011). Clinical practice, chronic Hepatitis C infection. *The New England Journal of Medicine*, *364*(25), 2429–2438. doi:10.1056/NEJMcp1006613 PMID:21696309

Sterling, R. K., & ... . (2006). Development of a simple non-invasive index to predict significant fibrosis in patients with HIV/ HCV co-infection. *Hepatology (Baltimore, Md.)*, *43*, 1317–1325. doi:10.1002/hep.21178 PMID:16729309

Zhang, L., Li, Q.-Y., Duan, Y.-Y., Yan, G.-Z., Yang, Y.-L., & Yang, R.-J. (2012). Artificial neural network aided noninvasive grading evaluation of hepatic fibrosis by duplex ultrasonography. *BMC Medical Informatics and Decision Making*, *12*(1), 55. doi:10.1186/1472-6947-12-55 PMID:22716936

Chapter 6
Deep Neural Network-Based Android Malware Detection (D-AMD)

Sangeetha D.
Anna University, MIT Campus, India

Umamaheswari S.
https://orcid.org/0000-0002-9931-9243
Anna University, MIT Campus, India

Rakshana Gopalakrishnan
https://orcid.org/0000-0003-2182-9494
Anna University, MIT Campus, India

ABSTRACT

Android is an operating system that presently has over one billion active users for their mobile devices in which a copious quantity of information is available. Mobile malware causes security incidents like monetary damages, stealing of personal information, etc., when it's deep-rooted into the target devices. Since static and dynamic analysis of Android applications to detect the presence of malware involves a large amount of data, deep neural network is used for the detection. Along with the introduction of batch normalization, the deep neural network becomes effective, and also the time taken by the training process is less. Probabilistic neural network (PNN), convolutional neural network (CNN), and recurrent neural network (RNN) are also used for performance analysis and comparison. Deep neural network with batch normalization gives the highest accuracy of 94.35%.

DOI: 10.4018/978-1-7998-2108-3.ch006

INTRODUCTION

Android, developed by Google, is a layered and open-source operating system that uses a Linux kernel, and is primarily designed for touch screen mobile devices (Liu & Yu, 2011; McLaughlin et al., 2017). The core applications, middleware, and an operating system form the basis of Android. As the Android operating system (OS) became increasingly desired, widely used Android devices include smartphones, tablets, and e-readers.

Despite the different hardware platforms and OSs, computer users encounter the threat of malware (Zabidi, Maarof, & Zainal, 2012). It is not possible to prevent malware from entering the system. With proper detection and monitoring, a system can be kept safe from any kind of hacking problems. Detection of malware in any application is very important to keep things safe.

A major threat of cybersecurity is mobile malware. Moreover, new security threats are pioneered by the daily emerging new mobile malware (Sen, Aydogan, & Aysan, 2018). Mobile malware causes loss or leakage of confidential information and the collapse of the system. Also, it is being a tedious task to make certain that the wireless-enabled personal digital assistants and the mobile phones are safer and possess enough security to withstand these malware attacks, as these devices are having high complexity. Mobile devices are a potential cryptocurrency mining tool, as they are manufactured with powerful graphics processors. In addition, since they are universal and easy to use, it was observed that there is an increase in trojan miner attacks manifold in 2018. According to a Kaspersky security report (Pei, J. Li, H. Li, Gao, & Wang, 2017), 884,774 new malware, three times higher than the number present in 2014, were introduced in 2015. Symantec (2016) additionally rumored that one zero-day attack per week on the average was discovered in 2015. Along with the new families of malware in 2015 (6%), there was a big increase within the volume of Android variants (40%) (2016). As a legitimate software package, malware has evolved over the years and comes with diverged functions, depending on the intent of the developer. Though the amount of recently developed families of Android mobile malware seemed to diminish in the past two years, it has been observed that there is an imperative growth in diversity in the variations of Android malware families (2016).

Owing to their high popularity, Android devices are highly targeted. As per the GDATA report (Pei et al., 2017), 750,000 new Android malware were found in the first quarter of 2017. A large range of mobile malware is expected to develop in the future (Pei et al., 2017). With the development of high quality mobile devices such as smartphones or tablets, attacks on them are increasing. Also, as Android OS allows the users to install third-party applications, it can deceive the users to download the malware from the attacker's servers. In order to address the fast increase within the range of mobile malware, several firms have delivered their own personal mobile protection options, which principally support both static and dynamic analysis. Moreover, several studies within the literature and academia also propose mobile malware detection methods.

Detecting malware in Android applications is a tedious process, since the growth of malware is exponential. Hence, some of the challenges the developers encounter are: Malware writers introduce garbage calls to confuse the analysts with faux application programming interface (API) calls; malware writers encrypt the necessary details at intervals of the malware body (e.g., XOR); malware is also packed using a documented packer. "Packing" is a technique to compress Windows executables. Malware may well be analyzed finely by unpacking them. However, the analysis on the code by unpacking the .apk file without the help of packer is tough and time-consuming, and also safe setting is required to confirm that the malware being analyzed will not infect elsewhere. Many malware Android functions are repackaged around popular Android applications that are available on the market. The repacked malware functions consist of taking advantage of root degree exploits, altering of the machine password, communicating with external servers and making cellphone calls.

Malware detection approaches are categorized into two categories: Dynamic analysis based detection and static analysis based detection (Sen et al., 2018). The static analysis generally uses syntactical choices that might also be extracted, even besides executing the application, whereas when the application is analyzed dynamically, it uses semantic selections that can be monitored while the application is being executed in a controlled environment. Dynamic analysis has a bonus that it is plausible to cope with malicious functions that use some obfuscation strategies (e.g., code cryptography or packing), and static analysis has comparatively low overhead of the operations for evaluation.

Deep learning algorithms excel at extracting patterns from raw data, and, with large datasets, they have been very successful in natural language and computer vision applications (Goh, Sakloth, Siegel, Vishnu, & Pfaendtner, 2018). Deep learning in malware detection offers far better performance and scalability than the other classical machine learning algorithms. Since, deep learning allows the models to learn the data with multiple levels of abstraction, it has supremacy in terms of accuracy when trained with a huge amount of data. Besides, the benefit of deep learning models is their ability to perform feature learning. Each algorithm in the hierarchy of the models of deep learning applies a non-linear transformation on its entry. Besides, it uses what it learns to create a statistical model as output and the number of epochs for which the model is trained is dependent on the stop criteria.

LITERATURE SURVEY

This section provides an overview of the previous studies on mobile malware detection.

Wu, Mao, Wei, Lee, & Wu (2012) proposed a static feature-based mechanism is used to classify the application into either benign or malware. The feature they used is API calls and intent messages passing. Also, multiple clustering algorithms are used for malware modelling. Since the variants of Android malware are differentiated using these models, it is effective and also scalable. No dynamic simulation, deployment, and manual efforts in investigation are required, thus there is no increase in the environmental cost.

API invocation calls are used to create malware. Sundarkumar and Ravi (2013) proposed a static analysis method of extracting features by text mining the series of API calls from the dataset provided by the CSMINING group. The extracted features are subjected to various data mining techniques such as group method for data handling, support vector machine (SVM), multi layer perceptron, PNN, decision tree and one class SVM. These authors observed that one class SVM and SVM achieved a sensitivity of 100% by enforcing oversampling.

Huang, Tsai, and Hsu (2013) developed a model in which an Android application is identified as malware or benign based on the permissions it requests from the user. It also uses some easily retrievable features for better efficiency. This proposal allows to obtain quick filtering of applications that are malicious. However, it requires a second pass to perform a complete analysis of the application. The application identified as malicious may not be malware just because of the requested permissions.

API calls are used by the malware authors to fabricate malware which is considered to be one of the cyber frauds. Sundarkumar and Ravi (2013) proposed a static analysis method that uses text mining in tandem with data mining to obtain API call sequences to detect malware. A series of API calls from the dataset is extracted from the dataset as features using text mining. Further, for feature selection, mutual information is used and over-sampling is applied for balancing the dataset. PNN is applied for malware detection. Though it achieved high sensitivity, only static analysis is used, while all kinds of malware cannot be detected only with the static analysis.

Pei, J. Li, H. Li, Gao and Wang (2017) developed a technique in which the presence of malware is identified in the API level using an automated tool that uses a framework of API-level security certification of Android applications (ASCAA). The analysis of cloud platform, rather than an individual mobile, improves performance, and also the combination of static and dynamic features improves accuracy. However, it uses high CPU utilization, thus affecting the performance of host cellphone. Also, several security critical calls at the native code level were not identified.

McLaughlin et al. (2017) developed a method where a deep convolutional neural network is used for the detection process. A disassembled program of raw opcode sequence is statistically analyzed to classify the malware. The network routinely learns the features indicative of malware from the raw opcode sequence, thus eliminating the want for hand-engineered malware features. Since the time taken for training and testing is linearly proportionate to the tally of malware samples, it is more computationally effective than the present n-gram-based malware classification. The problem is that it uses only static analysis of features. More data augmentation schemes for malware detection are not investigated. Data augmentation helps to increase precision and accuracy.

Android malware detection can be of two types, namely behavior-based and code-based detection. Zhao, Zheng, Gong, Zhang and Wang (2018) proposed a behavior-based quick and accurate malware detection method using high sensitive APIs. The API calls are extracted through reverse analysis and represented as eigenvectors, and mutual information is applied between API calls and malware to identify the sensitive APIs. An ensemble model of k-nearest neighbors (KNN) classifier and decision tree is used for identification of unknown applications. Though this model achieves an accuracy of 92%, it uses a small set of malware and benign data, and also possesses a slightly high false positive rate.

Tao, Zheng, Guo and Lyu (2018) developed a random forest classifier model which mines the hidden patterns of the malicious applications to extract the widely used highly sensitive APIs used by the Android malware. MalPat, an automated malware detection tool, was constructed to address the unknown malicious applications. This model gives an F1 score of 98.24%. Though this model has high efficiency, it shows some problems. First, to avoid overfitting, a large scale dataset can be used. Second, since only a limited number of types of malware are present on the Internet, mining patterns would be hard to expand. Third, since categories of benign applications are not considered, categorical features may affect the identification of benign applications.

Kim, Kang, Rho, Sezer, and Im (2019) developed a method in which a multimodal deep learning model is used for the classification of the Android applications into malware or benign by analyzing the class files, manifest file, and the shared library files. The features used are dynamically analyzed for better efficiency. The important benefit of this method is that the dynamic analysis of features selected from the .apk file allows to classify the application more efficiently than the static analysis. Since a multimodal DNN is used, high computational power is required for the implementation; and also, a large number of computations is necessary to train the model.

Ioffe and Szegedy (2015) suggested that the training is tricky because the parameters of all preceding layers influence the inputs to each layer. Since the layers need to adapt to interminable new distributions, a problem arises when there is a change in input layers' distribution. When the input distribution to a learning machine changes, it is stated that the covariance shift is experienced. In order to reduce this shift and to use much higher learning rates, batch normalization (BN) can be used.

For the sparse nonlinearity inputs, no observations are made with and without BN. BN is also not tested on whether it can help in domain adaptation in its traditional sense, that is, whether the new data distributions are easily generalized by the models that use BN.

Some of the previous approaches use either static or dynamic analysis for feature extraction in the process of malware detection. However, not all kinds of malware can be detected when by these approaches. Also, not all the features can be extracted from the .apk file accurately, thus reducing the precision. Since multimodal DNN uses high computational power for the training process, DN with BN is used to decrease the computational power and number of computations, thus reducing the training time.

THE PROPOSED DEEP NEURAL NETWORK BASED ANDROID MALWARE DETECTION

Overview of the Proposed Deep Neural Network Based Android Malware Detection

Previous works that are associated with the Android malware detection systems use solely restrained kinds of features to identify the malware. Only a few properties of the application can be characterized by each feature. On the contrary, the proposed deep neural network based Android malware detection (D-AMD) uses different characteristics of applications retrieved from the information of the multiple features, considered from multiple aspects to detect malware, and uses an algorithm, BN to decrease the training time with increased learning rate. The proposed D-AMD first extracts seven feature types (ie., permission feature, environment feature, component feature, string feature, method API feature, method OpCode feature, and shared library function feature), and converts them into respective feature vectors, which are then combined to five feature vectors (ie., permission/environment/component feature vectors, string feature vector, method API feature vector, method OpCode feature vector, and shared library function feature vector). Despite having multiple similar properties between malware and benign applications, similarity-based method and existence-based method can be used for feature vector generation for effectively distinguishing between malware and benign. With the generated feature vectors, the DNN, PNN and recurrent neural network (RNN) are used to train and classify the data acquired from the dataset for which a performance analysis is performed. BN algorithm is used for the normalization of weights when input is passed from one layer to another to reduce the internal covariance shift. This algorithm helps the deep learning models to reduce the training time effectively.

Analysis of Malware Data

The dataset the authors collected are .apk files that contain both malware and benign application files. Android package kit (APK) file has an extension of .apk which is of JAR file format which is packed in the form of zip format-type packages; also, it is a type of archive file. It has around 349 malware .apk files and 340 benign .apk files (2018, August 31). The malware dataset is collected from the VirusShare (2019) and Android malware dataset (n.d.). The benign dataset is collected from the Google Play Store and the University of New Brunswick (2017). Table 1 shows the basic file structure of an .apk file.

Architecture

Table 1. Types of files in an Android application

File Name	Description
AndroidManifest.xml	An XML file that contains details about the application required to be executed in Android OS
Classes.dex	A file which contains the bytecode of all classes in the application
Resourses.arsc	A binary XML that is compiled from the pre-defined resources
res/	A folder that contains resources.arsc
assets/	A folder that holds the assets of the application (optional)
lib/	A folder that contains the compiled code that can be used as dependency in other applications.
META-INF/	A folder that stores metadata about the JAR files in manifest.mf file

The proposed D-AMD consists of four phases, which are raw data extraction, feature extraction, feature vector generation, and detection using DNN. Figure 1 shows the architecture of the proposed D-AMD.

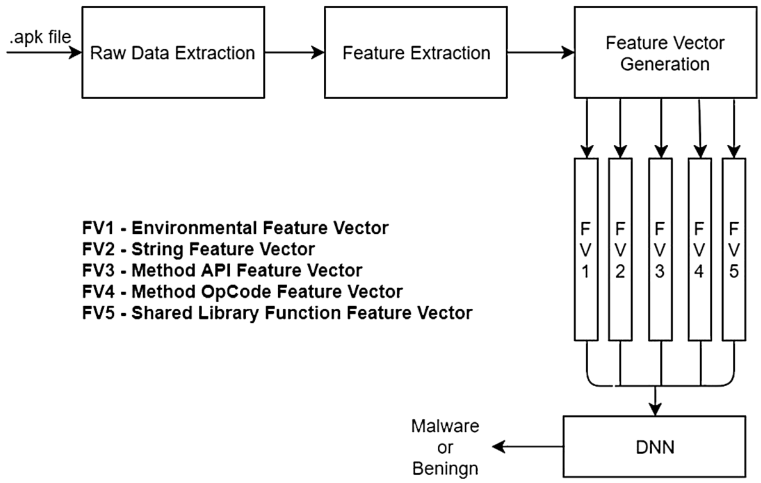
In the initial step, the .apk file is unzipped or decompiled to extract the source code of the considered Android application. The unzipped Android application will contain all the files required to run an application in mobile. In the second step, the extracted files are accessed to retrieve the seven feature types considered (ie., permission feature, environment feature, component feature, method opcode feature, string feature, shared library function feature and method API feature). The first three types are from .manifest file, the fourth and sixth type from the .smali file, the fifth type from .dex file, and seventh type from .so file. From the feature types extracted from all the Android applications in the dataset, a malicious database is created with the values extracted from the dataset. With this dataset, seven types of feature vectors are generated and are later converted into five types of feature vectors. These generated feature vectors are used to train the DNN.

Modules of Deep Neural Network Based Android Malware Detection

The proposed D-AMD is categorized into five modules based on the functionality:

1. **Vulnerability Analysis**: In this phase, a scan is done to find potential security vulnerabilities in Android applications to identify any code that could inject malware, use of risky and unwanted APIs or any dangerous influences in databases and data storage units, and file and security permissions injected through errors in code. If an Android application does not have any critical vulnerabilities, then the probability of injection of malware is less to minimum, ergo examining whether the application is vulnerable to malware injection. The tools AndroBugs ("AndroBugs framework", n.d.), a Python-based Android vulnerability detection system to identify security violations in Android applications, and Quixxi ("How secure is your mobile app?", n.d.), a Web application tool that uses penetration testing in Android applications, are used for vulnerability analysis tests. The common OWASP mobile vulnerabilities are:

Figure 1. Architecture of the proposed deep neural network based Android malware detection



a. **Improper Platform Usage**: It covers the misuse of a platform function or failure to use platform security controls of the mobile software system.
b. **Insecure Data Storage**: Insecure information storage vulnerabilities occur when the data in the file system of the mobile device are not properly encrypted.
c. **Broken Authentication**: The prevalence of broken authentication is significant due to the layout and improper implementation of most access and identity controls. Session management is the bedrock of the access controls and authentication, and exists in all stateful applications. Attackers can notice damaged authentication through manual means and take advantage of them through the usage of computerized tools with password lists and dictionary attacks. Session management attacks can be launched with the assist of unexpired session tokens ("OWASP mobile top 10", n.d.).
d. **Insecure Communication:** Mobile applications often do not defend network traffic. Only authentication is secured by SSL/TLS. This inconsistency ends up in the chance of exposing knowledge and session IDs to interception. The employment of transport layer security does not ensure that the application has enforced it properly and the exploitability issue can be found by observing a network for insecure communications ranges. In general, targeted attacks are easier to perform.
e. **Insufficient Cryptography:** Insecure use of cryptography is common in most mobile applications. This vulnerability can end in the unauthorized retrieval of sensitive data from the mobile device.
f. **Insecure Authorization:** Authorization is the act of checking that the known individual has the permissions necessary to perform the act. Once the hacker understands the authorization theme is vulnerable, logging in as a legitimate user bypassing the authentication management can be done with success. Once past authentication, a vulnerable endpoint may be force-browsed to execute administrative body practically. This submission method is usually done via mobile malware at intervals the device or botnets owned by the aggressor. Steps of Vulnerability Analysis:

   i. Identify known protection exposures earlier than attackers locate them
   ii. Create a stock of all the devices on the network, together with purpose and system information. This also consists of vulnerabilities associated with a specific device
   iii. Create an inventory of all units in the business enterprise to help with the planning of improvements and future assessments
   iv. Define the stage of danger that exists on the network
   v. Establish a commercial enterprise risk/benefit curve and optimize safety investments

2. **Raw Data Extraction:** In order to interpret and read the APK files, raw data extraction process is performed. To extract the data, the associated APK file is unzipped with the help of the APKtool ("A tool for reverse engineering Android apk files", n.d.), a third-party, closed, and binary tool for reverse-engineering the Android applications which will rewrite resources to almost original type; it needs Java eight (JRE Toolkit 8). It is used to extract the .manifest file, .dex file, and shared library files (i.e., .so files). The shared library files, the dex file, and the manifest file are decoded or disassembled using IDA-Pro ("About IDA", n.d.), a disassembler for computer software which generates assembly language source code from machine-executable code. However, it requires greater human intervention as it is not completely automated and the disassembling process is not of 100% accuracy.
3. **Feature Extraction Process**: To gather the necessary information from the extracted raw data from the android application, the feature extraction process is executed. The features String and Method opcode is extricated from the .smali files. Method API feature is extracted from the class files which are the disassembled results of a .dex file where disassembling is done using IDA-Pro. The instruction sequences of each function present in the disassembled .so files are scanned line by line to extract the essential feature information regarding assembly opcode frequency. From the .manifest XML file, the environmental features, the permission features and the component features are retrieved. The following list shows the features and its description after extraction from the .apk file.
   a. **Permission Feature:** From the .manifest xml file, the android:name from the <permission> and <uses-permission> tags are extracted as the required values. WRITE_CALENDAR, WRITE_CONTACTS,NFC,ACCESS_COARSE_LOCATION,READ_PHONE_STATE,INTERNET,WRITE_EXTERNAL_STORAGE,PROCESS_OUTGOING_CALLS,RECORD_AUDIO,READ_CALENDAR,CALL_PHONE,READ_LOGS,READ_HISTORY_BOOKMARKS,READ_SMS,RECEIVE_MMS,GET_ACCOUNTS,SEND_SMS,MOUNT_UNMOUNT_FILESYSTEMS,WRITE_SMS,WRITE_HISTORY_BOOKMARKS are some of the risky permissions. Some Benign and malicious applications share a fair amount of same requested permissions.
   b. **Environmental Feature:** This feature consists of the hardware and software used for the development of Android applications. These details are extracted from the <uses-sdk>, <uses-library> and <uses-feature> tags present in .manifest xml file. ANDROID.HARDWARE.CAMERA.FLASH, ANDROID.HARDWARE. TELEPHONY are examples for the environmental features.
   c. **Component Feature:** This feature contains the details of the four components of the Android: Content Provider, Activity, Broadcast Receiver, and Service. An Activity the window in which the application draws its user interface. A Service is a component that can function long-running operations in the background, and would not supply a UI (2019, December 27).

Android apps can send or receive broadcast messages from the Android system and other Android applications. Applications can register to receive unique broadcasts. A provider is part of an Android application, which regularly presents its very own UI for working with the data. Intents are a component of an Android application that allows it to register and receive messages, and is used to transfer any principal information to any other component or to commence the working of any component. It also defines the working mechanism of communication between the components. The manifest file is parsed to get these attributes from <activity>, <service>, <receiver> and <provider> XML tags, which are paired with their corresponding <intent-filter> tags.

d. **String Feature:** From a set of .smali files, this type of feature is retrieved and can be accessed from the disassembled .apk file. The smali files are parsed, and *const-string/jumbo* and *const-string* operand values are extracted. The extracted strings are hashed using SHA-512 since all of them are not of the same length.

e. **Method Opcode Feature:** This feature embodies Dalvik Opcode frequency, as it implies a coder's coding habits and behaviors. Each method's byte code is scanned to get Method Opcode frequency.

f. **Method API Feature:** This feature embodies API invocation calls, as it also implies a coder's coding habits and behaviors. Each method is scanned for specific API call invocation and its frequency is calculated. Only selected sets of API invocations are considered and scanned in the Java code. Statistical patterns of API utilization are studied, and the top 25 most regularly used APIs in both benign and malware applications are taken into account. Some of the API IDs that appear more frequently in the malicious applications present in the dataset are 9723, 14824, 15057, 15225, and 17626.

g. **Shared Library Function Opcode Frequency:** In Android, native libraries can be used through the Java native interface. Since the native code is not covered by the security model, Android security mechanisms are defeated leaving it vulnerable to malicious attacks. From the native code, Advanced RISC machine opcode and systems call invocation frequencies are calculated by counting the system call invocations and opcodes in each function to prevent the malware from hiding in native codes.

4. **Feature Vector Generation**: The malicious feature database is an archive containing the seven feature representations of already known malicious applications. The extracted features from the extraction process are used to construct feature vectors. Based on the feature representation, the seven types of feature vectors are divided into two categories: Similarity-based and existence-based feature vectors. The similarity-based feature vector is constructed based on the similarities between the malware present in the feature database. This type of feature is clustered using k-means clustering algorithm and the distance between the centroid of the clusters is calculated using Euclidean distance. The existence-based feature vector is the feature vector that represents the presence of features in the malicious database. The values in this type of feature vector are designated as follows: If the extracted value is present in the malicious feature database, the position of that feature in the database is represented as 1, otherwise as 0.

5. **Malware Detection:** Once the seven types of feature vectors are generated by the two methods, they are given as an input to the neural network models to perform the detection process. However, before feeding the input feature vectors, the component feature vector, the permission feature vector, and the environmental feature vector are merged into a single feature vector. Therefore, the proposed

D-AMD considers five feature vectors as the input feature vectors, and, in each layer, mathematical calculations are performed based on the activation functions (rectified linear unit [ReLU]). The given input Android applications are assigned with a label of either benign or malware once the neural network model has completed its execution successfully. To decrease the internal covariance shift between each layer in the DNN, weights are normalized using BN.

a. **Multimodal Deep Neural Network (m-DNN):** The information from multiple sources is used for the multimodal learning (Liu & Yu, 2011). In the multimodal fusion setting, data from all modalities are available at all phases (Jiquan et al., 2011). Combining different modalities or types of information for improving performance seems intuitively an appealing task, but it is also a challenging and humungous task, as varying types of modalities are need to be combined. Moreover, modalities have different quantitative influence over the prediction output. The high level embeddings from the different inputs are combined by concatenating them; then, applying a softmax is the frequently used method. The m-DNN comprises of five DNNs which train the five feature vectors separately. The results from these DNNs are combined by concatenating through a softmax layer (merging layer), which is given to another DNN for classification. Each of the five DNNs encompasses three hidden layers. The sixth DNN contains a merging layer, three hidden layers, and an output layer. The activation function used in hidden layers is ReLU and in the output layer it is a Sigmoid function.
b. **Deep Neural Network**: Deep learning is the future of *a*rtificial *i*ntelligence. The deep learning network is efficient, as it can handle a humungous amount of data through multiple layers*;* also, highly complex features of the data can be learnt by the network at each layer. The weight, bias, and activation function of each layer determine the strength of the corresponding layer in the network model. The proposed D-AMD uses three hidden layers with ReLU activation function, as it reduces the likelihood of vanishing gradient and the output layer utilizing a *s*igmoid activation function. A sigmoid function asymptotes to zero and asymptotes to one, outputting values between zero and one. A value of above or below 0.5, indicates which of the two classes the data is predicted to be classified. *In order to* reduce the overfitting of the DNN model, a dropout rate of 0.2 is used as dropout is used for regularization.
c. **Batch normalization**: In order to reduce the internal covariance shift, normalization algorithms can be used. As the distribution of the activations between the intermediate layers constantly changes during the training process (Ioffe & Szegedy, 2015), the training time is increased because in every training step, each layer must learn to adapt to a new distribution. BN can be used to normalize the input values between the layers. The ratio of the difference between the output from the previous layer and batch mean to the batch standard deviation is used to normalize the previous layer output to increase the stability of the network. Also, BN helps each layer of the network to be independent of the other layers, higher learning rates can be used, and it also reduces overfitting, since it possesses slight regularization effects. The algorithm used for BN is given by Sergey Ioffe, Christian Szegedy (2015) is,
    i. 5. c. i. Calculate the mean and variance of the layers input.

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \text{ // Batch Mean}$$

$$\sigma^2_B = \frac{1}{m}\sum_{i=1}^{m}\left(x_i - \mu_B\right)^2 \text{ //Batch Variance}$$

ii. 5. c. ii. Normalize the layer inputs based on already calculated batch values.

$$\overline{x_i} = \frac{\left(x_i - \mu_B\right)}{\sqrt{\sigma_B^2 + \varepsilon}}$$

iii. 5. c. iii. Scale and shift in order to obtain the layer output.

$$y_i = \gamma \overline{x_i} + \beta$$

In the above algorithm, m represents number of samples in the mini-batch considered, x_i represents the attribute value of a training sample, ε is arbitrary constant added for numerical stability, y_i is the output of corresponding training sample, γ and β are constants that are learned during optimization process, μ_B is the batch mean, and σ^2_B is the batch variance. The BN is performed for each mini-batch of the training dataset by normalizing the input features for each layer by calculating the corresponding batch mean and batch variance, and then the normalized weights are used to obtain the output of the layer.

PERFORMANCE EVALUATION

Experimental Environment

A Windows 10 Pro laptop with Intel Core i5-8th Gen CPU and 8GB RAM was used to set up the proposed framework. Python with PyPy applicator was used to implement the modules. The DNN model uses Keras library. Scikitlearn and Tensorflow are utilized to implement the machine learning and clustering algorithms.

Comparison With Different Neural Networks

The authors evaluated the performance of the proposed D-AMD by comparing it with the other existing neural networks, namely PNN, RNN, m-DNN, and DNN without BN. Table 2 shows the prediction accuracy, sensitivity, and specificity of different existing neural networks.

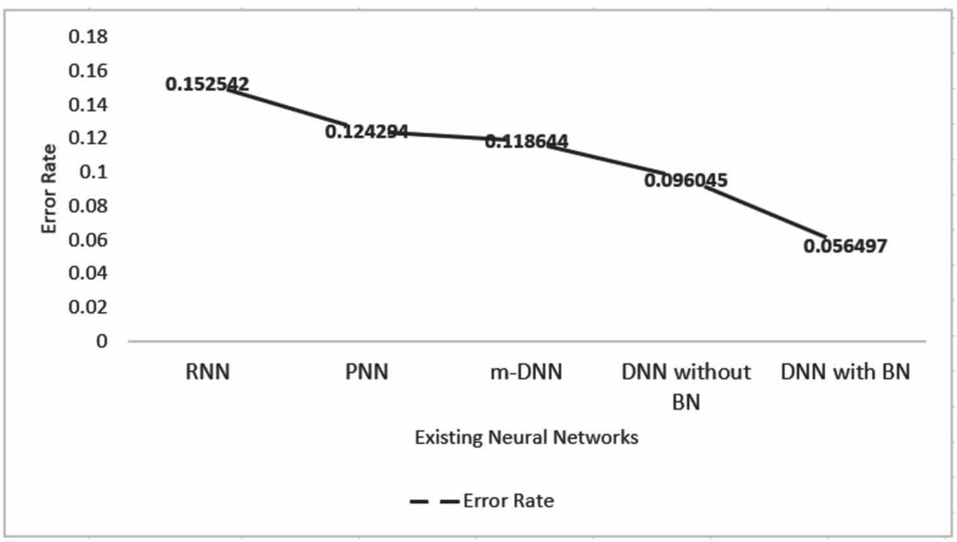
Sensitivity, also known as recall or true positive rate, is the ratio of actual positives to a total number of positives in the dataset. Specificity, also known as true negative rate, is the ratio of actual negatives to the total negatives considered. Sensitivity and specificity are the highest for DNN with BN with the values 0.927536 and 0.953704, respectively. Also, error rate is calculated for all the five existing models from which it is observed that RNN has the highest value of 0.152542 and DNN with BN has the low-

Table 2. Comparison of accuracy, sensitivity and specificity of predictions between the existing neural networks

Algorithm	Accuracy (%)	Sensitivity	Specificity
PNN	87.57	0.811594	0.916667
RNN	84.74	0.797101	0.879630
m-DNN	88.13	0.826087	0.916667
DNN without BN	90.39	0.869565	0.925926
DNN with BN	94.35	0.927536	0.953704

est error rate of 0.056497. Figure 2 shows the graph of the comparison of the error rates of the neural network models, and Figure 3 depicts the statistical measures sensitivity, specificity, and the accuracy of the considered neural networks.

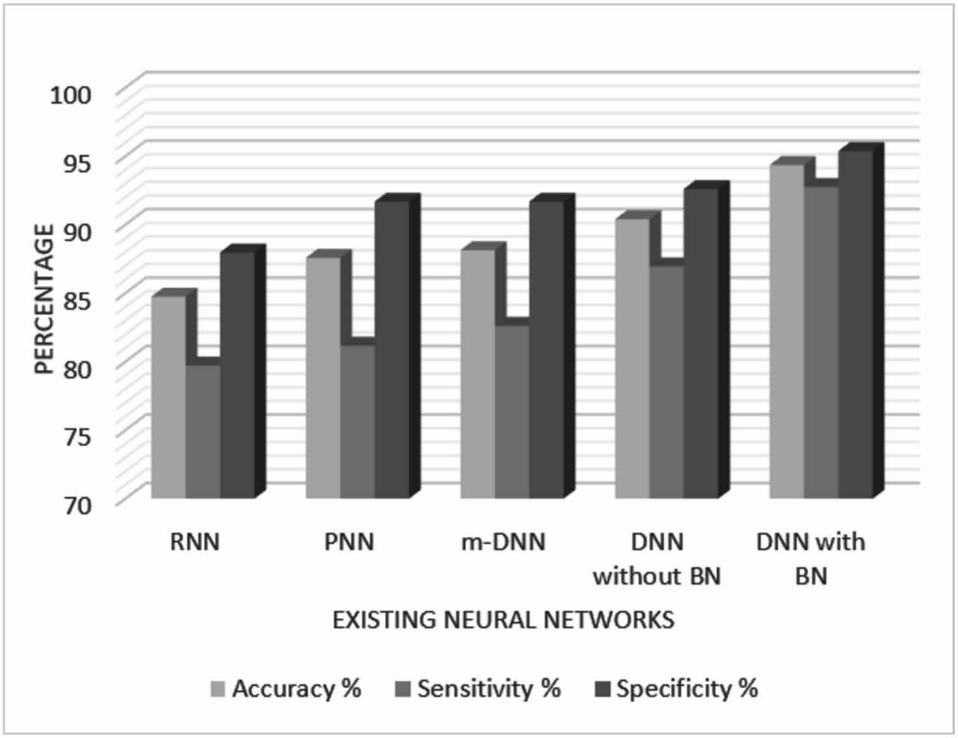
Figure 2. Comparison of error rates of existing neural networks models



In Figure 3, RNN has the lowest sensitivity and specificity, along with the lowest accuracy, whereas, when DNN with BN is used, the sensitivity and the specificity are the, indicating that DNN with BN had detected the potential malware with better accuracy.

The multimodal technique is used to aid in improvement in classification precision and accuracy. M-DNN uses a total of six DNNs within it which takes a large amount of time for the learning process. As the DNN uses only single network, even though its computational complexity is high when compared to multimodal network, it takes less time, since it takes time to learn a single network rather than six networks. The use of BN increases the learning speed and also accuracy. In the proposed D-AMD, the m-DNN takes one hour more than DNN with BN for training process.

Figure 3. Statistical measures of five neural network models



FUTURE RESEARCH DIRECTIONS

Though a small dataset will give a high precision of predictions, it does not take all kinds of Android applications into account. As this proposed D-AMD uses a small dataset, it can be ameliorated by using a large dataset for learning of the D-AMD. Also, this model can be modified to be maneuvered as a real-time model to predict the presence or absence of the malware in the Android application. Since this proposed framework only focuses on malware detection, it can be altered to remove the malware as well.

CONCLUSION

Utilizing an Android application with a malware embedded in it will affect the normal functioning of the application, and may also infect the mobile phones leading to security and privacy concerns. Hence, an application should be determined whether it is safe to use. Thus, for the malware detection, the authors proposed D-AMD, which utilizes the static and dynamic features extracted from the executable codes embedded in the compressed .apk file for identification of the presence of the malware. These features, once extracted are then converted to feature vectors of two types – similarity-based feature vectors and existence-based feature vectors. These feature vectors are used for the malware detection process (ie., to classify with the help of the algorithm DNN). The multimodal technique is used to aid in improvement in classification precision and accuracy. However, it takes more time to learn the model. Hence, the

DNN is used to reduce the training time. In order to increase the accuracy of predictions, BN algorithm is applied to DNN. Also, the dataset considered is trained with PNN and RNN, and the performance analysis is done on the basis precision of predictions. The authors observed that DNN with BN rendered a maximum accuracy of 94.35% while DNN without BN produced an accuracy of 0.903955 for prediction of malware or benign and took one hour less than the m-DNN for the training process.

REFERENCES

A tool for reverse engineering Android apk files. (n.d.). Retrieved from https://ibotpeaches.github.io/Apktool/

AboutI. D. A. (n.d.). Retrieved from https://www.hex-rays.com/products/ida/

AndroBugs framework. (n.d.). Retrieved from https://github.com/AndroBugs/AndroBugs_Framework

Android Malware Dataset. (n.d.). Retrieved from http://amd.arguslab.org/

Developer guides. (2019, December 27). Retrieved from https://developer.android.com/guide

Goh, G.B., Sakloth, K., Siegel, C., Vishnu, A., & Pfaendtner, J. (2018). *Multimodal Deep Neural Networks using Both Engineered and Learned Representations for Biodegradability Prediction*. ArXiv, abs/1808.04456

How secure is your mobile app? (n.d.). Retrieved from https://vulnerabilitytest.quixxi.com/#/

Huang, C. Y., Tsai, Y. T., & Hsu, C. H. (2013) Performance Evaluation on Permission-Based Detection for Android Malware. In Advances in Intelligent Systems and Applications - Volume 2. Smart Innovation, Systems and Technologies, vol 21. Springer. doi:10.1007/978-3-642-35473-1_12

Index of /iscxdownloads/CICAndMal2017/APKs. (2018, August 31). Retrieved from https://iscxdownloads.cs.unb.ca/iscxdownloads/CICAndMal2017/APKs

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *Proceedings of the 32nd International Conference on Machine Learning, in PMLR, 37*, 448-456.

Kim, T., Kang, B., Rho, M., Sezer, S., & Im, E. G. (2019). A Multimodal Deep Learning Method for Android Malware Detection Using Various Features. IEEE Trans. on Info Forensics and Sec., 14(3). doi:10.1109/TIFS.2018.2866319

Liu, J., & Yu, L. (2011). Research on Development of Android Applications. *4th International Conference on Intelligent Networks and Intelligent Systems*, 69-72. doi: 10.1109/ICINIS.2011.40

McLaughlin, N., Martinez del Rincon, J., Kang, B., Yerima, S., Miller, P., Sezer, S., ... Joon Ahn, G. (2017). Deep Android Malware Detection. *Proceedings of the ACM Conference on Data and Applications Security and Privacy (CODASPY) 2017 Association for Computing Machinery (ACM)*. 10.1145/3029806.3029823

OWASP mobile top 10. (n.d.). Retrieved from https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10

Pei, W., Li, J., Li, H., Gao, H., & Wang, P. (2017). ASCAA: API-level security certification of android applications. *IET Software, 11*(2), 55-63. doi:10.1049/iet-sen.2015.0040

Sen, S., Aydogan, E., & Aysan, A. I. (2018). Coevolution of Mobile Malware and Anti-Malware. *IEEE Transactions on Information Forensics and Security*, *13*(10), 2563–2574. doi:10.1109/TIFS.2018.2824250

Sundarkumar, G. & Vadlamani, R. (2013). Malware detection by text and data mining. *2013 IEEE International Conference on Computational Intelligence and Computing Research, IEEE ICCIC 2013*, 1-6. . doi:10.1109/ICCIC.2013.6724229

Symantec. (2016). *Internet security threat report*. Retrieved from https://www.east.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf

Tao, G., Zheng, Z., Guo, Z., & Lyu, M. R. (2018). MalPat: Mining Patterns of Malicious and Benign Android Apps via Permission-Related APIs. *IEEE Transactions on Reliability*, *67*(1), 355–369. doi:10.1109/TR.2017.2778147

VirusShare. (2019). Retrieved from https://virusshare.com/

Wu, D., Mao, C., Wei, T., Lee, H., & Wu, K. (2012). DroidMat: Android Malware Detection through Manifest and API Calls Tracing. *2012 Seventh Asia Joint Conference on Information Security*, 62-69. 10.1109/AsiaJCIS.2012.18

Zabidi, M. N. A., Maarof, M. A., & Zainal, A. (2012). Challenges in high accuracy of malware detection. *IEEE Control and System Graduate Research Colloquium,* 123-125. doi: 10.1109/ICSGRC.2012.6287147

Zhao, C., Zheng, W., Gong, L., Zhang, M., & Wang, C. (2018). Quick and Accurate Android Malware Detection Based on Sensitive APIs. *2018 IEEE International Conference on Smart Internet of Things (SmartIoT)*, 143-148. doi: 10.1109/SmartIoT.2018.00034

Chapter 7
Deep Learning With Conceptual View in Meta Data for Content Categorization

Asokan R.
https://orcid.org/0000-0003-1674-8200
Kongunadu College of Engineering and Technology, India

Preethi P.
Kongunadu College of Engineering and Technology, India

ABSTRACT

Data gathered from various devices have to be observed by human operators manually for extended durations which is not viable and may lead to imprecise results. Data are analyzed only when any unwanted event occurs. Machine-learning technology powers many aspects of modern society, from web searches to content filtering on social networks to recommendations on e-commerce websites, and it is increasingly present in consumer products. Machine-learning systems are used to identify objects in different forms of data. For decades, constructing a pattern-recognition, machine-learning system required careful engineering and domain expertise to design a feature extractor that transformed the raw data into a suitable internal representation, which the learning subsystem could detect patterns in the input by making use of and integrating ideas such as backpropagation, regularization, the softmax function, etc. This chapter will cover the importance of representations and metadata appendage and feature vector construction for the training deep models optimization.

INTRODUCTION

Artificial intelligence innovation powers numerous parts of current society, from web searches and substance separating on asocial networks it proposals on web-based business sites, and it is progressively more present in shopper items, for example, cameras and cell phones. AI (Artificial intelligence) frameworks are utilized to recognize questions in pictures, translate discourse into content, coordinate

DOI: 10.4018/978-1-7998-2108-3.ch007

news, posts, and items within clients' interests, and select pertinent aftereffects of pursuit. Progressively, these applications utilize a class of strategies called profound learning.

Customary AI systems were restricted in their capacity to process irregular information in their crude structure. For decades, constructing an example acknowledgment or AI framework required incautious building and impressive space skill to structure a feature extractor that could change the crude information, (for example, the pixel estimations of a picture) into a reasonable inward portrayal or highlight vector from which the learning subsystem, regularly a classifier, could recognize or order designs in the information.

By creating basic, non-straight modules that change the portrayal at one level (beginning with the crude contribution) into a portrayal at a higher, somewhat increasingly theoretical, level. With the structure of enough of these types of changes, exceptionally complex capacities can be educated. For characterization errands, higher layers of portrayal intensify parts of the information that are significant for segregation and smother unimportant varieties. A picture, for instance, comes as a variety of pixel esteems, and the educated highlights in the primary layer of portrayal commonly speak to the nearness or nonappearance of edges at specific indirections and areas in the picture. The subsequent layer commonly identifies themes by identifying specific courses of action of edges, paying little mind to small varieties in the edge impositions. The third layer may amass themes into bigger blends that compare to parts of commonplace articles, and ensuing layers would distinguish protests as a mixture of these parts. The key part of profound learning is that these layers of highlights are not planned by human architects; they are discovered from information utilizing a broadly useful learning methodology.

Deep learning is making real advances in the resolving of issues that have opposed the best endeavors of the man-made brainpower community for a long time. It has ended up being truly adept at finding confusing structures in high-dimensional information and is in this way applicable to numerous areas of science, business, and government. Notwithstanding, beating records in picture acknowledgment (Kumar, McCann, Naughton, & Patel, Bengio, 2015; Delalleau, & Le Roux, 2005; Bordes, Chopra, & Weston, 2014; Bottou2007) and discourse acknowledgment (Ciodaro, Deva, de Seixas, & Damazio, 2012; Collobert et al., 2011; it has beaten other AI strategies at foreseeing the activity of potential medication atoms (Ferrucci & Lally,2004) breaking down atom smasher information (Duda & Hart, 1973) recreating mind circuits and anticipating the impacts of transformations in non-coding DNA on quality articulation and malady (Farabet, Couprie, Najman, & LeCun, Helmstaedter e al., 2013). Perhaps more shocking is that profound learning has delivered amazingly encouraging outcomes for different errands in normal language understanding (Hinton et al., 2012), especially subject arrangement, conclusion examination, question noting (Miao, Li, Davis, & Deshpande, Miao et al., 2017) and language interpretation.

In general, deep learning will have many more accomplishments sooner rather than later in light of the fact that it requires very little manual designing, so it can undoubtedly exploit increments in the measure of accessible computation and information. A new learning calculations and models that are presently being created for profound neural systems will accelerate this advancement.

SUPERVISED LEARNING

The most widely recognized type of artificial intelligence (AI) is supervised learning. The vision is to manufacture a framework that can arrange pictures containing, a state, a house, a vehicle, an individual, or a pet. The initial stage is to gather an enormous informational collection of pictures of houses, automobiles, individuals, and pets, each marked with its classification. During preparation, the machine demonstrates a picture and creates a yield as a vector of scores, one for every class. It needs the ideal classes and to have the most elevated scores of all things considered, however this is will most likely not occur before preparation. Processing begins with a target work that estimates the blunder (or distance) between the yield scores and the ideal example of scores. The machine, at that point, changes its inward flexible parameters to decrease this mistake. These flexible parameters, frequently called "loads," are genuine numbers that can be viewed as "handles" that characterize the information yield function of the machine. In a typical profound learning framework, there may be a large number of these flexible loads, and a large number of named models with which to prepare the machine.

To appropriately change the weight vector, the learning calculation computes a slope vector that, for each weight, shows by what sum the mistake would increment or abatement, if the weight were expanded just barely. The weight vector is then balanced in the contrary direction to the slope vector.

The goal work, arrived at the midpoint of all the preparation models, can be regarded as a sort of uneven scene in the high-dimensional space of weight esteems. The negative angle vector shows the course of steepest plunge in this scene, taking it more like a base, where the yield mistake is generally less.

Practically speaking, most experts utilize a technique called "stochastic angle plummet" (SGD). This comprises of demonstrating the information vector for a couple of models, processing the yields and the blunders, computing the normal inclination for those models, and modifying the loads appropriately. The procedure is rehashed for some little arrangements of models, from the preparation set, until the normal of the target capacity stops diminishing. It is called stochastic in light of the fact that every little arrangement of models provides a loud gauge of the normal slope over all models. This basic methodology typically finds a decent arrangement of loads surprisingly immediate when contrasted, and unquestionably increasingly expands advancement techniques. Subsequent to preparing the presentation, the framework is estimated on an alternate arrangement of models called a "test set". This serves to test the speculation capacity of the machine, which is its capacity to create reasonable answers using new sources of information that were not observed during preparing.

Huge numbers of the current functional uses of AI utilize straight classifiers over hand-designed highlights. A two-class straight classifier processes the weighted whole of the element vector parts. On the chance that the weighted aggregate is over the edge, the information is named giving it a place with a specific class.

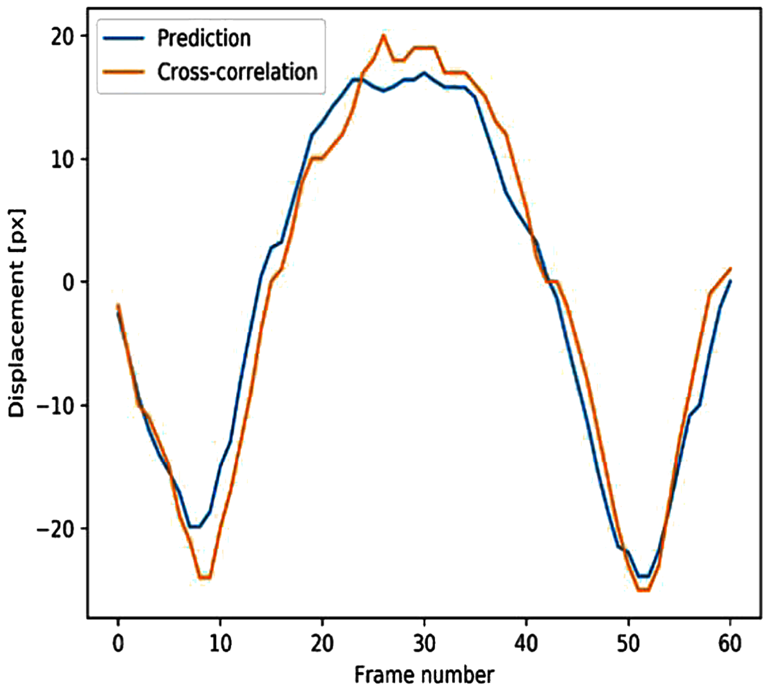
Since the 1960s, it was realized that direct classifiers can cut their information space into straightforward locales, specifically half-spaces separated by a hyper plane (Heller stein et al., 2017). Issues, for example, pictures and discourse recognition require the information yield capacity to be indifferent towards unessential varieties of the information. For example, varieties in position, direction, or enlightenment of an article, or varieties in the pitch or highlight of discourse, while being extremely sensitive to specific moment varieties (for instance, the contrast between a white wolf and a type of wolf-like white canine called a Samoyed). At the pixel level, pictures of two Samoyeds in various stances and in various situations might be altogether different from one another, while two pictures of a Samoyed and a poser positioned and on comparable foundations might be fundamentally the same as one another.

A direct classifier, or some other "shallow" classifier, working on a multi-layer neural system (appeared by the associated spots) can twist the information space to make the classes of information (instances of which are on the red and blue lines) straightly detachable.

How a standard framework (appeared on the left) in the information space and additionally changed (appeared in the center board) by concealed units. This is an illustrative model with just two information units, two shrouded units, and one yield unit, yet the systems utilized for item acknowledgment or regular language preparing contain a huge number of units. Recreated with authorization from Colah.

The chain standard (Graph 1) of subsidiaries discloses how two small impacts (that of a little difference in x on y, and that of y on z) are made. The little change Δx in x gets changed first into the little change Δy in y by getting increased by $\partial y/\partial x$ (that is, the meaning of fractional subsidiary).

Figure 1. Reconstructed three-dimensional perfusion distribution



Additionally, the changed Δy makes a change in Δz in z. Substituting one condition for other results in the chain standard of subsidiaries—how Δx gets transformed into Δz through augmentation by the result of $\partial y/\partial x$ and $\partial z/\partial x$. It also works when x, y, and z, is vectors (and the subordinates are Jacobian grids). The conditions utilized for figuring the forward go in a neural net with two shrouded layers and one yield layer, each comprising a module through which one can back propagate angles. At each yield layer, initially the absolute information z is figured to every unit, which is a weighted whole of the yields of the units in the layer underneath.

At that point, a non-straight capacity f(.) is connected to z to obtain the yield of the unit. For straightforwardness, predisposition terms should be excluded. The non-direct capacities utilized in neural systems incorporate the redressed straight unit (ReLU) $f(z) = \max(0,z)$, normally utilized in the same manner as the more ordinary sigmoid, for example, the hyperbolic digression

$$f(z) = (\exp(z) - \exp(-z))/(\exp(z) + \exp(-z))$$

and calculated capacity strategic,

$f(z) = 1/(1 + \exp(-z)\ d);$

the conditions utilized for registering the retrogressive pass.

At each concealed layer, compute the mistake subordinate concerning the yield of every unit, which is a weighted total of the blunder subsidiaries regarding the complete contributions to the units in the layer above. At that point, convert the blunder subordinate as for the yield into the mistake subsidiary as for the contribution by duplicating it by the slope of f(z). At the yield layer, the mistake subsidiary concerning the yield of a unit is processed by separating the cost capacity. This gives yl Δ tl, if the cost-capacity for unit l is 0.5 (yl Δ tl) 2, where tl is the objective worth. When the ΔE/Δzk is known, the mistake subsidiary for the weight wjk on the association from unit j in the layer underneath is simply yj ΔE/Δzk.

Crude pixels couldn't recognize the last two, while putting the previous two in a similar classification. This is the reason shallow classifiers require a decent component extractor that fathoms the selectivity-invariance situation-one that produces portrayals that are specific to the parts of the picture that are significant for separation, yet that are invariant to superfluous angles, for example, the posture of a creature.

To make classifiers all the more dominant, one can utilize conventional non-direct highlights, similarly as with portion strategies (Pimentel et al., 2017), however nonexclusive highlights, for example, those emerging with the Gaussian piece, do not enable the student to generalize well away from the preparation models (Adam-Bourdarios, 2015). The customary choice is to hand structure great element extractors, which requires a considerable measure of building ability and area mastery. This would all be able to be maintained from a strategic distance if great highlights could be adapted consequently utilizing a universally useful learning technique. This is the most favorable position of deep learning.

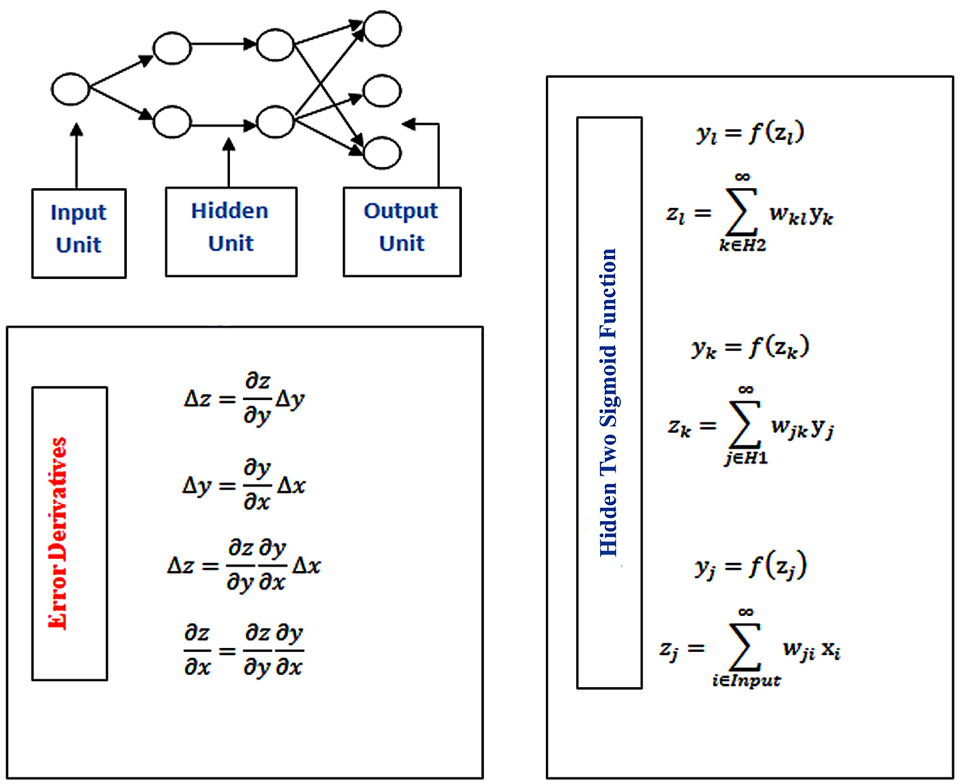
Deep learning engineering is a multilayer pile of straightforward modules, all (or most) of which are likely to learning, and a significant number of which process non-direct information yield mappings. Every module in the stack changes its contribution to increment both the selectivity and the invariance of the portrayal. Different non-direct layers express a profundity of 5 to 20, which is a framework that can actualize very complex functions of its sources of information that are delicate to minute subtleties—recognizing Samoyeds from white wolves—and indifferent towards enormous unimportant varieties, for example, the foundation, posture, lighting, and encompassing articles.

BACKPROPAGATION TO TRAIN MULTILAYER ARCHITECTURES

From the most punctual long periods of example acknowledgment (Krizhevsky, Sutskever & Hinton, Leung, Xiong, Lee, & Frey, 2014), the point of researchers has been to supplant hand-designed highlights with trainable multilayer systems, yet notwithstanding its straightforwardness. The arrangement was not generally comprehended until the mid-1980s. Multilayer models can be prepared by straightforward stochastic inclination plummet. For whatever length of time the modules are moderately smooth elements of their sources of information and of their inside loads, one can register slopes utilizing the back propagation methodology. The possibility this should be possible, and that it worked, was found freely by a few unique gatherings during the 1970s and 1980s (Ma, Liaw, Dahl, Svetnik,Feurer, Springenberg, & Hutter, 2015).

The back propagation strategy to process the slope of a target work concerning the loads of a multilayer heap of modules is just the use of the chain rule for subsidiaries. The key understanding is that the subsidiary (or gradient) of the goal for the contribution of a module can be processed by working in reverse from the inclination for the yield of that module (or the contribution of the resulting module) (Figure 2). The back propagation condition can be connected more than once to engender angles through all modules, beginning from the yield at the top (where the system delivers its expectation) right to the base (where the outer information is sustained). When these angles were processed, it was clear to see the slopes of the loads of every module.

Figure 2. Multilayer neural networks and back propagation



Numerous utilizations of profound learning use feed forward neural network models as seen in Figure- 1, which determine how to delineate a fixed-size contribution (for instance, a picture) to a fixed-size yield (for instance, a probability for every one of a few classes). To begin with one layer, then move onto the next, many units figure a weighted entirety of their contributions from the past layer and pass the outcome through a non-direct capacity. At present, the most well-known non-direct capacity is the amended straight unit (ReLU), which is basically the half-wave rectifier f(z) = max(z, 0). In past decades, neural nets utilized smoother non-linearity's, for example,

tanh(z) or 1/(1 + exp(−z)),

yet the ReLU normally adapts much quicker in systems with numerous layers, permitting preparing of a internally managed organization without unaided pre-preparing (Ma, Sheridan, Liaw, Dahl, & Svetnik, 2015). Units that are not in the information or yield layer are expectedly called shrouded units. The shrouded layers can be viewed as contorting the contribution to a non-direct way with the goal that classifications become straightly distinct by the last layer (Fig. 1).

In the late 1990s, neural nets and back propagation were, to a great extent, spurned by the AI people group and overlooked by the PC vision and discourse acknowledgment networks. It was broadly believed that learning valuable, multistage, extractor-included information not validated by previous research, was not considered feasible. Specifically, most researchers believed that a straightforward inclination plunge would become caught in poor nearby minima—weight setups for which no little change would diminish the normal mistake.

Poor neighborhood minima are occasionally an issue with huge networks. Despite the underlying conditions, the framework almost consistently achieves arrangements of fundamentally the same quality. Ongoing hypothetical and observational outcomes unequivocally recommend that neighborhood minima are not a difficult issue as a rule. Rather, the scene is pressed with a combinatorial huge number of seat focuses where the angle is zero, and the surface bends up in many measurements and bends down in the rest. The investigation appears to demonstrate that seat focuses with just a couple of descending bending headings are available in huge numbers, however practically every one of them has fundamentally the same estimations of the objective capacity. Thus, it does not make much of a difference at which of these seat focuses the calculation stalls out at.

Enthusiasm for profound feed forward systems was restored around 2006 by a gathering of scientists united by the Canadian Institute for Advanced Research (CIFAR). The analysts introduced solo learning methodology that could make layers of highlight finders without requiring marked information. The target in adapting each layer of highlight locators was to have the option to remake or display the exercises of highlight finders (or crude contributions) in the layer beneath. By "pre-preparing" a few layers of logically progressively complex component finders utilizing this remaking objective, the loads of a profound system could be introduced to reasonable qualities. A last layer of yield units could then be added to the highest point of the system and the entire profound framework could be calibrated utilizing standard back propagation. This worked astoundingly well for perceiving manually written digits and for identifying walkers, particularly when the measure of named information was a very limited 36.

The primary real use of this pre-preparing approach was in discourse acknowledgment, and it was made conceivable by the arrival of quick designs handling units (GPUs) that were helpful for programming, and enabled analysts to prepare systems 10 much quicker. In 2009, the methodology was utilized to guide short, fleeting, windows of coefficients removed from a sound wave to many probabilities for the different pieces of discourse that could be spoken to by the edge in the focal point of the window. It accomplished record-breaking results, on a standard discourse acknowledgment benchmark, that utilized a small vocabulary and was immediately created to produce record-breaking results on a huge vocabulary task.

By 2012, variants of the profound net from 2009 were being created by numerous individuals of the real discourse groups and were, at that point, being sent into Android telephones. For littler informational indexes, solo pre-preparing averts over fitting, prompting fundamentally better speculation when the quantity of marked examples is small, or in an exchange setting where bunches of models for some "source" assignments, however not very many for some "target" errands. When profound learning was restored, it worked out that the pre-preparing stage was required for small informational indexes.

There was, notwithstanding, one specific type of profound, feed forward network that was simpler to prepare and which summed up much superior to systems with full networks between nearby layers. This was the convolutional neural system (ConvNet). It accomplished numerous functional victories during the period when neural systems were out of support, and it has as of late been generally embraced by the PC vision network.

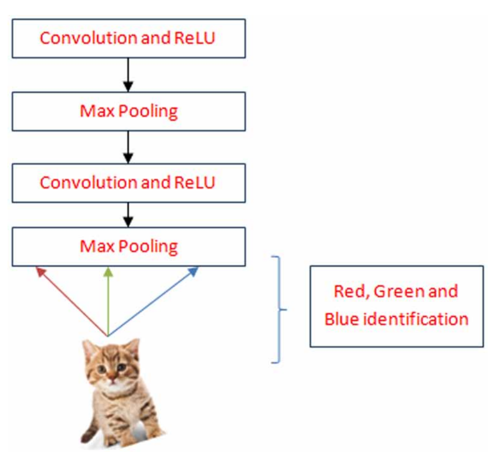
CONVOLUTIONAL NEURAL SYSTEMS

ConvNets are intended to process information that appears as different clusters, for example, a shading picture made out of three 2D exhibits containing pixel powers in the three shading channels. Numerous information modalities are as various exhibits: 1D for sign and groupings, including language; 2D for pictures or sound spectrograms; and 3D for video or volumetric pictures. There are four key thoughts behind ConvNets that exploit the properties of regular sign: nearby associations, shared loads, pooling, and the utilization of numerous layers.

