

Springer Proceedings in Mathematics & Statistics

Rajiv Misra · Rana Omer ·
Muttukrishnan Rajarajan ·
Bharadwaj Veeravalli ·
Nishtha Kesswani ·
Priyanka Mishra *Editors*

Machine Learning and Big Data Analytics

2nd International Conference
on Machine Learning and Big Data
Analytics-ICMLBDA, IIT Patna, India,
March 2022

 Springer

**Springer Proceedings in Mathematics &
Statistics**

Volume 401

This book series features volumes composed of selected contributions from workshops and conferences in all areas of current research in mathematics and statistics, including data science, operations research and optimization. In addition to an overall evaluation of the interest, scientific quality, and timeliness of each proposal at the hands of the publisher, individual contributions are all refereed to the high quality standards of leading journals in the field. Thus, this series provides the research community with well-edited, authoritative reports on developments in the most exciting areas of mathematical and statistical research today.

Rajiv Misra • Rana Omer •
Muttukrishnan Rajarajan • Bharadwaj Veeravalli •
Nishtha Kesswani • Priyanka Mishra
Editors

Machine Learning and Big Data Analytics

2nd International Conference on Machine
Learning and Big Data Analytics-ICMLBDA,
IIT Patna, India, March 2022

 Springer

Editors


Rajiv Misra
Dept. of Computer Science & Engineering
Indian Institute of Technology Patna
Patna, Bihar, India

Rana Omer
Cardiff University
Cardiff, UK

Muttukrishnan Rajarajan
Dept. of EE Engineering
University of London
London, UK

Bharadwaj Veeravalli
Dept. of ECE
National University of Singapore
Singapore, Singapore

Nishtha Kesswani
Dept. of Computer Science
Central University of Rajasthan
Tehsil Kishangarh, Rajasthan, India

Priyanka Mishra 
Dept. of CSE
Indian Institute of Information Technology,
Kota
Jawahar Lal Nehru Marg, Rajasthan, India

ISSN 2194-1009 ISSN 2194-1017 (electronic)
Springer Proceedings in Mathematics & Statistics
ISBN 978-3-031-15174-3 ISBN 978-3-031-15175-0 (eBook)
<https://doi.org/10.1007/978-3-031-15175-0>

Mathematics Subject Classification: 68-06, 68Txx

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

It is our privilege and pleasure to welcome you to the 2nd International Conference on Machine Learning and Big Data Analytics (ICMLBDA) 2022 held on March 23–24, 2022, at IIT Patna, India.

As an established international event for the machine learning and big data community, our conference showcases research on fundamentals and applications of big data analytics and machine learning. The submissions underwent a rigorous peer review process by the strong Program Committee that included experts from all over the world.

We report 30% acceptance rate for regular papers with additional 30% of submissions that were accepted as short articles.

The main conference included over four technical sections that focused on fundamentals of AI and data science applications in mechanical engineering, big data in biology, Internet of Things, networks and security, and artificial intelligence and machine learning.

Importantly, this conference basically focused on advanced automation and computational optimization of machine learning in all engineering-based applications as well as included specific plenary sessions, invited talks, and paper presentations focusing on the applications of ML and BDA in the fields of computer, electronics, electrical, mechanical, chemical, and textile engineering; healthcare and agriculture; business and social media; and other relevant domains.

We would like to acknowledge the many individuals and organizations that have made our conference possible. The hard work and support of the members of the Organizing Committee allowed us to deliver a successful conference program on time.

We are very grateful to the members of the Program Committee who tirelessly and timely reviewed submissions from a broad range of machine learning topics and many application areas.

We also thank all the authors who decided to submit fruits of their labor to ICMLBDA 2022. We very much appreciate your support and your trust in

our handling of your manuscripts. We hope that you will consider submitting to ICMLBDA again next year. Last but not least, our thanks go to the publisher Springer PROMS and the International Association of Academicians (IAASSE) for their generous support.

Patna, Bihar, India

Cardiff, UK

London, UK

Singapore, Singapore

Tehsil Kishangarh, Rajasthan, India

Jawahar Lal Nehru Marg, Rajasthan, India

Rajiv Misra

Rana Omer

Muttukrishnan Rajarajan

Bharadwaj Veeravalli

Nishtha Kesswani

Priyaanka Mishra

Contents

A Comprehensive Analysis on Mobile Edge Computing: Joint Offloading and Resource Allocation Perspective	1
Antharaju K. Chakravarthy, T. Rama Reddy, and N. Ramakrishnaiah	
Finding Significant Project Issues with Machine Learning	13
Narasimha Rao Vajjhala and Kenneth David Strang	
Prediction of Heart Disease Using Various Data Analysis and Machine Learning Techniques	23
Anjali Singh, Deepanshi Vij, Alpana Jijja, and Sherry Verma	
Artificial Intelligence: Recent Trends, Opportunities, and Challenges in Real-World Scenarios	37
Bosubabu Sambana, Yagireddi Ramesh, Satyabrata Patro, N. P. Patnaik M, P. J. V. Prakasa Rao, Adimalla Rama Rao, and Priyanka Mishra	
Bilingual Documents Text Lines Extraction Using Conditional GANs	49
Sukhandeep Kaur, Seema Bawa, Ravinder Kumar, and Munish Kumar	
Performance Comparison of YOLO Variants for Object Detection in Drone-Based Imagery	59
Manpreet Kaur and Padmavati Khandnor	
A Microservice Architecture with Load Balancing Mechanism in Cloud Environment	75
Satyantarayana Mummana, Bosubabu Sambana, Budimure Ramanababu, Preethi Gandreti, P. Pratima Rani, Priyanka Mishra, A. Chandrasekhar, and K. Narasimha Raju	
An IoT Application for Detection and Monitoring of Manhole	93
Sai Pavan Tangella, Krishna Pavan Inala, Phani Gourinath Lella, and Syed Arshad	

Resource Allocation in 5G and Beyond Edge-Slice Networking Using Deep Reinforcement Learning	105
Rohit Kumar Gupta, Praduman Pannu, and Rajiv Misra	
The Important Influencing Factors in Machine Translation	119
Debajyoty Banik	
Damaged Units Return Investigation in Printer-Producing Industry Utilizing Big Data	127
Gali Nageswara Rao, Yagireddi Ramesh, Bondu Venkateswarlu, and Bosubabu Sambana	
Colorization of Grayscale Images: An Overview	141
Medha Wyawahare, Tejas Kolhe, Akshay Joshi, Yogesh Metkari, and Kshitij Patil	
Evolutionary Approaches Toward Traditional to Deep Learning-Based Chatbot	159
Arpan Maity, Srijita Guha Roy, and Debajyoty Banik	
Analysis of Machine Learning Algorithms for Detection of Cyberbullying on Social Networks	171
K. V. Soujanya and Omprakash Tembhurne	
Sentiment Analysis of Political Tweets for Israel Using Machine Learning	191
Amisha Gangwar and Tanvi Mehta	
A Novel Approach for Real-Time Vehicle Re-identification Using Content-Based Image Retrieval with Relevance Feedback	203
N. Shankaranarayan and S. Sowmya Kamath	
Extractive and Abstractive Text Summarization Model Fine-Tuned Based on BERTSUM and Bio-BERT on COVID-19 Open Research Articles	213
Jhansi Lakshmi Durga Nunna, V. K. Hanuman Turaga, and Srilatha Chebrolu	
RevCode for NLP in Indian Languages	225
Ashis Samal, Akash Sambhangi, and Charan Singh	
Application for Mood Detection of Students Using TensorFlow and Electron JS	235
Marada Srinivasa Rao, Pasala Sandhya, Bosubabu Sambana, and Priyanka Mishra	
Using CNN Technique and Webcam to Identify Face Mask Violation	245
Bodduru Keerthana, Tata Rao Vana, Marada Srinivasa Rao, Bosubabu Sambana, and Priyanka Mishra	

A Review on Internet of Things-Based Cloud Architecture and Its Application 255
 D. Dakshayani Himabindu, Keesara Sravanthi, Raswitha Bandi, and Bharathi Panduri

Prediction of Maneuvering Status for Aerial Vehicles Using Supervised Learning Methods 269
 Abhishek Gupta, Sarvesh R. Thustu, Riti R. Thakor, Saniya A. Patil, Raunak Joshi, and Ronald Melvin Laban

HRescue: A Modern ML Approach for Employee Attrition Prediction ... 279
 Rudresh Veerhare, Parshwa Shah, Jiten Sidhpura, and Sudhir Dhage

Using Machine Learning to Detect Botnets in Network Traffic..... 295
 Shambhavi Rai, S. A. Sajidha, V. M. Nisha, and B. Mahalakshmi

Reducing Peak Electricity Demands of a Cluster of Buildings with Multi-Agent Reinforcement Learning 307
 Manoj Kumar Balwant, Sai Rohan Basa, and Rajiv Misra

Virus Texture Classification Using Genetic Algorithm and Pre-trained Convolutional Neural Networks 319
 Chandra Mohan Bhuma

Fog Computing-Enabled Internet of Things for Resource Optimization .. 329
 Meenaxi M. Raikar and Meena S M

Network Media Content Model in the Era of Smart Devices 341
 Adapa Venkateswara Rao, Molli Srinivasa Rao, and J. Durga Prasad Rao

Sentimental Analysis of Stock Market via Twitter 355
 S. V. S. S. Lakshmi, T. Vineetha Sai Durga, N. V. L. Tejaswi, and A. Yeshwanth Sai Kumar

Detection and Classification of Tumor Tissues in Colorectal Cancer Using Pathology Images..... 365
 Ponnarasee B. K and Lalithamani N

Challenges Encountered in the Implementation of Machine Learning in the Healthcare Industry..... 377
 Rita Roy, Subhodeep Mukherjee, Manish Mohan Baral, Ajay Kumar Badhan, and Marada Ravindra

Performance Evaluation of Deep Learning Architectures for Recognition of Moisture in Dried Coconut Copra..... 387
 K. Padma Vasavi, G. Srinivasa Rao, S. Hanumantha Rao, M. Prema Kumar, and P. Ravi Kumar

Training Generative Adversarial Networks (GANs) Over Parameter Server and Worker Node Architecture	401
Amit Ranjan and Rajiv Misra	
Handwritten Digit Recognition Using Neural Network with Gabor Filter for Information Fusion	411
Akshay Kumar and Brindha Murugan	
FAFOC: Fog-Based Energy-Efficient Clustering Technique for Wireless Sensor Networks	423
R. Dayana and G. Maria Kalavathy	
Evaluation of Supervised Classifiers for Fake News Detection Using Twitter Dataset	435
Vinita Nair and Jyoti Pareek	
Analysis of Pest Recognition Using Lightweight CNN	447
C. Nandhini and M. Brindha	
Early Prediction of Alzheimer’s Disease Using Ensemble Learning Models	459
Divjot Singh and Ashutosh Mishra	
Cattle Identification from Muzzle Print Image Pattern Using Hybrid Feature Descriptors and SVM	479
Amanpreet Kaur, Munish Kumar, and M. K. Jindal	
Evaluation and Detection of Breast Cancer Using Data Mining Models ..	491
B. S. Panda, Satyabrata Patro, and Jyotirmaya Mishra	
Lung Cancer Disease Prediction Using Machine Learning Techniques ...	501
Selvani Deepthi Kavila, S. V. S. S. S. Lakshmi, Rajesh Bandaru, Shaik Riyaz, and Neelamsetty Sai Venkata Rushikesh	
Video to Text Generation Using Sentence Vector and Skip Connections ..	515
Hanumant Mule and Dinesh Naik	
Machine Learning Techniques for Covid-19 Pandemic Updates for Analysis, Visualization, and Prediction System	529
D. Radha, P. Ratna Kumari, and M. Dhanalakshmi	
Design and CFD Analysis of Drone Thrust with Duct	545
K. S. S. Gurudatta, V. Harikiran, M. V. D. K. Raju, B. Rama Krishna, and I. Jagadeesh	
Index	563

A Comprehensive Analysis on Mobile Edge Computing: Joint Offloading and Resource Allocation Perspective



Antharaju K. Chakravarthy, T. Rama Reddy, and N. Ramakrishnaiah

Keywords Mobile edge computing · Offloading · Resource allocation · Internet of Things · Intelligent edge

1 Introduction

From the last decade, usage of smart devices and cellular networks increased exponentially, and due to the enormous growth of sensors, smart device and mobile device usage is a clear indication of the mobile IoT trend, which brings data, processes and people together to make human life more comfortable and valuable. According to CISCO report [1], it is estimated that the number of connected devices to IP networks will be more than three times the global population by 2023. The share of machine-to-machine connections will grow from 33% in 2018 to 50% in 2023. It is estimated that around the world there will be 14.7 billion machine-to-machine connections by 2023. This gives us a new sign for a wide number of business opportunities and challenges to the technical world. The sensor's prime

A. K. Chakravarthy (✉)

Department of Computer Science and Engineering, Jawaharlal Nehru Technological University Kakinada, Kakinada, Andhra Pradesh, India

Department of Information Technology, Aditya Engineering College (A), Surampalem, Andhra Pradesh, India

Jawaharlal Nehru Technological University Kakinada, Kakinada, Andhra Pradesh, India

T. Rama Reddy

Jawaharlal Nehru Technological University Kakinada, Kakinada, Andhra Pradesh, India

Department of Computer Science and Engineering, Aditya Engineering College (A), Surampalem, Andhra Pradesh, India

N. Ramakrishnaiah

Department of Computer Science and Engineering, Jawaharlal Nehru Technological University Kakinada, Kakinada, Andhra Pradesh, India

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

R. Misra et al. (eds.), *Machine Learning and Big Data Analytics*,

Springer Proceedings in Mathematics & Statistics 401,

https://doi.org/10.1007/978-3-031-15175-0_1

objective is to generate data in the IoT environment, and processing that data is a challenging operation at cloud data centre [2]. Massive growth of the data from those devices leads to computational and communicational bottleneck in device level and network level, respectively. Applications like autonomous vehicles and unmanned aerial vehicles need very low delay sensitivity. All these applications are emerged with lower delay and higher computing efficiency.

In the recent years there is a wide range of approaches available to fulfil the above requirement such as cloud computing (CC), mobile cloud computing (MCC), cloudlets and mobile edge computing (MEC). The conventional cloud computing solves the problem partially. In the cloud environment data needs to be processed at the centralized server, which is suffering with the delay-sensitive applications [3]. Even though there are enormous studies on the cloud computing, still a research gap exists at this juncture. MCC came into the picture to come up with a virtualization concept, but that suffers with the offloading problem. MEC is the promising technology to address the above research gaps. Computing at the network's edge is referred to as MEC. MEC applications run in a radio access network (RAN) environment [3–9]. MEC is defined by ETSI as any network site where computation, other resources and services are available closer to the user than in a central data centre or cloud. Certain base stations, such as cell towers, data centres, Wi-Fi routers and hotspots, serve as the network's edge. The edge's proximity decreases latency to milliseconds and ensures that users have a steady connection. The below prominent technologies are significant at delay-sensitive application.

1.1 Various Technologies

I. Cloud Computing

Cloud computing technology enables to run computations in a centralized data centre. In the recent years cloud is the most desirable technology which addresses most of the problems. Domestic users and industrial users are benefited from the cloud properties such as its availability and ubiquitous nature. Cloud suffers from delay and resource allocation issue in emerging IoT environment, which leads to a scope for the MEC technology to come in front.

II. Cloudlets

Cloudlets are well known for virtualization concept. The fundamental concept in cloudlets is bringing the cloud services to the edge network level by virtualizing the cloud data centre. Cloudlets are efficient at cloud applications by bringing the cloud services closer to the user. Furthermore, cloudlets are a suboptimal cloud solution since advanced applications such as autonomous vehicles and unmanned vehicles suffer from latency concerns.

III. Mobile Cloud Computing

Cloud computing and mobile computing are combined in MCC. The concept behind this technology is to create a virtualized isolated environment in the cloud with various resources such as computing, storage and communication that end users can access remotely. The MCC approach enables the app developer to build applications designed especially for mobile user, which can be used without being bound to the device's operating system or the storage capacity.

IV. Mobile Edge Computing (MEC)

Most of the cloud stakeholders are very much concern on the following aspects such as network security, emergency efficiency, network manageability and bandwidth utilization. MEC addresses most of the above challenges within the radio access networks. The basic idea behind MEC is to bring the resources to RAN. This technique improves the communication and computational capabilities for delay-sensitive applications.

1.2 Literature Study

MEC has emerged as a phenomenal technology in most of the delay-sensitive applications. There are numerous studies found in this field [10–26], and few of them are outlined in Table 1.

Tang et al. [10] worked on fog computing-based IoT environment; they tried to address decentralized computation offloading. Fog computing environment can provide higher quality of experience (QoE) and battery life by introducing energy harvesting [EH] technique. IoT devices make decisions in the suggested environment by using a decentralized partially observable Markov decision process (Dec-POMDP). As this Dec-POMDP is a complex process, a learning-based algorithm is proposed to simplify this process. This process needs an optimization algorithm to increase the performance. The optimal solution for the stated problem is found using a Lagrangian approach and the policy gradient method (Table 1). It is observed that remaining pending tasks, battery life time and available fog resources are the factors which affect offloading technique.

In the MEC context, the authors have shed light on effective and efficient job assignment and computation offloading methodologies [11]. With the rapid development of mobile devices in recent years, multimedia services such as multimedia content delivery and video calling are the most significant areas that need to be addressed in wireless networks. In this paper the task offloading problem is divided in to several subproblems by using Lyapunov optimization technique, Stochastic optimization algorithms provide an alternative approach that permits less optimal local decisions to be made within the search procedure that may increase the probability of the procedure locating the global optima of the objective function (Table 1).

Zahed et al. [12] focused on security breaches in MEC environment. There is a scope of malicious attacks and eavesdropping in multiuser environment especially when they offload the task. Therefore assignment of security services in task offloading is essential. This problem is solved by designing a resource-constrained optimization model and jointly optimizing energy consumption and security breach cost while maintaining delay condition. In the case of large number of servers involved in the computation delay calculation system, complexity will increase, and to solve this bottleneck, a two-stage heuristic solution is implemented to overcome this limitation and achieve an acceptable solution within a reasonable time limit. According to numerical results 60% security breach cost is reduced and 15% of energy consumption is reduced by comparing the current MEC technique. Mobility and incentive mechanisms are proposed as a future work for this study.

Li et al. [13] concentrated offloading problem in MEC by introducing MNL P optimization technique. Mixed-integer nonlinear programming (MNL P) addresses a very general class of optimization problems with nonlinearities in the objective and/or constraints as well as continuous and integer variables. Channel allocation, transmission power and computational resource allocation problems are also addressed by subdividing original problem in to two subproblems. Intelligent algorithms and quasi-convex and convex optimization technology are used as key algorithms to address the above subproblems (Table 1).

Guo et al. [14] studied on computational offloading strategy in single-user multiple antennas where data transmission applications come into the picture. Problem statement is figured in MEC environment and multiple antennas are considered. In this study they proposed intelligent offloading strategy for MEC networks assisted by array signal processing. Array signal processing is centred on the ability to fuse temporal and spatial information captured via sampling signals emitted from a number of sources at the sensors of an array in order to carry out a specific estimation task. Two kinds of array signal processing are employed, i.e. maximum ratio transmission (MRT) and selection combining (SC).

Chen et al. [15] worked on offloading technique in small-cell MEC environment. In a small cell, mobile devices are attached to small base stations, and these base stations are attached to micro base station (MBS). Small cell mobile computation offloading decision depends on two factors, i.e. communication and computation resources. Small base stations collaboratively take offloading decision, and this is a complex task at the network level. First problem is formulated as a mixed binary nonlinear programming problem and then converted as general binder decomposition (GBD). A heuristic solution is proposed in order to invoke more MBS to involve in decision making.

Wang et al. [16] focused on MEC with 5G network, which gives a great breakthrough in transmission time and execution delay. In the practical scenario MEC has limited resources, and in order to have efficient utilization of the resources, we need an optimized computation offloading, resource allocation and data cache services. The proposed problem is NP-hard since it is formulated as a mixed-integer programming (MIP) problem. The original problem was broken into two subproblems: downlink resource allocation combined with offloading decision and

computation resource allocation. These two subproblems can be solved with the help of convex and non-convex optimization techniques. Convex optimization problems are more generic than linear programming problems, although they share some of the same desired characteristics: Up to a very larger frame, they can be addressed rapidly and consistently. Proposed iterative algorithm gives better results than the benchmark algorithms.

Lyu et al. [17] worked on MEC offloading techniques along with energy minimization technique. High reliability and low latency are the desirable features in IoT environment, and MEC will address the above issue by introducing a variety of offloading techniques; this may give phenomenal improvement in storage and computational limitations. In this study they proposed a three-tier architecture by incorporating MEC, cloud and IoT devices. In this study they proposed a three-tier architecture by incorporating MEC, cloud and IoT devices, a lightweight request, and admission control scalability issues. Various parameters like input size, required cpu cycles, uplink and downlink capabilities and offloading decisions vary based on these parameters.

1.3 Organization of the Paper

The paper is organized as follows. Section 2 addresses MEC architecture. Section 3 describes need of offloading. Section 4 addresses types of offloading. Section 5 focuses on problem formulation and optimization techniques. Section 6 addresses various parameters which affect offloading. Section 7 describes various simulators and Sect. 8 covers conclusion.

2 MEC Architecture

Figure 1 describes a typical MEC architecture. MEC servers are placed close to the user equipment. In this scenario, edge devices are connected to local base station or a Wi-Fi router. With the advancement of 5G technology and edge computing, it is possible to fill the communication gap between heterogeneous devices. In this case MEC servers play a vital role to coordinate edge devices. Generated data at edge level is being processed at MEC server with no time delay. This enables MEC to support delay intensity applications such as autonomous vehicles and IoT applications. This paradigm can be applicable to most of the delay effective applications. The edge devices are connected to local access point (AP) or BTS (base transceiver station). BTS is equipped with edge server to run user application. The advantage of the below architecture is to reduce the burden on the cloud by preprocessing the data at the edge server.

Table 1 Problem formulation and optimization methods

References	Objective	Problem formulation	Optimization technique	Contributions
[10]	Computation offloading	Decentralized partially observable Markov decision process (Dec-POMDP)	Lagrangian approach and the policy gradient method	To discover the approximate optimal solution, a learning-based decentralized offloading mechanism with low complexity is given
[11]	Computation offloading	Stochastic optimization	Lyapunov optimization technique	Under various settings, the algorithm may achieve reasonable energy consumption and delay performance
[12]	Computation offloading	Constrained nonlinear program (NLP) optimization problem	Two-stage heuristic algorithm	Achieved optimal solution by TCO, CMEC, STO algorithms
[13]	Computation offloading	Mixed-integer nonlinear programming problem (MINLP)	Immune algorithm (IA), quasi-convex optimization and convex optimization techniques	Achieved better results on different constraints
[15]	Computation offloading and resource allocation	Mixed binary nonlinear programming problem	Heuristic algorithm using general bender decomposition	Results in significantly reduced worst-case complexity while maintaining high energy performance across a wide range of system parameters
[16]	Computation offloading	Mixed-integer programming (MIP) one which is NP-hard	Convex and non-convex optimization techniques	Obtained better results comparatively with the benchmark algorithms
[20]	Computation offloading	Multiuser computation offloading game	Distributed computation offloading algorithm that can achieve a Nash equilibrium	This algorithm performs well when number of users increases
[21]	Computation offloading	Mixed-integer nonlinear programming problem that is NP-hard	Novel heuristic algorithm	Reduce mobile device energy consumption while meeting application completion time limitations
[23]	Computation offloading	Physical resource block (PRB) allocation	Graph colouring method	Reduce mobile device energy consumption while meeting application completion time limitations

(continued)

Table 1 (continued)

References	Objective	Problem formulation	Optimization technique	Contributions
[25]	Computation offloading	Combines local computing and data offloading	Gradient descent method	Shows various trade-offs based on different parameters
[26]	Computation offloading	Weighted sum latency-minimization problem	Convex optimization problem	It performs better than other benchmark algorithms

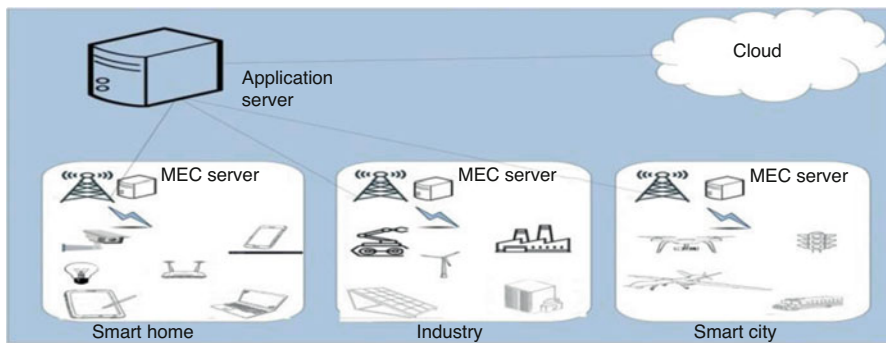


Fig. 1 Illusion of mobile edge computing architecture

3 Need of Offloading

Future world will be densities with IoT and smart devices, and this situation leads to a tremendous load on cloud with respect to computation and communication accepts. Especially delay-sensitive applications needed more attention such as autonomous cars, AR/VR applications, gaming and unmanned aerial vehicles. At this juncture it is necessary to reduce burden on cloud by introducing offloading technique at edge devices. The basic approach in offloading depends on various factors such as current network capacity, computation (size and number of pending tasks), transmission, traffic, power-delay trade-off, energy, communication, caching, number of available edge resources, computational resource allocation, channel allocation and many more [5–8]. Depending on the above factor an efficient offloading technique needs to decide where to send computational intensive task, either to nearby edge server or cloud data centre, so that result may not be effected with the delay factors. There are a wide range of approaches available to address the above issue and are discussed as below.

4 Types of Offloading

I. Centralized Offloading

Based on the below literature offloading can be classified as centralized and distributed. Most of the cloud computing applications follow centralized offloading technique by executing business logic at the centralized data centre. There are numerous kick stoppers that can be found in this approach among which network latency and security are the most desirable issues.

II. Distributed Offloading

Introducing distributed mechanism in cloud environment we can overcome the above problem. MEC is most well known for its distributed mechanism where data preprocessing takes place at the data generation point only introducing distributed environment in MEC. Offloading decision process executes at mobile edge servers which respond with no time delay due to its proximity from the edge node.

5 Problem Formulation and Optimization Techniques

Problem formulation is a key factor to find optimal solution in offloading. Different researchers adopt different techniques, but it is observed that most of the researchers are convinced that the problems are NP-hard, game-based approach and convex and non-convex problems. All the above problems are solved in a polynomial time interval. The below table helps us to summarize various problem formulation approaches to obtain optimal output with respect to offloading and resource allocation strategies.

6 Parameters

A number of parameters are observed in the study [16] which is tabulated in Table 2. Each parameter has its own significance.

Table 2 Parameters

Parameters	References	Significance
Data size	[10, 12, 13, 15]	Bandwidth utilization
CPU cycles	[10]	Task computation
Number of devices	[10–13, 15]	To calculate network complexity
Task load	[13]	Capacity of the devices
Average delay	[10]	Used for delay calculation
Average energy consumption	[10]	Energy-efficient scheduling
Number of time slots	[11, 18]	Task execution, time calculation
Available CPU cores in MEC server	[11]	Task computation
Channel bandwidth	[12]	Increase network performance

7 Tools and Simulators

The researchers have recommended a few key tools, which are summarized in Table 3.

Table 3 Tools and simulator

Theme	References	Simulator
Offloading and resource allocation	[10, 13]	MATLAB
Offloading	[12]	IBM ILOG CPLEX and MATLAB R 2016b
Offloading and resource allocation	[15]	MATLAB and CVX
Offloading and resource allocation	[19]	OpenAI Gym and Python

8 Conclusion

MEC is becoming a popular solution as the Internet of Things (IoT) and smart gadgets grow in popularity. Many researchers worldwide contributed papers on offloading, but still there exist research gaps at this juncture. In this paper we have surveyed existing literature on offloading strategies and resource allocation. Comparison of various strategies of offloading is presented. To conclude our research, we looked into the architecture of edge computing for IoT, the performance targets, job offloading schemes and accompanying edge computing countermeasures, as well as typical offloading algorithms as examples. Through this study, it is anticipated that research efforts and outputs in this exciting new field will skyrocket in the coming decades due to various smart IoT applications.

References

1. CISCO (2020) Cisco Annual Internet Report (2018–2023). White Paper. Source: <https://www.cisco.com/c/en/us/solutions/ollateral/executiveperspectives/annualinternet-report/white-paper-c11741490.pdf>.
2. Vanathi Arunachalam, Nagamalleswara Rao Nallamothu, “Load Balancing in RPL to Avoid Hotspot Problem for Improving Data Aggregation in IoT” in *International Journal of Intelligent Engineering & Systems*, vol. 14, no. 1, 2021 <https://doi.org/10.22266/ijies.2021.0228.49>.
3. Z. Zhao *et al.*, “A Novel Framework of Three-Hierarchical Offloading Optimization for MEC in Industrial IoT Networks,” in *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5424–5434, Aug. 2020, <https://doi.org/10.1109/TII.2019.2949348>.
4. J. Guo, H. Zhang, L. Yang, H. Ji and X. Li, “Decentralized Computation Offloading in Mobile Edge Computing Empowered Small-Cell Networks,” *2017 IEEE Globecom Workshops (GC Wkshps)*, 2017, pp. 1–6, <https://doi.org/10.1109/GLOCOMW.2017.8269049>.
5. Q. Pham, T. Leanh, N. H. Tran, B. J. Park and C. S. Hong, “Decentralized Computation Offloading and Resource Allocation for Mobile-Edge Computing: A Matching Game Approach,” in *IEEE Access*, vol. 6, pp. 75868–75885, 2018, <https://doi.org/10.1109/ACCESS.2018.2882800>.
6. Y. Mao, J. Zhang and K. B. Letaief, “Dynamic Computation Offloading for Mobile-Edge Computing With Energy Harvesting Devices,” in *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016, <https://doi.org/10.1109/JSAC.2016.2611964>.
7. T. X. Tran and D. Pompili, “Joint Task Offloading and Resource Allocation for Multi-Server Mobile-Edge Computing Networks,” in *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 856–868, Jan. 2019, <https://doi.org/10.1109/TVT.2018.2881191>.
8. Y. Mao, J. Zhang and K. B. Letaief, “Joint Task Offloading Scheduling and Transmit Power Allocation for Mobile-Edge Computing Systems,” *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, 2017, pp. 1–6, <https://doi.org/10.1109/WCNC.2017.7925615>.
9. Y. Mao, J. Zhang, S. H. Song and K. B. Letaief, “Stochastic Joint Radio and Computational Resource Management for Multi-User Mobile-Edge Computing Systems,” in *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5994–6009, Sept. 2017, <https://doi.org/10.1109/TWC.2017.2717986>.
10. Q. Tang, R. Xie, F. R. Yu, T. Huang and Y. Liu, “Decentralized Computation Offloading in IoT Fog Computing System With Energy Harvesting: A Dec-POMDP Approach,” in *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4898–4911, June 2020, <https://doi.org/10.1109/JIOT.2020.2971323>.
11. Y. Sun, T. Wei, H. Li, Y. Zhang and W. Wu, “Energy-Efficient Multimedia Task Assignment and Computing Offloading for Mobile Edge Computing Networks,” in *IEEE Access*, vol. 8, pp. 36702–36713, 2020, <https://doi.org/10.1109/ACCESS.2020.2973359>.
12. M. I. A. Zahed, I. Ahmad, D. Habibi and Q. V. Phung, “Green and Secure Computation Offloading for Cache-Enabled IoT Networks,” in *IEEE Access*, vol. 8, pp. 63840–63855, 2020, <https://doi.org/10.1109/ACCESS.2020.2982669>.
13. Z. Li, C. Du and S. Chen, “HIQCO: A Hierarchical Optimization Method for Computation Offloading and Resource Optimization in Multi-Cell Mobile-Edge Computing Systems,” in *IEEE Access*, vol. 8, pp. 45951–45963, 2020, <https://doi.org/10.1109/ACCESS.2020.2977988>.
14. Y. Guo *et al.*, “Intelligent Offloading Strategy Design for Relaying Mobile Edge Computing Networks,” in *IEEE Access*, vol. 8, pp. 35127–35135, 2020, <https://doi.org/10.1109/ACCESS.2020.2972106>.
15. H. Chen, D. Zhao, Q. Chen and R. Chai, “Joint Computation Offloading and Radio Resource Allocations in Small-Cell Wireless Cellular Networks,” in *IEEE Transactions on Green Communications and Networking*, vol. 4, no. 3, pp. 745–758, Sept. 2020, <https://doi.org/10.1109/TGCN.2020.2976932>.