The engineering of a commonplace ConvNet is organized as a progression of stages (Figure 3). The initial couple of stages are made out of two types of layers: convolutional layers and pooling layers. Units in a convolutional layer are composed in highlight maps, inside which every unit is associated with neighborhood fixes in the component maps of the past layer through loads called a "channel bank." The aftereffect of this neighborhood-weighted entirety is then put through a non-linearity, for example, a ReLU. All units in an element guide share a similar channel bank. The different highlight maps in a layer utilize distinctive channel banks. The purpose behind this design is twofold. Initially, have cluster information, for example, photos, whereby neighborhood gatherings of qualities are frequently very corresponded, framing particular nearby themes that are effectively identified. Second, the neighborhood measurements of photos and different signs are invariant to the area. If a theme shows up in one piece of the photo, it could show up in any place, henceforth the possibility of units at various areas having similar loads and distinguishing a similar example in various pieces of the exhibit. Mathematically, the separating activity performed by a component guide is a discrete convolution, henceforth the name.

In spite of the fact that the job of the convolutional layer is to recognize neighborhood conjunctions of highlights from the past layer, the job of the pooling layer is to blend semantically comparative high-

Figure 3. System architecture to handle metadata



lights into one. Since the general places of the highlights shaping a theme can shift fairly dependably, distinguishing the theme should be possible by coarse-graining the position of each element. A run of the mill pooling unit processes the limit of a neighborhood fix of units in a single element map (or in a couple of highlight maps). Neighboring pooling units take contribution from patches that are moved by more than one line or section, in this manner, diminishing the element of the portrayal and making it invariant to small moves and distortions. A few phases of convolution, non-linearity and pooling are stacked, trailed by more convolutional and completely associated layers. Back propagating angles through a ConvNet is as basic as through a normal profound system, permitting every one of the loads in all the channel banks to be prepared.

Profound neural systems abuse the property that numerous common signals are compositional pecking orders; in which higher-level highlights are acquired by forming lower-level ones. In pictures or photos, nearby combinations of edges structure themes, then the themes collect into parts, and the parts structure objects. A comparable pecking order exists in discourse and content from sounds to telephones, phonemes, syllables, words, and sentences (Yogapriya, Saravanabhavan, Asokan, Vennila, Preethi, & Nithya, 2018). The pooling enables portrayals to fluctuate very little when components in the previous layer change in position and appearance. The convolutional and pooling layers in ConvNets are legitimately propelled by the great thoughts of basic cells and complex cells in visual neuroscience, and the general engineering is reminiscent of the LGN–V1–V2–V4–IT pecking order in the visual cortex ventral pathway.

A study by Rajalingham et al. (2018), studied the visual object recognition behavior of humans, monkeys, and state-of-the-art deep artificial neural networks. What was discovered was that when ConvNet models and monkeys were provided with equivalent pictures, the enactments of abnormal state units in the ConvNet clarified half of the difference of irregular arrangements of 160 neurons in the monkey's infer temporal cortex. ConvNets have their foundations in the neocognitron, the design of which was, to some degree, comparative, yet did not have a start to finish administered learning calculation, for example, back propagation. A crude 1D ConvNet called a "period defer neural net" was utilized for the acknowledgment of phonemes and straightforward words.

There have been various uses of convolutional networks returning to the mid-1990s, beginning with time-delay neural systems for discourse acknowledgment (Crankshaw et al., 2017; Ferrucci et al., 2004 and Duda & Hart, 1973), and record perusing (Python, 2011; Farbet et al., 2013; Helmstader et al., 2013 and Hinton et al., 2012). The archive perusing framework utilized a ConvNet prepared mutually with a probabilistic model that actualized language limitations. By the late 1990s, this framework was perusing 10% of the considerable number of checks in the United States. Various ConvNet-based optical character recognition and penmanship acknowledgment frameworks were later sent by Microsoft. ConvNets were also tried on different things in the mid-1990s for article identification in normal pictures, including picture appearances and hands (Miao et al., 2017a, 2017b; Jean, Cho, Memisevic, & Bengio, 2015; Vanschoren et al., 2017; Hellerstein et al., 2017), and for face acknowledgment (Miao et al., 2017a & b).

PROVENANCE OF META DATA WITH DEEP LEARNING

The significance of metadata keeps on developing as associations are coming to understand that to completely abuse the business and operational capability of AI, profound learning, and man-made reasoning, necessitates that the crude information be upgraded with metadata. Keep in mind that while developing volumes of genuine information there is significantly more information, or metadata, around the use and wellspring of the real information.

Metadata is pigeonholed as a slice of information that portrays and provides data about other information. The telephone call outlines the bits of knowledge that can be mined just from the metadata. Research from Stanford University has demonstrated that the metadata of telephone calls uncovers a lot of individual data without getting into genuine voice records. Chart investigations of telephone call metadata can uncover recurrence, recency, quality, and the ideas of connections amongst individuals

To address the problems in extracting, storing, and managing metadata and provenance information of common artifacts helps information researchers, in their everyday assignments, to propose a minor framework for processing the metadata of ML tests through the techniques of machine learning mentioned above, such as supervised learning, back propagation, and convolution neural networks. This framework takes into consideration overseeing the metadata (e.g., who made the model at what time? Which hyper parameters were utilized? What included changes have been connected?) And genealogy (e.g., which dataset was the model determined from? Which dataset was utilized for figuring the assessment information?) of created curios, and gives a passage point to questioning the continued metadata. Information researchers can use this administration to empower an assortment of already difficult-to-accomplish usefulness, for example, ordinary mechanized examinations of models being developed into more established models (like relapse tests for programming). Also, the proposed administration causes information researchers to effectively and specially test their models being developed and provides a beginning stage to evaluating the exactness enhancements that groups accomplish after some time towards an explicit ML objective, e.g., by putting away and examining the assessment aftereffects of their models after some time, and demonstrating them through a leader board.

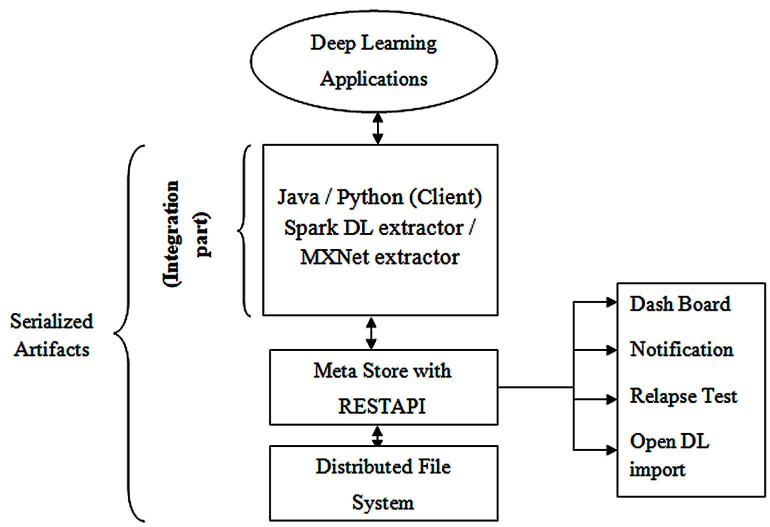
So as to facilitate the appropriation of our metadata following framework, it is necessary to investigate procedures to consequently separate experimentation metadata from normal reflections utilized in ML pipelines, for example, "information casings," which hold de-normalized social information, and ML pipelines, which include an approach to characterize complex element change chains made out of individual administrators. For applications assembled over these reflections, metadata following ought not to require more exertion than uncovering a couple of information structures to our following code. As of now, the foundation of a focal metadata archive as an establishment for further usefulness to be based over. Such a store of recorded experimentation information would pave the way for cutting-edge Meta learning. The proposed framework depicts the hidden database pattern, which permits us to store general, decisive portrayals of ML tests.

SYSTEMS DESIGN TO HANDLE META DATA

The design of the proposed framework is represented in Figure 4. The experimentation metadata is put away in an archive database and our framework backend keeps running in a serverless way on AWS. Customers speak with this backend by means of a REST API, for which it is a must to give autogenerated customers an assortment of dialects. As of now, offer low-level REST-based customers for the JVM and Python, as well as abnormal state customers that are equipped towards famous ML libraries, for example, SparkML (Miao et al., 2017a, 2017b), scikit-learn (Vanschoren et al., 2017), MXNet (Ciodaro et al., 2012), and UIMA. These abnormal state customers give comfort usefulness for the libraries, particularly in the type of computerized metadata extraction from inside information structures. Models incorporate removing the structure of pipelines and DataFrames in SparkML or the representative system definition in MXNet. The metadata and provenance data is devoured by applications, for example, dashboards, leaderboards, email notifiers, and relapse tests against recorded assessment results, and can likewise be questioned intelligently by individual clients. Moreover, this work suggests bringing in openly accessible experimentation information from storehouses, for example, OpenML.

AUTOMATING METADATA EXTRACTION

Figure 4. System architecture to handle metadata



This section discusses metadata extraction usefulness for two unique types of ML outstanding tasks at hand: parameterized "pipelines" in SparkML, and scikit-realize, which work on unthinkable information and neural systems characterized in MXNet by means of emblematic computational diagrams that work on tensor information (Feurer, et al., 2015a, 2015b; Machine learning et al., 2017; Milkov, Deoras, Povey, Burget, & Černocký, 2011).

ML outstanding tasks at hand in SparkML work on data frames, a social deliberation for a divided (and regularly denormalized) table with a well-characterized construction (demonstrated through Spark's StructType reflection) (Polyzotis et al., 2017). The changes connected to the information are demonstrated as phases of a pipeline (via Pipeline Stage deliberation), enlivened by scikit-learn's famous pipeline deliberation. Normally, such a pipeline comprises preprocessing and highlights change activities, trailed by a model preparing stage. Subsequent to being "fitted" to information, a pipeline creates an alleged Pipeline Model (Bai & Preethi, 2016; Parker & Logic, 1985; Preethi & Asokan, 2019), which speaks to a prepared model with a fixed element change technique that can be connected to new information. The design of SparkML pipelines enables one to naturally track all the outline changes (e.g., perusing, including and evacuating sections), which pipeline administrators lead, just as the parameterization of the administrators. The reflection does not enable one to examine the interior execution subtleties (Rosanblatt, 1957; Sainath, Mohamed, Kingsbury, & Ramabhadran, 2013; Saravanan, Asokan, & Venkatachalam, 2013) (e.g., the scientific tasks connected by ML models), as pipeline administrator executions have been planned as secret elements. So, to follow the information changes connected by the pipeline in detail, one would extricate the pattern of the information casing and replay the progressions the pipeline leads by calling the transform Schema strategy for every pipeline stage and then recording the subsequent outline changes. Examination of certain pipeline arrange parameters (displayed by Spark's Params class) like HasInputCol or HasFeaturesCol, will indicate which segments (Schölkopf, Smola, & Bach, 2002) of the data frame an administrator expends as information. This enables one to make a coordinated non-cyclic chart (DAG) portrayal of the information changes connected by the pipeline, where the vertices of the DAG compare to data frame sections and the edges signify pipeline administrators, which devour said segments to create extra sections. The intention is to model these DAGs by means of a chart of transformation elements from our database composition and provide them to clients for visual examination of their information handling pipelines (Selfridge, 1959; Szegedy et al., 2015; Sutskever, Vinyals, & Le, 2014; Tompson, Jain, LeCun, & Bregle, 2014). Moreover, examine what is more, and store the parameterization of every pipeline administrator, including the hyperparameters of models, for example, MAXiter, meaning the most extreme number of cycles to run when learning a relapse model.

Part of the Code for Meta Data Management

```
BEGIN
mod = dx . mod . Module (dlp)
path = try-out
pave_training (mod)
mod . fit (
train_data = trainset
eva_data = validationset
epoch_end_callback =
path. log_train_metric ()
eva_end_callback =
path. log_eval_metric ())
try-out . track_model (mod
name =’MXNet ..., trained_on =
trainset_metadata
training =path, serialize = True)
accuracy = mx . metric . Accuracy ()
mod . score (testset, ’account’)
try-out . track_evaluation
(by_model = model_metadata
on_dataset = testset_metadata
scores = accuracy)
end
```

THE FUTURE OF METADATA MANAGEMENT

Most ventures know that in building up a metadata, the board procedure is vital to remaining aggressive—particularly in a fast moving business sector and many are not sure how or where to start. At the Semantic Web Company (SWC), it is accepted that creation of smart information ought to be a need for any advanced business. While numerous major organizational undertakings have groups of information and examination specialists assisting them to guarantee that they are obtaining judicious information, SWC's Pool Party Semantic Suite furnishes an extensive stage with the majority of the apparatuses and systems required to use the estimation of one's metadata by including a semantic layer top: beginning from structure-up business glossaries, scientific classifications and ontologies, to large-scale enterprise knowledge graphs that are created dependent upon a combined methodology abusing information designing, content mining, and AI.

PoolParty's methodology offers IT investigators several principle favorable circumstances, including an advantage from having steady metadata over the association, including semantic varieties, and multilingualism. One would begin by labeling their information consequently by coordinating the importance of terms and setting found in content and archives against their benchmarks-based enterprise knowledge graph, then finding the learning that their undertaking has by discovering designs in their information that cover up, beforehand, undetected bits of learning. Performing a top to bottom investigation from the organized and unstructured information will provide the investigator with examples and points of premium that will assist them in remaining agreeable with all significant administrative systems and additionally help with the making of well-educated choices.

Any metadata management methodology should be worked around the knowledge that the estimation of metadata is profoundly subject to its programmed interpretability, which should work freely from the applications that utilize the metadata. The PoolParty method can help by utilizing a gauges-based methodology to sort out metadata, information, and substance, in an exceedingly adaptable manner.

REFERENCES

Adam-Bourdarios, C., Cowan, G., Germain, C., Guyon, I., Kégl, B., & Rousseau, D. (2015, August). The Higgs boson machine learning challenge. In *NIPS 2014 Workshop on High-energy Physics and Machine Learning* (pp. 19-55). Academic Press.

Bai, D. P., & Preethi, P. (2016). Security enhancement of health information exchange based on cloud computing system. *International Journal of Scientific and Engineering Research*, *4*(10), 79–82.

Bengio, Y., Delalleau, O., & Le Roux, N. (2005). The curse of highly variable functions for local kernel machines. Proc. Advances in Neural Information Processing Systems, 18, 107–114.

Bordes, A., Chopra, S., & Weston, J. (2014). Question answering with subgraph embeddings. In *Proc. Empirical Methods in Natural Language Processing*. https://arxiv.org/abs/1406.3676v3

Bottou, L., & Bousquet, O. (2007). The tradeoffs of large scale learning. n Proc. *Advances in Neural Information Processing Systems*, *20*, 161–168.

Ciodaro, T., Deva, D., de Seixas, J., & Damazio, D. (2012). Online particle detection with neural networks biased on topological calorimetry Information. *Journal of Physics: Conference Series*, *368*, 012030. doi:10.1088/1742-6596/368/1/012030

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, *12*(Aug), 2493–2537.

Crankshaw, D., Wang, X., Zhou, G., Franklin, M. J., Gonzalez, J. E., & Stoica, I. (2017). Clipper: A low-latency online prediction serving system. In *14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 17)* (pp. 613-627). USENIX.

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis* (Vol. 3). New York: Wiley.

Farabet, C., Couprie, C., Najman, L., & LeCun, Y. (2013). Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1915–1929. doi:10.1109/TPAMI.2012.231

Ferrucci, D., & Lally, A. (2004). UIMA: An architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, *10*(3-4), 327–348. doi:10.1017/S1351324904003523

Feurer, Klein, Eggensperger, Springenberg, Blum, & Hutter. (2015a). Efficient and robust automated machine learning. NIPS, 2962–2970.

Feurer, M., Springenberg, J. T., & Hutter, F. (2015b, February). Initializing bayesian hyperparameter optimization via meta-learning. *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Hellerstein, J. M., Sreekanti, V., Gonzalez, J. E., Dalton, J., Dey, A., Nag, S., & Donsky, M. (2017, January). Ground: A Data Context Service. CIDR.

Helmstaedter, M., Briggman, K. L., Turaga, S. C., Jain, V., Seung, H. S., & Denk, W. (2013). Connectomic reconstruction of the inner plexiform layer in the mouse retina. *Nature*, *500*(7461), 168–174. doi:10.1038/nature12346

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, *29*(6), 82–97. doi:10.1109/MSP.2012.2205597

Jean, S., Cho, K., Memisevic, R., & Bengio, Y. (2015). On using very large target vocabular for neural machine translation. *Proc. ACL-IJCNLP*. https://arxiv.org/abs/1412.2007

Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). ImageNet classification with deep convolutional neural networks. Proc. Advances in Neural Information Processing Systems, 25, 1090–1098.

Kumar, A., McCann, R., Naughton, J., & Patel, J. M. (2016). Model selection management systems: The next frontier of advanced analytics. *SIGMOD Record*, *44*(4), 17–22. doi:10.1145/2935694.2935698

Leung, M. K., Xiong, H. Y., Lee, L. J., & Frey, B. J. (2014). Deep learning of the tissue regulated splicing code. *Bioinformatics (Oxford, England)*, *30*(12), 121–129. doi:10.1093/bioinformatics/btu277

Ma, J., Sheridan, R. P., Liaw, A., Dahl, G. E., & Svetnik, V. (2015). Deep neural nets as a method for quantitative structure-activity relationships. *Journal of Chemical Information and Modeling*, *55*(2), 263–274. doi:10.1021/ci500747n

Miao, H., Li, A., Davis, L. S., & Deshpande, A. (2017a April). Towards unified data and lifecycle management for deep learning. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE* (pp. 571-582). IEEE. 10.1109/ICDE.2017.112

Miao, H., Li, A., Davis, L. S., & Deshpande, A. (2017b, May). On model discovery for hosted data science projects. In *Proceedings of the 1st Workshop on Data Management for End-to-End Machine Learning* (pp. 1-4). 10.1145/3076246.3076252

Mikolov, T., Deoras, A., Povey, D., Burget, L., & Černocký, J. (2011, December). Strategies for training large scale neural network language models. In *2011 IEEE Workshop on Automatic Speech Recognition & Understanding* (pp. 196-201). IEEE. 10.1109/ASRU.2011.6163930

Parker, D. B., & Logic, L. (1985). *Technical Report TR-47.* Massachusetts Institute of Technology.

Pimentel, J. F., Murta, L., Braganholo, V., & Freire, J. (2017). noWorkflow: A tool for collecting, analyzing, and managing provenance from python scripts. *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, *10*(12), 1841–1844. doi:10.14778/3137765.3137789

Polyzotis, N., Roy, S., Whang, S. E., & Zinkevich, M. (2017, May). Data management challenges in production machine learning. In *Proceedings of the 2017 ACM International Conference on Management of Data* (pp. 1723-1726). 10.1145/3035918.3054782

Preethi, P., & Asokan, R. (2019). A high secure medical mage storing and sharing in cloud environment using hex code cryptography method—Secure genius. *Journal of Medical Imaging and Health Informatics*, *9*(7), 1337–11345. doi:10.1166/jmihi.2019.2757

Python, P. (2011). Scikit-learn: Machine Learning. *JMLR*, *12*, 2825–2830.

Rajalingham, R., Issa, E. B., Bashivan, P., Kar, K., Schmidt, K., & DiCarlo, J. J. (2018). Large-scale, high-resolution comparison of the core visual object recognition behavior of humans, monkeys, and state-of-the-art deep artificial neural networks. *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*, *38*(33), 7255–7269. doi:10.1523/JNEUROSCI.0388-18.2018

Rosenblatt, F. (1957). *The Perceptron — A Perceiving and Recognizing Automaton*. Tech. Rep. 85 460-1. Cornell Aeronautical Laboratory.

Sainath, T., Mohamed, A. R., Kingsbury, B., & Ramabhadran, B. (2013, May). *Deep convolutional neural networks for LVCSR. In 2013 IEEE international conference on acoustics, speech and signal processing* (pp. 8614–8618). IEEE. doi:10.1109/ICASSP.2013.6639347

Saravanan, K., Asokan, R., & Venkatachalam, K. (Nov 2013). Neuro-Fuzzy Based Clustering Of Distributed Denial of Service (DDoS) Attack Detection Mechanism. *International Information Institute (Tokyo). Information, 16*(11), 8137.

Schölkopf, B., Smola, A. J., & Bach, F. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT Press.

Selfridge, O. G. (1959). Pandemonium: A paradigm of learning. Mechanisation of Thought Processes. *NPL Symposium 10*.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Advances in neural information processing systems (pp. 3104–3112). Academic Press.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1-9). IEEE.

Tompson, J. J., Jain, A., LeCun, Y., & Bregler, C. (2014). Joint training of a convolutional network and a graphical model for human pose estimation. In Advances in neural information processing systems (pp. 1799-1807). Academic Press.

Vanschoren, J., Van Rijn, J. N., Bischl, B., & Torgo, L. (2017). OpenML: Networked science in machine learning. *SIGKDD Explorations*, *15*(2), 49–60. doi:10.1145/2641190.2641198

Yogapriya, J., Saravanabhavan, C., Asokan, R., Vennila, I., Preethi, P., & Nithya, B. (2018). A study of image retrieval system based on feature extraction, selection, classification and similarity measurements. *Journal of Medical Imaging and Health Informatics*, *8*(3), 479–484. doi:10.1166/jmihi.2018.2326

Chapter 8
A Fully Automated Crop Disease Monitoring and Management System Based on IoT:
IoT-Based Disease Identification for Banana Leaf

K. Seetharaman
https://orcid.org/0000-0002-5863-6883
Annamalai University, India

ABSTRACT

In recent years, the IoT has evolved and plays a significant role in many fields like smart city, precision farm, traffic signal control system, and so on. In this chapter, an IoT-based crop disease management (CDM) system is proposed that adopts statistical methods for identifying disease, recognizing a right pesticide, and recommending a right pesticide to farmers. The proposed CDM system monitors the agricultural crops with the help of a CCD camera. The camera continuously photographs the crops and sends them to a Raspberry PI processor, which is placed at a workstation and it is connected to the camera with the help of IoT components. The proposed CDM system analyses the crop leaf images, such as removes noise; segments region of interest (RoI), that is, diseased part of the leaf image; extracts features from the RoI; and identifies the disease and takes appropriate measures to control the disease. The proposed IoT-based CDM system was experimented, and the results obtained encourage both the farmers and the researchers in this field.

INTRODUCTION

The agriculture sector plays a noteworthy role in the development of the country's economy as such, it provides large-scale employment opportunities to people in countries like India, Colombia, USA, China, Brazil, etc. So, the growth of the agricultural sector is necessary for the development of the economic

DOI: 10.4018/978-1-7998-2108-3.ch008

condition of a country. The UN Food and Agriculture Organization has reported that the world has to produce 70 percent more food in 2050 than it produced in 2006 to meet out the food demands of the growing population across the world. With the advent of the advanced technologies in both the agricultural industry and, information and communication technology, human society can produce enough food to meet the demand of more than 7 billion people. However, there exists a threaten to food security by a number of factors including climate change (Harvy et al., 2014), the decline in pollinators (Report of the Plenary of the Inter-governmental Science-Policy Platform on Biodiversity Ecosystem and Services on the work of its fourth session, 2016), plant diseases (Sanchez & Swaminathan, 2005), and some other factors. Plant diseases are not only a threat to food security at the global level but also cause disastrous for smallholder farmers whose livelihoods rely on healthy crops. In the developing countries like India, more than 80 percent of the agricultural production is generated by smallholder farmers (Strange & Scott, 2005), and a report suggests that there is a loss of more than 50% in yield due to pests and diseases (Tai et al., 2014). Moreover, a study reveals that around 50% of the people live in smallholder farming households making smallholder farmers as a group, that is, particularly vulnerable to pathogen-derived disruptions in food supply (UNEP, 2013). So, this is the right time to improve productivity and enhance the quality of the agricultural products, unless otherwise, the people will face food scarcity in the future across the world. But, unfortunately, the farmers face a lot of difficulties in day-to-day crop monitoring and management system like more use of pesticide and fertilizer, disturbance of insects, some types of plant diseases, food preservation, rainfall water scarcity, and so on. Particularly, nowadays, varieties of diseases attack crops so that it creates a huge amount of loss in yield and the farmers have to spend more money on curing the attack. Moreover, many farmers still use the traditional method of farming, which results in low yield; it could also be one of the main factors for the low yield.

To assist the farmers, that is, minimize the expenses and maximize the yield, the precision agriculture (PA) was formed in1929 in Illinois, USA. The PA adopts a management strategy that uses Information and Communication Technology (ICT) to bring data from multiple sources to make a decision associated with the increase of crop production. The PA has been developed not only in countries like the USA, European Union, and Australia where the average farm sizes are relatively large, but also in Japan, China, and Korea where average farm sizes are relatively small. Since the mid-1990s the PA concept has been widely used in research by academic groups and the agricultural companies by adopting ICT with the use of commercially available sensors, wireless sensors networks, controllers, software programs, and so on. The Scope of the PA has become wide and it has introduced new terms, such as precision citrus farming, precision horticulture, precision viticulture, precision livestock farming, and precision aquaculture, etc. For effective use of these new aspects of PA, recent technological advancements in data communication and data analysis, such as ICT, IoT, Pattern Recognition, Big Data Analytics, and Data Mining techniques, have been fused.

To overcome this problem, the farmers and the agro-chemical industries can adopt the futuristic of the Internet of Things (IoT) for analytics and greater production capabilities. Over a decade, there has been a swift development of the wireless sensor networks and new smart devices that can connect to the Internet and be controlled using applications remotely. This network of devices and other items embedded with sensors, electronics, software, and connectivity is called the Internet of Things. In recent years, the IoT has begun to play a noteworthy role in day-to-day activities by extending our concepts and ability to modify the environment around us. Particularly, the agro-chemical industrial and environmental domains apply the IoT in both disease diagnostics and control. Besides, it can provide information to the end-user or consumer about the origin and attributes of the product. Thus, this chapter focuses on applying IoT for computer-aided optimization of the agriculture crop management system.

In such optimization of smart farming, that is, adoption of ICT and IoT and other advanced technologies like Big Data Analytics, Data Mining techniques, Pattern matching, and Image processing methods in the smart farming could encourage the farmers and improve the yield (Capello et al., 2016; Fang et al., 2014; Hashim et al., 2015; Kodali et al., 2014). In smart farming, the prediction or characterizing the field variables such as soil and weather conditions, growth and healthiness of the crop, time of applying the fertilizer and pesticides, and biomass of plants, can be identified easily in time by the farmers. It can also be used to assess and control variables such as temperature, humidity, vibrations, or shocks during product transport Pang et al., 2015). Moreover, the IoT can be applied to monitor and control the factors that affect crop growth and yield. They can also be employed to determine the optimum time to harvest, and which farm is more suitable for what conditions, detect diseases, control machinery, etc. (Ndzi et al., 2014). Muangprathuba et al. (2019) have studied the characterization of the temperature, humidity, and soil moisture in the crop fields.

The diseases related to the agricultural leaf of the plant such as blight, leaf curl, leaf spot and, the Canker and Gall appear in root and stem of the plant; and some other diseases like Powdery Mildew, Root Rot, Wilt, Stunting, and Chlorosis, etc. damage the plant. Moreover, these diseases cause major production and economic losses as well as affect both the quality and quantity of agricultural crop production. The review of the literature reveals that the loss caused by the plant diseases accounts for at least a 10% reduction in global food production. It is better to identify leaf diseases at the early stage on leaf health, and disease detection can facilitate the farmers to control the diseases through proper management strategies. This chapter focuses on characterizing the diseases and taking an appropriate measure to control them. This chapter demonstrates the applications of the IoT in the identification of different diseases and controlling measures of it on Banana trees. In addition to that this chapter constructs an individual database, such as the Banana trees database, disease database, pesticide database. The crop database consists of types of banana crops and their characteristics; the disease database consists of information about the varieties of diseases and their causes of impacts; the pesticide database contains information about the quantity of pesticide that has to be applied.

To develop a proper IoT-based smart farming system, we require a data repository and an approach to discover knowledge from accumulated data and interactions with the users. A database system is designed and implemented with the IoT-based applications. The stored data are used for decision making and control of the automatic disease monitoring and management system. The data related to disease gathered through IoT are analyzed to take corrective measures and control the pest, and to predict the measure which protects the crop from the insects in the future. The key contribution of this chapter is agricultural crop leaf disease identification and recommendation of the right pesticide to farmers with the help of IoT components.

The high-speed internet, mobile devices, and low-cost satellites (for imagery and positioning) are the key technologies that characterize the precision agriculture trend and that could be adopted by the manufacturer to improve and accelerate the agriculture growth according to the requirement of the agricultural product. For example, a smart GPS-based robot performs weeding, spraying, moisture sensing, bird and animal scaring, keeping vigilance. It also performs the automated smart irrigation with smart control and intelligent decision-making based on right real-time field data. Besides it monitors smart warehouse management, like temperature and humidity maintenance and theft detection, etc. The review of the literature reveals that there is some lacuna in the system and they are not comprehensively fulfilling the farmers' requirements. Nevertheless, this chapter concentrates on the crop disease management system through IoT. Because the crop disease is the main factor, which decreases the yield or even it causes decease to the crops.

BACKGROUND

A new Integrated Information System (IIS) with a combination of IoTs, such as Cloud Computing, Geo-informatics, and e-Science for environmental monitoring and management, has been introduced for a case study on regional climate change and its ecological responses (Fang et al., 2014). Fourati et al. (2014) have introduced a web-based decision support system with the help of wireless sensor networks for irrigation scheduling in olive fields. It uses sensors to measure humidity, solar radiation, temperature, and rain. Hashim et al. (2015) have reviewed the control with an electronic device, such as Arduino of temperature and soil moisture, and used Android-based smartphone applications for flexibility and functionality. They found advantages in low cost and flexibility for agriculture control in contrast with expensive components, such as high-end personal computers. Li et al. (2014) have presented an information system for agriculture, based on IoTs, with a distributed architecture. They have applied distributed IoTs servers for tracking and tracing the whole agricultural production process. Also, they have developed an information-discovery system and applied to implement, capture, standardize, manage, locate, and query business data from agricultural production. Xian (2017) has introduced a new online monitoring system for IoTs, based on cloud computing concepts. Mattihalli et al. (2018) have proposed a system, which identifies the disease in plant leaves. They have developed the system based on IoT, and have applied image preprocessing methods. To identify and differentiate diseases, they have applied the Bhattacharyya distance measure with histogram techniques. Jagtap & Hambarde (2014) have developed a method, based on morphological feature extraction techniques, which identifies leaf diseases; the region of interest is segmented using a fuzzy c-means algorithm, and a back-propagation method is employed for classification of diseases. Stewart & McDonald (2014) have introduced an automated image analysis method, which analyses disease symptoms of infected wheat leaves caused by *Zymoseptoria tritici*. This method enabled the quantification of the size and density of the pycnidia along with other traits and their correlation. The method provides good accuracy and precision compared to human visual estimates of virulence. Wang et al. (2017) have proposed a method, based on a deep convolutional neural network, which is trained to diagnose the severity of the disease and control it. Johannes (2017) has introduced an image processing method, based on candidate hot-spot detection techniques with a combination of statistical inference, which identifies diseases in wild conditions. This method analyses the performance of early identification of three types of European endemic wheat leaf diseases: septoria, rust, and tan spot. Lu et al. (2017) have trained a deep convolutional neural network (CNN) with a dataset of 500 natural images of diseased and healthy paddy crop leaves and stems, which identifies 10 common paddy crop leaf diseases. A deep convolutional neural network (DCNN) has been presented in (Ma et al., 2018) which detects four different kinds of leaf spot diseases such as anthracnose, downy mildew, powdery mildew, and target leaf spots of the cucumber plant symptom-wise. Ferentinos (2018) has conducted an empirical study on plant disease detection and diagnosis using deep learning convolutional neural network method and he has reported that the proposed method achieved a success rate of 99.53% (top-1 error of 0.47%) in the classification of 17,548 previously unseen by the model plant leaves images (testing set). Qin et al. (2016) have proposed a method to identify and diagnose the leaf spot diseases affect alfalfa leaf. It recognizes four kinds of leaf spot diseases: *common leaf spot, rust, Leptosphaerulina leaf spot,* and *Cercospora leaf spot*. The leaf spot diseases are recognized based on pattern recognition methods, such as feature extraction, clustering, classification, and segmentation. They have employed twelve lesion segmentation methods, which integrates the algorithms of both supervised and unsupervised. After a comprehensive comparative study, they have integrated the *K*-median clustering algorithm and LDA technique and employed to obtain the segmented sub-images. Dos Santos et al. (2019) have introduced a model, called AgriPrediction, which joins together the short and medium-range wireless network with a prediction engine to proactive potential crop dysfunctions to notify the farmer for remedial actions as soon as possible. To achieve this, the AgriPrediction model presents a framework, based on both LoRa IoT technology and the ARIMA model.

In this chapter, based on the review of literature, it is believed that the statistical methods could provide good results for agricultural crop leaf disease detection and diagnoses. Thus, this chapter motivates to apply the statistical methods such as *modified Wishart* and the *Hotelling's* T^2 *test statistics* for the CDM system, which deals with leaf disease detection and diagnosis.

Outline of the Proposed CDM System

This chapter proposes a *fully automated crop disease management* (CDM) system, based on a *futuristic IoT technology with statistical pattern recognition methods*. The pattern recognition method comprises the *modified Wishart* and *Hotelling's* T^2 *test statistics*. First, the proposed method preprocesses, such as noise removal and segmentation of the region of interest (disease-affected region) of the given input leaf image. The noise is removed using a *weighted median filter*. Now, the modified Wishart and Hotelling's T^2 tests, compare the covariance matrices and the mean vectors of the two groups respectively. The proposed CDM system facilitates the farmers to photograph the crops through an IP camera, which is connected to the remotely located computers or mobiles. The captured crop image is automatically submitted to the CDM system, which *analyses the image* and provides information like *Name of the diseases, stages of the diseases, causes of the diseases, and recommends more apt pesticides, fungicides, and fertilizer with proportionate quantity*. The key feature of the proposed CDM system is that it comprises an inbuilt database, which contains several crop images, crop prone diseases, types of measures needed to be taken, types of pesticides, types of fungicides; and the quantity of the pesticide or fungicide to be applied. The proposed CDM system, also, facilitates the farmers to ascertain the information about the crops, and educate the farmers in all aspects of sitting at one place in one touch.

IoT COMPONENTS

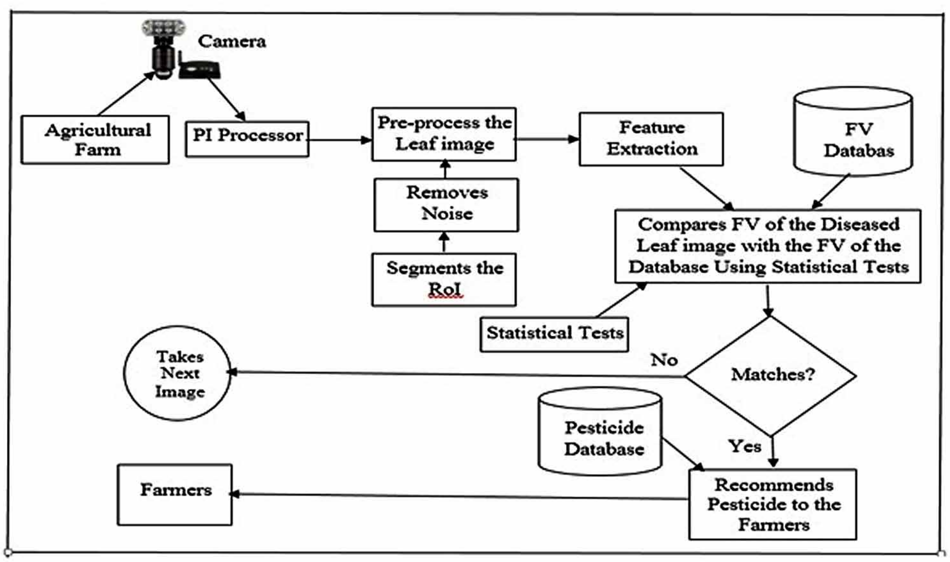
The research community of the IoT, academia, IoT device manufacturers, and the beneficiaries or the end-users of the IoT strongly feel that the IoT will make a revolution in ICT and society, and also it will emerge as a rival that of the Internet itself (Dos Santos et al., 2019; Henschke, 2017; Levy, 2016). The size and wide-range use of the IoT is expected to be very large in near future, i.e. by 2020, and 20 to 50 billion things could be connected as part or components of the IoT (Scroxton, 2016). The IoT facilitates the objects to equip with sensing, recognizing, networking and processing capabilities that allow them to communicate with each other and other devices, and provide services over the Internet to achieve some useful objectives (Mohammed, 2016). Though, still, there is no proper definition for IoT, which can be referred to as a complicated network of interactive and technical components that clustered around three key elements such as sensors Whitemore et al., 2015), informational processors (Mohammed, 2016; Li et al., 2015), and actuators (Agarwal & Dey, 2016). Mohammed (2016) suggests that the IoT is ultimately not about the *things*, it is about the *services*. The analysts have determined that in future the IoT industry will shift their focus on providing good quality of services with quantifiable impact instead of simply adding device quantities. Thus, this chapter focuses on services (pattern recognition and decision making) than IoT devices. Whilst, the proposed IoT-based CDM system utilizes the IoT devices – CCD camera, Raspberry PI processor, Wireless sensor networks and other related accessories – that are discussed below.

A camera is permanently fixed at a farm, which monitors and captures the soil and weather conditions, growth and healthiness of the crop, insects, and diseases affect the crops, time of applying fertilizer and pesticides, the biomass of the plants, etc. The camera sends the captured image to a computer with a Raspberry PI processor through wireless sensor networks. The PI processor is fixed at the workstation of precision farming and decision making. This chapter concentrates only on disease monitoring and control.

Block Diagram of the Proposed System

An outline of the overall functions of the proposed CDM system has been presented in Figure 1.

Figure 1. Outline of the Proposed CDM System



FILTERING

There are possibilities of inclusion of noise in the crop leaf images so that the proposed disease monitoring system performs preprocesses such as noise removal and segmentation. The preprocessing methods are discussed in the next section.

Noise Removal

Inclusion of noise could lead to a wrong result in the segmentation of the RoI and it may affect the decision making on disease identification. So that it is better to remove the noises before performing the disease identification process. To remove the noise, the *weighted median high-pass filter* is applied. The weighted median filter is briefed for a better understanding of the functions of the system as follows.

Figure 2. Weight factor and Subimage.

$$W_{ij} = \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 3 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array} \qquad I_{ij} = \begin{array}{|c|c|c|}\hline 138 & 167 & 114 \\ \hline 118 & 135 & 113 \\ \hline 116 & 126 & 123 \\ \hline \end{array}$$

Weighted Median Filter assigns weights to the filter position as in the mask below. Insert each pixel within the filter region, W_{ij} times into extended pixel vector, EI_{ij}; and the extended vector is sorted in ascending order.

$$EI_{ij} = \left[138, 167, 167, 114, 118, 118, 135, 135, 135, 113, 113, 116, 126, 126, 123\right]$$

The above vector is sorted in ascending order, and the mid-value (the 8-th element highlighted with bold-face) is chosen as the median value.

$$EI_{ij} = \left[113, 113, 114, 116, 118, 118, 123, 126, 126, 135, 135, 135, 138, 167, 167\right]$$

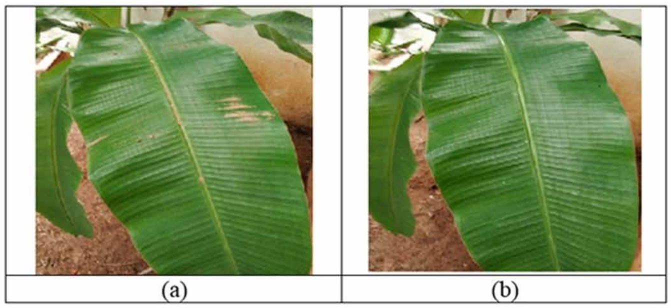
Figure 3. Median Filter and High-pass Filter.

	126	

138	167	126
126	135	126
26	126	126

For a sample, the outcome of the weighted median filter is presented in Figures 2 and 3; the experimental result is shown in Figure 4.

Figure 4. Image Filter: (a) Noised image; (b) Noise removed image.



Disease Segmentation

To identify the disease and its severity (level), the features should be extracted from the RoI (diseased part of the leaf). To extract the features from the RoI, the diseased region is segmented using the algorithm proposed in (Stankovic, 2014). The segmented regions, for a sample, have been presented in Figure 5.

FEATURE EXTRACTION

Color Features

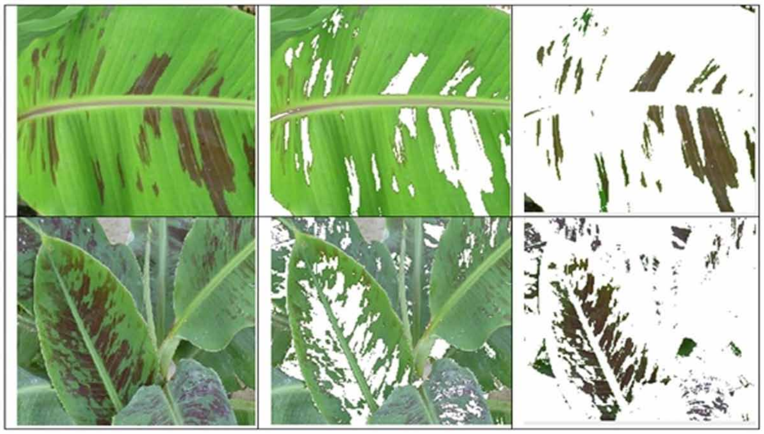
Since the color plays a significant role in agricultural crop and its leaf images, it is considered as one of the main features for agricultural crop leaf disease identification. This chapter takes into account of RGB color for disease recognition. For this purpose, the R, G, and B colors are segmented individually and considered for analysis purposes.

Texture Feature

The surface of the agricultural crop leaf image is smooth and fine. The smooth surface of the leaf changes to a rough surface, while the disease grows towards the advanced stage. Therefore, the texture features play a significant role in differentiating the diseased part from the normal leaf image and identifying the stages of the disease. The autocorrelation function (ρ) plays a noteworthy role in characterizing the texture (Stankovic, 2014). Thus, this chapter takes into account autocorrelation for characterizing the agricultural crop leaf image in terms of texture properties.

The degree of aggressiveness of the crop leaf diseases is measured based on the surface of the diseased part of the leaf. The surface of the diseased part is characterized by the first-order autocorrelation. To compute the relationship, the RoI is subjected to the experiment with sliding windows of size 3×3. The autocorrelation function expressed in Eq. (2) is employed to compute the autocorrelation between

Figure 5. Column 1: Actual diseased images; column 2: segmented background region; column 3: segmented diseased region of interest.



the center pixel of the sliding window and its neighboring pixels. The expression in Eq. (1) is rewritten as in Eq. (2) for a computational purpose. The neighboring pixels to the center pixel of the window are formed as a one-dimensional vector for computational simplicity.

$$\rho_k = \frac{E\left(f_t - \mu\right)E\left(f_{t+k} - \mu\right)}{\sigma^2} \tag{1}$$

where, f_t represents the pixel intensity value; μ represents mean intensity value; σ^2 represents variation among the pixel intensity values; *k* is the *lag* variable, which denotes the order of the autocorrelation. In this case, the order is one.

$$\rho_k = \frac{E\left(f_c - \mu\right)E\left(f_k - \mu\right)}{\sigma^2} \tag{2}$$

In Eq. (2), f_c and f_k represent the center and its neighboring pixels respectively.

The autocorrelation values are treated as one of the features, which effectively characterizes the surface of the diseased part of the leaf images, in terms of texture properties. The surface properties play an important role in identifying the stages of the diseases.

Furthermore, the statistical features, such as Coefficient of Variation (CoV), Skewness (Skw), and Kurtosis (Kur), are extracted that have been expressed in Eqns. (3), (4), and (5). The extracted features are indexed and formed as a feature vector as depicted in Eq. (6), and they are combined and formed as an FVs database as discussed in (Seetharaman & Palanivel, 2013). The extracted features are classified into various groups according to their nature using the *fuzzy c-means* algorithm. For each group, the median value is calculated; using the median value, the FVs are indexed. Based on the classes of the FVs, images in the database are classified into different groups, and it establishes a link between images and their corresponding FVs of each class. Now, the extracted FVs of the diseased image are compared

Figure 6. Structure of the feature vectors and their corresponding images

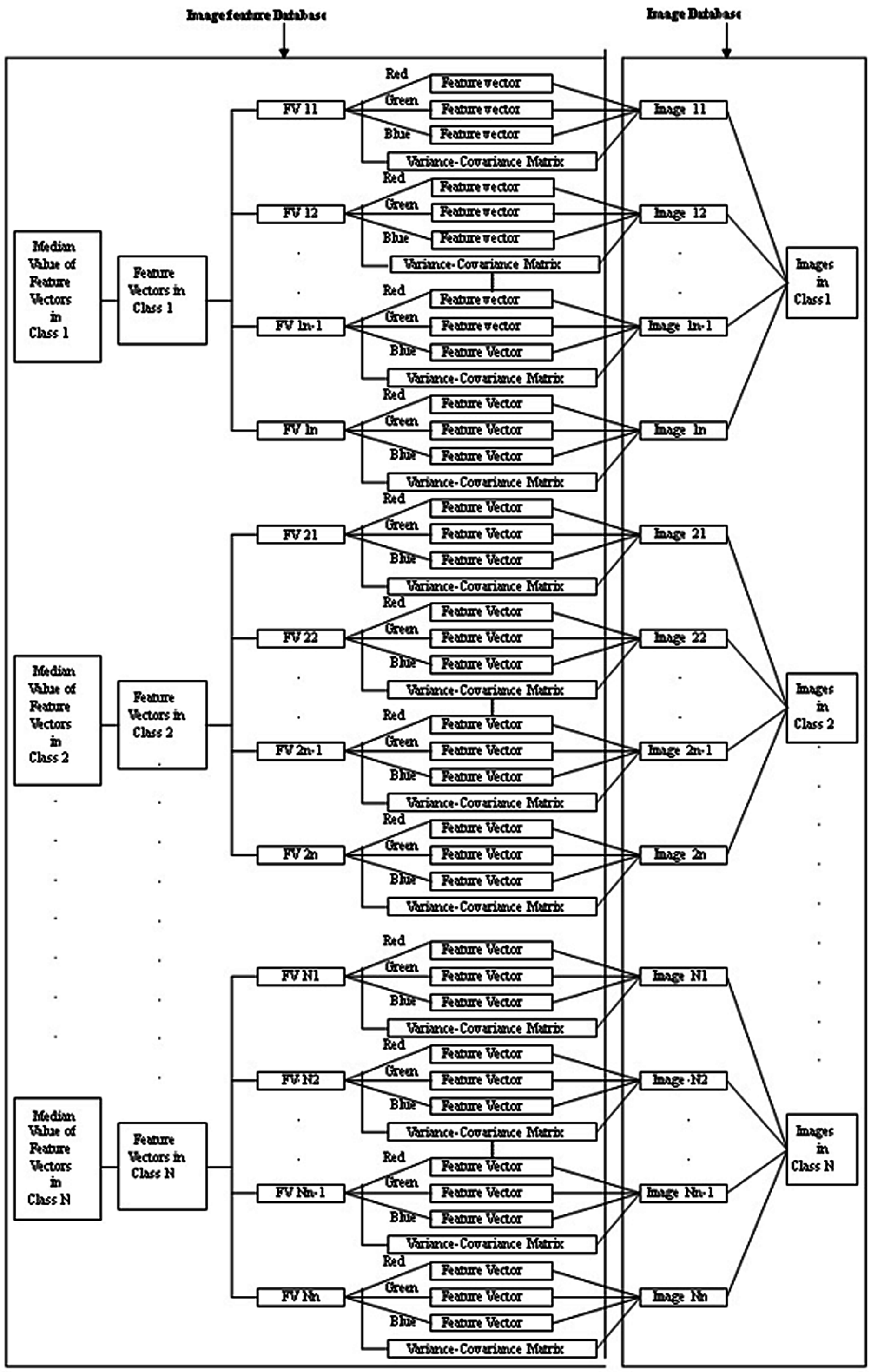
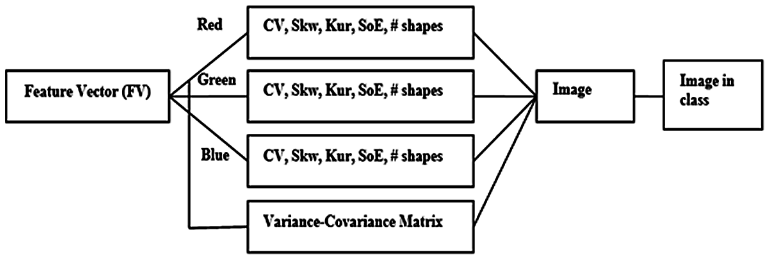


Figure 7. Features of the Feature Vector



to those of the index of the FVs in the image feature database, based on the modified Wishart distance measure (Seetharaman & jeyakarthic, 2014) and Hotelling's T^2 test statistic. Then the identified image and its FVs are grouped to the corresponding class. If the FV of the query leaf image does not match with any classes of the FV database, then it is formed as a new FV class. The extracted color and texture features are formed as a bag-of-features (BoF) as depicted in Figure 6 and 7.

$$Coefficient\,of\,Variation = \frac{\sigma}{\mu} \tag{3}$$

where, σ and μ represent standard deviation and spectrum of intensity values of the pixels.

$$Skewness = \sum_{i=1}^{N} \frac{(x_i - \overline{x})^3}{(N-1)s^3} \tag{4}$$

$$Kurtosis = \sum_{i=1}^{N} \frac{(x_i - \overline{x})^4}{(N-1)s^4} \tag{5}$$

where, x_i represents intensity value of the i-th pixel; $\overline{x}$ represents the mean intensity value; N is the number of pixels in the image; 's' means the standard deviation of the intensity values.

FV = (ACC, CoV, Skw, Kur) (6)

DISEASE IDENTIFICATION AND PESTICIDE MATCHING

Basis of the Proposed Method

Let *f* be a random process that represents intensity value with additive noise (ε) of a pixel at location (*k*, *l*) in an image. The term ε is a random process. In a color image, the pixel, $f(k,l) \in \Re^3$, is a linear combination of red, green, and blue colors. The mean intensity value of each color is represented by notations M_r, M_g and M_b; and the covariance matrix is denoted by Σ, which is a symmetric and positive definite.

The i-th diagonal element σ_{ii} of the covariance matrix Σ is the variance of the i-th component of $f(k,l)$. The mean vector of each color of the pixel in an image is

$$\mathrm{M} = \mathrm{E}\left(F\right) = \mathrm{E}\begin{bmatrix} M_r \\ M_g \\ M_b \end{bmatrix} = \begin{bmatrix} m_r \\ m_g \\ m_b \end{bmatrix} \tag{7}$$

and the covariance matrix is

$$\Sigma = \begin{bmatrix} \sigma_{rr} & \sigma_{rg}\rho & \sigma_{rb}\rho \\ \sigma_{gr}\rho & \sigma_{gg} & \sigma_{gb}\rho \\ \sigma_{br}\rho & \sigma_{bg}\rho & \sigma_{bb}\rho \end{bmatrix} = \begin{bmatrix} \sigma_r^2 & \sigma_{rg}\rho & \sigma_{rb}\rho \\ \sigma_{gr}\rho & \sigma_g^2 & \sigma_{gb}\rho \\ \sigma_{br}\rho & \sigma_{bg}\rho & \sigma_b^2 \end{bmatrix} \tag{8}$$

where, $\sigma_r^2, \sigma_g^2,$ and σ_b^2 are the variations among the intensity values of red, green, and blue colors; σ_{rg} represents the interaction between the red and green colors; likewise, σ_{rb} and σ_{gb} represent the interaction between the corresponding colors; ρ represents interrelationship between the corresponding color pixels.

Test for Interaction

In order to recognize the diseased crop leaf, the R, G, and B colors of the segmented region of the diseased part are segregated and that are compared with the color feature vectors of the feature vector database using the modified non-central Wishart distance ($MWD_{non\text{-}central}$) measure which tests whether the covariance of the input diseased crop leaf image and the color feature vectors in the feature vector database are same or not. The Wishart distance measure is expressed as follows.

$$MWD_{non\text{-}central} = 2\ln\left|\bar{\Sigma}_w\right| - \ln\left|\bar{\Sigma}_X\right| - \ln\left|\bar{\Sigma}_Y\right| \tag{9}$$

where,

$$\bar{\Sigma}_w = \frac{1}{N_q + N_t}\left[\sum_{i=1}^{L_q}\left(f_i^q - m\right)\left(f_i^q - m\right)^H + \sum_{i=1}^{L_t}\left(f_i^t - m\right)\left(f_i^t - m\right)^H\right]$$

$$\bar{\Sigma}_x = \frac{1}{N_q}\left[\sum_{i=1}^{L_q}\left(f_i^q - m^q\right)\left(f_i^q - m^q\right)^H\right]$$

and

$$\bar{\Sigma}_t = \frac{1}{N_t}\left[\sum_{i=1}^{L_t}\left(f_i^t - m^t\right)\left(f_i^t - m^t\right)^H\right]$$

where, N_q represents number of pixels in the segmented region of the diseased leaf part, N_t represents the number of pixels in the diseased leaf part which is to be tested with query diseased leaf image; m^q and m^t represent the mean intensity value of the query diseased leaf image and the targeted FV in the FV database, respectively. The 'H' denotes the Hermitiance matrix.

- ***Critical region:*** Since the $NCMWD_{non\text{-}central}$ statistic is asymptotically distributed to Chi-square (χ^2) distribution with degrees of freedom d. The $NCMWD_{non\text{-}central}$ is compared to the critical value, χ_α^2, with the statistical table, where ***α*** is the level of significance. If $NCMWD_{non\text{-}central} \le \chi_d^2\left(1-\alpha\right)$ at the level of significance ***α***, then it is inferred that the query diseased leaf image and the targeted FV of the FV database of the disease are same or similar (i.e., belong to the same class); otherwise, it is assumed that the query diseased leaf image does not match with the FVs of the disease database.

Test for Equality of Mean Vectors

To confirm the disease, furthermore, the diseased image is compared to the FVs of the disease FV database in terms of the mean value. To achieve this, the Hotelling's T^2 statistic is employed, which tests whether the mean vectors of the query diseased leaf image and the targeted disease of the disease FV database. The Hotelling's T^2 test statistic is briefed as follows.

To test the hypothesis that $m^{(q)} = m^{(t)}$, a special vector, we consider the squared statistical distance, $\overline{x}^q - \overline{x}^t$.

Now,

$$E(\overline{x}^q - \overline{x}^t) = E(\overline{x}^q) - E(\overline{x}^t) = \mu^{(q)} - \mu^{(t)} \tag{10}$$

Since the $\overline{x}^q$ and the $\overline{x}^t$ are independent and thus $Cov(\overline{x}^q - \overline{x}^t) = 0$, it follows that

$$Cov(\overline{x}^q - \overline{x}^t) = Cov(\overline{x}^q) + Cov(\overline{x}^t) = \frac{1}{n_1}\Sigma + \frac{1}{n_2}\Sigma = \left(\frac{1}{n_1} + \frac{1}{n_2}\right)\Sigma \tag{11}$$

Because S_{pooled} estimates Σ, we see that $\left(\frac{1}{n_1} + \frac{1}{n_2}\right)S_{pooled}$ is an estimate of the $Cov(\overline{x}^q - \overline{x}^t)$.

The likelihood ratio test of $H_0 : \mu^{(q)} = \mu^{(t)}$ is based on the square of the statistical distance, T^2 is

$$T^2 = \left(\overline{x}^q - \overline{x}^t - \delta_o\right)' \left[\left(\frac{1}{n1} + \frac{1}{n2}\right) S_{pooled}\right]^{-1} \left(\overline{x}^q - \overline{x}^t - \delta_o\right) > c^2 \tag{12}$$

where the critical distance, c^2, is determined from the distribution of the two-sample Hotelling's T^2-statistic.

$$c^2 = \frac{(n_q + n_t + 2)p}{(n_q + n_q - p - 1)} F_{p, n_q + n_t - p - 1}(\alpha) \tag{13}$$

$$S_{pooled} = \frac{\sum_{j=1}^{n_q} \left(\overline{x}_j^q - \overline{x}_j^t\right)\left(\overline{x}_j^q - \overline{x}_j^t\right)' + \sum_{j=1}^{n_t} \left(\overline{x}_j^q - \overline{x}_j^t\right)\left(\overline{x}_j^q - \overline{x}_j^t\right)'}{n_q + n_t - 2} \tag{14}$$

$$S_{pooled} = \frac{n_q - 1}{n_q + n_t - 2} S_q + \frac{n_t - 1}{n_q + n_t - 2} S_t \tag{15}$$

$$\overline{x}^q = \frac{1}{n_q} \sum_{j=1}^{n_q} \overline{x}_j^q \tag{16}$$

$$S_q = \sum_{j=1}^{n_q} \left(x_j^q - \overline{x}^q\right)\left(x_j^q - \overline{x}^q\right)' \tag{17}$$

are the sample mean vectors and sum of the product of the sample covariance matrix of the query diseased image.

$$\overline{x}^t = \frac{1}{n_t} \sum_{j=1}^{n_t} \overline{x}_j^t \tag{18}$$

$$S_t = \sum_{j=1}^{n_t} \left(x_j^t - \overline{x}^t\right)\left(x_j^t - \overline{x}^t\right)' \tag{19}$$

are the sample mean vectors and sum of the product of the sample covariance matrix of the FV of disease database.

- ***Critical region:*** The query diseased leaf image and the FVs of the disease database are judged to be same, if $T^2 \leq c^2$, where, c^2 is the upper critical value of the F-distribution with $(n_q + n_t - 2)$ degrees of freedom at a significance level α; otherwise, it is inferred that the query diseased leaf image does not match with the FVs of the disease database.

Distance Metrics

Furthermore, to emphasize the efficacy of the proposed system, the extracted feature vector of the query diseased leaf image is compared to the feature vectors of the disease database using Canberra distance metric. Kokare et al. (2003) have performed a comparative study of the nine distance metrics – Manhattan(L1), Weighted-mean-variance (WMV), Euclidean (L2), Chebychev (L¥), Mahalanobis, Canberra, Bray-Curtis, Squared chord, and squared chi-squared distances – for texture image retrieval. They have reported that the Canberra distance metric yields a better result than the others. Thus, this chapter adopts the Canberra distance metric to recognize diseases and identify the right pesticides. The Canberra distance metric is presented in Eq. (20).

$$D_{canberra}\left(FV^q, FV^d\right) = \sum_{i=1}^{r} \frac{\left|FV^q - FV^d\right|}{\left|FV^q\right| + \left|FV^d\right|} \tag{20}$$

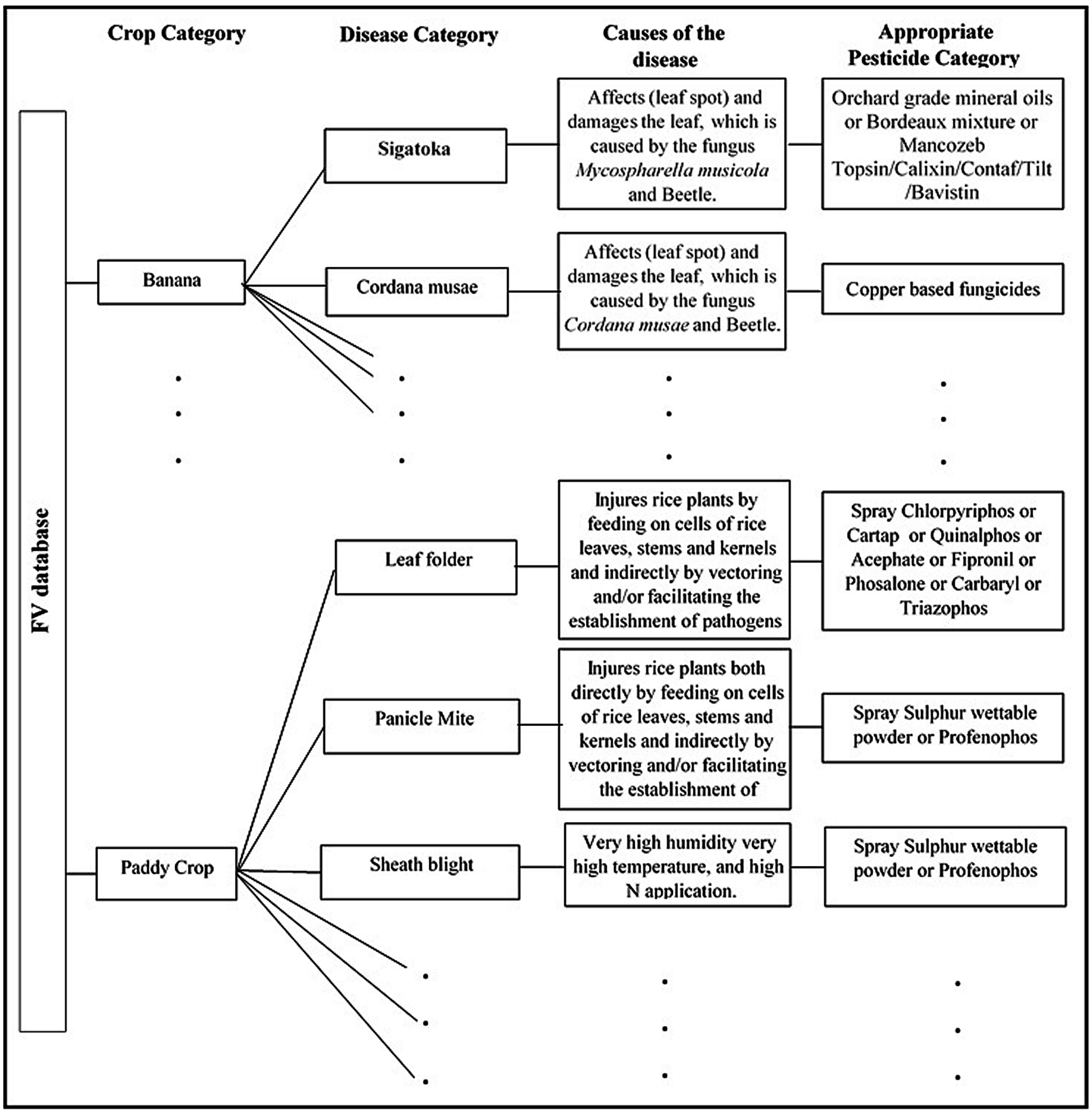
where, 'i' represents the i-th FV of the disease database.

EXPERIMENTS AND RESULTS

To identify diseases and recognize appropriate pesticides, the diseased part of the leaf image (RoI) was identified and segmented separately, which has been illustrated in Figure 4. Before performing the segmentation process of the RoI, the input leaf image was preprocessed, that is, noise removed. The color features, such as R, G, and B colors, were extracted from the preprocessed leaf images. The extracted crop features assumed to be a Gaussian random field. To identify and characterize the diseases, first, the modified Wishart distance measure applied. The Wishart distance measure was performed to diagnose the diseases in terms of variation between the input query leaf image and the features of the FV database. If the query FV and the FVs of the disease database passed the Wishart distance, then the Hotelling's T^2 statistic was employed on the color features. If the query FV and the FVs of the disease database fail to pass the test, then the test was dropped and has taken another FV from the disease database. The Hotelling's T^2 statistic was performed to diagnose the disease in terms of mean vectors (spectrum of energy) of the query diseased leaf image and the features of the FV database. If the query FV and the FV of the disease database pass both Wishart distance and Hotelling's T^2 statistics, then the specific diseases were almost confirmed.

Furthermore, some statistical features, such as Autocorrelation (ACC), Coefficient of Variation (CoV), Skewness, (Skw), and Kurtosis (Kur), were extracted from the RoI, and the Canberra distance applied. The outcome of the Canberra distance noticed that the leaf was infected. Based on the disease type and severity of the diseases, the system identified the right pesticide from the pesticide database. Finally, the system recommends the pesticide, quantity of pesticide to be applied, and the procedure to apply, to the farmers. For example, the structure of the disease database, pesticide database, and their feature vectors have been depicted in Figure 8. The proposed IoT-based CDM system experimented

Figure 8. Detailed structures of the Disease database, Pesticide database



with a model (system) at a farm in which Banana trees were cultivated. How the model CDM system experimented has been illustrated herein.

A camera was fixed at a farm where banana trees were cultivated, and it was connected to a Raspberry PI processor. The PI processor was placed at the workstation, which had been fully facilitated with the IoT components. The camera continuously photographed the Banana trees and sent them to the PI processor. The CDM system identified a leaf image that was infected by the disease. The identified leaf image was preprocessed and the diseased part of the leaf was segmented separately. Then the system extracted the features from the diseased part (RoI). The extracted features were formed as a feature vector and compared with the features of the disease database, based on the theoretical concepts discussed in Section 6, and recognized that the Black Sigatoka disease has affected the leaf. The Sigatoka is a fungal and leaf spot disease of the Banana, and it was caused by the *Mycosphaerella fijiensis*. Moreover, the system characterized the disease with a view of remedial treatments to be applied, so that it identified the right pesticides from the pesticide database; and the proposed IoT-based CDM system recommended the right pesticide to the farmer.

Comparative Study

The proposed IoT-based CDM system correctly detects the normal (not disease affected) leaf images up to 98.98 percent while it detects diseased leaf images at an early stage up to 92.89 percent; detects the diseases at final stage up to 86.94 percent.

The obtained results of the proposed system were compared with the existing methods proposed by Wang et al. (2017), and the results have been tabulated in Table 1.

Table 1. Average Disease Detection Rate in Different Stages and Computational Time Complexity

Methods	Early Stage (%)	End-Stage (%)	Middle Stage (%)	Time Complexity
Proposed Method	93.86	88.01	83.45	51 seconds
Deep VGG16 Model	93.10	87.0	83.3	68 seconds

Though the proposed method gives a moderate accuracy detection rate for different stages of diseases, it requires a minimum of time complexity compared to that of the deep VGG16 model proposed in (Wang et al., 2017). The VGG16 model proposed in (Wang et al., 2017) adopts a deep convolutional neural network, so it takes more computational time.

CONCLUSION

The proposed IoT-based CDM system experimented at a banana tree farm where a camera was fixed. The camera continuously photographed the banana trees in the field, and sent them to the Raspberry PI processor at a workstation, which was connected to the camera. The proposed IoT-based CDM system had taken only the leaf images into account of disease diagnose and for remedial treatment. After receiving the leaf images, the PI processor performed preprocesses, that is, noise removal and segmentation. Then extracted features from the disease infected parts of the leaf, and compared with the FVs database of the infected crop leaf images. If the leaf was infected, then recognize the right pesticide from the pesticide database. The recognized pesticide was recommended to the farmers. The outcome of the proposed IoT-based CDM system encourages both farmers' communities and researchers in this field. The proposed method gives better results than the existing methods in terms of disease detection accuracy rate and computational time complexity.

REFERENCES

Agarwal, Y., & Dey, A. K. (2016). Toward building a safe, secure, and easy-to-use internet of things infrastructure. *IEEE Comput.*, *49*(4), 88–91. doi:10.1109/MC.2016.111

Capello, F., Toja, M., & Trapani, N. (2016). A real-time monitoring service based on industrial internet of things to manage agrifood logistics. *Proceedings of 6th International Conference on Information Systems, Logistics and Supply Chain*, 1-8.

Dabboor, M., Yackel, J., Hossain, M., & Braun, A. (2013). Comparing matrix distance measures for unsupervised POLSAR data classification of sea ice based on agglomerative clustering. *International Journal of Remote Sensing*, *34*(4), 1492–1505. doi:10.1080/01431161.2012.727040

Dos Santos, U. J. L., Pessin, G., da Costa, G. A., & da Rosa Righi, R. (2019). AgriPrediction: A proactive internet of things model to anticipate problems and improve production in agricultural crops. *Computers and Electronics in Agriculture*, *161*, 202–213. doi:10.1016/j.compag.2018.10.010

Fang, S., Da Xu, L., Zhu, Y., Ahati, J., Pei, H., Yan, J., & Liu, Z. (2014). An integrated system for regional environmental monitoring and management based on internet of things. *IEEE Transactions on Industrial Informatics*, *10*, 159–1605.

Ferentinos, K. P. (2018). Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, *145*, 311–318. doi:10.1016/j.compag.2018.01.009

Fourati, M. A., Chebbi, W., & Kamoun, A. (2014). Development of a web-based weather station for irrigation scheduling. *Information Science and Technology (CIST)*, 37-42. 10.1109/CIST.2014.7016591

Harvey, C. A., Rakotobe, Z. L., Rao, N. S., Dave, R., Razafimahatratra, H., Rabarijohn, R. H., ... MacKinnon, J. L. (2014). Extreme vulnerability of smallholder farmers to agricultural risks and climate change in madagascar. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, *369*(1639), 20130089. doi:10.1098/rstb.2013.0089 PMID:24535397

Hashim, N., Mazlan, S., Aziz, M. A., Salleh, A., Jaafar, A., & Mohamad, N. (2015). Agriculture monitoring system:a study. *J. Teknologi*, *77*, 53–59.

Jagtap, S., Hambarde, S. (2014). Agricultural Plant Leaf Disease Detection and Diagnosis Using Image Processing Based on Morphological Feature Extraction, *IOSR Journal of VLSI and Signal Processing*, *4*, 24–30.

Johannes, A., Picon, A., Alvarez-Gila, A., Echazarra, J., Vaamonde, S. R., Navajas, A. D., & Ortiz-Barredo, A. (2017). Automatic plant disease diagnosis using mobile capture devices, applied on a wheat use case. *Computers and Electronics in Agriculture*, *138*, 200–209. doi:10.1016/j.compag.2017.04.013

Kodali, R. K., Rawat, N., & Boppana, L. (2014). WSN sensors for precision agriculture. In *Region 10 Symposium*, (pp. 651-656). IEEE. 10.1109/TENCONSpring.2014.6863114

Kokare, M., Chatterji, B. N., & Biswas, P. K. (2003). Comparison of similarity metrics for texture image retrieval. *Proc. IEEE Transactions on TENCON 2003, Conference on Convergent Technologies for Asia-Pacific Region*, 2, 571-575. 10.1109/TENCON.2003.1273228

Levy, A. (2016). *Cisco Wagers $1.4 billion on Internet of Things, CNBC*. Retrieved from: https://www.cnbc.com/2016/02/03/cisco- wagers- 14- billion- on- internet- of- things.html

Li, M., Chen, G., & Zhu, Z. (2014). Information service system of agriculture IoT. *Automatika J. Control, Meas. Electron. Comput. Commun*, *54*, 41–426.

Li, S., Xu, L. D., & Zhao, S. (2015). The Internet of Things: A survey. *Information Systems Frontiers*, *17*(2), 243–259. doi:10.100710796-014-9492-7

Lin, P., Abney, K., & Jenkins, R. (Eds.), *Robot Ethics 2.0*. New York: Oxford University Press.

Lu, Y., Yi, S., Zeng, N., Liu, Y., & Zhang, Y. (2017). Identification of rice diseases using deep convolutional neural networks. *Neurocomputing*, *267*, 378–384. doi:10.1016/j.neucom.2017.06.023

Ma, J., Dua, K., Zhenga, F., Zhangb, L., Gongc, Z., & Suna, Z. (2018). A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network. *Computers and Electronics in Agriculture*, *154*, 18–24. doi:10.1016/j.compag.2018.08.048

Mattihalli, C., Gedefaye, E., Endalamaw, F., & Necho, A. (2018). Plant leaf diseases detection and automedicine. *Internet of Things*, *2*, 67–73. doi:10.1016/j.iot.2018.08.007

Mohammed, J. (2016). *Predictions for the Internet of Things in 2016. World Economic Forum (2015)*. Retrieved from https://www.weforum.org/agenda/2015/12/5-predictions-for-the-internet-of-things-in-2016

Muangprathuba, J., Boonnama, N., Kajornkasirata, S., Lekbangponga, L., Wanichsombata, A., & Nillaorb, P. (2019). IoT and agriculture data analysis for smart farm. *Computers and Electronics in Agriculture*, *156*, 467–474. doi:10.1016/j.compag.2018.12.011

Ndzi, D. L., Harun, A., Ramli, F. M., Kamarudin, M. L., Zakaria, A., Shakaff, A. Y. M., & Farook, R. S. (2014). Wireless sensor network coverage measurement and planning in mixed crop farming. *Computers and Electronics in Agriculture*, *105*, 8–94. doi:10.1016/j.compag.2014.04.012

Pang, Z., Chen, Q., Han, W., & Zheng, L. (2015). Value-centric design of the internet-of things solution for food supply Chain: Value creation, sensor portfolio and information fusion. *Information Systems Frontiers*, *17*(2), 289–319. doi:10.100710796-012-9374-9

Qin, F., Liu, D., Sun, B., Ruan, L., Ma, M., & Wang, H. (2016). Identification of Alfalfa Leaf Diseases Using Image Recognition Technology. *PLoS One*, *11*(12), 1–26. doi:10.1371/journal.pone.0168274 PMID:27977767

Sanchez, P. A., & Swaminathan, M. S. (2005). Cutting world hunger in half. *Science*, *307*(5708), 357–359. doi:10.1126cience.1109057 PMID:15661994

Scroxton, A. (2016). *UK IoT research hub opens with support from academic world, ComputerWeekly.com*. Retrieved from https://www. computerweekly.com/ news/4500270034/UK-IoT-research-hub-opens-with-support-from-academic-world

Seetharaman, K., & Jeyakarthic, M. (2014). Statistical Distributional Approach for Scale and Rotation Invariant Color Image Retrieval Using Multivariate Parametric tests and Orthogonality Condition. *Journal of Visual Communication and Image Representation*, *25*(5), 727–739. doi:10.1016/j.jvcir.2014.01.004

Seetharaman, K., & Palanivel, N. (2013). Texture Characterization, Representation, Description and Classification Based on a Family of Full Range Gaussian Markov Random Field Model. *International Journal of Image and Data Fusion*, *4*, 342–362. doi:10.1080/19479832.2013.804007

Stankovic, J. A. (2014). Research directions for the Internet of Things. *IEEE Int. Things J.*, *1*(1), 3–9. doi:10.1109/JIOT.2014.2312291

Stewart, E. L., & McDonald, B. A. (2014). Measuring quantitative virulence in the wheat pathogen zymoseptoria tritici using high-throughput automated image analysis. *Phytopathology*, *104*(9), 985–992. doi:10.1094/PHYTO-11-13-0328-R PMID:24624955

Strange, R. N., & Scott, P. R. (2005). Plant disease: A threat to global food security. *Phytopathology*, *43*(1), 83–116. doi:10.1146/annurev.phyto.43.113004.133839 PMID:16078878

Tai, A. P., Martin, M. V., & Heald, C. L. (2014). Threat to future global food security from climate change and ozone air pollution. *Nature Climate Change*, *4*(9), 817–821. doi:10.1038/nclimate2317

UNEP Smallholders. (2013). *Food Security, and the Environment. Rome: International Fund for Agricultural Development (IFAD)*. Retrieved from: https://www.ifad.org/documents/10180/66cac2414b643c2876d9c2d1f01d5dd

Wang, G., Sun, Y., & Wang, J. (2017). Automatic Image-Based Plant Disease Severity Estimation Using Deep Learning. *Computational Intelligence and Neuroscience*, *2017*, 1–8. doi:10.1155/2017/2917536 PMID:28757863

Whitmore, A., Agarwal, A., & Xu, L. D. (2015). The Internet of Things: A survey of topics and trends. *Information Systems Frontiers*, *17*(2), 261–274. doi:10.100710796-014-9489-2

Xian, K. (2017). Internet of things online monitoring system based on cloud computing. *Int. Jo. Online Eng.*, *13*(9), 123–131. doi:10.3991/ijoe.v13i09.7591

Chapter 9
Analysis of Heart Disorder by Using Machine Learning Methods and Data Mining Techniques

Sarangam Kodati
Teegala Krishna Reddy Engineering College, India

Jeeva Selvaraj
Brilliant Institute of Engineering and Technology, India

ABSTRACT

Data mining is the most famous knowledge extraction approach for knowledge discovery from data (KDD). Machine learning is used to enable a program to analyze data, recognize correlations, and make usage on insights to solve issues and/or enrich data and because of prediction. The chapter highlights the need for more research within the usage of robust data mining methods in imitation of help healthcare specialists between the diagnosis regarding heart diseases and other debilitating disease conditions. Heart disease is the primary reason of death of people in the world. Nearly 47% of death is caused by heart disease. The authors use algorithms including random forest, naïve Bayes, support vector machine to analyze heart disease. Accuracy on the prediction stage is high when using a greater number of attributes. The goal is to function predictive evaluation using data mining, using data mining to analyze heart disease, and show which methods are effective and efficient.

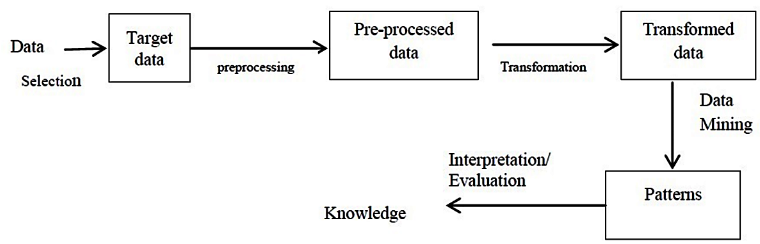
INTRODUCTION

The World Health Organization (WHO) estimates that by 2030, nearly 23.6 million people will die due to heart disease. The focus of this study is motivated by the WHO statistics and is focused on predicting heart disease using data mining techniques. To minimize the risk, estimates of heart disease should continue to be done. One of the most difficult and complex tasks in healthcare is analysing patient

DOI: 10.4018/978-1-7998-2108-3.ch009

symptoms and characteristics to correctly diagnose disease. Heart disease prediction uses the different parameters of a patient's diagnostic tests. This is a multi-layered issue as indicated by false presumptions and unpredictable impacts. Present day medical regions produce a vast amount of raw data about patients, clinic resources, disease analysis, systems that store patients' data, medicinal devices, etc. This vast amount of raw data is the essential asset that can be productively pre-prepared and analysed for data extraction that can directly or indirectly motivate a clinical organization's cost-effectiveness and support decision-making. Valid analysis about heart disease cannot remain conceivable by utilizing only human intelligence (Chaitrali et al, 2012). Data mining is a method concerned with separating large amounts of data from a vast amount of data. The data mining process is called Knowledge Discovery in Databases (KDD).

Figure 1. Knowledge discovery in databases



Most clinics keep their patient data in the form of electronic medical record (EMC) databases. These frameworks contain huge amounts of data. Emergency clinic data can be classified by the type of content data between the types of images. This necessity is driven by the utilization of KDD, which is in charge of changing information concerning low-level data into an abnormal state of learning for basic management. Data mining is one of the KDD process aims for discovering helpful examples from large datasets. These patterns can be further analysed and the outcomes can be utilized for effective decision-making and analysis. The number of tasks of data mining is classified as clustering and association analysis. In this study, different data mining classification methods are applied to clinical healthcare information related to heart diseases.

LITERATURE SURVEY

According to Sathish et al. (2015), heart disease can be estimated, from traits obtained from a patient's information, and the work of these researchers has presented a framework comprised of the characteristics of an individual's way of life, including essential characteristics such as gender, blood pressure level, cholesterol level, and other attributes obtained from the heart disease dataset. Data mining classification (for example, Naive Bayes), and machine-learning tools, such as Weka, are used to make predictions and analyse the results of a heart disease dataset. Comparative analysis (Kodati, 2018) classification techniques for heart disease, with data mining, using the Weka tool, utilize data mining algorithms such as KNN, Support Vectors Machine, and Random Forestto analyse precision and recall analysis of a heart disease dataset (Shamsher et al, 2013) in addition to healthcare medical data.

Svetlana (2004) presented a step-by-step explanation for WEKA data mining software in a *WEKA Explorer Tutorial*. It contained descriptions of data mining tasks such as data pre-processing, data classification, data clustering, data association, attribute selection, and visualization, using the Weka tools. Toward the end of the tutorial, the researcher gives a representation of the results with clarifications.

In Kalpana Rangra et al. (2014), can be found a theoretical and comprehensive analysis of different types of data mining tools of the comparative study analysis of machine learning algorithms and data mining classification. The study described data mining tools using specification, specialization features, and applications. This paper presented the specific details and descriptions of different types of data mining tools enlisted in the area of specialization. Comparative analysis in diagnosis of heart disease with data mining orange tool using machine learning algorithms includes Naive Base Classifier, Support Vector Machine, Random Forest, K-Nearest Neighbor, and Accuracy of Comparative Analysis of heart disease precision and recall.

HEART DISEASE DATASET

Figure 2 provides an overview of the database from the heart disease dataset of the UCI repository. It incorporates 13 attributes. The coronary illness dataset is included in this research work comprising a total of 270 instances with no missing values.

Figure 2. Heart disease dataset

No	Attribute	Type
1	age	Real
2	sex	{female, male}
3	cp	{typ_angina, asympt, non_anginal, atyp_angina}
4	trestbps	Real
5	chol	Real
6	restecg	{left_vent_hyper, normal, st_t_wave_abnormality}
7	thalach	real
8	restecg	{left_vent_hyper, normal, st_t_wave_abnormality}
9	exang	{no, yes}
10	oldpeak	real
11	slope	{up, flat, down}
12	ca	real
13	thal	{fixed_defect, normal, reversable_defect}
14	num	{'<50', '>50_1','>50_2', '>50_3', '>50_4'}

DATA MINING TOOLS

Data mining tools are an automatic collection and integration of data from a variety of internal data sources. Data mining software allows for the analysing of large volumes of raw data from healthcare systems, applications, database, websites, and text-based mediums. It extracts and manipulates information that is hidden in the raw data and is then reported in a well-formed structure. Parameters for comparing data mining tools include supported platforms and multiple types of algorithms.

Weka

Weka is an open-source software system where the code is publically available, and it has some machine learning algorithms (ML) can be used for data mining tasks. WEKA requires a file with the format (arff). The original home cardiac catheterization data set file stored in the Microsoft Excel (spreadsheet) with the format (xls). This study used WordPad software to change over the home data set into (arff) format, WEKA was created by the Waikato University at New Zealand.

Orange

The orange tool can be utilized in explorative data investigation and perception (Kodati et al, 2018). It provides a platform for test choice and preprocessing predictive and suggestion systems, and can be utilized in different types of area research, such as data mining, bioinformatics, etc. The orange tool is a data mining technique that is always preferred when the factors of novelty, reliability, and quality are involved.

Matlab

Matlab support for data mining provides an intelligent domain area for numerical, visualization, computation, and programming. Data investigation, developing algorithms, and various models and applications, such as the heart disease dataset and built-in math functions, and language tool explore a number of methods and help investigators reach a solution faster than using a spreadsheet of supporting programming languages, such as C, C++ and JAVA.

DATA MINING ALGORITHMS

Data mining is a classification of algorithms chosen for conducting tests. Different classifications of algorithms, such as Naïve Bayes, Random Forest, SVM, Decision Tree, etc., are discussed in the follows section.

Decision Tree

Decision Tree is a data mining supervised learning algorithm for classification and use for machine learning algorithms. The node at the topmost position in the tree-like graphs is called the root node. Decision Tree is widely used in the medical healthcare field, particularly for diagnosis of heart disease. Using Decision Tree, decision-making can select the best alternative, and travel from the root node according to the leaf nodes indicating a unique class based on the greatest data gain (Gupta, 2017).

Random Forest

Random Forest is a data mining classification of algorithm used for data mining tool that is built by constructing multiple decision trees using training time, and producing the class by voting of separate or individual trees. It is similar to the Decision Tree algorithm, but the data-mining algorithm constructs a forest of decision trees with locations of attribute selection at random. It has the advantage of computing efficiency, improving estimated accuracy without a huge increment in the computational cost. It can also estimate up to a multiple of explanatory variables (Li et al, 2015).

Support Vector Machine (SVM)

This classification algorithm is for supporting vector machines and is used as a data-mining tool (Ghumbre et al. 2011). The data mining supervised learning model is applied mainly for classification algorithms. The basic SVM works as a binary classifier where the training data is divided amongst two classes or multiclass problems. The primary SVM algorithm is executed repeatedly about the training data. The SVM algorithm maps feature vectors, with respect to a higher dimensional vector space, where a maximum margin edge hyper-plane is introduced among it space. The supporting vector machine utilization, with respect to separations close to the hyper-plane utilization is indicated by the close information point on each side that is maximized. This maximizes the margin and, thereby, produces the largest possible distance between the separating hyper-plane and the instances on each aspect on such has been confirmed after reducing an upper bound regarding the predicted generalization error.

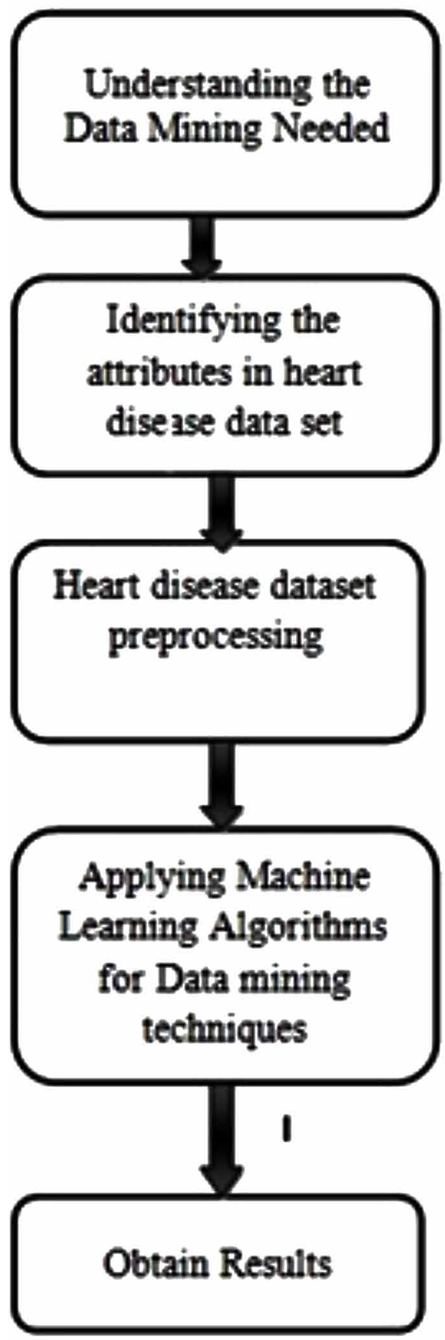
Naive Bayes

Data mining is a classification algorithm for Naive Bayes that is used as a data- mining tool. Naive Bayes characterization is a statistical-based classifier, which is based on the Bayes Theory. It expects that qualities are measurably free. This classifier is based on probabilities. It needs to determine $P(H|X)$, the probability as the hypothesis H holds given proof, i.e. data sample X. According to Bayes theorem, the $P(H|X)$ is expressed as $P(H|X) = (P(X|H)\,P(H)) / P(X)$. These Bayesian probabilities are utilized to decide the next occasion for the given case with regard to all the training data. Restrictive probabilities are resolved from the preparation information. This classifier yields an ideal forecast (given the presumptions). It can likewise deal with discrete or numeric attribute values.

DATA COLLECTION AND METHOD IMPLEMENTS PROPOSED

The heart disease dataset used in this study was collected from UCI. Understanding data, pre-processing data, and using data mining tools, such as are WEKA, ORANGE, and MATLAB, are the proposed implements.

Figure 3. The proposed system overview



Performance Analysis

The metrics analysis of the heart disease dataset using Precision, Recall, and F-Measure are discussed in this section. To analyze the performance and measurement of system stability, some parameters are calculated and analyzed. Some of these are:

TP is the quantity of true positive.
TN is the quantity of true negative.
FP is the quantity of false positive.
FN is the quantity of false negative.

Precision (P) is that a part of massive instances between the retrieved instances.

The equation of precision $= \frac{TP}{\left(TP + FP\right)}$

Recall (R) is the portion of instances that have true positive class and are predict as much positive.

The equation of Recall $= \frac{TP}{\left(TP + FN\right)}$

F-Measure

The F-measure is an analysis of considerations supporting the two times, regarding the precision times recall divided by the sum of the precision and recall. The equation of F-Measure given is:

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

RESULTS

In this area, the authors researched the analysis of heart disease and machine learning tool implementations, including like WEKA, ORANGE, and MATLAB, according to the results of the data mining technique classification algorithms, such as the Naive Bayes algorithm, Support Vector Machine algorithm, and Random Forest algorithm. Comparative analysis of the recall, F-Measure, and precision, classification algorithms over the dataset and across multiple performance measures. Ranking stability of an algorithm means that this algorithm should always produce accurate results and rank high over all datasets and across all evaluation measures precision, recall, and F-Measure. Compared to other support SVM and Naive Bayes, Random Forest shows good performance and it can analysis across all measures in the heart disease dataset.

Table 1. Performance measure or analysis of heart disease dataset using in Weka, Orange, and MatLab

Performance Measures	Precision	Recall	F-Measure
Naive Bays (WEKA)	0.837	0.837	0.8819
Random Forest (WEKA)	0.818	0.819	0.9659
SVM (WEKA)	0.84	0.836	0.8673
Naïve Bays (MATLAB)	0.7881	1.000	0.8818
Random Forest (MATLAB)	0.9341	1.000	0.9658
SVM (MATLAB)	0.7656	1.000	0.8674
Naïve Bays (ORANGE)	0.824	0.806	0.8819
Random Forest (ORANGE)	0.779	0.734	0.9659
SVM (ORANGE)	0.817	0.705	0.8673

CONCLUSION

In this research, data mining classifications and machine learning algorithms are discussed in different analysis, which are used to obtain an efficient heart disease dataset. Data mining techniques and tools were used, in addition to implementation of the Weka, MatLab and Orange tools, to imitate health decision-making with improved accuracy. Moreover, a general analysis was conducted using a heart disease dataset.

Figure 4. Precision value of heart disease using in Weka, MatLab, and Orange precision

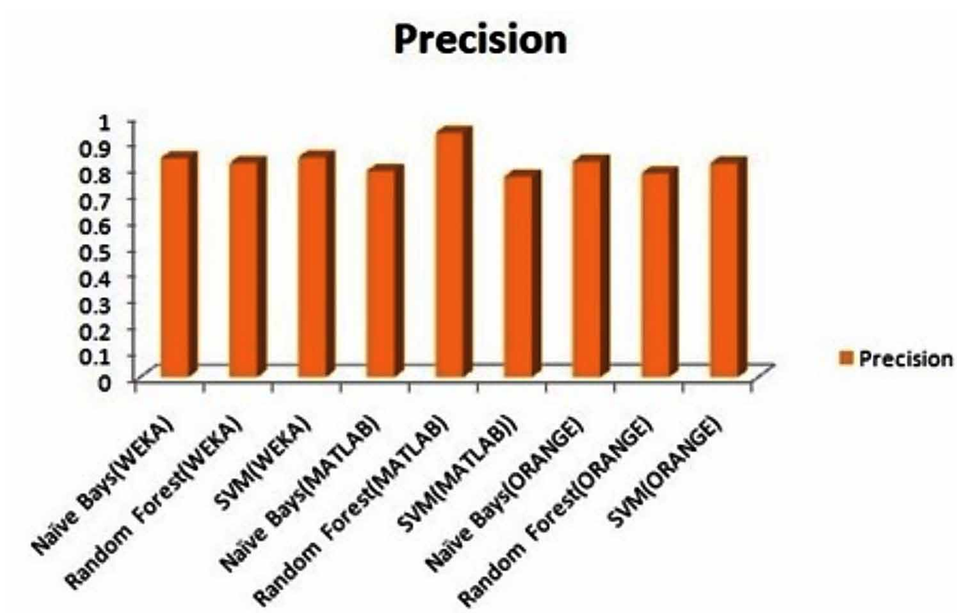
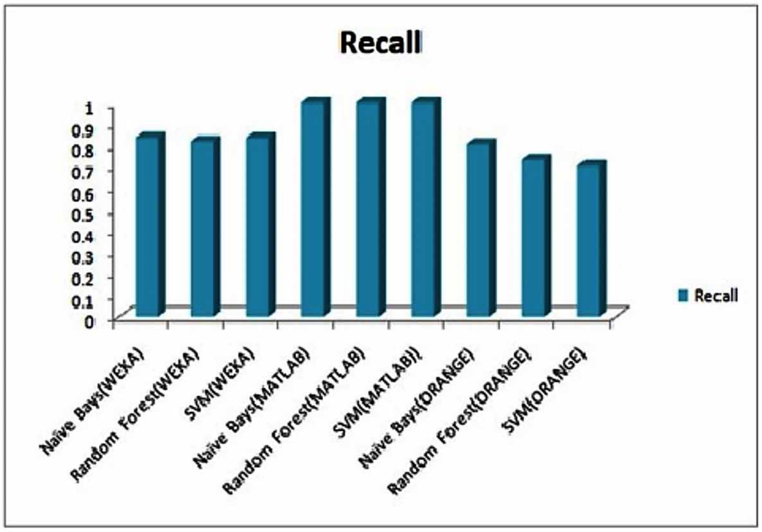
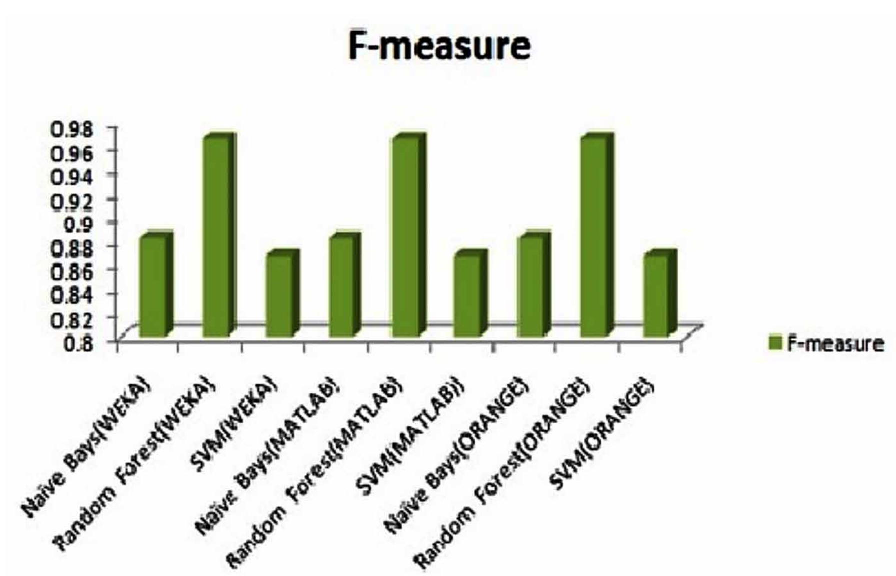


Figure 5. Recall value of heart disease using in Weka, MatLab. and Orange recall



It was noted that various authors utilized different techniques, technologies, and an alternate number of properties to study various innovations that have provided precision and recall contingent upon various thoughts regarding attributes. Heart disease was detected via precision, recall, and F-Measure using Naïve Bays, Random Forest, and support vector classification algorithm levels, which also provided for several of the attributes in this paper using data mining classification and machine learning algorithms to analyse the heart disease dataset.

Figure 6. F-measure value of heart disease using in Weka, MatLab, and Orange F-Measure



In the future, data mining unsupervised learning algorithms and using different data mining tools in research work can provide a heart disease dataset. The heart disease estimates can utilize real on-going information obtained from local clinical care centres.

REFERENCES

Aksenova, S. S. (2004). Machine learning with Weka. *Weka Explorer Tutorial for WEKA Version*, *3*(4), 2–44.

Dangare, C., & Apte, S. (2012). A data mining approach for prediction of heart disease using neural networks. *International Journal of Computer Engineering and Technology*, *3*(3).

Ghumbre, S., Patil, C., & Ghatol, A. (2011, December). Heart disease diagnosis using support vector machine. International conference on computer science and information technology (ICCSIT') Pattaya.

Gupta, B. (2017). Analysis of various decision tree algorithms for classification in data mining. *International Journal of Computers and Applications*, *163*(8), 15–19.

Kirkby, R. (2002). *Weka explorer user guide for version 3-3-4.* University of Weikato.

Kodati, S., & Vivekanandam, R, R. (2018). Comparative analysis of classification techniques for heart disease with data mining Weka tool. *Journal of Adv Research in Dynamical & Control Systems*, *10*(9), 2224–2227.

Lakshmi, K., Krishna, M. V., & Kumar, S. P. (2013). Performance comparison of data mining techniques for predicting of heart disease survivability. *International Journal of Scientific and Research Publications*, *3*(6), 1–10. https://archive.ics.uci.edu/ml/datasets/Heart+Disease

Li, L., Wu, Y., & Ye, M. (2015). Experimental comparisons of multi-class classifiers. Informatica, 39(1).

Obenshain, M. K. (2004). Application of data mining techniques to healthcare data. *Infection Control and Hospital Epidemiology*, *25*(8), 690–695. doi:10.1086/502460 PMID:15357163

Patel, S. B., Yadav, P. K., & Shukla, D. P. (2013). Predict the diagnosis of heart disease patients using classification mining techniques. *IOSR Journal of Agriculture and Veterinary Science, 4*(2), 61-64.

Rangra, K., & Bansal, K. L. (2014). Comparative study of data mining tools. *International Journal of Advanced Research in Computer Science and Software Engineering*, *4*(6), 216–223.

Sarangam Kodati, D. R. V. (2018). *Analysis of heart disease using in data mining tools Orange and Weka*. *Global Journal of Computer Science and Technology*.

Sathish, M., & Sridhar, D. (2015). Prediction of heart diseases in data mining techniques. *International Journal of Computer Trends and Technology*, *24*(1), 45–49. doi:10.14445/22312803/IJCTT-V24P109

Chapter 10
Deep Learning in Engineering Education:
Implementing a Deep Learning Approach for the Performance Prediction in Educational Information Systems

Deepali R. Vora
Vidyalankar Institute of Technology, India

Kamatchi R. Iyer
Amity University, India

ABSTRACT

The goodness measure of any institute lies in minimising the dropouts and targeting good placements. So, predicting students' performance is very interesting and an important task for educational information systems. Machine learning and deep learning are the emerging areas that truly entice more research practices. This research focuses on applying the deep learning methods to educational data for classification and prediction. The educational data of students from engineering domain with cognitive and non-cognitive parameters is considered. The hybrid model with support vector machine (SVM) and deep belief network (DBN) is devised. The SVM predicts class labels from preprocessed data. These class labels and actual class labels act as input to the DBN to perform final classification. The hybrid model is further optimised using cuckoo search with levy flight. The results clearly show that the proposed model SVM-LCDBN gives better performance as compared to simple hybrid model and hybrid model with traditional cuckoo search.

DOI: 10.4018/978-1-7998-2108-3.ch010

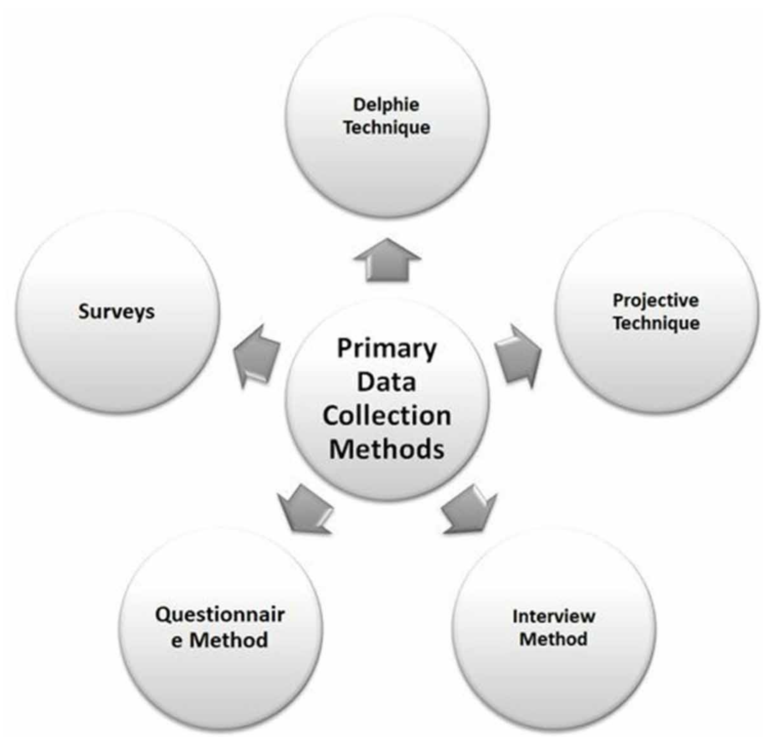
INTRODUCTION

Educational Information System lies at the heart of any educational institute to monitor the educational goals. One important goal of the educational system among many is tracking the performance of the student. Many techniques and algorithms are used to track the progress of students. This domain has gained importance with the increase in data volume and the development of new algorithms. (Vora & Iyer, 2018)

Data generated from various educational sources is explored using different methods and techniques in EDM. The multidisciplinary research that deals with the development of such methods and techniques are the focus of EDM. Analysis of educational data could provide information about student's behaviours, based on which education policies could be enhanced further (Sukhija, Jindal, & Aggarwal, 2015, October). EDM discusses the techniques, tools, and research intended for automatically extracting the meaning from large repositories of educational systems' data.

According to Davies (Davis, 1998), "Education has become a commodity in which people seek to invest for their own personal gain, to ensure equality of opportunity and as a route to a better life." Because of this Higher education providers are competing mainly for students, funding, research and recognition within the wider society. It seems important to study data of students studying professional courses as for the growth of any nation producing better professionals is the key to success. Higher education system faces two main challenges: finding placements and students dropping out. Analysis of educational data

Figure 1. Predictive Modelling



can help in answering the two major challenges satisfactorily. Predicting the performance leads to better placements and minimise the dropouts.

A statistical technique to predict future behaviour is known as Predictive modelling. Predictive analytics is used widely in the area of product management and recommendation. It is a powerful tool to understand the data at hand and get useful insights from it. Figure 1 represents Predictive analytics in education.

One of the most popular methods for predictive analytics is Machine learning to predict future behaviour. From the plethora of algorithms available, it is always interesting to discover which algorithm or technique is most suitable for analysis of data under consideration. Educational Data Mining is the area of research where predictive modelling is most useful. Predictive analytics in Education can help in many ways such as; to identify weaker and dropout students, to identify best learning practices, to predict students' performance at every stage, tracking the placements of education etc. There is a need for the evolution of more and more new techniques to create an accurate classification of data and prediction based on that.

Machine Learning (ML) has become very popular among researchers because of the astonishing results the algorithms are giving for diverse data and applications. But when data is growing enormously simple ML are not efficient and beneficial. Meantime there are lot many advances in hardware and software. So it was possible to have more complex and hybrid architectural models performing various DM or Big Data tasks. Big data is already posing a challenge on traditional ML models for efficiency and accuracy. Various hybrid models are proposed and tested in many domains to tackle these challenges and are proved to be useful. Thus applying a hybrid model in the education domain will be useful.

ML is changing in a better way to tackle new age data and one of such advances is Deep Learning. Nonlinear data analysis can be effectively done using deep learning. Characteristics of the data can be effectively analysed using layers in the deep learning model. Deep learning is being applied in many domains; predominantly in image processing and natural language processing (Deng & Yu, 2014). Thus it is interesting to apply Deep Learning in the field of education.

This chapter addresses the main objectives as:

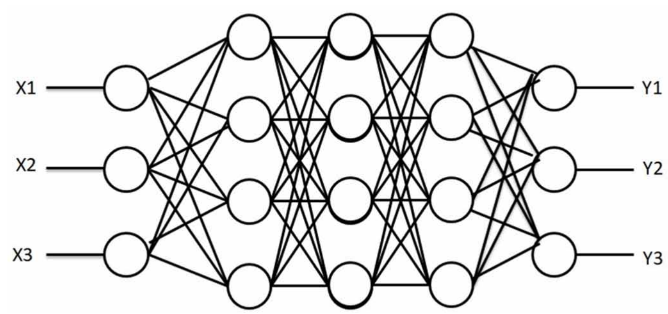
1. Identification of recent state of EDM and EDM techniques
2. Identification of areas where Deep Learning is applied and is useful
3. Applying Machine learning techniques on Educational Data for classification
4. Applying hybrid classification method using Deep Learning on Educational Data for Classification

BACKGROUND

Deep Learning

Hinton and colleagues suggested the concept of Deep Learning in the year 2006. Deep Learning (DL) is capable of learning from small data sets. The learning is through a nonlinear network structure. The Deep Learning is made up of the network structure with normally more than 4 hidden layers with one input and one output layer. Such a network can transform the raw features of images into superior features thereby making classification and prediction better (Bengio, 2009) (Najafabadi, et al., 2015).

Figure 2. A Deep Architecture



A powerful framework for supervised learning is provided by DL. To analyse complex representations more layers can be added with more number of units. Given a sufficient amount of training data, a deep algorithm is able to transform input X to output Y efficiently. For supervised learning, deep algorithms are well suited.

DL differs from ML in many ways. In terms of accuracy of algorithms, DL performs much better than normal ML. when data increases, DL learns fast from such ever increasing data thereby increasing accuracy. In contrast, ML algorithms are restricted by the representation of data which hampers the response time and accuracy of the system using such algorithms. Consider an example of email spam filtering. To identify if an email is a spam or not, the ML algorithm is given various representations of a good and bad email. Using which incoming emails are categorized as good or bad. ML algorithm directly without any representations will not be able to decide on anything.

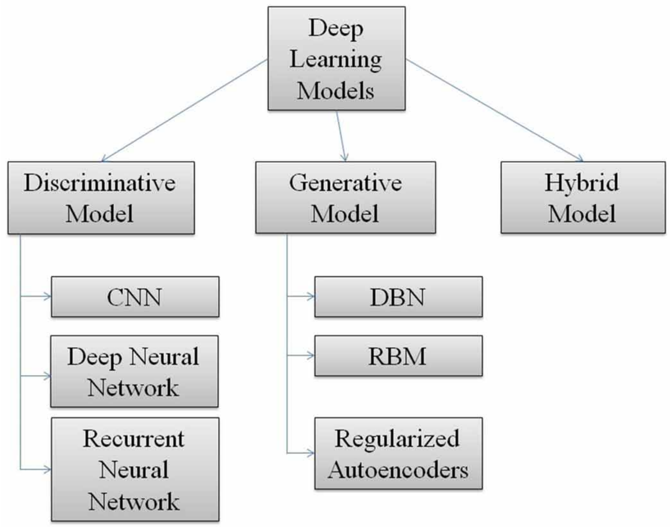
Here, DL comes to the rescue. Identification of important features and learning from them is easily performed by DL. DL algorithms can identify the features from the raw data and create representations for learning Deep learning has numerous algorithms of machine learning. These algorithms attempt to model high-level abstractions in data. They create or design architectures which are composed of many non-linear transformations. Deep architectures can be modelled using any combinations of layers of a network, but still, it has set of traditional algorithms such as Stacked AutoEncoder, Deep Boltzmann Machines, Deep Convolutional Networks and Deep Belief Networks. Figure 3 shows the set of predefined DL models.

In general, the model of deep learning technique can be classified into discriminative, generative, and hybrid models (Alwaisi & Baykan, 2017).

Discriminative models are used for modelling dependency of unobserved (target) variable Y on observed variables X whereas the generative models are used for learning the joint probability distribution. The generative model learns the full relationship between input X (features) and label Y giving maximum flexibility at the time of testing. Discriminative models learn from the only X to predict Y using conditional probability. By using few modelling assumptions these models can use existing data more efficiently. CNN, deep neural network and recurrent neural network are Discriminative models and DBN, restricted Boltzmann machine, and regularized autoencoders are generative models. Hybrid deep models are a combination of discriminative and generative models.

These DL models are used in various different application areas to gain better accuracy or output. Omid E. Davidt, Nathan S. Netanyahu (Davidt & Netanyahu, 2015,July) used DBN for malware detec-

Figure 3. Deep Learning Models



tion. Dropout method was used while training the network. Various layers were used to detect malware signatures. The network was trained using a GPU to detect 30 signatures.

Ni GAO et. al (GAO, GAO, Gao, & Wang, 2014,November) used DBN for intrusion detection. DBN has proved more accurate than SVM and Artificial Neural Network (ANN). DBN with 4 different configurations was used. The performance of shallow DBN is same as SVM and ANN. DBN with 2 and more hidden layers gave better output. The DBN is used for multiclass classification.

Grigorios & Aristidis (Tzortzis & Likas, 2007,October) used DBN for Spam Filtering. The performance was compared with SVM and DBN and was found more accurate.

Nguyen, Fookes & Sridharan (Nguyen, Fookes, & Sridharan, 2015,September) has used Deep Convolutional DBN for classification of images. It was observed that accuracy is improved and training time for the deep network reduced.

Pascal Vincent et al. (Vincent, Larochelle, Lajoie, Bengio, & Manzagol, 2010) used DBN and DAE (Denoising AutoEncoder) for analysing the images. Experimental results show that DAE was helpful for learning of higher level representations. DBN and DAE gave better accuracy for image classification when combined with SVM.

Kim, Minho Lee & Shen (Kim, Minho, & Shen, 2015,July) experimented with a new model created by combining Autoencoder, Deep SVM and GMM. The input was fed to SVM and then to GMM forming one layer. Thus deep layers were constructed for feature extraction and then a Naïve Bays algorithm was used for classification.

Kuwata & Shibasaki (Kuwata & Shibasaki, 2015,July) used SVR (Support Vector Regression) with Linear Rectifier Units for estimating the crop yields from remotely sensed data. This paper described Illinois crops yield estimation using deep learning and machine learning algorithm. Experimentation was done using Caffe tool. SVM with Gaussian Radial Bias function was used for the same experimentation and proved that traditional SVM overfits the regression model making accuracy low.

Francis (Lauzon, 2012) used Stacked Autoencoder with SVM for classification. They used a hybrid model of SDA with SVM and SDA with logistic regression. Also, the simple SVM and KNN for classification were also used on MINST dataset for classification of images. Furthermore, it was observed that SDA with SVM proved to be more accurate than SDA with logistic regression. It was also observed that preprocessing of data was not required with a hybrid deep model.

Wiering et al. (M. A. Wiering, Millea, Meijster, & Schomaker, 2016) used Deep SVM for the regression analysis. The deep model was constructed by stacking two layers of SVM. Initial layers were used for extracting the important features and final layer was used for classification.

Yue Deng et al. (Deng, Zhiquan Ren, Kong, Bao, & Dai, 2017) used Fuzzy Deep Neural Network for the classification of financial trading data. The deep network was given a high-level representation of data. This representation was generated by the fuzzy model and the neural network model.

Bo Guo et al. (Guo, Zhang, guang, Shi, & Yang, 2015,July) used deep learning to predict students' performance. They used sparse Autoencoders for classification and prediction.The network was trained using a backpropagation algorithm. The experimentation was done on data collected from 9th-grade high school children. The experimentation was carried out on GPU and CPU. The observed accuracy of DL algorithm was higher than SVM and Naive Bayes algorithm.

Arjun Raj Rajanna et. al. (Rajanna, Aryafar, Shokoufandeh, & Ptucha, 2015) used deep Neural network for classification of music. Rectilinear Unit (RLU) was used as an activation function in a deep neural network with 2 hidden layers. The accuracy of the classifier is improved significantly.

In addition to the above mentioned, there are many applications in various domains where DL algorithms or deep networks are used very effectively.

From the study of various articles, it is evident that DL is applied widely in many areas. The improvement in hardware has also made application of DL feasible. These algorithms are proved to give better accuracy in many cases than other traditional machine learning algorithms. Still, there are many domains where Deep Learning may prove beneficial, one of those being Educational System. In many articles, the Deep Learning algorithms are compared with traditional machine learning algorithms and are observed to be more accurate. Many articles proved that DL algorithms improve accuracy over traditional ML algorithms.

Also, the review of articles suggests that applying Deep Learning algorithm with other generalised algorithms may give better results in classification and prediction tasks. Through the survey, it is observed that hybrid models are more popular than plain DL algorithm based models (Vora & Iyer, A Survey of Inferences from Deep Learning Algorithms, 2017). In many applications standard dimensionality reduction algorithms are used to reduce the features and then DL algorithms are applied to improve accuracy.

Educational Data Mining

EDM is a popular research area and an ample amount of research articles are available for study. These research articles indicate the experimentation and algorithms used in EDM for performing various tasks. For the performance prediction, various new techniques and ML algorithms have experimented. There are many factors or features which have a significant effect in predicting the performance of the students. These factors are classified as cognitive and non-cognitive factors. Cognitive factors refer to characteristics of the person that affect performance and learning. But non-cognitive factors also play an important role in various EDM goals.

Wattana & Nachirat (Punlumjeak & Rachburee, 2015,October) used various techniques like K-Nearest Neighbourhood, Naïve Bays, and Neural Network to classify the students' data. The features considered were very few and majority attributes were related to marks of students.

Norlida, Usamah & Pauziah (Buniyamin, Mat, & Arshad, 2015, November) used Neuro Fuzzy algorithms to predict the performance of the engineering student. Here only 6 linguistic parameters are used for prediction.

Camilo, Elizabeth & Fabio (Guarín, Guzmán, & González, 2015) used Decision tree and Bayesian Classifier for prediction of students' performance. Students' admission test score and academic information were used for prediction. In addition, few socio-economic parameters were also used for prediction. The major stress was on the admission parameters.

Phung, Chau & Phung (Phung, Chau, & Phung, 2015,November) used Rule Extraction algorithm for classification in EDM. The algorithm is able to handle discrete and continuous data. The algorithm has a major challenge in creating compact rules. The numerous rules formed made the system difficult to use with more parameters.

Wen and Patrick (Shiau & Chau, 2016) and Sadaf & Eydgahi (Ashtari & Eydgahi, 2017) used Statistical modelling for EDM. Statistical methods are not able to support the change in population and size. Also, it was difficult to handle lead time bias.

Fernando et al. (Koch, Assunção, Cardonha, & Netto, 2016) used Partial Least square method and proved that it was cost effective. Here the method was sensitive to the choice of parameters. The parameters used were few.

Janice et al. (Gobert, Kim, Pedro, Kennedy, & Betts, 2015) and Anjana, Smija & Kizhekkethottam (Pradeep, Das, & J, 2015) used Decision trees in EDM. Limited features were used while predicting the performance. As well tree structure was prone to sampling error. The accuracy was affected by imbalanced data.

Evandro B. Costa et al. (Costa, Fonseca, Santana, Araújo, & Rego, 2017) used Naïve Bays, Decision Tree, SVM and Neural Network to predict the performance of the students. The data used was collected from distance learning and on-campus students. Performance data per week for the four weeks was collected and analysed for the effectiveness of the algorithms. Here only test results per week were considered, no other factors were considered.

Wanli et al. (Xing, Guo, Petakovic, & Goggins, 2015) used genetic programming for predicting Students' performance. The genetic algorithm produced an optimised prediction rate. While predicting, less consideration was given to the qualitative aspects. They monitored closed classroom learning of students and identified the factors which affect the performance. The participation of the student in various activities was majorly considered.

Xin Chen, et al. (Chen, Vorvoreanu, & Madhavan, 2013) studied social data to identify the factors which affect the behaviour or performance of students as study-life balance, lack of sleep, lack of social engagement, and lack of diversity.

Michail N. Giannakos et al. (Giannakos, et al., 2017, April) identified various cognitive factors like academic performance, attendance etc. and its effect on students' performance.

Hijazi & Naqvi (Hijazi & Naqvi, 2006) and Shoukat (Shoukat, 2013) has studied the impact of various cognitive and non-cognitive on students' performance.

Mushtaq & Khan (Mushtaq & Khan, 2012) proved that communication, learning facilities, proper guidance and family stress has a direct impact on students' performance. As well as Omar & Dennis (2015) used many factors for study and identified which factors played a vital role in students' performance.

Suryawan & Putra (Suryawan & Putra, 2016) did a detailed survey to identify the factors which affect students GPA. Also, regression tests and correlation analysis were done on various factors. It proved that the entrance exam and attendance in the class were important factors. Lecturer quality was also important and has an effect on GPA.

Park, Luo & Kim (Park, Luo, & Kim, 2015,October) studied parameters like gender, the academic performance of previous semesters, derailed enrolment, major related, credit, stop out years, age are considered. It was found that gender and age are not significant, previous academic performance is important.

In an interesting article in 2016, Pooja Mondal (Mondal) identified various factors like intellect, learning, physical, mental, social and economic as factors which affected students' behaviour and performance.

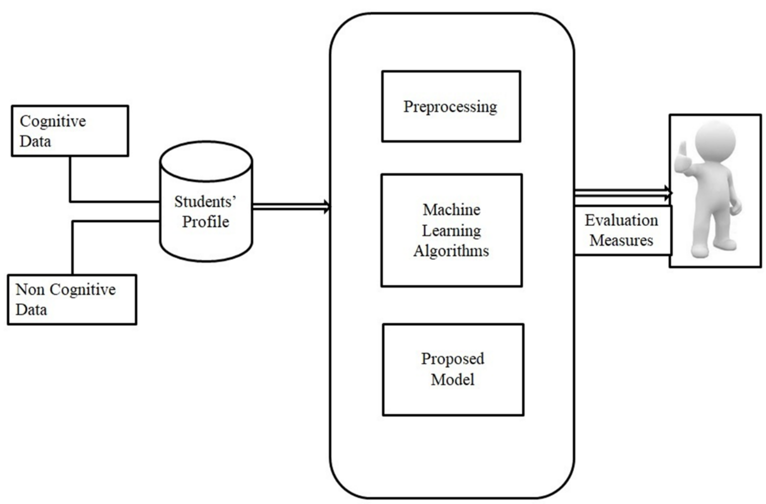
Most of the research is centred on the application of Data Mining and Machine Learning techniques in the classification task for students' performance. Classification and prediction task widely uses Classification methods based on learning from examples, such as Decision Tree, Artificial Neural Networks and Support Vector Machine algorithms. Although hybrid algorithms gained popularity for solving complex problems, they are not cited as commonly as the other methods in students' performance classification and prediction (Vora & Kamatchi, EDM – Survey of Performance Factors and Algorithms Applied, 2018).

PERFORMANCE PREDICTION USING MACHINE LEARNING

Experimental Setup

The experimental setup is divided into three parts as (i) Data Collection and preparation (ii) Design and implementation of the model and finally, (iii) evaluation of the model for the problem identified. Figure 4 shows these steps clearly.

Figure 4. Experimental Setup



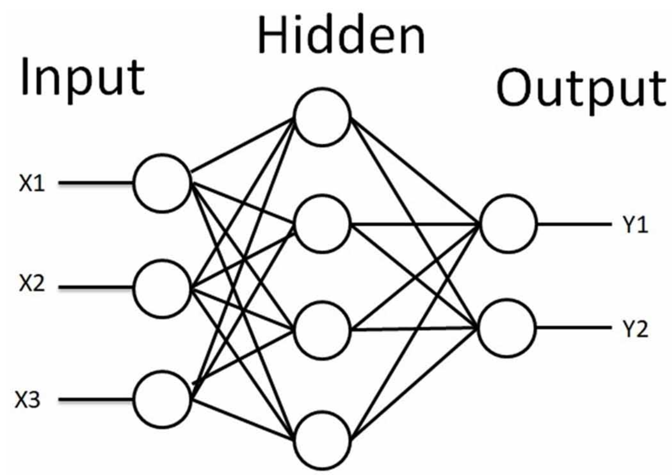
For any experimentation, input data plays a vital role. Thus one of the important steps of experimental design is the collection of relevant data. The only collection of data is not sufficient, but making it ready as per the requirement of the model is also important. Thus Data Preprocessing becomes a quintessential step in an experimental setup. Once data is prepared or preprocessed as per the requirement of the model, existing models and new suggested models are implemented. The comparative results provide new insights regarding the usefulness and accuracy of the model.

Algorithms

Neural Network

The Artificial Neural Network (ANN) is a method that is majorly applied to provide a solution for data mining applications. NN is composed of a set of processing units in a closely interconnected network. Such a structure exhibits the features of biological neurons' structure thereby providing an opportunity to implement a parallel concept. The parallelism can be achieved at each level and makes NN a fault tolerant structure. Neural structure organised in layers is shown in Figure 5.

Figure 5. Artificial Neural Network



The connecting neurons are the strengths of the neural computations. Each processing element has inputs with weights, transfer function and one output (Alpaydin, 2004).The main characteristic of NN is the learning process which is iterative in nature. There are many types of NN but most popular and widely used is feedforward network also called as feedforward multilayer perceptron. In a Multilayer Perceptron type of NN based on the complexity of application and data the number of hidden layers is increased. But generally, one or two hidden layer NN are most popular to handle classification problems. NN is used widely in educational data mining to perform various EDM tasks.

The neural network used has the number of input neurons representing each attribute and the number of output neurons equal to the number of output classes. The numbers of attributes are 35 and unique numbers of classes are 4. There are four classes so numbers of output neurons are four. One hidden layer with 10 neurons was devised for classification. The PCA(Principal Component Analysis) is used to reduce the number of input attributes to 18. Eq.1 represents the mathematical representation of a single layer neural network.