16. K. Wang et al., “Joint Offloading and Charge Cost Minimization in Mobile Edge Computing,” in *IEEE Open Journal of the Communications Society*, vol. 1, pp. 205–216, 2020, <https://doi.org/10.1109/OJCOMS.2020.2971647>.
17. X. Lyu et al., “Selective Offloading in Mobile Edge Computing for the Green Internet of Things,” in *IEEE Network*, vol. 32, no. 1, pp. 54–60, Jan.-Feb. 2018, <https://doi.org/10.1109/MNET.2018.1700101>.
18. L. Liu, X. Qin, Z. Zhang and P. Zhang, “Joint Task Offloading and Resource Allocation for Obtaining Fresh Status Updates in Multi-Device MEC Systems,” in *IEEE Access*, vol. 8, pp. 38248–38261, 2020, <https://doi.org/10.1109/ACCESS.2020.2976048>.
19. N. Kiran, C. Pan, S. Wang and C. Yin, “Joint resource allocation and computation offloading in mobile edge computing for SDN based wireless networks,” in *Journal of Communications and Networks*, vol. 22, no. 1, pp. 1–11, Feb. 2020, <https://doi.org/10.1109/JCN.2019.000046>.
20. X. Chen, L. Jiao, W. Li and X. Fu, “Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing,” in *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, October 2016, <https://doi.org/10.1109/TNET.2015.2487344>.
21. Y. Geng, Y. Yang and G. Cao, “Energy-Efficient Computation Offloading for Multicore-Based Mobile Devices,” *IEEE INFOCOM 2018 – IEEE Conference on Computer Communications*, 2018, pp. 46–54, <https://doi.org/10.1109/INFOCOM.2018.8485875>.
22. J. Zhao, Q. Li, Y. Gong and K. Zhang, “Computation Offloading and Resource Allocation For Cloud Assisted Mobile Edge Computing in Vehicular Networks,” in *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7944–7956, Aug. 2019, <https://doi.org/10.1109/TVT.2019.2917890>.
23. C. Wang, F. R. Yu, C. Liang, Q. Chen and L. Tang, “Joint Computation Offloading and Interference Management in Wireless Cellular Networks with Mobile Edge Computing,” in *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7432–7445, Aug. 2017, <https://doi.org/10.1109/TVT.2017.2672701>.
24. S. Bi and Y. J. Zhang, “Computation Rate Maximization for Wireless Powered Mobile-Edge Computing With Binary Computation Offloading,” in *IEEE Transactions on Wireless Communications*, vol. 17, no. 6, pp. 4177–4190, June 2018, <https://doi.org/10.1109/TWC.2018.2821664>.
25. H. Sun, F. Zhou and R. Q. Hu, “Joint Offloading and Computation Energy Efficiency Maximization in a Mobile Edge Computing System,” in *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 3052–3056, March 2019, <https://doi.org/10.1109/TVT.2019.2893094>.
26. J. Ren, G. Yu, Y. Cai, Y. He and F. Qu, “Partial Offloading for Latency Minimization in Mobile-Edge Computing,” *GLOBECOM 2017 – 2017 IEEE Global Communications Conference*, 2017, pp. 1–6, <https://doi.org/10.1109/GLOCOM.2017.8254550>.

Finding Significant Project Issues with Machine Learning



Narasimha Rao Vajjhala  and Kenneth David Strang 

Keywords Project · Machine learning · Failure · Random forest · Artificial intelligence · Algorithms · Models · Big data

1 Introduction

Since 1984, the Standish Group International has maintained the Chaos database, which investigates the factors of software project success and failure. Over 120,000 IT projects from over 1000 firms have been registered in the database [1]. According to the Standish Group's Chaos study, just 29% of large-scale software projects are successful [2]. The success criteria were that these initiatives generated acceptable results and were delivered on-time and on-budget. 53% of the initiatives were severely beyond budget and schedule, and 18% of them failed to provide any outcomes [2]. Machine learning algorithms have an advantage over statistical and mathematical machine learning algorithms in that they may learn and improve based on generated predictions by calculating the best estimate that closely matches the observed data and any prior information maintained by the learner [3].

The research question (RQ) for this study was, could random forest machine learning (ML) identify the most significant issues linked with project failure using declassified data from the military industry? The next section reviews the relevant literature. That is followed by methods, results, discussion, and conclusions.

N. R. Vajjhala (✉)
American University of Nigeria, Yola, Nigeria

K. D. Strang
W3-Research, Data Analytics, St. Thomas, VI, USA
e-mail: professor@kennethstrang.com

2 Review of Literature

2.1 Overview of Machine Learning

Industry 4.0, often known as the fourth industrial revolution, is a term used by researchers to characterize industrial Internet of Things (IoTs) and cyber-physical systems [4]. The use of big data and artificial intelligence (AI) analytics powered by machine learning (ML) and deep learning (DL) algorithms applied to data in real-time is one of Industry 4.0's technology pillars. Artificial intelligence (AI) is an industry 4.0 technology that has the potential to change a variety of businesses and fields [5]. AI is described as a machine's ability to communicate with and emulate human capabilities [5]. ML is a subset of AI that includes methods for learning relationships from data without relying on good assumptions about the underlying mechanisms. When it comes to learning representations from raw data, DL differs from ML. DL is a machine learning field that uses multi-layered deep neural networks (DNNs) to model abstraction from large-scale data [6, 7]. Artificial neural networks (ANNs) were used to create the first deep learning framework in the early 1980s [6].

Machine learning is a broad and interdisciplinary science that is classified as a subfield of AI [8]. Artificial intelligence (AI) is an industry 4.0 technology that has the potential to change a wide range of industries and fields [9]. Face recognition, spam filtering, audio recognition, weather forecasting, anomaly detection, churn prediction, failure detection, and document categorization are among challenges that ML algorithms can solve [8]. Machine learning is also defined by some academics as any circumstance in which several inputs, such as data, are utilized to create a model that may be used to provide predictions and important insights [10]. Pattern recognition, statistical learning, data mining, speech recognition, computer vision, natural language processing, and intrusion detection are just a few of the disciplines where machine learning has proven successful [11].

In a variety of information technology applications, AI is gaining traction as a competitive advantage [5]. Algorithms, architectures, data formalisms, and methodological procedures are all examples of AI techniques [5]. Machine learning [9–14] is a technology that allows computers to learn from their mistakes by constructing a prediction model from a set of training data. Machine learning methods are becoming more popular due to two factors: increasing processing power and the rapidly growing amount of collected data [10]. Linear regression, decision trees, support vector machines (SVMs), Bayesian inference, random forest, and artificial neural networks (ANNs) are all popular machine learning techniques [9].

2.2 Machine Learning Techniques

Machine learning algorithms can be categorized as unsupervised learning, supervised learning, semi-supervised learning, and reinforcement learning algorithm

[15]. In supervised learning, the learner uses a series of labeled samples as training data and predicts all the unseen instances. Supervised learning algorithms must have explicitly defined inputs corresponding to known outputs by a determined target variable which is used for classification or prediction [3]. Some of the supervised learning scenarios include anomaly detection, face recognition, image classification, and weather forecasting [8]. In unsupervised learning, the learner only receives unlabeled training data and predicts unseen instances. Some of the example scenarios of unsupervised learning include clustering and reduction of dimensionality [8]. In semi-supervised learning, the learner receives the training sample that includes the labeled and unlabeled instances and predicts all the unseen instances. Some of the scenarios' where semi-supervised learning is used are the situations in which it is easy to access unlabeled data. Still, the labels are costly to obtain, for instance, in ranking processes [8]. Semi-supervised learning techniques help deduce a function grounded on fewer labeled training datasets and a considerably larger number of unlabeled datasets [16]. Ensemble learning techniques using multiple models with different modeling algorithms or training sets is another approach for modeling metabolic pathways [17].

Clustering, classification, anomaly detection, structured annotation, translation, and regression are some of the learning techniques and methodologies utilized in machine learning [8, 18]. Classification is a task that necessitates the use of machine learning algorithms to learn how to assign a class label to problem domain instances. Classifying emails as “spam” or “not spam” is an easy-to-understand example. Clustering is an unsupervised learning exercise used in exploratory data analysis to uncover previously unknown patterns in data that are difficult to categorize. The most illustrative task in the data mining process is clustering. Optimizers are strategies or approaches for reducing losses by changing the properties of your neural network, such as weights and learning rate. Clustering algorithms can divide data into numerous groups, with the user just having to identify and enter the number of clusters into the algorithm [8, 19, 20]. Classification is like clustering, except that the classifier is trained on a set of previously classified data [8, 16, 21, 22]. Linear discriminant analysis (LDA), naive Bayes (NB), k-nearest neighbors (k-NNs), artificial neural networks (ANNs), support vector machines (SVMs), and decision trees are some of the most prominent classification techniques [8].

2.3 Machine Learning Methods and Algorithms

Based on observed data, machine learning techniques enhance their performance at certain activities [23]. If a machine learning model can forecast unobserved data using training on observed data, it is deemed well-defined. To express various types of uncertainty, probabilistic models use probability theory as a framework [23]. Bayesian learning is the process of learning from data using probability theory [23]. The purpose of machine learning is to make conclusions from the training

sample; hence it uses statistical analysis to construct models. As a result, the training algorithm's efficiency is just as crucial as its classification accuracy [10].

Artificial neural networks (ANNs), case-based reasoning (CBR), and decision trees are examples of machine-learning algorithms that have exhibited high levels of performance in cost estimation [3]. With a mean magnitude relative error (MMRE) of 35–55%, these algorithms were more accurate than statistical algorithms [3]. ANN is a nonlinear modeling tool that consists of an interconnected set of input and output units that does not require any prior information, i.e., there is no need to define the influence between each node subjectively [10]. The ANN model in the human brain is inspired by neurons, and its function is translated to nodes and their linkages. An input layer, an output layer, and a hidden layer between the input and output layers are all present in a standard ANN [8]. ANNs are a type of information processing that may be used to detect patterns and models in large amounts of data [5]. To correlate input and output streams from and process units, ANNs are based on mathematical regression [5]. ANN is also resistant to noise in data. Because of its adaptability and reliance on substantial experimental data, ANNs are utilized as the fundamental technique in computational intelligence [5]. Better accuracy, classification speed, and tolerance for highly interdependent attributes are all advantages of ANNs [24].

Support vector machines (SVMs) are a data classification method that uses a linear classifier to discover patterns in noisy and complex data sets [5]. SVM is a revolutionary machine-learning technique that uses statistical learning theory to determine the best separating hyperplane in a high-dimensional space to separate various classes [10]. Because of kernels and the lack of local minima, SVM can produce very accurate results even on noisy data, making it excellent for modeling complex linear and non-linear problems with high accuracy [3]. SVMs are known for their accuracy, classification speed, and tolerance for irrelevant attributes [24]. The speed of learning in terms of the number of characteristics and model parameter handling are also weaknesses of SVMs [24].

Automated planning, k-nearest neighbor (kNN), and decision trees are some of the other machine learning approaches and algorithms that can be used in project management [25]. Automated planning is an AI technique that investigates a deliberative process that selects and organizes actions by computationally anticipating their expected outcomes [5]. When the samples are not balanced, however, classification results are likely to be skewed [11]. The speed of learning in terms of the amount of attributes and attempts at incremental learning are two of kNN's strengths [24]. The speed of classification, tolerance for missing values, and highly interrelated variables are all weaknesses of kNN [24, 25]. To classify samples, decision trees structure the rules in a tree [11]. While the generalization process of a decision tree is simple to grasp and explain, when the training data is too extensive, overfitting learning can lead to poor model generalization [15].

3 Methods

In this study, we adopted a post-positivist ideology. We were focused on examining historical big data to identify the likely causes of IT project failure. In the literature review, we narrowed the selection of relevant methods to predictive-conceptual ML approaches, referring to Fig. 1 lower right quadrant. Based on the Han et al. [26] study, we further examined their recommendation to select ANN or random forest ML techniques for ambiguous big data exploratory sample prediction. Our goal was not to find the best ML technique but rather to apply the most appropriate ML technique to answer the RQ.

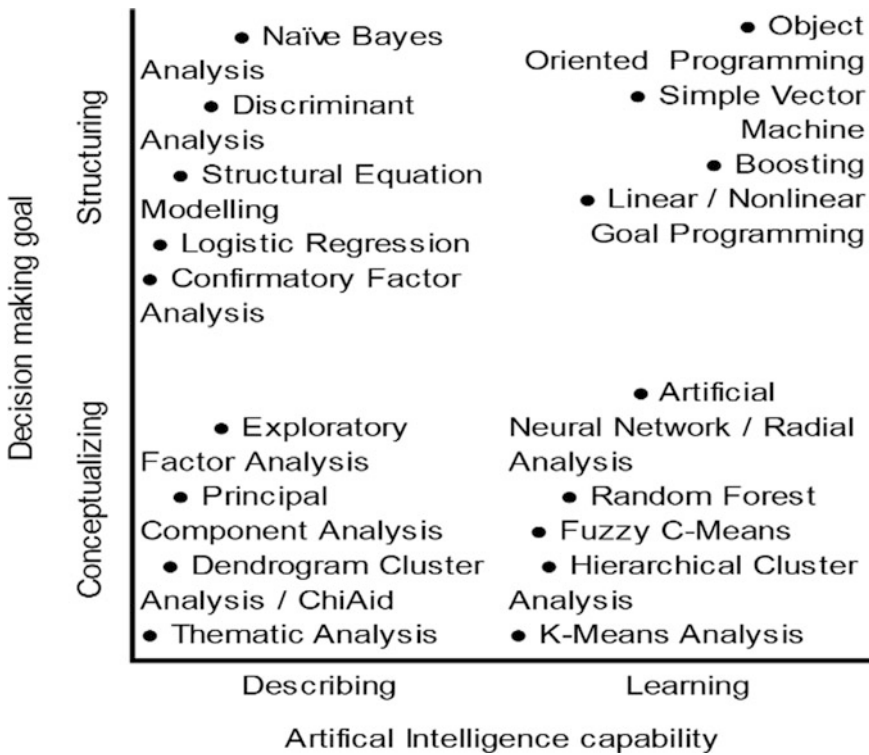


Fig. 1 Common ML technique typology.

We found the random forest a more flexible ML approach than ANN. The latter ANN was better suited for multivariate big data analysis using unspecified predictors created through bootstrap analysis. We determined that random forest was better suited for big data analytics when numerous mixed data type predictors were available and unknown. There was only one important metric-dependent variable (a binary number was acceptable).

Therefore, we proceeded with the ML analysis without specifying the hypothesis according to the current practice. Instead, our RQ was refined as – Can the random forest ML technique identify the most important factors associated with IT project failure based on declassified big data from the defense industry? We obtained ethical clearance from employers and approval to access a big data repository. The dependent variable was coded as failure (breach) = 1 or success (no breach) = 0. The ML training subset and statistical results are available to other scholars upon request to the corresponding author. The dataset may be requested from <http://defense.gov> (USA).

4 Results and Discussion

The average age was 41.9 years (SD = 13.3) at the individual level of analysis, and 94% of the participants were male, with the remaining 6% female. Obviously, the model’s statistical skew would be caused by the high male-to-female ratio and the associated low female-to-male ratio. PMs had 14.1 years of relevant experience on average (SD = 5.9). In terms of educational attainment, 95% had a college diploma, associate degree, or bachelor’s degree. A handful had a master’s or doctorate, while others merely had a high school diploma. 55.6% had a bachelor’s degree or more, 24.4% had a vocational or trade school diploma, 10.9% had an associate degree, and 4.4% had just completed elementary school.

Table 1 Random forest ML confusion matrix

		Predicted	
		No	Yes
Observed	No	958	77
	Yes	190	181

The essential factor classification estimations using the random forest ML approach are listed in Table 1. Training samples ($n = 4500$), validation samples ($n = 1126$), and test samples ($n = 1406$) were separated from the records. With no more than four predictors per split, a total of 94 nodes were created. The test validation accuracy was 78.4%, and the confirmation accuracy was 78.4%, with a 50% out-of-the-bag estimate. When compared to other empirical ML investigations, such as Han et al. [26], they were promising results. The accuracy out-of-the-bag (OOB) is a forecast estimate based on unseen data. A plot of the OOB estimates is shown in Fig. 2. The OOB plot shows that the training and testing outcomes were nearly identical, with 72% accuracy achieved early in the training process, well below 10 trees.

The important confusion matrix estimations from the random forest ML model testing process are summarized in Table 1. We may calculate the marginal and conditional probabilities of recall sensitivity, specificity, accuracy, and precision by

focusing on the joint frequencies at the intersections of the observed vs predicted cells.

The true negative (TN) is represented by the No/No cell (958) while the true positive (TP) is represented by the Yes/Yes cell (181) (TP). The cell with the observed Yes and expected No (190) is a false negative (FN), while the cell with the observed No and predicted Yes (77) is a false positive (FP) (FP). The recall sensitivity rate is computed using the formula $TP / (TP + FN)$, which equals $181 / (181 + 190) = 48.8\%$. $TP / (TP + FP) = 181 / (181 + 77) = 70.1\%$ is used to calculate precision. $TN / (TN + FP) = 958 / (958 + 77) = 92.6\%$ $TN / (TN + FN) = 958 / (958 + 190) = 81\%$ yields the accuracy. These figures show that the model's forecast accuracy was good.

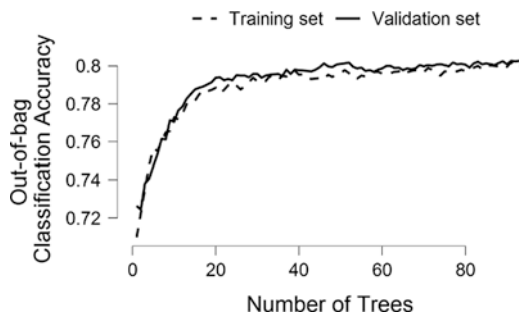


Fig. 2 Random forest ML out-of-bag classification accuracy plot

A plot of the ROCa using the random forest ML evaluation data estimations is shown in Fig. 3. The breach condition in the diagram denotes project failure; a no indicates that the project was not breached and so was successful. The true positive and erroneous negative estimates are plotted in this graph. It shows that both conditions were accurately predicted up to about an 80% true positive level, with a 20% false positive rate. This is like the Pareto 80/20 principle from Lean Six Sigma, which states that in general, 80% of the success/failure outcomes can be attributed to 20% of the causal factors.

5 Conclusion

The most important contribution of this study was to show how to select the most appropriate ML technique to match the research design strategy, data types, and size. In this study, RF was chosen because the design was positivistic, with quantitative data types, the RQ was to identify predictive factors with a dependent variable known.

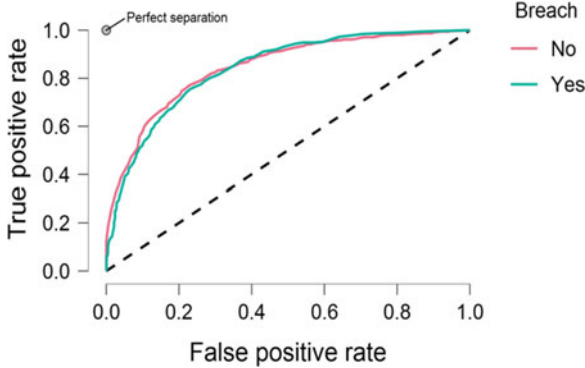


Fig. 3 Random forest ML ROC area curves plot

Overall, the random forest ML technique was accurate and practical for analyzing big data from over 7000 declassified military projects to identify the critical failure factors. The ML model training accuracy was 0.784 and the validation accuracy based on the confusion matrix was 81%. Another key ML model statistic was the average area under the ROC which was 85%. These estimates were comparable to similar studies including the IT software defect project paper published by Han et al. [26].

Nevertheless, although we concluded the random forest ML was useful for initially identifying important factors from big data, we would recommend additional statistical techniques be applied as a next step to fine-tune the predictive model. Furthermore, in contrast to the recommendations of Han et al. [26], we did not conduct preliminary analysis using a general linear model (GLM) to identify the most likely factors because we believe ML would accomplish that step. We thought GLM or similar predictive regression technique would be a logical next step after we initially identified the most important factors related to project success or failure. Therefore, we would recommend a logistic linear regression model be developed as the next project using the same big data or similar sample. The dependent variable would be project outcome: success or failure. All factors with importance of at least 0.01 from the random forest ML project would be set as predictors.

As with any big data study, the limitations are tied to the returned records and their accuracy. While the random forest ML technique performed well for this big data sample with 81% accuracy, we would encourage researchers to try other ML techniques in the structured-learning category besides random forest used in this study because those are the procedures capable of predicting behavior and generating reliability statistical estimates.

References

1. Kurek, E., J. Johnson, and H. Mulder, *Measuring the Value of Enterprise Architecture on IT Projects with CHAOS Research*. Systems, Cybernetics, and Informatics, 2017. **15**(7): p. 13–18.
2. Masticola, S.P. *A Simple Estimate of the Cost of Software Project Failures and the Breakeven Effectiveness of Project Risk Management*. in *2007 First International Workshop on the Economics of Software and Computation*. 2007.
3. Pospieszny, P., B. Czarnacka-Chrobot, and A. Kobylinski, *An effective approach for software project effort and duration estimation with machine learning algorithms*. Journal of Systems and Software, 2018. **137**(2): p. 184–196.
4. Haseeb, M., et al., *Industry 4.0: A Solution towards technology challenges of sustainable business performance*. Social Sciences, 2019. **8**(5).
5. Toorajipour, R., et al., *Artificial intelligence in supply chain management: A systematic literature review*. Journal of Business Research, 2021. **122**: p. 502–517.
6. Cao, C., et al., *Deep Learning and Its Applications in Biomedicine*. Genomics, Proteomics & Bioinformatics, 2018. **16**(1): p. 17–32.
7. Oliveira, A.L., *Biotechnology, big data and artificial intelligence*. Biotechnology journal, 2019. **14**(8): p. 45–53.
8. Subasi, A., *Chapter 3 – Machine learning techniques*, in *Practical Machine Learning for Data Analysis Using Python*, A. Subasi, Editor. 2020, Academic Press. p. 91–202.
9. Memeti, S., et al., *A review of machine learning and meta-heuristic methods for scheduling parallel computing systems*, in *Proceedings of the International Conference on Learning and Optimization Algorithms: Theory and Applications*. 2018, Association for Computing Machinery: Rabat, Morocco. p. 23–35.
10. Hu, Y., et al. *An Intelligent Model for Software Project Risk Prediction*. in *2009 International Conference on Information Management, Innovation Management and Industrial Engineering*. 2009.
11. Wang, M., et al., *A Survey on Large-Scale Machine Learning*. IEEE Transactions on Knowledge and Data Engineering, 2020: p. 1–12.
12. Lawson, C.E., et al., *Machine learning for metabolic engineering: A review*. Metabolic Engineering, 2021. **63**: p. 34–60.
13. Istiaque Ahmed, K., et al., *Machine Learning for Authentication and Authorization in IoT: Taxonomy, Challenges and Future Research Direction*. Sensors, 2021. **21**(15): p. 5122.
14. Antonio, A., et al., *Machine learning applied in the stock market through the Moving Average Convergence Divergence (MACD) indicator*. Investment Management and Financial Innovations, 2020. **17**(4): p. 44–60.
15. Wang, J., et al., *A Survey on Trust Evaluation Based on Machine Learning*. ACM Comput. Surv., 2020. **53**(5): p. Article 107.
16. Ahmad, A., et al., *A Systematic Literature Review on Using Machine Learning Algorithms for Software Requirements Identification on Stack Overflow*. Security & Communication Networks, 2020: p. 1–19.
17. Helmy, M., D. Smith, and K. Selvarajoo, *Systems biology approaches integrated with artificial intelligence for optimized metabolic engineering*. Metab Eng Commun, 2020. **11**: p. e00149.
18. Nakano, F.K., M. Lietaert, and C. Vens, *Machine learning for discovering missing or wrong protein function annotations*. BMC Bioinformatics, 2019(1).
19. Cuperlovic-Culf, M., *Machine Learning Methods for Analysis of Metabolic Data and Metabolic Pathway Modeling*. Metabolites, 2018. **8**(1).
20. Patel, D., R. Modi, and K. Sarvakar, *A Comparative Study of Clustering Data Mining: Techniques and Research Challenges*. International Journal of Latest Technology in Engineering, Management & Applied Science, 2014. **3**(9): p. 67–70.
21. Alsawalqah, H., et al., *Software Defect Prediction Using Heterogeneous Ensemble Classification Based on Segmented Patterns*. Applied Sciences, 2020. **10**(5): p. 1745.

22. Lu, X., F. Feng, and Z. O'Neill, *Occupancy Sensing in Buildings through Social Media from Semantic Analysis*. ASHRAE Transactions, 2020. **126**(1).
23. Ghahremani, L., S. Niknami, and M. Nazari, *The Prediction of Physical Activity Intention and Behavior in Elderly Male Residents of a Nursing Home: A Comparison of Two Behavioral Theories*. Iranian Journal of Medical Sciences, 2012. **37**(1): p. 23–31.
24. Omar, S.J., K. Fred, and K.K. Swaib, *A state-of-the-art review of machine learning techniques for fraud detection research*, in *Proceedings of the 2018 International Conference on Software Engineering in Africa*. 2018, Association for Computing Machinery: Gothenburg, Sweden. p. 11–19.
25. Mabayoje, M., et al., *Parameter tuning in KNN for software defect prediction: an empirical analysis*. Jurnal Teknologi dan Sistem Komputer, 2019. **7**: p. 121–126.
26. Han, W., C.-H. Lung, and S. Ajila, *Using source code and process metrics for defect prediction – A case study of three algorithms and dimensionality reduction*. Journal of Software, 2016. **11**(9): p. 883–902.

Prediction of Heart Disease Using Various Data Analysis and Machine Learning Techniques



Anjali Singh, Deepanshi Vij, Alpana Jijja, and Sherry Verma

Keywords Data analysis · Machine learning · Support vector machine · Decision tree · K-nearest neighbour · Logistic regression

1 Introduction

One of the major branches of artificial intelligence (AI) is machine learning (ML), and this remains to be the most rapidly developing subfield of AI research. The increase of data from various sources makes it even more important to use ML for data analysis. It deals with imperfect data and data interpretation. There are several applications of ML, and one of the significant areas is the healthcare industry. Machine learning algorithms provide several techniques that detect patterns associated with the disease. Within the healthcare industry, large amounts of existing data need to be processed to infer the un-analysed data to visualize dormant trends for a more effective and meaningful interpretation. These, in turn, will address the diagnostics and prognostics problems currently prevailing in several medical domains. A huge application area that provides significant assistance in medical diagnosis is the advanced computer-based medical image interpretation system. Recent applications include cardiac surgery and other heart-related diagnoses where many researchers are contributing as heart attacks are increasing at a very alarming rate. The heart is the most vital organ of the human body which can fit into the fist of our hand. The blood is carried to different organs of the body through the veins and nerves. The majority of deaths in the elderly is heart attacks or heart strokes. Now deaths due to heart attack or heart stroke is seen rising among the younger generations as well and it is difficult to know how long the person will live after the

A. Singh (✉) · D. Vij · A. Jijja · S. Verma
Computer Science and Engineering, Sushant University, Gurgaon, Haryana, India
e-mail: anjalisingh.btech18@sushantuniversity.edu.in;
deepanshivij.btech18@sushantuniversity.edu.in; alpanajijja@sushantuniversity.edu.in;
sherryverma@sushantuniversity.edu.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
R. Misra et al. (eds.), *Machine Learning and Big Data Analytics*,
Springer Proceedings in Mathematics & Statistics 401,
https://doi.org/10.1007/978-3-031-15175-0_3

attack. A multitude of risk factors contribute to heart stroke. The most common being excessive use of tobacco/substance abuse, excessive alcohol consumption and smoking, high blood pressure, lack of physical activity, and inappropriate diet. Cardiovascular ailment is the leading cause of death worldwide, with 17.9 million people perishing each year, as per the World Health Organization [1]. These life-threatening diseases are on rise every year; still sufficient research has to be carried out so that early detection can be made possible. If the problem is diagnosed early enough, the majority of cases can be solved. This is only possible with the advancement of technology, such as data analysis and machine learning. The state-of-the-art technology, Internet of Things is emerging as a modern technology that can be integrated with artificial neural network-based supervised machine learning approach [2].

The key objective of this study is to identify the machine learning algorithms that outperform different vitals impacting heart attack. In addition, this will help address patients with cardiovascular condition to predict and understand their health status based on their current health and previous heart ECG readings. Subsequently, users can review their heart condition in case they are at risk of a heart attack. The proposed study uses both machine learning and data analysis algorithms to predict and analyse an individual's health conditions. One of the important goals of this research work is to classify and predict a person's cardiac signals acquired from ECG recordings.

2 Related Work

A vast number of researchers for almost two decades have undertaken various projects to predict heart disease using various artificial intelligence techniques. A slew of machine learning techniques used to predict heart condition have remained the major part of this research. Various algorithms, namely the decision trees, naive Bayes, neural networks, kernel density, automatically specified classes, the Bagging algorithm and Support Vector Machine are differently supervised learning algorithms that result in better accuracy on a significant number of databases of cardiovascular patients from different parts across the world.

Coiera [3] designed a system for cardiac bypass patients for detecting normal and abnormal cardiac conditions. To achieve the highest accuracy prediction in the medical field artificial neural network (ANN) has been used extensively [4]. Cheng [5] proposed a hybrid model for heart patients that gave better results than ANN, DT and SVM. The performance of the hybrid model was 86.8. Tomas and Princy [6] implemented several data mining techniques that helped successfully in detecting and analysing heart problems, making it one of the averred techniques for researchers [7]. The data mining methods using DT, NN, SVM and KNN algorithms were implemented on a heart dataset. From among these methods, the results from

SVM show improved accuracy in the prediction of disease. The distinguishing factor amongst these previous works to the present work has been the incomplete data and data analysis techniques and latest machine learning libraries. Multiple authors have listed various parameters and databases, for checking the accuracy [8]. Verma has proposed Internet of Things with wireless sensors in various domains. Security is one of the key features and can be used to secure the data.

3 Proposed Work

The proposed research is divided into two parts: the pre-processing phase, in which the most important attributes are selected. In the second phase, different machine learning algorithms are applied and compared to determine the best algorithm that contributes to better accuracy. The data analysis is implemented to plot the graphical representation of various interrelated attributes. Concurrently the proposed work presented the data classification based on various supervised machine learning algorithms, namely, logistic regression, K-nearest neighbour (KNN), decision tree (DT), and support vector machine (SVM).

3.1 Data Collection

The patient dataset is compiled from the following sources:

- I. Cleveland Clinic Foundation (Cleveland.data)
- II. Hungarian Institute of Cardiology, Budapest (hunarian.data)
- III. V.A. Medical Center, Long Beach, CA (long-beach-va.data)
- IV. University Hospital, Zurich, Switzerland (Switzerland.data)

This database's data is all in the same format. There are 76 attributes in this database, but only 14 are included. (No. of rows)

3.2 Pre-processing

The database used extensively for machine learning by researchers till date has been the Cleveland database. The patients' names and social security numbers were deleted from the database and replaced with dummy values. Subsequently, the data now includes a "target" column to predict the chances of heart attack. With the assigned value of 0 (no or less chance of heart attack) and 1 (more chance of heart attack). Table 1 depicts the numerical values for all of the 14 attributes, with their values used in the datasets.

Table 1 Dataset attributes.

S_no	Attributes	Explanation
1	age	Age in years
2	sex	Sex (1 = male; 0 = female)
3	cp	Chest pain type
4	trestbps	Resting blood pressure (in mm Hg on admission to the hospital)
5	chol	Serum cholesterol in mg/dl
6	fbs	Fasting blood sugar >120 mg/dl (1 = true; 0 = false)
7	restecg	Resting electrocardiographic
8	thalach	Maximum heart rate achieved
9	exang	Exercise-induced angina (1 = yes; 0 = no)
10	oldpeak	ST depression
11	slope	The slope of the peak exercise ST segment
12	ca	Number of major vessels (0–3) coloured by fluoroscopy
13	thal	0 = normal; 1 = fixed defect; 2 = reversible defect
14	Target	0 = less chance of heart attack; 1 = more chance of heart attack

The most important step is to clean the data because it might contain some noisy data that is not needed for our work. Data cleaning and data integration helps reduce dimensionality while also improving accuracy and efficiency. The collected data will be trained and fed into the system, with many classification techniques used to extract useful information. The ultimate step is to predict the probability of heart attack using the various parameters observed in the patient. This paper includes the different machine learning algorithms used and the results including the accuracy of each algorithm. To achieve the objective, the initial process involves selecting the features relevant for the analysis and later splitting of data in training and test datasets. Further several machine learning algorithms are applied to get the result of analysis.

3.3 Features Selection

The correlation matrix is used in this step. Initially, the data includes a list of 20 attributes. The 13 attributes (age, sex, cp, fbs, trestbps, chol, restecg, exang, thalach, oldpeak, slope, ca, thal) are finalized that are related and dependent on each other. This is shown using the Pearson correlation matrix. Figure 1 shows the columns selected for applying different machine learning algorithms.

The pre-processed dataset is split into two parts: The training dataset has an 80% ratio, while the test dataset has a 20% size. The aim of dividing the data in 80:20 ratio is that the dataset should have the general essence of our original dataset. Figure 2 shows division of data in training and test data.


```
X = data.drop('target', axis=1)
y = data['target']
```

Fig. 1 Features selection

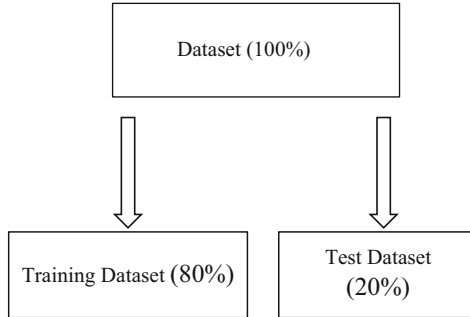


Fig. 2 Proposed model

4 Machine Learning Algorithms Implementation

There are many algorithms proposed by researchers which effectively help to predict heart attack, some of them are decision tree, K-means algorithms, logistic regression and support vector machine. Each of them is explained as follows:

4.1 Logistic Regression

Logistic regression (LR) endures to remain as to be one of the most extensively used approaches in data mining in general and binary data classification in particular [9]. The logistic regress is one of the very effective predictive algorithms in machine learning. The main focus is on implementing this algorithm on the dataset and accessing its efficiency compared to other algorithms.

4.2 Support Vector Machine

The extremely preferred supervised machine learning technique is SVM that can be implemented in both classification and regression problems. It finds a hyperplane in the feature space that differentiates between the groups for classification. The extreme points or vectors also called as support vectors help in creating hyperplane. The training data points are represented as points in the feature space by an SVM

model, and is mapped in such a way that points falling in different classes are separated by a large margin [10]. The test data points are eventually plotted into the same space and classified according to where they fall on the margin. Maximum margin in hyperplane indicates better decision boundary.

4.3 *K-Nearest Neighbour*

The K-nearest neighbour algorithm is based on the supervised learning technique and is one of the most primitive machine learning algorithms. The K-NN method stores all available data and classifies a new data point based on its similarity to the existing data. This means that new data can be quickly sorted into a well-defined category using the K-NN method [11].

4.4 *Decision Tree*

One of the ways to represent an algorithm is via decision trees. It is a well-known machine learning algorithm. Several factors contribute to heart disease, including smoking, blood pressure, hypertension, and age [12]. The selection of the root node is the decision tree's biggest obstacle. This element must be included in the root node. Sort the information into distinct categories. The decision tree is easy to interpret. They are non-parametric and they implicitly do feature selection.

Different learning algorithms, such as logistic regression, SVM, decision tree, and K-nearest neighbour, are all evaluated and checked for predicting the chances of heart attack in the machine learning approach [13]. The findings revealed that SVM algorithm had the best results for predicting the likelihood of a heart attack, with the highest accuracy (89.98%).

5 Experimental Results and Discussions

The outcome of the research work is a description of the patient data on the occurrence or absence of a heart attack. The accuracy of various machine learning algorithms on the heart dataset is depicted in Table 2.

Table 2 Evaluation of algorithms on accuracy.

Model	Accuracy
Logistic regression	85.245902
K-nearest neighbour	88.524590
Decision tree	81.967213
Support vector machine	89.985880

The decision tree algorithm generated the lowest accuracy of 81.97%, whereas SVM algorithm has shown the highest accuracy of 89.98%. Figure 3 depicts the results obtained using various machine learning algorithms. It is observed that the support vector machine algorithm shows higher accuracy.

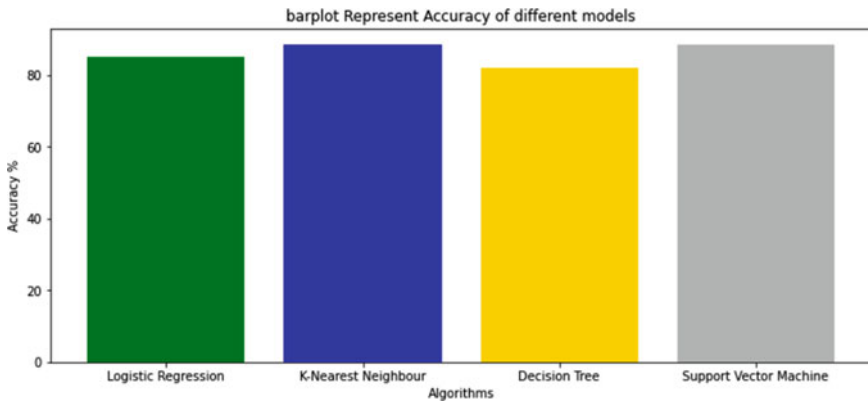


Fig. 3 Bar plot representing the accuracy of different algorithms

5.1 Data Analysis Approach

The important statistical insights were obtained from the data using exploratory data analysis, and the distributions of the different attributes, correlations of the attributes with each other, and the target variable were determined. For the categorical attributes, significant odds and proportions were also determined. Pearson correlation is used to examine the relationship between variables, as shown in Fig. 4. The most significant correlation was discovered between thalach and heart attack.

To determine the feature that has maximum effect on determining the target variable, a bar plot representing features' importance was drawn and found that exercise-induced angina affected the chances for heart attack the most. Figure 5 depicts the differences in heart attack between males and females as a function of age.

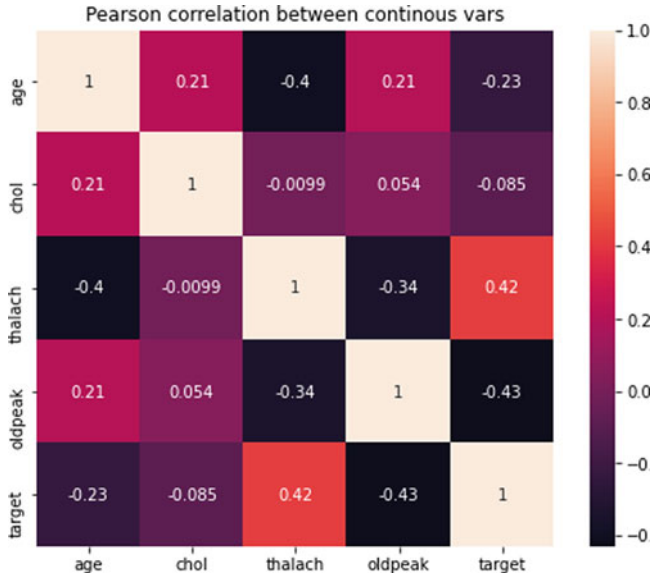


Fig. 4 Pearson correlation among continuous variables.

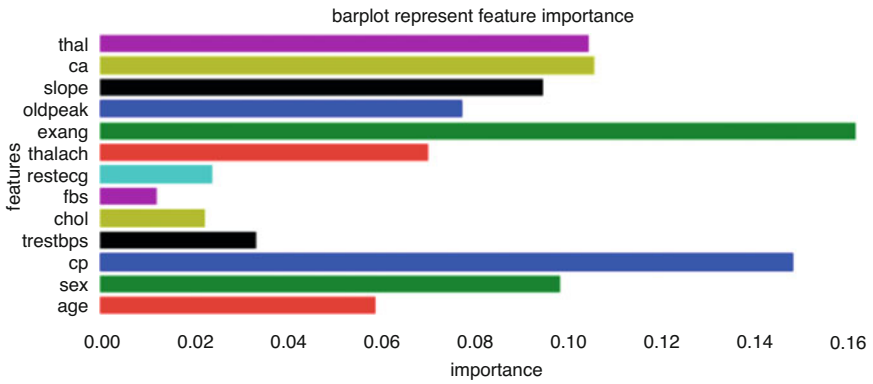


Fig. 5 Bar plot for feature importance

The chances of having a heart attack are depicted in Fig. 6 as a function of age group. The graph clearly shows the relationship of age with gender.

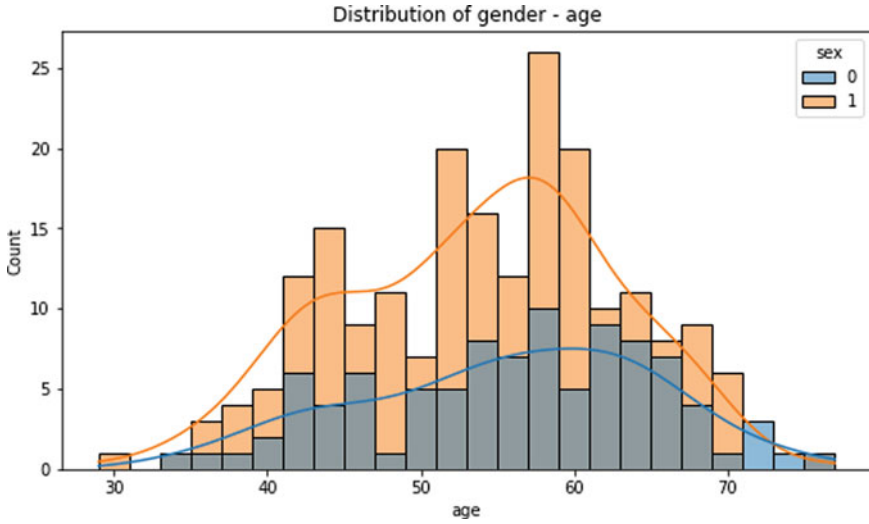


Fig. 6 Bar plot of gender with age.

Figure 7 depicts that 75% of women and 45% of men have a chance of having a heart attack. Moreover, from Fig. 8, it can be observed that those people who exercise regularly have a lower risk of heart attack.

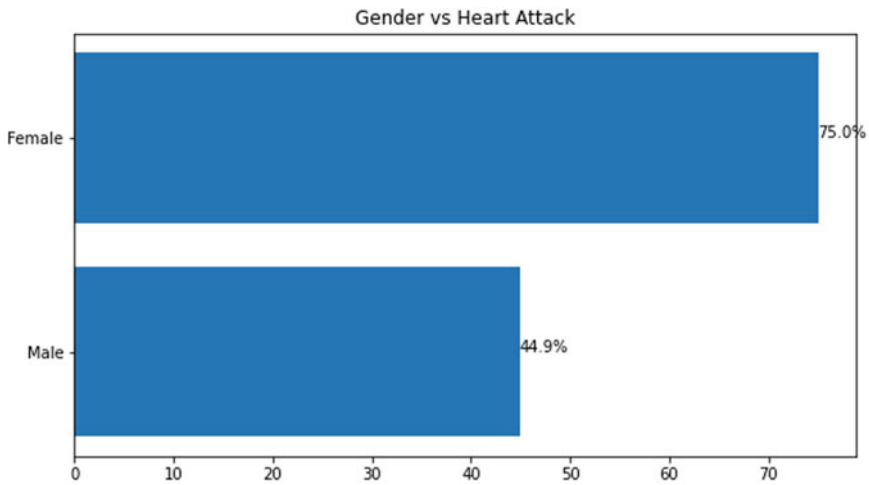


Fig. 7 Heart attack percentage with gender.

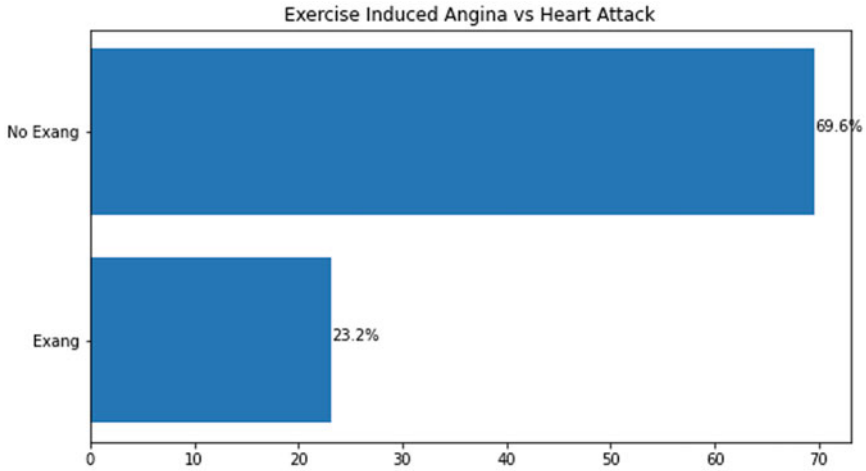


Fig. 8 Heart attack percentage with respect to exercise-induced angina.

Figure 8 shows that people who have non-anginal discomfort are more likely to suffer a heart attack. Figure 9 depicts the percentage of heart attack with respect to chest pain.

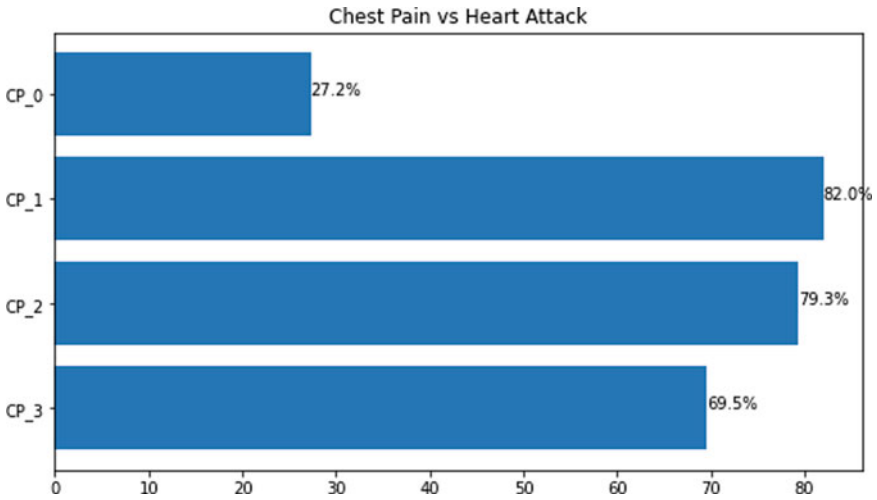


Fig. 9 Heart attack percentage with respect to chest pain.

Figure 10 shows that fasting blood sugar has no effect on heart attack while Fig. 11 depicts those with electrocardiographic RESTECG 1 have an increased risk of

getting heart attack, whereas people with RESTEKG 2 have a relatively lower risk of heart attack.

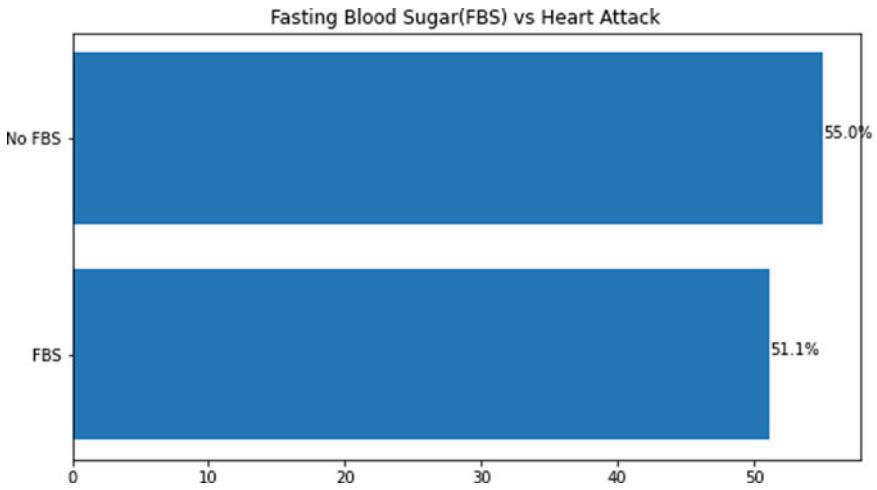


Fig. 10 Heart attack percentage with respect to fasting

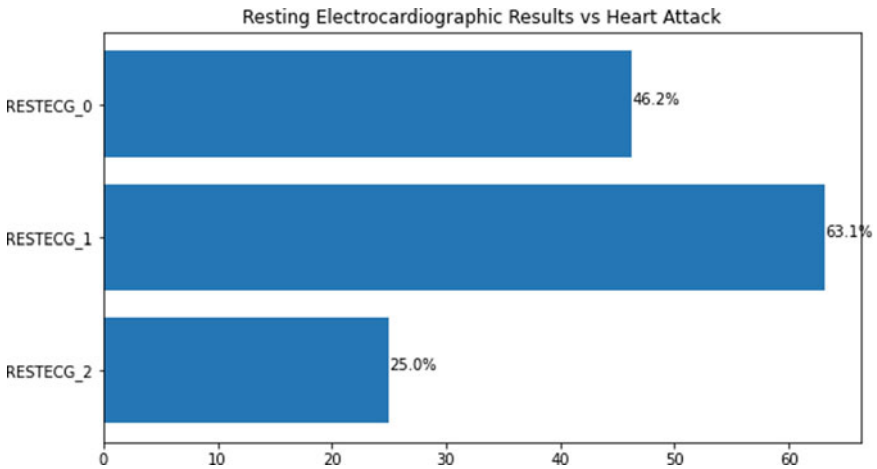


Fig. 11 Heart attack percentage with respect to resting electrocardiographic.

Various researches have worked on the various health dataset; the comparison analysis of SVM algorithm that has proved to be best in the proposed algorithm is shown in Table 3.

Table 3 Comparative analysis of SVM algorithm results.

S_no	Authors	Model	Accuracy (%)
1.	Nahar et al. [14]	SVM	87
2.	Vijayshree et al. [15]	SVM	79
3.	Golpour [16]	SVM	71
4.	Assegie [17]	SVM	77.63
5.	Proposed algorithm	SVM	89.98

6 Conclusion

The main objective of this study is to identify and present the best method to forecast a heart stroke with utmost accuracy. The patient's data, such as cholesterol, age, glucose level, gender, and so on, were used in the proposed study. Machine learning and data analytics were both used in this study using various predictive models, such as logistic regression, decision tree, K-means neighbour, and SVM. These machine learning algorithms compare and find the most reliable model that can predict heart stroke with great accuracy. The graphical representation and visualization of data is the next logical step. Various data analytics procedures revealed and analysed the outputs that were found to be effective in reducing the risk of heart attack. This research may eventually be the most efficient way to cut down the annual death rate due to cardiac arrest across the globe. This proposed research can be implemented in the medical domain, enabling medical professional to rapidly identify heartbeat recordings and subsequently take all required precautions, leading to the saving of numerous lives. The proposed method is very cost-effective and worthwhile as it can predict heart stroke based on a patient's simple data rather than an ECG scan, which is very expensive and out of reach for the economically weaker section of the society. The basic flow for estimating the probability of a heart attack has been explored in this paper, as well as the key attributes that are evaluated. Various algorithms for heart disease prediction were discussed, including decision tree, K-means algorithms, logistic regression, and SVM. The model/system accuracy was subjected to a thorough examination of the different techniques used by the researchers.

There are several ways to boost the prediction system's scalability and accuracy. As can be seen, all authors used the standard 13–14 attributes in their work; however, in future work, the attributes can be decreased in order to improve accuracy by using a feature selection-brute force algorithm with more additional data using a classification technique.

References

1. "Cardiovascular diseases (CVDs)," June.11, 2021. Accessed on: Mar. 25, 2022. [Online]. Available: [https://www.who.int/en/news-room/factsheets/detail/cardiovascular-diseases-\(cvds\)](https://www.who.int/en/news-room/factsheets/detail/cardiovascular-diseases-(cvds))

2. Sharma, S, Verma. V.K, AIEMLA:” Artificial intelligence enabled machine learning approach for routing attacks on internet of things”, *The Journal of Supercomputing, An International Journal of High-Performance Computer Design, Analysis, and Use*, 77, 13757–13787, 2021.
3. Hau, D., and Coiera, E. “Learning qualitative models of dynamic systems”. *Machine Learning*, 26, 177–211, 1997.
4. L. Baccour, “Amended fused TOPSIS-VIKOR for classification (ATOVIC) applied to some UCI data sets”, *Expert Syst. Appl.*, vol. 99, pp. 115–125, Jun. 2018.
5. C.-A. Cheng and H.-W. Chiu, “An artificial neural network model for the evaluation of carotid artery stenting prognosis using a national-wide database”, *Proc. 39th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, pp. 2566–2569, Jul. 2017.
6. J. Thomas and R. T. Princy, “Human heart disease prediction system using data mining techniques”, *Proc. Int. Conf. Circuit Power Comput. Technol. (ICCPCT)*, pp. 1–5, Mar. 2016.
7. Ricco RKOTOMALALA, “TANAGRA: a free software for research and academic purposes”, in *Proceedings of EGC’2005, RNTI-E-3*, vol. 2, pp.697–702, 2005. (in French)
8. Sharma, S, Verma. V.K “An Integrated Exploration on Internet of Things and wireless” *Springer Wireless Personal Communications*, 2022.
9. S. Palaniappan and R. Awang, “Intelligent heart disease prediction system using data mining techniques,” pp. 108–115, 2008
10. M. Hölbl, M. Kompara, A. Kamišalić, and L. N. Zlatolas, “A systematic review of the use of blockchain in healthcare,” *Symmetry (Basel)*, vol. 10, no. 10, p. 470, Oct. 2018.
11. R. Indrakumari, T. Poongodi, and S. R. Jena, “Heart Disease Prediction using Exploratory Data Analysis,” in *Procedia Computer Science*, Jan. 2020, vol. 173, pp. 130–139.
12. Marimuthu, M. Abinaya, K. S. Hariesh, and K. Madhankumar, “A Review on Heart Disease Prediction using Machine Learning and Data Analytics Approach,” *Artic. Int. J. Comput. Appl.*, vol. 181, no. 18, pp. 975–8887, 2018.
13. D. E. Salhi, A. Tari, and M.-T. Kechadi, “Using Machine Learning for Heart Disease Prediction,” 2021, pp. 70–81.
14. Jesmin Nahar, Tasadduq Imam, et al.,” Association rule mining to detect factors which contribute to heart disease in males and females, *J. Expert Syst. Appl.*, vol. 40, pp. 1086–1093, 2013.
15. Vijayashree, J.; Sultana, H. Parveen, “A Machine Learning Framework for Feature Selection in Heart Disease Classification Using Improved Particle Swarm Optimization with Support Vector Machine Classifier”, *Programming and Computer Software*, vol. 44, pp. 388–397, 2018.
16. Golpour, P., Ghayour-Mobarhan, M., Saki, A., Esmaily, H., Taghipour, A., Tajfard, M., ... Ferns, G. A., “Comparison of Support Vector Machine, Naïve Bayes and Logistic Regression for Assessing the Necessity for Coronary Angiography”, *International Journal of Environmental Research and Public Health*, vol. 17(18), pp. 6449. 2020.
17. Assegie, T, “ A Support Vector Machine Based Heart Disease Prediction”, *Journal of Software & Intelligent Systems*, vol. 4, 2019.

Artificial Intelligence: Recent Trends, Opportunities, and Challenges in Real-World Scenarios



Bosubabu Sambana , Yagireddi Ramesh , Satyabrata Patro ,
N. P. Patnaik M , P. J. V. Prakasa Rao , Adimalla Rama Rao,
and Priyanka Mishra 

Keywords Artificial intelligence · Nanotechnology · ANN · Computer vision · Robots

1 Introduction

Artificial intelligence is increasing in management science and operational research areas. Intelligence is commonly considered the ability to collect and reason about knowledge to solve complex problems. Shortly, intelligent machines will replace human capabilities in many areas.

B. Sambana (✉)

Department of Computer Science and Engineering, Lendi Institute of Engineering and Technology (A), Vizianagaram, Andhra Pradesh, India

Jawaharlal Nehru Technological University Kakinada, Andhra Pradesh, India

Department of Computer Science and Engineering, Indian Institute of Information Technology Kota, Jaipur, Rajasthan, India

e-mail: 2021krpc1001@iiitkota.ac.in

Y. Ramesh

Department of Computer Science and Engineering, Aditya Institute of Technology and Management (A), Tekkali, India

S. Patro

Department of Computer Science and Engineering, Raghu Engineering College (A), Visakhapatnam, India

N. P. Patnaik M · P. J. V. Prakasa Rao · A. Rama Rao

Department of Computer Science and Engineering, Lendi Institute of Engineering and Technology (A), Vizianagaram, Andhra Pradesh, India

P. Mishra

Department of Computer Science and Engineering, Indian Institute of Information Technology Kota, Jaipur, Rajasthan, India

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

R. Misra et al. (eds.), *Machine Learning and Big Data Analytics*,

Springer Proceedings in Mathematics & Statistics 401,

https://doi.org/10.1007/978-3-031-15175-0_4

Artificial intelligence is the study and development of intelligent machines and software that can reason, learn, gather knowledge, communicate, manipulate, and perceive objects. John McCarthy coined the term in 1956 as the branch of computer science concerned with making computers behave like humans.

The study of computation makes it possible to perceive reason and act. Artificial intelligence is different from psychology because of its emphasis on analysis and is separate from computer science because of its focus on perception, reasoning, and action. It makes machines more innovative and more valuable. It works with the help of artificial neurons (artificial neural networks) and scientific theorems (if-then statements and logics).

AI technologies have matured to offer practical benefits in many of their applications. Major artificial intelligence areas are Expert Systems, Natural Language Processing, Speech Understanding, Robotics and Sensory Systems, Computer Vision and Scene Recognition, Intelligent Computer-Aided Instruction, and Neural Computing. These Expert Systems are rapidly growing technologies that significantly impact various fields of life. The multiple techniques applied in artificial intelligence are neural networks, fuzzy logic, evolutionary computing, and hybrid artificial intelligence [1].

This paper gives an overview of AI and the current state and possible future directions for the AI applications; nanotechnology plays a vital role in AI, the group of emerging technologies in which the structure of the atom is controlled under a nanometer scale to produce novel materials and devices that have valuable and unique properties. NANOTECHNOLOGY is nowadays widely used in the following areas Electronics, Automobiles, Clothing and Nano-Medicines, and Space Technologies by using Artificial Intelligence. Gaining better control over the matter has been a primary project of our species.

Since we started chipping flint, the cost of our product depends on how difficult it is for us to manufacture. Many fields endeavor to contribute to nanotechnology, including molecular physics, material sciences, chemistry, biology, computer science, electrical engineering, and mechanical engineering [2].

2 Background Work

Artificial intelligence has advantages over natural intelligence. It is more permanent, consistent, less expensive, has ease of duplication and dissemination, can be documented, and can perform specific tasks faster and better than humans.

The Turing Test Approach The Turing test was proposed by Alan Turing (1950). This test was designed to test whether a particular machine can think or not. The test involves a human interrogator who interacts with a human and with a device and has to tell who is human and which one is a machine. After posing some written questions, the computer passes the test if an interrogator cannot tell whether the written response is coming from a human or the engine. Nanotechnology is the

engineering of functional systems at the molecular scale. This covers both current work and concepts that are more advanced. In its original sense, nanotechnology refers to the projected ability to construct items from the bottom up, using techniques and tools developed today to make complete, high-performance products [3].

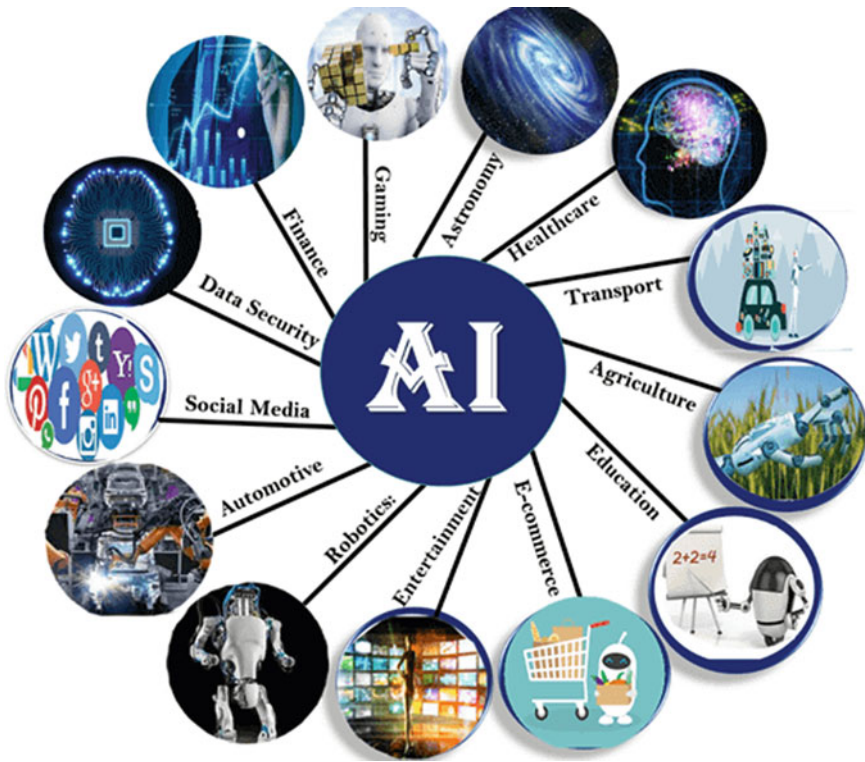


Fig. 1 Artificial intelligence applications in various fields

One nanometer (nm) is one billionth, or 10^{-9} , of a meter. By comparison, typical carbon-carbon bond lengths, or the spacing between these atoms in a molecule, are in the range 0.12–0.15 nm, and a DNA double-helix has a diameter of around 2 nm. On the other hand, the smallest cellular life-forms, the bacteria of the genus *Mycoplasma*, are about 200 nm in length. By convention, nanotechnology is taken as the scale range 1–100 nm, following the definition used by the National Nanotechnology Initiative in the United States. The lower limit is set by the size of atoms (hydrogen has the tiniest particles, which are approximately a quarter of an nm diameter) since nanotechnology must build its devices from atoms and molecules [4]. The upper limit is more or less arbitrary but is around the size that

phenomena not observed in larger structures start to become apparent and can be used in the nanodevice.

2.1 Literature Survey

Raúl Lara-Cabrera, Mariela Nogueira-Collazo, Carlos Cotta, and Antonio J. Fernández-Leiva et al. proposed General Game Playing (GGP). Can a bot play different games without being previously specifically trained for them? This question underlies the research to generate automated general game players.

In some sense, this issue is related to the creation of human-like behaviors, as a general player mimics a human that learns the rules of a game and subsequently can play it without being previously trained on it. The skill to play would be acquired with the game experience, and this is another of the fundamentals under the GGP concept [3, 5].

Artificial neural network: The application of ANNs in medicine includes, but is not limited to, the diagnosis, imaging, back pain, dementia, pathology, and prognosis evaluation of appendicitis, myocardial infarction, acute pulmonary embolism arrhythmias, or psychiatric disorder diseases (Fig. 1). As stated by [8], some of the advantages of ANN are: neural networks can learn linear and nonlinear models. Also, the accuracy of models created by the neural network can be measured statistically.

Finding the best available resource: Clustering assists in improving the quality and efficiency of search engines. In this, the client's query can be initially compared to the clusters irrespective of comparing it directly to the texts, and the search results can be arranged easily [6].

Mohit Sharma and Pranjal Singh "Text mining in big data and similarity measures": Mining is an important technique that organizes many objects into a small number of coherent groups, and it leads to efficient and effective use of these texts for information retrieval. Clustering algorithms require a similarity metric to identify how the two different readers are related/similar to each other [7].

This difference is often measured by some distance measures such as Cosine similarity, Jaccard, and others. The well-known five different distance measures are used in work, and compare their performance on datasets using a k-means clustering algorithm. However, this work lacks efficient feature selection and representation of the terms.

3 Related Work

These new phenomena make nanotechnology distinct from devices that are merely miniaturized versions of an equivalent macroscopic device; such devices are on a larger scale and come under the description of microtechnology. To put that

scale in another context, the comparative size of a nanometer to a meter is the same as that of a marble to the extent of the earth. Or another way of putting it: a nanometer is an amount an average man’s beard grows when it takes him to raise the razor to his face. Two main approaches are used in nanotechnology. In the “bottom-up” approach, materials and devices are built from molecular components chemically by molecular recognition principles. In the “top-down” approach, nano-objects are constructed from larger entities without atomic-level control [9]. Areas of physics such as nanoelectronics, nanomechanics, nanophotonics, and nanoionics have evolved during the last few decades to provide a basic scientific foundation for nanotechnology [10].

3.1 Areas of Artificial Intelligence

A. Natural language processing The ability to “understand” and respond to the natural language (Fig. 2). To translate from spoken language to a written form and from one natural language to another natural language.

- Speech understanding
- Semantic information processing (computational linguistics)
- Question answering
- Information retrieval
- Language translation

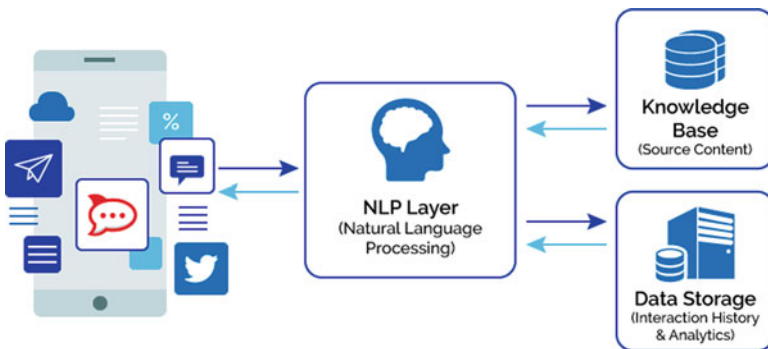


Fig. 2 Artifacts empowered by artificial intelligence

B. Learning and adaptive systems The ability to adapt behavior based on previous experience and develop general rules concerning the world based on such knowledge (Fig. 3).