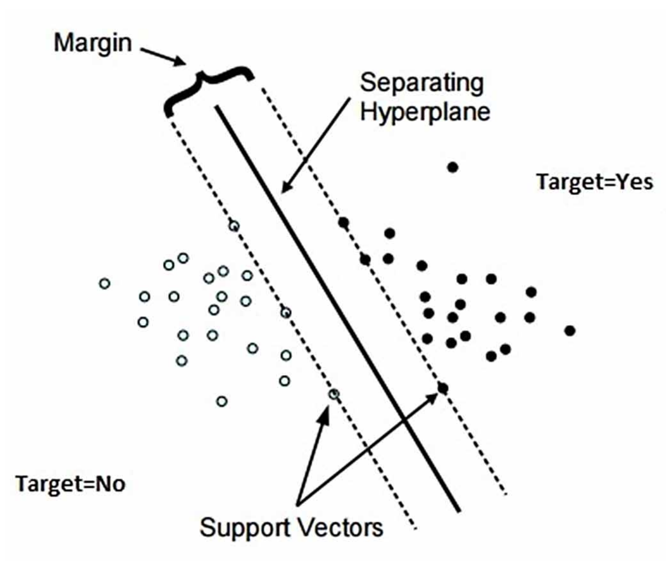
$$y = \sum_{i=1}^{n} Wi * Xi + W_0 \tag{1}$$

Here *Xi* represents the input parameters, *Y* represents the output class, and *W* represents the weights for each input layer neurons. The activation function used is sigmoid which is represented by Eq.2

$$y\left(z\right) = \frac{1}{1 + \exp\left(-zt\right)} \tag{2}$$

Support Vector Machine

Figure 6. SVM – Support Vectors and Hyperplane Representation



SVM (Yuan, et al., 2017) is a supervised machine learning algorithm for classification or regression problems. Basically, SVM is a two-class classifier that generates a hyperplane to classify two data segments. SVM is a discriminant based method. Instead of identifying the class densities it tries to identify the class boundaries. According to the statistical theory, the major aim of SVM is the finding of an optimal (maximize) margin. This optimal margin is defined by the minimum distance between the hyperplane and any of the sample points.

Figure 6 shows the support vectors and hyperplane.

The hyperplane of a two-class linearly separable issue in an n-dimensional feature space is defined as in Eq. 3

$$Y = \begin{cases} +1, & if\, w^T X + w_0 \geq 0 \\ -1, & otherwise \end{cases} \tag{3}$$

The point that makes Y=+1 or -1 is named as support vector, X represents the data point and w represents the weights. Here w_0 is the distance from hyperplane to the origin.

The distance of perpendicular from a specific point to the hyperplane is given as in Eq. (4).

$$y = \frac{\left| W^T X^t + W_0 \right|}{\left\| W \right\|} \tag{4}$$

Which can be rewritten as, when y^t is in the range {-1,+1}

$$\frac{y^t \left(W^T X^t + W_0 \right)}{\left\| W \right\|} \tag{5}$$

The ultimate aim of SVM is the identification of a hyperplane to maximize the distance between hyperplane as well as the points of training data that are closest to the hyperplane. This issue is then modified into the given equivalent convex quadratic issue, which is defined in Eq. (6).

$$\min \frac{1}{2} \left\| W \right\|^2 \tag{6}$$

subject to the condition as defined below

Eq. (6) is a standard quadratic optimization problem. To use this equation in a non-linearly separable problem domain there is a need to map the problem into a new space with nonlinear basis functions. So Eq. (6) is modified using Lagrange multiplier as follows:

$$-\frac{1}{2} \sum_t \sum_s \alpha^t \alpha^s y^t y^s (x^t)^T x^s + \sum_t \alpha^t \tag{7}$$

Which is maximized with respect to α^t and the constraints are:

$$\sum_t \alpha^t y^t = 0 \text{ and } \alpha^t \geq 0, \forall \text{t}$$

Here one can solve for α^t to get the values of w_0. Only a small percentage of x^t have the values $\alpha^t > 0$ and these are nothing but support vectors. The w is written as a weighted sum of training samples which are selected as Support Vectors. From the above equation w_0 can be calculated as:

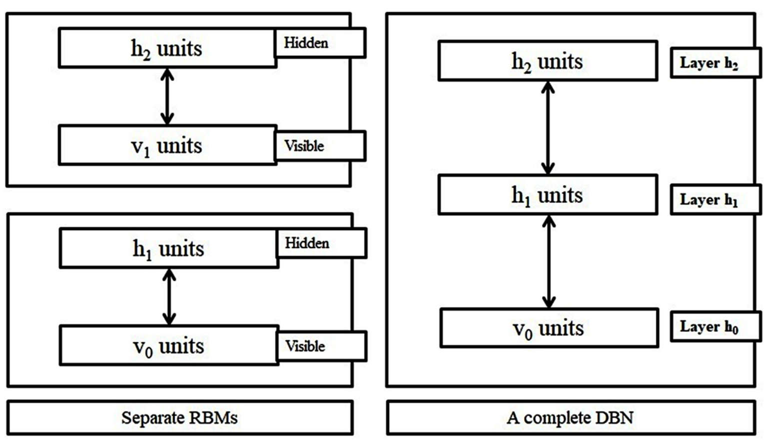
$$W_0 = y^t - W^T X^t \tag{8}$$

The calculation for w_0 is done for all support vectors and the average is taken. This discriminant calculation is called Support Vector Machine. Generally, data might be overlapped, and thus getting the accurate training data division is a challenging aspect, which could lead to least generalization.

Deep Belief Networks

A DBN is a graphical model with multiple layers constructed by RBM. If multiple RBM is stacked then it is possible to train such network layer by layer using a greedy approach. The network with such training is not a multilayer BM but it is a generative model called as Deep Belief Network.

Figure 7. Combination of two RBM to form a DBN structure



A DBN is composed of layers with both directed and undirected connections. Similar to RBM, the DBN also do not have communication between the nodes of the same layer. But a DBN has multiple hidden layers which are connected to neigboring hidden layers. The hidden layers are generally binary but visible layer can be binary or real. Figure 7 shows how the two RBM units can be combined to form a 3 Layer DBN structure. In the model, PCA is used to reduce the number of attributes. The DBN is constructed then with 2 layers of RBM. The input layer has 3 neurons. The RBM layer is constructed with 3 neurons each.

The DBN requires a conditional probability distribution which is taken directly from the conditional probability distribution of component RBMs. The joint probability distribution of DBN can be represented as follows:

$$P\left(x,h^{1},\dots h^{l}\right)=\prod_{k=0}^{l-2}P\left(h^{k}\mid h^{k+1}\right)P\left(h^{l-2},h^{l}\right) \tag{9}$$

Where $x=h^{0},P(h^{k-1}\mid h^{k})$ represents the conditional distribution for the visible layer units conditioned on the hidden units of the RBM at level *k*, and $P\left(h^{l-1},h^{l}\right)$ is the visible to hidden joint distribution in the top-level RBM.

The training of DBN is done using a greedy approach of layer by layer training. The process can be shortly summarized as follows:

1. Training starts at the first layer, training it like the first layer of RBM. This layer of RBM models the raw input $x=h^{(0)}$ as the visible layer.
2. The representation of input obtained from the first layer is used as data for the second layer of RBM. One can calculate as mean activation or as a sample. Mean activation can be calculated as $p\left(h^{(1)}=1\mid h^{(0)}\right)$ and samples can be chosen as $\left(h^{(1)}\mid h^{(0)}\right)$.
3. After this, the second layer is trained again as RBM. Here the samples or mean activations are used as sample training data.
4. Repeat step (2) and (3) for the layers which are present in DBN.
5. The results obtained can be fine-tuned by using supervised training criteria so that this can be used for giving predictions or classification.

After training the RBM using CD the last layer of Multi-Layer Perceptron (MLP) is added to predict the class label. The DBN can be used as a generative model but the objective is to improve the classification. So, adding a classifier layer at the end helps in improving the overall DL model. There are many choices of layers which can be added as the last layer for classification in DBN like Softmax layer or Logistic Regression Layer. But the structure of RBM is basically a structure inherited from ANN so the choice of MLP is most suitable for the data in hand. The MLP layer is constructed with 3 neurons in the layer and one output neuron for predicting the class value. Logistic Regression is used as an activation function for the MLP layer.

The weights from the DBN can be used to define the MLP as:

$$h^{(1)}=Sigm\left(b^{(1)}+v^{T}W^{(1)}\right) \tag{10}$$

$$h^{(1)}=Sigm\left(b_{i}^{(i)}+h^{(l-1)T}W^{(l)}\right)\forall l\ \ 2,\dots m \tag{11}$$

Once the DBN is trained it will generate weights and biases which can be used by MLP to perform classification.

Dataset

Population refers to the total number of items for which information is desired. The population is that group of individuals that have similar characteristics which are of interest to the researcher. To predict the performance of students, the private engineering college students are decided as population. The engineering colleges under Mumbai University are selected as population.

There are more than 50+ engineering colleges under Mumbai University. As the population is too large, sampling is required to collect data. Mumbai University consists of the engineering colleges from Mumbai, New Mumbai and Thane region. So at the first stage, it was decided to collect samples from Mumbai. At the second stage, it was decided to concentrate on the geographical centre part of Mumbai. The samples are collected from engineering colleges which are centrally located in Mumbai City. Data collected here is Primary data and data collection is done using a questionnaire. Data collection through the questionnaire is the most popular method in case of big enquiries.

Various parameters are identified which have a direct or indirect effect on the performance of students. Careful selection of questions was important while keeping in mind that the questionnaire does

Figure 8. Cognitive Parameters

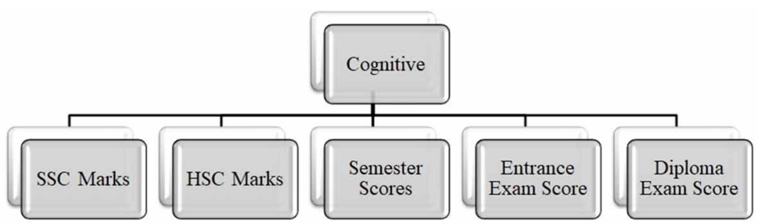
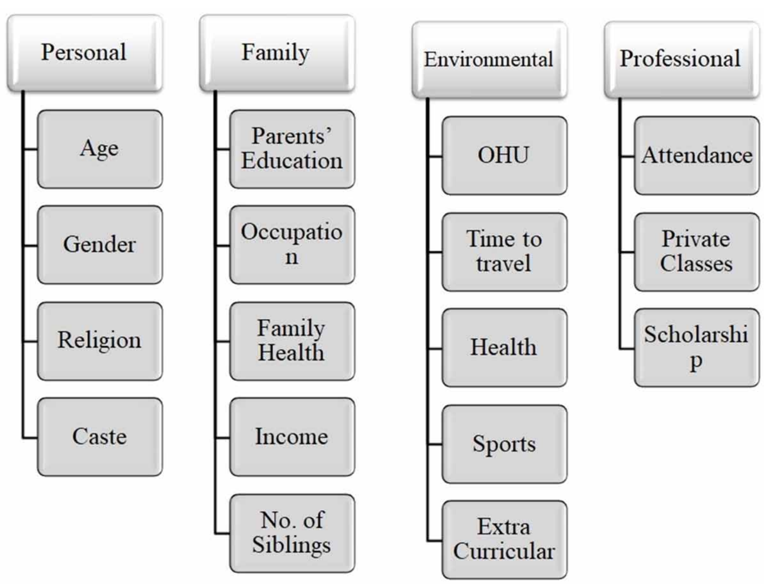


Figure 9. Non-cognitive Parameters



not pose a burden on respondents. The cognitive and non-cognitive parameters are identified. The effect of cognitive and non-cognitive parameters in performance prediction is carefully understood by studying various articles. The parameters identified are shown below:

Cognitive factors are the characteristic of a person which affect the performance and learning directly. These factors are measurable. Non-cognitive factors are the parameters which are not directly linked to but may have an effect on the performance and learning. Studies have shown that non-cognitive parameters have an equivalent effect on performance and learning. Non-cognitive factors are not directly measurable. Keeping in mind the scenario of engineering students and colleges, few non-cognitive factors which may have an indirect effect on the performance of students are decided.

Based on the parameters identified the class label is decided based on CGPA (Cumulative Grade Point Average) score of 5^{th} semester. The parameter 'class' indicates the CGPA score of the student in Semester 5. The CGPA score is calculated on the scale of 1 to 10. This parameter indicates the performance of the student in the coming semester. The implemented system predicts the performance of the student as a CGPA score range.

Table 1. Output class label

Class (CGPA Score)	Class Label
<5	1
Bet 5 and 7	2
Bet 7 and 9	3
More than 9	4

There are 6% and 8% samples out of total samples in class 1 and 4 respectively. There are 36% and 50% samples out of total samples in class 3 and 4 respectively.

Evaluation Measures

To evaluate the effectiveness of the Machine Learning algorithms basic measures like Accuracy, Precision, Recall and F1-Measure (Han & Kamber, 2012) were adopted. Squared error based cost functions are inconsistent for solving classification problems. Also, these measures are widely used in domains such as information retrieval, machine learning and other domains that involve classification (Olson & & Delen, 2008). A confusion matrix is a base for the determination of these measures.

Confusion Matrix

The confusion matrix can be represented as follows:

Accuracy

Accuracy indicates the closeness of a predicted or classified value to its real value. The state of being correct is called Accuracy. It can be calculated as:

Table 2. Confusion Matrix

		Predicted/Classified	
		Negative	**Positive**
Actual	**Negative**	True Negative (TN)	False Positive (FP)
	Positive	False Negative (FN)	True Positive (TP)

Where –
True Positive (TP) = Number of positive instances correctly classified as positive.
False Positive (FP) = Number of positive instances incorrectly classified as negative.
True Negative (TN) = Number of negative instances correctly classified as negative.
False Negative (FN) = Number of negative instances incorrectly classified as positive

Accuracy= (TP+TN)/(TP+TN+FP+FN)

Precision

Precision can be defined as the number of relevant items selected out of the total number of items selected. It represents the probability that an item is relevant. It can be calculated as:

Precision = TP/(FP+TP)

Precision is the measure of exactness.

Recall

The Recall can be defined as the ratio of relevant items selected to relevant items available. The recall represents a probability that a relevant item is selected. It can be calculated as:

Recall = TP/(FN+TP)

The recall is the measure of completeness.

F1-Measure

F1-Measure is the harmonic mean between Precision and Recall as described below:

F1-Measure= 2 * (Precision * Recall) / (Precision +Recall)

It creates a balance between precision and recall. Accuracy may be affected by class imbalance but F1 Measure is not affected by class imbalance. So with accuracy F1-measure is also used for evaluation of classification algorithms.

Sensitivity

Sensitivity is used to find out the proportion of positive samples that are correctly identified also called a true positive rate. It is calculated as:

Sensitivity=TP/P

Where,

P = Total Number of Positive Samples
N = Total number of Negative Samples

Specificity

Specificity is used to find out the proportion of negative samples that are correctly identified and also called a true negative rate. It is calculated as:

Specificity=TN/N

False Positive Rate (FPR)

FPR is used to find out the proportion of negative samples that are misclassified as positive samples. It is calculated as:

FPR=FP/N

False Negative Rate (FNR)

FNR is used to find out the proportion of positive samples which are misclassified as negative samples. It is calculated as:

FNR=FN/P

Negative Predictive Value (NPV)

NPV is used to find out the number of samples which are true negative. It is calculated as:

NPV=TN/(TN+FN)

False Discovery Rate (FDR)

FDR is also called an error rate. It is used to find out a proportion of false positive among all the samples that are classified as positive. It is calculated as:

FDR=FP/(FP+TP)

Matthews's Correlation Coefficient (MCC)

It is calculated as:

MCC=(TP*TN)-(FP*FN) /SQRT((TP+FP)(TP+FN)(TN+FP)(TN+FN))

MCC is a balanced measure based on a confusion matrix. This measure is used even if the classes are of different sizes. It is a correlation coefficient between the actual classes and predicted classes. The value of MCC lies between -1 to 1. The value near to +1 indicates the prediction is perfect. The value 0 indicates random prediction. The value -1 indicates a total disagreement between the actual and predicted values. MCC score above zero indicates balanced classification. MCC is a good measure when the data have varying classes, unbalanced dataset and random data (Jurman, Riccadonna, & Furlanello, 2012). With F1-score the MCC guides in a better way to determine the suitable algorithm for classification.

Results

The algorithms chosen for the study are evaluated on the collected dataset. To test the algorithms the dataset is required to be divided into training and testing dataset. Table 1 shows the results of the implementation.

Discussion

From the results, it is evident that Pure deep learning model with DBN is giving better results than the other two models. Also for some parameters, like Accuracy and Precision, SVM and DBN are giving similar results. So Deep Learning model is behaving similar to other machine learning algorithms. The MCC score of the DBN is better than the other two algorithms. Also, MCC score of SVM and DBN suggest that these classification models are relevant to the considered data.

From the results, it is evident that the hybrid model may yield better results than the standard model. Also, the MCC score and F1-score can improve if more related models like SVM and DBN are combined for classification. This gave the motivation to create and evaluate a combined model based on SVM and DBN.

Table 3. Evaluation Measures of different algorithms on the dataset

Measure	Algorithm		
	NN	SVM	DBN
Specificity	0.27	0.48	0.84
Sensitivity	0.08	0.18	0.78
Accuracy	0.4	0.65	0.7
Precision	0.29	0.36	0.38
FPR	0.83	0.83	0.23
FNR	0.74	0.53	0.17
NPV	0.08	0.18	0.78
FDR	0.72	0.65	0.63
F1_Score	0.28	0.41	0.54
MCC	0.02	0.04	0.09

SOLUTIONS AND RECOMMENDATIONS

Pure DL based models can give the same performance as the advanced ML models. In such cases, it is always interesting to investigate the combinatory models to find out the classification experience. Combinatory models may give better performance as compared to pure generative models.

After evaluating the performance of the Machine Learning algorithms including Deep Learning, a combinatorial model is implemented using Principal Component Analysis (PCA), SVM and DBN.

Algorithms

Dimensionality Reduction Using Principal Component Analysis (PCA)

In the proposed prediction model, PCA (Maćkiewicz & Ratajczak, 1993) is used for reducing the vast data. There are many reasons why one wishes to reduce the dimensionality of input data. Many times the complexity of the model depends on the number of dimensions in the data as well as the size of the data sample. When the dimensions are reduced then the cost related to extracting the not required dimensions is reduced. Many times the models are robust when the dataset is small and gives accurate results. Also, data with few dimensions can be visualized properly to reduce the outliers.

Consider a p-dimensional random variable U with the dispersion matrix $\sum$ and let λ_1 .. λ_n be the eigenvalues. Consider that $P_1 \ldots P_n$ are the corresponding Eigenvectors of $\sum$. Then one can write:

$$\Sigma = \lambda_1 P_1 P_1^{'} + \ldots + \lambda_p P_p P_p^{'} \tag{12}$$

$$\Sigma = P_1 P_1^{'} + \ldots + P_p P_p^{'} \tag{13}$$

$$P_1^{'}\Sigma P_i = \lambda_i\,,\ P_1^{'}\Sigma P_j = 0\,, \text{i}\neq\text{j} \tag{14}$$

The transformed random variables can be represented as:

$$Y_i = P_1^{'}U, i = 1,\ldots.P \tag{15}$$

Here Y is the new random variable vector and P is the orthogonal matrix then Y can be obtained from U by the orthogonal transformation as Y=PU. Here this random variable vector Y_i is called as the i^{th} principal component of U.

Only the basic steps of PCA are followed here. These basic steps of PCA are given in Algorithm 1.

Algorithm 1: Steps of PCA

Step 1 Standardize the input data
Step 2 Evaluate the covariance of the data
Step 3 Deduce Eigenvectors and Eigenvalues
Step 4 Re-orient data with respect to Principal Components
Step 5 Plot re-oriented data
Step 6 Bi-plot

These steps can be explained as follows:

- Standardization or normalization of input data is important for the working of PCA. This normalization is done by subtracting the means from the corresponding columns.
- Next, the covariance is calculated. A covariance matrix is a matrix whose (i,j) th value represents the covariance between i^{th} row and j^{th} column.
- Now the Eigenvectors and Eigenvalues are deduced. There are various ways to do that but for the large data analysis, the popular NIPALS (Nonlinear Iterative Partial Least Squares) algorithm is used. It is a simple and efficient way to deduce the principal components from the input data. One advantage of using the NIPALS algorithm is that it can be modelled to handle missing data. This algorithm is very efficient on large datasets.
- Once the principal components are found out then these components are sorted in the order from strong component to weak component. The weak components can be ignored thereby reducing the dimensions. Using the strong principal components it is possible to construct a strong representation of data.
- The new input data is plotted or reconstructed using the important features extracted based on the values of Eigenvectors. The Eigenvectors are arranged in the order of decreasing values.

Support Vector Machine (SVM)

PCA acts as dimensionality reduction techniques. The features are given as an input to the PCA and reduced extracted features are considered for further computation. If there are 12 features then PCA reduces it to 6 features and so on.The reduced dimension and class labels are given to the SVM for prediction of the class. The SVM here will get reduced dimensions to work on. As the model is working on reduced but important data the predictions are more accurate.

The data considered here consists of 35 features and one class label. Providing these many features directly to DBN makes it computationally intensive as well results may not be so accurate. So intermittently SVM is used to generate near accurate class labels which are fed to the DBN. SVM with a linear kernel is used to generate the class labels.Here the tuning of SVM is obviously not so accurate with the resultant prediction (in which class the performance fall). Hence, the resultant class labels from SVM are considered as the features to DBN classifier. DBN classifier classifies the students' overall performance.

Deep Belief Network (DBN)

Generally, DBN includes multiple layers, and each and every layer has visible neurons, which establish the input layer, and hidden neurons form the output layer. Further, there presents a deep connection with hidden and input neurons; but there was no connection among hidden neurons and no connections are present in the visible neurons. The connection among visible as well as hidden neurons is symmetric and exclusive. This corresponding neuron model defines an accurate output for the input.

Since the stochastic neurons' output in Boltzmann network is probabilistic, Eq. (16) denotes the output and Eq. (17) specifies the possibility in sigmoid-shaped function, where t^P indicates the pseudo-temperature. The deterministic model of the stochastic approach is given in Eq. (18).

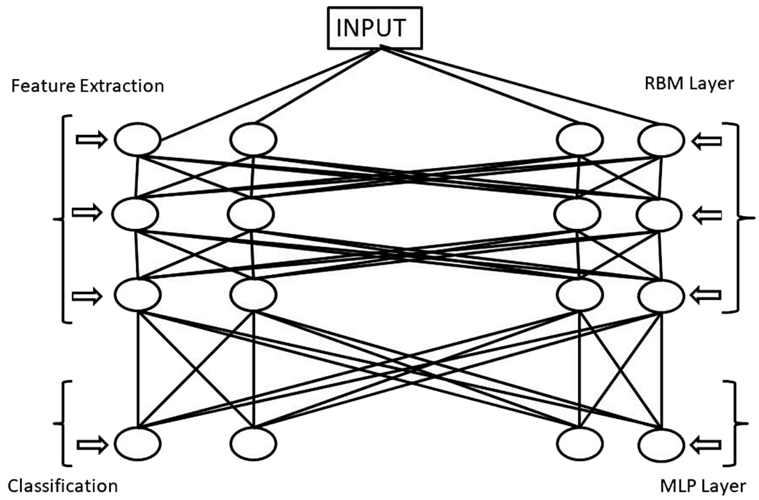
$$P_q\left(\zeta\right) = \frac{1}{1 + e^{\frac{-\zeta}{t^p}}} \tag{16}$$

$$PO = \begin{cases} 1, & with\, 1 - \overline{P}_q\left(\zeta\right) \\ 0, & with\, \overline{P}_q\left(\zeta\right) \end{cases} \tag{17}$$

$$\lim_{t^p \to 0^+} \overline{P}\left(\zeta\right) = \lim_{t^p \to 0^+} \frac{1}{1 + e^{\frac{-\zeta}{t^P}}} = \begin{cases} 0\, for\, \zeta < 0 \\ \frac{1}{2}\, for\, \zeta = 0 \\ 1\, for\, \zeta > 0 \end{cases} \tag{18}$$

The diagrammatic representation of the DBN model is in Figure 10, in which the process of feature extraction takes place through a set of RBM layers and the process of classification takes place via MLP. The arithmetic model exposes the energy of Boltzmann machine for the creation of neuron or binary state b_i, and that is defined in Eq. (19), where $W_{a,l}$ indicates the weights among neurons and θ_a indicates the biases.

Figure 10. Architecture of DBN in the proposed model



$$\Delta EN\left(bi_a\right) = W_{a,l} + \theta_a \tag{19}$$

The progression of energy in terms of the joint composition of visible as well as hidden neurons $\left(x, y\right)$ is defined in Eq. (20), Eq. (21) and Eq. (22). In this, x_a indicates either the binary or neuron state of a visible unit, B_l indicates the binary state of l hidden unit, and k_a is constant.

$$EN\left(x, y\right) = \sum_{(a,l)} W_{a,l} x_a y_l - \sum_a k_a x_a - \sum_l B_l y_a \tag{20}$$

$$\Delta EN\left(x_a, \overline{y}\right) = \sum_l W_{al} y_l + k_a \tag{21}$$

$$\Delta EN\left(\vec{x}, y_a\right) = \sum_l W_{al} x_a + B_l \tag{22}$$

The input data's possibility dissemination is encoded into weight (parameters), which is spread as RBM's learning pattern. RBM training can attain distributed possibilities, and the consequent weight assignment is defined by Eq. (23).

$$\hat{W}_{(\hat{M})} = \max_{W} \prod_{x \in N} C\left(\vec{x}\right) \tag{23}$$

For the visible and hidden vectors pair $\left(\vec{x},\overrightarrow{hi}\right)$, the possibility assigned RBM approach is given in Eq. (24), where PR^F specifies the partition function as in Eq. (25).

$$c\left(\vec{x},\overrightarrow{hi}\right)=\frac{i}{PR^F}e^{-EN\left(\vec{x},\vec{y}\right)} \tag{24}$$

$$PR^F=\sum_{\vec{x},\vec{y}}e^{-EN\left(\vec{x},\vec{y}\right)} \tag{25}$$

The DBN is trained using CD (Contrastive Divergence) (Goodfellow, Bengio, & Courville, 2016) algorithm. The steps of the CD training are as follows:

Step 1: Choose the *x* training samples and brace it into visible neurons.
Step 2: Evaluate the feasibility of hidden neurons c_y by identifying the product of $\hat{W}$ weight matrix and visible vector
Step 3: Examine the *y* hidden states from c_y probabilities.
Step 4: Evaluates the *x* exterior product of vectors and c_y that is measured as a positive gradient $\varnothing^+ = x \cdot c_y^{t^P}$.
Step 5: Examine the reconstruction of x' visible states from *y* hidden states. Further, it is needed to evaluate y' hidden states from the reconstruction of x'.
Step 6: Evaluate the x' and y''s exterior product, be it as a negative gradient $\varnothing^- = x' \cdot y'^{t^P}$.
Step 7: Define the updated weight as defined in Eq. (26), where � indicates the learning rate.

$$\Delta\hat{W}=\eta(\varnothing^+-\varnothing^-) \tag{26}$$

Step 8: Update the weights with new values.

The following step defines the progression of DBN training with MLP training (normal) and RBM training (pre-training)

Step 1: Initialize the DBN model with weights, biases and further associated parameters, which are randomly selected.
Step 2: Firstly, the initialization of RBM model is progressed with the input data that serves the potentials in its visible neurons and gives the unsupervised learning.
Step 3: Here, the input to the subsequent layer is subjected by potential sampling that processed in the hidden neurons of the preceding layer. Further, it follows the unsupervised learning.
Step 4: The above-specified steps are continued for the corresponding count of layers. Hence, the pre-training stage by RBM is processed till it reaches the MLP layer.

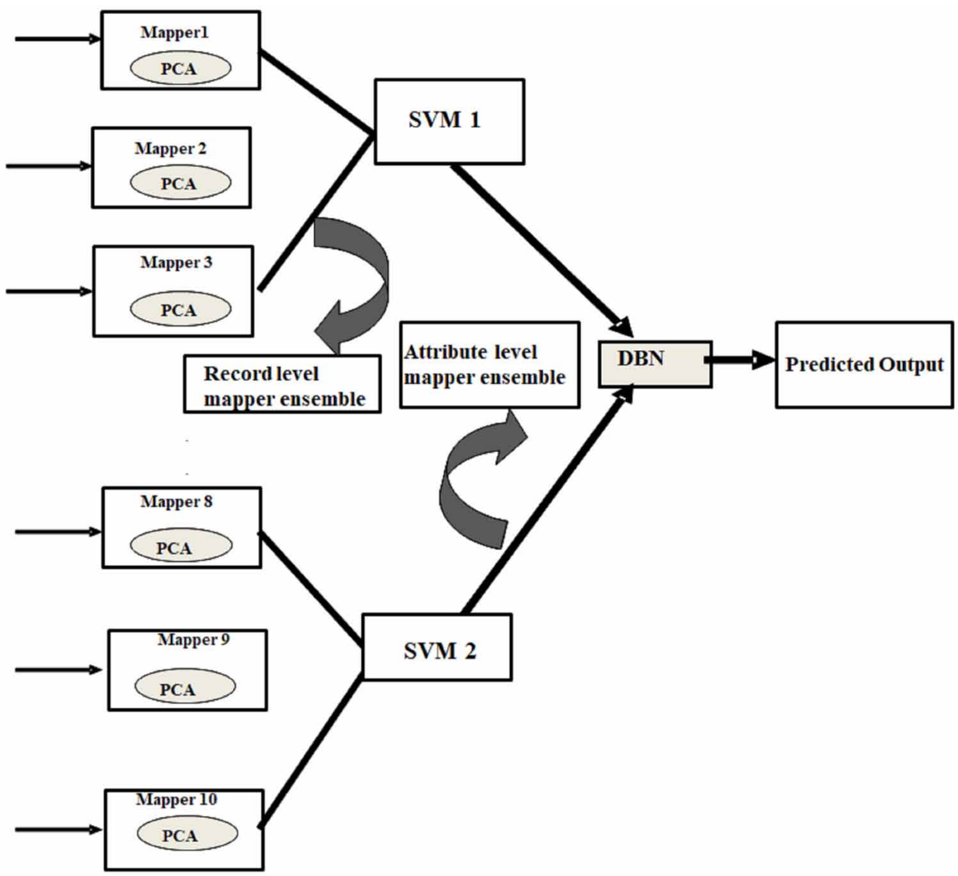
Step 5: MLP phase specifies the attained learning by supervised format and is continued till it attains the target error rate.

Finally, the classifier predicts the students' performance with increased accuracy rate. The predictions are evaluated on the basis of various evaluation measures identified.

System Model

Figure 11 shows the improved hybrid model.

Figure 11. System Model



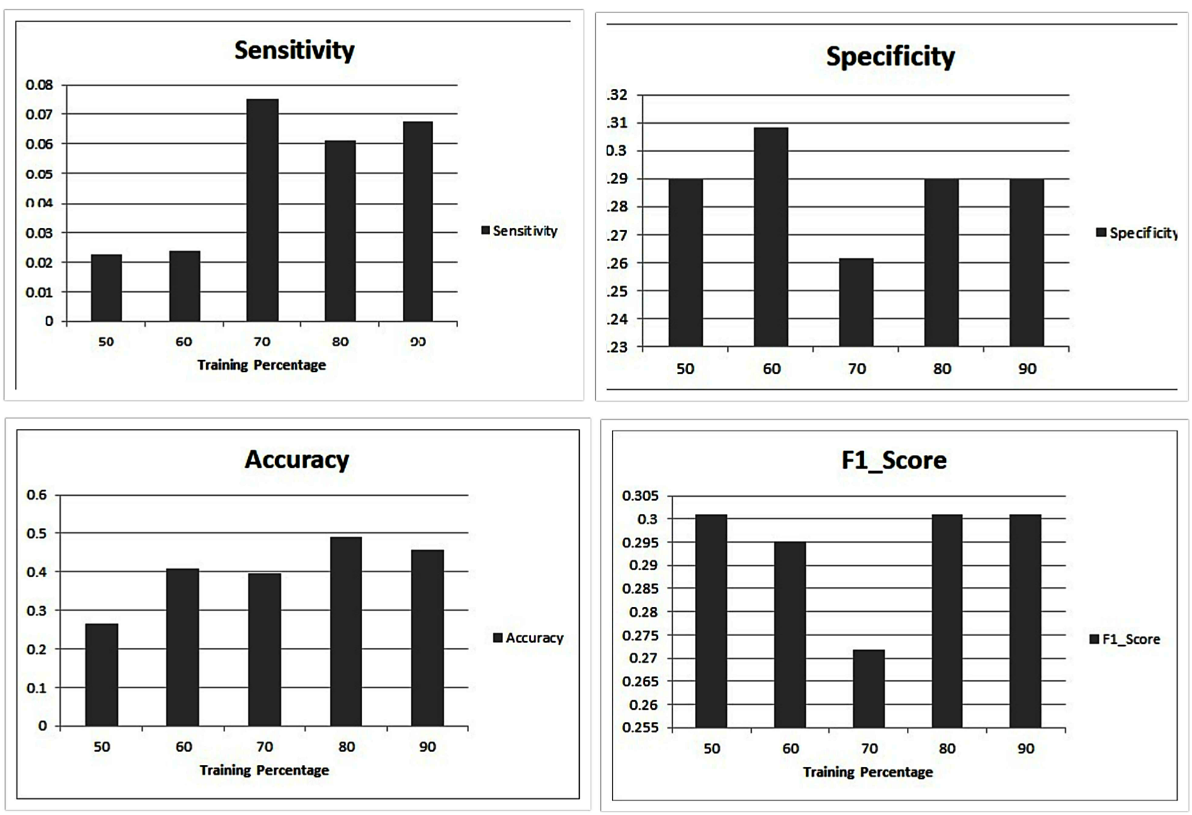
The input parameters are split and are fed to PCA to reduce parameters. The output from each PCA is given to the individual SVM for prediction of class label. The class labels predicted by the SVM; acts as the input to the DBN. Using this input and actual class labels, the DBN predicts the classes for the data. The DBN is constructed with 2 layers of RBM. One layer of RBM represents a hidden layer and a visible layer. The RBM layers are constructed with 3 neurons each and the activation function used is a sigmoid function. The numbers of input neurons are 3. After RBM layers an MLP layer is added for prediction of class. The MLP layer has 3 neurons and logistic regression is used as an activation function. The output layer has one neuron to predict the class label.

The model follows a parallel framework. If the number of features increases in future then more PCA and SVM components can be introduced. Vertical fragmentation suggests model can be easily adapted in the Map Reduce framework (Maitrey & C.K.Jha, 2015) for Big Data processing. As well as horizontal fragmentation is also done to suggest the suitability for Big Data application. Here the horizontal fragmentation may give multiple calls to single PCA block.

Results

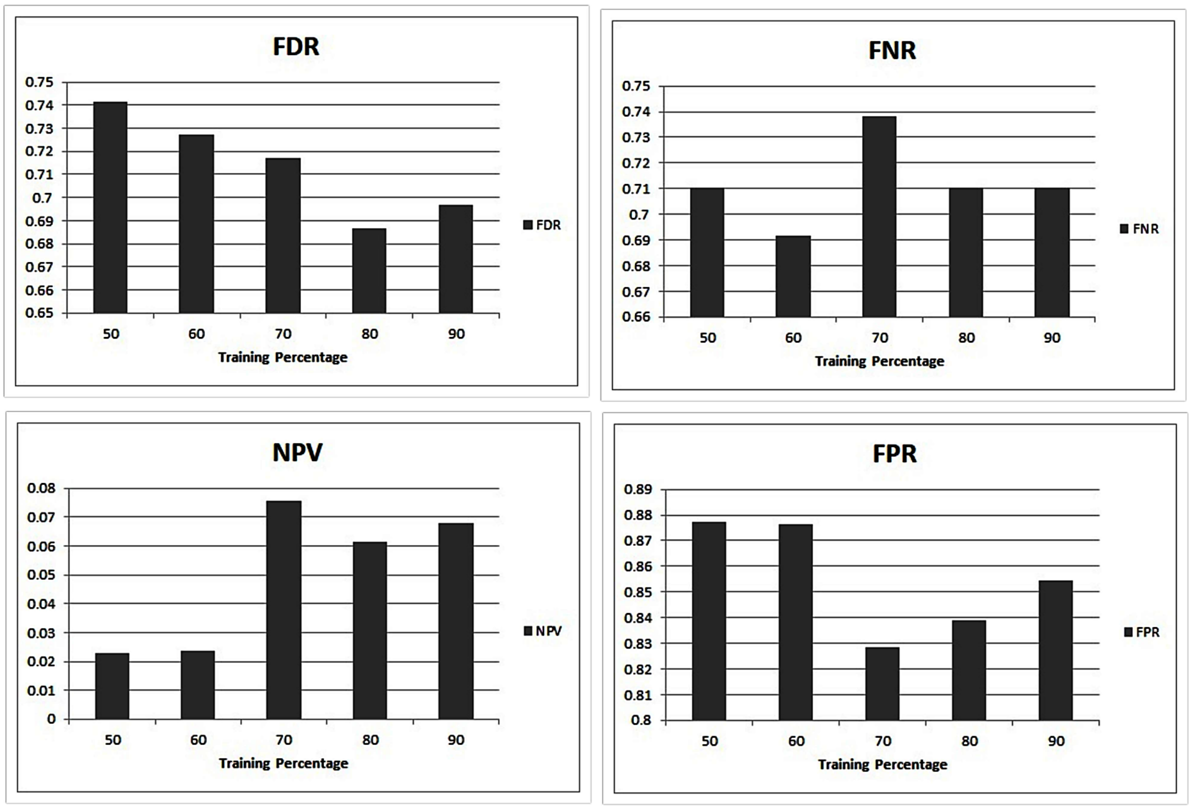
The results for various evaluation measures for the various training percentages are indicated in figure 12, 13 and 14

Figure 12. Results for Proposed Model: Sensitivity, Specificity, Accuracy and F1_Score



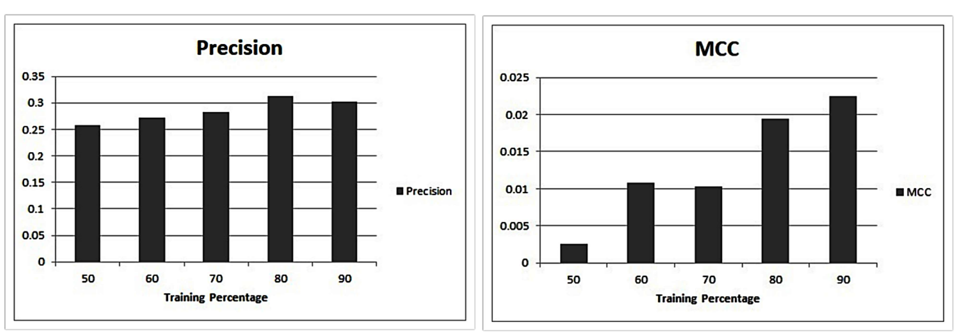
The specificity of the proposed model remains almost same to 0.85 for all training percentages. The sensitivity score is 0.80. The precision is also almost constant to 0.78 and is increased for 60% training. The NPV score is good with an average value of 0.80. The accuracy graph shows a variation from 69% to 75% for different training percentages. The accuracy is good with 50% training. The accuracy has

Figure 13. Results for Proposed Model: FDR, FNR, NPV and FPR



improved and is better than pure SVM and DBN for the considered data. The F1-score is with a value of 0.81. The MCC score shows a variation with the values ranging from 0.13 to 0.18. The value of MCC score is far better than the MCC scores of SVM, NN and DBN. The good and positive MCC score suggests that the proposed model is better suited for the data under consideration. There is a considerable improvement in MCC score indicating the suitability of the model for the educational data.

Figure 14. Results for Proposed Model: Precision and MCC



It is important to understand if the proposed model is better than other models. It is necessary to look at evaluation measures to find the performance and suitability of the proposed classification method for the collected educational data.

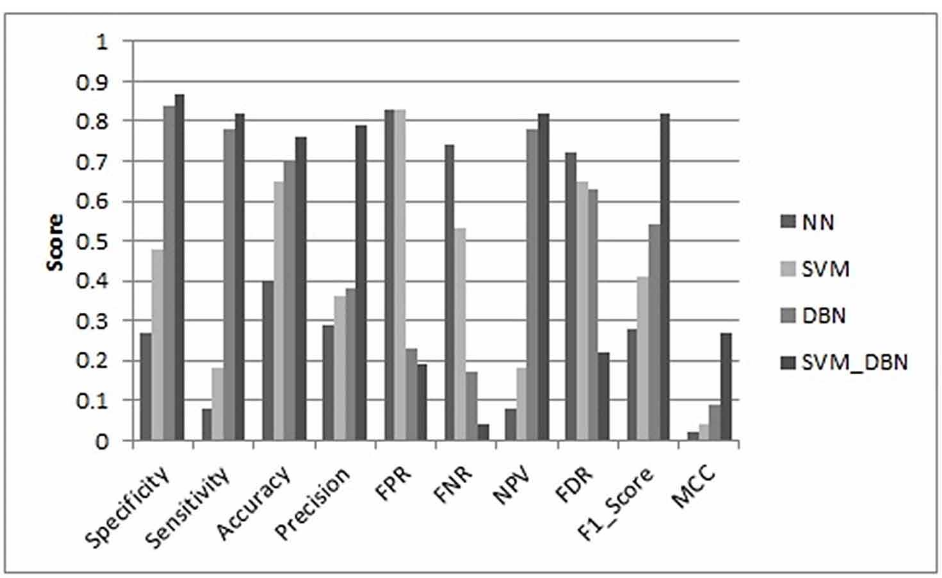
Table 4 shows the overall performance of the proposed hybrid classification model over other models.

Table 4. Overall Performance of proposed classification model over other methods

Measures	Algorithms			
	NN	SVM	DBN	SVM With DBN
Specificity	0.27	0.48	0.84	0.87
Sensitivity	0.08	0.18	0.78	0.82
Accuracy	0.4	0.65	0.7	0.76
Precision	0.29	0.36	0.38	0.79
FPR	0.83	0.83	0.23	0.19
FNR	0.74	0.53	0.17	0.04
NPV	0.08	0.18	0.78	0.82
FDR	0.72	0.65	0.63	0.22
F1- Score	0.28	0.41	0.54	0.82
MCC	0.02	0.04	0.09	0.27

From this, it is observed that the proposed prediction model is more superior to other methods with respect to all measures. Particularly, the specificity of proposed SVM with Deep Learning model is bet-

Figure 15. Performance of Proposed model over other algorithms



ter from DBN, SVM and NN. The accuracy of the proposed model is 5.06% superior to DBN, 6.69% superior to SVM and 46.87% superior to NN. The proposed model also attained great precision over other methods. Similarly, the FPR of the proposed model is 2.26%, 76.62%, and 76.70% better from DBN, SVM, and NN, respectively with less FPR.

The FNR of the proposed model is very low than other methods. The proposed model attained high NPV, and it is 0.59% better from DBN, 84.12% better than SVM and 96.30% better than NN. FDR of the proposed method is also very low, and it is 63.13%, 65.24% and 67.22% enhanced than DBN, SVM, and NN, respectively. Then the F_1-Score of the proposed method is 49.27%, 59.64%, and 60.09% better from DBN, SVM, and NN, respectively. From this analysis, it is proved that the proposed prediction model is highly efficient when compared to other conventional methods.

The Graph in figure 15 shows the overall performance of the proposed model. The proposed model has better accuracy, F1 score and MCC indicating that the proposed hybrid model created using SVM and DBN is able to classify the educational data in a better way.

Discussion

Table 5 (a), (b), (c) and (d) shows the performance score of the proposed model for evaluation parameters Accuracy, F1 Measure, FPR and MCC for different classes. The training percentage is 60%.

Table 5a. Scores of Accuracy for Different Classes for Proposed Model

Class	Algorithm			
	NN	SVM	DBN	SVM-DBN
1	0.12	0.47	0.77	0.77
2	0.12	0.36	0.76	0.8
3	0.12	0.37	0.79	0.79
4	0.12	0.15	0.37	0.43

Table 5b. Scores of FPR for Different Classes for Proposed Model

Class	Algorithm			
	NN	SVM	DBN	SVM-DBN
1	1	0.75	0.09	0.01
2	1	0.84	0.09	0.01
3	1	0.88	0.09	0.03
4	0.88	0.09	0.01	0.01

Accuracy for the various classes is improved drastically for the hybrid model, mainly for the classes where data samples are less. Even F1 score and MCC score is better of the hybrid model than other models. Low FPR indicates that prediction of classes by the hybrid model is improving through the data is imbalanced.

Table 5c. F1-Score for Different Classes for Proposed Model

Class	Algorithm			
	NN	SVM	DBN	SVM-DBN
1	0.03	0.5	0.58	0.5
2	0.03	0.49	0.5	0.5
3	0.03	0.5	0.53	0.5
4	0	0.03	0.5	0.5

Table 5d. Scores of MCC for Different Classes for Proposed Model

Class	Algorithm			
	NN	SVM	DBN	SVM-DBN
1	0.04	0.05	0.17	0.19
2	0.03	0.05	0.18	0.18
3	0.01	0.06	0.17	0.21
4	0.05	0.13	0.17	0.29

The accuracy is increased to 75%. Still, there is scope for improvement in the model to achieve better accuracy. The model can be further optimised to gain better accuracy by improving the training. The scores of evaluation measures for various training percentages indicated that the model can be improved with improved training experience.

FUTURE RESEARCH DIRECTIONS

The chapter represents one of the ways to analyse the Educational Data using a hybrid model. There are many other ways to work in the area of EDM and DL together. Some improvement in the DL model is also beneficial to improve the accuracy of prediction in Educational Domain.

The model considered here is a hybrid model for classification using SVM – DBN. The model is used for performance prediction. There are many other tasks in EDM like course recommendation where such hybrid models may be effective. SVM and DBN being generative models are applicable in many domains.The performance of the DBN can be further improved if training is improved. There are many optimization techniques which can be combined to improve the training of the DBN. It is interesting to find out if optimised training increases the accuracy of the model.

CONCLUSION

A Deep Learning model for the Performance Prediction of Students in Educational Information System is implemented. The work started with the motivation to implement a hybrid model with better accuracy in Educational Domain. Educational Data Mining (EDM) is a popular research area which focuses on finding various tools and techniques to analyse students' data. As the data is increasing tremendously in educational systems, analysis is becoming a challenge. Many traditional and ML algorithms are applied in EDM but the accuracy of the algorithms is less.

Machine Learning and Deep Learning are the fields of Artificial Intelligence where algorithms learn by themselves. These algorithms can be applied in many emerging areas where they may be effective. These algorithms are found useful in many areas with an increase in data size. The main aim of using DL model is to increase accuracy. Before devising the hybrid model, ML algorithms are applied to the data collected from the educational domain. The ML algorithms like NN, SVM and pure DL algorithm - DBN are applied on the collected data. Additional evaluation measures are used to test the algorithms. Balanced evaluation measure like MCC particular for the ML domain is used with traditional F1-score. The results show that pure DL and advanced ML algorithms are giving similar accuracy. Hence a new hybrid model for performance prediction of students to get better accuracy is implemented

A new students' performance prediction hybrid model is proposed. It uses a new hybridized classifier – SVM and DBN - to predict the performance. The data is trained by SVM and the resultant class labels from SVM are considered as the features to DBN, where it has classified the performance. The performance of the proposed prediction model is compared to other conventional methods. The results clearly indicated that proposed prediction model is better than advanced ML and pure DL algorithms.

REFERENCES

Alpaydin, E. (2004). *Introduction to Machine Learning*. Cambridge: MIT Press.

Alwaisi, S., & Baykan, O. K. (2017). Training Of Artificial Neural Network Using Metaheuristic Algorithm. *International Journal of Intelligent Systems and Applications in Engineering*.

Ashtari, S., & Eydgahi, A. (2017). Student perceptions of cloud applications effectiveness in higher education. *Journal of Computational Science*, *23*, 173–180. doi:10.1016/j.jocs.2016.12.007

Bengio, Y. (2009). Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, *2*(1), 1–127. doi:10.1561/2200000006

Buniyamin, N., Mat, U. b., & Arshad, P. M. (2015, November). Educational Data Mining for Prediction and Classification of Engineering Students Achievement. *IEEE 7th International Conference on Engineering Education (ICEED)*, (pp. 49-53). Kanazawa: IEEE.

Chen, X., Vorvoreanu, M., & Madhavan, K. (2013). Mining Social Media Data for Understanding Students' Learning Experiences. *IEEE Transactions on Learning Technologies*.

Costa, E., Fonseca, B., Santana, M. A., Araújo, F., & Rego, J. (2017). Evaluating the effectiveness of educational data mining techniques for early prediction of students' academic failure in introductory programming courses. *Computers in Human Behavior*, *73*, 247–256. doi:10.1016/j.chb.2017.01.047

Davidt, O. E., & Netanyahu, N. S. (2015, July). DeepSign: Deep Learning for Automatic Malware Detection. *International Joint Conference on Neural Networks (IJCNN)*, (pp. 1-8). , 978-1-4799-1959-8/1510.1109/IJCNN.2015.7280815

Davis, D. (1998). The virtual university: A learning university. *Journal of Workplace Learning*, *10*(4), 175–213. doi:10.1108/13665629810213935

Deng, L., & Yu, D. (2014). *Deep Learning for Signal and Information Processing*. Redmond, WA: Microsoft Research.

Deng, Y., Ren, Z., Kong, Y., Bao, F., & Dai, Q. (2017). A Hierarchical Fused Fuzzy Deep Neural Network for data classification. *IEEE Transactions on Fuzzy Systems*, *25*(4), 1006–1012. doi:10.1109/TFUZZ.2016.2574915

Gao, N., Gao, L., Gao, Q., & Wang, H. (2014, November). An Intrusion Detection Model Based on Deep Belief Networks. *Second International Conference on Advanced Cloud and Big Data*, (pp. 247-252). Huangshan: Academic Press.

Giannakos, M. N., Aalberg, T., Divitini, M., Jaccheri, L., Mikalef, P., Pappas, I. O., & Sindre, G. (2017, April). Identifying Dropout Factors in Information Technology Education: A Case Study. *IEEE Global Engineering Education Conference (EDUCON)*, (pp. 1187-1194). 10.1109/EDUCON.2017.7942999

Gobert, J. D., Kim, Y. J., Pedro, M. A., Kennedy, M., & Betts, C. G. (2015). Using educational data mining to assess students' skills at designing and conducting experiments within a complex systems microworld. *Thinking Skills and Creativity*, *18*, 81–90. doi:10.1016/j.tsc.2015.04.008

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Guarín, C., Guzmán, E., & González, F. (2015). A Model to Predict Low Academic Performance at a Specific Enrollment Using Data Mining. *IEEE Journal of Latin-American Learning Technologies, 10*(3).

Guo, B., Zhang, R., Guang, X., Shi, C., & Yang, L. (2015, July). Predicting Students performance in educational data mining. *International Symposium on Educational Technology (ISET)*, (pp. 125-128). 10.1109/ISET.2015.33

Han, J., & Kamber, M. (2012). *Data Mining: Concepts and Techniques* (3rd ed.). Morgan Kaufmann.

Hijazi, S. T., & Naqvi, S. R. (2006). Factors Affecting Students' Performance. *Bangladesh e-Journal of Sociology*, 3.

Jurman, G., Riccadonna, S., & Furlanello, C. (2012). A comparison of MCC and CEN error measures in multi-class prediction. *PLoS One*, *7*(8), e41882. doi:10.1371/journal.pone.0041882 PMID:22905111

Kim, S. M. L., & Shen, J. (2015, July). A novel deep learning by combining discriminative model with generative model. *International Joint Conference on Neural Networks (IJCNN)*, (pp. 1-6). 10.1109/IJCNN.2015.7280589

Koch, F., Assunção, M. D., Cardonha, C., & Netto, M. A. (2016). Optimising resource costs of cloud computing for education. *Future Generation Computer Systems*, *55*, 473–479. doi:10.1016/j.future.2015.03.013

Kuwata, K., & Shibasaki, R. (2015, July). Estimating crop yields with deep learning and remotely sensed data. *International Geoscience and Remote Sensing Symposium (IGARSS)*, (pp. 858-861). 10.1109/IGARSS.2015.7325900

Lauzon, F. Q. (2012). An introduction to deep learning. *International Conference on Information Science, Signal Processing and their Applications (ISSPA)*, (pp. 1438-1439). Montreal: Academic Press.

Maćkiewicz, A., & Ratajczak, W. (1993). Principal Components Analysis (PCA). *Computers & Geosciences*, *19*(3), 303–342. doi:10.1016/0098-3004(93)90090-R

Maitrey, S., & Jha, C. K. (2015). MapReduce: Simplified Data Analysis of Big Data. *Procedia Computer Science*, *57*, 563–571. doi:10.1016/j.procs.2015.07.392

Mondal, P. (n.d.). *7 Important Factors that May Affect the Learning Process*. Retrieved from http://www.yourarticlelibrary.com/learning/7-important-factors-that-may-affect-the-learning-process/6064/

Mushtaq, I., & Khan, S. N. (2012). Factors Affecting Students' Academic Performance. *Global Journal of Management and Business Research*, *12*(9).

Najafabadi, M. M., Villansutre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). *Deep learning applications and challenges in big data analytics*. Springer Open Journal of Big Data. doi:10.118640537-014-0007-7

Nguyen, K., Fookes, C., & Sridharan, S. (2015, September). Improving Deep Convolutional Neural Networks with Unsupervised Feature Learning. *IEEE International Conference on Image Processing (ICIP)*, (pp. 2270-2271). Quebec City: IEEE. 10.1109/ICIP.2015.7351206

Olson, D. L., & Delen, D. (2008). *Advanced data mining techniques* (1st ed.). Springer Publishing Company.

Park, J.-Y., Luo, H., & Kim, W. H. (2015,October). Factors affecting students' completion: A study of an online Master's program. *International Conference of Educational Innovation through Technology (EITT)*, (pp. 275-278). 10.1109/EITT.2015.33

Phung, L. T., Chau, V. T., & Phung, N. H. (2015, November). Extracting Rule RF in Educational Data Classification from a Random Forest to Interpretable Refined Rules. *International Conference on Advanced Computing and Applications (ACOMP)*, (pp. 20-27). 10.1109/ACOMP.2015.13

Pradeep, A., & Das, S., & J, J. (2015). Students Dropout Factor Prediction Using EDM Techniques. *International Conference on Soft-Computing and Network Security*. 10.1109/ICSNS.2015.7292372

Punlumjeak, W., & Rachburee, N. (2015, October). A Comparative Study of Feature Selection Techniques for Classify Student Performance. *7th International Conference on Information Technology and Electrical Engineering (ICITEE)*, (pp. 425-429). 10.1109/ICITEED.2015.7408984

Rajanna, A. R., Aryafar, K., Shokoufandeh, A., & Ptucha, R. (2015). Deep Neural Networks: A Case Study for Music Genere Classification. *14th International Conference on Machine Learning and Applications (ICMLA)*, (pp. 665-660). 10.1109/ICMLA.2015.160

Shiau, W.-L., & Chau, P. Y. (2016). Understanding behavioral intention to use a cloud computing classroom: A multiple model comparison approach. *Information & Management*, *53*(3), 355–365. doi:10.1016/j.im.2015.10.004

Shoukat, A. (2013). Factors Contributing to the Students' Academic Performance: A Case Study of Islamia University Sub-Campus. *American Journal of Educational Research*, 283–289.

Sukhija, K., Jindal, D. M., & Aggarwal, D. N. (2015, October). The Recent State of Educational Data Mining: A Survey and Future Visions. *IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education*, (pp. 354-359). doi: 10.1109/MITE.2015.7375344

Suryawan, A., & Putra, E. (2016). Analysis of Determining Factors for Successful Student's GPA Achievement. *11th International Conference on Knowledge, Information and Creativity Support Systems (KICSS)*, (pp. 1-7). Yogyakarta: Academic Press.

Tzortzis, G., & Likas, A. (2007, October). Deep Belief Networks for Spam Filtering. *IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)*, (pp. 306-309). Patras: Academic Press.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P.-A. (2010). Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *Journal of Machine Learning Research*, *11*, 3371–3408.

Vora, D., & Iyer, K. (2017). A Survey of Inferences from Deep Learning Algorithms. *Journal of Engineering and Applied Sciences, 12*(SI), 9467-9472.

Vora, D., & Kamatchi, R. (2018). EDM – Survey of Performance Factors and Algorithms Applied. *International Journal of Engineering & Technology, 7*(2.6), 93-97.

Vora, D. R., & Iyer, K. (2018, December). *Implementing a Deep Learning Approach for the Performnace Prediction in Educational Information System (*Unpublished Thesis). Amity University, Mumbai, India.

Wiering, M. A. M. S., Millea, A., Meijster, A., & Schomaker, L. (2016). *Deep Support Vector Machines for Regression Problems.* Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.718.987&rep=rep1&type=pdf

Xing, W., Guo, R., Petakovic, E., & Goggins, S. (2015). Participation-based student final performance prediction model through interpretable Genetic Programming: Integrating learning analytics, educational data mining and theory. *Computers in Human Behavior*, *47*, 168–181. doi:10.1016/j.chb.2014.09.034

Yuan, Y., Zhang, M., Luo, P., Ghassemlooy, Z., Lang, L., Wang, D., ... Han, D. (2017). SVM-based detection in visible light communications. *Optik (Stuttgart)*, *151*, 55–64. doi:10.1016/j.ijleo.2017.08.089

ADDITIONAL READING

Asif, R., Merceron, A., Ali, S. A., & Haider, N. (2017, October). Analyzing undergraduate students' performance using educational data mining. *Computers & Education*, *11*, 177–1943. doi:10.1016/j.compedu.2017.05.007

Fu, J., Chang, J., Huang, Y., & Chao, H. (2012). A Support Vector Regression-Based Prediction of Students' School Performance. *International Symposium on Computer, Consumer and Control*. 10.1109/IS3C.2012.31

Guo, X., Huang, H., & Zhang, J. (2014). Comparison of Different Variants of Restricted Boltzmann Machines. *2nd International Conference on Information Technology and Electronic Commerce(ICITEC)*. 10.1109/ICITEC.2014.7105610

KEY TERMS AND DEFINITIONS

Cognitive Factors: Characteristics of the student that have a direct effect on learning and performance of the student.

Data Mining: The process of extracting useful patterns from the data by following the systematic steps.

Educational Data Mining: Tools and techniques to extract meaningful patterns from educational data.

Non-Cognitive Factors: Characteristics of the student which do not have as such a direct effect on learning and performance but may have an indirect effect on performance and learning.

Predictive Analytics: Exploration of data to predict the future using various methods like statistics, ML, etc.

Chapter 11
Deep Learning Solutions for Agricultural and Farming Activities

Asha Gowda Karegowda
Siddaganga Institute of Technology, India

Devika G.
https://orcid.org/0000-0002-2509-2867
Government Engineering College, India

Geetha M.
Bharat Institute of Engineering and Technology, India

ABSTRACT

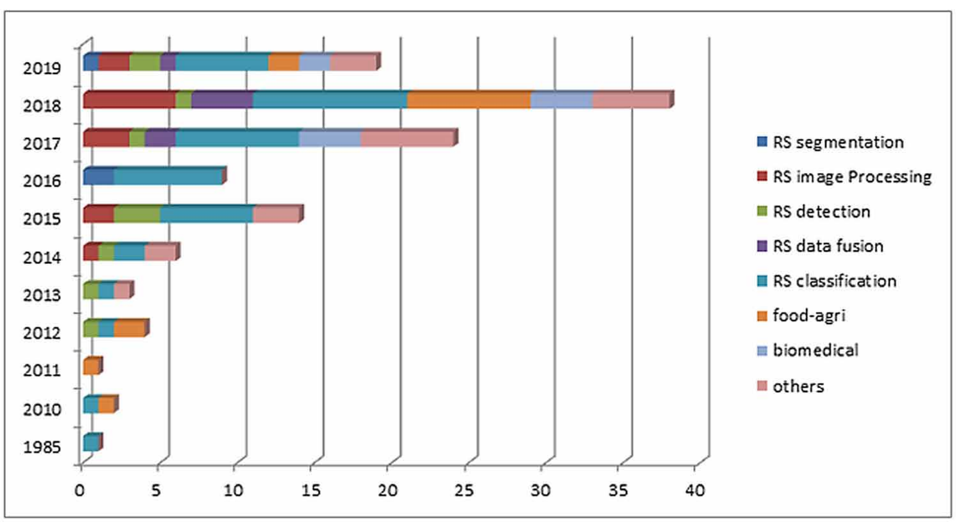
The continuously growing population throughout globe demands an ample food supply, which is one of foremost challenge of smart agriculture. Timely and precise identification of weeds, insects, and diseases in plants are necessary for increased crop yield to satisfy demand for sufficient food supply. With fewer experts in this field, there is a need to develop an automated system for predicting yield, detection of weeds, insects, and diseases in plants. In addition to plants, livestock such as cattle, pigs, and chickens also contribute as major food. Hence, livestock demands precision methods for reducing the mortality rate of livestock by identifying diseases in livestock. Deep learning is one of the upcoming technologies that when combined with image processing promises smart agriculture to be a reality. Various applications of DL for smart agriculture are covered.

DOI: 10.4018/978-1-7998-2108-3.ch011

INTRODUCTION

With the ever-increasing global population, smart agriculture emphasizes (i) enhancing agricultural productivity and quality of food and (ii) safeguarding the natural ecosystem. It is comprised of automation of the identification of plant diseases, yield prediction, weed detection, insect detection, crop identification, and livestock management using modern techniques, such as cloud-based services, machine learning (ML), Internet of things (IoT), image processing, big data analytics, and many more. Smart agriculture promotes automated farming and the collection of field data using various means, including cameras, micro-controllers, actuators, and others. The data is analysed by IoT or machine learning to deliver useful information for decision making. Traditional machine learning needs to extract the domain features of input image data followed by classification. Feature extraction expects domain expertise as a prerequisite. Furthermore, traditional machine learning methods are not robust enough to handle high volumes of high-dimensional data. Both of these issues are handled by Deep Learning (DL). DL is widely used for automating various aspects of smart agriculture for two major reasons: it can handle huge amounts of data and does feature engineering on input images on its own (Zhu, 2018; Tseng, 2018). The popularity of this technique and its applicability over the years can be seen in Figure 1, which provides information on papers published in 1985 and between 2010 and 2019 on the application of Deep Learning to different techniques.

Figure 1. Papers published on different applications of deep learning, by year



The next section covers Deep Learning and its advantages and disadvantages, followed by a brief explanation of commonly used DL algorithms: CNN, RNN, LSTM, and GRU. Finally, the authors elaborate on Deep Learning-based agriculture applications and present the conclusions of the study.

BACKGROUND

Deep Learning is a collection of machine learning algorithms that models high-level abstractions in data by means of architectures comprising multiple nonlinear transformations. DL circumvents the prerequisite of feature extraction needed for classification tasks. DL is a deep multi-layered Neural Network (NN), with large numbers of neurons and the objective of capturing complex, nonlinear relationships in input image data (Kamilaris, 2018; Lecun, 1995; Tseng, 2018).

The major advantages of DL are:

- It has the capacity to carry out feature engineering on its own.
- It can handle huge amounts of unstructured data and can be applied for different domains. including computer vision, time series, language and speech processing, games, and many more.
- It provides precise results compared to traditional machine learning.
- The pre-processing required is much lower as compared to traditional classification algorithms.

The major disadvantages of DL are:

- It performs well only with a very large amount of data for training.
- It takes more time for training (because it does the task of feature engineering on its own).
- There is a lack of transparency in interpreting outcomes (i.e., the system is a black box, and researchers do not know how it drives solutions.)
- It requires high-end machines equipped with expensive high-performance GPUs and a large amount of storage to train the models.

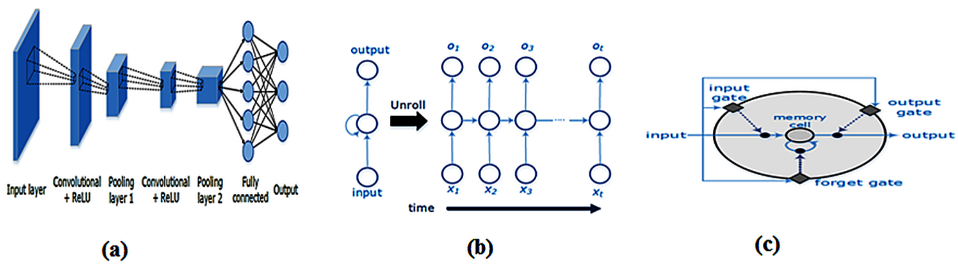
Frequently Used DL Models

Convolutional NNs (CNN) and Recurrent NNs (RNN) are the two most frequently used DL models. Two variants of RNN, Long short-term memory (LSTM) and Gated Recurrent Unit (GRU) are also briefed in this section.

Convolutional NN (CNN)

A CNN architecture typically consists of convolution layers, pooling layers, and fully connected layers. CNN learns complex problems at a faster rate, as it supports weight sharing and complex models, which permit massive parallelization. CNN are characterized by local connectivity between layers and are a multilayer NN composed of different types of layers, with each neuron layer consisting of a weight matrix. Convolution is the first layer to extract features from an input image. The pooling layers section reduces the dimensionality features extracted in the convolution layer. The fully connected layers of CNN are tasked with classification or regression tasks (Kamilaris, 2018; Lecun, 1995; Tseng, 2018). The CNN architecture is depicted in Figure 2(a).

Figure 2. Commonly used Deep Learning neural networks: (a) CNN, (b) RNN, and (c) LSTM
(Kamilaris, 2018)



Recurrent NN (RNN)

RNN are ML techniques, that are well-designed for use in different fields of activity, such as signal processing, natural language processing, and speech recognition. CNN considers only the current input, while RNN considers the current input in addition to previously received inputs. The internal memory of RNN is used to store the previous input, as shown in Figure 2(b). CNN is appropriate for spatial data, such as images, whereas RNN is useful for temporal data, or sequential data. RNN use their internal memory to process arbitrary sequences of temporal data. CNN considers only the present input, while RNN considers both the present and recent past inputs. This is possible because RNN has its own internal memory. The two variants of RNN, LSTM and GRU (Kamilaris, 2018; Lecun, 1995; Tseng, 2018), are shown in Figure 2.

Long Short-Term Memory (LSTM)

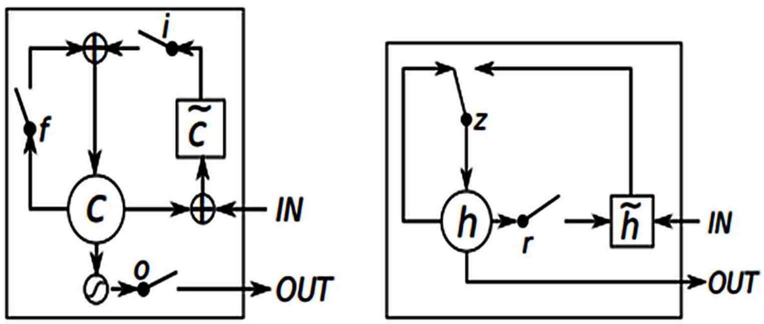
LSTM is an extension of RNN. RNN has problems with vanishing and exploding gradients that are solved by LSTM. LSTM has three gates: the forget gate, the read/input gate, and the write/output gate. If the forget gate is active, the neuron writes its data to itself. If the forget gate is turned off by sending a 0, then the neuron forgets its last content. If the write gate is set to 1, then connected neurons can write to that neuron. When the read gate is set to 1, the connected neurons can read the contents of the neuron. Figure 2(c) depicts the structure of LSTM. The main difference between RNN and LSTM units is that LSTM utilizes forget gates to actively control the cell states and ensure they do not degrade. The gates can use sigmoid or tanh as their activation function. Activation functions cause the problem of vanishing gradient during back propagation and during the training phase of other models that use them (Deng, 2014; Tseng, 2018).

Gated Recurrent Unit (GRU)

In order to assist the computation and implementation of the LSTM model, the neuron GRU was introduced. GRU uses update and reset gates to carry forward information over many time periods in order to influence a future time period. That value is stored in memory for a certain amount of time, and at a critical point, pulls that value out and uses it with the current state to update the next data point. GRU requires a smaller number of parameters; hence, it needs less memory and less training time, making it faster compared to LSTM.

Figure 3(a) depicts LSTM with i, f, and o as input, forget, and output. C and $\tilde{C}$ denote memory and the new memory cell content. Figure 3(b) shows r and z, or reset and update gate, respectively, for GRU; and h and $\tilde{h}$ or activation and the candidate activation, respectively. Both LSTM and GRU are preferred for time series data for forecasting. GRU uses fewer training parameters and, therefore, needs less memory, executes faster, and trains faster than LSTM, whereas LSTM is more accurate for datasets that use a longer sequence.

Figure 3. Variants of RNN: (a) LSTM, and (b) GRU



APPLICATIONS OF DEEP LEARNING FOR SMART AGRICULTURE

Deep Learning for smart agriculture aims to increase both the quality and quantity of agriculture products. The traditional methods for monitoring the health of crops grown in large geographical areas are not only time-consuming, but also difficult to use due to shortages of workers and time. Going forward, there is a need for new technologies that gather and store data (e.g., using drones, satellite images, cameras, sensors, big data, and cloud storage) followed by the analysis of data using artificial intelligence (e.g., DL) with less human involvement. A lot of work has been carried out in this direction using DL. We have identified the following areas in agriculture where DL has been applied successfully as a step toward smart agriculture:

- crop mapping using remote sensed images,
- weed detection
- plant disease detection
- insect detection
- predicting crop yield
- livestock (or animal detection), and
- Irrigation

Deep Learning for Crop Mapping and Identification Using Remote Sensed Images

Remote sensed images are high resolution images that cover larger geographical areas and are suitable for agricultural and land cover classification. Most of the earlier work on crop mapping has been carried out using image processing and traditional machine learning algorithms: Support Vector Machine (SVM), K Nearest Neighbour (KNN), Artificial Neural Network (ANN), etc. This section reviews some of the work related to DL-based crop mapping (Ndikumana, 2018; Weijia, 2017).

Sentinel-1 radar images of Camargue, France have been garnered for agricultural land cover mapping using both (i) traditional machine learning approaches (i.e., SVM, Random Forest (RF) and KNN) and (ii) two deep RNN methods (i.e., LSTM and GRU; Ndikumana, 2018). A field survey was done to accumulate the land cover information, and boundary of reference plots were marked manually. A total of 11 classes were considered: rice, sunflower, lawn, irrigated grassland, wheat, alfalfa, tomato, melon, clover, swamps, and vineyard. Temporal filtering was applied to radar time series data to reduce noise, retaining the fine structures (i.e., without reducing spatial resolution) present in the images. RNN was applied because it takes into consideration the temporal correlation of the data with the objective of identifying different varieties of crops. The results of both LSTM and GRU were found to be superior to that of SVM, RF, and KNN. Among the RNNs, GRU was better compared to LSTM.

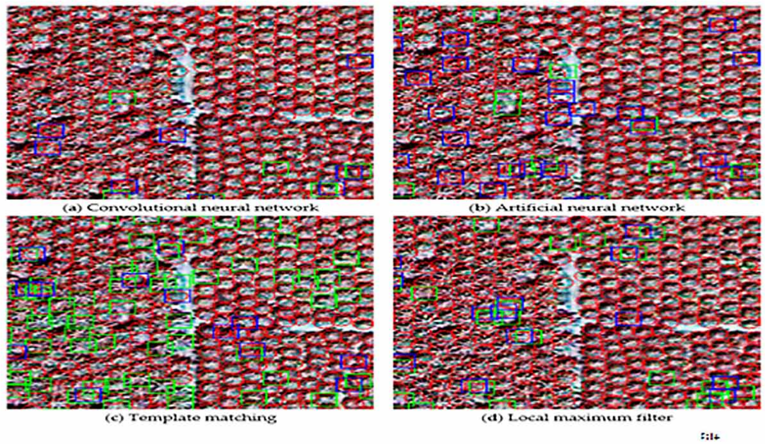
Oil palm tree (an efficient oil seed crop) detection and counting has been accomplished using CNN for Quickbird satellite images. The anticipated results outperformed the traditional methods: Artificial NN (ANN), template matching, and local maximum filter (8). CNN was trained on manually interpreted training samples and tested on autonomously collected test samples. The image dataset for label prediction was garnered using a sliding window of 17 by 17 pixel size, consistent with the feature size of the training and test samples. The earlier trained CNN was used to predict the label of this image dataset (obtained using the sliding window). The coordinates corresponding to the samples that were predicted as the “palm tree” class were merged with the coordinates corresponding to the same palm tree (ground truth) sample to make one coordinate, as to obtain the final palm tree detection results. The results of the proposed CNN model, ANN, template matching, and local maximum filter are depicted in Figure 4, where a red circle represents a correctly identified palm tree (true positive), a green square represents a palm tree in ground truth but not identified as a palm tree in the data (false negative), and a blue square represents a background sample wrongly identified as a palm tree (false positive). Performance is measured using precision, recall, and overall accuracy.

A density map regression model for aerial oil palm images was used for detection and counting the oil palms using Deep Neural Network (Oleh, 2017).

The major concern during a rainless summer is how to effectively utilize water for agriculture. In this regard, a total of 13 summer crop types were considered for the study and land used for anything other than crops was grouped as “other.” The authors worked with 37 Level-2 images taken in 2014 to create time series data: 19 came from Enhanced Thematic Mapper Plus (ETM+), and 18 were from the Landsat-8 Operational Land Imager (OLI). One-dimensional (1D) EVI time series data (computed using blue, red, and near infrared surface reflectance from multi-temporal Landsat images) were the input data chosen by the authors; this was done so that experts in crop phenology and time series interpretation can evaluate the outcome of Multi-Layer Perceptron, one-dimensional convolutional NN (Conv1D), and LSTM of the RNN family (10). The performance of LSTM was low compared to Conv1D. Lower Conv1D layers of the optimized model captured small-scale temporal variations, and seasonal patterns were taken by the upper layers to focus on overall seasonal patterns. The findings suggest that Conv1D was superior to that of three traditional classifiers: XGBoost, Random Forest (RF), and SVM.

Self-organizing maps (SOM) based segmentation followed by an ensemble of CNN has been applied for solving land cover and crop classification (water, forest, grassland, bare land, winter wheat, winter rapeseed, spring cereals, soybeans, maize, sunflowers, and sugar beet) using multi-temporal multisource

Figure 4. Comparative performance of: (a) CNN, (b) ANN, (c) Template matching, and (d) Local maximum filter
(Oleh, 2017).

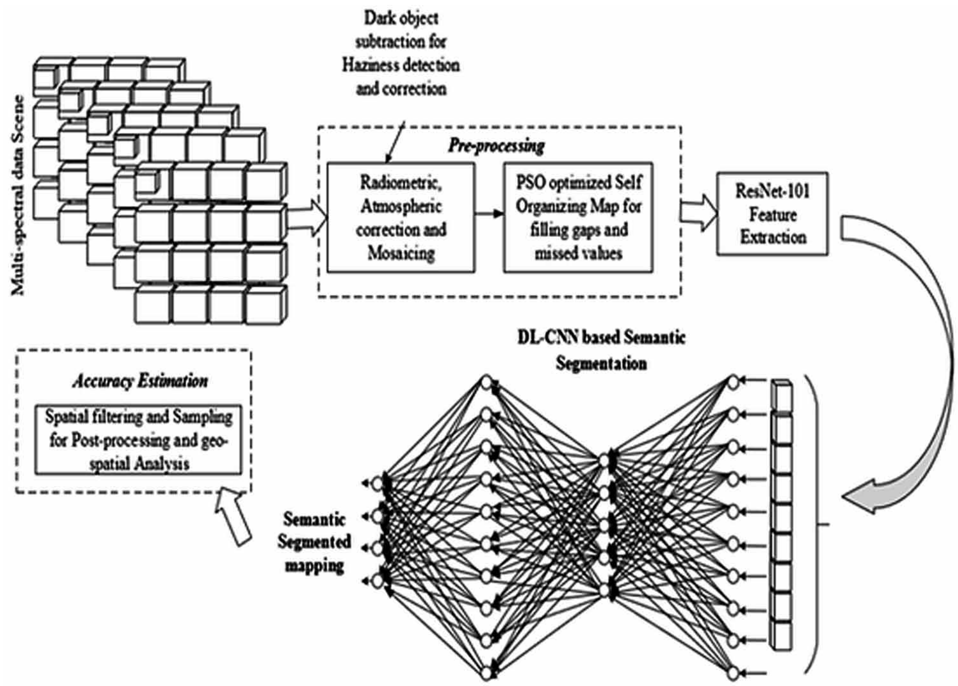


images captured by Landsat-8 and Sentinel-1A satellites (11). Four-level architecture as proposed for classiðcation of crop types involves the following steps. First, data are pre-processed to deal with missing data owing to the presence of clouds and shadows. SOMs are trained using non-missing values for each spectral band, and the coefficients for neuron weight are used to restore the missing pixel values. Second, supervised classiðcation using 1D CNN architecture is used to perform classification of each pixel of the input image and 2D CNN to classify a class for a window with a size of 7 by 7 pixels. Next, post-processing using various filtering algorithms and geospatial data further improves the classification map. Finally, geospatial analysis is completed to interpret classification results for decision making (i.e., to estimate crop areas). One-dimensional CNN and 2D CNN outperformed when compared to an ensemble of multilayer Perceptrons (EMP), and Random Forest classiðers, in particular for maize and soybean crops. Among the CNNs, 2D CNN exhibited better results compared to 1D CNN.

The multispectral remote sensing (RS) images have been used for agricultural and land cover classification (e.g., for crops such as sugar beet, soybeans, rice, maize and wheat) using CNN, and results are compared with SVM, NN, ENN, and RF classifier (Ji, 2018). The proposed method is depicted in Figure 5. The pre-processing steps include radiometric calibration, atmospheric correction, and mosaicking and are performed on the multispectral remote sensed imaged. In addition, missing values are filled in using Particle Swarm Optimization (PSO) optimized SOM. Feature extraction is carried out using ResNet-101 followed by classification using CNN.

The 3-D CNN is used for classification of trees, corn, rice, and soybeans (Zhonga, 2019). With spatiotemporal remote sensing data, 2-D CNN loses the temporal information, which is better handled by 3-D CNN. The difference in performance of each method is illustrated in Figure 6. The images of study were captured during May, June, July, September, and October. Figure 6 (a) depicts 2-D convolution, where Ä2 indicates a 2-D convolution operator where there is no relation between features extracted in different colors in a temporal direction. Å is the sum operator, where all features are collapsed. Figure 6 (b) depicts 3-D convolution where Ä3 indicates a 3-D convolution operator with length 3 in a temporal direction. The Ä3 operator is executed three times serially, as shown by the red, green, and blue arrows through temporal direction. The features indicated by the same-color arrows thus contain temporal

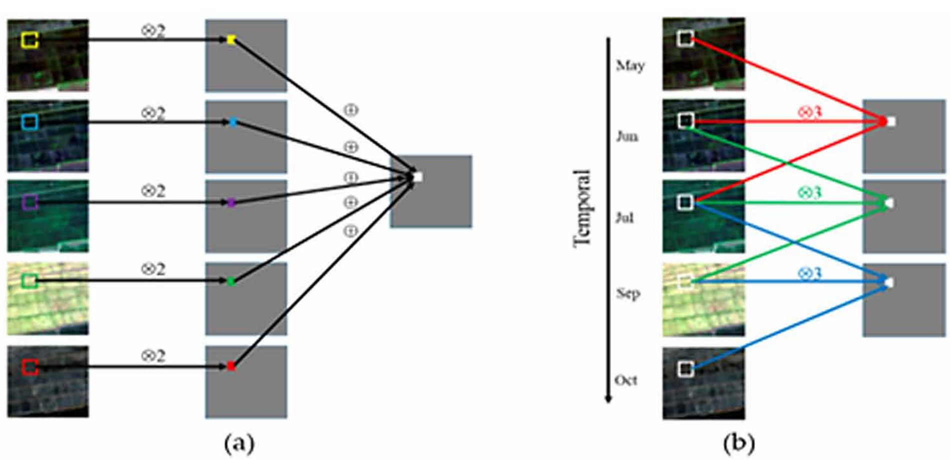
Figure 5. Proposed model for automatic semantic segmentation and classification of multispectral RS data (Ji, 2018)



information, and the output map is also a 3-D tensor. Performance is evaluated using overall accuracy (OA) and Kappa value. The 3-D CNN outperforms 2-D CNN, SVM, KNN and (PCA+KNN) for the classification of tree, corn, rice, and soybeans.

CNN-based rice crop mapping is carried out using spectral bands of the Landsat 8 OLI images with different combinations of land surface temperature (LST), normalized difference vegetation index (NDVI),

Figure 6. Comparison of (a) two-dimensional (2-D) and (b) three-dimensional (3-D) convolution (Zhonga, 2019)



and phonological variables (PV) as input features (Maa, 2018). Vegetation phenology captures vegetation information of various growth stages of crops. LST is a very useful parameter to distinguish rice from other vegetables. Synthetic NDVI was generated by STARFM (spatial-temporal adaptive reflectance fusion mode) by fusion of MODIS and Landsat 8. LST was derived from Landsat 8 OLI images. The dataset of the Landsat 8 spectral images with NDVI, PV, and LST features resulted in superior classification of rice with 97.06% accuracy. Results are compared with the SVM and RF methods. Overall accuracy, kappa coefficient, user accuracy, and producer accuracy were used as performance measures. In addition, the McNemar test was also used to gauge the importance of the classification accuracies of CNN, SVM, and RF.

Deep Learning for Weed Detection

Weeds are unwanted plants that grow along with agricultural crops, and hence compete for water, sunlight, fertilizers, and other resources. Weed detection helps when spraying herbicides to control weed growth precisely, quickly, and with low labor costs. Precise use of herbicides indirectly enhances yield and safeguards the environment. Traditional methods for weed detection were carried out by extracting color, shape, and texture features of weeds followed by classifying weeds and plants using five classifiers: SVM, ANN, RF, decision tree, and AdaBoost algorithms.

In research by Jeon et al. (2011), ANN was applied for weed detection using Gabor filter features. The authors used shape features with SVM for classification of maize crops and weeds (Ahamed, 2012). Texture features extracted from wavelet sub-images were used as input for NN to detect four types of weeds in sugar beet fields (17), and shape features as input to SVM and ANN were used to detect four species of weeds, also in sugar beet fields (Bakhshipour, 2018).

Convolutional NNs (ConvNets, or CNNs) have been proposed for two types of weed detection, grass and broad leaf weed, in soybean crop images with the aim of using a specific herbicide based on the type of weed detected (Santos, 2017). A Simple Linear Iterative Clustering (SLIC) algorithm was applied for identification of weeds in crop images captured by drone. The SLIC algorithm groups pixels based on five dimensions: color, similarity, color of the pixel in the CIE Lab, color space representing three dimensions, and the position (x and y as two dimensions) of the pixels in the image. The groups or segments are used for construction of the image dataset with appropriate class label. A total of 15,000 images of soil, soybean, and two types of weeds (broad leaf and grass weed) were acquired using drones. The CaffeNet architecture was used to train the NN. The proposed ConvNet outperformed the results of SVM, AdaBoost, and RF with input features as shape, color, and texture features. Figure 7 shows the

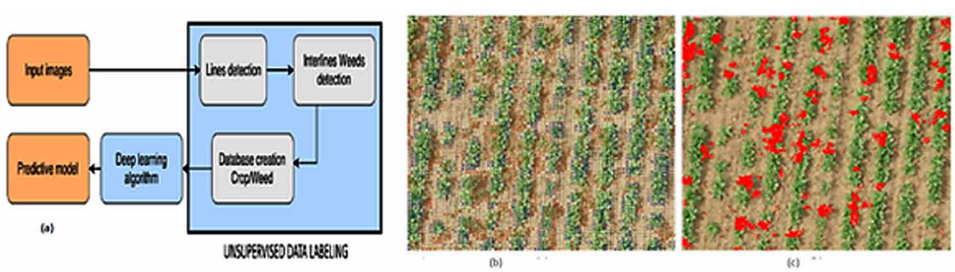
Figure 7. Images of soybeans and weed detection: (a) original image of soybeans, (b) weed detection by CNN, and (c) weed detection by SVM
(Santos, 2017)

following images: (a) the original image for the soybean crop, (b) the image as classified by ConvNet, and (c) the image as classified by SVM. Green indicates soybeans, purple shows soil, red depicts broad leaf weed, and blue is grass weed. The original image had only broad leaf weeds and no grass weed, which is correctly classified by ConvNet with no blue traits, as shown in Figure 7(b). SVM misclassified 3.19% of broad leaf weeds as grass weed, shown as blue traits in Figure 7(c).

One of the major objectives of smart agriculture is to reduce the use of herbicides, as to avoid environmental damage. One must know the growth stage and type of weed to use an herbicide appropriately, thus safeguarding the natural ecosystem. In this context, and as shown in Teimouri (2018), DNN was used to estimate the growth stage of weeds in terms of the number of leaves. The dataset used for experimental work was gathered during three growing seasons to cover a total of 18 weed species in various crops from diverse regions of Denmark; the data also covered a range of soil types, image resolutions, and light conditions. Each image was first classified manually by experts in terms of species and growth stage. The Google Inception-v3 architecture was trained on the 11,907 training images to categorize weeds into nine varieties of growth-stage classes (e.g., 1 leaf, 2 leaves, up to 8 leaves, and greater than 8 leaves).

A majority of crops are grown in regular rows, parted by predeðned spaces, depending on the type of crop. Usually, the plants that grow out of these regular rows are denoted as inter-line weeds. Manual marking of weeds in images for generating data is time-consuming, and there is a need for a large training dataset for DL; the pixel-level annotations are also a time-consuming task. To overcome these problems, unsupervised methods can be adopted to develop training data with the Hough transform method for identification of intra- (i.e., plants, including beans and spinach) and inter-line (i.e., weeds) vegetation. The generated training dataset is classified as weeds and crops using a Residual Neural Network (ResNet; Dian, 2018). The proposed model is depicted in Figure 8. In addition to unsupervised data labelling, supervised training datasets were labelled manually, and an overlapping window was applied to classify each pixel in UAV images using CNNs to provide the probability of the pixel being weeds or crops. The center of the extracted image is marked as blue, red, and white dots to indicate vegetation identified as weed, crop, and uncertain (if both probabilities are very close to 0.5), respectively. Uncertain cases were handled using crop lines information and superpixels. A superpixel is assigned as crop or weed based on whether the majority of dots are in blue or red, respectively. The superpixel with a majority of white dots is classified using crop lines information. Compared to this extra task, the proposed model with

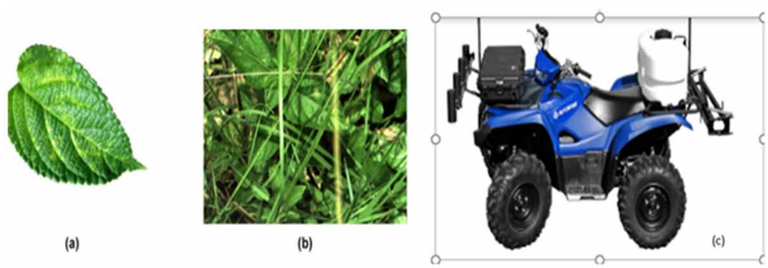
Figure 8. Weed detection model: (a) proposed model for detection of weeds using unsupervised data labelling and ResNet. Classification of spinach and weeds with (b) samples obtained after using a sliding window without crop lines and background information, and (c) detected weeds in red after crop lines and background information have been applied.
(Dian, 2018).



unsupervised data labelling, shown in Figure 8(a), is fully automatic and fits well, even for large-scale training data. Samples obtained after using a sliding window without crop lines or background information are shown in Figure 8(b). Figure 8(c) depicts detected weeds in red after crop lines and background information have been applied.

The dataset DeepWeeds contains eight types of weeds and a total of 17,509 weed images. The data were garnered from eight locations in Australia, and it is the first large public dataset for weeds. Weed classification was carried out using benchmark DL models, Inception-v3, and ResNet-50 (Olsen, 2019) on DeepWeeds. The specialty of the dataset is that it has images that have been captured in realistic environmental conditions, as shown in Figure 9(b), to train a ground-based weed control robot. The proposed prototype robot, AutoWeed, is shown in 9(c) with a high-resolution multispectral camera for crop and weed detection using CNN with unsupervised dataset summarization.

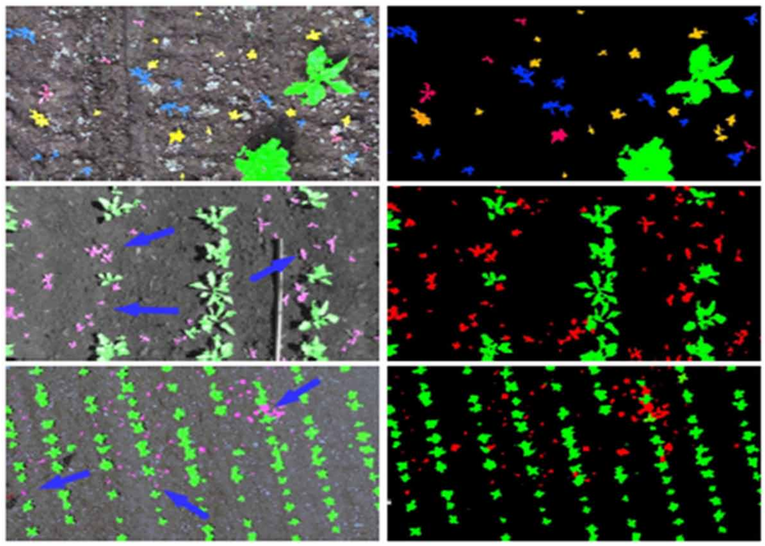
Figure 9. (a) weed image under lab environment, (b) weed captured in realistic environmental condition, and (c) proposed prototype ground-based weed control robot, AutoWeed
(Olsen, 2019)



In (Lottes, 2017), the authors proposed a model to provide a detailed map of the sugar beet crops and weeds on a per-plant basis, even if the crops were not sowed in rows using the Random Forest classifier for UAV-captured RGB+NIR images or RGB images. The authors claimed that the proposed model provides good results in challenging conditions, such as overlapping plants, and identifies weeds in intra-row space. Furthermore, they also showed that the proposed model is able to exploit an arrangement prior to crop rows using a line model, in addition to benefitting from geometric features that capture spatial relationships in a local neighborhood. This work has been tested using three datasets: the PHANTOM-dataset, JAI-dataset, and MATRICE-dataset. The results of the proposed method on three datasets are shown in Figure 10. Arrows in the figure point to detected weeds in intra-row space.

In (Huang,2018), the Fully Convolutional Network (FCN) has been used for detection of weeds in rice. The data set was collected using multi-rotor UAV-captured images from a rice field with two naturally infested types of weed species. In the UAV imagery, all the objects were divided into three classes: rice, weeds, and others (including cement ground, water, etc.). The images were manually labelled (ground truth) as green for rice, red for weeds, and gray for others. The main difference between CNN and FCN is explained in Figures 11(a) and 11(b). The CNN is composed of multiple convolutional layers (for high future representation) interlaced with a pooling layer, followed by some fully connected layers, which reduces the image dimension and puts out a 1-D distribution over classes to classify one image to one single label. The FCN network transforms all of the fully connected layers into convo-

Figure 10. Images in the right column are the ground truth images; those in the left column are the result of the proposed method for weed and sugar beet detection. The first row shows multi-class results for the PHANTOM dataset; the second row shows results for the JAI-dataset, and the third row presents the results for the MATRICE-dataset. Green, blue, yellow, and red colors denote sugar beets, chamomile weed, saltbush weed, and other weeds, respectively.
(Lottes, 2017).

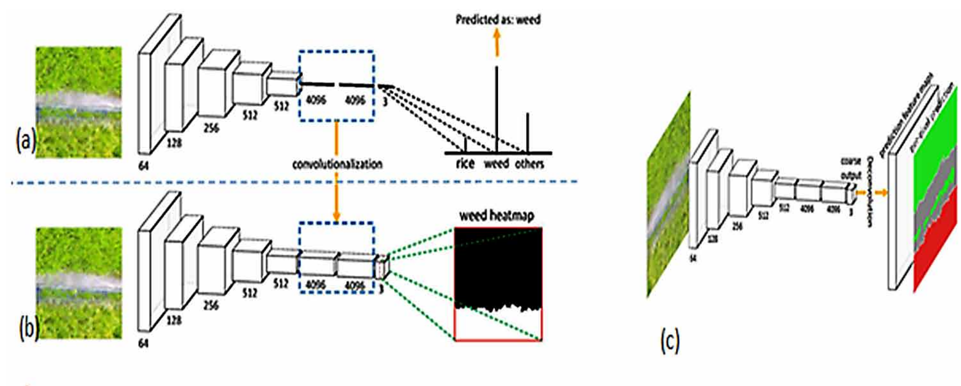


lutional layers and enables a classification net to create a heatmap, as shown in Figure 11(b). As with the up-sampling approach, the deconvolutional layers in FCN restore the output to full resolution, and pixel-to-pixel prediction architecture was established, as shown in Figure 11(c). The channels of the last deconvolutional layer are equal to the quantity of classes, and each feature map represents the heat-map for a certain class. The three feature maps correspond to rice, weeds, and others, shown in green, red, and gray, respectively. For each pixel of the feature maps, the class with maximum probability is used as predicted class labels. The FCN method is used for weed mapping of the collected image. The performance of FCN, the Patch-based CNN algorithm (each patch is input to the network as an image, and the relationship between different patches is ignored, resulting in discontinuities between patches), and Pixel-based CNN method (the whole image is processed pixel by pixel) are depicted in Figure 12. From Figure 12(c), it is observed that the results of Patch-based CNN indicate most of the weeds as rice. Figure 12(d) suggests that Pixel-based CNN misclassified lots of rice as weeds. Comparatively, Figure 12(e) shows an accurate map produced by FCN.

Deep Learning for Insect Detection

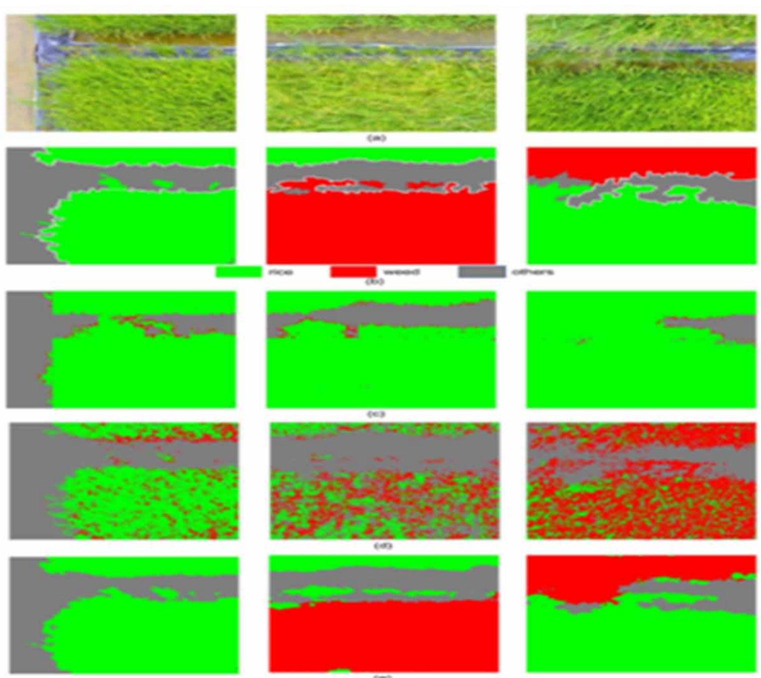
Insects cause severe damage to crops, as well to stored grains. Not all species of insects are helpful to the human economy, thus there is a need for precise identification of insects; by doing so, humans can adopt the proper use of pesticides to avoid killing cooperative insects (e.g., the honey bee, antlion, blue dasher dragonfly, cross spider, etc.) and reduce environmental hazards and investments in pesticides. In stored food grain, one needs to identify different varieties and sizes of insects in addition to broken

Figure 11. Architecture of (a) CNN, (b) FCN, and (c) deconvolutional operation for per-pixel classification tasks
(Huang, 2018)



grains, which are fine materials. Much research has been carried out by combining image processing and machine learning to extract an insect's features, which include: wing structures, texture, and color, followed by classification using SVM, KNN, ANN, RF, multinomial logistic regression, extreme machine learning, ensemble methods, and others.

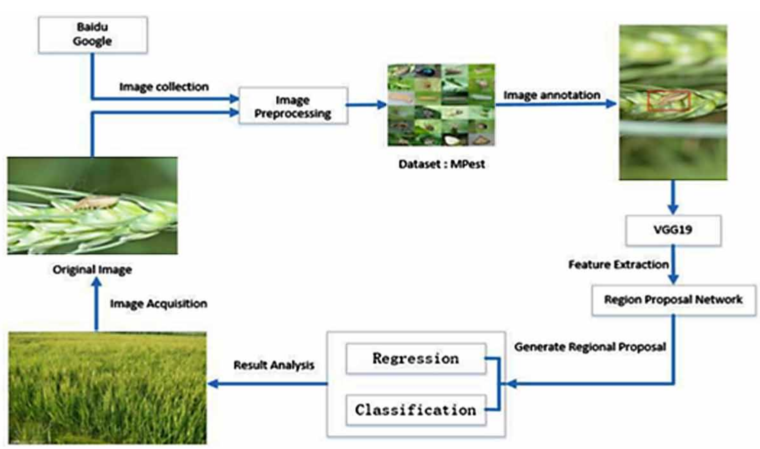
Figure 12. Results of classification: (a) UAV imagery, (b) Ground truth label, (c) Patch-based CNN, (d) Pixel-based CNN, and (e) Proposed FCN-8s
(Huang, 2018)



Among the various types of *Lepidopterans*, few are harmful and few produce materials like silk, which is useful to humans. Thus, it is essential to identify these insects so that the use of pesticides does not destroy insects that are friendly to humans. CNN combined with SVM has been used for the identification of 22 types of *Lepidopterans* (Zhu, 2017). The dataset for the study has images of 22 species of *Lepidoptera* within seven families. The authors used pre-trained parameters of Deep CNNs (DCNNs), such as Alex Net and VGG16, to extract the features from the insect images. These features are used for training SVM for the classification of *Lepidopterans*. The proposed model is able to detect insects in the images in about 200 ms.

The Region Proposal Network (RPN) modelled for crop insects' recognition and classification (Xia, 2018) is depicted in Figure 13. Images of 24 species of insects were manually collected using the Baidu and Google search engines. As part of feature extraction, in the ðrst stage, the VGG19 CNN network is adopted to extract high-dimensional features of insects. The RPN endorses the location of insects from a feature map. Depending on the threshold value set by the user, the proposal region is assigned a positive label if the ratio between the intersection area of the insect proposal box and ground truth box and the union area of the insect proposal box and ground truth box is high; everything else is assigned a negative label. The region of interest (ROI) pooling converts insect-like regions into a fixed spatial size to generate feature vectors with equal dimensions (describing the location and category information of the target), which is an input to the fully connected layers. The dimension feature vector is an input to the softmax layer for the purpose of classifying the insects. The proposed model, as shown in Figure 13, outperformed the Single Shot Multibox Detector (SSD) and the Fast Region-based Convolutional NN (RCNN) both in terms of accuracy and training time.

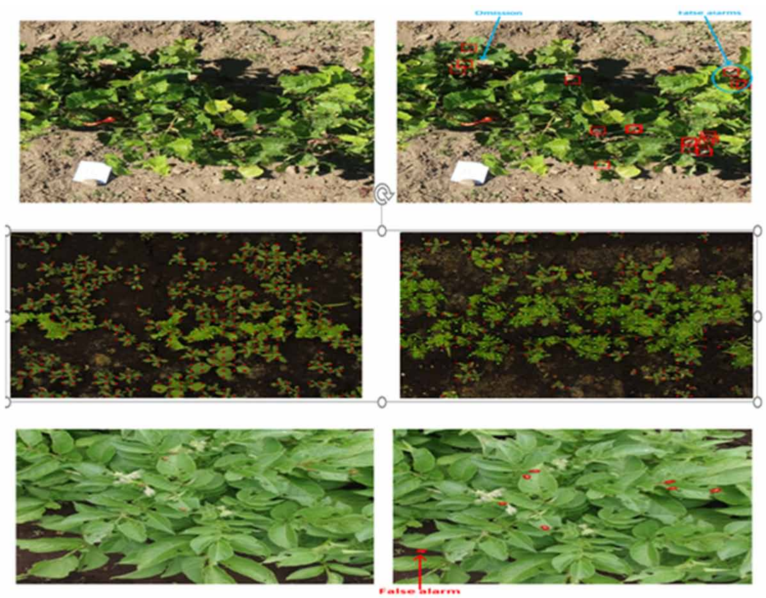
Figure 13. Diagram of RPN-based insect recognition and classiðcation
(Zia, 2018)



The ten-lined potato beetle, or the potato bug, is one of the major pests of potato crops. CNN CaffeNet combined with the SVM classifier is used for multiple tasks, including detecting diseases, beetles, grazing, and weeds, using UAV images (Bouroubi, 2018).

The first row of Figure 14 illustrates the detection of mildew symptoms (i.e., as dried leaves) on one of the Phantom-3 images associated with GPS coordinates recorded during the flight. The mapping

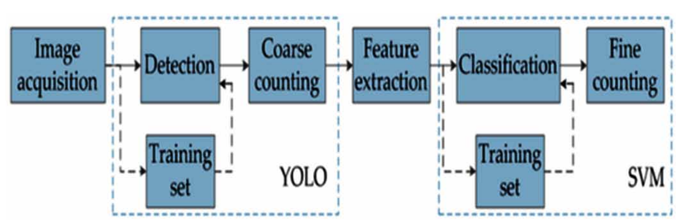
Figure 14. Detection of (a) mildew symptoms (dry leaf) in the first row, (b) weeds in red in lettuce and carrot crops, respectively, in the second row, and (c) the presence of the beetle, or potato bug, in the third row using DL
(Bouroubi, 2018)



provides information about the extent and severity of infections to establish a phytosanitary intervention plan. Photos in the second row indicate the detection of potato beetles on a Phantom-3 image, except for one false alarm. The work also includes the detection of weeds in lettuce, carrot, and onion fields using Deep Learning. The third row depicts weed detection in red in lettuce (left) and carrot (right) crops using the DL method.

The YOLO (You Only Look Once) DL method in combination with the SVM classifier is applied for the identification of six types of insects: bee, fly, mosquito, chafer, moth and fruit fly (Zhong, 2018). The work is part of an integrated agricultural monitoring service platform with pest information as one of the components. The process for the proposed detection of insects model is shown in Figure 15. The yellow sticky trap is installed in the wild to collect images of flying insects influenced by various challenges, such as varying illumination, presence of insect excrement, dead leaves, water droplets, and mud spots. The input image is divided into an S by S grid. Then, each grid predicts B bounding boxes of objects if the object's center falls into the grid. The bounding box has five parameters: x and y as center point

Figure 15. Course counting and detection of flying insects using YOLO and SVM (28)



coordinates, width and height of the box relative to the whole image, and confidence. The confidence is computed as the product of the probability that the box has an insect and Intersection Over Union (IOU, which is the intersection of the predicted box and ground truth box). Finally, bounding boxes with relatively low class-speciðc conðdence scores and non-maximum suppression (NMS) are used to remove the redundant bounding boxes. To speed up the training, the YOLO trained using 1,000 classes of ImageNet data. YOLO considers only two classes: insect and non-insect. The output of YOLO (course count of insects) along with additional shape, color, and texture features and a histogram of oriented gradient (HOG) feature are provided as an input to SVM for final classification of six types of insects. SVM was explored with four kernel functions: linear, polynomial, radial basis function (RBF), and sigmoid. SVM resulted in improved performance with shape, color, and texture features using the RBF kernel function.

In (Nazri, 2018), the PENYAK model is used for classifying the brown planthopper (BPH) insect and benign insects in paddy fields. Data were collected from the paddies after 70 days of planting, having been infested with BPH. The steel plate with a plastic sheet sprayed by sticky glue was put on the base of the paddy stem to trap the BPHs by tapping the paddy stem. Images of sticky pads with BPHs were manually marked as bounding circles were captured. The BPH and benign insect images were cropped from the sticky pad marked image. Median filtering was applied to remove noise and to preserve the edges of the insect region. An iterative multiple thresholding algorithm was applied to distinguish the image pixels in the foreground and background. The binary images were transformed by one of the six selected binary operations: outline, fill hole, Skeletonize, Euclidean Distance Map (EDM), Watershed, and Voronoi. EDM achieved favorable results; hence, it was selected for further evaluation with CNN VGG16 for BPH classification. The results were compared with the RF classifier with different combinations of input features (Grey-Level Co-Occurrence Matrix (GLCM), Gabor filter, Local Binary Patterns (LBP) histograms, RGB histograms, Hue Saturation and Value (HSV) histogram, and gray-level histograms). The proposed PENYEK architecture of VGG16 and the EDM with pre-trained weights and biases outperformed the RF classifier for classifying BPH insects.

Deep Learning for Plant Disease Detection

Timely detection of crop diseases is necessary for ensuring the sustainable production of crops. Among the various causes (e.g., climate conditions, soil type, seed type), crop disease is one of the key threats to the decline in food production. There are three main causes for plant disease: a susceptible host plant, presence of a pathogen, and a favorable environment for interaction between the host plant and pathogen. The presence of a pathogen leads to plant disease, which is identified by the presence of spots, wilts, and scabs on the leaves, fruit, stem, and roots. Botanists, plant pathologists, and agriculture engineers generally practice visual inspections of plants followed by lab experiments, if necessary, which is a time-consuming and tedious job; thus, there is a need for automated plant disease detection using image processing and machine learning, which does not demand knowledge of diseases and pests. The traditional method for the detection of diseases and pests has been carried out using SVM, k-means clustering, Genetic Algorithm (GA), ANN, probabilistic NN (PNN), extreme learning machines (ELM), and KNN methods, to name a few. DL methods, including Faster Region-based Convolutional NN (Faster R-CNN), Region-based Fully Convolutional Network (R-FCN), SSD, and CNN have been used for disease detection in plant leaves, stems, and grain for various crops and fruits.

One of the major aspects of precision agriculture is the timely and accurate diagnosis of plant diseases. The Deep Learning CNN model was able to detect leaf presence and differentiate between healthy leaves and 13 diseases (Sladojevic, 2016). The dataset was comprised of images of healthy leaves and a background image; diseased leaves from 13 categories were mentioned as paired plant names and leaf diseases: (a) Pear, cherry, and peach, porosity; (b) Peach, powdery mildew; (c) Peach, *Taphrina deformans*; (d) Apple, pear, *Erwinia amylovora*; (e) Apple, pear, *Venturia*; (f) Apple, powdery mildew; (g) Apple, rust; (h) Pear, *Gymnosporangium sabinae*; (i) Pear, gray leaf spot; (j) Grapevine, wilt; (k) Grapevine, mites; (l) Grapevine, powdery mildew; and (m) Grapevine, downy mildew. Data augmentation was done using the affine transformation, perspective transformation, and simple image rotations to increase the number of training and test data from 3,000 to 30,000 in order to avoid over-fitting.

In Lu (2017), the authors used CNN inspired by LeNet-5 and AlexNet to detect ten rice diseases: rice blast, rice false smut, rice brown spot, rice bakanae disease, rice sheath blight, rice sheath rot, rice bacterial leaf blight, rice bacterial sheath rot, rice seeding blight, and rice bacterial wilt. The dataset of 500 images of healthy and diseased rice, leaves, and stems, as mentioned above, was considered. Instead of max pooling, the authors used stochastic pooling, claiming that the latter improves the generalizability and avoids over-ðtting.

Chowdhury et al. () have used DNN for rice disease and pest detection using around 1,500 images comprising six diseases (false smut, sheath blight, sheath rot, bacterial leaf blight, neck blast, and brown spot) and three pests (brown plant hopper, stemborer, and hispa). Eight types of CNN architectures (VGG16, ResNet50, InceptionV3, InceptionResNetV2, Xception, DenseNet121, DenseNet169, and DenseNet201) were explored with three types of training methods. In baseline training, training was done from the scratch with random initial weights. This type of training results in good accuracy but at the cost of a higher converging time. In fine tuning, the convolution layers are trained from their pre-trained ImageNet weights, and the dense layers are trained from randomly initialized weights. In transfer learning, the pre-trained ImageNet weights are retained and the dense layers are trained from their randomly initialized weights. Among the eight CNN architectures, and for all three training methods, VGG16 architecture resulted in consistently high accuracy (Rafeed, 2018).

In Xiao (2018), LSTM, a special type of RNN, was applied to predict the occurrence of four types of pests and cotton diseases (bollworm, whiteñy, jassid, and leaf blight). The data recorded cotton documents for 10 insect pests and cotton diseases with various weather features: maximum temperature (MaxT, in °C), minimum temperature (MinT, in °C), relative humidity in the morning (RH1, as a percentage), relative humidity in the evening (RH2, as a percentage), rainfall (RF, in millimeters), wind speed (WS, in kilometres per hour), hours of sunshine (SSH, in hours), and evaporation (EVP, in millimeters) of six locations in India. The LSTM identifies the relationship in the data in the form of (Xi, Yi) where Xi is the weather features and Yi is the outcome of the prediction (i.e., the future occurrence of pest and disease). Accuracy (ACC), area under the curve (AUC), and F1-score are used to measure the effectiveness of LSTM. Results of LSTM were found to be superior when compared with KNN, SVM, and RF results.

The DCNN was trained on an NVIDIA Quadro P4000 with CUDA for identification of cucumber diseases (Maa, 2018). The stochastic gradient descent with momentum (SGDM) was used to optimize the network weights. An image dataset was prepared with images from the Internet, as well as from camera-acquired images, for four cucumber diseases: anthracnose, downy mildew, powdery mildew, and target leaf spots. A region growing segmentation method was adopted using three color components—the Excess Red Index (ExR), the H component of HSV color space, and the b*component of L*a*b* color space—so as to discriminate between disease spots and clutter in the background. Data augmentation was done to increase the size of the dataset and to reduce over-fitting. DCNN resulted in better performance when compared with Random Forests, Support Vector Machines, and AlexNet. A total of 54 feature sets comprising of color (mean variance of nine channels, three each from RGB, HSV, and L*a*b) and GLCM texture features (contrast, correlation, energy, and homogeneity of the nine channels) were the inputs for SVM and RF.

GoogLeNet and Cifar10 models based on DL (Zhang, 2018) were applied for investigating eight varieties of maize leaf disease detection: Curvularia leaf spot, dwarf mosaic, gray leaf spot, northern leaf blight, brown spot, round spot, rust, and southern leaf blight. The network models were optimized by the SGD algorithm. Images were collected from different sources—the PlantVillage and Google websites—followed by data augmentation via: rotating the images 90 degrees, 180 degrees, and 270 degrees; by mirroring each rotated image; by cutting the center of the image by the same size; and by converting all processed images to grayscale. In Arivazhagan (2018), CNN was applied for the identification of healthy leaves and five types of leaf diseases in mango plants: anthracnose, alternaria leaf spots, leaf gall, leaf webber, and leaf burn.

Faster Region-based CNN, Region-based Fully CN (R-FCN), and Single Shot Multibox Detector were adopted for disease and pest detection in tomato plants (i.e., stem, leaves, fruits, etc.) by collecting around 5,000 images of different stages of the tomato diseases from Korea (Fuentes, 2017). Each of these Faster R-CNN, R-FCN, and SSD were combined with VGG16 and ResNet as deep feature extractors. The proposed model was used for the detection of nine types of diseases, pests, and other syndromes in tomato plants: gray mold, canker, leaf mold, plague, leaf miner, white, low temperature, nutritional excess or deðciency, and powdery mildew.

A novel structure of a deep CNN based on the AlexNet model was designed by eliminating partial fully connected layers, adding pooling layers, and introducing the GoogLeNet Inception structure into the network model for detecting four types of apple leaf diseases: *Alternaria* leaf spot, mosaic, rust, and brown spot. Stochastic Gradient Descent (SGD) was used to update the weight for CNN, which may get stuck with a "local optimum" problem (Liu, 2018). To overcome this problem, training of the network was done using Nesterov's Accelerated Gradi (NAG) optimization algorithm in place of SGD; furthermore, NAG had a higher rate of convergence. Pathological images of apples were acquired and processed using digital image processing technologies, such as directional disturbance (a pathological image is rotated at 90 degrees, 180 degrees, and 270 degrees with horizontal symmetry), light disturbance (brightness adjustment randomly increasing or decreasing the RGB pixel values), and adding noise using Gaussian and PCA (Principal Components Analysis) jittering to disturb natural images as to generate a larger dataset and to lessen over-fitting at the training stage. The outcome of the proposed method beat the results of SVM, BP, AlexNet, GoogLeNet, ResNet-20, and VggNet-16.

In Tooa (2018), an assessment of four DL architectures (VGG16; Inception V4; ResNet with 50, 101, and 152 layers; and DenseNets with 121 layers) was carried out for the identification of 38 classes of diseased and healthy images of 14 plants from the PlantVillage dataset. In DenseNets, all layers were connected directly with each other in a feed-forward manner to ensure maximum information ñow between layers in the network. The feature maps of the previous layer were used as inputs for the next layer. DenseNets alleviated the vanishing-gradient problem and substantially reduced the number of parameters. All the models used for this study were loaded with pre-trained weights from ImageNet and the model was fine-tuned using the SGD algorithm. No data augmentation was done for any of the four networks. Among the networks, DenseNets not only beat the rest of the architecture, but also required signiðcantly fewer parameters and less time for plant and disease classification.

In Türkoğlu (2018), frequently used DNNs (AlexNet, GoogleNet, GG16, VGG19, ResNet50, ResNet101, InceptionV3, Inception ResNetV2, and SqueezeNet) were examined for plant disease (five types) and pest (three types) images from Turkey using transfer learning and deep feature extraction. The features extracted by DNN were inputs to traditional classifiers (SVM, extreme learning machine, and KNN) for classification using 10-fold cross-validation. The results indicated that the proposed architecture (i.e., deep feature extraction as an input to a traditional classifier) had higher classification accuracy than networks based on transfer learning-based networks. Among the traditional methods, deep feature extraction and the SVM/ELM classifier resulted in superior results than networks based on transfer learning. The performance of SVM, ELM, and KNN with traditional futures (LBP, HOG, GLCM, and color) were poor, compared to deep features.

Deep Learning for Yield Estimation

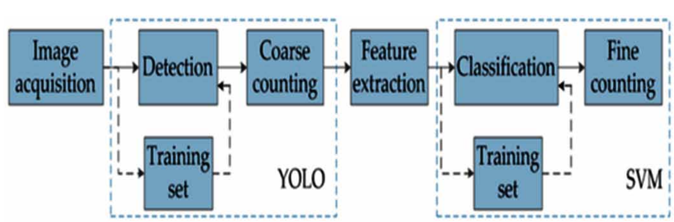
The food supply issue can be addressed by crop yield forecasting. A few of the advantages of crop yield forecasting are that it:

- helps farmers make financial decisions;
- allows policy makers to make decisions related to the import and export of crops in a timely manner by being able to predict the crop yield during the growing season;
- forecasts market prices in order to minimize the impact of crop losses on farmers; and
- can be used to plan the overall food supply for the nation as a whole.

Most of the traditional methods adopted for crop yield prediction include support vector machines, decision trees, regression models, Auto Regressive Integrated Moving Average (ARIMA) models, association rule mining, and artificial NNs. Most of the earlier research on yield estimation includes the use of statistical data or image processing.

In Oliveira (2018), RNN is trained with features that include: monthly precipitation data, monthly air temperature data (i.e., minimum and maximum temperatures), and soil properties (i.e., information about clay, silt, and sand contents plus ðne earth and coarse fragments bulk density) as well as past yield data for soybeans and maize. Most of the earlier yield predictions using satellite images were based on the Normalized Difference Vegetation Index (NDVI). The proposed model is capable of precise forecasting, which helps the farmer to make decisions regarding selection of superior genetic variations of crop or another crop type so as to sustain future climatic variations. The input data are split into two types: static data (soil data) and dynamic data (weather data, including precipitation and temperature). The static data

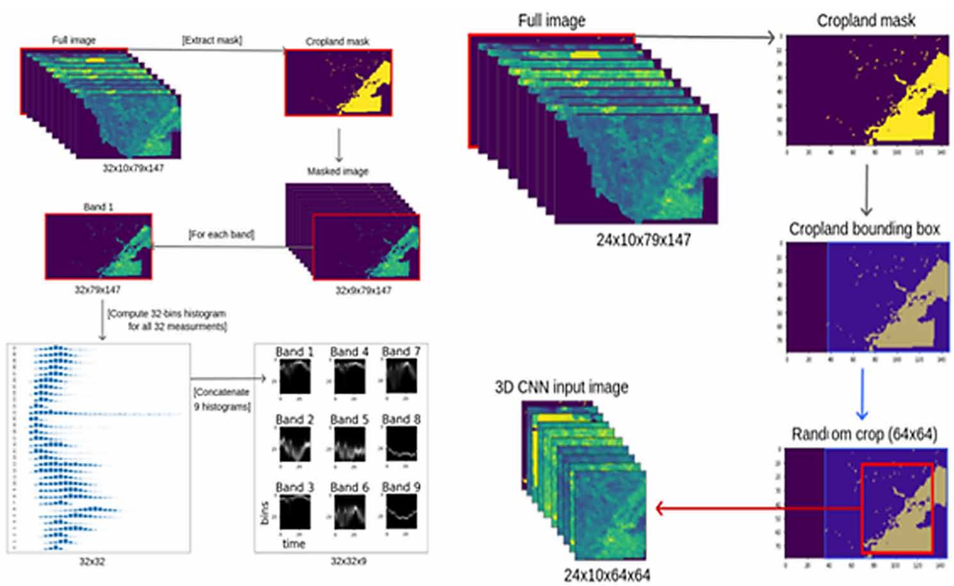
Figure 16. Proposed RNN model trained with precipitation, temperature, and soil properties as features and past yield data for preseason forecasting
Oliveira (2018)
Note: The abbreviations "lat" and "lon" represent latitude and longitude.



ñow through a two-layer fully connected NN, and the dynamic data ñow through a three-layer LSTM NN (because LSTM are good at dealing with temporal data). After both the static and dynamic data paths are computed, the network joins the data as inputs to fully connected layers and finally releases a single forecasted value as output. Performance is measured using Mean Absolute Error (MAE) and scaled exponential linear units (SELUs). The proposed RNN model for yield prediction is depicted in Figure 16.

In addition to weather data, genotype information is provided as an input to LSTM for soybean yield

Figure 17. (a) of the pre-processing of 3-D histograms, and (b) of the random cropping procedure for training inputs to the 3-D CNN
(You et al,2017)

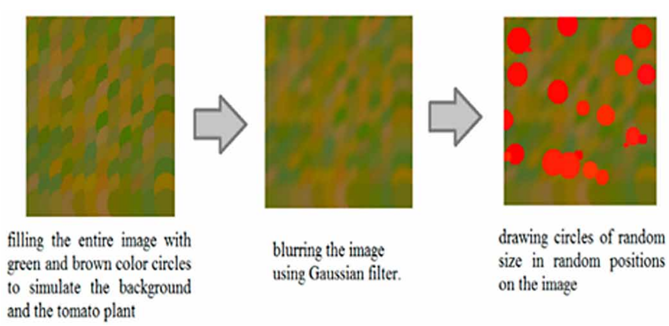


predictions (Shook, 2018). The authors claim that the model can be used for phenotype prediction (i.e., trait response: protein, oil, days to maturity, plant height, seed size, etc.) for plant breeding, site selection, yield estimation, and making financial decisions related to crops.

In Russello et al. (2018), the authors chose a histogram CNN model as they claimed it performs better than the LSTM model. The counties of 11 major soybean-producing states were selected. Yearly yield data at the county level were downloaded from USDA NASS during 2003 to 2015 and were used as ground truth targets. Surface reñectance, land surface temperature, and land cover were taken for every selected country 32 times during the years 2003 to 2015 from MODIS via Google Earth. The remote sensing data were transformed into 3-D histograms, as illustrated in Figure 17(a). These images of pixel intensities were provided as inputs to HistCNN (as proposed by Russello, 2018) for crop yield predictions. In addition to yield predictions, they provided two important investigation results. First, they proved that HistCNN would provide better yield predictions with recently collected training data compared to old data. Second, they found that the HistCNN model trained on particular source locations would work on a new target location if both the source region and target region had similar or generalizable ecosystems. The histogram CNN only allows us to extract temporal and spectral features from the satellite images while discarding spatial information. Likewise, it predicts crop yields based on only the spectral response of crops through time with the assumption that the positions of the pixels mark only the location and are not associated with yield predictions (You et al. 2017). In this context, the authors proposed using 3-D CNN which provided superior crop yield predictions by including spatial information (e.g., soil- and eco-properties of the area, such as presence or absence of water sources near the croplands and whether or not the area was close to urban neighborhoods) in addition to spectral and temporal signals. The workñow of the random cropping procedure for training input to the 3-D CNN is shown in Figure 17(b). The results of 3-D CNN were superior to the results of HistCNN, Ridge, decision tree, and CNN.