- Cybernetics
- Concept formation

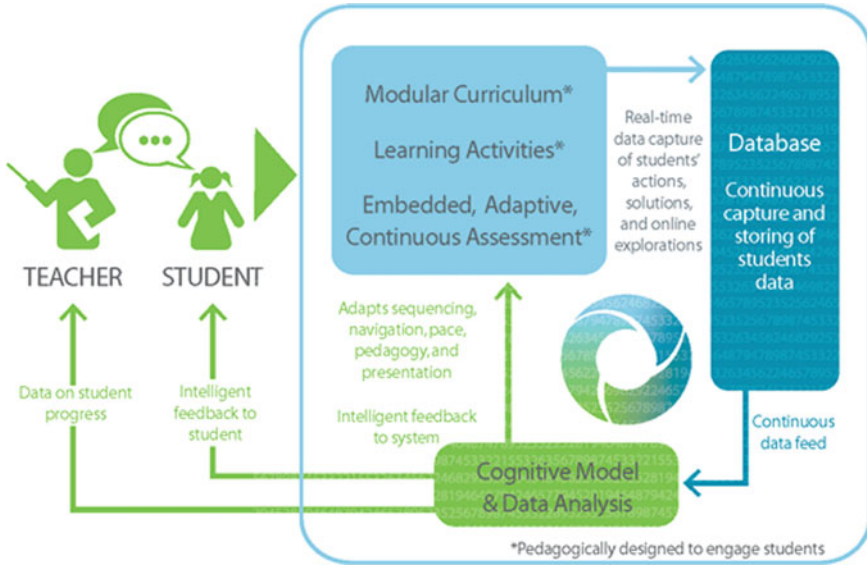
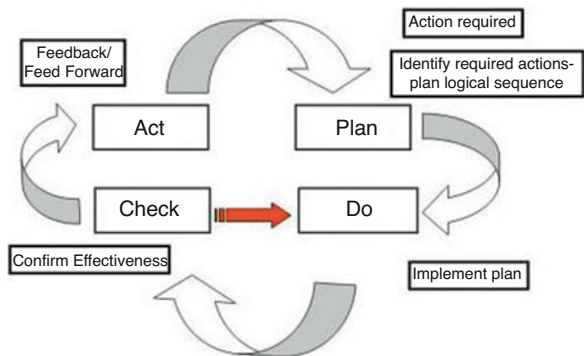


Fig. 3 AI by learning and adaptive systems

C. Problem solving Ability to formulate a problem in a suitable representation, plan for its solution, and know when new information is needed and how to obtain it (Fig. 4).

- Inference (resolution-based theorem proving, plausible inference, and inductive inference)
- Interactive problem solving
- Automatic program writing
- Heuristic search

Fig. 4 Problem solving using agents by using artificial intelligence



D. Perception (visual) The ability to analyze a sensed scene by relating it to an internal model representing the perceiving organism’s “knowledge of the world.” This analysis results in a structured set of relationships between entities in the scene (Fig. 5).

- Pattern recognition
- Scene analysis

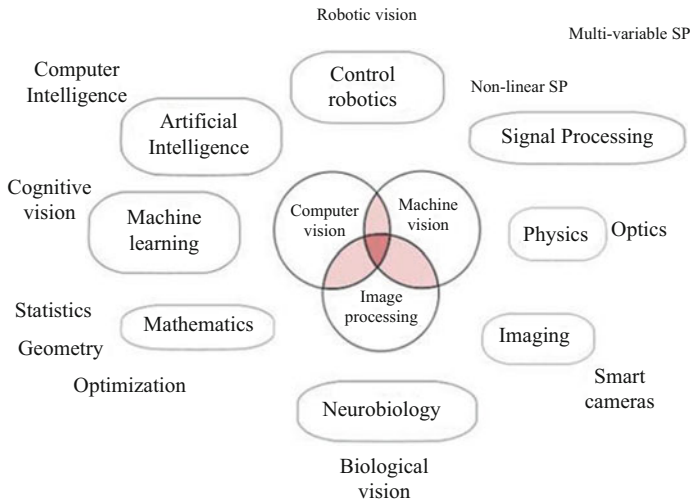


Fig. 5 Computer vision

E. Modeling The ability to develop an internal representation and set of transformation rules that can predict the behavior and relationship (Fig. 6) between some set of real-world objects or entities.

- The representation problem for problem solving systems
- Modeling natural systems (economic, sociological, ecological, biological, etc.)
- Robot world modeling (perceptual and functional representations)

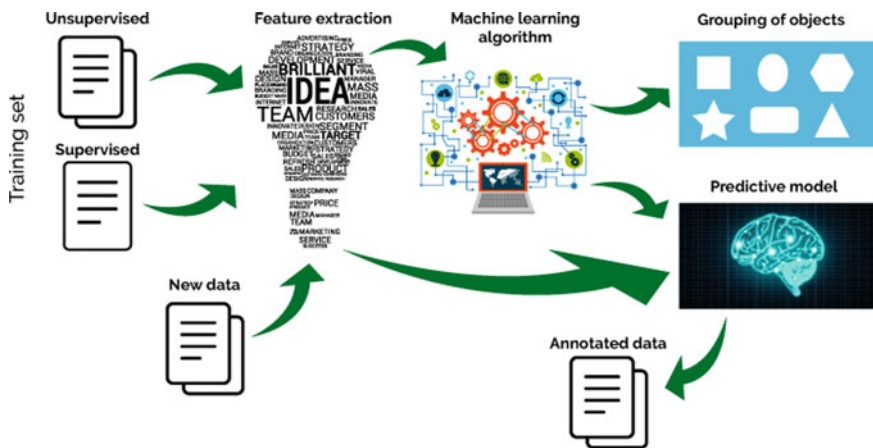


Fig. 6 Machine learning

F. Robots A combination of most or all of the above abilities to move over terrain and manipulate objects (Fig. 7).

- Exploration
- Transportation/navigation
- Industrial automation (e.g., process control, assembly tasks, executive tasks)
- Security
- Other (agriculture, fishing, mining, sanitation, construction, etc.)
- Military
- Household

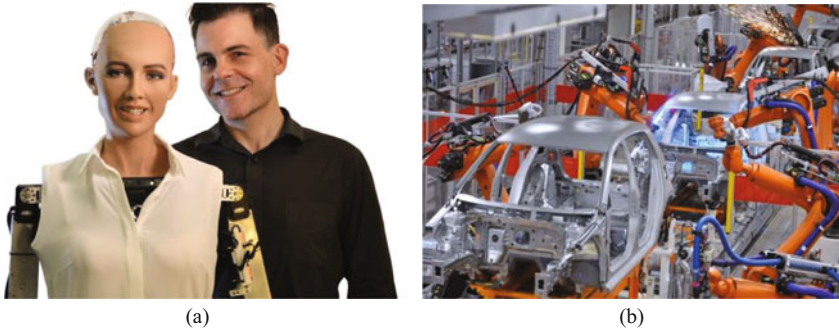


Fig. 7 (a) Famous roboticist Dr. David Hanson (right) says androids like his creation Sophia and (b) robots in industrial automation

G. Games The ability to accept a formal set of rules for games such as Chess, Go, Kalah, and Checkers and to translate these rules into a representation (Fig. 8) or structure that allows problem solving and learning abilities to be used in reaching an adequate level of performance.

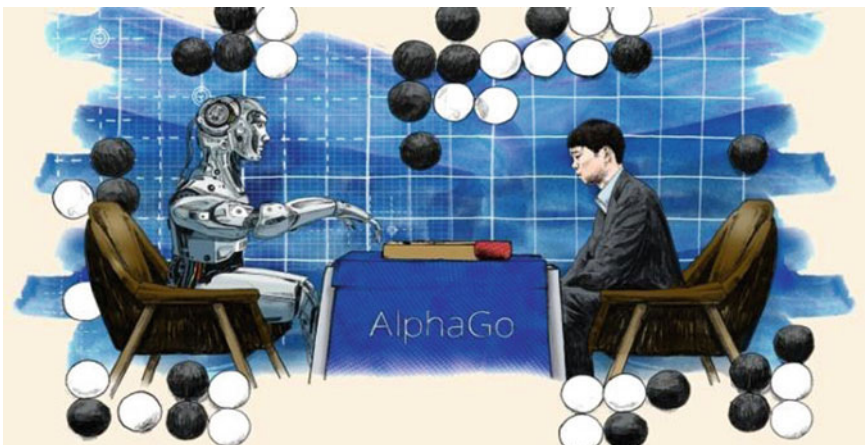


Fig. 8 AI with games

3.2 Applications of Artificial Intelligence

A. Application of Artificial Intelligence Techniques in Power System Stabilizers (PSSs) Design Since the 1960s, PSSs have been used to add damping to electromechanical oscillations. The PSS is an additional control system, often applied as a part of an excitation control system. The primary function of the PSS is to use a signal to the excitation system, producing electrical torques to the rotor in phase with speed differences that damp out power oscillations (Fig. 9). They perform within the generator’s excitation system to create a part of electrical torque, called damping torque, proportional to speed change. This mechanism mainly supports artificial neural networks (ANNs) and fuzzy logic (FL) [9].

B. Artificial Intelligence Techniques in Network Intrusion Detection Systems (IDS) Use various artificial intelligence techniques to protect computer and communication networks from intruders. An intrusion detection system (IDS) monitors the events occurring in network and detects the signs of intrusion.

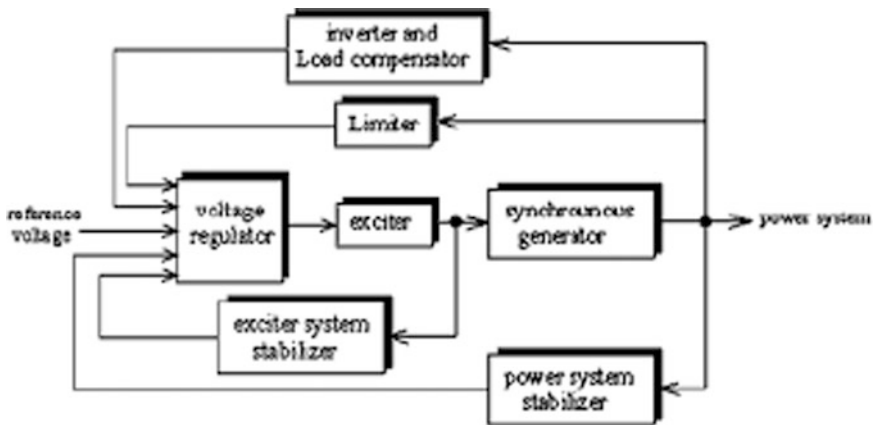


Fig. 9 Artificial intelligent techniques in power system stabilizers

C. Application of Artificial Intelligence Techniques in the Medical Area Artificial intelligence techniques have the potential to be applied in almost every field of the medical area (Fig. 10). Those are follows:

- Artificial Intelligence in Medicine
- Evolutionary Computation in Medicine
- Using Artificial Intelligence to Improve Hospital Inpatient Care: Clinical decision support systems (CDSS) were one of the first successful applications of AI focusing [11]

- Artificial Intelligence Approaches for Medical Image Classification
- Artificial Neural Networks Approach on Diagnostic Science like Endoscopic Images and MRI Brain Tumor Analysis [12]

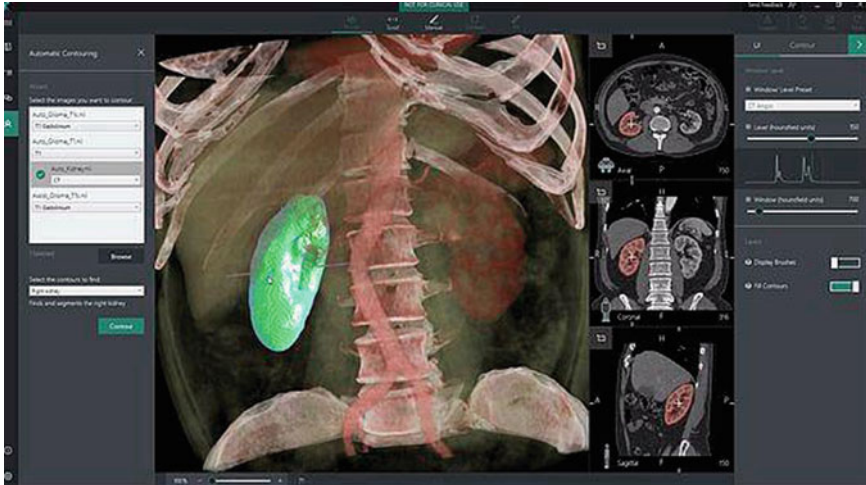


Fig. 10 Artificial intelligence applications using medical devices

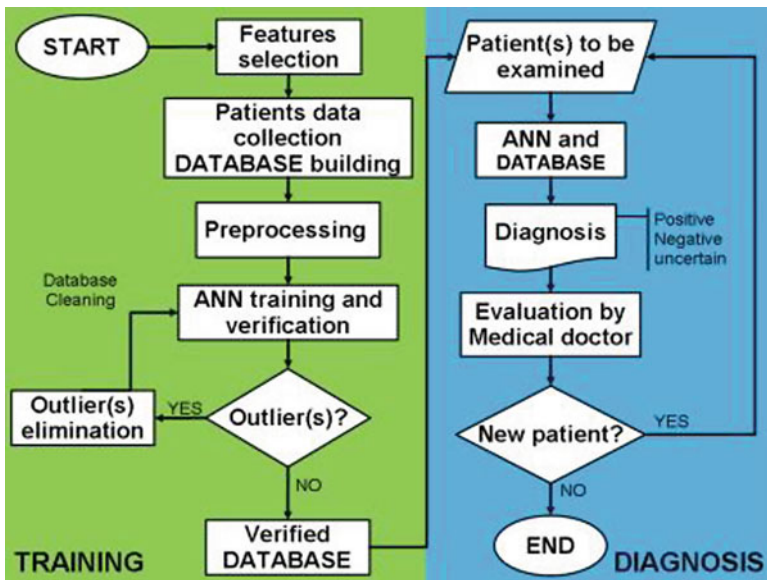


Fig. 11 Fundamental steps in ANN-based medical diagnostic

D. Application of Artificial Intelligence Techniques in the Computer Games

Playing games is one of the most popular uses of computer technology. In the evolution of computer games, they have grown from modest text-based to three-dimensional graphical games with complex and large worlds (Fig. 11). When putting together, the systems such as graphics rendering, playing audio, user input, and artificial game intelligence (AI) provide the expected entertainment and make a worthwhile computer game.

- *Computer Game Problems Solved with AI:* Artificial intelligence solves the three common problems: nonplaying character (NPC) movement, NPC decision-making, and NPC learning. The four artificial intelligence techniques used are path finding, Bayesian networks, fuzzy logic, and genetic algorithms, which help a computer game provide non-playing character pathfinding and decision-making and learning [10].

4 Conclusion

The field of artificial intelligence gives the ability to machines to think analytically, using concepts. Artificial intelligence techniques have significantly contributed to various areas over the last two decades, and artificial intelligence will continue to play an increasingly important role in multiple fields.

This paper is based on the concept of artificial intelligence, areas of artificial intelligence applications, and the artificial intelligence techniques used in the field of various operations to maintain the system stability and damping of oscillation and provide high-quality performance, network intrusion detection to protect the network from intruders, in the medical area in the field of medicine, for medical image classification, and military operations, healthcare operations, in the financial and described how these AI techniques are used in computer games to solve the common problems and to provide features to the games to have fun.

There is a bright future in the analysis of artificial intelligence, and there is also a definite future in applications. We conclude that further research can be done as encouraging and profitable results from such techniques. At the same time, scientists have not yet realized artificial intelligence's full potential and ability. This technology and its applications will likely have far-reaching effects on human life in the years to come.

References

1. C. Sampada, et al., "Adaptive Neuro-Fuzzy Intrusion Detection Systems," Proceedings: International Conference on Information Technology: Coding and Computing (ITCC'04), 2004.

2. Daniel B. Neill, "Using Artificial Intelligence to Improve Hospital Inpatient Care."
3. Daniel E.O. "Leary Artificial Intelligence and Expert System in Accounting Databases: Survey and Extensions," *Expert Systems with Applications*, vol-3, 1991.
4. Fatai Adesina Anifowose, Safiriyu Ibiyemi Eludiora, "Application of Artificial Intelligence in Network Intrusion Detection," *World Applied Programming*, Vol (2), No (3), March 2012.
5. F. D. Laramee, *Genetic Algorithms: Evolving the Perfect Troll, AI Game Programming Wisdom*, Charles River Media, Inc., Hingham, MA, 2002
6. Holland JH, "Adaptation in Natural and Artificial Systems," 1975.
7. Oscar Firschein, Martin A. Fischler, L. Stephen Coles, Jay M. Tenenbaum, "FORECASTING AND ASSESSING THE IMPACT OF ARTIFICIAL INTELLIGENCE ON SOCIETY," unpublished.
8. S.N. Deepa, B. Aruna Devi, "A survey on artificial intelligence approaches for medical image classification," *Indian Journal of Science and Technology*, Vol. 4 No. 11 (Nov 2011).
9. Vassilis S Kodogiannis and John N Lygouras (2008) Neuro-fuzzy classification system for wireless-capsule endoscopic images. *J. World Acad. Sci. Engg. & Technol.*, 45, 620–628.
10. <http://www.wikipedia.org/nanotechnology>
11. Fierce drug delivery Freitas RA Jr (2005) Nanotechnology, nanomedicine, and nanosurgery. *Int J Surg* 3: 243–246.
12. W. Karen. (May 2013). GNS aims to help MDs know which treatment will work the best for each patient. *The Boston Globe*. [Online]. Available: <http://www.bostonglobe.com/business/2013/05/12/personalized-medicine-goal-big-data-scientist/28gTkXjCDj6Zh6KP5tpNBO/story.html>

Bilingual Documents Text Lines Extraction Using Conditional GANs



Sukhandeep Kaur, Seema Bawa, Ravinder Kumar, and Munish Kumar

Keywords GAN · Encoder · Decoder · Pix2Pix · U-Net · PatchGAN · Image-to-image translation

1 Introduction

Bilingual or multilingual document recognition is the next step in OCR research and handwritten recognition. Most of the time people mix two or more than two languages or scripts during writing or speaking in day-to-day life. The prime examples of such documents are filling the application forms, displaying notes on the notice board, taking notes and official letters, etc. The most common languages used in such applications is English along with any other regional language. Apart from this, the digitization of documents is also a need of the day. It makes the storage, retrieval of data, and searching and indexing of thousands of papers easy for the longer period.

Handwritten document text recognition has more challenges as compared to printed documents due to irregular spacing between the lines, huge variations in individual handwriting, irregular page layouts, presence of skewed, curved, and touching data. Further in multilingual documents, due to variations in writing styles of scripts, it is hard to design a single approach to recognize the data from various scripts. Segmentation of text in Gurumukhi script is more challenging due to the presence of header line, connected characters, half characters and overlapping of characters due to modifiers, etc. The characters connected in Gurumukhi script have different definition as compared to the characters connected in some cursive scripts.

S. Kaur (✉) · S. Bawa · R. Kumar
Thapar Institute of Engineering and Technology, Patiala, Punjab, India
e-mail: seema@thapar.edu; ravinder@thapar.edu

M. Kumar
Maharaja Ranjit Singh Punjab Technical University, Bathinda, Punjab, India

A text line of any Indic script has three horizontal parts: upper zone, lower zone, and middle zone except Urdu script [9].

Document text recognition has several steps including OCR and segmentation as the inherent part of the process. Segmentation is the preliminary step in document recognition process which effects the results of other subprocesses. It can be performed at page, line, word, and character level. There are mainly three categories for segmentation approaches classic, recognition, and holistic based. In classic approaches, we have many morphological based approaches, projection profiles, Hough transform, bounding box analysis, and connected component analysis, while in recognition based approaches many machine learning models have been considered. The recognition based model demands high training data and computational power to extract the features of documents. It requires a lot of time to collect the various writing styles and patterns with huge variety for handwritten documents. Due to this, recognition based approaches are less explored in segmentation of textual data [18].

Furthermore, the simplicity of classical approaches makes it difficult for a single approach to segment all kinds of documents. The learning based approaches learn each and every individual feature in an iterative manner. Hence, these can give more correct results. The problem with these approaches is the unavailability of training data. However, Goodfellow has designed GAN models to overcome this issue. They have generated the synthetic data similar to ground truth data to increase the large data requirements of deep networks. There are a number of variants of GANs which have been successfully implemented for many other tasks apart from generating data like semantic segmentation, image translation, object detection, dialog generation, video predictions and image editing, etc. [19]. In GAN, two networks compete with each other simultaneously, i.e., generator and discriminator. Generator tries to generate the data which mimic the real data while discriminator competes for differentiating the generated data from the real data. To get a more control over the generated data, many GANs have been formed with some changes in generator network like Deep Convolutional GAN (DCGAN), Conditional GAN (CGAN), Wasserstein loss GAN (WGAN), Least Square GAN (LSGAN), etc.

In this research, we are using the modeling power of Generative models to perform semantic segmentation of bilingual handwritten documents. The text segmentation task has been considered as the image translation task to extract the text lines. Pix2Pix GAN models are the best considered models for image translation which have been used in this research. This is the first kind of work which uses GAN models in text segmentation for bilingual documents for Gurumukhi-English text. The major motivations behind this work are as follows:

- To design a robust text segmentation approach for wide variety of documents.
- The machine learning models have been less explored for text segmentation in Gurumukhi script.
- To handle the irregular spacing between the lines and page layouts.

2 Literature Survey

Text line segmentation has been mainly performed by classical approaches like projection profiles and Hough transforms. Santose et al. [2] have extracted the text lines using morphological and projection profile information. Morphological operations extract the features of an image while projection histograms locate the position of text lines. A number of projection histograms with threshold values have been applied to minimize the loss of segmented text lines. Susan et al. [3] have used adaptive thresholding method to segment the text in document images. The distance matrix has been calculated between the texture features of document images and fixed training template images. The adaptive thresholding using iterative gamma correction plays the role of human eye for varying light intensities to make the text area clearer against background. Many benchmark document image datasets like DIBCO 2009, 2010, 2011, 2012, and 2013 have been used for evaluation. Similarly, Pal et al. [4] have used horizontal and vertical projection profiles to segment the unconstrained handwritten Bangla text at line, word, and character level. Many statistical and structural properties of text have been considered for text line and word segmentation. For character segmentation, water reservoir based approach has been applied. Another work by Jindal et al. [5] has solved the problem of horizontally overlapping lines in printed Indian scripts in text line segmentation using projection profiles. The presence of lower and upper zone modifiers in Indian scripts makes the task of line segmentation difficult. In this, the statistical approach has been designed to solve major issues in overlapped lines. Sanasam et al. [7] have used mid-point detection and gap trailing for text line segmentation in Meitie Mayek handwritten documents. Initially, horizontal projection profiles are taken, and to track the gap between two adjacent lines mid-point is detected. The detected mid-point generates a starting point to trail through the gap between lines using projection profiles and statistical information. Projection profile based methods are threshold dependent which creates hurdles for segmentation of unconstrained handwritten text. Hence, Ptak et al. [8] have used variable threshold with projection profiles to segment the text lines in handwritten documents. Furthermore, Jindal et al. [9] have proposed text line segmentation of handwritten Gurumukhi manuscripts with touching and skewed lines. Documents have been divided into vertical stripes, and further projection profiles are taken. To segment the touching components, connected component analysis has been performed. Mohammad et al. [10] have explored the text line extraction in printed Arabic documents. This is based on baseline estimation approach with connected component analysis using projection profiles in an iterative manner. Zhu et al. have designed the TextMountain named scene text detector. It uses the center and border information of text written to separate the text instances [20].

In recognition based approaches, Sharma et al. [6] have used machine learning in solving the word segmentation task. SURF based features have been extracted to calculate the inter and intra word gap for word segmentation. The extracted features are trained using structural Support Vector Machine (SVM). The results

of proposed approach on ICDAR 2009 and ICDAR 2013 are found efficient. Jo et al. [1] have explored the convolutional neural networks for handwritten text segmentation. It is the pixel-level classification approach which does not require any kind of preprocessing. They had adopted the U-Net based architecture with encoder and decoders which results into segmentation of text. The data synthesis has been performed using mixture of printed and handwritten data. They have also proposed cross entropy based loss function. Majid et al. [15] have used segmentation-free handwritten recognition using sequential based faster R-CNN. To train the networks, Vgg16 based transfer learning has been adopted. The method has been tested on essay scripts from Bangla handwriting and found promising results for word and character error rates. Similarly, Dutta et al. have used multi-scale CNN for line segmentation in printed and handwritten documents containing, text, graphics, tables, etc. [21]. They have used semantic segmentation concept along with deep architectures.

Zhao et al. [1] have used GAN based text detector for camera-captured document images. To learn the high variations between the lines and characters data, conditional GANs have been used as an image-to-image generation. To detect the candidate text, directional text map scores have been generated corresponding to the text. It can detect the characters written in various directions using the printing direction information of text. Text line extraction has been performed by Kundu et al. [17] using conditional GANs for Chinese text dataset and found promising results. Yanagi et al. have used GAN architecture for scene retrieval in video images with the help of text to image generation capability of GANs [22]. Apart from this, Alonso et al. [12] have used GANs to generate the synthetic handwritten word images to meet the need of training deep networks. The bidirectional Long Short Term Memory with CTC loss has been used in GAN architecture to get the embedding of word. GAN architecture has been trained on both adversarial loss and CTC loss. The proposed network efficiently generates the word images of Arabic and French data. Cai et al. [13] have explored the power of GAN networks as Chinese handwritten recognition system. The basic structure of GAN uses the convolutional neural network based encoder–decoder architecture. Further to incorporate the edge and skeleton information of characters, the weighted mean squared error loss function has been adopted in generator architecture. Jha et al. [14] have explored GANs in generating handwritten digits dataset of 4 Indian scripts. The motive was to enhance the available dataset of handwritten digits for the use of machine learning models. They had proved that the addition of generated data with real data improves the accuracy of classification process. Kong et al. [16] have proposed GARN, i.e., GAN based handwritten character recognition. They had modified the discriminator part by making it multinomial classifier for character recognition. For various GAN architectures, Cheng et al. [11] have performed an in-depth analysis on MNIST dataset. The comparison has been made on the basis of various parameters like architecture, training method, learning type, benefits, and performance metrics. Further to evaluate the different types of GAN models, the quality of generated images, classification accuracy, generator and discriminator loss, and computational time have been considered.

2.1 Proposed Approach

GAN models have two parts, generator and discriminator, where generator generates the new plausible synthetic images and discriminator classifies it into real or synthetic image. The conditional GANs generate the images based on the condition implemented on generated data. In conditional GAN networks, we have Pix2Pix model for image-to-image translation. Pix2Pix GANs are widely adopted for many image translation tasks like Google maps to satellite images, etc. Fig. 1 depicts the architecture of pix2pix model for image translation.

In pix2pix network, deep convolutional neural networks act like discriminator which takes source (image generated by generator) and target (real image) image. It uses the PatchGAN based architecture where the output of model is corresponding to some number of pixels (patch) of input image. With this, we can train the model with images of any size as it works on patches of images. The generator in pix2pix network has encoder–decode structure and is based on U-Net architecture. In U-Net’s structure, we have many skip connections between encoding and decoding layers resulting into U-shaped architecture as shown in Fig. 2. The generator has task to down-sample the input data using encoder layers and then up-sample the data using decoder layer to form the shape of output image (target image). The encoder and decoder blocks have convolution, batch normalization, dropout, and activation layers.

Image translation is the task of converting one type of images into another type in a controlled and specific manner, for example, converting Google maps from satellite images and photograph of landscape from day to night, etc. It requires specialized models and handcrafted loss functions. Hence Pix2Pix network gives a general purpose network and loss function for image translation. The generator model in pix2pix is U-Net which takes input images as input rather than using random points from latent space. The working of U-Net is similar to ResNets as the output of earlier layers is combined with later layers. Moreover, the skip connections in U-Net does not require any kind of resizing or projections. In image-to-image translation, the difference between resulting image generated from input image passed to parametric function and the ground truth image is called conditional adversarial loss represented by Eq. 1.

$$L_{cGAN}(G, D) = E_{x,y}[\log D(x, y)] + E_{x,z}[\log(1 - D(x, G(x, z)))] \quad (1)$$

$$L_{L1}(G) = E_{x,y,z}[|y - G(x, z)|] \quad (2)$$

Different combinations of loss functions can also be used. In our work, we have trained generator using combination of adversarial and L1 (Eq. 2) loss which results into more clear generation of images. The combined loss is represented by

$$G^* = \operatorname{argmin}_G \max_D L_{cGAN}(G, D) \quad (3)$$

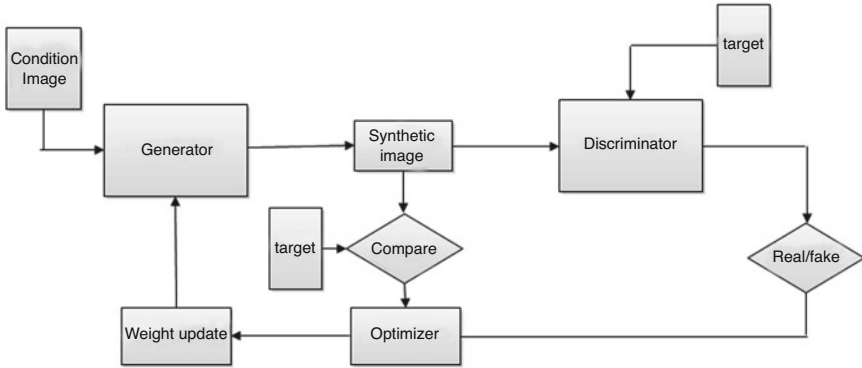


Fig. 1 Pix2Pix model for image translation

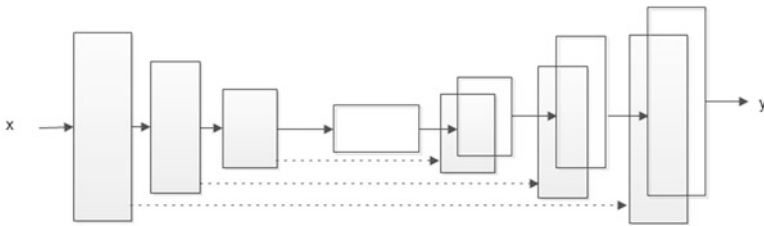


Fig. 2 U-Net with skip connections for image translation

3 Experimental Results and Discussion

In this section, we will discuss the database used for image-to-image translation task and training of pix2pix GAN architecture. The task considered in this research is text line separation of bilingual handwritten documents. We are considering the handwritten documents written in Gurumukhi-Latin scripts with special characters and alphanumeric. We have collected 150 document images from academic domain written by various writers of varying age groups. The document images are scanned and converted to binary using Otsu’s method before inserting into GAN architecture. Training deep neural networks requires the ground truth information corresponding to input data. Hence, we generated the paired ground truth information corresponding to each input image. We have manually separated the text lines in documents of training set with red color boundaries for ground truth generation.

3.1 Training Network

The images are resized to 256×256 for pix2pix GAN data and stored in two different folders, i.e., training and ground truth data. The discriminator of network is a deep convolutional network with 5 convolutional layers which implements 70×70 PatchGAN discriminator architecture along with batch normalization. To train discriminator, binary cross entropy loss function has been implemented with loss weight of 0.5. This helps in slowing down the changes in discriminator as compared to generator model. To optimize the network, Adam optimizer with 0.0002 has been used. In generator, we have a sequence of encoders and decoders which are block of convolution, batch normalization, and activation. In each encoder, we used Lekey Relu activation function along with convolutional layer. Similarly, decoder uses transposed convolutional layers along with dropout layer having Relu activation function. The complete network has been trained for 100 epochs.

3.2 Results

We experimented our proposed method with bilingual dataset of 150 document images. To train GAN networks, this amount of dataset was not sufficient. Hence, by manually segmenting the 150 document images, we have generated 650 document images (samples are shown in Fig. 3). Results of generated images corresponding to ground truth images (Fig. 4) are shown in Fig. 5 for pix2pix GAN architecture. We have considered four performance metrics, i.e., detection rate (DR), recognition accuracy (RA), error rate(ER), and F-measures (FM). Table 1 shows the error rate, detection rate, recognition accuracy, and F1-measure. The detection rate is the ratio of one-to-one mapping of text lines corresponding to text lines in ground truth images (G). Similarly, recognition accuracy is defined as the ratio of one-to-one mapping of text lines in detected resultant (R) as represented by Eq. 4. To compare the trained models, we have used simple encoder–decoder generator without skip connections.

$$DR = m2m \div G; RA = m2m \div R; ER = 1 - RA; FM = 2DR.RA \div (DR + RA) \quad (4)$$

Table 1 Results for image translation

Model	DR	RA	ER	FM
Pix2Pix3	84.62%	88%	2%	86.27%
Encoder–decoder	69.2%	72.2%	27.8	70.81%

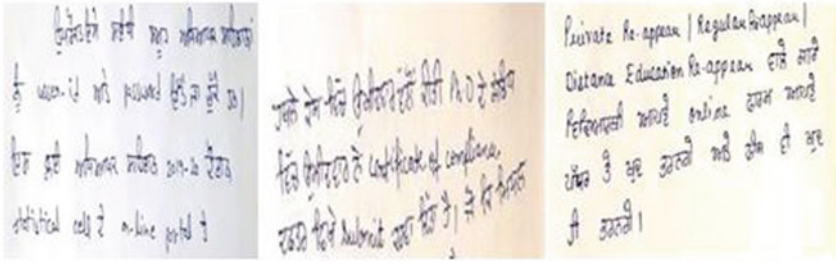


Fig. 3 Samples for image translation

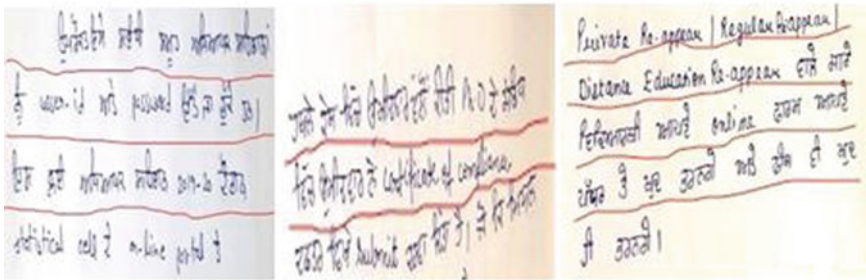


Fig. 4 Ground truth for image translation

4 Conclusion and Future Work

In this work, we have explored the text line extraction task in bilingual documents with the help of image-to-image translation methods. The GAN model used in this for generating the images with text line separation has shown effective results. The F-measure of 86.27% has been reported for our bilingual dataset of Gurumukhi-English script. The high training time required to train the GAN model for image translation demands high computational power systems. The accuracy of network is dependent on the hyper parameters used in training of network. This is the first kind of work in Gurumukhi of using GAN models for image translation. In future, the results can be improved using high number of training epochs and more optimization methods.

Acknowledgments This research was supported by Council of Scientific and Industrial Research (CSIR) funded by the Ministry of Science and Technology (09/677(0031)/2018/EMR-I) as well as the Government of India.

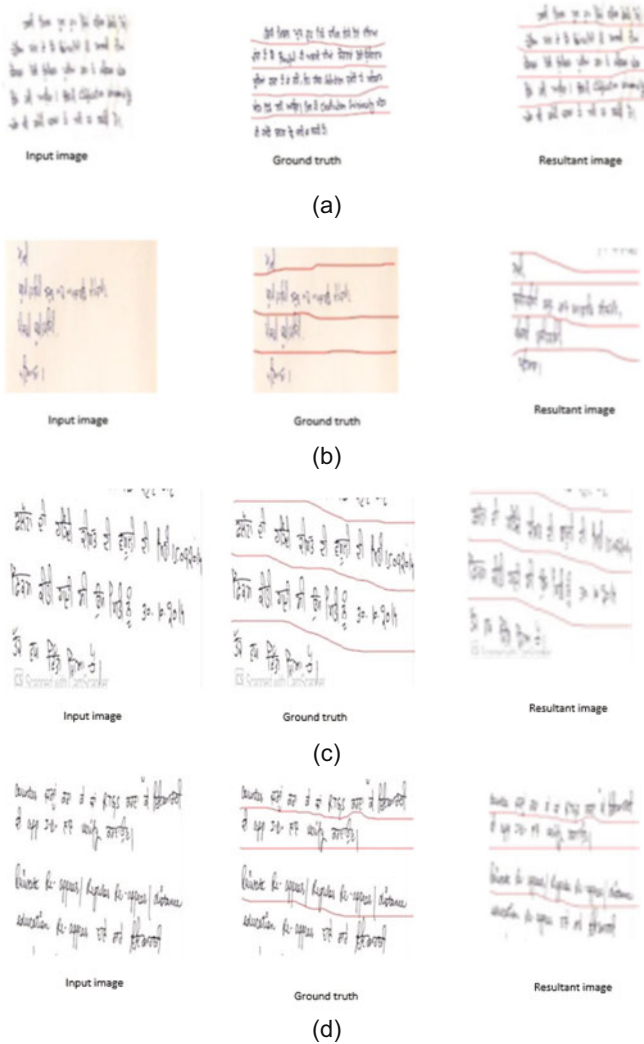


Fig. 5 Image translation results: (a) for sample 1, (b) for sample 2, (c) for sample 3, and (d) for sample 4

References

1. Jo, Junho, et al. "Handwritten Text Segmentation via End-to-End Learning of Convolutional Neural Networks." *Multimedia Tools and Applications* 79.43 (2020): 32137–32150.
2. dos Santos, Rodolfo P., et al. "Text line segmentation based on morphology and histogram projection." *2009 10th International Conference on Document Analysis and Recognition*. IEEE, 2009.

3. Susan, Seba, and KM Rachna Devi. "Text area segmentation from document images by novel adaptive thresholding and template matching using texture cues." *Pattern Anal. Appl.* 23.2 (2020): 869–881.
4. Pal, U., and Sagarika Datta. "Segmentation of Bangla unconstrained handwritten text." *Seventh International Conference on Document Analysis and Recognition*, 2003. Proceedings. Vol. 3. IEEE Computer Society, 2003.
5. Jindal, Payal, and Balkrishan Jindal. "Line and word segmentation of handwritten text documents written in Gurmukhi script using mid point detection technique." *2015 2nd International Conference on Recent Advances in Engineering & Computational Sciences (RAECS)*. IEEE, 2015.
6. Sharma, Dharam Veer, and Gurpreet Singh Lehal. "An iterative algorithm for segmentation of isolated handwritten words in Gurmukhi script." *18th International Conference on Pattern Recognition (ICPR'06)*. Vol. 2. IEEE, 2006.
7. Sanasam, Inunganbi, Prakash Choudhary, and Khumanthem Manglem Singh. "Line and word segmentation of handwritten text document by mid-point detection and gap trailing." *Multimedia Tools and Applications* 79.41 (2020): 30135–30150.
8. Ptak, Roman, Bartosz Żygadło, and Olgierd Unold. "Projection-based text line segmentation with a variable threshold." *International Journal of Applied Mathematics and Computer Science* 27.1 (2017): 195–206.
9. Jindal, Simpel, and Gurpreet Singh Lehal. "Line segmentation of handwritten Gurmukhi manuscripts." *Proceeding of the workshop on document analysis and recognition*. 2012.
10. Mohammad, Khader, et al. "An adaptive text-line extraction algorithm for printed Arabic documents with diacritics." *Multimedia Tools and Applications* 80.2 (2021): 2177–2204.
11. Cheng, Keyang, et al. "An analysis of generative adversarial networks and variants for image synthesis on MNIST dataset." *Multimedia Tools and Applications* 79.19 (2020): 13725–13752.
12. Alonso, Eloi, Bastien Moysset, and Ronaldo Messina. "Adversarial generation of handwritten text images conditioned on sequences." *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019.
13. Cai, Junyang, et al. "TH-GAN: Generative adversarial network based transfer learning for historical Chinese character recognition." *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019.
14. Jha, Ganesh, and Hubert Cecotti. "Data augmentation for handwritten digit recognition using generative adversarial networks." *Multimedia Tools and Applications* 79.47 (2020): 35055–35068.
15. Majid, Nishatul, and Elisa H. Barney Smith. "Segmentation-free Bangla offline handwriting recognition using sequential detection of characters and diacritics with a faster R-CNN." *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019.
16. Kong, Hao, et al. "GARN: A novel generative adversarial recognition network for end-to-end scene character recognition." *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019.
17. Kundu, Soumyadeep, et al. "Text-line extraction from handwritten document images using GAN." *Expert Systems with Applications* 140 (2020): 112916.
18. Kaur, Sukhandeep, Seema Bawa, and Ravinder Kumar. "A survey of mono-and multi-lingual character recognition using deep and shallow architectures: Indic and non-Indic scripts." *Artificial Intelligence Review* 53.3 (2020): 1813–1872.
19. Goodfellow, Ian, et al. "Generative adversarial nets." *Advances in neural information processing systems* 27 (2014).
20. Zhu, Yixing, and Jun Du. "TextMountain: Accurate scene text detection via instance segmentation." *Pattern Recognition* 110 (2021): 107336.
21. Dutta, Arpita, et al. "Segmentation of text lines using multi-scale CNN from warped printed and handwritten document images." *International Journal on Document Analysis and Recognition (IJ DAR)* 24.4 (2021): 299–313.
22. Yanagi, Rintaro, et al. "Query is GAN: scene retrieval with attentional text-to-image generative adversarial network." *IEEE Access* 7 (2019): 153183–153193.

Performance Comparison of YOLO Variants for Object Detection in Drone-Based Imagery



Manpreet Kaur  and Padmavati Khandnor

Keywords Object Detection · Drone images · YOLO · VisDrone · Darknet

1 Introduction

Computer vision is an interdisciplinary field that has become quite important in recent years due to its various applications in the areas like surveillance, transportation, agriculture, smart city and health care. Object detection and object tracking are two major problems in computer vision that are being extensively researched by numerous researchers. This research has resulted in many benchmarks like Caltech, KITTI, MS COCO and ImageNet for object detection, and OT, UA-DETRAC and MOTChallenge for object tracking [1].

With the development of technology, drones with cameras are being used in various areas like surveillance, disaster management, search and rescue operations, aerial photography and fast delivery services. These applications require that the drone systems must have the capability to perceive the surroundings, parse the scene, and react accordingly, with scene parsing being the most important task. Different levels of scene parsing capabilities are required for various drone applications, along with recognizing objects and their locations and determining the bounding boxes for each object [11]. Accordingly, the task of scene parsing can be divided into three subparts, i.e. image classification, object detection and semantic segmentation. Out of these, object detection is being mostly used in drones for scene parsing activities. Despite extensive study and research in computer vision and the development of a number of object detectors, these algorithms are not best suitable for drone-based images [1]. Detecting objects through drones is a challenging task because of [10, 12] (1) small target objects as compared to background images, (2)

M. Kaur (✉) · P. Khandnor

Department of Computer Science and Engineering, Punjab Engineering College (Deemed to be University), Chandigarh, India

e-mail: manpreetkaur.mtcse20@pec.edu.in; padmavati@pec.edu.in

sparse and non-uniform distribution of the objects, (3) constant change in height and angle of the camera, (4) weather and lighting conditions, (5) rapid movement of camera, (6) large viewpoint and scale changes, (7) relative movement between the camera and object, (8) occlusion, (9) type of drones and cameras used and (10) limited availability of large and open datasets. However, in recent years this under-researched problem has now started receiving more and more attention in the research community, and a lot of new real-time object detection algorithms and architectures are being developed.

2 Related Work

The various object detection methods can be mainly classified into the following two categories:

Two-Stage Detectors The object detection is done in two stages in these algorithms and it is treated as a classification problem. These detectors mainly comprise the backbone network, region proposal module and detection header as their components [11]. In the first stage, the region proposal module creates a set of image regions with a high likelihood of containing an object. In the second stage, using these regions as input, detection headers perform the final detection and classification of objects [4]. The most used algorithms in this category include FPN [24], R-CNN [15], Fast R-CNN [13], Faster R-CNN [14] and R-FCN [16]. These detectors generally have a good accuracy compared to single-stage detectors. However, region proposal modules require a lot of computation and run-time memory, because of which detection speed is slow in these algorithms.

Single-Stage Detectors In these algorithms, object detection is performed in a single pass and it is considered as a regression problem. Instead of requiring extra branch networks as in two-stage detectors, these algorithms use predefined anchors that cover different scales, aspect ratios and spatial positions across an image. These detectors perform the task of classification and feature extraction in the single-pass itself, because of which these are fast as compared to the two-stage detectors. The most common algorithms in this category include You Only Look Once (YOLO) series models [5–7], RetinaNet [17] and SSD [18]. Out of these, YOLO series models, i.e. YOLOv1-YOLOv5, have emerged as the fastest object detection algorithms and are still comparable to the two-stage detectors, making them suitable for real-time applications. Among the various YOLO models, YOLOv3 [5], YOLOv4 [6] and YOLOv5 [7] are most commonly used these days because of their generalization to a variety of datasets, high speed and accuracy. As published in [5–7], the performance of the YOLO algorithms for object detection has been carried out on the MS COCO benchmark dataset.

With reference to the VisDroneDet-2020 challenge results [2], the highest accuracy detectors are based on ensemble methods. Several top performance detectors

submitted in the challenge have used the test dataset to increase the accuracy. The highest accuracy achieved on the VisDrone dataset is 34.57, and 58.21 with mAP 0.5 on test-challenge dataset. With reference to VisDroneDet-2019 challenge results [3], the submitted detectors with the highest accuracy are based on FPN, cascade R-CNN and attention modules. However, it has been concluded that the performance of all the submitted detectors is not satisfactory for real-time applications [2, 3]. Authors in [22] have proposed an improved version of the YOLOv4 for efficient detection of small objects by adding upsampling layers and compared its performance with the original YOLOv4 on VisDrone Dataset. Authors in [11] have proposed Slim-YOLO with a smaller number of trainable parameters as compared to YOLOv3 and have compared its performance with YOLOv3-tiny and YOLOv3 on VisDrone2018-Det benchmark dataset. Authors in [8] have proposed SPB-YOLO based on YOLOv5 for real-time object detection of small and dense objects. They have proposed the use of the Strip Bottleneck (SPB) Module for better understanding of the scale of images and added one more detection head in YOLOv5 to handle the detection of dense objects. The proposed model has been compared with slim YOLO [11], PG YOLO [29], YOLOv5 [7] and other real-time object detectors using the VisDrone dataset.

3 Experiments and Implementation

3.1 Dataset

In this work, we have used the VisDrone2019 benchmark dataset. It is an object detection and tracking benchmark dataset collected and published by the AISKY-EYE team at Tianjin University's Machine Learning laboratory, China. The dataset is a rich repository of 10,209 static images and 288 videos with rich annotations, bounding boxes, truncation ratios and occlusion, captured using different types of cameras mounted on drones. The images have been divided into three categories with 6471 for training, 548 for validation and 3190 for testing. It is the biggest and most complex dataset published till date for drone platforms [25]. It has been collected across 14 different cities of China. It has both sparse and crowded images with different lighting and weather conditions, which makes the detection of the smaller objects even harder. The dataset has ten different classes, i.e. pedestrian, person, car, van, bus, truck, motor, bicycle, awning-tricycle and tricycle.

3.2 Models

In this paper, we have implemented and compared the performance of the various YOLO algorithm variants for the drone-based benchmark VisDrone2019-Det dataset.

YOLO is a single-stage real-time object detector in which a single neural network is used to predict the class probabilities and bounding boxes from the complete image in a single pass itself. Thus, it performs the task of classification of objects and feature extraction in a single pass, because of which it is fast as compared to the other two-stage detectors. In YOLO, the given image is divided into $S \times S$ grid and the grid in which the centre of the object falls is said to have detected the object. Each grid predicts B bounding boxes that indicate the object location and confidence score. Different versions of the YOLO algorithm predict different number of bounding boxes based on their capabilities. However, in each version, the bounding box consists of the five predictions, i.e. (b_x, b_y, b_w, b_h) and a box confidence score, where (b_x, b_y) indicates the centre of the bounding box w.r.t the grid cell, (b_w, b_h) indicates the width and height of the bounding box w.r.t the image. The confidence score indicates the confidence of the model prediction that the box contains an object and its accuracy w.r.t the ground truth and is calculated by the following equation:

$$\text{Confidence (c)} = P_c(\text{Obj}) * \text{IOU}_{\text{pred}}^{\text{truth}}, P_c(\text{Obj}) \in \{0, 1\} \quad (1)$$

where $P_c(\text{Obj})$ is the object probability and IOU is the intersection over union of the algorithm's predicted box and the ground truth box and it indicates the similarity between them. The concept of non-max suppression (NMS) has been used to handle multiple detections by different bounding boxes of the same object. Further, the idea of anchor boxes has been added from YOLOv3 onwards to detect the different shape objects by the same bounding box.

In this work, we have compared the performance of nine different models: YOLOv3, YOLOv4-CSP, three variants of scaled YOLOv4, namely YOLOv4-tiny, YOLOv4-P5, YOLOv4-P6 and four variants of YOLOv5, namely, YOLOv5s, YOLOv5m, YOLOv5l and YOLOv5x. All these models have been pre-trained on MS COCO [20] benchmark. These pre-trained weights are available on GitHub, which can be used to train the other models. Below is a brief description of the selected models.

YOLOv3 This is a variant of the original YOLO model [21]. It uses a larger network as compared to the original model to perform the feature extraction. It is based on the Darknet-53 backbone, which has newly added residual blocks with skip connections to avoid the network saturation at very deep layers during training. To improve the accuracy and have better detections for small objects, YOLOv3 can detect objects at three distinct scales, i.e. 13×13 , 26×26 and 52×52 , compared to its previous counterparts, which were able to detect single-scale objects.

YOLOv4 A new architecture has been used in YOLOv4 [6]. It is based on CSPDarkNet53 [19] as a backbone, which is a hybrid version of cross-stage partial network (CSPNet) and DarkNet-53 of YOLOv3. The addition of CSP in the backbone has increased the learning capability of the model. To increase the receptive field without having any effect on the velocity, the neck of the model has

been augmented with spatial pyramid pooling (SPP) and path aggregation network (PAN). The class subnet and box subnet have been used in the head, as in the case of YOLOv3. Apart from this novel architecture, authors have given “Bag of Freebies”, which includes Mosaic, CutMix and DropBlock regularization and “Bag of Specials” with SAM-block, cross-stage partial connections (CSP), Mish activation, etc., which increases the detection accuracy.

Scaled YOLOv4 This model uses the YOLOv4-CSP as the base model and adds the concept of network scaling in terms of depth, width, network structure and its resolution to attain state-of-the-art accuracy while maintaining the optimal speed [23]. This model can scale up and down for small and large networks to provide YOLOv4-tiny and YOLOv4-large. YOLOv4-tiny has been designed for systems that have computational, memory and bandwidth constraints. To handle these constraints, a new concept of one shot aggregation (OSA) has been added, which ensures the aggregation of features only at the last feature map instead of aggregating them at each stage. YOLOv4-large has been designed to improve accuracy and is for high-end GPU devices or cloud-based GPUs. To improve the accuracy, firstly it scales up the input size and number of stages and then dynamically changes the width and depth to meet real-time inference speed requirements. It has three different variants, i.e. YOLOv4-P5, YOLOv4-P6 and YOLOv4-P7, with different computational capability requirements. Usually, all these models require multiple GPUs to train them. The network structure of these models is shown in Fig. 1.

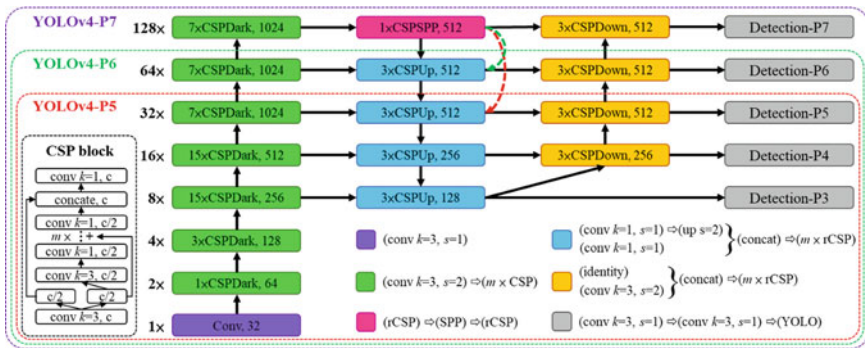


Fig. 1 Network architecture of scaled YOLOv4 [23]

YOLOv5 It is the latest release in the YOLO series. It is the first of the YOLO models which has been written in Pytorch as compared to Darknet. It is very lightweight and easy to use. However, not many architectural changes have been done in this version compared to the previous Version. YOLOv5 has four variants, namely YOLOv5s, YOLOv5m, YOLOv5l and YOLOv5x. The first one is the smallest and least accurate and the last one is the largest with maximum accuracy.

All these variants do not differ in terms of the number of layers compared to the previous YOLO versions. However, they differ in scaling multipliers of width and depth of the network.

3.3 Evaluation Metrics

The following metrics have been used for the evaluation of all the models:

Precision It is defined as the ratio of correct positive predictions to all the positive predictions made by the model and is given by the following equation:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (2)$$

Recall It is the ratio of correct positive predictions to the total number of relevant objects and is given by the following equation:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (3)$$

where TP is true positive, FP is false positive and FN is false negative.

F1-Score It is the harmonic mean of precision and recall and is calculated using the following equation:

$$\text{F1} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

Mean Average Precision (mAP 0.5) calculated at single intersection over union (IOU) threshold of 0.5 over all categories.

Average Precision (mAP 0.5:0.95) calculated by averaging over all 10 intersection over union thresholds (in the range [0.50: 0.95] with the uniform step size 0.05) of all categories.

3.4 Experiments on Benchmark

Implementation and System Architecture

All the models have been implemented using python. We developed the deep learning environment based on Darknet, CUDA-11.2 and cuDNN-8.1.1 to train YOLOv3, Yolov4-CSP and YOLOv4-tiny. All variants of YOLOv5, scaled

YOLOv4-P5 and scaled YOLOv4-P6 have been implemented using Pytorch version 1.9.0. All the models have been trained using Tesla V100-DGXS, whose configuration is given in Table 1. Multi GPU training has been done for scaled YOLOv4-P5 and scaled YOLOv4-P6, where two cores of the DGXS were used to train these models. However, all the other models have been trained on a single core of the GPU.

Table 1 Specifications of training system

Parameter	Configuration
CPU	Intel Xeon E5-2698 v4 2.2GHz (20 Cores)
CPU Memory	256 GB DDR4
GPU	4 × Tesla V100
GPU Memory / Core	32 GB
SDD	3 × 1.92 TB RAID
OS	Ubuntu 20.04

Training of Models

Following the concept of transfer learning, we used the pre-trained weights of MS COCO [20] to train all the YOLO models [9]. YOLOv3, YOLOv4-CSP and YOLOv4-tiny have been trained for 20,000 maximum batches, with a batch size of 64. The maximum batches have been set based on the number of classes in the VisDrone2019-Det dataset. The HSB (hue, saturation and brightness) parameters of the images have been augmented to have images with different colours and brightness during training. The initial learning rate has been used as 0.0013 and a momentum of 0.949 has been used for gradient calculation. All variants of YOLOv5, YOLOv4-P5 and YOLOv4-P6 have been trained for 300 epochs, where the initial 3 epochs were used as warm-up epochs with a learning rate 0.1 and a momentum of 0.8. The VisDrone dataset has images with different resolutions, namely, 960×540 , 1920×1080 and 2000×1500 . But for the current implementation and training, input image size has been fixed to 416×416 for all the models except YOLOv4-P5 and YOLOv4-P6. The input image size of 896×896 and 1280×1280 have been used for YOLOv4-P5 and YOLOv4-P6 respectively.

4 Results and Discussions

4.1 Quantitative Evaluation

Performance Comparison Based on Accuracy

The mAP is normally used to measure the accuracy in object detection problems. The performance comparison of all models has been shown in Table 2. Figure 2

depicts the comparison of mAP for IOU 0.5 and mAP for IOU 0.5:0.95 for various models on the validation data. The class-wise detection performance of the various models has been summarized in Table 3.

As shown in Table 2 and Fig. 2, scaled YOLOv4-P6 has achieved the best results among all the tested YOLO detectors, with mAP of 59.5 at 0.5 IOU and Tiny-YOLOv4 has achieved the least accuracy of 14.6 at 0.5 IOU. The number of instances for each class in Table 3 indicates that the dataset is highly imbalanced. The effect of class imbalance can also be observed from the class-wise performance in Table 3, which indicates that the object classes with a higher number of instances like car and van have more accuracy than the classes with fewer instances like awning-tricycle.

Performance Comparison Based on Model Complexity

Table 2 Comparison of various YOLO models on validation set of VisDrone2019 dataset

Model	Backbone	Input image size	Precision	Recall	F1-Score	mAP ^{val} 0.5	mAP ^{val} 0.5:0.95
YOLOv3	Darknet-53	416	52.0	50.0	51.0	33.88	17.3
YOLOv4-CSP	CSPDarknet-53	416	57.0	54.0	55.0	40.23	22.10
YOLOv4-Tiny	CSPDarknet-53-tiny	416	75.0	16.0	27.0	14.06	7.13
YOLOv5s	CSPDarknet-Bottleneck	416	36.6	23.6	28.7	22.27	11.22
YOLOv5m	Bottleneck	416	39.5	26.7	31.9	24.42	12.97
YOLOv5l		416	42.7	31.3	36.1	29.30	16.36
YOLOv5x		416	51.6	30.2	38.1	31.66	17.46
Scaled YOLOv4-P5	CSPDarknet-53	896	35.8	64.2	46.0	51.4	31.5
Scaled YOLOv4-P6		1280	43.7	70.7	54.0	59.5	37.2

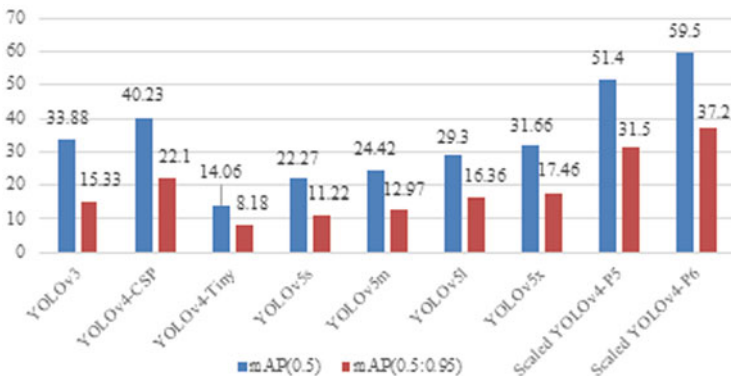


Fig. 2 Performance comparison of various YOLO algorithms on VisDrone dataset

Table 3 Class-wise comparison of mAP (at IOU 0.5) of different models

Class	Instances	YOLOv3	YOLOv4-CSP	YOLOv4-Tiny	YOLOv5s	YOLOv5m	YOLOv5l
Pedestrian	8840	30.51	37.38	4.86	25.5	27.2	32.3
People	5120	29.07	30.75	2.22	20.2	23.2	26.6
Bicycle	1290	11.88	16.72	1.70	4.7	5.80	8.1
Car	14,100	71.96	76.91	39.45	61.4	64.0	67.0
Van	1980	41.07	46.42	23.17	21.0	24.2	30.9
Truck	750	31.32	40.76	19.13	21.3	20.5	28.0
tricycle	1040	24.63	32.05	10.20	10.2	12.7	17.0
Awning-tricycle	532	11.55	16.30	5.47	4.70	6.20	9.9
Bus	251	50.33	58.23	27.81	30.6	33.9	41.8
Motor	4890	36.50	46.79	6.64	23.1	26.7	31.2
Overall	38,800	33.88	40.23	14.06	22.3	24.4	29.3

Table 4 Comparison of various models based on complexity

Model	mAP ^{val} (0.5)	Parameters (million)	Weight file (MB)
YOLOv3	33.88	61.6	235.1
YOLOv4-CSP	40.23	64.0	244.3
YOLOv4-Tiny	14.06	5.89	22.5
YOLOv5s	22.27	7.09	13.7
YOLOv5m	24.42	21.09	42.5
YOLOv5l	29.30	46.68	89.4
YOLOv5x	31.66	87.30	167.0
Scaled YOLOv4-P5	51.4	70.33	269.2
Scaled YOLOv4-P6	59.5	126.80	484.9

The complexity of the model is used to measure the resources and computational power required by the model during training and inference. The comparison of various YOLO models based on the complexity in terms of parameters and weight file has been summarized in Table 4 and Fig. 3.

As shown in Table 4 and Fig. 3, scaled YOLOv4-P6 has the maximum accuracy. However, it is also the most complex model of all the tested YOLO models with the largest number of trainable parameters and weight file size. So scaled YOLOv4-P6 can be used for high-end drone systems or systems that can offload the complex computations and detection tasks to the remote cloud, and can perform the low-level object detection and navigation tasks with onboard drone hardware. Further, for secure communication between drone and remote cloud, various security mechanisms discussed in [26–28] can be implemented. After the scaled YOLOv4 versions, YOLOv4-CSP has achieved a considerable accuracy of 40.23 with half of the weight file size compared to scaled YOLOv4-P6, making it suitable for the drone systems with medium computational and memory devices. Moreover, compared to the proposed YOLOv4 model in [22], our YOLOv4-CSP model has



Fig. 3 Comparison of various YOLO models based on parameters and weight file size

better performance for the same input image size of 416×416 . The experiments show that YOLOv4-Tiny has the least number of parameters and YOLOv5s has the smallest weight file. Hence, these models can be used for low-end resource-constrained drone systems. Compared to Tiny-YOLOv3 [11], Tiny-YOLOv4 in our experiments has given an accuracy better by 4.96. So based on experiments and results, we suggest the use of Tiny-YOLOv4 as compared to Tiny-YOLOv3 for drone systems with low-end hardware.

The performance characteristics, i.e. model loss during training and validation, accuracy, various metrics like precision, recall, mAP 0.5 and mAP 0.5:0.95 for the various models during training and validation have been depicted in Figs. 4, 5, and 6.

4.2 Qualitative Evaluation

The visualized results of the detected objects using the scaled YOLOv4-P6 model along with their class and confidence score have been shown in Fig. 7. Different colours have been used for the bounding box of each class.

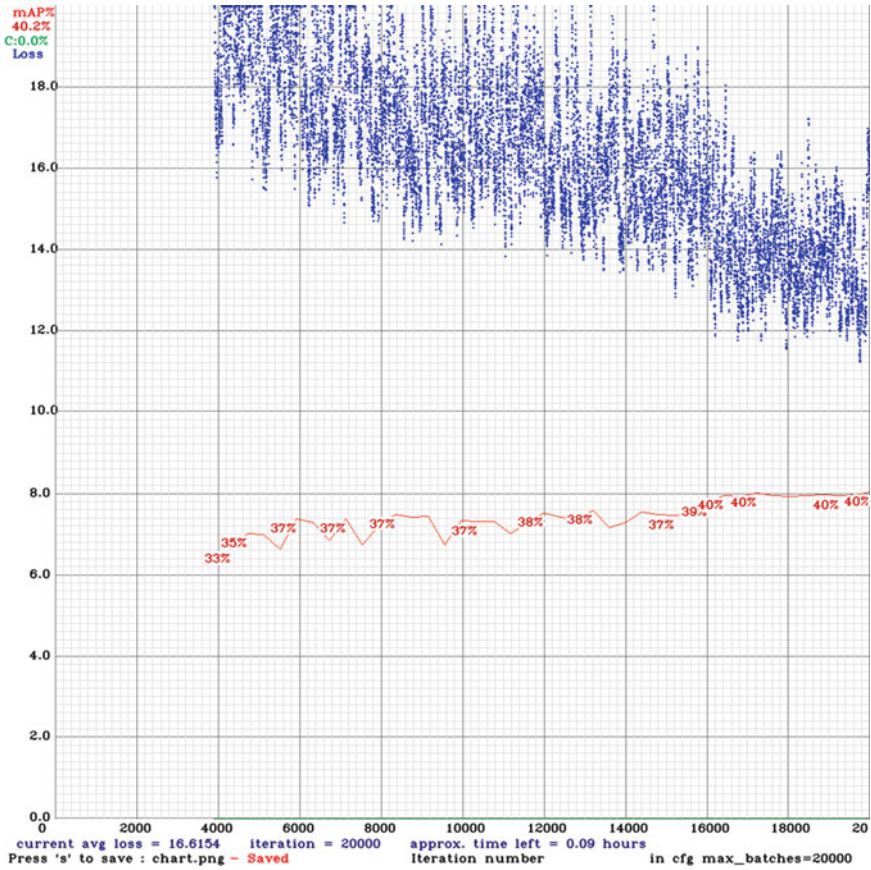


Fig. 4 Performance characteristics of YOLOv4 model

5 Conclusion

In this paper, we have implemented and compared the performance of different versions of YOLOv3, YOLOv4 and YOLOv5 models for real-time object detection in drone-based images using the VisDrone2019-DET benchmark dataset. We have considered the different variants of YOLO, suitable for each class of drones, i.e. systems with low-end onboard hardware being used in day-to-day applications and the systems with high-end onboard computational power and resources being used in defence applications. To the best of our knowledge, this is the first attempt in the literature, where such a wide range of YOLO variants have been tested on the VisDrone dataset. We conclude our study and work with the proposal of scaled YOLOv4 versions for the high-end drone systems and tiny YOLOv4 or YOLOv5s

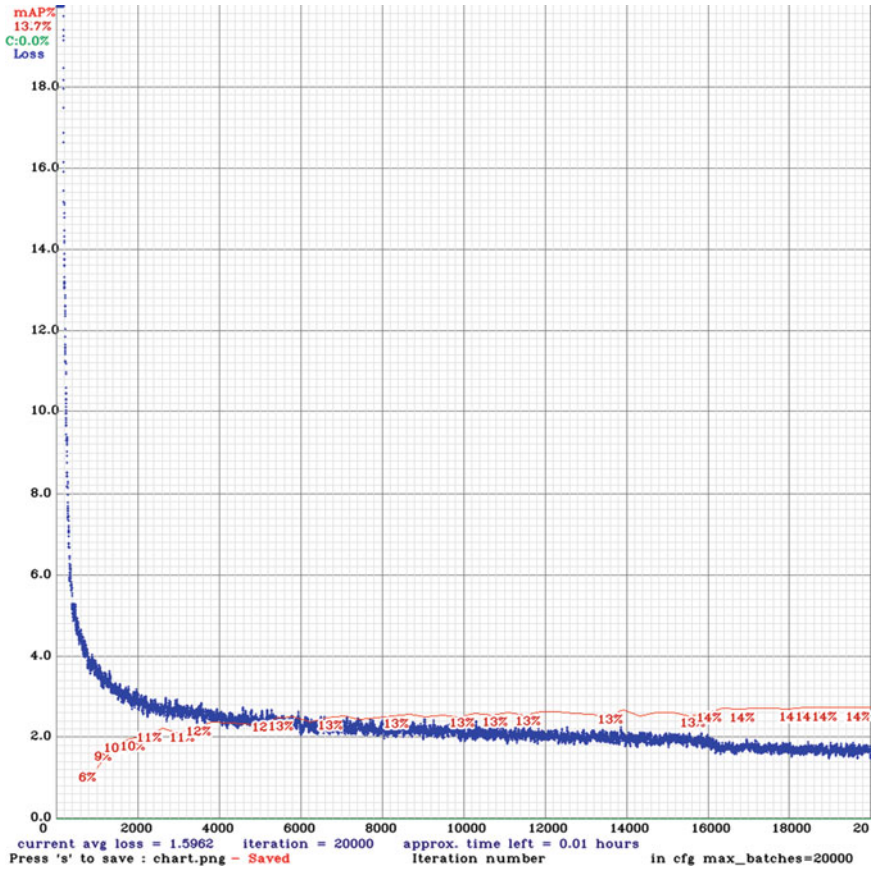


Fig. 5 Performance characteristics of Tiny-YOLOv4 model

for the low-cost and low-end drone systems. For moderate performance, intermediate versions, i.e. YOLOv3, YOLOv4 and YOLOv5x can be used depending on the available onboard computational capabilities. However, in the present work, we have not considered the detection speed of the various models and these have been compared based on the prediction accuracy and model complexity. Moreover, in the present work, all models except YOLOv4-P5 and YOLOv4-P6 have been trained on 416×416 input image size. The resolution of the input image may be increased further to improve the model accuracy.