In Rahnemoonfar (2017), a modified version of Inception-ResNet architecture was adopted to capture multiple features for counting tomato fruits using synthetic data. The proposed method was robust enough to predict the number of tomatoes (yield) even if the fruits were under shadow or obstructed by other vegetation or branches, and even in the presence of overlap among the tomatoes. Because more images with real counts were not available, synthetic data were generated for the study, as shown in Figure 18. The modified Inception-ResNet with three parallel layers connected as one, was added to the activations of the previous layers and passed through the rectified linear function. In addition, the modified reduction module was used to simultaneously reduce the image size and increase the number of filters. The modifications were introduced with the objective of avoiding model over-fitting and provided better results than did the real test data. Finally, the fully connected layer provides the expected output (i.e., the count of tomatoes). The network was trained using 24,000 synthetic data points and the Adam optimizer; it was then tested using 2,400 synthetic and 100 randomly-selected real tomato images from Google Images. The regression line fitted well over the actual count data and computed the count for the proposed model. The proposed method was fast, and the results were significantly better compared to the area-based technique (which calculated the number of fruits based on the total area of fruits and an individual fruit), shallow NN, and the original Inception-ResNet.

Figure 18. Synthetic data for training improved ResNet for automated counting of tomatoes (Rahnemoonfar, 2017)



Assessment of fruit load can be used for early yield estimation in addition to monitoring harvest and thinning. In Keresztes (2018), an innovative generic method was proposed for real-time fruit detection of any type. The pre-processing step was based on the circular Hough transform method intended to recognize the positions of fruit regardless of color or stage of development, followed by DNN for classification. The model was tested for grapes and apples of different types and phenological stages and provided promising results.

In Liu (2018), deep NN was applied for counting fruits across an image sequence. The three major stages of proposed work are depicted in Figure 19. The FCN was used to segment the image as fruit and not fruit pixels. The tracking stage used the segmented information of fruits from FCN to obtain associations across the image frames, so as to obtain fruit counts for the entire image sequence as opposed to an individual image. Tracking steps provided an initial count of the number of fruits in an image sequence. This count was prone to errors because: fruit may be double counted, it may consider the fruits from a tree in another row rather than the row in which the fruits were counted, or there may be false positives or outliers. This was overcome by combining tracking results with a Structure from Motion (SfM) algorithm to estimate relative 3-D locations and size to get a correct count. The proposed work was tested on apple and orange datasets with a lot of depth variation and uncontrolled illumination.

Figure 19. The FCN+tracking+3-D localization model used to accurately count fruits
(Liu, 2018)

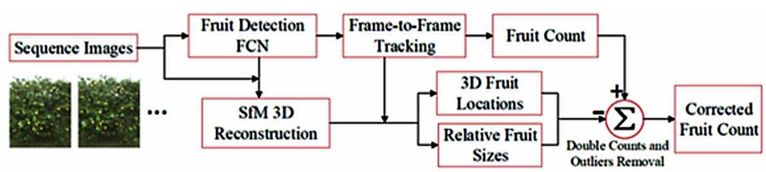


Image processing-based plant phenotyping helps identify variations in plant growth and, thus, assist crop diagnostics. Moreover, plant phenotyping can contribute to attaining the global food supply. Traditionally, plant phenotyping is carried out by experts by manually measuring annotating plant traits (i.e., plant size, shape, and the number of leaves and flowers) which is no doubt a time-consuming task. DL with image processing can play a major role in plant phenotyping. Work is carried out to reduce the computation time of CNN (by reducing the number of CNN parameters), hence the name "lite" CNN for pixel-wise segmentation of the Oxford flower dataset. The objective in this case was to deploy the model on low-cost devices for plant phenotyping. Two metrics were used: variations on pixel accuracy and region IOU were used for common pixel-wise segmentation evaluations (Atanbori, 2018).

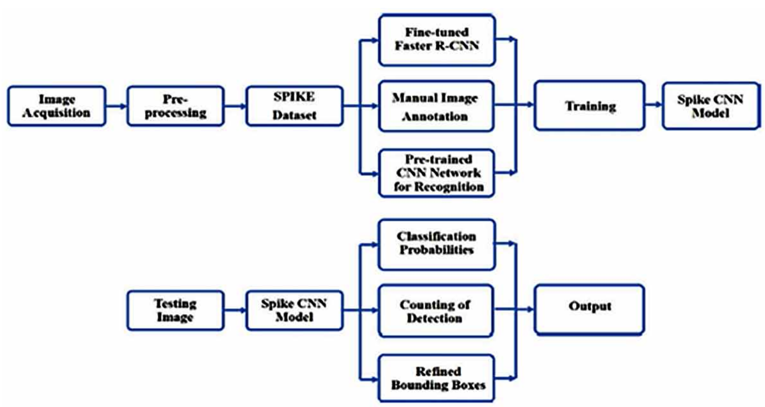
CNN was applied to extract features of individual plants and was provided as the input to LSTM for identifying temporal changes in them (i.e., growth rate, number of leaves, and texture changes) to model the relationships between their phenotypes and genotypes. Performance of the proposed CNN-LSTM model was superior when compared with various models: (i) CNN, (ii) handcrafted features with SVM, (iii) handcrafted features with LSTM, and (iv) Conditional Random Fields (CRF) with LSTM. CRF is a popular probabilistic graphical model for encoding structural and temporal information of sequential data (Naming, 2018).

Genomic selection (GS) is a promising breeding strategy which helps in predicting phenotypes from genotypes. The DL technique is used to develop deep GS models for predicting phenotypes of wheat from genotypes. In (Ma,2018), the authors carried out work on a GS dataset:CIMMYT (International Maize and Wheat improvement Center known by its Spanish acronym CIMMYT for Centro Internacional de Mejoramiento de Maiz y Trigo), which has 2,000 cases of Iranian bread wheat (*Triticum aestivum*) and landrace accessions genotyped with 33,709 DArT (Diversity Array Technology). The mean normalized discounted cumulative gain value (MNV) was used as a prediction performance measure. Results were compared with Ridge regression-based linear unbiased prediction (rrBLUP), which is the most commonly used regression model for GS. Furthermore, the authors integrated DeepGS and rrBLUP to provide much superior predictions. The weights signifying the contribution of DeegGS and rrBLUP for predictions were optimized using the particle swarm optimization (PSO) algorithm (Ma, 2018).

Detection and characterization of wheat spikes from wheat images in fields would be helpful in an assortment of high yielding wheat varieties. In Hasan (2018), the authors made the SPIKE dataset public; this dataset contains images of 10 varieties of wheat that underwent three types of fertilizer treatment: no treatment, early treatment, and late treatment. The robust R-CNN model (Figure 20) was trained on spike images of 10 varieties of wheat; these varieties were in several growth stages: GSGC images contained green spikes and a green canopy, GSYC images contained green spikes and a yellow canopy, and YSYC images contained yellow spikes and a yellow canopy for predicting high-yield wheat varieties for improved crop productivity. Wheat images were acquired in the field, followed by automated cropping ROI. Training images were then manually annotated with bounding boxes. Both the cropped and annotated images were passed to the R-CNN model for training.

Deep Learning for Livestock Farming

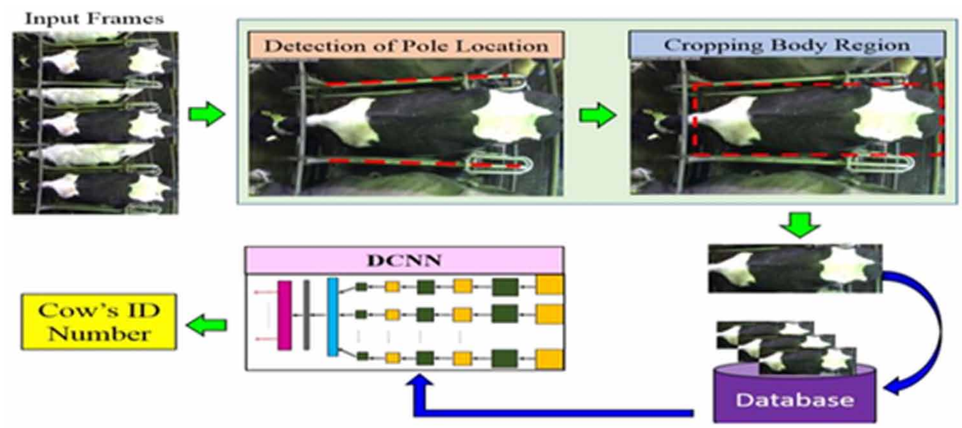
Figure 20. Training and testing of the proposed R-CNN model for wheat spike detection and yield prediction (Hasan, 2018)



There is a need for an automatic monitoring system of cows in livestock farms, as the number of cows is increasing every year, and it is cumbersome to hire workers to conduct this task. Radio Frequency Identification (RFID) technology is one solution, but it comes at the cost of causing stress on the animals, high expenses, and unreliability. The identification of lameness in cattle has been conducted using traditional machine learning algorithms: ANN, KNN, and SVM. The input features include: steps per day, dimensionless, overall walking per day, lying per day, and eating per day (Liakos, 2017). In addition, work has been carried out for cow identification using iris and muzzle print images, and the use of ear tags has also been proposed.

Recently, individual cow identification has been carried out using DCNN by providing cropped images of a cow's body using inter-frame differences between two consecutive frames of video. This is done to detect the moving pole location, followed by transforming the inter-frame differencing result into a binary image; finally, a horizontal histogram is applied to the binary image (Zin, 2018). The cow dataset consisted of 45 cows' patterns from the 22 days of video data with a recording frame rate of 30 fps. The DCNN for Cow detection (Figure 21) performed well, even with the cows' body images with rotation variation and under varying illumination conditions.

Figure 21. Cow identification using DCNN
(Zin, 2018)



Body Condition Score (BCS) is a measure used to indicate fat reserves and accumulated energy balance in cows, which is used for predicting mild production, reproduction, and health conditions of cows. CNN has been used to estimate the BCS value (Alvarez, 2018). As part of the image pre-processing phase, cow images were segmented to separate the cow from the background, followed by conversion into one image composed of three channels: depth, Fourier transform, and edge. This pre-processed image was provided as input CNN to predict BCS score.

Traditionally, the animal's behavior and welfare were analysed by experts who watched videos to study the social interactions between cows in dairy farms. However, many parts of the video are of no interest to these experts, making this a very time-consuming task. The parts of the videos with cows were identified using CNN, and the remaining portions were removed so that only the segments of interest to the experts were analysed, saving a lot of time. In Ardo (2018), this objective was achieved using a cow detector, which was carried out in two steps. First, four landmarks—head, front middle, cow's center, and back middle—were detected using landmark CNN. Second, the cow CNN was used to take the landmark CNN output as input and detect the cow and its orientation, which was represented with rotated bounding boxes. The proposed cow detector was able to discard 50% of the recorded video.

Automated identification of cattle breeds is essential in e-livestock management for assisting the government in decision making related to exporting and importing livestock. CNN has been used for the identification of five types of cattle breeds (Bali, Pasuruan, Aceh, Madura, and Pesisir) using GLCM features (Santoni, 2015). Saliency maps were applied to input images to reduce the background so that

GLCM features, which are the input to CNN, were extracted in less time. The results of GLCM-CNN were superior to that of CNN alone and also better when compared to traditional methods like SVM and KNN (without segmentation) and SVM and KNN (Gaussian Mixture Model-Graph Cut with segmentation). In addition to superior performance, the GLCM-CNN also had another advantage, which is that it does not require segmentation.

One of the methods adopted to determine a pig's health status is to study their feeding behavior. Various image processing techniques (e.g., Otsu segmentation, mixed Gaussian model with maximum entropy segmentation, multilevel threshold segmentation, and many more) have been adopted using different feature extractions for this task. In addition, RFID tags attached to a pig's ear were also used to accomplish the task. Recently, Faster RCNN has been explored to understand the feeding behavior of individual pigs (Yang, 2018). First, frames were extracted from video sequences, followed by locating each pig. In the proposed Faster R-CNN Pig Detector (FRPD), the convolution layer was shared by RPN and Fast R-CNN. As part of the first stage of RPN training, fine-tuning was done through the initialization of ZF-net. ZF-net is one of the trained models provided by Faster R-CNN and is used to extract image features. The outcome of the first phase was the Regional proposal network. The output of the previous RPN was the input for Fast RCNN. As part of the second stage of RPN training, the Fast R-CNN detector network was used to initialize RPN, and those layers that were unique to RPN were trained. Non-public layers were fine-tuned in the second stage, which was the training of Fast R-CNN. The FRPD detected both the pig's body with the pig's identification and head. Finally, each detected pig's head was matched to its body using the intersection ratio algorithm.

CNNs have been used for real-time detection of cattle using images captured by drone platform when the flight was in progress (Khan, 2011). One can find out the livestock count using the platform without having knowledge of controlling the ñight of the multirotor. This is because the developed system will cover the user-defined area by automatically adjusting the flight height and bestowing on it the characteristics of the auxiliary camera of the multirotor without needing to attach GPS devices to animals.

Deep Learning for Irrigation

Poor water management is another major issue in agriculture. Likewise, developing proper models for predicting correct water requirements is one of the major challenges in smart agriculture. In (Oliveira, 2018), traditional machine learning algorithms (decision trees; ANN; Systematically Developed Forest of Multiple Trees, or SysFor; SVM; logistic regression; and traditional ETC-based methods) were applied for predicting irrigation water requirements. The input data are composed of various weather parameters, such as maximum and minimum temperature (T-Max and T-Min), wind speed, humidity, rainfall, and solar radiation, combined with soil type, crop type, and crop water usage (to be predicted). Data are pre-processed using Reference Evapotranspiration-based Pre-processing (REP) for estimating daily crop water usage. The proposed method performed better when compared to other tested methods.

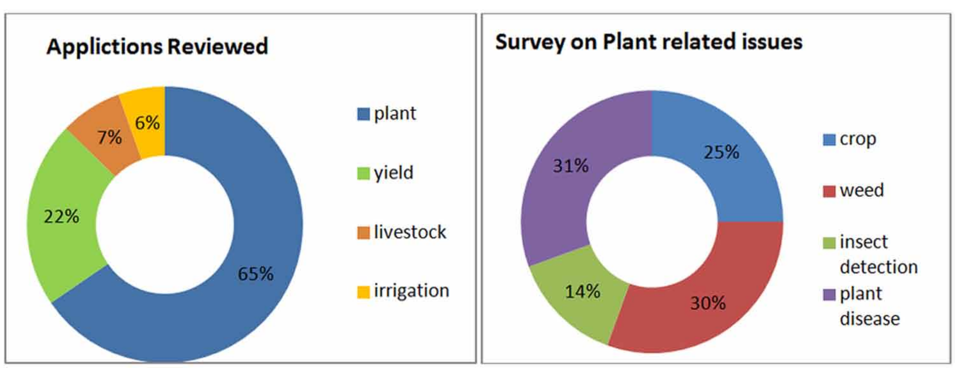
Center pivot irrigation systems (CPIS) are used to irrigate large areas of crops with reductions in both water and labor costs. In Zhang (2018), the author used LeNet, the VGGNet-based network, and the AlexNet network for locating the center of a CPIS with high accuracy from Landsat images. This helped in investigating land use changes, accurately evaluating water intake, and suitably assigning water resources. The LeNet-based network outperformed the other networks.

In Calvo (2009), the authors developed a hybrid model—CNN + fuzzy inference system (FIS) + Genetic Algorithm—for forecasting one-day-ahead daily water demands by farmers. The daily water demand data for 1988, 1989, and 1990 are the inputs to CNN. Data for 1991 were used for CNN validation. The FIS parameters were optimized by GA to improve the forecasting accuracy of CNN. The proposed hybrid model results were better than multivariate and univariate autoregressive NN models. The outcome of the proposed model would be useful for scheduling pumping efforts and reducing operation costs of water dispersal systems.

Soil moisture conditions are one of the factors which can be used to regulate the amount of irrigation needed in an area. Because no data with pairs of aerial images and local soil moisture conditions at the plant level are available, Tseng (2019) generated synthetic data using ðrst-order simulators that modeled the local soil moisture at each plant over time using a discrete time, linear approximation of the Richards equation for soil water ñow. Seven methods—constant prediction baseline, SVM, RF Uncorrelated Plant (RFUP), RF Correlated Field (RFCF), two-layer Neural Networks (NN), Deep CNN Uncorrelated Plant (CNNUP), and DCNN Correlated Field (DCNNCF)—were applied for dissipation rate prediction. The CNN Correlated Field method had the lowest median absolute test error compared to the other methods.

From the papers surveyed, the present study has found that a lot of work has been explored for crop mapping, detection of weeds and plant disease, insect detection, and prediction of crop yield. There is also a large scope of further work that uses smart architecture for water management and livestock care (animal identification and disease detection). The pie chart in Figure 22(a) illustrates the number of papers covering different applications of smart agriculture using Deep Learning algorithms. Figure 22(b) shows the relative percentage of papers surveyed that examine plant-related applications of Deep Learning. Overall, the performance of DL was superior to that of traditional machine learning algorithms (ANN, SVM, KNN, and decision tree). The relative percentage of machine learning methods used in research included in this survey is shown in Figure 23. There is still a lot of work to be carried out by exploring different variants of Deep Learning on the mentioned applications of smart agriculture and many more.

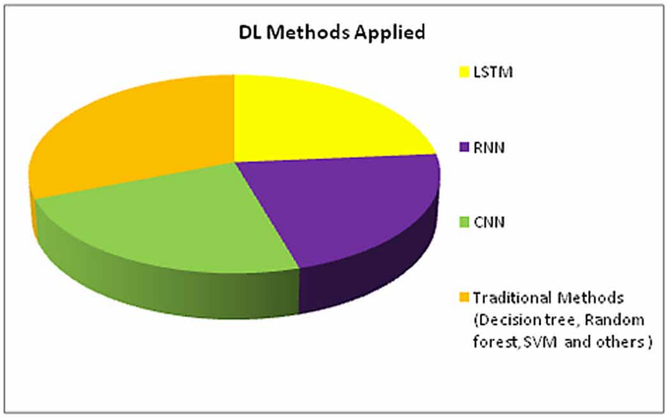
Figure 22. Relative percentage of reviewed papers for this chapter pertaining to: (a) DL for smart agriculture applications, and (b) DL for plant-related issues



CONCLUSION

This detailed survey will help researchers to gain a holistic picture of major DL algorithms applied to various applications of smart agriculture. A few of the major smart agriculture applications covered include: crop mapping and crop identification using satellite images; detection of diseases, weeds, and

Figure 23. Relative percentage of machine learning algorithms applied in smart agriculture surveyed papers for this chapter



insects in plants; yield prediction; prediction of phenotype using genotype; identification of livestock; and water requirement predictions. The survey of DL in various smart agriculture applications, reveal that, DL outperforms the traditional machine learning algorithms: ANN, SVM, KNN, ELM, and others. The survey also reveals that CNN, RNN, and LSTM have been explored in abundance for weed detection, crop identification, and plant disease detection. Yet, there is still a lot to explore for applications including fruit and flower counting, detecting the various stages of crops, and fruit ripening stages. Work can also be carried out in the detection of soil properties to guide the selection of crops to be grown, resulting in higher yield. Further, the survey discloses that there is also a lot of research on livestock disease detection, but almost no work was found on poultry. DL will undisputedly help smart agriculture to attain its major goal: environmentally friendly enhanced food productivity.

REFERENCES

Ahmed, F., Al-Mamun, H. A., Bari, A. S. M. H., Hossain, E., & Kwan, P. (2012). Classification of crops and weeds from digital images: A Support Vector Machine Approach. *Crop Protection (Guildford, Surrey)*, *40*, 98–104. doi:10.1016/j.cropro.2012.04.024

Alessandro dos Santos, F., Freitasa, D. M., Gonçalves da Silvaa, G., Pistorib, H., & Folhesc, M. (2017). Weed detection in soybean crops using ConvNets. *Computers and Electronics in Agriculture*, *143*, 314–324.

Ardö, H. (2018). Convolutional NN-based cow interaction watchdog. *IET Computer Vision*, *12*(2), 171–177. doi:10.1049/iet-cvi.2017.0077

Arivazhagan, S., & Vineth Ligi, S. (2018)Mango Leaf Diseases Identification Using Convolutional NN. *International Journal of Pure and Applied Mathematics*, *120*(6), 1067–11079.

Atanbori, J., Chen, F., French, A., & Pridmore, T. (2018). Towards Low-Cost Image-based Plant Phenotyping using Reduced-Parameter CNN. *Proceedings of the Computer Vision Problems in Plant Phenotyping (CVPPP).*

Bah, M. D., Hafinane, A., & Canals, R. (2018). Deep learning with unsupervised data labelling forweed detection in line crops in UAV images. Remote Sens, 10, 1690-1705.

Bakhshipour, A., & Jafari, A. (2018, February). Evaluation of SVMand artificial NNs in weed detection using shape features. *Computers and Electronics in Agriculture*, *145*, 153–160. doi:10.1016/j.compag.2017.12.032

Bakhshipour, A., Jafari, A., Nassiri, S. M., & Zare, D. (2017). Weed segmentation using texture features extracted from wavelet sub-images. *Biosystems Engineering*, *157*, 1–12. doi:10.1016/j.biosystemseng.2017.02.002

Bouroubi, Y., Bugnet, P., Nguyen-Xuan, T., Gosselin, C., Bélec, C., Longchamps, L., & Vigneault, P. (2018). Pest Detection on UAV Imagery using a Deep Convolutional Neural Network. *Proceedings of the 14th International Conference on Precision Agriculture*.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent NNs on sequence modeling. arXiv preprintarXiv:1412.3555v1 [cs.NE]

Deng, L. (2014). A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing*, *3*, 1–29. doi:10.1017/atsip.2013.9

Dian, M. (2018). *DL with unsupervised data labelling for weeds detection on UAV images*. doi:10.20944/preprints201809.0088.v1

Fuentes, A., Yoon, S., Kim, S. C., & Park, D. S. (2017). A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. *Sensors (Basel)*, *17*(9), 2022. doi:10.339017092022 PMID:28869539

Hasan. (2018). Detection and analysis of wheat spikes using Convolutional NNs. *Plant Methods*, doi:10.118613007-018-0366-8

Huang, H., Deng, J., & Lan, Y. (2018). A fully convolutional network for weed mapping of unmanned aerial vehicle (UAV) imagery. *PLoS One*, *26*(April). doi:10.1371/journal.pone.0196302 PMID:29698500

Jagannath, K. (2018). Automatic semantic segmentation and classification of remote sensing data for agriculture. *Mathematical Models in Engineering*, *4*(2), 112–137. doi:10.21595/mme.2018.19840

Jeon, H. Y., Tian, L. F., & Zhu, H. (2011). Robust crop and weed segmentation under uncontrolled outdoor illumination. *Sensors (Basel)*, *11*(6), 6270–6283. doi:10.3390110606270 PMID:22163954

Ji, S., Zhang, C., Xu, A., Shi, Y., & Duan, Y. (2018). 3D Convolutional Neural Networks for Crop Classification with Multi-Temporal Remote Sensing Images. *Remote Sensing*, *10*(2), 75. doi:10.3390/rs10010075

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018, April). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, *147*, 70–90. doi:10.1016/j.compag.2018.02.016

Keresztes, B., Abdelghafour, F., Randriamanga, D., Da Costa, J.-P., & Germain, C. (2018). Real-time Fruit Detection Using Deep NN. *14th International Conference on Precision Agriculture*, Montreal, Canada.

Khan, Islam, & Hafeez. (2011). Irrigation Water Requirement Prediction through Various Data Mining Techniques Applied on a Carefully Pre-processed Dataset. *CRPIT*, 121.

Kussul, N., Lavreniuk, M., Skakun, S., & Shelestov, A. (2017, May). DL Classification of Land Cover and Crop Types Using Remote Sensing Data. *IEEE Geoscience and Remote Sensing Letters*, *14*(5), 778–782. doi:10.1109/LGRS.2017.2681128

LeCun, Y., & Bengio, Y. (1995). Convolutional networks for images, speech, and time series. In M. A. Arbib (Ed.), *The Handbook of Brain Theory and NNs* (pp. 255–258). Cambridge, MA: MIT Press.

Li, W., Fu, H., Yu, L., & Cracknell, A. (2017). Deep learning Based Oil Palm Tree Detection and Counting for High Resolution Remote Sensing Images. *Remote Sensing*, *9*(1), 22. doi:10.3390/rs9010022

Liakos, K., Moustakidis, S., Tsiotra, G., Bartzanas, T., Bochtis, D., & Parisses, C. (2017). Machine Learning Based Computational Analysis Method for Cattle Lameness Prediction. *8th International Conference on Information and Communication Technology in Agriculture, Food and Environment*.

Liu, Chen, Aditya, Sivakumar, Dcunha, Qu, … Kumar. (2018). *Robust Fruit Counting: Combining Deep Learning, Tracking, and Structure from Motion*. arXiv:1804.00307v2 [cs.CV]

Liu, B., Zhang, Y., & He, D. J. (2018). Yuxiang, Identification of Apple Leaf Diseases Based on Deep Convolutional NNs. *Symmetry*, *10*, 11. doi:10.3390ym10010011

Lottes, P., Khanna, R., Pfeifer, J., Siegwart, R., & Stachniss, C. (2017). *UAV-Based Crop and Weed Classification for Smart Farming*. IEEE International Conference on Robotics and Automation (ICRA), Singapore.

Lu, Y., Yi, S., Zeng, N., Liu, Y., & Zhang, Y. (2017). Identiðcation of rice diseases using deep convolutional NNs. *Neurocomputing*, *267*, 378–384. doi:10.1016/j.neucom.2017.06.023

Ma, W., Qiu, Z., Song, J., Cheng, Q., Ma, C., & Deep, G. S. (2018). Predicting phenotypes from genotypes using Deep Learning. *Planta*. doi:10.100700425-018-2976-9

Maa, J., Dua, K., Zhenga, F., Zhangb, L., Gongc, Z., & Suna, Z. (2018). A recognition method for cucumber diseases using leaf symptom images based on deep convolutional NN. *Computers and Electronics in Agriculture*, *154*, 18–24. doi:10.1016/j.compag.2018.08.048

Namin, S. T., Esmaeilzadeh, M., Najafi, M., Brown, T. B., & Borevitz, J. O. (2018). DL for temporal phenotype/genotype classification. *Plant Methods*, *14*, 66. doi:10.118613007-018-0333-4

Nazri, A., Mazlan, N., & Muharam, F. (2018). PENYEK: Automated brown planthopper detection from imperfect sticky pad images using deep convolutional NN. *PLoS One*, *13*(12), e0208501. doi:10.1371/journal.pone.0208501 PMID:30571683

Ndikumana, E., Dinh, H. T. M., Baghdadi, N., Courault, D., & Hossard, L. (2018). Deep Recurrent NN for Agricultural Classification using multitemporal SAR Sentinel-1 for Camargue, France. *Remote Sensing*, *10*(8), 1217. doi:10.3390/rs10081217

Oleh. (n.d.). *Automatic Plant Counting using Deep NNs* (Master of Science thesis). http://er.ucu.edu.ua/handle e/1/1334

Oliveira, Cunha, Silva, & Netto. (n.d.). *A Scalable Machine Learning System for Pre-Season Agriculture Yield Forecast*. arXiv:1806.09244

Olsen, Konovalov, Philippa, Ridd, Wood, Johns, … White. (2019). DeepWeeds: A Multiclass Weed Species Image Dataset for DL. *Scientific Reports, 9*(2058). 2 doi:10.103841598-018-38343-3

Pascanu, R., Gulcehre, C., Cho, K., & Bengio, Y. (2013). *How to construct deep recurrent NNs*. arXiv preprint arXiv:1312.6026v5[cs.NE]

Pulido-Calvo, I., & Gutierrez-Estrada, J. C. (2009). Improved irrigation water demand forecasting using a soft-computing hybrid model. *Biosystems Engineering*, *102*(2), 202–218. doi:10.1016/j.biosystemseng.2008.09.032

Rahmana, Arkoa, Alia, Khanb, Wasifa, Janic, & Kabirb. (2018). *Identification and Recognition of Rice Diseases and Pests 32 Using Deep Convolutional NNs*. Submitted to Expert Systems with Applications.

Rahnemoonfar, M., Sheppard, C., & Count, D. (2017). Fruit Counting Based on Deep Simulated Learning. *Sensors (Basel)*, *17*(4), 905. doi:10.339017040905 PMID:28425947

Rivas, A., Chamoso, P., González-Briones, A., & Corchado, J. M. (2018). Detection of Cattle Using Drones and Convolutional NNs. *Sensors (Basel)*, *18*(7), 2048. doi:10.339018072048

Rodríguez Alvarez, J. (2018). *Body condition estimation on cows from depth images using Convolutional NNs* (Vol. 155). Comput. Electron. Agric.

Russello, H. (2018). *Convolutional NNs for Crop Yield Prediction using Satellite Images* (MSc Artificial Intelligence Master Thesis). University of Amsterdam.

Santoni, M. M., Sensuse, D. I., Arymurthy, A. M., & Fanany, M. I. (2015). Cattle race classification using gray level co-occurrence matrix convolutional NNs. *Procedia Computer Science*, *59*, 493–502. doi:10.1016/j.procs.2015.07.525

Shook, J. (2018). Integrating genotype and weather variables for soybean yield prediction using deep learning. *bioRxiv*, *25*. doi:10.1101/331561doi

Sladojevic, Arsenovic, Anderla, Culibrk, & Stefanovic. (2016). Deep NNs Based Recognition of Plant Diseases by Leaf Image Classification. *Computational Intelligence and Neuroscience.* doi:10.1155/2016/3289801

Teimouri, N., Dyrmann, M., Nielsen, P. R., Mathiassen, S. K., Somerville, G. J., & Jørgensen, R. N. (2018). Weed Growth Stage Estimator Using Deep Convolutional NNs. *Sensors (Basel)*, *18*(5), 1580. doi:10.339018051580 PMID:29772666

Tooa, E. C., Yujiana, L., Njukia, S., & Yingchunb, L. (2018). A comparative study of ðne-tuning DL models for plant disease identiðcation. *Computers and Electronics in Agriculture*. doi:10.1016/j.compag.2018.03.032]

Tseng, D., Wang, D., Chen, C., Miller, L., Song, W., Viers, J., . . . Towards, A. P. I. (2018). *DL to Infer Local Soil Moisture Conditions from Synthetic Aerial Agricultural Images*. In 2018 IEEE 14th International Conference on Automation Science and Engineering, CASE, Munich, Germany.

Türkoğlu, M. (n.d.). Plant disease and pest detection using deep learning-based features. *Turkish Journal of Electrical Engineering and Computer Sciences*. doi:10.3906/elk-1809-181]

Xia, D., Chen, P., Wang, B., Zhang, J., & Xie, C. (2018). Insect Detection and Classification Based on an Improved Convolutional NN. *Sensors (Basel)*, *18*(12), 4169. doi:10.339018124169 PMID:30486481

Xiao, Q., Li, W., Chen, P., & Wang, B. (2018). Prediction of Crop Pests and Diseases in Cotton by Long Short Term Memory Network. *ICIC*, *2018*, 11–16. doi:10.1007/978-3-319-95933-7_2

Yang, Xiao, & Lin. (2018). Feeding behaviour recognition for group-housed pigs with the FasterR-CNN. Computers and Electronics in Agriculture, 155, 453–460.

You, J., Li, X., Low, M., Lobell, D., & Ermon, S. (2017). Deep Gaussian Process for Crop Yield Prediction Based on Remote Sensing Data. AAAI, 4559–4566.

Zhang, C., Yue, P., Di, L., & Wu, Z. (2018). Automatic Identiðcation of Center Pivot Irrigation Systems from Landsat Images Using Convolutional Neural Networks. *Agriculture*, *8*(10), 147–156. doi:10.3390/agriculture8100147

Zhang, M., Lin, H., Wang, G., Sun, H., & Fu, J. (2018, November 20). Mapping Paddy Rice Using a Convolutional NeuralNetwork (CNN) with Landsat Datasets in the Dongting Lake Area, China. *Remote Sensing*, *10*(11), 1840. doi:10.3390/rs10111840

Zhang, M., Lin, H., Wang, G., Sun, H., & Fu, J. (2018). Mapping Paddy Rice Using a Convolutional NeuralNetwork (CNN) with Landsat 8 Datasets in the Dongting Lake Area, China. *Remote Sensing*, *10*(11), 1840. doi:10.3390/rs10111840

Zhang, X., Qiao, Y., Meng, F., Fan, C., & Zhang, M. (2018). Identification of Maize Leaf Diseases Using Improved Deep Convolutional NNs. *IEEE Access: Practical Innovations, Open Solutions*, *6*. doi:10.1109/ACCESS.2018.2844405]

Zhong, Y., Gao, J., Lei, Q., & Yao Zhou, A. (2018). Vision-Based Counting and Recognition System for Flying Insects in Intelligent Agriculture. *Sensors (Basel)*, *18*(5), 1489. doi:10.339018051489 PMID:29747429

Zhonga, L., Hub, L., & Zhouc, H. (2019). DL based multi-temporal crop classification. *Remote Sensing of Environment*, *221*, 430–443. doi:10.1016/j.rse.2018.11.032

Zhu, L.-Q., Ma, M.-Y., Zhang, Z., Zhang, P.-Y., Wu, W., Wang, D.-D., ... Wang, H.-Y. (2017). Hybrid DL for automated lepidopteran insect image classification. *Oriental Insects*, *51*(2), 79–91. doi:10.1080/00305316.2016.1252805

Zhu, N. Y., Liu, X., Liu, Z. Q., Hu, K., Wang, Y. K., Tan, J. L., ... Guo, Y. (2018). Deep learning for smart agriculture: Concepts, tools, applications, and opportunities. *International Journal of Agricultural and Biological Engineering*, *11*(4), 32–44. doi:10.25165/j.ijabe.20181104.4475

Zin. (2018). Image Technology based Cow Identification System Using DL. *Proceedings of the International MultiConference of Engineers and Computer Scientists 2018*, 1.

KEY TERMS AND DEFINITIONS

Aerial Image: Photos or images snaps taken from airplanes, UAVs, or satellite to assist in remote analysis of field.

Contour Map: Combination of intensity or yield level by kiging or interpolating.

Geo-Stationary Satellite: An orbital path of a satellite synchronized with earth's orbit.

Remote Sensing: Monitoring objects without any direct contact between sensor and object.

Soil Map: Graph with proper indication of soil properties such as texture, fertility, pH, organic matter, and others.

UAVs: Unmanned aerial vehicles, in common language known as drone without human pilot.

VRA: Variable rate Application, adjustment of the amount of crop input such as seed, fertilizer, pesticides to match conditions in a field.

Zone Management: Information based split up of land into smaller areas for specific application management.

Compilation of References

A tool for reverse engineering Android apk files. (n.d.). Retrieved from https://ibotpeaches.github.io/Apktool/

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., & Dean, J. (2016). Tensorflow: A system for large-scale machine learning. *12th {USENIX} Symposium on Operating Systems Design and Implementation*, 265-283.

Abomhara, M. (2015). Cyber security and the internet of things: Vulnerabilities, threats, intruders and attacks. *Journal of Cyber Security and Mobility*, *4*(1), 65–88. doi:10.13052/jcsm2245-1439.414

AboutI. D. A. (n.d.). Retrieved from https://www.hex-rays.com/products/ida/

Abrantes, G. A. (1999). MPEG-4 facial animation technology: Survey, implementation, and results. *Circuits and Systems for Video Technology*. *IEEE Transactions on*, *9*(2), 290–305.

Abu Alsheikh, M., Lin, S., Niyato, D., & Tan, H.-P. (2016). Rate-distortion balanced data compression for wireless sensor networks. *IEEE Sensors Journal*, *16*(12), 5072–5083. doi:10.1109/JSEN.2016.2550599

Adam-Bourdarios, C., Cowan, G., Germain, C., Guyon, I., Kégl, B., & Rousseau, D. (2015, August). The Higgs boson machine learning challenge. In *NIPS 2014 Workshop on High-energy Physics and Machine Learning* (pp. 19-55). Academic Press.

Agarwal, Y., & Dey, A. K. (2016). Toward building a safe, secure, and easy-to-use internet of things infrastructure. *IEEE Comput.*, *49*(4), 88–91. doi:10.1109/MC.2016.111

Aggarwal, Ashish, & Sheth. (2013). *The Internet Of Things: A Survey From The Data-Centric Perespective, Managing And Mining Sensor Data*. Springer Science+Business Media.

Agrawal, S., & Vieira, D. (2013). A Survey On Internet Of Things. *Abakós, Belo Horizonte VOL.*, *1*(2), 78–95.

Ahlberg, J. (2002). An active model for facial feature tracking. *EURASIP Journal on Applied Signal Processing*, ▪▪▪, 566–571.

Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications, 60*, 19–31. Available: https://www.sciencedirect.com/science/article/pii/S1084804515002891

Ahmed, F., Al-Mamun, H. A., Bari, A. S. M. H., Hossain, E., & Kwan, P. (2012). Classification of crops and weeds from digital images: A Support Vector Machine Approach. *Crop Protection (Guildford, Surrey)*, *40*, 98–104. doi:10.1016/j.cropro.2012.04.024

Aksenova, S. S. (2004). Machine learning with Weka. *Weka Explorer Tutorial for WEKA Version*, *3*(4), 2–44.

Aleksic, P. S., & Katsaggelos, A. K. (2006). Automatic facial expression recognition using facial animation parameters and multistream HMMs. *Information Forensics and Security*. *IEEE Transactions on*, *1*(1), 3–11.

Alessandro dos Santos, F., Freitasa, D. M., Gonçalves da Silvaa, G., Pistorib, H., & Folhesc, M. (2017). Weed detection in soybean crops using ConvNets. *Computers and Electronics in Agriculture*, *143*, 314–324.

Alkhateeb, A., Alex, S., Varkey, P., Li, Y., Qu, Q., & Tujkovic, D. (2018). Deep Learning Coordinated Beamforming for Highly-Mobile Millimeter Wave Systems. *IEEE Access: Practical Innovations, Open Solutions*, *6*, 37328–37348. doi:10.1109/ACCESS.2018.2850226

Alodini, Q. (2012). Prevalence of Hepatitis B Virus (HBV) and Hepatitis C Virus (HCV) infections among blood donors at Al-Thawra Hospital Sana'a City-Yemen. *Yemeni Journal for Medical Sciences*, *6*, 16–20.

Alpaydin, E. (2004). *Introduction to Machine Learning*. Cambridge: MIT Press.

Alwaisi, S., & Baykan, O. K. (2017). Training Of Artificial Neural Network Using Metaheuristic Algorithm. *International Journal of Intelligent Systems and Applications in Engineering*.

Amato, G., Carrara, F., Falchi, F., Gennaro, C., Meghini, C., & Vairo, C. (2016). Deep learning for decentralized parking lot occupancy detection. *Expert Systems with Applications*, 72.

Aminanto, M. E., & Kim, K. (2016). Detecting impersonation attack in WiFi networks using deep learning approach. *Proc. International Workshop on Information Security Applications*, 136–147.

Aminian, M., & Aminian, F. (2000). Neural-network based analog-circuit fault diagnosis using wavelet transform as preprocessor. *IEEE Transactions on Circuits and Systems. 2, Analog and Digital Signal Processing*, *47*(2), 151–156. doi:10.1109/82.823545

AndroBugs framework. (n.d.). Retrieved from https://github.com/AndroBugs/AndroBugs_Framework

Android Malware Dataset. (n.d.). Retrieved from http://amd.arguslab.org/

Angelov, P., & Sperduti, A. (2016). Challenges in deep learning. *24th European Symposium on Artificial Neural Networks*, 489-495.

Angermueller, C., Lee, H., Reik, W., & Stegle, O. (2016). Accurate prediction of single-cell dna methylation states using deep learning. *bioRxiv*, 055715.

Anthimopoulos, M., Christodoulidis, S., Ebner, L., Christe, A., & Mougiakakou, S. (2016, May). Lung pattern classification for interstitial lung diseases using a deep convolutional neural network. *IEEE Transactions on Medical Imaging*, *35*(5), 1207–1216. doi:10.1109/TMI.2016.2535865

Ardö, H. (2018). Convolutional NN-based cow interaction watchdog. *IET Computer Vision*, *12*(2), 171–177. doi:10.1049/iet-cvi.2017.0077

Arivazhagan, S., & Vineth Ligi, S. (2018)Mango Leaf Diseases Identification Using Convolutional NN. *International Journal of Pure and Applied Mathematics*, *120*(6), 1067–11079.

Asaduzzaman, A., Martinez, A., & Sepehri, A. (2015). *A time efficient image processing algorithm for multicore/many core parallel computing*. IEEE Southeast Conference Fort Lauderdale.

Ashtari, S., & Eydgahi, A. (2017). Student perceptions of cloud applications effectiveness in higher education. *Journal of Computational Science*, *23*, 173–180. doi:10.1016/j.jocs.2016.12.007

Atanbori, J., Chen, F., French, A., & Pridmore, T. (2018). Towards Low-Cost Image-based Plant Phenotyping using Reduced-Parameter CNN. *Proceedings of the Computer Vision Problems in Plant Phenotyping (CVPPP)*.

Audio (MPEG Mtg). M. V. (1998). ISO/MPEG N2503. Atlantic City.

Avendi, M., Kheradvar, A., & Jafarkhani, H. (2016). Acombined deep-learning and deformable-model approach to fully automatic segmentation of the left ventricle in cardiac mri. *Medical Image Analysis*, *30*, 108–119. doi:10.1016/j.media.2016.01.005

Babu, G. S., Zhao, P., & Li, X.-L. (2016). Deep convolutional neural network based regression approach for estimation of remaining useful life. In *International Conference on Database Systems for Advanced Applications*. Springer. 10.1007/978-3-319-32025-0_14

Badler, S. P. (1981). Animating facial expressions. *Computer Graphics*, *15*(3), 245–252. doi:10.1145/965161.806812

Baeldung. (2018). *A Guide to Deeplearning4j*. Retrieved may 31, 2019 from https://www.baeldung.com/deeplearning4j

Bah, M. D., Hafinane, A., & Canals, R. (2018). Deep learning with unsupervised data labelling forweed detection in line crops in UAV images. Remote Sens, 10, 1690-1705.

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *International Conference on Learning Representations (ICLR 2015)*.

Bahrampour, S., Ramakrishnan, N., Schott, L., & Shah, M. (2016). Comparative study of caffe, neon, theano, and torch for deep learning. *Proceedings of the International Conference on Learning Representations*, 1-11.

Bai, D. P., & Preethi, P. (2016). Security enhancement of health information exchange based on cloud computing system. *International Journal of Scientific and Engineering Research*, *4*(10), 79–82.

Baizhen, Z. a. (2006). Facial feature extraction using improved deformable templates. *Signal Processing, 2006 8th International Conference on, 4*.

Bakhshipour, A., & Jafari, A. (2018, February). Evaluation of SVMand artificial NNs in weed detection using shape features. *Computers and Electronics in Agriculture*, *145*, 153–160. doi:10.1016/j.compag.2017.12.032

Bakhshipour, A., Jafari, A., Nassiri, S. M., & Zare, D. (2017). Weed segmentation using texture features extracted from wavelet sub-images. *Biosystems Engineering*, *157*, 1–12. doi:10.1016/j.biosystemseng.2017.02.002

Banihashemian, S. S., Adibnia, F., & Mehdi, A. (2018). Sarram. A new range-free and storage-efficient localization algorithm using neural networks in wireless sensor networks. *Wireless Personal Communications*, *98*(1), 1547–1568. doi:10.100711277-017-4934-4

Bartlett, M. S., Littlewort, G., Fasel, I., & Movellan, J. R. (2003, June). Real Time Face Detection and Facial Expression Recognition: Development and Applications to Human Computer Interaction. *Computer Vision and Pattern Recognition Workshop,2003. CVPRW '03. Conference on, 5*(53), 16-22.

Bedossa, B., & Carrat, F. (2009). Liver biopsy: The best, not the gold standard. *Journal of Hepatology*, *50*(1), 1–3. doi:10.1016/j.jhep.2008.10.014 PMID:19017551

Bengio, Y., Delalleau, O., & Le Roux, N. (2005). The curse of highly variable functions for local kernel machines. Proc. Advances in Neural Information Processing Systems, 18, 107–114.

Bengio, Y. (2009). Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, *2*(1), 1–127. doi:10.1561/2200000006

Bengio, Y., & LeCun, Y. (2007). Scaling learning algorithms towards, AI. In L. Bottou, O. Chapelle, D. DeCoste, & J. Weston (Eds.), *Large Scale Kernel Machines* (Vol. 34, pp. 321–360). Cambridge, MA: MIT Press. http://www.iro.umontreal.ca/~ lisa/pointeurs/bengio+lecun_chapter2007.pdf

Bernas, M., & Płaczek, B. (2015). Fully connected neural networks ensemble with signal strength clustering for indoor localization in wireless sensor networks. *International Journal of Distributed Sensor Networks*, *11*(12), 403242. doi:10.1155/2015/403242

Bertino, E., & Islam, N. (2017). Botnets and internet of things security. *Computer*, *50*(2), 76–79. doi:10.1109/MC.2017.62

Bhargavi, K., & Babu, B. S. (2017). Accelerating the Big Data Analytics by GPU-Based Machine Learning: A Survey. *International Symposium on Sensor Networks, Systems and Security*, 63-83.

Bidnessetc.com. (2014). *Internet of Things: Connecting (And Disrupting) Your Universe*. Available at: https://www.bidnessetc.com/business/internet-of-things-connecting-and-disrupting-your-universe/

Bishop, C. (1992). Pattern recognition and machine learning. Springer.

Bonacini, M., Hadi, G., Govindarajan, S., & Lindsay, K. L. (1997). Utility of a discriminant score for diagnosing advanced fibrosis or cirrhosis in patients with chronic hepatitis C virus infection. *The American Journal of Gastroenterology*, *92*, 1302–1304. PMID:9260794

Bordes, A., Chopra, S., & Weston, J. (2014). Question answering with subgraph embeddings. In *Proc. Empirical Methods in Natural Language Processing*. https://arxiv.org/abs/1406.3676v3

Bordes, A., Glorot, X., Weston, J., & Bengio, Y. (2012)/ Joint learning of words and meaning representations for open-text semantic parsing. *International Conference on Artificial Intelligence and Statistics*, 127–135.

Borgerding, M., Schniter, P., & Rangan, S. (2017). AMP-inspired deep networks for sparse linear inverse problems. *IEEE Transactions on Signal Processing*, *65*(16), 4293–4308. doi:10.1109/TSP.2017.2708040

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In Y. Lechevallier, & G. Saporta (Eds.), *Proceedings of COMPSTAT'2010, Physica-Verlag HD*. 10.1007/978-3-7908-2604-3_16

Bottou, L., & Bousquet, O. (2007). The tradeoffs of large scale learning. n Proc. *Advances in Neural Information Processing Systems*, *20*, 161–168.

Bouroubi, Y., Bugnet, P., Nguyen-Xuan, T., Gosselin, C., Bélec, C., Longchamps, L., & Vigneault, P. (2018). Pest Detection on UAV Imagery using a Deep Convolutional Neural Network. *Proceedings of the 14th International Conference on Precision Agriculture*.

Brosch, T. (2013). Manifold learning of brain mris by deep learning. *Proc. MICCAI*, 633–640. 10.1007/978-3-642-40763-5_78

BulwerJ. (1649). Retrieved from https://www.library.northwestern.edu/spec/hogarth/physiognomics1.html

Buniyamin, N., Mat, U. b., & Arshad, P. M. (2015, November). Educational Data Mining for Prediction and Classification of Engineering Students Achievement. *IEEE 7th International Conference on Engineering Education (ICEED)*, (pp. 49-53). Kanazawa: IEEE.

Cambria, E., Huang, G. B., Kasun, L. L. C., Zhou, H., Vong, C. M., Lin, J., ... Liu, J. (2013, November). Extreme learning machines [trends controversies]. *IEEE Intelligent Systems*, *28*(6), 30–59. doi:10.1109/MIS.2013.140

Cao. (2016). Improving tuberculosis diagnostics using deep learning and mobile health technologies among resource-poor and marginalized communities. *IEEE Connected Health, Appl., Syst. Eng. Technol.*, 274–281.

Capello, F., Toja, M., & Trapani, N. (2016). A real-time monitoring service based on industrial internet of things to manage agrifood logistics. *Proceedings of 6th International Conference on Information Systems, Logistics and Supply Chain*, 1-8.

Castera, L. (2012). Noninvasive methods to assess liver disease in patients with hepatitis B or C. *Gastroenterology*, *142*(6), 1293–1302. doi:10.1053/j.gastro.2012.02.017 PMID:22537436

Challita, U., Saad, W., & Bettstetter, C. (2018). Deep reinforcement learning for interference-aware path planning of cellular connected UAVs. *Proc. IEEE International Conference on Communications (ICC)*. 10.1109/ICC.2018.8422706

Chang, C.-C. J., & Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*(3), 1–27. doi:10.1145/1961189.1961199

Che, Purushotham, Khemani, & Liu. (2015). *Distilling knowledge from deep networks with applications to healthcare domain*. ArXiv e-prints.

Chen, Lingys, Chen, & Liu. (2018). AuTO: scaling deep reinforcement learning for datacenter-scale automatic traffic optimization. *Proc. ACM Conference of the ACM Special Interest Group on Data Communication (SIGCOMM)*, 191–205.

Cheng, J.-Z., Ni, D., Chou, Y.-H., Qin, J., Tiu, C.-M., Chang, Y.-C., ... Chen, C.-M. (2016). Computer-aided diagnosis with deep learning architecture: Applications to breast lesions in us images and pulmonary nodules in ct scans. *Scientific Reports*, *6*(1), 24454. doi:10.1038rep24454

Cheng, Y. (1995). Mean shift, mode seeking, and clustering. *Pattern Analysis and Machine Intelligence*. *IEEE Transactions on*, *17*(8), 790–799.

Chen, H., Ni, D., Qin, J., Li, S., Yang, X., Wang, T., & Heng, P. A. (2015, September). Standard plane localization in fetal ultrasound via domain transferred deep neural networks. *IEEE Journal of Biomedical and Health Informatics*, *19*(5), 1627–1636. doi:10.1109/JBHI.2015.2425041

Chen, L., Yang, D., Zhang, D., Wang, C., Li, J., & Nguyen, T.-M.-T. (2018). Jonathan Li, and Thi-Mai-Trang Nguyen. Deep mobile traffic forecast and complementary base station clustering for C-RAN optimization. *Journal of Network and Computer Applications*, *121*, 59–69. doi:10.1016/j.jnca.2018.07.015

Chen, M., Ju, C. J. T., Zhou, G., Zhang, T., Chen, X., Chang, K. W., & Wang, W. (2018). Lasagna: Multifaceted Protein-Protein Interaction Prediction Based on Siamese Residual RCNN. *bioRxiv*, 501791.

Chen, S., Xu, H., Liu, D., Hu, B., & Wang, H. (2014, August). A Vision of IoT: Applications, Challenges,and Opportunities With China Perspective. *IEEE Internet of Things Journal, 1*(4), 349–359. doi:10.1109/JIOT.2014.2337336

Chen, X., Vorvoreanu, M., & Madhavan, K. (2013). Mining Social Media Data for Understanding Students' Learning Experiences. *IEEE Transactions on Learning Technologies*.

Chen, X.-W., & Lin, X. (2014). Big data deep learning: Challenges and perspectives. *IEEE Access: Practical Innovations, Open Solutions*, *2*, 514–525. doi:10.1109/ACCESS.2014.2325029

Chen, Z., & Smith. (2018). Heterogeneous machine-type communications in cellular networks: Random access optimization by deep reinforcement learning. *Proc. IEEE International Conference on Communications (ICC)*, 1–6.

Chen, Z., Li, C., & Sanchez, R.-V. (2015). *Gearbox fault identification and classification with convolutional neural networks* (Vol. 2015). Shock and Vibration.

Chen, Z., & Li, W. (2017). *Multi-sensor feature fusion for bearing fault diagnosis using sparse auto encoder and deep belief network*. *IEEE Transactions on IM*.

Chetlur, S., Woolley, C., Vandermersch, P., Cohen, J., Tran, J., Catanzaro, B., & Shelhamer, E. (2014*). cudnn: Efficient primitives for deep learning*. arXiv preprint arXiv:1410.0759

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. arXiv:1406.1078

Choi, S., Kim, E., & Oh, S. (2013). Human behavior prediction for smart homes using deep learning. *Proc. IEEE RO-MAN*, 173–179.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). *Empirical evaluation of gated recurrent neural networks on sequence modeling*. arXiv preprint arXiv:1412.3555v1 [cs.NE]

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent NNs on sequence modeling. arXiv preprintarXiv:1412.3555v1 [cs.NE]

Ciodaro, T., Deva, D., de Seixas, J., & Damazio, D. (2012). Online particle detection with neural networks biased on topological calorimetry Information. *Journal of Physics: Conference Series*, *368*, 012030. doi:10.1088/1742-6596/368/1/012030

Collobert, R., & Bengio, S. (2004). Links between perceptrons, MLPs and SVMs. *Proc. twenty-first ACM international conference on Machine learning*, 23. 10.1145/1015330.1015415

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, *12*(Aug), 2493–2537.

Cootes, T. F., G. E. (1998). A comparative evaluation of active appearance model algorithms. *British Machine Vision Conference*, 2, 680–689. 10.5244/C.12.68

Costa, E., Fonseca, B., Santana, M. A., Araújo, F., & Rego, J. (2017). Evaluating the effectiveness of educational data mining techniques for early prediction of students' academic failure in introductory programming courses. *Computers in Human Behavior*, *73*, 247–256. doi:10.1016/j.chb.2017.01.047

Costa, K. A. P., & Pereira, R. Y. M. (2015). *A nature-inspired approach to speed up optimum-path forest clustering and its application to intrusion detection in computer networks* (Vol. 294). Information Sciences. Available https://www.sciencedirect.com/science/article/pii/S0020025514009311

Cowie, R. E. (2008). Recognition of emotional states in natural human-computer interaction. In *Multimodal user interfaces.* Springer Berlin Heidelberg.

Crankshaw, D., Wang, X., Zhou, G., Franklin, M. J., Gonzalez, J. E., & Stoica, I. (2017). Clipper: A low-latency online prediction serving system. In *14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 17)* (pp. 613-627). USENIX.

Crisan, D. (2012). Prospective noninvasive follow-up of liver fibrosis in patients with chronic hepatitis C. *Journal of Gastrointestinal and Liver Diseases; JGLD*, *21*, 375–382. PMID:23256120

Cristinacce, D. A. (2006). Feature Detection and Tracking with Constrained Local Models. *British Machine Vision Conference*, *2*(5). 10.5244/C.20.95

Cristinacce, D. A. (2007). Boosted Regression Active Shape Models. *British Machine Vision Conference*.

Cusumano, M.A. (2005). Google: What it is and what it is not. *Commun ACM - Med Image Moeling, 48*(2), 15–17. doi:10.1145/1042091.1042107

Dabboor, M., Yackel, J., Hossain, M., & Braun, A. (2013). Comparing matrix distance measures for unsupervised POL-SAR data classification of sea ice based on agglomerative clustering. *International Journal of Remote Sensing*, *34*(4), 1492–1505. doi:10.1080/01431161.2012.727040

Dangare, C., & Apte, S. (2012). A data mining approach for prediction of heart disease using neural networks. *International Journal of Computer Engineering and Technology*, *3*(3).

Daniel, E. (2013). O'Leary. Artificial intelligence and big data. *IEEE Intelligent Systems*, *28*(2), 96–99. doi:10.1109/MIS.2013.39

Darwin, C. (1904). The Expression of the Emotions in Man and Animals (2nd ed.; J. Murray, Ed.). London.

Davidt, O. E., & Netanyahu, N. S. (2015, July). DeepSign: Deep Learning for Automatic Malware Detection. *International Joint Conference on Neural Networks (IJCNN)*, (pp. 1-8). , 978-1-4799-1959-8/1510.1109/IJCNN.2015.7280815

Davis, D. (1998). The virtual university: A learning university. *Journal of Workplace Learning*, *10*(4), 175–213. doi:10.1108/13665629810213935

Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Le, Q., . . . Ng, A. (2012). Large scale distributed deep networks. In P. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems, (Vol. 25, pp. 1232–1240). http://books.nips.cc/papers/files/nips25/NIPS2012_0598.pdf

Deng, L., & Yu, D. (2014). *Deep Learning for Signal and Information Processing*. Redmond, WA: Microsoft Research.

Deng, L. (2014). A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing*, *3*, 1–29. doi:10.1017/atsip.2013.9

Deng, L., & Yu, D. (2014). *Deep learning: methods and applications*. Foundations and Trends in Signal Processing.

Deng, Y., Ren, Z., Kong, Y., Bao, F., & Dai, Q. (2017). A Hierarchical Fused Fuzzy Deep Neural Network for data classification. *IEEE Transactions on Fuzzy Systems*, *25*(4), 1006–1012. doi:10.1109/TFUZZ.2016.2574915

Deutsch & He. (2016). *Using deep learning based approaches for bearing remaining useful life prediction*. Academic Press.

Developer guides. (2019, December 27). Retrieved from https://developer.android.com/guide

Devi, M. N. (2011). *Pattern Recognition: An Algorithmic Approach* (1st ed.). Springer-Verlag London.

Devi, S., & Neetha, T. (2017). Machine Learning based traffic congestion prediction in a IoT based Smart City. *Int. Res. J. Eng. Technol.*, *4*, 3442–3445.

Di Huang, C., Chien, Y. W., & Jia, C. W. (2016). Human action recognition system for elderly and children care using three stream ConvNet. *International Conference on Orange Technologies*, 5–9.

Dian, M. (2018). *DL with unsupervised data labelling for weeds detection on UAV images*. doi:10.20944/preprints201809.0088.v1

Ding, X., & He, Q. (2017). *Energy-fluctuated multiscale feature learning with deep convnet for intelligent spindle bearing fault diagnosis*. *IEEE Transactions on IM*. doi:10.1109/TIM.2017.2674738

Diro, A., & Chilamkurti, N. (2018). *Distributed attack detection scheme using deep learning approach for internet of things* (Vol. 82). Future Generation Computer Systems. Available https://www.sciencedirect.com/science/article/pii/S0167739X17308488

Donato, G. e. (1999). Classifying facial actions. *Pattern Analysis and Machine Intelligence*. *IEEE Transactions on*, *21*(10), 974–989.

Dörner, S., Cammerer, S., Hoydis, J., & ten Brink, S. (2018). Deep learning based communication over the air. *IEEE Journal of Selected Topics in Signal Processing*, *12*(1), 132–143. doi:10.1109/JSTSP.2017.2784180

Dos Santos, U. J. L., Pessin, G., da Costa, G. A., & da Rosa Righi, R. (2019). AgriPrediction: A proactive internet of things model to anticipate problems and improve production in agricultural crops. *Computers and Electronics in Agriculture*, *161*, 202–213. doi:10.1016/j.compag.2018.10.010

Douarre, C., Schielein, R., Frindel, C., Gerth, S., & Rousseau, D. (2016). Deep learning based root-soil segmentation from X-ray tomography. *bioRxiv*, 071662.

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis* (Vol. 3). New York: Wiley.

Edward, D. (2017). *Intro to TensorFlow and Machine Learning*. Retrieved may 31, 2019 from https://meetings.internet2.edu/media/medialibrary/2017/10/18/20171018-doan-intro-tensorflow-machine-learning.pdf

Ekman, W. a. (1977). *Facial action coding system*. Palo Alto: Consulting Psychologists Press, Stanford University.

ElHefnawi, M. (2012). Accurate prediction of response to Interferon-based therapy in Egyptian patients with Chronic Hepatitis C using machine-learning approaches. In *Proceedings of IEEE/ACM Int. Conf. Advances Social Netw. Anal. Mining* (pp. 771-778). 10.1109/ASONAM.2012.140

Elliss-Brookes, L. (2015, Nov. 10). Big data in action: the story behind Routes to Diagnosis. *Public health matters blog*.

Enache, A.-C., Sgarciu, V., & Petrescu-Nit¸aˇ, A. (2015). Intelligent feature selection method rooted in binary bat algorithm for intrusion detection. *2015 IEEE 10th jubilee international symposium on applied computational intelligence and informatics (SACI)*, 517–521. 10.1109/SACI.2015.7208259

Erhan, D., Bengio, Y., Courville, A., & Vincent, P. (2009). *Visualizing higher-layer features of a deep network*. Technical Report 1341, Universit´e de Montr´eal.

Erickson, B. J., Korfiatis, P., Akkus, Z., Kline, T., & Philbrick, K. (2017). Toolkits and libraries for deep learning. *Journal of Digital Imaging*, *30*(4), 400–405. doi:10.100710278-017-9965-6 PMID:28315069

Evans, D. (2011). *The internet of things: How the next evolution of the internet is changing everything.* CISCO White Paper.

Falcini, F., & Lami, G. (2017). Deep Learning in Automotive: Challenges and Opportunities. *International Conference on Software Process Improvement and Capability Determination*, 279-288. 10.1007/978-3-319-67383-7_21

Fang, S., Da Xu, L., Zhu, Y., Ahati, J., Pei, H., Yan, J., & Liu, Z. (2014). An integrated system for regional environmental monitoring and management based on internet of things. *IEEE Transactions on Industrial Informatics*, *10*, 159–1605.