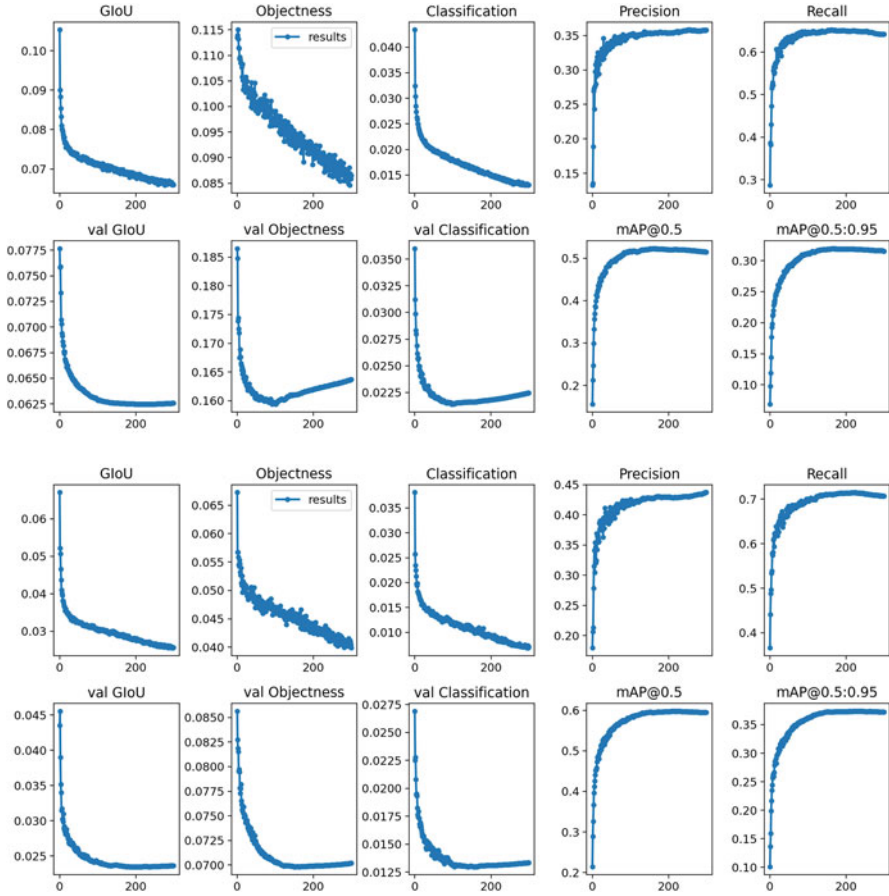


Fig. 6 Performance characteristics of scaled YOLOv4-P5 and scaled YOLOv4-P6 models

6 Future Work

VisDrone benchmark dataset is highly complex and challenging, which has a class imbalance problem. We have not handled the category imbalance problem in this present work. The effect of the class imbalance can be seen from Table 3, where we have achieved higher accuracy for the dominant classes like car and pedestrian as compared to classes with lower instances like awning-tricycle and bicycle. Moreover, this dataset has a large number of small and occluded objects, making their detection extremely challenging and hence reducing the overall accuracy of the model. In the current implementation, the occlusion parameter given in the dataset has not been used to handle the occluded objects. Approaches for solving the class imbalance and handling of small and occluded objects have been left for the future work.



Fig. 7 Visual results of object detection on test-set images of VisDrone (above) Actual image from test set (below) Output image with bounding boxes and confidence score for the detected objects



References

1. P. Zhu, L. Wen, X. Bian, L. Haibin, Q. Hu: Vision meets drones: a challenge. arXiv Prepr, arXiv1804.07437 (2018) 1–11.
2. Zhu, P., Wen, L., Du, D., Bian, X., Hu, Q., Ling, H (2020) Vision meets drones: past, present and future. arXiv:2001.06303
3. D. Du *et al.*: VisDrone-DET2019: The Vision Meets Drone Object Detection in Image Challenge Results. 2019 IEEE/CVF *International Conference on Computer Vision Workshop (ICCVW)*, 2019, pp. 213–226, <https://doi.org/10.1109/ICCVW.2019.00030>.
4. P. Zhu, L. Wen, D. Du, X. Bian, H. Ling, Q. Hu, and et al.: Visdrone-det2018: The vision meets drone object detection in image challenge results. In ECCVW, pages 437–468, 2018.
5. J. Redmon and A. Farhadi: YOLOv3: An incremental improvement. arXiv (2018).
6. A. Bochkovskiy, C. Y. Wang and H. Y. M. Liao: YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv, 2020.
7. Glenn-jocher, Alexstoken, and Borda: YOLOv5. May 2020. <https://github.com/ultralytics/yolov5>
8. X. Wang, W. Li, W. Guo and K. Cao: SPB-YOLO: An Efficient Real-Time Detector For Unmanned Aerial Vehicle Images. 2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC), pp. 099–104 (2021), <https://doi.org/10.1109/ICAIIIC51459.2021.9415214>.
9. Gishyan, K. M.: Improving UAV Object Detection through Image Augmentation. *Mathematical Problems of Computer Science*, 54, 53–68 (2020). 10.51408/1963-0059
10. Zhang, W.; Liu, C.; Chang, F.; Song, Y.: Multi-Scale and Occlusion Aware Network for Vehicle Detection and Segmentation on UAV Aerial Images. *Remote Sens.* 2020, 12, 1760. <https://doi.org/10.3390/rs12111760>

11. P. Zhang, Y. Zhong and X. Li: SlimYOLOv3: Narrower, Faster and Better for Real-Time UAV Applications. 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), 2019, pp. 37–45, <https://doi.org/10.1109/ICCVW.2019.00011>.
12. Payal Mittal, Raman Singh, Akashdeep Sharma: Deep learning-based object detection in low-altitude UAV datasets: A survey. *Image and Vision Computing*. Volume 104, 2020, 104046, ISSN 0262-8856.
13. R. Girshick: Fast r-cnn. In *ICCV*, pages 1440–1448, 2015.
14. S. Ren, K. He, R. Girshick, and J. Sun: Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.
15. R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014.
16. J. Dai, Y. Li, K. He, and J. Sun. R-FCN: Object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems 29*, pages 379–387. Curran Associates, Inc., 2016.
17. T.Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar: Focal Loss for Dense Object Detection. In *ICCV*, pages 2980–2988, 2017.
18. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.Y. Fu, and A. C. Ber.: Ssd: Single shot multibox detector. Pages 21–37, 2016.
19. C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh: CSPNet: A new backbone that can enhance learning capability. *Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 390–e391.
20. T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick: Microsoft COCO: Common objects in context. In *Proc. ECCV*, 2014, pp. 740–755.
21. J. Redmon, S. Divvala, R. Girshick and A. Farhadi: You Only Look Once: Unified, Real-Time Object Detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788, <https://doi.org/10.1109/CVPR.2016.91>.
22. S. Ali, A. Siddique, H. F. Ateş and B. K. Güntürk: Improved YOLOv4 for Aerial Object Detection. 2021 29th Signal Processing and Communications Applications Conference (SIU), 2021, pp. 1–4, <https://doi.org/10.1109/SIU53274.2021.9478027>.
23. Wang, C.-Y.; Bochkovskiy, A.; Liao, H.: Scaled-YOLOv4: Scaling cross stage partial network. *arXiv* 2020, arXiv:2011.08036v2.
24. T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie: Feature Pyramid Networks for Object Detection, 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 936–944, <https://doi.org/10.1109/CVPR.2017.106>.
25. Pengfei Zhu and Longyin Wen and Dawei Du and Xiao Bian and Q. Hu and Haibin Ling: Vision Meets Drones: Past, Present and Future, *ArXiv* 2020, abs/2001.06303.
26. Sharma, S., Verma, V.K. AIEMLA: artificial intelligence enabled machine learning approach for routing attacks on internet of things. *J Supercomput* 77, 13757–13787 (2021). <https://doi.org/10.1007/s11227-021-03833-1>
27. Sharma, S., Verma, V.K. Security explorations for routing attacks in low power networks on internet of things. *J Supercomput* 77, 4778–4812 (2021). <https://doi.org/10.1007/s11227-020-03471-z>
28. Verma VK, Ntalianis K, Moreno CM, Yang C-T. Next-generation Internet of things and cloud security solutions. *International Journal of Distributed Sensor Networks*. March 2019. doi:<https://doi.org/10.1177/1550147719835098>
29. Guo, W.; Li, W.; Li, Z.; Gong, W.; Cui, J.; Wang, X.: A Slimmer Network with Polymorphic and Group Attention Modules for More Efficient Object Detection in Aerial Images. *Remote Sens*. 2020, 12, 3750. <https://doi.org/10.3390/rs12223750>

A Microservice Architecture with Load Balancing Mechanism in Cloud Environment



Satyanarayana Mummana, Bosubabu Sambana , Budimure Ramanababu, Preethi Gandreti, P. Pratima Rani, Priyanka Mishra , A. Chandrasekhar, and K. Narasimha Raju

Keywords Cloud computing · Load balancing · Scheduling algorithm · MSA · Microservice chain.

S. Mummana · B. Ramanababu

Department of Computer Science and Engineering, Lendi Institute of Engineering and Technology (A), Vizianagaram, Andhra Pradesh, India

B. Sambana (✉)

Department of Computer Science and Engineering, Lendi Institute of Engineering and Technology (A), Vizianagaram, Andhra Pradesh, India

Jawaharlal Nehru Technological University, Kakinada, Andhra Pradesh, India

Department of Computer Science and Engineering, Indian Institute of Information Technology Kota, Jaipur, Rajasthan, India

e-mail: 2021krcp1001@iiitkota.ac.in

P. Gandreti

Department of Computer Science and Engineering, Gayatri Vidya Parishad College for Degree and PG Courses (A), Visakhapatnam, Andhra Pradesh, India

P. P. Rani

Department of Computer Science and Engineering, Vignan's Institute of Information Technology, Visakhapatnam, Andhra Pradesh, India

P. Mishra

Department of Computer Science and Engineering, Indian Institute of Information Technology Kota, Jaipur, Rajasthan, India

A. Chandrasekhar

Department of Computer Science and Engineering, Avanthi Institute of Engineering and Technology, Vizianagaram, Andhra Pradesh, India

Jawaharlal Nehru Technological University, Kakinada, Andhra Pradesh, India

K. Narasimha Raju

Department of Computer Science and Engineering, Gayatri Vidya Parishad College of Engineering (A) Visakhapatnam, Andhra University, Andhra Pradesh, India

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

R. Misra et al. (eds.), *Machine Learning and Big Data Analytics*,

Springer Proceedings in Mathematics & Statistics 401,

https://doi.org/10.1007/978-3-031-15175-0_7

1 Introduction

Homogeneous architectural applications have been subjected to the current internet environment's quick iterations and rapid changes in user needs, resulting in increased development, operation, and maintenance costs. As a result, microservice architecture is being adopted by an increasing number of applications. Microservices Architecture (MSA) is a Service-Oriented Architecture (SOA) variant that divides an extensive, complicated software application into numerous loosely linked microservices, clusters, and services. The scheduling and expansion granularity is decreased to the microservice instance level, which can lower development, deployment, and expansion costs [1, 2].

While reducing service granularity, service governance will face more significant challenges. Several related microservices usually process the request to return the expected result to the user. In the framework of complex microservices, how to manage requests and optimize the processing delay of recommendations is the goal of this paper.

This work employs the various benefits of the load-balancing technique that is based on microservice chain awareness, and many more other applications that are reduced service request channels, causing delay during the microservice services.

The proposed new architectures use load-balancing and microservice chain awareness approaches [1]. We describe the microservice architecture's deployment scenarios, draw the topological diagram of the cloud data center, and give the microservice instance, microservice chain, host description, and user request delay measurement standards to reduce the communication cost of the microservice chain in the data center network. Finally, simulation tests are built-in Python based on the formal model, and many sets of different experiments similar to the natural environment are set up.

Related Work

LEWIS defines the concept of microservices as equal to 2014. It is an architectural style [3] and a specific implementation of SOA [4]. The main idea is to split the traditional monolithic application into a series according to business logic and its functions. Independent design, development, deployment, operation, and maintenance of software service units, and under the premise of observing the service boundary, each microservice can cooperate to realize the value of the entire system [5].

MSA refers to an enterprise-level distributed application system architecture formed by pre-delineating service boundaries and defining microservices for service composition according to the business needs of the entire application system [6]. We document [7, 8] traditional cloud computing task scheduling and load balancing issues architecture and management services for the micro level, not further expansion. Literature [9] studies of microservice governance schemes based on service grids do not involve the governance of service chains.

Documentation [10, 11] of the container system will be based on the runtime performance of event analysis, the micro-container-based service system, and the

load balancer, which significantly affects the microservice. Literature [12] builds a performance model through sandboxed microservices, reasonably configures capacity for different services, and proposes a feasible solution to the load problem from another perspective. Literature [13] suggests a chain-oriented load-balancing algorithm based on message queues, which uses HTTP and message queues in combination, but will cause additional operational complexity and increase overhead in the microservice system.

Based on the evaluation of the studies, this paper examines the characteristics of clouded microservice architecture from the perspective of load balancing. It proposes a microservice chain-conscious request load balancing set of rules to resolve the trouble of shared microservice opposition in microservice chain calls. The algorithm is effectively evaluated through simulation experiments from the perspective of average request latency on the microservice chain and host request load balancing.

2 Request Load Balancing Algorithm for Microservice Chains

This section starts with a microservice chain invocation scenario and determines the best service execution path by modeling and evaluating the microservice chain responding to the request in advance and achieving complete load configuration and request delay optimization.

2.1 Application Scenario Description

Data Center Network

In the actual scenario, this paper researches the problem of microservice communication composition in the cloud data center. See the data center network topology in Fig. 1.

With micro-services deployed in the container and running on the host, the same host can deploy a plurality of microservice instances simultaneously, and therefore the host H_i is described as a subset of microservice instances, i.e., $H_i \subseteq I$. The formula represents the relationship between the microservice instance and the host (1):

$$\text{InHost} \left(H_i, I_j^k \right) = \begin{cases} 0, & \text{micro-server service control} \\ I_j^k & \text{No-bit to the main unit } H_i \\ 1, & \text{the micro-server service control} \\ I_j^k & \text{Position in the main machine } H_i \end{cases} \quad (1)$$

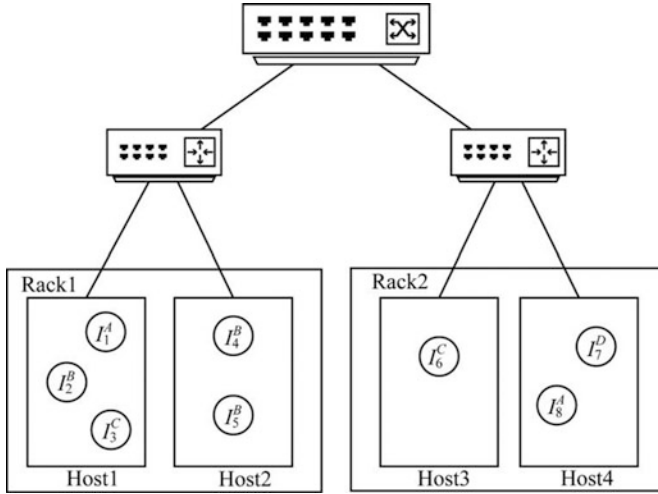


Fig. 1 Structure of data center network topology

The circle inside the main unit (in the form of I_j^k, I_k^j) represents the illustration of a microservice placed on the host, where i is the microservice instance's unique identifier, and j is the type of microservice to which the microservice model belongs.

Microservice Chain

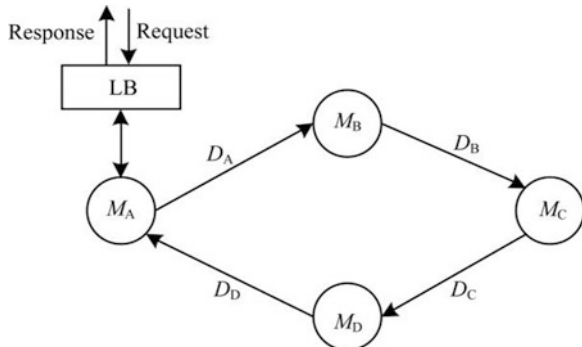
The microservice chain is an ordered crew regarding obtained microservices up to expectation, which can maintain the situation concerning the article's instruction concerning microservice communication. It is represented by the symbol C_i , where i is the microservice chain's unique identifier.

User request R & I is provided by a particular widget service chain; in response, I is the user request's unique identifier. A microservice chain will be associated with each user request [1]. Equation (2) describes the mapping relationship:

$$R \text{ to chain } (R_i) = G_j \tag{2}$$

Figure 2 illustrates a user request technique corresponding to a whole microservice chain. Among them, the direction of data transmission is represented by the arrow. The symbol D_A on the hand represents microservices, M_A represents transmission to microservices, and M_B represents the requested data size, D_B, D_C, D_D , and so on, in the service chain the invocation of the scene. The same transmission delay between the service performance of the service will have a more significant impact. Therefore, by D , a calculation of the transmission delay as link is weighted.

Fig. 2 Simplified topology of the microservice chain



The microservice chain can be expressed as follows:

$$C1 = \{M_A, M_B, M_C, M_D\} \tag{3}$$

M_i represents a microservice instance object, and i is a unique identifier.

Microservices and Microservice Examples

Nearby is a communication process through the many-to-many relationship among microservice chain relationship and its service instance, also maintaining the relationship and resolving the proposed solution this formula (4):

$$N \text{ In Chain } \left\{ \begin{array}{l} 0 : \text{microserver service control } M_j. \text{ It does not belong to the service business chain } C_i \\ 1 : \text{the microserver service control } M_j. \text{ It belongs to the service business chain } C_i \end{array} \right. \tag{4}$$

At the same time, the microservice instance can be defined as I_i^j , where i is the unique identifier of the microservice instance, and j indicates that the microservice instance provides the service of the microservice M_j .

2.2 System Model

Frequent and multiple requests in complex networks are the main reason for the increase in service delay [11]. This article expands the load-balancing problem to the problem of service communication composition invocation in the microservice

environment and takes the microservice chain as the research object to seek the service chain type.

Performance optimization strategy in combination: In this section, the key factors affecting the micro-environment of the service request-response and load configuration formalized the process to make it efficient (see Table 1) and constructed it based on this request for the service chain micro-response model.

Table 1 Formalized description of the parameters

Symbol	Definition
R	Request collection
R_i^j	The request ID is I , microservice chain j request
R^j	Total number of requests for microservice chain j
C_i	Microservice chain i
M_i	Microservice i
\bar{I}_i	Examples of ID I , micro-services j microservice instance
H_i	The host whose host ID is i
P_i^k	The request ID is i , the waiting and execution time of the <i>microservice</i> k
E^k	Microservice k execution time
W_i^x	The waiting time of request i on microservice instance x
$T_i^{x,y}$	The transmission time of request i from microservice instance x to microservice instance

To simplify the abstract system, this article focuses on the service request problem in the microservice environment [14]; the resources obtained by the microservice instance are set up to be the same, the host environment is homogeneous, and the host tasks are stringed together. Line-up.

A specific queue receives all requests sent to the load balancer, and the microservice communication combination model is constructed for the proposals in the line. The combination of the instances is traversed through the cost function to obtain the lowest cost instance combination [15]; the mixture is effective within a period.

2.3 Metrics

Measurement signs are crucial in the analysis of findings. This article uses the average request latency and host load at the microservice chain as measurement signs inside the microservice structure environment beneath the cloud machine [16].

The calculation method of the average request delay on the microservice chain is shown in formula (5):

$$L_c = \frac{\sum_i L_i}{R^c}, \{i \mid R \text{ to chain } (R_i) = R_c \quad (5)$$

where L_i is the delay time of a single request [17, 18]; formula (6) is developed mainly by the execution waiting time P_i^k and request data transfer time T_i^{xy} composition:

$$L_i = \alpha \sum P_i^k + \beta \sum T_i^{xy} \quad (6)$$

among them, waiting for execution time P_i^k by waiting for time W_i^x and the execution time $E^{m(k)}$ ($m(k)$ can obtain the microservice type of the microservice instance (k), as shown in Eq. (7):

$$P_i^k = W_i^k + E^{m(k)} \quad (7)$$

with waiting time W_i^k . The calculation is shown in Eq. (8), representing the sum of the time for the current instance to process all the waiting in the queue.

$$W_i^k = \sum E^{m(i(r))} s.t. \{r \mid r \in Q(h(k))\} \quad (8)$$

Among them, $h(k)$ is the host where the k instance is located; $Q(x)$ is all the requests in the queue of the host x ; $i(r)$ is the instance where the request r is obtained.

Furthermore, considering waiting time, the data transmission time also has a significant impact on the performance of the request response, so the weight of the transmission link is set as shown in Eq. (9):

$$T_i^{xy} = \frac{D_{m(x)}}{speed(x, y)} \quad (9)$$

The transmission speed is measured as y the distance between two microservice instances, as shown in Eq. (10):

$$speed(x, y) = \begin{cases} \text{same host, real cases } x \text{ and } y \\ \quad \text{in the same a main machine.} \\ \text{sameRack, real cases } x \text{ and } y \\ \quad \text{in with a machine frame.} \\ \text{diffRack, real cases } x \text{ and } y \text{ in} \\ \quad \text{not the same machionfame.} \end{cases} \quad (10)$$

2.4 Algorithm Design

Algorithm 1 shows the microservice chain's perception of the request load balancing algorithm.

Algorithm 1: Request load balancing algorithm for microservice chainsEnter the: **U**User request set **R**Output: **The order in which microservice instances are requested**Output: **The order of requesting microservice instances**

```

1. for each  $R_i \in R$ , do
2.   chain  $\leftarrow$  RToChain ( $R_i$ );
3.   MSset  $\leftarrow$  getMSfromChain (chain);
4.   DSset  $\leftarrow$  getDSfromMSset (MSset);
5.   insOrder  $\leftarrow$  setEmptyInsfromMSset (MSset);
6.   while DSset is not empty, do
7.     data, priorityMS  $\leftarrow$  findMaxDataSize (DSset);
8.     nextMS  $\leftarrow$  next (MSset, priorityMS);
9.     pMSIns  $\leftarrow$  getInsfromMS (priorityMS);
10.    nMSIns  $\leftarrow$  getInsfromMS (nextMS);
11.    .pIns, nIns  $\leftarrow$  findMinCost (pMSIns, nMSIns);
12.    if insOrder.find(priorityMS) is true and
insOrder.get(priorityMS) is null then
13.      insOrder.set(priorityMS, pIns);
14.    end if
15.    if insOrder.find(nextMS) is true and
insOrder.get(nextMS) is null then
16.      insOrder.set(nextMS, nIns);
17.    end if
18.    DSset.pop (priorityMS, data);
19.  end while
20. send ( $R_i$ , insOrder);
21. end for

```

Algorithm 1

In this paper, we have represented the step-by-step procedure. First, it will receive the user task (requested operations which are to be done) at an initial stage, primarily based on the content of the user request; here, it will analyze the microservice chain to which the function belongs and determine the form of microservices on the chain and their dependencies in keeping with the microservice chain, below which it utilizes the records transmission facts inside the microservice chain. It analyzes the communication combination regarding the microservices to be solved forward and afterward utilizes the fee feature in conformity with expostulating the situations within the mixture according to determining the instance volume including the youngest value, yet below this it makes use of the occasion mixture following replacement of the requested microservice instance collection.

This procedure will repeat until all microservice occurrences are chosen. Then, it can be given the request and placed ahead of the request collection for the microservice happenings and the proposal itself [1], finishing the load balancing of the demand. The range of microservice units and information sets within the microservice chain similar to the request is believed to be n , and the fact set DSset is pre-taken care of, lessening the time overhead. The sorting algorithm determines the majority of the algorithm's time complexity, and because the time cost of the loop is determined together, the total time complexity is $O(n \log n)$.

3 Simulation Experiment Results and Performance Analysis

The simulation experiment is written in Python, and the above hassle model is abstracted and reported through an object-orientated programming paradigm. There are currently many cloud simulators, such as CloudSim [19], but most cloud simulators are oriented to the virtual machine mode and a few support containers, a new type of process isolation “virtualization” method because the problem model is more unique. So, reference [20] adopts the scheme of a custom scheduler.

3.1 Parameter Setting

Based on the model designed above, the network architecture in the simulation environment is similar to that of a data center, with one router, two switches and three hosts. The simulated time slice in this simulation experiment is measured in milliseconds, and the data size is in kB. The returned data size for a microservice spans from 1 to 100 kB, and the instance processing request time ranges from 1–10 ms. Describe several microservices during program initialization to calculate the quantity of data returned and the processing time of a single request. This article treats the transmission link between data center networks as a fixed value, which can also be interpreted as the link’s weight for the two instances of the equal host [1].

The communicate charge among them is in the main limited by using a disc throughput for twice the equal body with distinctive hosts; the verbal exchange fee between them is on the whole restrained by using the community interface card throughput, and eventually, for multiple hosts, the communication fee between them is broadly speaking controlled by disc throughput. Given the strain on the data center backbone network, the rate for communication between rack instances should be the slowest of the three. The specific experimental parameters are shown in Table 2.

This paper’s serialization of host isomorphism assumes that the host performance is the same, and a weighted round-robin (RR) is not required. Therefore, the polling load balancing algorithm becomes the comparison algorithm of this article.

Table 2 Parameter settings of the experiment

Parameter	Parameter value	Remark
A	1	Waiting time weight
A	1	Weight of the requested transmission time
sameHost	512	Transmission speed between microservice examples on the same host
sameRack	102	Transmission speed between microservice instances in the same rack on different hosts
diffRack	30	Transmission speed between microservice instances in different racks

3.2 Microservice Chains

Three Microservice Chains

For micro-placement service instance herein see Fig. 3; the three microservice chains are provided in Fig. 4.

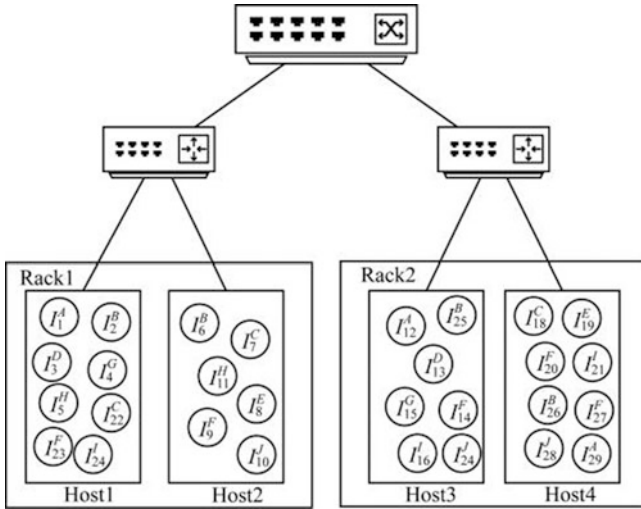


Fig. 3 Instance placement method of three microservice chains

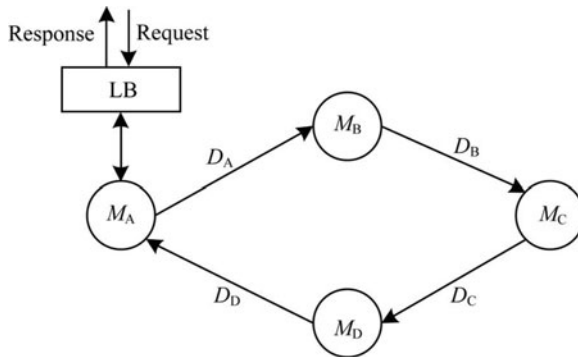


Fig. 4 Simplified topology of three microservice chains

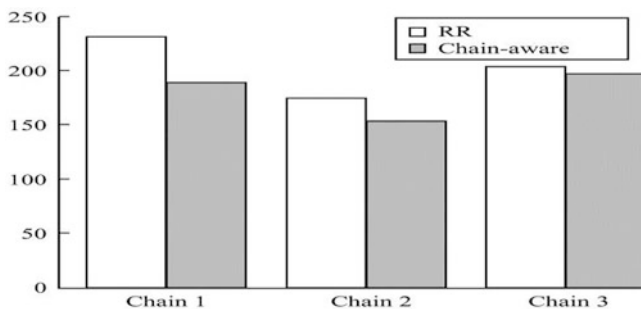
The three microservice chains are $C_1 = \{M_A, M_B, M_C, M_D\}$, $C_2 = \{M_E, M_F, M_G\}$ and $C_3 = \{M_H, M_I, M_J\}$.

The related microservice details are shown in Table 3.

Table 3 List of three microservices

Microservice type (ID)	Microservice task execution time/ms	Size of the data recovered by the microservice
A	1	100
B	5	200
C	2	400
D	1	300
E	1	100
F	3	200
G	1	50
H	2	100
I	4	300
J	3	100

In this paper, we apply the rotating request mode for three microservice chains, sending one service request to the system every two-time slice (2 ms) and calculating the average request latency of each microservice chain using formula (5); the results are shown in Fig. 5.

**Fig. 5** The average microservice delay on load balance

The average microservice delay of the load balancing method proposed in this paper is lower than that of the round-robin scheduling method (-18.47% , -12.30% , -2.62%), and good results have been achieved.

One Microservice Chain

In addition to the three microservice chains, this article also sets up an experiment with only one microservice chain ($C_1 = \{M_A, M_B, M_C, M_D\}$) to verify that the algorithm in this article requests at different frequencies. The detailed information of the microservices under the performance is shown in Table 4.

Table 4 List of one microservice

Microservice type (ID)	Microservice execution time/ms	Size of the data recovered by the microservice
A	1	100
B	3	200
C	5	400
D	2	300

In the case of a microservices chain, two kinds of request modes are provided herein to measure the performance and deployment mode as in Fig. 6. First, the system sends a request to turn the medium frequency, i.e., every three-time slice (3 ms) transmits a request, and an average microservice requested by the delay chain of formula (5) is calculated, as obtained in Fig. 7. The results are shown.

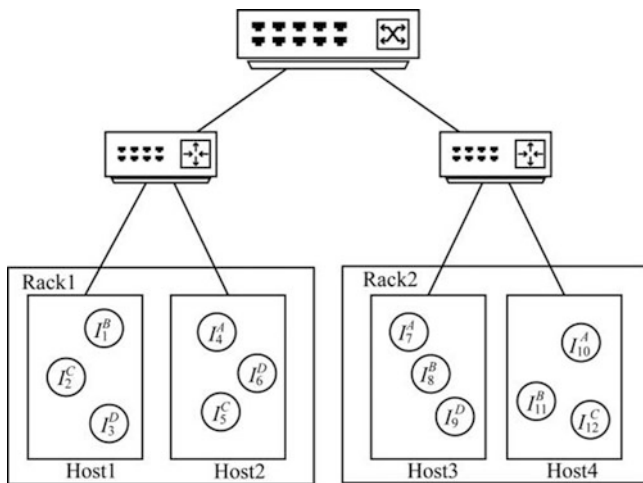


Fig. 6 Instance placement method of one microservice chain

Figure 7 shows that the method in this paper can significantly reduce the request delay by 73.6%.

Performance tests of extreme conditions (e.g., high concurrency scenario) in turn transmit a request of higher frequency, i.e., every two-time slot (2 ms) to send a request to give the system (Fig. 8); experimental results are shown.