Farabet, C., Couprie, C., Najman, L., & LeCun, Y. (2013). Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1915–1929. doi:10.1109/TPAMI.2012.231

Fei-Fei, L., Justin, J., & Serena, Y. (2017). *Deep learning software*. Retrieved may 31, 2019 from http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture8.pdf

Felbo, B., Sundsøy, P., Pentland, A., Lehmann, S., & de Montjoye, Y.-A. (2016). *Using deep learning to predict demographics from mobile phone metadata*. Available: https://arxiv.org/abs/1511.06660

Feng, Q., Dou, Z., Li, C., & Si, G. (2016). Anomaly detection of spectrum in wireless communication via deep autoencoder. *Proc. International Conference on Computer Science and its Applications*, 259–265.

Ferentinos, K. P. (2018). Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, *145*, 311–318. doi:10.1016/j.compag.2018.01.009

Ferrucci, D., & Lally, A. (2004). UIMA: An architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, *10*(3-4), 327–348. doi:10.1017/S1351324904003523

Feurer, Klein, Eggensperger, Springenberg, Blum, & Hutter. (2015a). Efficient and robust automated machine learning. NIPS, 2962–2970.

Feurer, M., Springenberg, J. T., & Hutter, F. (2015b, February). Initializing bayesian hyperparameter optimization via meta-learning. *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Fourati, M. A., Chebbi, W., & Kamoun, A. (2014). Development of a web-based weather station for irrigation scheduling. *Information Science and Technology (CIST)*, 37-42. 10.1109/CIST.2014.7016591

Freytag, A., Rodner, E., Bodesheim, P., & Denzler, J. (2013). Labeling Examples that Matter: Relevance-Based Active Learning with Gaussian Processes. In *35th German Conference on Pattern Recognition (GCPR).* Saarland University an Max-Planck-Institute for Informatics.

Friesen, P. E. (1969). Pan-Cultural Elements In Facial Display Of Emotion. *Science*, 86–88.

Fritscher, K., Raudaschl, P., Zaffino, P., Spadea, M. F., Sharp, G. C., & Schubert, R. (2016). Deep neural networks for fast segmentation of 3d medical images. *Proc. MICCAI*, 158–165. 10.1007/978-3-319-46723-8_19

Fuentes, A., Yoon, S., Kim, S. C., & Park, D. S. (2017). A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. *Sensors (Basel)*, *17*(9), 2022. doi:10.339017092022 PMID:28869539

Fujihashi, Koike-Akino, Watanabe, & Orlik. (2018). Nonlinear equalization with deep learning for multipurpose visual MIMO communications. *Proc. IEEE International Conference on Communications (ICC)*, 1–6.

Fu, T., Han, Y., Li, X., Liu, Y., & Wu, X. (2015). Integrating prosodic information into recurrent neual network language model for speech recognition. *Proceedings of Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, 1194–1197.

Futoma, J., Morris, J., & Lucas, J. (2015). A comparison of models for predicting early hospital readmissions. *Journal of Biomedical Informatics*, *56*, 229–238. doi:10.1016/j.jbi.2015.05.016

Galloway, Catterson, Fay, Robb, & Love. (2016). *Diagnosis of tidal turbine vibration data through deep neural networks*. Academic Press.

Gan, M., Wang, C., & Zhu, C. (2016). Construction of hierarchical diagnosis network based on deep learning and its application in the fault pattern recognition of rolling element bearings. *Mechanical Systems and Signal Processing*, *72*, 92–104. doi:10.1016/j.ymssp.2015.11.014

Gante, J., Falcão, G., & Sousa, L. (2018). *Beamformed Fingerprint Learning for Accurate Millimeter Wave Positioning*. arXiv preprint arXiv:1804.04112

Gao, N., Gao, L., Gao, Q., & Wang, H. (2014, November). An Intrusion Detection Model Based on Deep Belief Networks. *Second International Conference on Advanced Cloud and Big Data*, (pp. 247-252). Huangshan: Academic Press.

Garimella, V. R. K., Alfayad, A., & Weber, I. (2016). *Social media image analysis for public health. In Proc. CHIConf* (pp. 5543–5547). Human Factors Comput. Syst. Available http://doi.acm. org/10.1145/2858036.2858234

Gartner. (2015). *Big Data Management and Analytics*. Available: https://www.gartner.com/technology/topics/big-data.jsp

Geoffrey, E. (2006, July). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504–507. doi:10.1126cience.1127647

Gers, F., Schraudolph, N. N., & Schmidhuber, J. (2016). *Learning Precise Timing with LSTM Recurrent Networks*. ResearchGate.

GhaffarianHoseini, A. H., Dahlan, N. D., Berardi, U., GhaffarianHoseini, A., & Makaremi, N. (2013). The essence of future smart houses: From embedding ICT to adapting to sustainability principles. *Renewable & Sustainable Energy Reviews*, *24*, 593–607. doi:10.1016/j.rser.2013.02.032

Gheisari, M., Wang, G., & Md, Z. A. B. (2017). A survey on deep learning in big data. *Proc. IEEE International Conference on Computational Science and Engineering (CSE) and Embedded and Ubiquitous Computing (EUC), 2*, 173–180. 10.1109/CSE-EUC.2017.215

Ghumbre, S., Patil, C., & Ghatol, A. (2011, December). Heart disease diagnosis using support vector machine. International conference on computer science and information technology (ICCSIT') Pattaya.

Giannakos, M. N., Aalberg, T., Divitini, M., Jaccheri, L., Mikalef, P., Pappas, I. O., & Sindre, G. (2017, April). Identifying Dropout Factors in Information Technology Education: A Case Study. *IEEE Global Engineering Education Conference (EDUCON)*, (pp. 1187-1194). 10.1109/EDUCON.2017.7942999

Gill, K., Yang, S.-H., Yao, F., & Lu, X. (2009). A zigbee-based home automation system. *IEEE Transactions on Consumer Electronics*, *5*(2), 422–430. doi:10.1109/TCE.2009.5174403

Glorot, X., & Bengio, Y. (2010). *Understanding the difficulty of training deep feed-forward neural networks*. International Conference on Artificial Intelligence and Statistics, Chia Laguna Resort, Sardinia, Italy.

Gobert, J. D., Kim, Y. J., Pedro, M. A., Kennedy, M., & Betts, C. G. (2015). Using educational data mining to assess students' skills at designing and conducting experiments within a complex systems microworld. *Thinking Skills and Creativity*, *18*, 81–90. doi:10.1016/j.tsc.2015.04.008

Goh, G.B., Sakloth, K., Siegel, C., Vishnu, A., & Pfaendtner, J. (2018). *Multimodal Deep Neural Networks using Both Engineered and Learned Representations for Biodegradability Prediction*. ArXiv, abs/1808.04456

Gokhale, Bhat, & Bhat. (2007). Introduciton to IoT. *International Advanced Research Journal in Science, Engineering and Technology, 5*(1). 41 doi:10.17148/IARJSET.2018.517

Gomez, C., & Paradells, J. (2010). Wireless home automation networks: A survey of architectures and technologies. *IEEE Communications Magazine*, *48*(6), 92–101. doi:10.1109/MCOM.2010.5473869

Gonzalez, R., Garcia-Duran, A., Manco, F., Niepert, M., & Vallina, P. (2017). Network data monetization using Net2Vec. *Proc. ACM SIGCOMM Posters and Demos*, 37–39. 10.1145/3123878.3131978

Goodfello, I. (2016). *Deep learning*. MIT Press.

Goodfellow, I. J., Warde-Farley, D., Lamblin, P., Dumoulin, V., Mirza, M., Pascanu, R., & Bengio, Y. (2013). *Pylearn2: a machine learning research library*. arXiv preprint arXiv:1308.4214

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 2672–2680.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Gopalakrishnan, K. (2018). Deep Learning in Data-Driven Pavement Image Analysis and Automated Distress Detection: A Review. *Data*, *3*(3), 28. doi:10.3390/data3030028

Graves, A., & Schmidhuber, J. (2005). (Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *IJCNN*, *18*(5), 602–610.

Gravitz, L. (2011). A smouldering public-health crisis. *Nature*, *474*(7350), S2–S4. doi:10.1038/474S2a PMID:21666731

Greenspan, H., van Ginneken, B., & Summers, R. M. (2016, May). Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. *IEEE Transactions on Medical Imaging*, *35*(5), 1153–1159. doi:10.1109/TMI.2016.2553401

Griewank, A., & Walther, A. (2008). *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation* (2nd ed.). SIAM. doi:10.1137/1.9780898717761

Guan, W., Wu, Y., Xie, C., Chen, H., Cai, Y., & Chen, Y. (2017). High-precision approach to localization scheme of visible light communication based on artificial neural networks and modified genetic algorithms. *Optical Engineering (Redondo Beach, Calif.)*, *56*(10), 106103. doi:10.1117/1.OE.56.10.106103

Guarín, C., Guzmán, E., & González, F. (2015). A Model to Predict Low Academic Performance at a Specific Enrollment Using Data Mining. *IEEE Journal of Latin-American Learning Technologies, 10*(3).

Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet Of Things (Iot): A Vision, Architectural Elements, And Future Directions. *Future Generation Computer Systems*, *29*(7), 1645–1660. doi:10.1016/j.future.2013.01.010

Gulli, A., & Pal, S. (2017). *Deep Learning with Keras*. Packt Publishing Ltd.

Guo, B., Zhang, R., Guang, X., Shi, C., & Yang, L. (2015, July). Predicting Students performance in educational data mining. *International Symposium on Educational Technology (ISET)*, (pp. 125-128). 10.1109/ISET.2015.33

Guo, L., Gao, H., Huang, H., He, X., & Li, S. (2016). *Multifeatures fusion and nonlinear dimension reduction for intelligent bearing condition monitoring* (Vol. 2016). Shock and Vibration.

Guo, X., Chen, L., & Shen, C. (2016). Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis. *Measurement*, *93*, 490–502. doi:10.1016/j.measurement.2016.07.054

Gupta, B. (2017). Analysis of various decision tree algorithms for classification in data mining. *International Journal of Computers and Applications*, *163*(8), 15–19.

Ha, S., Yun, J. M., & Choi, S. (2015). Multi-modal convolutional neural networks for activity recognition. *Proc. Int. Conf. Syst., Man, Cybern*., 3017–3022. 10.1109/SMC.2015.525

Hamedani, K., Liu, L., Atat, R., Wu, J., & Yi, Y. (2018). Reservoir computing meets smart grids: Attack detection using delayed feedback networks. *IEEE Transactions on Industrial Informatics*, *14*(2), 734–743. doi:10.1109/TII.2017.2769106

Han, J., & Kamber, M. (2012). *Data Mining: Concepts and Techniques* (3rd ed.). Morgan Kaufmann.

Harvey, C. A., Rakotobe, Z. L., Rao, N. S., Dave, R., Razafimahatratra, H., Rabarijohn, R. H., ... MacKinnon, J. L. (2014). Extreme vulnerability of smallholder farmers to agricultural risks and climate change in madagascar. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, *369*(1639), 20130089. doi:10.1098/rstb.2013.0089 PMID:24535397

Hasan. (2018). Detection and analysis of wheat spikes using Convolutional NNs. *Plant Methods*, doi:10.118613007-018-0366-8

Hashem, S. (2014). A simple multi-linear regression model for predicting fibrosis scores in chronic Egyptian hepatitis C virus patients. *International Journal of Biological Technology*, *4*(3), 37–46.

Hashem, S., Esmat, G., Elakel, W., Habashy, S., Abdel Raouf, S., Darweesh, S., ... ElHefnawi, M. (2016). Accurate prediction of advanced liver fibrosis using the decision tree learning algorithm in chronic Hepatitis C Egyptian patients. *Gastroenterology Research and Practice*, *2016*, 2636390. doi:10.1155/2016/2636390 PMID:26880886

Hashim, N., Mazlan, S., Aziz, M. A., Salleh, A., Jaafar, A., & Mohamad, N. (2015). Agriculture monitoring system:a study. *J. Teknologi*, *77*, 53–59.

Havaei, M., Guizard, N., Larochelle, H., & Jodoin, P. (2016). *Deep learning trends for focal brain pathology segmentation in MRI*. Available: https://arxiv.org/abs/1607.05258

Hellerstein, J. M., Sreekanti, V., Gonzalez, J. E., Dalton, J., Dey, A., Nag, S., & Donsky, M. (2017, January). Ground: A Data Context Service. CIDR.

Helmstaedter, M., Briggman, K. L., Turaga, S. C., Jain, V., Seung, H. S., & Denk, W. (2013). Connectomic reconstruction of the inner plexiform layer in the mouse retina. *Nature*, *500*(7461), 168–174. doi:10.1038/nature12346

Hermanto, A., Adji, T. B., & Setiawan, N. A. (2015). Recurrent neural network language model for english-indonesian machine translation: Experimental study. *Proceedings of International Conference on Science in Information Technology*, 132–136. 10.1109/ICSITech.2015.7407791

He, Y., Liang, C., Richard Yu, F., Zhao, N., & Yin, H. (2017). Optimization of cache-enabled opportunistic interference alignment wireless networks: A big data deep reinforcement learning approach. *Proc.* 2017 *IEEE International Conference on Communications (ICC)*, 1–6. 10.1109/ICC.2017.7996332

He, Y., Richard Yu, F., Zhao, N., & Victor, C. M. (2017). Software-defined networks with mobile edge computing and caching for smart cities: A big data deep reinforcement learning approach. *IEEE Communications Magazine*, *55*(12), 31–37. doi:10.1109/MCOM.2017.1700246

Heydari & Minaei-Bidgoli. (2017). Reduce energy consumption and send secure data wireless multimedia sensor networks using a combination of techniques for multi-layer watermark and deep learning. *International Journal of Computer Science and Network Security*, *17*(2), 98–105.

Highfield, R. (2009). How your looks betray your personality. *New Scientist*.

Hijazi, S. T., & Naqvi, S. R. (2006). Factors Affecting Students' Performance. *Bangladesh e-Journal of Sociology*, 3.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, *29*(6), 82–97. doi:10.1109/MSP.2012.2205597

Hinton, G., & Salakhutdinov, R. (2011). Discovering binary codes for documents by learning deep generative models. *Topics in Cognitive Science*, *3*(1), 74–79. doi:10.1111/j.1756-8765.2010.01109.x

Hitaj, Gasi, Ateniese, & Perez- Cruz. (2017). *PassGAN: A deep learning approach for password guessing*. arXiv preprint arXiv:1709.00440

Hordri, N. F., Samar, A., Yuhaniz, S. S., & Shamsuddin, S. M. (2017). A systematic literature review on features of deep learning in big data analytics. *International Journal of Advances in Soft Computing & Its Applications*, *9*(1).

How secure is your mobile app? (n.d.). Retrieved from https://vulnerabilitytest.quixxi.com/#/

Huang, C. Y., Tsai, Y. T., & Hsu, C. H. (2013) Performance Evaluation on Permission-Based Detection for Android Malware. In Advances in Intelligent Systems and Applications - Volume 2. Smart Innovation, Systems and Technologies, vol 21. Springer. doi:10.1007/978-3-642-35473-1_12

Huang, C.-D., Wang, C.-Y., & Wang, J.-C. (2015). Human action recognition system for elderly and children care using three stream convnet. *Proc. Int. Conf. Orange Technol.*, 5–9. 10.1109/ICOT.2015.7498476

Huang, H., Deng, J., & Lan, Y. (2018). A fully convolutional network for weed mapping of unmanned aerial vehicle (UAV) imagery. *PLoS One*, *26*(April). doi:10.1371/journal.pone.0196302 PMID:29698500

Huang, H., Jie, Y., Song, Y., Hao, H., & Guan, G. (2018). Deep Learning for Super-Resolution Channel Estimation and DOA Estimation based Massive MIMO System. *IEEE Transactions on Vehicular Technology*, *67*(9), 8549–8560. doi:10.1109/TVT.2018.2851783

Huang, J., Zhou, W., Li, H., & Li, W. (2015). Sign language recognition using real-sense. *Proc. IEEE ChinaSIP*, 166–170.

Huang, S., & Lin, H. (2018). Fully optical spacecraft communications: Implementing an omnidirectional PV-cell receiver and 8 Mb/s LED visible light downlink with deep learning error correction. *IEEE Aerospace and Electronic Systems Magazine*, *33*(4), 16–22. doi:10.1109/MAES.2018.170179

Huang, W., Song, G., Hong, H., & Xie, K. (2014). Deep architecture for traffic flow prediction: Deep belief networks with multitask learning. *IEEE Transactions on Intelligent Transportation Systems*, *15*(5), 2191–2201. doi:10.1109/TITS.2014.2311123

Huijie, Z., Ting, R., Xinqing, W., You, Z., & Husheng, F. (2015). Fault diagnosis of hydraulic pump based on stacked autoencoders. *2015 12th IEEE International Conference on Electronic Measurement Instruments (ICEMI), 1*, 58–62. 10.1109/ICEMI.2015.7494195

Huttenlocher, P. F. (2005). Pictorial structures for object recognition. *International Journal of Computer Vision*, *61*(1), 55–79. doi:10.1023/B:VISI.0000042934.15159.49

Ibrahim, Yousri, Ismail, & El-Makky. (2014). Multi-level gene/mirna feature selection using deep belief nets and active learning. *Proc. Eng. Med. Biol. Soc.*, 3957–3960.

Ienco, D., Gaetano, R., Dupaquier, C., & Maurel, P. (2017). *Land Cover Classification via Multi-temporal Spatial Data by Recurrent Neural Networks*. arXiv preprint arXiv:1704.04055

Imtiaz, H., & Fattah, S. A. (2011). A wavelet-domain local dominant feature selection scheme for face recognition. *International Journal of Computing and Business Research, 3*(2).

Index of /iscxdownloads/CICAndMal2017/APKs. (2018, August 31). Retrieved from https://iscxdownloads.cs.unb.ca/iscxdownloads/CICAndMal2017/APKs

Innova.com.tr. (2015). *Innova deploys smart home and smart office services for TTNET | Innova*. Available at: http://www.innova.com.tr/en/newsdetail. asp?haber=777C888F-C922-8822-EEE2-BBB222FF333C

Ioannou, S. V., Raouzaiou, A. T., Tzouvaras, V. A., Mailis, T. P., Karpouzis, K. C., & Kollias, S. D. (2005). Emotion recognition through facial expression analysis based on a neurofuzzy network. *Neural Networks*, *18*(4), 423–435. doi:10.1016/j.neunet.2005.03.004 PMID:15963691

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *Proceedings of the 32nd International Conference on Machine Learning, in PMLR, 37*, 448-456.

Isidoro, S. S. (1998). Active blobs. In *Computer Vision, 1998. Sixth International Conference*, (pp. 1146–1153). Academic Press.

Jagannath, K. (2018). Automatic semantic segmentation and classification of remote sensing data for agriculture. *Mathematical Models in Engineering*, *4*(2), 112–137. doi:10.21595/mme.2018.19840

Jagtap, S., Hambarde, S. (2014). Agricultural Plant Leaf Disease Detection and Diagnosis Using Image Processing Based on Morphological Feature Extraction, *IOSR Journal of VLSI and Signal Processing*, *4*, 24–30.

James, W. (1884). What is Emotion. *Mind*, *9*(34), 188–205. doi:10.1093/mind/os-IX.34.188

Janssens, O., Slavkovikj, V., Vervisch, B., Stockman, K., Loccufier, M., Verstockt, S., ... Van Hoecke, S. (2016). Convolutional neural network based fault detection for rotating machinery. *Journal of Sound and Vibration*, *377*, 331–345. doi:10.1016/j.jsv.2016.05.027

Jean, S., Cho, K., Memisevic, R., & Bengio, Y. (2015). On using very large target vocabular for neural machine translation. *Proc. ACL-IJCNLP*. https://arxiv.org/abs/1412.2007

Jeon, H. Y., Tian, L. F., & Zhu, H. (2011). Robust crop and weed segmentation under uncontrolled outdoor illumination. *Sensors (Basel)*, *11*(6), 6270–6283. doi:10.3390110606270 PMID:22163954

Jia, X., Li, K., Li, X., & Zhang, A. (2014). A novel semi-supervised deep learning framework for affective state recognition on eeg signals. *Proc.Int. Conf. Bioinformat. Bioeng*., 30–37. 10.1109/BIBE.2014.26

Jia, F., Lei, Y., Lin, J., Zhou, X., & Lu, N. (2016J). Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data. *Mechanical Systems and Signal Processing*, *72*, 303–315. doi:10.1016/j.ymssp.2015.10.025

Jiang, Wang, Luo, Zhang, Xiong, & Pang. (2015). Convolutional neural networks in automatic recognition of trans-differentiated neural progenitor cells under bright-field microscopy. *Proc. Instrum. Meas., Comput., Commun. Control*, 122–126.

Jin, Y., & Raychowdhury, A. (2016). Internet of Things realize its potential. *Computer*, *49*(8), 112–116. doi:10.1109/MC.2016.245

Ji, S., Zhang, C., Xu, A., Shi, Y., & Duan, Y. (2018). 3D Convolutional Neural Networks for Crop Classification with Multi-Temporal Remote Sensing Images. *Remote Sensing*, *10*(2), 75. doi:10.3390/rs10010075

Johannes, A., Picon, A., Alvarez-Gila, A., Echazarra, J., Vaamonde, S. R., Navajas, A. D., & Ortiz-Barredo, A. (2017). Automatic plant disease diagnosis using mobile capture devices, applied on a wheat use case. *Computers and Electronics in Agriculture*, *138*, 200–209. doi:10.1016/j.compag.2017.04.013

Joshi, W. T. (2005). Learning controls for blend shape based realistic facial animation. ACM SIGGRAPH 2005 Courses, 8.

Julien, D. (2017). *Getting Started with Caffe*. Retrieved may 31, 2019 from https://indico.cern.ch/event/395374/contributions/939872/attachments/1186808/1721069/Getting_started_with_caffe_v2.pdf

Junbo, T., Weining, L., Juneng, A., & Xueqian, W. (2015). Fault diagnosis method study in roller bearing based on wavelet transform and stacked auto-encodeqr. In *The 27th Chinese Control and Decision Conference (2015 CCDC).* IEEE.

Jurman, G., Riccadonna, S., & Furlanello, C. (2012). A comparison of MCC and CEN error measures in multi-class prediction. *PLoS One*, *7*(8), e41882. doi:10.1371/journal.pone.0041882 PMID:22905111

Kaed, C., Leida, B., & Gray, T. (2016). Building management insights driven by a multi-system semantic representation approach. *2016 IEEE 3rd world forum on internet of things (WF-IoT)*, 520–525. 10.1109/WF-IoT.2016.7845433

Kalaivani, S., Chatterjee, P., Juyal, S., & Gupta, R. (2017). *Lung cancer detection using digital image processing and artificial neural networks*. International conference of Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India. 10.1109/ICECA.2017.8212773

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018, April). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, *147*, 70–90. doi:10.1016/j.compag.2018.02.016

Kanade, L. G. (2008). A generative shape regularization model for robust face alignment. In *Computer Vision–ECCV.* Springer Berlin Heidelberg.

Kanade, T. J. (2000). Comprehensive database for facial expression analysis. In *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition* (Cat. No. PR00580). IEEE. 10.1109/AFGR.2000.840611

Kang, J., Park, Y.-J., Lee, J., Wang, S.-H., & Eom, D.-S. (2018). Novel Leakage Detection by Ensemble CNN-SVM and Graph-based Localization in Water Distribution Systems. *IEEE Transactions on Industrial Electronics*, *65*(5), 4279–4289. doi:10.1109/TIE.2017.2764861

Karpouzis, S. K. (2005). Multimodal Emotion Recognition And Expressivity Analysis. *IEEE International Conference on Mulitimedia and Expo*, 779-783.

Kasfi, K. T., Hellicar, A., & Rahman, A. (2016). Convolutional neural network for time series cattle behaviour classification. In *Proceedings of the Workshop on Time Series Analytics and Applications*. ACM. 10.1145/3014340.3014342

Kass, M., Witkin, A., & Terzopoulos, D. (1988). Snakes: Active contour models. *International Journal of Computer Vision*, *1*(4), 321–331. doi:10.1007/BF00133570

Kendra, R. L., Karki, S., Eickholt, J. L., & Gandy, L. (2015). Characterizing the discussion of antibiotics in the Twittersphere: What is the bigger picture? *J. Med. Internet Res., 17*(6).

Keresztes, B., Abdelghafour, F., Randriamanga, D., Da Costa, J.-P., & Germain, C. (2018). Real-time Fruit Detection Using Deep NN. *14th International Conference on Precision Agriculture*, Montreal, Canada.

Ketkar, N. (2017). Introduction to pytorch. In *Deep learning with python* (pp. 195–208). Berkeley, CA: Apress. doi:10.1007/978-1-4842-2766-4_12

Khademi, M., & Nedialkov, N. S. (2015). Probabilistic graphical models and deep belief networks for prognosis of breast cancer. *Proc. IEEE 14th Int. Conf. Mach. Learn. Appl*., 727–732. 10.1109/ICMLA.2015.196

Khan, Islam, & Hafeez. (2011). Irrigation Water Requirement Prediction through Various Data Mining Techniques Applied on a Carefully Pre-processed Dataset. *CRPIT*, 121.

Khan, M. A., Khan, S., Shams, B., & Lloret, J. (2016). Distributed flood attack detection mechanism using artificial neural network in wireless mesh networks. *Security and Communication Networks*, *9*(15), 2715–2729. doi:10.1002ec.1204

Khorasani, F., & Naji, H. R. (2017). Energy efficient data aggregation in wireless sensor networks using neural networks. *International Journal of Sensor Networks*, *24*(1), 26–42. doi:10.1504/IJSNET.2017.084207

Kim, T., Kang, B., Rho, M., Sezer, S., & Im, E. G. (2019). A Multimodal Deep Learning Method for Android Malware Detection Using Various Features. IEEE Trans. on Info Forensics and Sec., 14(3). doi:10.1109/TIFS.2018.2866319

Kim, D. Y., & Park, J. W. (2004). Computer-aided detection of kidney tumor on abdominal computed tomography scans. *Acta Radiologica*, *45*(7), 791–795. doi:10.1080/02841850410001312 PMID:15624525

Kim, H.-Y., & Kim, J.-M. (2017). A load balancing scheme based on deep-learning in IoT. *Cluster Computing*, *20*(1), 873–878. doi:10.100710586-016-0667-5

Kim, M. Y., Jeong, W. K. Y., & Baik, S. K. (2014). Invasive and noninvasive diagnosis of cirrhosis and portal hypertension. *World Journal of Gastroenterology*, *20*(15), 4300–4315. doi:10.3748/wjg.v20.i15.4300 PMID:24764667

Kim, P. (2017). Matlab deep learning. In *With Machine Learning*, *Neural Networks and Artificial Intelligence.* Apress. doi:10.1007/978-1-4842-2845-6

Kim, S. M. L., & Shen, J. (2015, July). A novel deep learning by combining discriminative model with generative model. *International Joint Conference on Neural Networks (IJCNN)*, (pp. 1-6). 10.1109/IJCNN.2015.7280589

Kingma, D. P., & Ba, J. (2015).*Adam: A method for stochastic optimization*. Paper presented at ICLR, San Diego, CA.

Kingma, D. P., & Welling, M. (2014). Stochastic gradient VB and the variational auto-encoder. *Proceedings of the Second International Conference on Learning Representations (ICLR)*.

Kipf & Welling. (2107). Semi-supervised classification with graph convolutional networks. *Proc. International Conference on Learning Representations (ICLR)*.

Kirkby, R. (2002). *Weka explorer user guide for version 3-3-4*. University of Weikato.

Kleesiek, J., Urban, G., Hubert, A., Schwarz, D., Maier-Hein, K., Bendszus, M., & Biller, A. (2016). Deep MRI brain extraction: A 3D Convolutional neural network for skull stripping. *NeuroImage*, *129*, 460–469. doi:10.1016/j.neuroimage.2016.01.024

Koch, F., Assunção, M. D., Cardonha, C., & Netto, M. A. (2016). Optimising resource costs of cloud computing for education. *Future Generation Computer Systems*, *55*, 473–479. doi:10.1016/j.future.2015.03.013

Kodali, R. K., Rawat, N., & Boppana, L. (2014). WSN sensors for precision agriculture. In *Region 10 Symposium*, (pp. 651-656). IEEE. 10.1109/TENCONSpring.2014.6863114

Kodati, S., & Vivekanandam, R, R. (2018). Comparative analysis of classification techniques for heart disease with data mining Weka tool. *Journal of Adv Research in Dynamical & Control Systems*, *10*(9), 2224–2227.

Kokare, M., Chatterji, B. N., & Biswas, P. K. (2003). Comparison of similarity metrics for texture image retrieval. *Proc. IEEE Transactions on TENCON 2003, Conference on Convergent Technologies for Asia-Pacific Region*, 2, 571-575. 10.1109/TENCON.2003.1273228

Kondo, T., Ueno, J., & Takao, S. (2014). Medical image recognition of abdominal multi-organs by hybrid multi-layered GMDH-type neural network using principal component-regression analysis. *Proc. 2nd Int. Symp. Comput. Netw.*, 157–163. 10.1109/CANDAR.2014.62

Kondo, T., Ueno, J., & Takao, S. (2016). Medical image diagnosis of lung cancer by deep feedback GMDH-type neural network. *Robot. Netw. Artif. Life*, *2*(4), 252–257. doi:10.2991/jrnal.2016.2.4.11

Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems. Curran Associates, Inc.

Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). ImageNet classification with deep convolutional neural networks. Proc. Advances in Neural Information Processing Systems, 25, 1090–1098.

Krizhevsky, Sutskever, & Hinton. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 1097–1105.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Proceedings of the Advances in Neural Information Processing Systems (NIPS 2012)*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Proceedings of the conference Neural Information Processing Systems*.

Kuang & He. (2014). Classification on ADHD with deep learning. *Proc. Cloud Comput. Big Data*, 27–32.

Kuhad, P., Yassine, A., & Shimohammadi, S. (2015). Using distance estimation and deep learning to simplify calibration in food calorie measurement. *Proc. IEEE Int. Conf. Comput. Intell. Virtual Environ. Meas. Syst. Appl.*, 1–6. 10.1109/CIVEMSA.2015.7158594

Kumar, M. P., & Rajagopal, M. K. (2019). *Detecting facial emotions using normalized minimal feature vectors and semi-supervised twin support vector machines classifier*. Academic Press.

Kumar, A., McCann, R., Naughton, J., & Patel, J. M. (2016). Model selection management systems: The next frontier of advanced analytics. *SIGMOD Record*, *44*(4), 17–22. doi:10.1145/2935694.2935698

Kumar, R., Talton, J. O., Ahmad, S., & Klemmer, S. R. (2012). Data-driven web design. *Proceedings of the 29th International.*

Kurian, C., & Balakriahnan, K. (2012). Cont inuous speech recognition system for Malayalam language using PLP cepstral coefficient. *Journal of Computing and Business Research, 3*(1).

Kussul, N., Lavreniuk, M., Skakun, S., & Shelestov, A. (2017, May). DL Classification of Land Cover and Crop Types Using Remote Sensing Data. *IEEE Geoscience and Remote Sensing Letters*, *14*(5), 778–782. doi:10.1109/LGRS.2017.2681128

Kuwata, K., & Shibasaki, R. (2015). Estimating crop yields with deep learning and remotely sensed data. *Geoscience and Remote Sensing Symposium*, 858-861. 10.1109/IGARSS.2015.7325900

Ladhak, Nazi, & Huber. (2013). Using deep learning to enhance cancer diagnosis and classification. *Proc. Int. Conf. Mach. Learn.*, 1–7.

Lakshmi, K., Krishna, M. V., & Kumar, S. P. (2013). Performance comparison of data mining techniques for predicting of heart disease survivability. *International Journal of Scientific and Research Publications*, *3*(6), 1–10. https://archive.ics.uci.edu/ml/datasets/Heart+Disease

Laurenz Wiskott, J.-M. F. (1997). Face recognition by elastic bunch graph matching. *Pattern Analysis and Machine Intelligence*. *IEEE Transactions*, *19*(7), 775–779.

Lauzon, F. Q. (2012). An introduction to deep learning. *International Conference on Information Science, Signal Processing and their Applications (ISSPA)*, (pp. 1438-1439). Montreal: Academic Press.

LeCun, Y., & Bengio, Y. (1995). Convolutional networks for images, speech, and time series. In M. A. Arbib (Ed.), *The Handbook of Brain Theory and NNs* (pp. 255–258). Cambridge, MA: MIT Press.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. doi:10.1038/nature14539

Lee. (2017). Deep learning–based real-time query processing for wireless sensor network. *International Journal of Distributed Sensor Networks*, *13*(5).

Lee, H. S., Hong, H., & Kim, J. (2017). Detection and segmentation of small renal masses in contrast-enhanced CT images using texture and context feature classification. *Proceedings of IEEE International Symposium on Biomedical Imaging*. 10.1109/ISBI.2017.7950588

Le, Q. V., Zou, W. Y., Yeung, S. Y., & Ng, A. Y. (2011) Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. *Computer Vision and Pattern Recognition (CVPR) 2011 IEEE Conference On*, 3361–3368. 10.1109/CVPR.2011.5995496

Leung, M. K., Xiong, H. Y., Lee, L. J., & Frey, B. J. (2014). Deep learning of the tissue regulated splicing code. *Bioinformatics (Oxford, England)*, *30*(12), 121–129. doi:10.1093/bioinformatics/btu277

Levy, A. (2016). *Cisco Wagers $1.4 billion on Internet of Things, CNBC*. Retrieved from: https://www.cnbc.com/2016/02/03/cisco- wagers- 14- billion- on- internet- of- things.html

Li, L., Wu, Y., & Ye, M. (2015). Experimental comparisons of multi-class classifiers. Informatica, 39(1).

Li, Y. A. (2005). Shape parameter optimization for adaboosted active shape model. *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, 1*.

Liakos, K., Moustakidis, S., Tsiotra, G., Bartzanas, T., Bochtis, D., & Parisses, C. (2017). Machine Learning Based Computational Analysis Method for Cattle Lameness Prediction. *8th International Conference on Information and Communication Technology in Agriculture, Food and Environment*.

Liang, Shen, & Wu. (n.d.). *Exploiting noise correlation for channel decoding with convolutional neural networks*. Academic Press.

Liang, Z., Zhang, G., Huang, J. X., & Hu, Q. V. (2014). Deep learning for healthcare decision making with emrs. *Proc. Int. Conf. Bioinformat. Biomed.*, 556–559. 10.1109/BIBM.2014.6999219

Liao, L., Jin, W., & Pavel, R. (2016). Enhanced restricted boltzmann machine with prognosability regularization for prognostics and health assessment. *IEEE Transactions on Industrial Electronics*, *63*(11), 7076–7083. doi:10.1109/TIE.2016.2586442

Liao, R.-F., Wen, H., Wu, J., Song, H., Pan, F., & Dong, L. (2018). The Rayleigh Fading Channel Prediction via Deep Learning. *Wireless Communications and Mobile Computing*, *2018*, 1–11. doi:10.1155/2018/6497340

Li, F., Tran, L., Thung, K. H., Ji, S., Shen, D., & Li, J. (2015, September). A robust deep model for improved classification of ad/mci patients. *IEEE Journal of Biomedical and Health Informatics*, *19*(5), 1610–1616. doi:10.1109/JBHI.2015.2429556

Li, G., Zhu, H., Cheng, G., Thambiratnam, K., Chitsaz, B., Yu, D., & Seide, F. (2012). Context-dependent deep neural networks for audio indexing of real-life data. In *Spoken Language Technology Workshop (SLT), 2012 IEEE* (pp. 143–148). IEEE. doi:10.1109/SLT.2012.6424212

Li, K., & Wang, Q. (2015). Study on signal recognition and diagnosis for spacecraft based on deep learning method. *Prognostics and System Health Management Conference (PHM)*, 1–5.

Li, M., Chen, G., & Zhu, Z. (2014). Information service system of agriculture IoT. *Automatika J. Control, Meas. Electron. Comput. Commun*, *54*, 41–426.

Lin, P., Abney, K., & Jenkins, R. (Eds.), *Robot Ethics 2.0*. New York: Oxford University Press.

Lin, Z., & Yin, M. (2017). Deep generative models of urban mobility. *IEEE Transactions on Intelligent Transportation Systems*.

Lipton, Kale, Elkan, & Wetzell. (2015). Learning to diagnose with LSTM recurrent neural networks. *International Conference on Learning Representations (ICLR 2016)*.

Lipton, Z. C., Kale, D. C., Elkan, C., & Wetzel, R. C. (2015). *Learning to diagnosewithLSTMrecurrent neural networks*. vailable: https://arxiv.org/abs/1511.03677

Li, Q., Cai, W., Wang, X., Zhou, Y., Feng, D. D., & Chen, M. (2014). Medical image classification with convolutional neural network. *IEEE International Conference on Control Automation Robotics & Vision (ICARCV)*. 10.1109/ICARCV.2014.7064414

Li, S., Xu, L. D., & Zhao, S. (2015). The Internet of Things: A survey. *Information Systems Frontiers*, *17*(2), 243–259. doi:10.100710796-014-9492-7

Litjens, G., Kooi, T., Bejnordi, B. E., Ciompi, F., Ghafoorian, M., Jeroen, A. W. M., ... Sanchez, C. I. (2017). A survey on deep learning in medical image analysis. *Medical Image Analysis*, *42*, 60–88. doi:10.1016/j.media.2017.07.005 PMID:28778026

Littlewort, G., Bartlett, M. S., Fasel, I., Susskind, J., & Movellan, J. (2006). Dynamics of facial expression extracted automatically from video. *Image and Vision Computing*, *24*(6), 615–625. doi:10.1016/j.imavis.2005.09.011

Liu, Chen, Aditya, Sivakumar, Dcunha, Qu, … Kumar. (2018). *Robust Fruit Counting: Combining Deep Learning, Tracking, and Structure from Motion*. arXiv:1804.00307v2 [cs.CV]

Liu, B., Zhang, Y., & He, D. J. (2018). Yuxiang, Identification of Apple Leaf Diseases Based on Deep Convolutional NNs. *Symmetry*, *10*, 11. doi:10.3390ym10010011

Liu, C. H., Chen, Z., Tang, J., Xu, J., & Piao, C. (2018). Energy-efficient UAV control for effective and fair communication coverage: A deep reinforcement learning approach. *IEEE Journal on Selected Areas in Communications*, *36*(9), 2059–2070. doi:10.1109/JSAC.2018.2864373

Liu, H., Li, L., & Ma, J. (2016). *Rolling bearing fault diagnosis based on stft-deep learning and sound signals* (Vol. 2016). Shock and Vibration.

Liu, J., & Yu, L. (2011). Research on Development of Android Applications. *4th International Conference on Intelligent Networks and Intelligent Systems*, 69-72. doi: 10.1109/ICINIS.2011.40

Liu, S., & Du, J. (2016). Poster: Mobiear-building an environment independent acoustic sensing platform for the deaf using deep learning. *Proc. 14th ACM Annual International Conference on Mobile Systems, Applications, and Services Companion*, 50–50. 10.1145/2938559.2948831

Liu, X., Xu, Y., Jia, L., Wu, Q., & Anpalagan, A. (2018). Anti-jamming Communications Using Spectrum Waterfall: A Deep Reinforcement Learning Approach. *IEEE Communications Letters*, *22*(5), 998–1001. doi:10.1109/LCOMM.2018.2815018

Liu, Y., & Wu, L. (2016). Geological disaster recognition on optical remote sensing images using deep learning. *Procedia Computer Science*, *91*, 566–575. doi:10.1016/j.procs.2016.07.144

Li, W., Fu, H., Yu, L., & Cracknell, A. (2017). Deep learning Based Oil Palm Tree Detection and Counting for High Resolution Remote Sensing Images. *Remote Sensing*, *9*(1), 22. doi:10.3390/rs9010022

Lottes, P., Khanna, R., Pfeifer, J., Siegwart, R., & Stachniss, C. (2017). *UAV-Based Crop and Weed Classification for Smart Farming*. IEEE International Conference on Robotics and Automation (ICRA), Singapore.

Lu, W., Wang, X., Yang, C., & Zhang, T. (2015). A novel feature extraction method using deep neural network for rolling bearing fault diagnosis. *The 27th Chinese Control and Decision Conference (2015 CCDC)*, 2427–2431. 10.1109/CCDC.2015.7162328

Lu, C., Wang, Z.-Y., Qin, W.-L., & Ma, J. (2017). Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification. *Signal Processing*, *130*, 377–388. doi:10.1016/j.sigpro.2016.07.028

Lucey, P. J. (2010). The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*, (pp. 94-101). IEEE. 10.1109/CVPRW.2010.5543262

Ludo Maat, M. P. (n.d.). Retrieved from MMI Facial Expression Database: https://mmifacedb.eu/

Luo, C., Ji, J., Wang, Q., Yu, L., & Li, P. (2018). Online power control for 5G wireless communications: A deep Q-network approach. *Proc. IEEE International Conference on Communications (ICC)*, 1–6. 10.1109/ICC.2018.8422442

Luo, T., & Nagarajany. (2018). Distributed anomaly detection using autoencoder neural networks in WSN for IoT. *Proc. IEEE International Conference on Communications (ICC)*, 1–6.

Lu, Y., Yi, S., Zeng, N., Liu, Y., & Zhang, Y. (2017). Identification of rice diseases using deep convolutional neural networks. *Neurocomputing*, *267*, 378–384. doi:10.1016/j.neucom.2017.06.023

Lyu, W., Zhang, Z., Jiao, C., Qin, K., & Zhang, H. (2018). Performance evaluation of channel decoding with deep neural networks. *Proc. IEEE International Conference on Communications (ICC)*, 1–6. 10.1109/ICC.2018.8422289

Maceina, T. J., & Manduchi, G. (2017). Assessment of general purpose GPU systems in real-time control. *IEEE Transactions on Nuclear Science*, *64*(6), 1455–1460. doi:10.1109/TNS.2017.2691061

Maćkiewicz, A., & Ratajczak, W. (1993). Principal Components Analysis (PCA). *Computers & Geosciences*, *19*(3), 303–342. doi:10.1016/0098-3004(93)90090-R

Maitrey, S., & Jha, C. K. (2015). MapReduce: Simplified Data Analysis of Big Data. *Procedia Computer Science*, *57*, 563–571. doi:10.1016/j.procs.2015.07.392

Ma, J., Dua, K., Zhenga, F., Zhangb, L., Gongc, Z., & Suna, Z. (2018). A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network. *Computers and Electronics in Agriculture*, *154*, 18–24. doi:10.1016/j.compag.2018.08.048

Ma, J., Sheridan, R. P., Liaw, A., Dahl, G. E., & Svetnik, V. (2015). Deep neural nets as a method for quantitative structure-activity relationships. *Journal of Chemical Information and Modeling*, *55*(2), 263–274. doi:10.1021/ci500747n

Mansoor, A., Cerrolaza, J. J., Idrees, R., Biggs, E., Alsharid, M. A., Avery, R. A., & Linguraru, M. G. (2016, August). Deep learning guided partitioned shape model for anterior visual pathway segmentation. *IEEE Transactions on Medical Imaging*, *35*(8), 1856–1865. doi:10.1109/TMI.2016.2535222

Mao, W., He, J., Li, Y., & Yan, Y. (2016). Bearing fault diagnosis with autoencoder extreme learning machine: A comparative study. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 1–19.

Mao, H., Netravali, R., & Alizadeh, M. (2017). Neural adaptive video streaming with pensieve. In *Proc. Conference of the ACM Special Interest Group on Data Communication (SIGCOMM)*, (pp. 197–210). ACM. 10.1145/3098822.3098843

Mariolis, I., Peleka, G., & Kargakos, A. (2015). Pose and category recognition of highly deformable objects using deep learning. *Advanced Robotics (ICAR), 2015 International Conference on*, 655-662. 10.1109/ICAR.2015.7251526

Martin, H. (2018). *What is Keras? The deep neural network API explained*. Retrieved may 31, 2019 from https://www.infoworld.com/article/3336192/what-is-keras-the-deep-neural-network-api-explained.html

Maryam, M. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*, *2*(1), 1.

Mattihalli, C., Gedefaye, E., Endalamaw, F., & Necho, A. (2018). Plant leaf diseases detection and auto-medicine. *Internet of Things*, *2*, 67–73. doi:10.1016/j.iot.2018.08.007

Ma, W., Qiu, Z., Song, J., Cheng, Q., Ma, C., & Deep, G. S. (2018). Predicting phenotypes from genotypes using Deep Learning. *Planta*. doi:10.100700425-018-2976-9

McLaughlin, N., Martinez del Rincon, J., Kang, B., Yerima, S., Miller, P., Sezer, S., ... Joon Ahn, G. (2017). Deep Android Malware Detection. *Proceedings of the ACM Conference on Data and Applications Security and Privacy (CODASPY) 2017 Association for Computing Machinery (ACM)*. 10.1145/3029806.3029823

Mehrabi, S. (2015). Temporal pattern and association discovery of diagnosis codes using deep learning. *Proc. Int. Conf. Healthcare Informat.*, 408–416. 10.1109/ICHI.2015.58

Miao, H., Li, A., Davis, L. S., & Deshpande, A. (2017a April). Towards unified data and lifecycle management for deep learning. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE* (pp. 571-582). IEEE. 10.1109/ICDE.2017.112

Miao, H., Li, A., Davis, L. S., & Deshpande, A. (2017b, May). On model discovery for hosted data science projects. In *Proceedings of the 1st Workshop on Data Management for End-to-End Machine Learning* (pp. 1-4). 10.1145/3076246.3076252

Mikolov, T., Le, Q.V., & Sutskever, I. (2013). *Exploiting similarities among languages for machine translation*. CoRR.

Mikolov, T., Chen, K., & Dean, J. (2013). Efficient estimation of word representations in vector space. CoRR. *Computing*.

Mikolov, T., Deoras, A., Povey, D., Burget, L., & Černocký, J. (2011, December). Strategies for training large scale neural network language models. In *2011 IEEE Workshop on Automatic Speech Recognition & Understanding* (pp. 196-201). IEEE. 10.1109/ASRU.2011.6163930

Milioto, A., Lottes, P., & Stachniss, C. (2017). Real-time blob-wise sugar beets vs weeds classification for monitoring fields using convolutional neural networks. *Proceedings of the International Conference on Unmanned Aerial Vehicles in Geomatics*. 10.5194/isprs-annals-IV-2-W3-41-2017

Miotto, R., Li, L., & Brian, A. (2016). Deep patient: An unsupervised representation to predict the future of patients from the electronic health records. *Scientific Reports*, *6*(1), 26094. doi:10.1038rep26094

Mittal, Yagnik, Garg, & Krishnan. (2016). Spotgarbage: smartphone app to detect garbage using deep learning. *Proc. ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 940–945.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533. doi:10.1038/nature14236

Mo, Y. J., Kim, J., Kim, J. K., Mohaisen, A., & Lee, W. (2017). *Performance of deep learning computation with TensorFlowsoftware library in GPU-capable multi-core computing platforms*. International Conference on Ubiquitous and Future Networks (ICUFN), Milan, Italy.

Mohammed, J. (2016). *Predictions for the Internet of Things in 2016. World Economic Forum (2015)*. Retrieved from https://www.weforum.org/agenda/2015/12/5-predictions-for-the-internet-of-things-in-2016

Mohsen, H., El-Dahshan, E. S. A., El-Horbaty, E. S. M., & Salem, A. B. M. (2017). Classification using deep learning neural networks for brain tumors. *Future Computing and Informatics Journal*, *3*(1), 68–71. doi:10.1016/j.fcij.2017.12.001

Molchanov, P., Yang, X., Gupta, S., Kim, K., Tyree, S., & Kautz, J. (2016). Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network. *Proceedings of the IEEEConference on Computer Vision and Pattern Recognition*, 4207–4215. 10.1109/CVPR.2016.456

Mondal, P. (n.d.). *7 Important Factors that May Affect the Learning Process*. Retrieved from http://www.yourarticlelibrary.com/learning/7-important-factors-that-may-affect-the-learning-process/6064/

Montagu, J. (1994). *The Expression of the Passions: The Origin and Influence of Charles Le Brun's 'Conférence sur l'expression générale et particulière*. Yale University press. doi:10.1111/j.1467-8357.1994.tb00386.x

Mori, S., Suen, C. Y., & Kamamoto, K. (1992, July). Historical review of OCR research and development. *Proceedings of the IEEE*, *80*(7), 1029–1058. doi:10.1109/5.156468

Mredhula, L., & Dorairangaswamy, M. A. (2015). Detection and classification of tumours in CT images. *Indian Journal of Computer Science and Engineering*, *6*(2), 52–59.

Muangprathuba, J., Boonnama, N., Kajornkasirata, S., Lekbangponga, L., Wanichsombata, A., & Nillaorb, P. (2019). IoT and agriculture data analysis for smart farm. *Computers and Electronics in Agriculture*, *156*, 467–474. doi:10.1016/j.compag.2018.12.011

Muhammad, K., Ahmad, J., & Baik, S. W. (2018). Early fire detection using convolutional neural networks during surveillance for effective disaster management. *Neurocomputing*, *288*, 30–42. doi:10.1016/j.neucom.2017.04.083

Munoz-Organero, M., Ruiz-Blaquez, R., & Sánchez-Fernández, L. (2018). Automatic detection of traffic lights, street crossings and urban roundabouts combining outlier detection and deep learning classification techniques based on GPS traces while driving. *Computers, Environment and Urban Systems*, *68*, 1–8. doi:10.1016/j.compenvurbsys.2017.09.005

Mushtaq, I., & Khan, S. N. (2012). Factors Affecting Students' Academic Performance. *Global Journal of Management and Business Research*, *12*(9).

Naik, N. (2017). Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. *Proceedings of the 2017 IEEE international systems engineering symposium (ISSE)*, 1–7. 10.1109/SysEng.2017.8088251

Najafabadi, M. M., Villansutre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). *Deep learning applications and challenges in big data analytics*. Springer Open Journal of Big Data. doi:10.118640537-014-0007-7

Namin, S. T., Esmaeilzadeh, M., Najafi, M., Brown, T. B., & Borevitz, J. O. (2018). DL for temporal phenotype/genotype classification. *Plant Methods*, *14*, 66. doi:10.118613007-018-0333-4

Nashaat, E. (2010). Lipid profile among chronic hepatitis C Egyptian patients and its levels pre and post treatment. *Nature and Science*, *8*(7), 83–89.

Naumovich, V. V. (1998). *Statistical learning theory*. New York: Wiley.

Nazri, A., Mazlan, N., & Muharam, F. (2018). PENYEK: Automated brown planthopper detection from imperfect sticky pad images using deep convolutional NN. *PLoS One*, *13*(12), e0208501. doi:10.1371/journal.pone.0208501 PMID:30571683

Ndikumana, E., Dinh, H. T. M., Baghdadi, N., Courault, D., & Hossard, L. (2018). Deep Recurrent NN for Agricultural Classification using multitemporal SAR Sentinel-1 for Camargue, France. *Remote Sensing*, *10*(8), 1217. doi:10.3390/rs10081217

Ndzi, D. L., Harun, A., Ramli, F. M., Kamarudin, M. L., Zakaria, A., Shakaff, A. Y. M., & Farook, R. S. (2014). Wireless sensor network coverage measurement and planning in mixed crop farming. *Computers and Electronics in Agriculture*, *105*, 8–94. doi:10.1016/j.compag.2014.04.012

Neverova, N., Wolf, C., & Taylor, G. W. (2014). Multi-scale deep learning for gesture detection and localization. Computer Vision-ECCV 2014 Workshops, 474-490.

Ng, J. R., Wong, J. S., Goh, V. T., Yap, W. J., Yap, T. T. V., & Ng, H. (2019). Identification of Road Surface Conditions using IoT Sensors and Machine Learning. In *Computational Science and Technology* (pp. 259–268). Singapore: Springer. doi:10.1007/978-981-13-2622-6_26

Nguyen, Al-Mannai, Joty, Sajjad, Imran, & Mitra. (2017). Robust classification of crisis-related data on social networks using convolutional neural networks. ICWSM, 632–635.

Nguyen, K., Fookes, C., & Sridharan, S. (2015, September). Improving Deep Convolutional Neural Networks with Unsupervised Feature Learning. *IEEE International Conference on Image Processing (ICIP)*, (pp. 2270-2271). Quebec City: IEEE. 10.1109/ICIP.2015.7351206

Nie, D., Zhang, H., Adeli, E., Liu, L., & Shen, D. (2016). 3d deep learning for multi-modal imaging-guided survival time prediction of brain tumor patients. *Proc. MICCAI*, 212–220. 10.1007/978-3-319-46723-8_25

Nie, L., Wang, M., Zhang, L., Yan, S., Zhang, B., & Chua, T. S. (2015, August). Disease inference from health-related questions via sparse deep learning. *IEEE Transactions on Knowledge and Data Engineering*, *27*(8), 2107–2119. doi:10.1109/TKDE.2015.2399298

O'Shea, Erpek, & Clancy. (2017). *Deep learning based MIMO communications*. arXiv preprint arXiv:1707.07980

O'Shea, Hitefield, & Corgan. (2016). End-to-end radio traffic sequence recognition with recurrent neural networks. *Proc. IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 277–281.

O'Shea, Karra, & Clancy. (2016). Learning to communicate: Channel auto-encoders, domain specific regularizers, and attention. *Proc. IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, 223–228.

Obenshain, M. K. (2004). Application of data mining techniques to healthcare data. *Infection Control and Hospital Epidemiology*, *25*(8), 690–695. doi:10.1086/502460 PMID:15357163

Oda, T., Obukata, R., Ikeda, M., Barolli, L., & Takizawa, M. (2017). Design and implementation of a simulation system based on deep Q-network for mobile actor node control in wireless sensor and actor networks. *Proc. 31st IEEE International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, 195–200. 10.1109/WAINA.2017.67

Oh, Jeon, Jung, & Youn. (2016). *Smart diagnosis of journal bearing rotor systems: Unsupervised feature extraction scheme by deep learning*. Academic Press.

Oleh. (n.d.). *Automatic Plant Counting using Deep NNs* (Master of Science thesis). http://er.ucu.edu.ua/handle e/1/1334

Oliveira, Cunha, Silva, & Netto. (n.d.). *A Scalable Machine Learning System for Pre-Season Agriculture Yield Forecast*. arXiv:1806.09244

Olsen, Konovalov, Philippa, Ridd, Wood, Johns, … White. (2019). DeepWeeds: A Multiclass Weed Species Image Dataset for DL. *Scientific Reports, 9*(2058). 2 doi:10.103841598-018-38343-3

Olson, D. L., & Delen, D. (2008). *Advanced data mining techniques* (1st ed.). Springer Publishing Company.

Ong, B. T., Sugiura, K., & Zettsu, K. (2015). Dynamically pre-trained deep recurrent neural networks using environmental monitoring data for predicting pm2. 5. *Neural Computing & Applications*, *27*, 1–14.

Ong, B. T., Sugiura, K., & Zettsu, K. (2016). Dynamically pre-trained deep recurrent neural networks using environmental monitoring data for predicting PM2. 5. *Neural Computing & Applications*, *27*(6), 1553–1566. doi:10.100700521-015-1955-3

Ortiz, A. M., Hussein, D., Park, S., Han, S. N., & Crespi, N. (2014, June). The cluster between Internet of Things and social networks: Review and research challenges. *IEEE Internet Things J.*, *1*(3), 206–215. doi:10.1109/JIOT.2014.2318835

Ostermann, A. M. (2000). Face and 2-D mesh animation in MPEG-4. *Signal Processing Image Communication*, *15*(4), 387–421.

OWASP mobile top 10. (n.d.). Retrieved from https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10

Owens, J. D., Houston, M., Lubeke, D., Green, S., & Stone, J. E. (2008). GPU computing. *Proceedings of the IEEE*, *96*(5), 879–899. doi:10.1109/JPROC.2008.917757

Oyama, Y., Ben-Nun, T., Hoefler, T., & Matsuoka, S. (2018). Accelerating deep learning frameworks with micro-batches. *IEEE International Conference on Cluster Computing (CLUSTER)*, 402-412. 10.1109/CLUSTER.2018.00058

Padgett, C. a. (1997). Representing face images for emotion classification. *Advances in Neural Information Processing Systems*, 894–900.

Pandzic, I. S. (2002). *MPEG-4 Facial Animation: The Standard, Implementation and Applications*. Chichester, UK: John Wiley&Sons. doi:10.1002/0470854626

Pang, Z., Chen, Q., Han, W., & Zheng, L. (2015). Value-centric design of the internet-of things solution for food supply Chain: Value creation, sensor portfolio and information fusion. *Information Systems Frontiers*, *17*(2), 289–319. doi:10.100710796-012-9374-9

Pan, S. J., & Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359. doi:10.1109/TKDE.2009.191

Pantic, M. A. (2010.). Induced disgust, happiness and surprise: an addition to the mmi facial expression database. *Proc. 3rd Intern. Workshop on EMOTION (satellite of LREC): Corpora for Research on Emotion and Affect.*

Pantic, M., & Patras, I. (2006). Dynamics of facial expression: recognition of facial actions and their temporal segments from face profile image sequences. *Systems, Man, and Cybernetics, Part B: Cybernetics IEEE Transactions on*, *36*(2), 433–449. PMID:16602602

Parke, F. I. (1982). Parameterized models for facial animation. *IEEE Computer Graphics and Applications*, *2*(9), 61–68. doi:10.1109/MCG.1982.1674492

Parker, D. B., & Logic, L. (1985). *Technical Report TR-47.* Massachusetts Institute of Technology.

Parkes, J., Guha, I. N., Roderick, P., & Rosenberg, W. (2006). Performance of serum marker panels for liver fibrosis in chronic hepatitis C. *Journal of Hepatology*, *44*(3), 462–474. doi:10.1016/j.jhep.2005.10.019 PMID:16427156

Park, J.-Y., Luo, H., & Kim, W. H. (2015,October). Factors affecting students' completion: A study of an online Master's program. *International Conference of Educational Innovation through Technology (EITT)*, (pp. 275-278). 10.1109/EITT.2015.33

Parrott. (2000, October). *Emotions in Social Psychology.* Psychology Press.

Pascanu, R., Gulcehre, C., Cho, K., & Bengio, Y. (2013). *How to construct deep recurrent NNs*. arXiv preprint arXiv:1312.6026v5[cs.NE]

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., & DeVito, Z. (2017). Automatic differentiation in pytorch. *31st Conference on Neural Information Processing Systems (NIPS 2017)*, 1-4.

Patel, S. B., Yadav, P. K., & Shukla, D. P. (2013). Predict the diagnosis of heart disease patients using classification mining techniques. *IOSR Journal of Agriculture and Veterinary Science, 4*(2), 61-64.

Paulo Victor, R. (2018). Multi-objective Reinforcement Learning for Cognitive Satellite Communications using Deep Neural Network Ensembles. *IEEE Journal on Selected Areas in Communications*.

Pei, W., Li, J., Li, H., Gao, H., & Wang, P. (2017). ASCAA: API-level security certification of android applications. *IET Software, 11*(2), 55-63. doi:10.1049/iet-sen.2015.0040

Pentland, K. A. (1991). Recognition of facial expression from optical flow. *IEICE Transactions on Information and Systems*, *74*(10), 3474–3483.

Pentland, P. M. A. (1991). Face recognition using eigenfaces. *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91., IEEE Computer Society Conference on*, 586-591.

Pereira, D. R., Papa, J. P., Saraiva, G. F. R., & Souza, G. M. (2018). Automatic classification of plant electrophysiological responses to environmental stimuli using machine learning and interval arithmetic. *Computers and Electronics in Agriculture*, *145*, 35–42. doi:10.1016/j.compag.2017.12.024

Perumal, K., & Manohar, M. (2017). *A Survey on Internet of Things: Case Studies, Applications, and Future Directions*. Cham: Springer International Publishing.

Peter, A. (2017). *A Practical Introduction to Deep Learning with Caffe*. Retrieved may 31, 2019 from https://panderson.me/images/Caffe.pdf

Petridis, S. B. (2013). *The MAHNOB laughter database*. Academic Press.

Phan, N., Dou, D., Piniewski, B., & Kil, D. (2015). Social restricted Boltzmann machine: Human behavior prediction in health social networks. *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, 424–431. 10.1145/2808797.2809307

Phung, L. T., Chau, V. T., & Phung, N. H. (2015, November). Extracting Rule RF in Educational Data Classification from a Random Forest to Interpretable Refined Rules. *International Conference on Advanced Computing and Applications (ACOMP)*, (pp. 20-27). 10.1109/ACOMP.2015.13

Pimentel, J. F., Murta, L., Braganholo, V., & Freire, J. (2017). noWorkflow: A tool for collecting, analyzing, and managing provenance from python scripts. *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, *10*(12), 1841–1844. doi:10.14778/3137765.3137789

Pitas, I. K. (2007). Facial expression recognition in image sequences using geometric deformation features and support vector machines. *Image Processing*. *IEEE Transactions on*, *16*(1), 172–187.

Poggi, M., & Mattoccia, S. (2016). A wearable mobility aid for the visually impaired based on embedded 3d vision and deep learning. *Proceedings - IEEE Symposium on Computers and Communications*, 208–213. doi:10.1109/ISCC.2016.7543741

Polyzotis, N., Roy, S., Whang, S. E., & Zinkevich, M. (2017, May). Data management challenges in production machine learning. In *Proceedings of the 2017 ACM International Conference on Management of Data* (pp. 1723-1726). 10.1145/3035918.3054782

Popovic, B. (2007). Automatic rigging and animation of 3d characters. *ACM Transactions on Graphics*, *26*(3), 72. doi:10.1145/1276377.1276467

Pouladzadeh, P., Kuhad, P., Peddi, S. V. B., Yassine, A., & Shirmohammadi, S. (2016). Food calorie measurement using deep learning neural network. *Proc. IEEE Int. Instrum. Meas. Technol. Conf. Proc*., 1–6. 10.1109/I2MTC.2016.7520547

Pound, M. P. (2016). Deep Machine Learning provides state-of-the-art performance in Punjani. *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, 3223-3230.

Pradeep, A., & Das, S., & J, J. (2015). Students Dropout Factor Prediction Using EDM Techniques. *International Conference on Soft-Computing and Network Security*. 10.1109/ICSNS.2015.7292372

Preethi, P., & Asokan, R. (2019). A high secure medical mage storing and sharing in cloud environment using hex code cryptography method—Secure genius. *Journal of Medical Imaging and Health Informatics*, *9*(7), 1337–11345. doi:10.1166/jmihi.2019.2757

Pulido-Calvo, I., & Gutierrez-Estrada, J. C. (2009). Improved irrigation water demand forecasting using a soft-computing hybrid model. *Biosystems Engineering*, *102*(2), 202–218. doi:10.1016/j.biosystemseng.2008.09.032

Pulkit, S. (2018). *An Introductory Guide to Deep Learning and Neural Networks.* Retrieved may 31, 2019 from https://www.analyticsvidhya.com/blog/2018/10/

Punlumjeak, W., & Rachburee, N. (2015, October). A Comparative Study of Feature Selection Techniques for Classify Student Performance. *7th International Conference on Information Technology and Electrical Engineering (ICITEE)*, (pp. 425-429). 10.1109/ICITEED.2015.7408984

Putin, E. (2016). Deep biomarkers of human aging: Application of deep neural networks to biomarker development. *Aging (Albany NY)*, *8*(5), 1–021.

Python, P. (2011). Scikit-learn: Machine Learning. *JMLR*, *12*, 2825–2830.

Qi, Yi, Su, & Guibas. (2017). PointNet++: Deep hierarchical feature learning on point sets in a metric space. Advances in Neural Information Processing Systems, 5099–5108.

Qin, F., Liu, D., Sun, B., Ruan, L., Ma, M., & Wang, H. (2016). Identification of Alfalfa Leaf Diseases Using Image Recognition Technology. *PLoS One*, *11*(12), 1–26. doi:10.1371/journal.pone.0168274 PMID:27977767

Qi, Z., Wang, T., Song, G., Hu, W., Li, X., & Zhang, Z. M. (2018). Deep air learning: Interpolation, prediction, and feature analysis of fine-grained air quality. *IEEE Transactions on Knowledge and Data Engineering*, *30*(12), 2285–2297. doi:10.1109/TKDE.2018.2823740

Quang, D., Chen, Y., & Xie, X. (2014). Dann: A deep learning approach for annotating the pathogenicity of genetic variants. *Bioinformatics (Oxford, England)*, *31*(5), 761–763. doi:10.1093/bioinformatics/btu703

Rahmana, Arkoa, Alia, Khanb, Wasifa, Janic, & Kabirb. (2018). *Identification and Recognition of Rice Diseases and Pests 32 Using Deep Convolutional NNs*. Submitted to Expert Systems with Applications.

Rahnemoonfar, M., & Sheppard, C. (2017). Deep count: Fruit counting based on deep simulated learning. *Sensors (Basel)*, *17*(4), 905. doi:10.339017040905

Rajalingham, R., Issa, E. B., Bashivan, P., Kar, K., Schmidt, K., & DiCarlo, J. J. (2018). Large-scale, high-resolution comparison of the core visual object recognition behavior of humans, monkeys, and state-of-the-art deep artificial neural networks. *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*, *38*(33), 7255–7269. doi:10.1523/JNEUROSCI.0388-18.2018

Rajanna, A. R., Aryafar, K., Shokoufandeh, A., & Ptucha, R. (2015). Deep Neural Networks: A Case Study for Music Genere Classification. *14th International Conference on Machine Learning and Applications (ICMLA)*, (pp. 665-660). 10.1109/ICMLA.2015.160

Rajendran, S. (2018). *Deep learning models for wireless signal classification with distributed low-cost spectrum sensors. IEEE Transactions on Cognitive Communications and Networking*.

Ramsundar, Kearnes, Riley, Webster, Konerding, & Pande. (2015). *Massively multitask networks for drug discovery*. ArXiv e-prints.

Rangra, K., & Bansal, K. L. (2014). Comparative study of data mining tools. *International Journal of Advanced Research in Computer Science and Software Engineering*, *4*(6), 216–223.

Rao, P., Pereira, N. A., & Srinivasan, R. (2016). *Convolutional neural networks for lung cancer screening in computed tomography scans*. IEEE International Conference on Contemporary Computing and Informatics (IC3I), Noida, India. 10.1109/IC3I.2016.7918014

Raouzaiou, A., Tsapatsoulis, N., Karpouzis, K., & Kollias, S. (2002). Parameterized facial expression synthesis based on MPEG-4. *EURASIP Journal on Advances in Signal Processing*, *10*(10), 1–18. doi:10.1155/S1110865702206149

Ravi, D., Wong, C., Lo, B., & Yang, G. Z. (2016). Deep learning for human activity recognition: A resource efficient implementation on low-power devices. *Proc. 13th Int.Conf.Wearable Implantable Body Sens.Netw.*, 71–76. 10.1109/BSN.2016.7516235

Ray, S., Jin, Y., & Raychowdhury, A. (2016). The changing computing paradigm with internet of things: A tutorial introduction. *IEEE Design & Test*, *33*(2), 76–96. doi:10.1109/MDAT.2016.2526612

Razzak, M. I., Naz, S., & Zaib, A. (2017). *Deep learning for medical image processing: Overview, challenges and future. Classification in BioApps, Lecture Notes in Computational Vision and Biomechanics, 26.*

Rebetez, J., Satiza'Bal, H. F., & Mota, M. (2016). Augmenting a convolutional neural network with local histograms - A case study in crop classification from high-resolution UAV imagery. *ESANN 2016 proceedings*, *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*.

Reddy, Sarkar, & Giering. (2016). *Anomaly detection and fault disambiguation in large flight data: A multi-modaldeep auto-encoder approach*. In Annual conference of the prognostics and health management society, Denver, CO.

Regev, A., Berho, M., Jeffers, L. J., Milikowski, C., Molina, E. G., Pyrsopoulos, N. T., ... Schiff, E. R. (2002). Sampling error and intra-observer variation in liver biopsy in patients with chronic HCV infection. *The American Journal of Gastroenterology*, *97*(10), 2614–2618. doi:10.1111/j.1572-0241.2002.06038.x PMID:12385448

Ren, Y., & Wu, Y. (2014). Convolutional deep belief networks for feature extraction of EEG signal. In *Neural Networks (IJCNN)*, *2014 International Joint Conference on*, (pp. 2850–2853). IEEE. 10.1109/IJCNN.2014.6889383

Rivas, A., Chamoso, P., González-Briones, A., & Corchado, J. M. (2018). Detection of Cattle Using Drones and Convolutional NNs. *Sensors (Basel)*, *18*(7), 2048. doi:10.339018072048

Robertson, S. E., & Walker, S. (1994). Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 232–241. 10.1007/978-1-4471-2099-5_24

Robinson, S. (2017). *The Best Machine Learning Libraries in Python*. Stack Abuse.

Rodríguez Alvarez, J. (2018). *Body condition estimation on cows from depth images using Convolutional NNs* (Vol. 155). Comput. Electron. Agric.

Roger Highfield, R. W. (2009, February). How your looks betray your personality. *New Scientist*.

Rosenblatt, F. (1957). *The Perceptron — A Perceiving and Recognizing Automaton*. Tech. Rep. 85 460-1. Cornell Aeronautical Laboratory.

Rosen, H. (2011). Clinical practice, chronic Hepatitis C infection. *The New England Journal of Medicine*, *364*(25), 2429–2438. doi:10.1056/NEJMcp1006613 PMID:21696309

Roth, H. R. (2015). Anatomy-specific classification of medical images using deep convolutional nets. *Proc. IEEE Int. Symp. Biomed. Imag.*, 101–104. 10.1109/ISBI.2015.7163826

Russello, H. (2018). *Convolutional NNs for Crop Yield Prediction using Satellite Images* (MSc Artificial Intelligence Master Thesis). University of Amsterdam.

Ryder, B., & Wortmann, F. (2017). Autonomously detecting and classifying traffic accident hotspots. *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and 2017 ACM International Symposium on Wearable Computers*, 365–370. 10.1145/3123024.3123199

Sainath, T., Mohamed, A. R., Kingsbury, B., & Ramabhadran, B. (2013, May). *Deep convolutional neural networks for LVCSR. In 2013 IEEE international conference on acoustics, speech and signal processing* (pp. 8614–8618). IEEE. doi:10.1109/ICASSP.2013.6639347

Salakhutdinov, R., & Hinton, G. (2009) Semantic hashing. *Int J Approximate, Reasoning, 50*(7), 969–978. 10.1016/j.ijar.2008.11.006

Salam, H. (2013). *Multi-Object modeling of the face. Signal and Image processing* (Thesis). Université Rennes 1.

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, *24*(5), 513–523. doi:10.1016/0306-4573(88)90021-0

Samal, A., & Iyengar, P. A. (1992). Automatic recognition and analysis of human faces and facial expressions: A survey. *Pattern Recognition*, *1*(25), 65–77. doi:10.1016/0031-3203(92)90007-6

Sanchez, P. A., & Swaminathan, M. S. (2005). Cutting world hunger in half. *Science*, *307*(5708), 357–359. doi:10.1126cience.1109057 PMID:15661994

Santoni, M. M., Sensuse, D. I., Arymurthy, A. M., & Fanany, M. I. (2015). Cattle race classification using gray level co-occurrence matrix convolutional neural networks. *Procedia Computer Science*, *59*, 493–502. doi:10.1016/j.procs.2015.07.525

Sarangam Kodati, D. R. V. (2018). *Analysis of heart disease using in data mining tools Orange and Weka*. *Global Journal of Computer Science and Technology*.

Saravanan, K., Asokan, R., & Venkatachalam, K. (Nov 2013). Neuro-Fuzzy Based Clustering Of Distributed Denial of Service (DDoS) Attack Detection Mechanism. *International Information Institute (Tokyo). Information, 16*(11), 8137.

Sathish, M., & Sridhar, D. (2015). Prediction of heart diseases in data mining techniques. *International Journal of Computer Trends and Technology*, *24*(1), 45–49. doi:10.14445/22312803/IJCTT-V24P109

Schölkopf, B., Smola, A. J., & Bach, F. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT Press.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). *Proximal policy optimization algorithms.* arXiv preprint arXiv:1707.06347

Scroxton, A. (2016). *UK IoT research hub opens with support from academic world, ComputerWeekly.com*. Retrieved from https://www. computerweekly.com/ news/4500270034/UK-IoT-research-hub-opens-with-support-from-academic-world

Sedjelmaci, H., Senouci, S. M., & Al-Bahri, M. (2016). A lightweight anomaly detection technique for low-resource iot devices: A game-theoretic methodology. *IEEE International Conference on Communications (ICC)*, 1–6. 10.1109/ICC.2016.7510811

Seetharaman, K., & Jeyakarthic, M. (2014). Statistical Distributional Approach for Scale and Rotation Invariant Color Image Retrieval Using Multivariate Parametric tests and Orthogonality Condition. *Journal of Visual Communication and Image Representation*, *25*(5), 727–739. doi:10.1016/j.jvcir.2014.01.004

Seetharaman, K., & Palanivel, N. (2013). Texture Characterization, Representation, Description and Classification Based on a Family of Full Range Gaussian Markov Random Field Model. *International Journal of Image and Data Fusion*, *4*, 342–362. doi:10.1080/19479832.2013.804007

Sehal, Gupta, & Paneri. (2017). *Crop planning using stochastic visual optimization*. Academic Press.

Selfridge, O. G. (1959). Pandemonium: A paradigm of learning. Mechanisation of Thought Processes. *NPL Symposium 10*.

Sen, S., Aydogan, E., & Aysan, A. I. (2018). Coevolution of Mobile Malware and Anti-Malware. *IEEE Transactions on Information Forensics and Security*, *13*(10), 2563–2574. doi:10.1109/TIFS.2018.2824250

Serpanos, D. (2018). The Cyber-Physical Systems Revolution. *Computer*, *51*(3), 70–73. doi:10.1109/MC.2018.1731058

Shalev-Shwartz, S. (2012). Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, *4*(2), 107–194. doi:10.1561/2200000018

Shamsabadi, A. S., Haddadi, H., & Cavallaro, A. (2018). Distributed one-class learning. *Proc. IEEE International Conference on Image Processing (ICIP)*.

Shan, J., & Li, L. (2016). A deep learning method for microaneurysm detection in fundus images. *Proc. IEEE Connected Health, Appl., Syst. Eng. Technol.*, 357–358. 10.1109/CHASE.2016.12

Shan, C. S., Gong, S., & McOwan, P. W. (2009). Facial expression recognition based on local binary patterns: A comprehensive study. *Image and Vision Computing*, *27*(6), 803–816. doi:10.1016/j.imavis.2008.08.005

Shanmugapriya, M., Nehemiah, K. H., Bhuvaneswaran, R. S., Kannan, A., & Sweetlin, J. D. (2017). Fuzzy discretization based classification of medical data. *Research Journal of Applied Sciences, Engineering and Technology*, *8*(8), 291–298. doi:10.19026/rjaset.14.4953

Shen, Han, Yang, Yang, Wang, Li, & Wen. (2018). CS-CNN: Enabling robust and efficient Convolutional neural networks inference for internet-of-things applications. *IEEE Access, 6*(13), 439–448. doi:10.1109/access.2018.2810264

Shiau, W.-L., & Chau, P. Y. (2016). Understanding behavioral intention to use a cloud computing classroom: A multiple model comparison approach. *Information & Management*, *53*(3), 355–365. doi:10.1016/j.im.2015.10.004

Shin, H., Lu, L., Kim, L., Seff, A., Yao, J., & Summers, R. M. (2015). *Interleaved text/image deep mining on a large-scale radiology database for automated image interpretation*. Available: https://arxiv.org/abs/1505.00670

Shin, H.-C., Roth, H. R., Gao, M., Lu, L., Xu, Z., Nogues, I., ... Summers, R. M. (2016, May). Deep convolutional neural networks for computeraided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE Transactions on Medical Imaging*, *35*(5), 1285–1298. doi:10.1109/TMI.2016.2528162

Shiraz, M., Gani, A., Khokhar, R., Rahman, A., Iftikhar, M., & Chilamkurti, N. (2017). A distributed and elastic application processing model for mobile cloud computing. *Wirel Pers Commun, 95*(4), 4403–4423.

Shook, J. (2018). Integrating genotype and weather variables for soybean yield prediction using deep learning. *bioRxiv*, *25*. doi:10.1101/331561doi

Shoukat, A. (2013). Factors Contributing to the Students' Academic Performance: A Case Study of Islamia University Sub-Campus. *American Journal of Educational Research*, 283–289.

Silverman, B. (1986). *Density estmaation for statistics and data analysis.* Chapman & Hall/CRC Press.

Simon, L., Jeffrey, C., & Jason, M. S. (2011). Deformable model fitting by regularized landmark mean-shift. *International Journal of Computer Vision*, *91*(2), 200–215. doi:10.100711263-010-0380-4

Simonyan, K., & Zisserman, A. (2014). Very deep Convolutional networks for large-scale image recognition. *International Conference on Learning Representations (ICLR 2015).*

Sladojevic, Arsenovic, Anderla, Culibrk, & Stefanovic. (2016). Deep NNs Based Recognition of Plant Diseases by Leaf Image Classification. *Computational Intelligence and Neuroscience.* doi:10.1155/2016/3289801

Smartpharma.co.za. (2015). *Welcome to Smart Pharmaceuticals*. Available at: http://www.smartpharma.co.za/

Socher, R., Lin, C. C., Ng, A., & Manning, C. (2011). Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning*. Omnipress.

Socher, R., & Perelygin, A. (2013). *Recursive deep models for semantic compositionality over a sentiment tree bank*. Emnlp.

Song, X., Zhang, G., Liu, F., Li, D., Zhao, Y., & Yang, J. (2016). Modeling spatio-temporal distribution of soil moisture by deep learning-based cellular automata model. *Journal of Arid Land*, *8*(5), 734–748. doi:10.100740333-016-0049-0

Sperduit, A., & Starita, A. (1997). Supervisedneurl networks for the classification of structures. *IEEE Trasctions on Neural Network, 8*(3).

Stankovic, J. A. (2014). Research directions for the Internet of Things. *IEEE Int. Things J.*, *1*(1), 3–9. doi:10.1109/JIOT.2014.2312291

Sterling, R. K., & ... . (2006). Development of a simple non-invasive index to predict significant fibrosis in patients with HIV/ HCV co-infection. *Hepatology (Baltimore, Md.)*, *43*, 1317–1325. doi:10.1002/hep.21178 PMID:16729309

Stewart, E. L., & McDonald, B. A. (2014). Measuring quantitative virulence in the wheat pathogen zymoseptoria tritici using high-throughput automated image analysis. *Phytopathology*, *104*(9), 985–992. doi:10.1094/PHYTO-11-13-0328-R PMID:24624955

Strange, R. N., & Scott, P. R. (2005). Plant disease: A threat to global food security. *Phytopathology*, *43*(1), 83–116. doi:10.1146/annurev.phyto.43.113004.133839 PMID:16078878

Sugie, N., M. K. (1978). A Perliminary note on pattern recognition of human emotional expression. *Proc. 4th Int.Joint Conf. on Pattern Recognition*, 408-410.

Suk, H.-I., Lee, S.-W., & Shen, D. (2014). Hierarchical feature representation and multimodal fusion with deep learning for ad/mci diagnosis. *NeuroImage*, *101*, 569–582. doi:10.1016/j.neuroimage.2014.06.077

Suk, H.-I., Lee, S.-W., & Shen, D. (2015). Latent feature representation with stacked auto-encoder for AD/MCI diagnosis. *Brain Structure & Function*, *220*(2), 841–859. doi:10.100700429-013-0687-3

Sukhija, K., Jindal, D. M., & Aggarwal, D. N. (2015, October). The Recent State of Educational Data Mining: A Survey and Future Visions. *IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education*, (pp. 354-359). doi: 10.1109/MITE.2015.7375344

Sun, L., Jia, K., Chan, T.-H., Fang, Y., Wang, G., & Yan, S. (2014). DL-SFA: Deeply-learned slow feature analysis for action recognition. *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*., 2625–2632. 10.1109/CVPR.2014.336

Sundarkumar, G. & Vadlamani, R. (2013). Malware detection by text and data mining. *2013 IEEE International Conference on Computational Intelligence and Computing Research, IEEE ICCIC 2013*, 1-6. . doi:10.1109/ICCIC.2013.6724229

Sun, W., Shao, S., Zhao, R., Yan, R., Zhang, X., & Chen, X. (2016). A sparse auto-encoder-based deep neural network approach for induction motor faults classification. *Measurement*, *89*, 171–178. doi:10.1016/j.measurement.2016.04.007

Sun, Y., Wu, L., Wu, S., Li, S., Zhang, T., Zhang, L., ... Cui, X. (2017, June). Attacks and countermeasures in the internet of vehicles. *Annals of Telecommunications, 72*(5), 283–295 doi:10.100712243-016-0551-6

Suryawan, A., & Putra, E. (2016). Analysis of Determining Factors for Successful Student's GPA Achievement. *11th International Conference on Knowledge, Information and Creativity Support Systems (KICSS)*, (pp. 1-7). Yogyakarta: Academic Press.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Advances in neural information processing systems (pp. 3104–3112). Academic Press.

Sutskever, O. (2014). Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 3104–3112.

Symantec. (2016). *Internet security threat report*. Retrieved from https://www.east.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., …Rabinovich, A. (2015). *Going deeper with convolutions*. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1-9). IEEE.

Tai, A. P., Martin, M. V., & Heald, C. L. (2014). Threat to future global food security from climate change and ozone air pollution. *Nature Climate Change*, *4*(9), 817–821. doi:10.1038/nclimate2317

Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., & Liang, J. (2016, May). Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, *35*(5), 1299–1312. doi:10.1109/TMI.2016.2535302

Tan, J., & Koo, S. G. M. (2014). A Survey of Technologies in Internet of Things. *IEEE International Conference on Distributed Computing in Sensor Systems*, 269-274.

Tan, L., & Wang, N. (2010). Future Internet: The Internet of Things. *3rd International Conference On Advanced Computer Theory And Engineering (ICACTE),* 376-380.

Tan, Q. (2017). Emerging Technologies: IoT, Big Data, and CPS with Sensory Systems. *Hindawi Journal of Sensors*, *2018*, 3407542. doi:10.1155/2018/3407542

Tao, S., Zhang, T., Yang, J., Wang, X., & Lu, W. (2015). Bearing fault diagnosis method based on stacked autoencoder and softmax regression. *Control Conference (CCC)*, *2015 34th Chinese*, 6331–6335. 10.1109/ChiCC.2015.7260634

Tao, G., Zheng, Z., Guo, Z., & Lyu, M. R. (2018). MalPat: Mining Patterns of Malicious and Benign Android Apps via Permission-Related APIs. *IEEE Transactions on Reliability*, *67*(1), 355–369. doi:10.1109/TR.2017.2778147

Tao, W. R. D. (2016). Non-local auto-encoder with collaborative stabilization for image restoration. *IEEE Transactions on Image Processing*, *25*(5), 2117–2129. doi:10.1109/TIP.2016.2541318

Taylor, C. J. T. (1992). Active Shape Models - 'Smart Snakes'. *British Machine Vision Conference*, 266-275.

Teimouri, N., Dyrmann, M., Nielsen, P. R., Mathiassen, S. K., Somerville, G. J., & Jørgensen, R. N. (2018). Weed Growth Stage Estimator Using Deep Convolutional NNs. *Sensors (Basel)*, *18*(5), 1580. doi:10.339018051580 PMID:29772666

Thirukovalluru, R., Dixit, S., Sevakula, R. K., Verma, N. K., & Salour, A. (2016). Generating feature sets for fault diagnosis using denoising stacked auto-encoder. In *Prognostics and Health Management (ICPHM), 2016 IEEE International Conference on*. IEEE. https://towardsdatascience.com/a-comphrehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a5

Tian, K., Shao, M., Zhou, S., & Guan, J. (2015). Boosting compound-protein interaction prediction by deep learning. *Proc. IEEE Int. Conf. Bioinformat. Biomed.*, 29–34.

Tian, Y.-T. (2001). Recognizing action units for facial expression analysis. *Pattern Analysis and Machine Intelligence. IEEE Transactions on*, *23*(2), 97–115.

Tompson, J. J., Jain, A., LeCun, Y., & Bregler, C. (2014). Joint training of a convolutional network and a graphical model for human pose estimation. In Advances in neural information processing systems (pp. 1799-1807). Academic Press.

Tooa, E. C., Yujiana, L., Njukia, S., & Yingchunb, L. (2018). A comparative study of ðne-tuning DL models for plant disease identiðcation. *Computers and Electronics in Agriculture*. doi:10.1016/j.compag.2018.03.032]

Tran, D., Ray, J., Shou, Z., Chang, S. F., & Paluri, M. (2017). *Convnet architecture search for spatiotemporal feature learning*. arXiv preprint arXiv:1708.05038

Tseng, D., Wang, D., Chen, C., Miller, L., Song, W., Viers, J., . . . Towards, A. P. I. (2018). *DL to Infer Local Soil Moisture Conditions from Synthetic Aerial Agricultural Images*. In 2018 IEEE 14th International Conference on Automation Science and Engineering, CASE, Munich, Germany.

Turaga, S. C., Murray, J. F., Jain, V., Roth, F., Helmstaedter, M., Briggman, K., ... Seung, H. S. (2010). Convolutional networks can learn to generate affinity graphs for image segmentation. *Neural Computation*, *22*(2), 511–538. doi:10.1162/neco.2009.10-08-881

Türkoğlu, M. (n.d.). Plant disease and pest detection using deep learning-based features. *Turkish Journal of Electrical Engineering and Computer Sciences*. doi:10.3906/elk-1809-181]

Tzortzis, G., & Likas, A. (2007, October). Deep Belief Networks for Spam Filtering. *IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)*, (pp. 306-309). Patras: Academic Press.

UNEP Smallholders. (2013). *Food Security, and the Environment. Rome: International Fund for Agricultural Development (IFAD)*. Retrieved from: https://www.ifad.org/documents/10180/66cac2414b643c2876d9c2d1f01d5dd

Valipour, S., Siam, M., Stroulia, E., & Jagersand, M. (2016). Parking-stall vacancy indicator system, based on deep convolutional neural networks. Internet of Things, 655–660. doi:10.1109/WF-IoT.2016.7845408

van Grinsven, M. J., van Ginneken, B., Hoyng, C. B., Theelen, T., & S'anchez, C. I. (2016, May). Fast convolutional neural network training using selective data sampling: Application to hemorrhage detection in color fundus images. *IEEE Transactions on Medical Imaging*, *35*(5), 1273–1284. doi:10.1109/TMI.2016.2526689

Vanschoren, J., Van Rijn, J. N., Bischl, B., & Torgo, L. (2017). OpenML: Networked science in machine learning. *SIGKDD Explorations*, *15*(2), 49–60. doi:10.1145/2641190.2641198

Vedaldi, A., & Lenc, K. (2015). Matconvnet: Convolutional neural networks for matlab. *Proceedings of the 23rd ACM international conference on Multimedia*, 689-692. 10.1145/2733373.2807412

Ventura, D. (2009). *SVM example.* Lecture notes.

Verma, N. K., Gupta, V. K., Sharma, M., & Sevakula, R. K. (2013). Intelligent condition based monitoring of rotating machines using sparse autoencoders. *Prognostics and Health Management (PHM)*, *2013 IEEE Conference on*, 1–7.

Vetrivel, A., Gerke, M., Kerle, N., Nex, F., & Vosselman, G. (2018). Disaster damage detection through synergistic use of deep learning and 3D point cloud features derived from very high resolution oblique aerial images, and multiple-kernel-learning. *ISPRS Journal of Photogrammetry and Remote Sensing*, *140*, 45–59. doi:10.1016/j.isprsjprs.2017.03.001

Vetter, V. B. (1999). A morphable model for the synthesis of 3d faces. *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, 187–194.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P.-A. (2010). Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *Journal of Machine Learning Research*, *11*, 3371–3408.

VirusShare. (2019). Retrieved from https://virusshare.com/

Vlacheas, P., Giaffreda, R., Stavroulaki, V., Kelaidonis, D., Foteinos, V., Poulios, G., ... Moessner, K. (2013, June). Enabling smart cities through a cognitive management framework for the Internet of Things. *IEEE Communications Magazine*, *51*(6), 102–111. doi:10.1109/MCOM.2013.6525602

Vora, D. R., & Iyer, K. (2018, December). *Implementing a Deep Learning Approach for the Performnace Prediction in Educational Information System (*Unpublished Thesis). Amity University, Mumbai, India.

Vora, D., & Iyer, K. (2017). A Survey of Inferences from Deep Learning Algorithms. *Journal of Engineering and Applied Sciences, 12*(SI), 9467-9472.

Vora, D., & Kamatchi, R. (2018). EDM – Survey of Performance Factors and Algorithms Applied. *International Journal of Engineering & Technology, 7*(2.6), 93-97.

Vu, T. H., Mousavi, H. S., Monga, V., Rao, A., & Rao, G. (2016). Histopathological image classification using discriminative feature oriented dictionary learning. *IEEE Transactions on Medical Imaging*, *35*(1), 738–751. doi:10.1109/TMI.2015.2493530 PMID:26513781

Wallbott, H. G. (1986). Cues and channels in emotion recognition. *Journal of Personality and Social Psychology*, 69–70.

Wang, Y. S. (2008). Enforcing convexity for improved alignment with constrained local models. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE. doi:10.1155/S1110865702203078

Wang, A., Song, C., Xu, X., Lin, F., Jin, Z., & Xu, W. (2015). Selective and compressive sensing for energy-efficient implantable neural decoding. *Proc. Biomed. Circuits Syst. Conf.*, 1–4. 10.1109/BioCAS.2015.7348375

Wang, B., Gu, X., Ma, L., & Yan, S. (2017). Temperature error correction based on BP neural network in meteorological wireless sensor network. *International Journal of Sensor Networks*, *23*(4), 265–278. doi:10.1504/IJSNET.2017.083532

Wang, G., Sun, Y., & Wang, J. (2017). Automatic Image-Based Plant Disease Severity Estimation Using Deep Learning. *Computational Intelligence and Neuroscience*, *2017*, 1–8. doi:10.1155/2017/2917536 PMID:28757863

Wang, J., Tang, J., Xu, Z., Wang, Y., Xue, G., Zhang, X., & Yang, D. (2017). Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach. *Proc. 36th Annual IEEE International Conference on Computer Communications (INFOCOM)*. 10.1109/INFOCOM.2017.8057090

Wang, L., Zhao, X., Pei, J., & Tang, G. (2016). Transformer fault diagnosis using continuous sparse autoencoder. *SpringerPlus*, *5*(1), 1. doi:10.118640064-016-2107-7

Wang, X., Gao, L., & Mao, S. (2015). PhaseFi: Phase fingerprinting for indoor localization with a deep learning approach. *Proc. IEEE Global Communications Conference (GLOBECOM)*, 1–6. 10.1109/GLOCOM.2015.7417517

Wang, X., Wang, X., & Mao, S. (2017). CiFi: Deep Convolutional neural networks for indoor localization with 5 GHz Wi-Fi. *Proc. IEEE International Conference on Communications (ICC)*, 1–6. 10.1109/ICC.2017.7997235

Wang, X., Zhou, Z., Xiao, F., Xing, K., Yang, Z., Liu, Y., & Peng, C. (2018). Spatio-temporal analysis and prediction of cellular traffic in metropolis. *IEEE Transactions on Mobile Computing*.

West & O'Shea. (2017). Deep architectures for modulation recognition. *Proc. IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 1–6.

Whitmore, A., Agarwal, A., & Xu, L. D. (2015). The Internet of Things: A survey of topics and trends. *Information Systems Frontiers*, *17*(2), 261–274. doi:10.100710796-014-9489-2

Wiering, M. A. M. S., Millea, A., Meijster, A., & Schomaker, L. (2016). *Deep Support Vector Machines for Regression Problems.* Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.718.987&rep=rep1&type=pdf

Wu, D., Mao, C., Wei, T., Lee, H., & Wu, K. (2012). DroidMat: Android Malware Detection through Manifest and API Calls Tracing. *2012 Seventh Asia Joint Conference on Information Security*, 62-69. 10.1109/AsiaJCIS.2012.18

Wulsin, D., Blanco, J., Mani, R., & Litt, B. (2010). Semi-supervised anomaly detection for eeg waveforms using deep belief nets. *Proc. 9th Int. Conf. Mach. Learn. Appl.*, 436–441. www.nytimes.com/2001/07/12/technology/news-watch-a-quick-way-to-search-for-images-on-the-web.html

Xi, O., Zhang, C., Zhou, P., & Jiang, H. (2016). *DeepSpace: An online deep learning framework for mobile big data to understand human mobility patterns*. arXiv preprint arXiv:1610.07009

Xia, D., Chen, P., Wang, B., Zhang, J., & Xie, C. (2018). Insect Detection and Classification Based on an Improved Convolutional NN. *Sensors (Basel)*, *18*(12), 4169. doi:10.339018124169 PMID:30486481

Xian, K. (2017). Internet of things online monitoring system based on cloud computing. *Int. Jo. Online Eng.*, *13*(9), 123–131. doi:10.3991/ijoe.v13i09.7591

Xiao, Q., Li, W., Chen, P., & Wang, B. (2018). Prediction of Crop Pests and Diseases in Cotton by Long Short Term Memory Network. *ICIC*, *2018*, 11–16. doi:10.1007/978-3-319-95933-7_2

Xin Shao, W., & Cheng, C. (2015). Weed seeds classification based on PCANet deep learning baseline. *IEEE Signal and Information Processing Association Annual Summit and Conference (APSIPA),* 408–415.

Xing, W., Guo, R., Petakovic, E., & Goggins, S. (2015). Participation-based student final performance prediction model through interpretable Genetic Programming: Integrating learning analytics, educational data mining and theory. *Computers in Human Behavior*, *47*, 168–181. doi:10.1016/j.chb.2014.09.034

Xiong, Y. (2017). *Install and Run cuda-convnet on Windows OS with VS2010*. Retrieved May 31, 2019 from http://yjxiong.me/others/install_cudaconvnet.html

Xu, J., Wang, B., & Li, J. (2017). *Deep learning application based on embedded GPU. IEEE International Conference on Electronics Instrumentation & Information Systems (EIIS)*, Harbin, China. 10.1109/EIIS.2017.8298723

Xu, T., Zhang, H., Huang, X., Zhang, S., & Metaxas, D. N. (2016). Multimodal deep learning for cervical dysplasia diagnosis. *Proc. MICCAI*, 115–123. 10.1007/978-3-319-46723-8_14

Yalc¸ın, H. (2016). Human activity recognition using deep belief networks. *Proc. Signal Process. Commun. Appl. Conf.*, 1649–1652. 10.1109/SIU.2016.7496073

Yang, K., & Zhang, Z. (2012). Summarize on IoT and exploration into technical system framework. *Proc. IEEE Symp. Robot. Appl. (ISRA),* 653–656.

Yang, X., Chen, S., Ding, V., & Pang, B. (2016). *A deep learning approach for tumor tissue image classification*. International Association of Science and Technology for Development (IASTED) International Conference on Biomedical Engineering, Innsbruck, Austria. 10.2316/P.2016.832-025

Yang, Xiao, & Lin. (2018). Feeding behaviour recognition for group-housed pigs with the FasterR-CNN. Computers and Electronics in Agriculture, 155, 453–460.

Yang, Y., Li, Y., & Fermüller, C. (2015). Robot learning manipulation action plans by watching unconstrained videos from the world wide web. *29th AAAI Conference on Artificial Intelligence (AAAI-15)*, Austin, TX.

Yan, Y., Qin, X., Wu, Y., Zhang, N., Fan, J., & Wang, L. (2015). A restricted Boltzmann machine based two-lead electrocardiography classification. *Proc. 12th Int. Conf. Wearable Implantable Body Sens. Netw.*, 1–9.

Yan, Z., Zhang, P., & Vasilakos, A. V. (2014). A survey on trust management for Internet of Things. *Journal of Network and Computer Applications*, *42*, 120–134. doi:10.1016/j.jnca.2014.01.014

Yan, Z., Zhan, Y., Peng, Z., Liao, S., Shinagawa, Y., Zhang, S., ... Zhou, X. S. (2016, May). Multi-instance deep learning: Discover discriminative local anatomies for bodypart recognition. *IEEE Transactions on Medical Imaging*, *35*(5), 1332–1343. doi:10.1109/TMI.2016.2524985

Yao, S., Hu, S., Zhao, Y., Zhang, A., & Abdelzaher, T. (2017). Deepsense: A unified deep learning framework for timeseries mobile sensing data processing. In *Proc. 26th International Conference on World Wide Web*, (pp. 351–360). International World Wide Web Conferences Steering Committee. 10.1145/3038912.3052577

Ye, Li, Juang, & Sivanesan. (2018). *Channel agnostic end-to-end learning based communication systems with conditional GAN*. arXiv preprint arXiv:1807.00447

Ye, H., Li, G. Y., & Juang, B.-H. (2018). Power of deep learning for channel estimation and signal detection in OFDM systems. *IEEE Wireless Communications Letters*, *7*(1), 114–117. doi:10.1109/LWC.2017.2757490

Yin, S., Li, X., Gao, H., & Kaynak, O. (2015, January). Data-based techniques focused on modern industry: An overview. *IEEE Transactions on Industrial Electronics*, *62*(1), 657–667. doi:10.1109/TIE.2014.2308133

Yi, X., Zhang, J., Wang, Z., Li, T., & Zheng, Y. (2018). Deep distributed fusion network for air quality prediction. *Proceedings of the 24th SIGKDD conference on Knowledge Discovery and Data Mining*. 10.1145/3219819.3219822

Yogapriya, J., Saravanabhavan, C., Asokan, R., Vennila, I., Preethi, P., & Nithya, B. (2018). A study of image retrieval system based on feature extraction, selection, classification and similarity measurements. *Journal of Medical Imaging and Health Informatics*, *8*(3), 479–484. doi:10.1166/jmihi.2018.2326

Yongsheng & Gao. (2003). Facial expression recognition from line-based caricatures. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on, 33*(3), 407-412.

You, J., Li, X., Low, M., Lobell, D., & Ermon, S. (2017). Deep Gaussian Process for Crop Yield Prediction Based on Remote Sensing Data. AAAI, 4559–4566.

Yousefi-Azar, M., Varadharajan, V., Hamey, L., & Tupakula, U. (2017). Autoencoder-based feature learning for cyber security applications. *Proc. IEEE International Joint Conference on Neural Networks (IJCNN)*, 3854–3861. 10.1109/IJCNN.2017.7966342

Yu, J., Weng, K., & Liang, G. (2013). A vision-based robotic grasping system using deep learning for 3D object recognition and pose estimation. *Robotics and Biomimetics (ROBIO), 2013 IEEE International Conference on*, 1175-1180.

Yu, J.-s., Chen, J., & Xiang, Z. (2015). A hybrid convolutional neural networks with extreme learning machine for WCE image classification. *Proc. IEEE Robot. Biomimetics*, 1822–1827.

Yuan, M., Wu, Y., & Lin, L. (2016). Fault diagnosis and remaining useful life estimation of aero engine using lstm neural network. *2016 IEEE International Conference on Aircraft Utility Systems (AUS)*, 135–140. 10.1109/AUS.2016.7748035

Yuan, Y., Zhang, M., Luo, P., Ghassemlooy, Z., Lang, L., Wang, D., ... Han, D. (2017). SVM-based detection in visible light communications. *Optik (Stuttgart)*, *151*, 55–64. doi:10.1016/j.ijleo.2017.08.089

Zabidi, M. N. A., Maarof, M. A., & Zainal, A. (2012). Challenges in high accuracy of malware detection. *IEEE Control and System Graduate Research Colloquium,* 123-125. doi: 10.1109/ICSGRC.2012.6287147

Zanella, A., Bui, N., Castellani, A., Vangelista, L., & Zorzi, M. (2014, February). Internet of Things for smart cities. *IEEE Internet Things J.*, *1*(1), 22–32. doi:10.1109/JIOT.2014.2306328

Zantalis, Koulouras, Karabetsos, & Kandris. (2019). A Review of Machine Learning and IoT in Smart Transportation. *Future Internet, 11*(94). www.mdpi.com/journal/futureinterne

Zelier, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. *Springer LNCS*, *8689*, 818–833.

Zeng & Liu. (2008). Markov random field-based statistical character structure modeling for handwritten Chinese character recognition. *On pattern analysis and machine intelligence, IEEE Trans., 30*(5).

Zeng. (2014). Convolutional neural networks for human activity recognition using mobile sensors. *Proc. MobiCASE*, 197–205. .mobicase.2014.257786 doi:10.4108/icst

Zhang, C., Sun, J. H., & Tan, K. C. (2015). Deep belief networks ensemble with multi-objective optimization for failure diagnosis. In *Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on*. IEEE. 10.1109/SMC.2015.19

Zhang, J., Zheng, Y., & Qi, D. (2017). Deep spatio-temporal residual networks for citywide crowd flows prediction. AAAI, 1655–1661.

Zhang, C., Lim, P., Qin, A., & Tan, K. C. (2016). Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. *IEEE Transactions on Neural Networks and Learning Systems*.

Zhang, C., Yue, P., Di, L., & Wu, Z. (2018). Automatic Identiðcation of Center Pivot Irrigation Systems from Landsat Images Using Convolutional Neural Networks. *Agriculture*, *8*(10), 147–156. doi:10.3390/agriculture8100147

Zhang, L., Li, Q.-Y., Duan, Y.-Y., Yan, G.-Z., Yang, Y.-L., & Yang, R.-J. (2012). Artificial neural network aided noninvasive grading evaluation of hepatic fibrosis by duplex ultrasonography. *BMC Medical Informatics and Decision Making*, *12*(1), 55. doi:10.1186/1472-6947-12-55 PMID:22716936

Zhang, M., Lin, H., Wang, G., Sun, H., & Fu, J. (2018, November 20). Mapping Paddy Rice Using a Convolutional NeuralNetwork (CNN) with Landsat Datasets in the Dongting Lake Area, China. *Remote Sensing*, *10*(11), 1840. doi:10.3390/rs10111840

Zhang, Q., Yang, L. T., Chen, Z., & Li, P. (2018). A survey on deep learning for big data. *Information Fusion*, *42*, 146–157. doi:10.1016/j.inffus.2017.10.006

Zhang, S., Zhou, J., Hu, H., Gong, H., Chen, L., Cheng, C., & Zeng, J. (2016). A deep learning framework for modeling structural features of rna-binding protein targets. *Nucleic Acids Research*, *44*(4), e32–e32. doi:10.1093/nar/gkv1025

Zhang, T., Antunes, H., & Aggarwal, S. (2014, February). Defending connected vehicles against malware: Challenges and a solution framework. *IEEE Internet Things J.*, *1*(1), 10–21. doi:10.1109/JIOT.2014.2302386

Zhang, X., Qiao, Y., Meng, F., Fan, C., & Zhang, M. (2018). Identification of Maize Leaf Diseases Using Improved Deep Convolutional NNs. *IEEE Access: Practical Innovations, Open Solutions*, *6*. doi:10.1109/ACCESS.2018.2844405]

Zhang, Y., Ji, Q., Zhu, Z., & Yi, B. (2008, October). Dynamic Facial Expression Analysis and Synthesis With MPEG-4 Facial Animation Parameters. *IEEE Transactions on Circuits and Systems for Video Technology*, *18*(10), 1383–1396. doi:10.1109/TCSVT.2008.928887

Zhao, C., Zheng, W., Gong, L., Zhang, M., & Wang, C. (2018). Quick and Accurate Android Malware Detection Based on Sensitive APIs. *2018 IEEE International Conference on Smart Internet of Things (SmartIoT)*, 143-148. doi: 10.1109/SmartIoT.2018.00034

Zhao, G. E. (2011). *Facial expression recognition from near-infrared videos*. Academic Press.

Zhao, R., Wang, J., Yan, R., & Mao, K. (2016). Machine helath monitoring with lstm networks. In *2016 10th International Conference on Sensing Technology (ICST).* IEEE.

Zhao, J. K. G. (1996). Classifying Facial Emotions by Backpropagation Neural Networks with Fuzzy Inputs. *Proceedings of Conference on Neural Information Processing*, *1*, 454–457.

Zhao, L., Chen, J., Chen, F., Wang, W., Lu, C.-T., & Ramakrishnan, N. (2015). Simnest: Social media nested epidemic simulation via online semisupervised deep learning. *Proceedings - IEEE International Conference on Data Mining*, 639–648. doi:10.1109/ICDM.2015.39

Zhen, X., Wang, Z., Islam, A., Bhaduri, M., Chan, I., & Li, S. (2016). Multi-scale deep networks and regression forests for direct bi-ventricular volume estimation. *Medical Image Analysis*, *30*, 120–129. doi:10.1016/j.media.2015.07.003

Zhonga, L., Hub, L., & Zhouc, H. (2019). DL based multi-temporal crop classification. *Remote Sensing of Environment*, *221*, 430–443. doi:10.1016/j.rse.2018.11.032

Zhong, Y., Gao, J., Lei, Q., & Yao Zhou, A. (2018). Vision-Based Counting and Recognition System for Flying Insects in Intelligent Agriculture. *Sensors (Basel)*, *18*(5), 1489. doi:10.339018051489 PMID:29747429

Zhu, L.-Q., Ma, M.-Y., Zhang, Z., Zhang, P.-Y., Wu, W., Wang, D.-D., ... Wang, H.-Y. (2017). Hybrid DL for automated lepidopteran insect image classification. *Oriental Insects*, *51*(2), 79–91. doi:10.1080/00305316.2016.1252805

Zhu, N. Y., Liu, X., Liu, Z. Q., Hu, K., Wang, Y. K., Tan, J. L., ... Guo, Y. (2018). Deep learning for smart agriculture: Concepts, tools, applications, and opportunities. *International Journal of Agricultural and Biological Engineering*, *11*(4), 32–44. doi:10.25165/j.ijabe.20181104.4475

Zin. (2018). Image Technology based Cow Identification System Using DL. *Proceedings of the International MultiConference of Engineers and Computer Scientists 2018*, 1.

Zipern, A. (2001). A Quick Way to Search For Images on the Web. *The New York Times*.

Zou, B., Lampos, V., Gorton, R., & Cox, I. J. (2016). On infectious intestinal disease surveillance using social media content. *Proc.6th Int. Conf. Digit. Health Conf.*, 157–161. 10.1145/2896338.2896372

About the Contributors

Karthikrajan Senthilnathan, Research Advisor - EV Charger, Product development team, Revolt. He completed his Ph.D. from the Vellore Institute of Technology. His research interest includes Power systems, back to back converters in power systems, cyber-physical systems, smart grids, EV Chargers. He holds two patents, authored and edited four books from AAP Press - Taylor and Francis, IGI global - USA, Wiley publishers, published 18 Journals and conference papers. Serving as editor in SAGE journal, IGI Global - USA and Bentham ebooks. Currently appointed as Bentham Publisher Brand Ambassador for India. Serving as working group member for IEEE Standards P2004, P2660.1, P1451.7.

Balamurugan Shanmugam is with Department of IT, Kalaignar Karunanidhi Institute of Technology, Coimbatore. He holds to his credit 100+ Publications in various International Journals and IEEE/ Elsevier International Conferences. He has authored a chapter in an International Book "Information Processing" published by I.K. International Publishing . He has published 4 international books titled "Principles of Social Network Data Security", "Principles of Scheduling in Cloud Computing", "Principles of Database Security", and "Principles of Security in Cloud Computing. He is a State Rank holder in 10th grade. He was Proficiency Award Winner in 11th grade and School First Rank Holder in 12th grade . He was University Rank holder M.Tech. Examinations, Anna University. He served as a Joint Secretary of IT Association, Department of IT, PSG College of Technology, India. He is the recipient of gold medal for best journal publication and best book publication by his host institution consecutively for 7 years. Some of his professional activities include editorial board member for 47 International Journals He has guided 40+ B.Tech projects and 10 M.Tech. projects. He is a life member of ISTE, CSI.

Dinesh Goyal is Professor of Cloud Computing and Security and Academic Dean of Suresh Gyan Vihar University, holding Ph.D. degree from Suresh Gyan Vihar University in 2014, also has 16 years of rich experience in teaching with 7 years in pure R&D sector of Image Processing, cloud computing/ information security and over 90 research papers in international publications followed by 21 papers presented in conferences, out of above 8 papers are SCI/Scopus/Web of Sciences/ Thomson Reuter Indexed Journals. Also, in order to have inquisitiveness for the current trends in technology, I have conducted and organized 4 conferences and 5 workshops followed by 5 National paper publications. In addition to this, I have strived hard to guide mentor over 35 research scholars during my career followed by 8 doctoral guidance with 5 as supervisor and 3 as Co-supervisor.

Iyswarya Annapoorani is an associate professor at the School of Electrical Engineering, Vellore Institute of Technology, Chennai, India. She completed a PhD on High voltage engineering. Her research led to a number of academic publications and presentations. Her main research interests include high voltage, back to back converters, smart grids, renewable energy system modelling and power system stability and control.

Ravi Samikannu is currently working as an Associate Professor, Electrical Engineering Department in Botswana International University of Science and Technology, Palapye, Botswana. Ravi Samikannu obtained his Ph.D. Degree in Electrical Engineering, from Anna University, Chennai, India. He has a total teaching experience of 14 years at under graduate and post graduate levels. At present 5 Ph.D and 4 Master students are doing research under his supervision. Three Ph.D students and 17 Master students completed their research work under his supervision. He has published 60 research papers in International Journals. He has presented 40 Papers in International and National Conferences, and has received the Best Paper Award two times for his presentation.

* * *

K. Bhargavi received her bachelors and masters degree in computer science and engineering from Visveswaraya Technological University (VTU). She is currently pursuing Ph.D. under VTU; her research interest includes application of cognitive agents in healthcare, developing context aware smart applications, converting SQL queries to XML, high performance computing, swarm intelligence, and machine learning.

Sangeetha D. is currently working as Assistant Professor in the Department of Information Technology, MIT Campus, Anna University. She received her B.E and M. E degree in Computer Science from University of Madras and Anna University respectively. She received her Ph. D in Information and Communication Engineering. Her research interests are Information Security, cloud computing and Data analytics. At present, she is guiding two Ph. D scholars in the area of machine learning and cloud security.

Chitra A. Dhawale is currently working as a Professor and Head, P. G. Department of Computer Application, P. R. Pote College of Engineering and Management, Amravati. Earlier she was associated with Symbiosis International University, Pune. To her credit so far 5 research scholars have been awarded Ph.D. under her guidance by S.G.B. Amravati and R.T.M.Nagpur University. Her research interest includes Image and Video Processing, Multi-Biometric, Cloud Security. She has published more than 60 research papers in peer reviewed International Journals and in IEEE, Springer, Elsevier, ACM conferences proceedings and 04 book chapters in IGI Global Publications. Dr. Chitra, One of the well known academician in Computer discipline has organized various National and International Conferences, Chaired Sessions and delivered invited talk in various Faculty Development Programs, Orientation, Refresher courses, National and International Conferences. She is a senior member of the IACSIT, Member of International Association of Engineers (IAENG), Hong Kong and Life Member of ISTE, New Delhi. She has been reviewing books from various prestigious publication and also reviewer for International journals. Dr. Chitra is also a motivational speaker and conducted several workshops on “Effective Research Methodology”, “Research Writing Skills”, “Placement Preparation”, “100 techniques for speed mathematics”, “Interview Techniques”, “Spiritual thoughts” etc. As a part of social contribution, she is executive member of NGO “AADHAR Foundation”, Amravati.

Kritika Dhawale is a student of B.Tech (ECE) of, Indian Institute of Information Technology, Nagpur, She is working in deep learning from one year.

Rakshana Gopalakrishnan graduated from Anna University - Madras Institute of Technology campus in April 2019 with a bachelor's of Technology in Information Technology. She holds a keen interest in Information Security, Operating Systems and Deep Learning.

R. Kamatchi is a Professor, Head, ISME, Mumbai. She completed her Ph.D with Mother Teresa Women's University, Kodaikanal under the guidance of Dr. Atanu Rakshit, Director, IIM, Rohtak. She has 16 years of teaching experience with various premier institutes in Mumbai. She has co-authored 5 books and presented more than 35 papers in National and International conferences. She has conducted various sessions on the topics of Service modeling, CRM, Internet Security, Current IT trends in various students and faculty forums. She was appointed as a Syllabus Revision Committee member with Mumbai University and developed the course content for B.Sc., and M.Sc. (Information Technology) courses. She is IRCA certified Lead auditor. She authored a course material for the subjects of CRM, Internet Security, Data communication for the M.C.A., Distance education Programme with Mumbai University. She is the approved guide for M.Phil. and Ph.D., programmes with Madurai Kamaraj, Barathiyar & Amity University. 7 students completed their M.Phil., dissertation under her guidance and Seven students are pursuing their doctoral research . She is in the editorial board of various peer reviewed journals and books. She is also an invited member of various Science & Technology forums.

Asha Karegowda is currently working as Associate Professor, Dept of MCA, Siddaganga Institute of Technology form last 20 years. Have authored few books on C and Data structures using C, currently writing books on Python, Data mining and Data structures using C++. Area of interest include Data mining, WSN, Remote sensing, Bio inspired computing. Have published papers both in international conferences and journals. Guiding 3 research scholars in the area of image processing and remote sensing. Handling subjects for both PG and UG students: Python, Data structures C and C++, Data mining, Data Analytics.

S. B. Kishor is a double Ph.D holder, Ph.D in Computer Science and Ph.D in Management. working as a Chairman of Board of Studies Computer Science, Gondawana University, Gachchiroli since 16 March 2012. He has authore55 books (89 with Revised Edition) in Computer and Management discipline including 9 books published by International Publication. Apart from receiving many awards, Recipients of "Ideal Teacher Award - 2012" and "Best Writer Award - 2013" of RTM Nagpur University, Nagpur and holder of "India Book of Records - 2014" for writing Maximum Books in Computer Science and Recipients of "Ideal Teacher Award - 2016" Gondwana University, Gadchiroli and "Distinguish HOD-2017" by CSI.

Srangam Kodati is an Assistant Professor, Brilliant institute of Engineering & Technology at Hyderabad, Telangana, India. His research interests include Data mining, Bioinformatics. He had much teaching and research experience with a good number of publications in reputed International Journals & Conferences. He awarded Ph.D at Sri Satya Sai University of Technology & Medical Sciences, Sehore, Bhopal, M.P, India. M.Tech completed at JNTU-CEH (Autonomous), Kukatpally, Hyderabad. B.Tech completed VNR VJIET, Hyderabad.

Seetharaman Krishnamoorthy received the M.Sc. degree in Statistics, and the Ph.D. in Computer Science and Engineering from Annamalai University, Tamil Nadu, India, in 1990 and 2006, respectively, and the M.S. degree in Software Systems from BITS, Pilani, India, in 1995; and M.Tech. degree in Computer and Information Technology from Manonmaniam Sundaranar University, Tamil Nadu, India, in 2013. Currently, he is working as Associate Professor in the Department of Computer and Information Science, Annamalai University. His research interests include Statistical Pattern Recognition, Image and Video Retrieval, Digital Image Analysis, Data Mining and Knowledge Discovery, and Big Data Analytics. He is actively involved in professional activities, viz. reviewer in the journals: IET Image Processing, Pattern Recognition (Elsevier), Swarm and Evolutionary Computation (Elsevier), Knowledge–Based SYSTEMS (Elsevier), Computer and Electronics in Agriculture(Elsevier). He served as an Associate Editor of the Aloy Journal of Soft Computing and Applications, which is an International Journal, and Editorial Board Member of the International Journal of Information Technology and Management Sciences. He is also member of the professional societies -- IEEE, ACM, IAENG, and Indian Science Congress Association. He has received R & D Major Research Project funds worth more than Fifty Lakhs from funding agencies, University Grants Commission (UGC), Council of Scientific and Industrial Research (CSIR) in India. He has authored the book entitled Image Processing: Stochastic Model Based Approach; published more than 75 articles in journals of the Elsevier, Springer, Taylor and Francis, IEEE, etc. He has successfully supervised 11 Ph.D. scholars and 7 M.Phil. scholars. Dr. Seetharaman has received Outstanding Scientist award for the contribution and achievement in the field of Image Processing, awarded by the Centre for Advanced Research and Design, Venus International Foundation, India, in November 2017.

Manoj Prabhakaran Kumar received a B.E. degree in Electronics and Communication Engineering from Anna University, Coimbatore, India, in 2011. M.Tech in Embedded Systems and Technology from SRM University, Chennai, India, in 2013 and pursuing a Ph.D. degree in School Electronics Engineering from VIT University, Chennai Campus, India. His current research interests include Human-Computer Interaction, Affective Computing, and Human Behaviors analysis, Machine learning and Image Processing. He has published nearly 4 papers in peer-reviewed international journals and conferences

Geetha M. completed MCA from Gulbarga University in the year 2001. Since 2006 working as Assistant Professor in the Dept. of Master of Computer Applications, Bapuji Institute of Engineering and Technology (BIET), Davangere, Karnataka. She holds life membership of ISTE, New Delhi. Her areas of interest include Data Mining, Java Programming, Operations Research, Design and Analysis of Algorithms, Data Structures. Pursuing Ph.D in the area of Remote Sensing and Image processing from Visvesvaraya Technological University, Belgaum, Karnataka. Published a research review paper in IJARCS and LULC in IJRTE.

Parameswari M. completed M.E Computer Science and Engineering in Francis Xavier Engineering College in the Year of 2019.

Preethi P. is Assistant Professor & Research Coordinator in the Department of Computer Science and Engineering, Kongunadu College of Engineering and Technology, Trichy, Tami Nadu, India. She received B.Tech., degree from Roever Engineering College, Perambalur in 2012. She was awarded with M.E., from Srinivasan Engineering College, Perambalur in 2014. She has 6 years of teaching experience and pursuing Ph.D., as part-time research scholar in Anna University, Chennai. Her area of interest lies in Cloud Computing, Network Security and published 11 papers in international journals and presented 13 papers in national and international conferences in that area.

Asokan R. is Professor/ECE & Principal, Kongunadu College of Engineering and Technology, Trichy, Tami Nadu, India. He has more than 32 years of teaching experience and 20 years of research experience. His area of specialization includes Wireless Networks, Image Processing. He has produced 10 Ph.D., scholars. He has published more than 70 papers in Journals and presented more than 85 papers in National and International Conferences . He has also published 1 book and obtained 3 patents. He has obtained funding from various funding agencies like DST, AICTE, DRDO etc. He is an Associate Editor in the Journal of Selected Areas in Telecommunications (JSAT)-Cyber journals. He is an Editorial Board Member in various International Journals.He has awarded 3 national awards. He is an active member of diverse professional bodies like IETE, ISTE, ACS, CSI, etc.

Manoj Kumar Rajagopal received his B.E degree in 2003 from the University of Madras and his Ph.D. degree in 2012 from Telecom Sud Paris, France. Currently, he is working as Associate Professor in VIT University, Chennai Campus, India. His research interest includes Computer Vision, Machine Learning and Image processing, Affective Computing. He has published nearly 10 papers in peer-reviewed international journals and conferences.

Umamaheswari S. is currently working as a Assistant Professor in Department of Information Technology, Anna University - Madras Institute of Technology campus. She received a M.E. in Applied Electronics, Ph.D in Parallel Architecture, VLSI from Anna University, Chennai. Her area of specialization includes VLSI & Embedded Systems, Parallel architectures, Computer Networks, image processing and Deep Learning. At present, she is guiding five Ph.D scholars.

Jeeva Selvaraj received his doctoral degree in Computer Science and Engineering from Vellore Institute of Technology University, Chennai, India in November 2018. He received his Master and Bachelor degrees in Computer Science and Engineering from Anna University, India. He has served in academics sector for more than four years. He has published research papers in various reputed peer reviewed journals. His areas of interest are image processing, video analytics, video processing and computer vision.

Singaravelan Shanmugasundaram received the B.E, degree in Electronics and Communication Engineering from Francis Xavier Engineering College in 2004. He received the M.E., and Ph.D., degrees in Computer Science and Engineering from Manonmaniam Sundaranar University, Tirunelveli in the year 2007 and 2016 respectively. Currently he is a Professor in Computer Science and Engineering Department of Computer Science and Engineering in P.S.R. Engineering College, Sivakasi, Tamil Nadu, India. He has authored / coauthored over 30 research articles in various Journals and Conferences in the areas of Image Processing, Data Mining and Softcomputing.

E. M. Vignesh was born in Erode, India on September 16, 1996. He received the B.Tech degree in information technology from Anna University, Chennai, India in 2018. He is currently working at Zoho Corporation, Chennai.

Deepali R. Vora is a Professor in Vidyalankar Institute of Technology, Mumbai and has completed her PhD in Computer Science and Engineering from Amity University, Mumbai. She has more than 18 years of experience in teaching and has published more than 40 research papers in reputed national, international conferences and journals. She has co-authored two books and delivered various talks in Data Science and Machine learning. She has conducted hands-on session in Data Science using Python for students and faculties. She was appointed as a Syllabus Revision Committee member with Mumbai University and developed the course content for B.E. (Information Technology) course. She has received grants for conducting research and organising training courses for faculties. She is acting as a reviewer for many International Conferences and Journals. She has organised many value added courses for the benefit of the students. More than 10 students have completed and currently, 4 students are pursuing their post-graduate studies under her guidance.

Index

H

I

K

L

M

N

O

P

R

S

T

U

V

Z