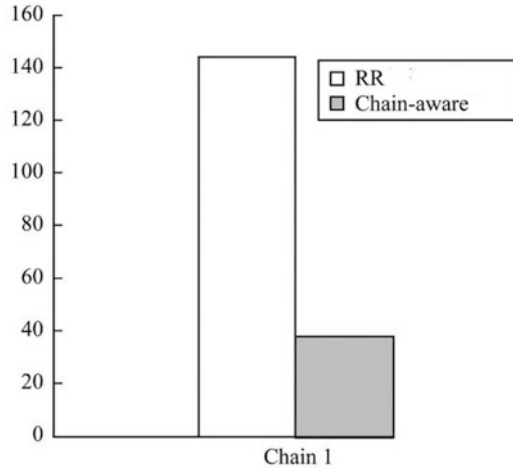


Fig. 7 Average latency of medium frequency requests

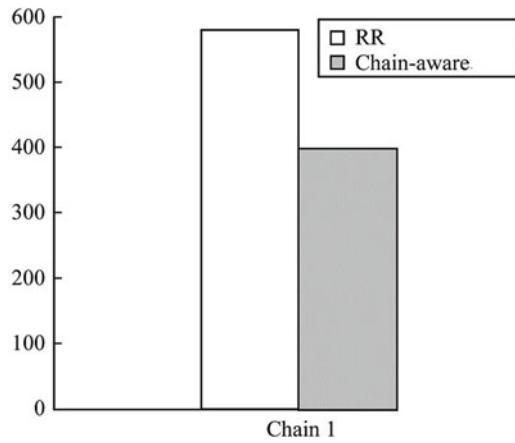


Fig. 8 Average latency of high-frequency requests

Figure 8 shows that the method in this paper has better performance even under high load conditions. Based on the above experimental results, in a complex environment with multiple microservice chains, that is, an environment closer to the actual microservice architecture, the algorithm can effectively reduce request latency.

3.3 Host Load Analysis

The repository performs sectioning of the host’s headquarters or computational resources as appropriate under the traits concerning basket system isolation. Based on the serialization admission indicated between portion 2.2, all on-hand computational sources concerning the current army can be old, so the preferred mission is running. As a result, as long as the appeal ready sequence on every army is aged within that dissertation, then the proof speed measurement is cut up through microservices, and the actual burden on the host stays accurately determined.

Three Microservice Chain Request Modes

The service chaining mode has been requested. Three kinds of micro-robin requests broadcast a service request to the system every two time slots (2 ms), which records the load value of each host in each time slice. The results obtained are shown in Fig. 9.

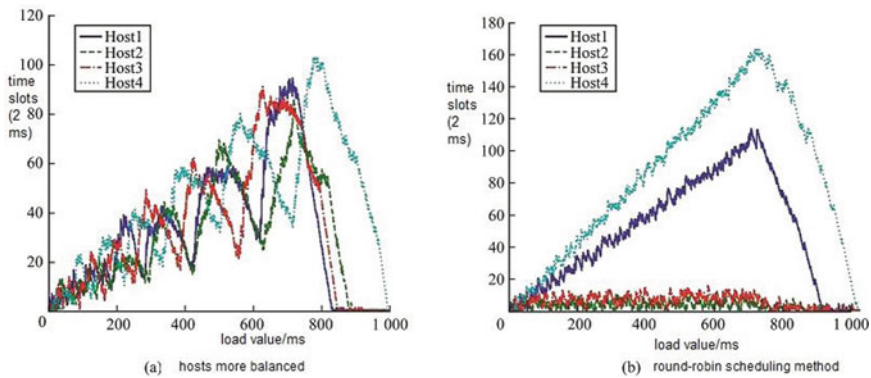


Fig. 9 The method described in this paper can help balance the load between hosts, whereas the round-robin scheduling method causes significant load tilt

One Microservice Chain Deployment Model

For a deployment pattern, the microservice chain first needs to send a request at a moderate frequency to the system, i.e., every three time slices (3 ms) it sends a request and then records the load value of the host of each time slice, as shown in Fig. 10. The host load situation under the frequency request in the one microservice chain is shown.

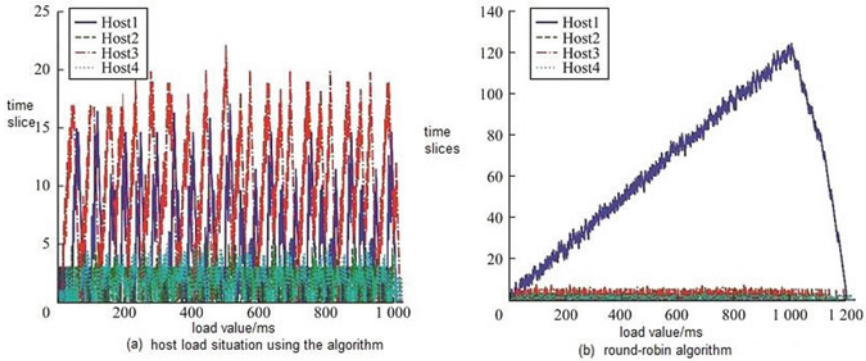


Fig. 10 Host load conditions under separate requests of one microservice chain

Figure 10a is the host load situation using the algorithm in this paper, and Fig. 10b is the host load situation using the round-robin algorithm. The two sets of studies show that the load between the hosts in this way is more balanced, whereas the round-robin scheduling method still has a significant load tilt, i.e., every two time slots (2 ms) transmits a request by sending a request to the high-frequency system, such as to give the 11 Host load shown. Figure 11 shows that in the two sets of experiments, the load between the hosts in this method will be more balanced, while the round-robin scheduling method will still have a severe load tilt.

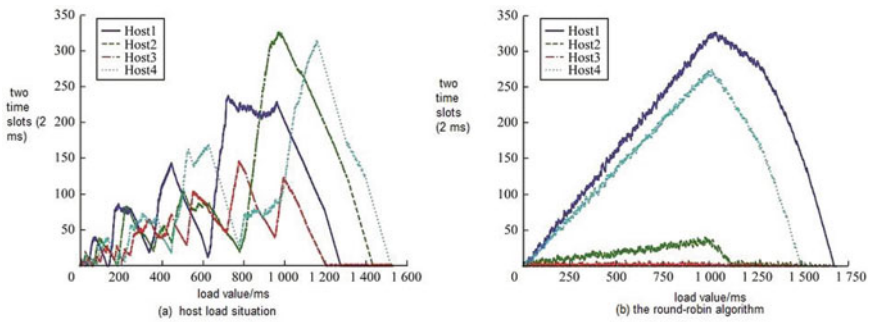


Fig. 11 Host load conditions under a single high-frequency request of one microservice chain

3.4 Experimental Results

In this paper, experimental results, which might lean toward the natural surroundings, are obtained via distinct experiments on three microservice chains and one microservice chain, using unique instance setting methods and request modes [1].

In the situation of three microservice chains, the experiment implementing the load version method of this challenge has a lower expected observed latency in the microservice band than the assessment strategy. They assign scenarios over the legion to maintain extra coherence compared to the evaluation approach. Simultaneously, the set of rules inside that studies plays higher now that the microservice chains are intersected (this is correlated), while keeping military load balance can notably minimize the average length over microservice chain requests. The excessive concurrency instances expand when numerous microservice times and choppy placement strategies are in the microservice chain. The weight balancing approach described in this article can efficaciously lessen the microservice chain's request time.

Nonetheless, the surroundings have excellent performance, indicating that the burden balancing method described in this text can successfully restrict requests put off below the complicated microservice chain, which is akin to the actual microservice structure surroundings. This paper's algorithm achieves a balanced result in phrases of host load, whereas the round-robin scheduling technique often results in substantial load tilt.

4 Conclusion

In this paper, we give qualitative work on initiative transmission calls and later intestinal load patterns based on microservice architecture, and combine them along assigned balancing to make assignments through scheduling algorithms and identify the advanced planet computing architectures functions. Also, we propose a microservice chain-conscious request load balancing set of rules based entirely on the know-how abilities furnished to the cloud environment via the microservices. Balance the load between hosts by reducing the extended time request in the microservice architecture and address the competition for shared microservices in provider chain invocation. Experimental penalties are shown as the algorithm can minimize the average put-off concerning requests underneath the difficult microservice and apply an invitation chain or the host's workload correctly.

Durability: With the non-stop improvement of the container era and microservice structure, the software program industry will advocate better optimization dreams for microservice governance. Since microservices are based on the characteristics of container runtimes, the next step is to combine microservice deployment and optimization to make them more suitable for deployment plans for specific scenarios.

References

1. THONES J. Microservices[J]. IEEE Software, 2015, 32(1): 116-126. DOI:<https://doi.org/10.1109/MS.2015.11>

2. BRONDOLIN R, FERRONI M, SANTAMBROGIO M. Performance-aware load shedding for monitoring events in container-based environments [C]//Proceedings of ACM SIGBED'19. New York, USA: ACM Press, 2019: 27-32.
3. LLOYD W. Serverless computing: an investigation of factors influencing microservice performance[C]//Proceedings of IEEE International Conference on Cloud Engineering. Washington DC, USA: IEEE Press, 2018: 159-169.
4. JINDAL A, PODOLSKIY V, GERNDT M. Performance modeling for cloud microservice applications[C]//Proceedings of ACM/SPEC International Conference on Performance Engineering. New York, USA: ACM Press, 2019: 25-32.
5. NIU YP, LIU FM, LI Z P. Load balancing across microservices[C]//Proceedings of IEEE International Conference on Computer Communications. Washington DC, USA: IEEE Press, 2018: 198-206.
6. FAZIO M. Open issues in scheduling microservices in the cloud[J]. IEEE Cloud Computing, 2016, 3(5): 81-88. DOI:<https://doi.org/10.1109/MCC.2016.112>
7. MA WJ, LIU Y, TANG X G. A dynamic programming approach for optimal signal priority control upon multiple high-frequency bus requests[J]. Journal of Intelligent Transportation Systems, 2013, 17(4): 282-293. DOI:<https://doi.org/10.1080/15472450.2012.729380>
8. NIKOLAOS A, HJALMTYSSON G. System, method and apparatus for network service load and reliability management[S]. USA Patent: 6, 760, 775, 2004-06-06.
9. BIRRELL AD, NELSON B J. Implementing remote procedure calls[J]. ACM Transactions on Computer Systems, 1984, 2(1): 39-59. DOI:<https://doi.org/10.1145/2080.357392>
10. DUSIA A, YANG Y, TAUFER M. Network quality of service in docker containers[C]//Proceedings of IEEE International Conference on Cluster Computing. Washington DC, USA: IEEE Press, 2015: 527-528.
11. RAJKUMAR B, RAJIV R, CALHEIROS R N. Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: challenges and opportunities[C]//Proceedings of International Conference on High-Performance Computing & Simulation. Leipzig, Germany: [sn], 2009: 1-11.
12. HASSELBRING W, STEINACKER G. Microservice architectures for scalability, agility, and reliability in e-commerce[C]//Proceedings of IEEE International Conference on Software Architecture. Washington DC, USA: IEEE Press, 2017: 243-246.
13. VICTOR M. Adaptive application scheduling under interference in kubernetes[C]//Proceedings of the 9th IEEE/ACM International Conference on Utility and Cloud Computing. Washington DC, USA: IEEE Press, 2016: 426-427.
14. FOWLER M, LEWIS J. Microservices[EB/OL]. (2014-03-25). [2020-05-20]. <http://martinfowler.com/arti-cles/microservices.html>.
15. ZIMMERMANN O. Microservices tenets[J]. Computer Science-Research and Development, 2017, 32(3/4): 301-310.
16. BI XH, LIU Y, CHEN F. Research and optimization of network performance for micro-service application platform [J] Computer Engineering, 2018, 44 (5):... 53-59 (in Chinese) Bixiao Hong, Liu Yuan, Chen Fei. Research and optimization of network performance of microservice application platform[J].Computer Engineering, 2018, 44(5): 53-59.
17. NWEAN S. Building microservices[M]. [S.l.]: O'Reilly Media, Inc, 2015.
18. JIANG SH, KIM T Y. The study of genetic algorithm-based task scheduling for cloud computing[J]. International Journal of Control and Automation, 2012, 5(4): 157-162.
19. FERRIS J M. Load balancing in cloud-based networks[S]. USA Patent: 8, 849, 971, 2014-11-30.
20. ZHENG JB, SHEN L Q. Research on service governance of microservice architecture based on service mesh [J] Computer Systems and Applications, 2019, 28 (2):... 55-61 (in Chinese) Zheng Jun praise, Shen Lam based. Research on Service Governance of Service Grid Microservice Architecture[J].Computer Systems & Applications, 2019, 28(2): 55-61.

An IoT Application for Detection and Monitoring of Manhole



Sai Pavan Tangella, Krishna Pavan Inala, Phani Gourinath Lella, and Syed Arshad

Keywords IoT · manholes · Water · Smart phone · Arduino · Underground drainage

1 Introduction

Manholes play an essential part in maintaining city sewage systems. They not only serve as maintenance access points, but also act as anchors in sewer system design. Sewage systems have been around almost since the dawn of civilization. A manhole is an opening to a confined space such as a shaft, utility vault, or large vessel. Manholes are also a key component in sewage system design. These service pipes are used to join different parts of sewers together. Manholes are often used as an access point for an underground public utility, allowing inspection, maintenance, and system upgrades. Most underground services have manholes, including water, sewers, electricity, storm drains, and gas. In this paper, we discuss manholes used for sewage and drainage systems. Most cities have adopted the underground drainage system, and it is the duty of the managing station (municipal corporation) to maintain cleanliness of the cities.

According to the traffic of the sewage or the area, there are different types of manhole systems [13–15]:

- Shallow manhole
- Normal manhole
- Deep manhole

S. P. Tangella · K. P. Inala (✉) · P. G. Lella · S. Arshad
Department of Electronics and Communication Engineering, Bapatla Engineering College (A),
Bapatla, Andhra Pradesh, India
e-mail: krishnapavan.inala@becbapatla.ac.in

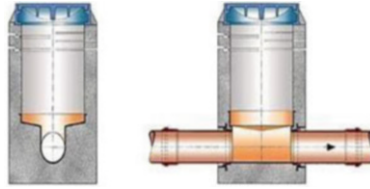


Fig. 1 Shallow manhole

Shallow manhole: If the depth of the manhole is 2 to 3 feet down, it is known as a shallow manhole. These are found at the beginning of a sewer branch as shown in Fig. 1.

Normal manhole: In this type of manhole, the depth ranges from 4 to 5 feet. Most manholes are round, but they may be rectangular or square instead as shown in Fig. 2.



Fig. 2 Normal manholes

Deep manhole: This type of manhole has depth >5 feet. They have heavy covers and gradually increasing diameters as shown in Fig. 3.



Fig. 3 Deep manholes

At present, the main problem in society is poor drainage systems, and many accidents occur because of lack of communication about broken manholes. If the

drainage maintenance is not proper, the pure water gets contaminated with drainage water and infectious diseases may be spread. If the drainage becomes blocked during rainy season, it will create problems for routine life such as traffic jams and environmental pollution, totally upsetting the public. Poor drainage systems are the main cause for the spread of diseases such as typhoid, dengue, and hepatitis. A main cause of diseases is sewage water on the roads. Thus, diseases may spread by direct or indirect contact. This happens because of the ignorance or unawareness of the municipal authorities. If officials of a municipal corporation (managing station) immediately learn about blockage of drainage in a manhole or broken lids, they can respond according to the situation, preventing numerous accidents. Therefore, we require a device that monitors manholes and detects changes, i.e., blockage of sewage water due to heavy rains, when the lid is open or broken, and increases or changes in the temperature or gas.

With modern technology, we can monitor and detect all these factors by using different technologies. One is by using IoT technology for monitoring all these factors. IoT describes the network of physical objects (things) that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet. These devices range from household to industrial tools. The manhole detection and monitoring system, which is based on IoT, can detect whether the lid is open and the water, gas, and temperature levels in the manhole; it sends the message to the managing authorities so that they can take certain measures.

There are different sensors, such as the tilt sensor, which is used to detect the opening or breakage of the lid, water level sensor, which sends information about the water level in the manhole, and gas sensor, which predicts the gas and temperature level; the data are sent to the managing authorities. With this device, many accidents can be prevented, which helps maintain a clean city.

2 Literature Survey

According to [1], today the entire world is trying to construct brilliant urban areas. Savvy underground foundations could be a significant part of the execution of shrewd urban areas. Seepage framework observation plays a significant part in keeping towns spotless and safe. There isn't sufficient labor for constant observation and differentiating minor issues. Therefore, we hope the IOT frameworks will help. The proposed framework is inexpensive and low maintenance. IoT-based program alerts the overseeing station through messages when any issues in the sewer vent are noted. In [1], the authors propose an inexpensive special sewer vent interruption identification framework with notice stages which shield the basic foundation. The proposed framework uses an Arduino Uno and different sensors to trigger the interruption at the right time before the copper links are taken.

According to [2], underground drainage systems are currently most widespread in large cities such as Mumbai. Because of the increasing population, there will

be a lot of waste. If the drainage is not properly maintained, pure water might get contaminated by drainage water, resulting in the spread of infectious diseases. For decades, many accidents have occurred because of lack of proper gas leakage systems; using gas sensors, the system can detect gases such as carbon monoxide, hydrogen sulfide, and methane.

From [3], monitoring of sewage outlet system has become very important recently. The presence of clogs in drainage systems results in overflow in streets, which results in spread of diseases. A recent survey found that hundreds of people die every year because of manual cleaning of drainage systems. The main intention of this project is to monitor and remove the clog in the drainage system before the water overflows. This project utilized a water level sensor to identify the clog in the manhole.

According to [4], most Indian cities have adopted an underground waste management system. This system is complex, so it is critical for this system to function properly. If the sewage framework isn't properly maintained, we could face serious problems. As a result, numerous technologies have emerged to identify, track, and manage these sewage networks. This challenge focuses on the ability to plan for underground observation using multiple methods.

From [5], drainage is the system or process by which water, sewage, or other liquids are drained from one place to another, and to maintain the proper drainage function, its condition should be monitored regularly. However, it is very difficult to monitor the entire area manually in areas where a human cannot reach. To solve these problems, in [5] the authors have developed and implemented a system using a wireless sensor network.

From [6], in India the waste framework is one of numerous issues. Because of the lack of sewage framework support, sewage water floods and obstructs the city and blends with the drinkable water, causing medical issue for the surrounding population. To tackle this issue, we propose the model called the Drainage Overflow Monitoring System (DOMS). This proposed framework will screen the water and gas levels within the sewage framework.

From [7], in urban areas, the Internet of Things (IoT) could be a significant and simple tool to provide a solution for public problems. Today, urban India is facing trouble with drain and sewage water. Most waste enters lakes, waterways, and groundwater without treatment. This is due to the ignorance of the municipal authorities in cities. This paper gives an IoT-based framework to deal with this issue.

From [8], certain measures have recently been taken to improve the health conditions and hygiene of the country. Many drainage or sewage problems lead to poor sanitation, causing life-threatening diseases. You can overcome these issues in innovative and effective ways. Humidity and dryness sensors can be used to detect overflow. To detect dangerous gases such as methane, gas sensors can be used, and temperature can be detected by temperature sensors. This also saves the lives of medical staff. Safety for sanitary workers is also important.

According to [9], savvy sensors are used for observing both storm waters and channels of the waste product organizations of the scientific grounds of the University of Lille in northern France. This represents a town of around 25,000

people. For every organization, the paper presents the checking framework and examination of the recorded information, and this investigation caused upgrading of our comprehension of the organization's working, even its improvement.

According to [10], in India, the introduction of drainage systems is very important. Many places are not clean and are clogged by drainage due to mishandling of sewer cleaners. The clever Sabotage Drainage System-Essentials is used in multiple infrastructures. If the drainage system cannot be maintained, clean water can be contaminated with wastewater. This spreads epidemics. Using an ultrasonic sensor, the information will be sent to the ESP32 microcontroller whenever a blockage is found.

From [11], manhole cover (MC) failure is increasing and typically impacts the protection and economic system of society. That is why the need for fully automatic tracking systems has become critical. Automatic tracking of MCs is part of the development of smart towns (SC) and internet of things (IoT), which can be the objectives for cutting-edge governments to govern and screen the resources in cities. This paper is a survey study on MC problems, providing a complete category with analysis for those problems based on the environmental effect, including the modern-day tracking techniques.

From [12], manhole problems within populated towns are a major problem. Manhole explosions have been a major threat recently. Manhole detection and alerts are especially based on detecting manholes that are open because of sewage/rainwater overflow. To avoid such incidents even before they affect the public, an alerting device is built in where the buzzer signals and sends the sensed information to the handling government using GSM strategies, so they can take precautions and shut the manhole for public safety.

3 System Model

3.1 Block Diagram

The system consist of both hardware and software components:

- Hardware components:
 - Arduino UNO
 - Temperature sensor
 - Water flow sensor
 - Tilt sensor
 - Gas sensor
 - LCD
 - GSM module
 - Buzzer
 - Power supply

- Software components:
 - Arduino

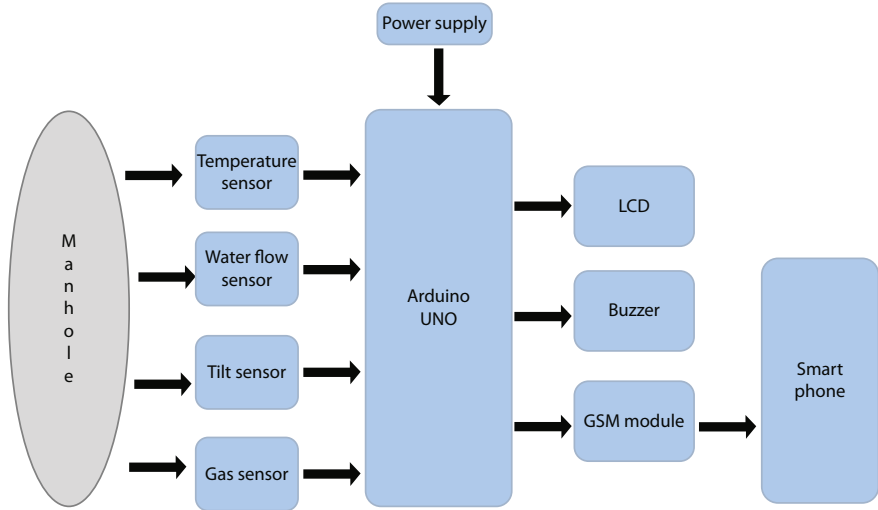


Fig. 4 Block diagram of the manhole detection and monitoring system

3.2 Hardware Components

Arduino UNO: Arduino is an open -source prototyping platform in electronics based easy-to-use hardware and software. It is low cost and flexible. This board contains a USB interface, i.e., USB cable.

Temperature sensor: Temperature sensors play a critical role in detecting the specific temperature in surroundings.

Water level sensor: Water flow sensor consists of a copper body, and it detects the water level by changes occurring in the resistance of the device due to changes in the distance from the top of the sensor to the surface of the water.

Gas sensor: Gas sensor is also known as gas detector. They are electronic devices that detect and identify different types of gases.

Tilt sensor: The tilt sensor is a device used for measuring the tilt of an object in multiple axes with reference to an absolute level plane.

LCD: liquid crystal display

GSM module: A GSM (Global System for Mobile Communications, originally Groupe Special Mobile) is a hardware device that uses GSM mobile telephone technology to provide a data link to a remote network.

Buzzer: Buzzer is an audio signaling device.

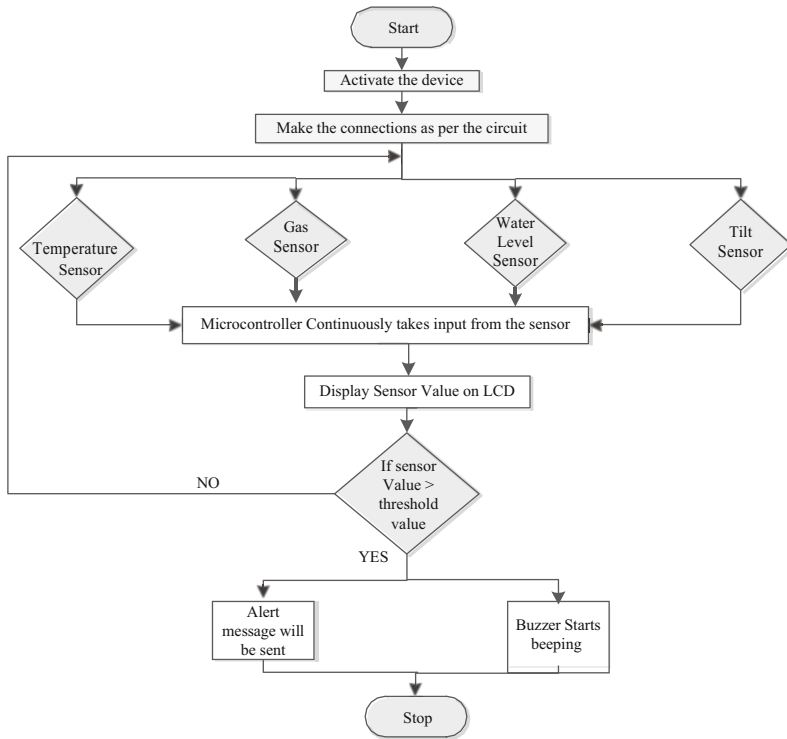


Fig. 5 Flowchart for the operation of system model

3.3 Software Description

Arduino microcontrollers are single-board computers that are popular in both the hobby and professional markets because to their ease of use and high performance. The hardware is affordable, and software development is free, because Arduino is an open-source initiative. It features 2 kB of RAM, 32 kB of flash memory for program storage, and 1 kB of EEPROM for parameter storage. At a clock speed of 16 MHz, it runs around 300,000 lines of C source code per second. The assessment board has 14 digital I/O pins and 6 analog input pins. When the application is run without being connected to the host computer, a USB connector is provided as well as a DC power jack for attaching an external 6–20 V power source to the I/O pins. The accompanying header connector is a 22-g single-wire connector.

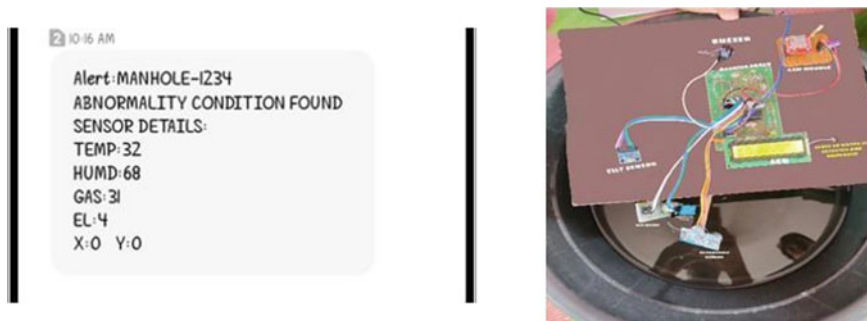


Fig. 6 Manhole detection using water level sensor

4 Operation of the System

An “IoT based manhole detection and monitoring system” will not only help in maintaining the proper health and safety of the city but also in reducing the work of government personnel. For this system, the connections were made as per the block diagram shown in Fig. 4. Various types of sensors like water level sensor, tilt sensor, temperature sensor, and gas sensor are interfaced with microcontroller Arduino Uno to make the system smart. Then, the output ports of Arduino Uno are connected with other components like the LCD, GSM module, and buzzer. First, the power supply is provided to the Arduino Uno board to start the working of the system or a device. A 5 v stabilized DC power supply is provided as an input to the Arduino board because this microcontroller is a low-power device and can be damaged if the input exceeds >5 v. The detailed flowchart for the operation of the system model is shown in Fig. 5.

After giving the power supply, the device starts working. First, the device senses all the sensors simultaneously; if there is an increase in temperature then it can be detected by using the temperature sensor. The temperature sensor detects the temperature and sends the digital data values as an input to the microcontroller. This microcontroller checks the data continuously and if the data value is more than the threshold voltage then the microcontroller sends the alert message to the managing authorities through smart phone by using the GSM module. The message will be displayed on the LCD and the buzzer starts an alarm. If there is a rise in water flow level then it can be detected by using a water level sensor (Fig. 6). Water level value w.r.t. water level sensor <5 cm indicates that a manhole is detected. In the received message, we obtained the water level as 4 cm. If the lid is tilted, then it can be detected by using a tilt sensor. The tilt sensor detects the orientation of a lid and gives its output (Fig. 7). Tilt in the manhole is detected when the coordinates are $X > 5$ or $X < -5$ and $Y > 5$ or $Y < -5$. In the received message, Y-coordinate is -10 . It indicates that a manhole has been detected. If the manhole is filled up with smoke or gases then it will be detected by using a gas sensor. This sensor senses the gas and sends the output (Fig. 8). If the gas sensor value is >50 ppm, then the

manhole is detected. In the received message, we obtained the gas level as 56 ppm. Municipal authorities will take the necessary action to resolve the problem.

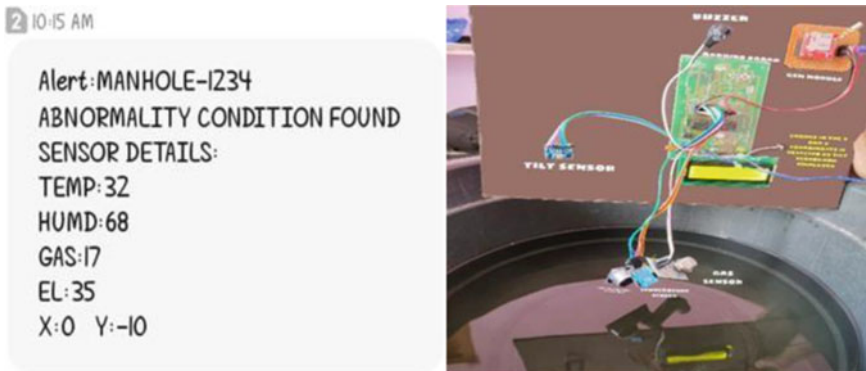


Fig. 7 Manhole detection using tilt sensor



Fig. 8 Manhole detection using a gas sensor

By this method, the officials could easily get notified about the problem and will take appropriate steps to solve the issue within a less interval of time. Also, Arduino Uno updates the live values of all the sensors in the manholes using IoT by displaying the message on the LCD, which gives clarity about the problem to the person in charge.

5 Conclusion

Manhole monitoring is currently a challenging problem. This paper provides different methods for monitoring and detecting problems in underground drainage systems. It explains various applications like underground drainage and manhole identification in real time. Various parameters like temperature, toxic gases, flow and level of water and lid breakage are being monitored and updated to the managing

authorities using the Internet of Things. This enables the person in charge to take the necessary actions regarding the same. In this way unnecessary trips to manholes are spared and repairs are only conducted when required; this reduces many risks to the population. Also, real-time updates to the managing authorities help maintain the regularity of drainage checks, thus avoiding hazards.

References

1. Lazarescu, M.T., "Design of a WSN Platform for Long-Term Environmental Monitoring for IoT Applications," IEEE Journal on Emerging and Selected Topics in Circuits And Systems, vol.3, no.1, pp.45, 54, March 2013.
2. Yuwat, C. and Kilaso, S. "A Wireless Sensor Network for Weather and Disaster Alarm System" .Proceedings of International Conference on Information and Electronics Engineering, Vol. 6, Singapore. pp 1 – 5, 2011..
3. [K. L. Keung, C. K. M. Lee, K. K. H. Ng and C. K. Yeung, "Smart City Application and Analysis: Real-time Urban Drainage Monitoring by IoT Sensors: A Case Study of Hong Kong," 2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), 2018, pp. 521-525, doi: <https://doi.org/10.1109/IEEM.2018.8607303>.
4. V. S. Velladurai, M. Saravanan, R. Vigneshbabu, P. Karthikeyan and A. Dhlipkumar, "Human safety system in drainage, unused well and garbage alerting system for smart city," 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2017, pp. 6-9, doi: <https://doi.org/10.1109/I-SMAC.2017.8058319>.
5. N. G. Haswani and P. J. Deore, "Web-Based Realtime Underground Drainage or Sewage Monitoring System Using Wireless Sensor Networks," 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), 2018, pp. 1-5, doi: <https://doi.org/10.1109/ICCUBEA.2018.8697512>.
6. R. Girisrinivaas and V. Parthipan, "Drainage overflow monitoring system using IoT (DOMS)," 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), 2017, pp. 2133-2137, doi: <https://doi.org/10.1109/ICPCSI.2017.8392094>.
7. N. Asthana and R. Bahl, "Iot device for sewage gas monitoring and alert system," in 1st International Conference on Innovations in Information and Communication Technology (ICICT), pp. 1–7, IEEE,, 2019.
8. S. Sathiyapriya, D. Rajeswari, P.Digvijayreddy, PeravarapuSaiKrishna,G. Yashwant "DRAINAGE OVERFLOW & MANHOLE MONITORING & DETECTION USING ARDUINO & NODE MCU" International Research Journal of Engineering and Technology (IRJET) , Volume:08 issue:4 2021.
9. Y. A. Rjeily, M. Sadek, F. H. Chehade, O. Abbas and I. Shahrour, "Smart system for urban sewage: Feedback on the use of smart sensors," 2017 Sensors Networks Smart and Emerging Technologies (SENSET), 2017, pp. 1-4, doi: <https://doi.org/10.1109/SENSET.2017.8125058>.
10. F. Wu, C. Rudiger and M. R. Yuce, "Design and field test of an autonomous IoT WSN platform for environmental monitoring," 2017 27th International Telecommunication Networks and Applications Conference (ITNAC), 2017, pp. 1-6, doi: <https://doi.org/10.1109/ATNAC.2017.8215386>.
11. H. H. Aly, A. H. Soliman and M. Mouniri, "Towards a fully automated monitoring system for Manhole Cover: Smart cities and IOT applications," 2015 IEEE First International Smart Cities Conference (ISC2), 2015, pp. 1-7, doi: <https://doi.org/10.1109/ISC2.2015.7366150>.
12. N. Nataraja, R. Amruthavarshini, N. L. Chaitra, K. Jyothi, N. Krupaa and S. S. M. Saqqaf, "Secure Manhole Monitoring System Employing Sensors and GSM Techniques," 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2018, pp. 2078- 2082, doi: <https://doi.org/10.1109/RTEICT42901.2018.9012245>.

13. <http://www.guinternational.com/products/manhole/shallow24.htm>
14. <https://civiljungle.com/manhole/>
15. <https://marchbridge.co.uk/portfolio/>

Resource Allocation in 5G and Beyond Edge-Slice Networking Using Deep Reinforcement Learning



Rohit Kumar Gupta, Praduman Pannu, and Rajiv Misra

Keywords Network slicing · Edge–fog cloud computing · Deep Q-learning · Reinforcement learning · Deep neural networks

1 Introduction

With the recent advancements in the networking technology of 5G and beyond, the demand for offloading various computational tasks from different Internet of Things (IoT) devices is increasing day by day. These computational tasks include analyzing the data generated from various sensors of IoT devices, cloud gaming, tasks related to artificial intelligent or machine learning systems, etc. Therefore, traditional ways of handling the offloading problem are not able to efficiently scale with the rising demand and dynamically increasing complexity of the problem. Demand for offloading computational tasks such as data processing, analysis, cloud gaming, etc. is exploding with an increasing daily surge in the number of IoT devices and the related compute-intensive tasks based on the generated data. Hence, efficient resource utilization and task offloading on the edge–fog cloud pose a complex problem and an optimal policy to balance the compute time and networking cost. 5G and Beyond Network capability provides multi-tenancy support with Quality of Service (QoS) and Quality of Experience (QoE) [1–3].

The MEC environment is characterized by high bandwidth, low latency, real-time insight, and proximity into RAN information followed by location awareness[4]. There are many MEC market drivers such as Business Transformation, Technical Integration, Industry collaboration, and several use cases like health, AR/VR. MEC computing improves network capabilities and performance to deploy and support different scenarios and use cases, such as remote surgery [5–7].

R. K. Gupta (✉) · P. Pannu · R. Misra
Indian Institute of Technology, Patna, India
e-mail: 1821cs16@iitp.ac.in; praduman.cs17@iitp.ac.in; rajivm@iitp.ac.in

Existing methods use conventional Q-learning-based reinforcement learning approaches like [15] to tackle the problem. However, these approaches need to maintain a Q-table for all the state action pairs which is not scalable or inefficient due to a large number of states and actions required to be maintained to calculate the immediate and future rewards. Instead, we can replace the Q-table with a deep learning model which can be trained to provide an approximation of rewards for a large sample space. The proposed Deep Q-learning Neural network (DQN)-based approach efficiently offload computational tasks to the edge, fog, and cloud resources with optimal task execution and networking cost [8, 9].

The remaining of the chapter is organized as follows. In Sect. 2, we discuss the related work. Section 3 introduces the system model including edge–fog cloud computing overview, resource allocation, and task offloading scheme. In Sect. 4, we discuss Deep Reinforcement Learning-based approach for resource allocation. Further, in Sect. 5, we discuss the simulation results followed by performance evaluations. Finally, in Sect. 6, we conclude this chapter with future works.

2 Related Work

There are many related works that jointly model the resource allocation problem and computing mode decision problem in edge networks. For instance, in [10], the authors investigate dynamic resource allocation problem for control and computing MEC resources in IIoT. The problem has been transformed into Markov decision process (MDP) to minimize the average delay of tasks. To solve the MDP problem, the authors have proposed deep reinforcement learning-based dynamic resource management and allocation algorithm. In paper [11], the authors adopted binary offloading policy for computation task to be fully offloaded or executed locally to an MEC server and proposed Deep Reinforcement learning-based Online Offloading framework so that it could learn and perform binary offloading decisions from the past experiences.

Other research is considered in the context of edge-IoT resource allocation using a reinforcement learning approach. For example, the work in [12] considered an enhanced Q-learning scheme that allocates resources to IoT applications from edge–cloud to maximize the utility. In the paper [13], the authors propose a resource management solution for DNN Inference in IIoT. To improve the resource utilization efficiency, the authors [14] proposed resource allocation IoT edge policy for computing system. The MEC system resource allocation is formulated as Markov decision process (MDP) further, deep reinforcement learning is applied to solve the problem. In the paper [15], the authors propose an offloading algorithm that works near optimal task based on ϵ -greedy Q-learning. In the paper [16], the authors propose edge–fog cloud distribution task processing and developed the Least Processing Cost First (LPCF) solution method for tasks assignments to the nodes which provide near optimal networking costs and optimal processing time and furthermore evaluated LPCF in different scenarios to demonstrate its effectiveness.

3 System Model

3.1 Resource Allocation and Task Offloading in Edge-Slice Networking

Edge-fog cloud computing is one of the promising areas of 5G which has huge potential to cater to the computational requirements of the ever expanding Internet of Things (IoT) infrastructure. There are thousands of cloud data centers, millions of fog nodes, and billions of edge devices in the edge-fog infrastructure as shown in Fig. 1. These resources can be used to offload tasks based on the various requirements such as latency, bandwidth, execution time, security, etc. Therefore, efficient offloading techniques are required which can process these requests in real time while providing optimal overall (task execution and network) cost.

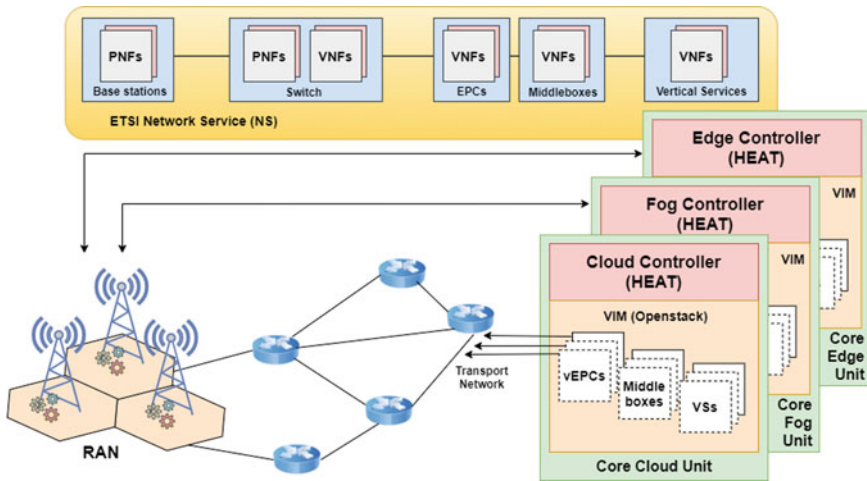


Fig. 1 Data plane overview of Edge-Slice Networking

There are various methods [12, 15, 16] to solve the resource allocation and task offloading problem including but not limited to the use of reinforcement learning. However, these approaches maintain a reward-based Q-table for all the states and actions. These approaches are memory inefficient in a real-time infrastructure as there can be infinitely large number of actions to offload a given task based on a given state of available resources.

To solve these issues, we can replace the Q-table with a deep neural network which can be trained to handle this state and give us the action with the highest rewards at each time epoch based on the current state of our network and the incoming offloading requests. This approach is also called a Deep Reinforcement learning or Deep Q-learning using a Neural Network (DQN).

3.2 Environment

In our proposed model as shown in Fig. 2, the task off-loader acts as the interface to our Deep Q-learning Neural network (DQN) agent. The off-loader, which also acts as the environment for the agent, sends the state (St) of the environment (cloud resources) to our DQN agent. It maintains the current status of the available network resources (edge, fog, and cloud) and receives incoming offloading requests from various devices across the network. For the sake of our simulation, we are creating the environment by taking the following parameters as input, D_i :

$$D_i \in R^N$$

a 1D array that represents the number of devices for edge, fog, and cloud, respectively. Then, $DevLimits_{i,j}$

$$DevLimits_{i,j} \in R^N \times R^D$$

a 2D array that represents the maximum and minimum compute limit which will be used during generation of these devices at random later during environment creation. Finally, we pass the $TaskLimits$, which is a tuple with the range of the compute power of the task which will be offloaded. Similarly, we also pass the C_i

$$C_i \in R^D$$

which denotes the networking cost of the respective devices. From this information, the environment creates an array $Devices_i$

$$Devices_i \in I^M$$

where M is the total number of generated devices and it contains the available compute power of device i. Once the network infrastructure is set up, the model then creates a queue of tasks $Tasks_i$

$$Tasks_i \in I^T$$

where, T is the total number of tasks to be offloaded and is generated based on $TaskLimits$ based as the input parameter as discussed above. $Devices_i$, $Tasks_i$, are sent as the state information for the DQN Agent.

4 Reinforcement Learning-Based Resource Allocation

4.1 DQN Agent

The agent passes the $Devices_i, Tasks_i$ appended together as state (S_t) as the input to the deep neural network which contains the input layer, hidden layers (ReLU, Linear, ReLU, softmax), and finally an output layer as given in Fig. 2. The output of the model is the predicted action (A_t) which the agent then sends back to the off-loader to perform the necessary action. The off-loader updates the $Devices_i$ array with the next state and also calculates the reward which is inversely proportional to the networking cost C_i . The off-loader executes the action, computes the reward (R_t) for the action, and sends the feedback to the DQN agent. The agent then updates the current state with the new state, takes a snapshot of the current experience, and pushes it into the memory queue. At the end of each episode, the agent then replays a portion of the past experiences to update the optimal policy of the DQN. This way the agent can optimally offload the tasks and improve itself in real time.

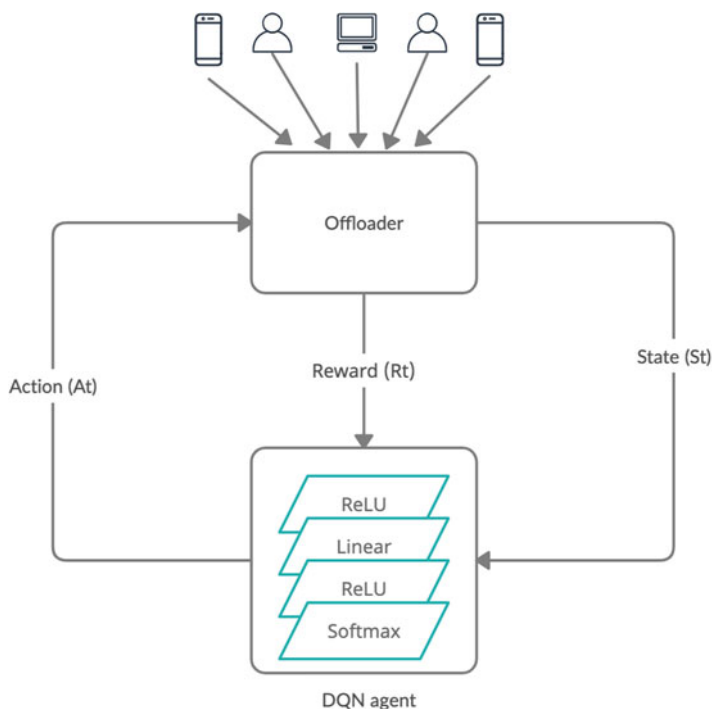


Fig. 2 Deep Q-learning Network (DQN) model

4.2 Problem Modelling

The DQN model follows the Markov decision Process (MDP) to formalize the offloading problem. It is defined by (S, A, R, P, gamma) where

S: set of possible states (which is the set of available edge, fog, and cloud resources in our case)

A: set of all possible actions (offloading to respective edge, fog, or cloud device)

R: distribution of reward for the given pair (St, At)

P: transition probability (calculated by the neural network instead of maintaining a Q-table for our proposed scheme)

gamma: discount factor

Algorithm: Deep Q-learning-based resource allocation

Input: state size, action size, $S_t, \gamma, \varepsilon, \varepsilon_{min}, \varepsilon_{decay}, lr$

Output: predicted action A_t

Initialization of the DQN agent with model parameters;

for e in range(*episodes*) **do**

Reset state in the beginning of each episode;

Generate new state from the off-loader;

for $task$ in *tasks* **do**

$A_t = agent.predict(S_t)$;

$S'_t, R_t = env.act(S_t, A_t)$;

$agent.memorize(S_t, R_t, S'_t)$;

$S_t = S'_t$;

end

$agent.replay()$;

end

In the above algorithm, the function $agent.predict()$ takes the input S_t and passes it to the DQN and return the action A_t . $env.act()$ performs the actual offloading task based on the prediction by the DQN and return the reward (R_t) and new state (S'_t). Then $agent.memorize()$ pushes the current experience into the memory deque, and finally $agent.replay()$ is used to fit the model on some random past experiences to optimize its policy.

The Q-function estimation used by the deep neural network can be expressed in terms of the Bellman equation as follows:

$$Q^*(S_t, A_t, \Theta_i) = \mathbb{E}_{S'_t \sim \varepsilon} [R_t + \gamma \max_{A'_t} Q^*(S'_t, A'_t) | S_t, A_t]$$

Loss function:

$$L_i(\Theta_i) = \mathbf{E}_{S_t, A_t \sim p(\cdot)} [(y_i - Q(S_t, A_t; \Theta_i))^2]$$

where

$$y_i = \mathbf{E}_{S'_t \sim \epsilon} [R_t + \gamma \max_{A'_t} Q(S'_t, A'_t; \Theta_{i-1}) | S_t, A_t]$$

We use a deep neural network here to estimate the transition function P as opposed to maintaining a traditional Q-table. Another challenge associated with reinforcement learning is the exploration–exploitation trade-off. The reinforcement learning model needs to choose between whether to exploit the current maximum reward greedily or explore new action knowing beforehand that those actions might result in suboptimal rewards. But exploration is necessary, especially in our case where there can be a lot of action state pairs to choose from, and therefore our model should be able to explore various actions first, before it starts to heavily exploit the best rewards. Therefore, we used a decaying exploration policy that provides a good exploration rate in the beginning that keeps reducing with time. In the following chapter, we will discuss the simulation parameters and result evaluation.

5 Results and Performance Evaluation

In this section, we are showing the results generated by the simulation. Table 1 shows the values of the simulation parameters used in the model evaluation. Here, gamma (γ) signifies the discount rate of reward, whereas epsilon is the initial exploration rate, epsilon (min) defines the lower bound, and epsilon (decay) is the decay factor by which exploration rate is decreasing as the time epochs increase. This is done to get enough explorations in the beginning and increasing the exploitation rate with time. This is done to take the exploration–exploitation trade-off into account. The memory deque is used to replay-based policy training at the end of an episode.

Table 1 Simulation and model parameters

Edge, fog, cloud devices	90, 30, 10
Edge device latency	0.1 ms
Fog device latency	0.2 ms
Cloud device latency	0.4 ms
Edge computing limits	2–4
Fog computing limits	8–10
Cloud computing limits	15–20
Memory (deque maxlen)	2000
Gamma (discount rate)	0.95
Epsilon (exploration rate)	0.8
Epsilon _{min}	0.01
Epsilon _{decay}	0.995
Learning rate	0.001

5.1 Resource Allocation

Since the cloud resources have high computational power followed by fog and then edge devices, we can see that the task execution time is inversely proportional to the computation power. For an optimal scheme, it is very important that model should offload tasks as to minimize the execution time as well as keep the networking cost low. Figure 3 shows that the execution time for proposed scheme is closer to that of cloud computing and lower than that of fog computing as time flows because the model is able to learn an optimal policy based on the constant feedback from the environment to offload compute-intensive tasks to the cloud resources and others to fog and edge in such a way that minimizes the overall computational time while maintaining low network latency.

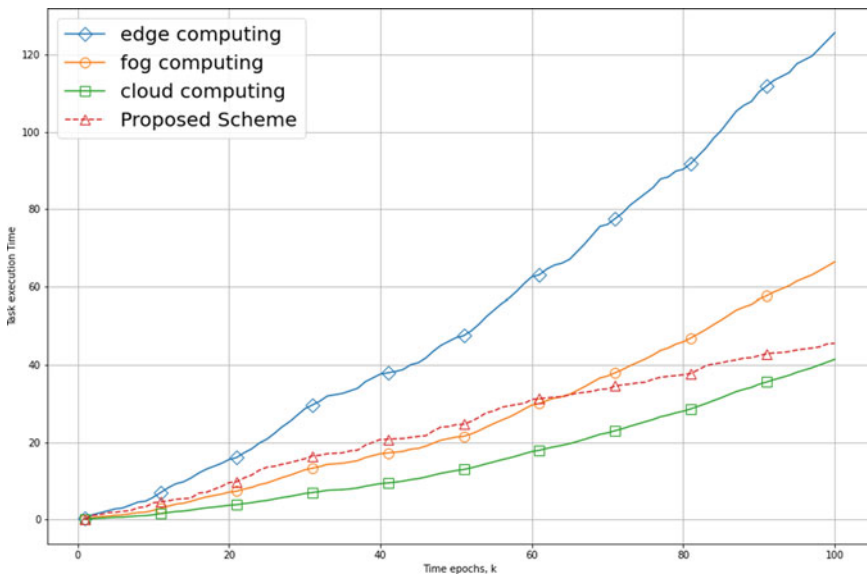


Fig. 3 Task execution versus time epochs for different computing schemes

5.2 Latency Analysis

The edge–fog cloud computing architecture as shown in Fig. 2 explains that the edge devices as the name suggests are at the edges of the network and therefore

have a very low networking latency or cost, whereas cloud devices have the highest costs due to the fact that they are placed in centralized locations far away from the end user or Internet of Things (IoT) devices. An optimal scheme should be able to balance between the execution time and networking cost to minimize the overall cost of task offloading. The results from our experiments show that the proposed scheme is able to optimally balance these objectives as shown in Fig. 4.

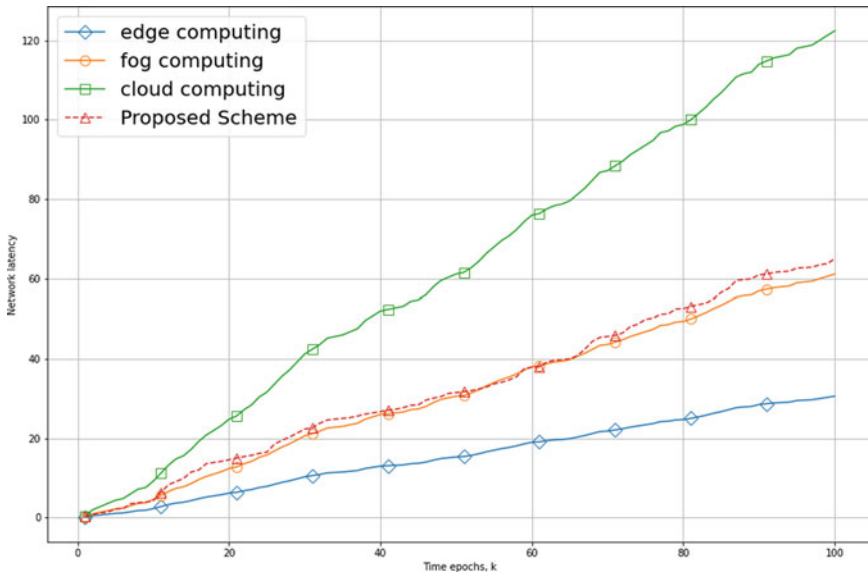


Fig. 4 Networking cost versus time epochs for different computing schemes

5.3 Model Performance

The following graphs demonstrate a detailed comparison of the model performance fine tuning with respect to different values of these model parameters.

Figure 5 shows that the model performs better when it has been assigned a higher exploration rate in the beginning. This also helps our argument of keeping a higher initial exploration rate and then using a decay policy on it is actually better than starting with a lower rate in the beginning. Also, Fig. 6 suggests us that the discount factor is better off at values around 0.85.

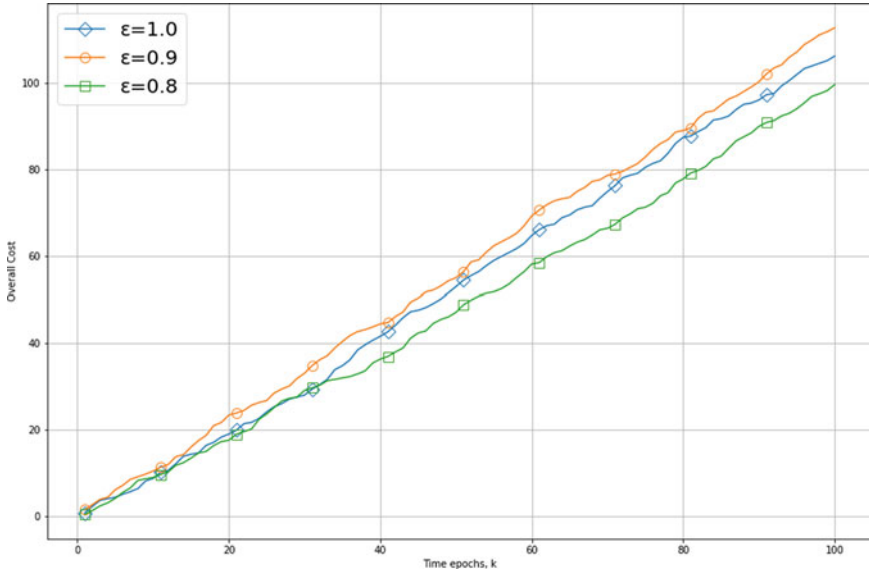


Fig. 5 Overall cost versus time epochs with different epsilon values

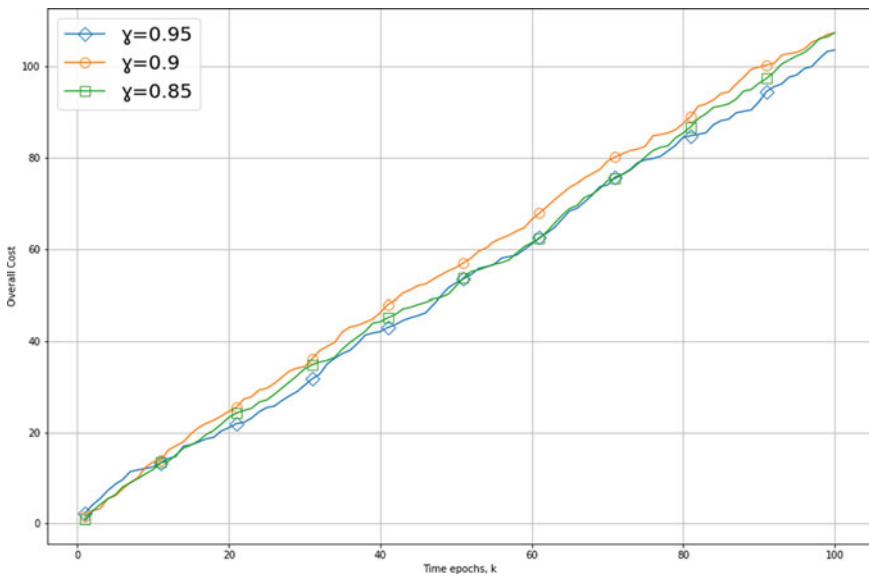


Fig. 6 Overall cost versus time epochs with different gamma values

For the exploration decay policy, Fig. 7 suggests that though we need to decrease the exploration policy with time, we should still allow a very small probability for further exploration. As a result, we should have epsilon values around 0.8, epsilon

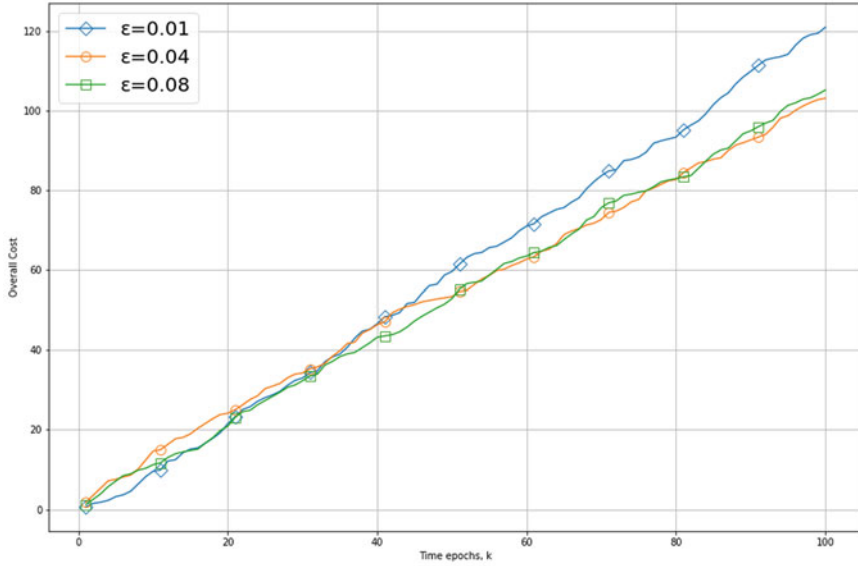


Fig. 7 Overall cost versus time epochs with different epsilon (min) values

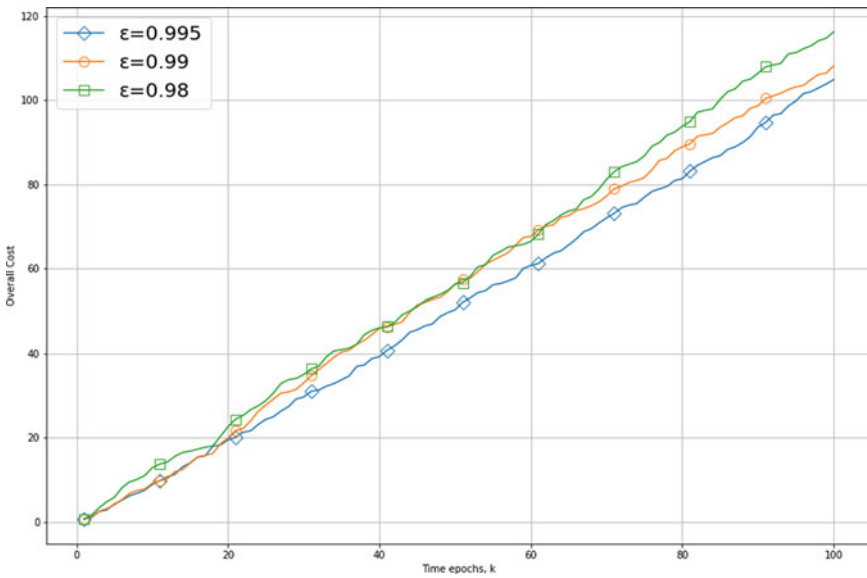


Fig. 8 Overall cost versus time epochs with different epsilon (decay) values

(min) around 0.04. Further, Fig. 8 shows the low decay speed with epsilon (decay) value 0.995. Also, having a learning rate of 0.001, Fig. 9 for the deep neural network gives a better optimisation policy as compared to higher values.

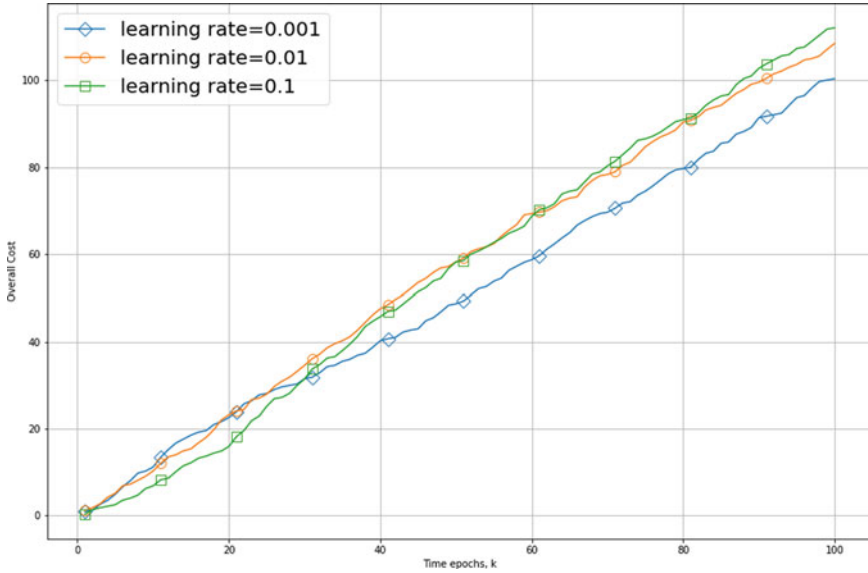


Fig. 9 Overall cost versus time epochs with different learning rates

6 Conclusions and Future Work

In this chapter, we discussed the Multi-access Edge Computing (MEC) with 5G Network Slicing capabilities that enables emerging use case requirements. We simulated the model using a deep Q-learning neural network (DQN)-based computational task off-loader that is able to give optimal performance by offloading resources to various devices in the edge-fog cloud network while achieving the task execution time closer to cloud computing while maintaining the networking latency close to that of fog computing. Further, this work may extend to implement in real-time scenario and to make MEC devices dynamic slice capable.

References

1. Afolabi, Ibrahim, et al. "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions." *IEEE Communications Surveys & Tutorials* 20.3 (2018): 2429–2453.
2. Gupta, R. K., & Sahoo, B. (2018). Security Issues in Software-Defined Networks. *IUP Journal of Information Technology*, 14(2).
3. Gupta, R. K., Choubey, A., Jain, S., Greeshma, R. R., & Misra, R. (2020, July). Machine learning based network slicing and resource allocation for electric vehicles (EVs). In *International Conference on Internet of Things and Connected Technologies* (pp. 333–347). Springer, Cham.
4. Hu, Yun Chao, et al. "Mobile edge computing—A key technology towards 5G." *ETSI white paper 11.11* (2015): 1–16.

5. Hassan, Najmul, Kok-Lim Alvin Yau, and Celimuge Wu. "Edge computing in 5G: A review." *IEEE Access* 7 (2019): 127276–127289.
6. Gupta, R. K., & Misra, R. (2019, December). Machine learning-based slice allocation algorithms in 5G networks. In 2019 International Conference on Advances in Computing, Communication and Control (ICAC3) (pp. 1–4). IEEE.
7. A. Filali, A. Abouaoumar, S. Cherkaoui, A. Kobbane and M. Guizani, "Multi-Access Edge Computing: A Survey," in *IEEE Access*, vol. 8, pp. 197017–197046, 2020, <https://doi.org/10.1109/ACCESS.2020.3034136>.
8. Li, J., Gao, H., Lv, T., & Lu, Y. (2018, April). Deep reinforcement learning based computation offloading and resource allocation for MEC. In 2018 IEEE Wireless Communications and Networking Conference (WCNC) (pp. 1–6). IEEE.
9. Gupta, R. K., Ranjan, A., Moid, M. A., & Misra, R. (2020, July). Deep-Learning based mobile-traffic forecasting for resource utilization in 5G network slicing. In International Conference on Internet of Things and Connected Technologies (pp. 410–424). Springer, Cham.
10. Y. Chen, Z. Liu, Y. Zhang, Y. Wu, X. Chen and L. Zhao, "Deep Reinforcement Learning-Based Dynamic Resource Management for Mobile Edge Computing in Industrial Internet of Things," in *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 4925–4934, July 2021, <https://doi.org/10.1109/TII.2020.3028963>.
11. L. Huang, S. Bi and Y. -J. A. Zhang, "Deep Reinforcement Learning for Online Computation Offloading in Wireless Powered Mobile-Edge Computing Networks," in *IEEE Transactions on Mobile Computing*, vol. 19, no. 11, pp. 2581–2593, 1 Nov. 2020, <https://doi.org/10.1109/TMC.2019.2928811>.
12. I. AlQerm and J. Pan, "Enhanced Online Q-Learning Scheme for Resource Allocation with Maximum Utility and Fairness in Edge-IoT Networks," in *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 3074–3086, 1 Oct.–Dec. 2020, <https://doi.org/10.1109/TNSE.2020.3015689>.
13. W. Zhang et al., "Deep Reinforcement Learning Based Resource Management for DNN Inference in IIoT," *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6, <https://doi.org/10.1109/GLOBECOM42002.2020.9322223>.
14. X. Xiong, K. Zheng, L. Lei and L. Hou, "Resource Allocation Based on Deep Reinforcement Learning in IoT Edge Computing," in *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1133–1146, June 2020, <https://doi.org/10.1109/JSAC.2020.2986615>.
15. X. Liu, Z. Qin and Y. Gao, "Resource Allocation for Edge Computing in IoT Networks via Reinforcement Learning," *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, Shanghai, China, 2019, pp. 1–6, <https://doi.org/10.1109/ICC.2019.8761385>.
16. N. Mohan and J. Kangasharju, "Edge-Fog cloud: A distributed cloud for Internet of Things computations," *2016 Cloudification of the Internet of Things (CIoT)*, Paris, France, 2016, pp. 1–6, <https://doi.org/10.1109/CIOT.2016.7872914>.

The Important Influencing Factors in Machine Translation



Debajyoty Banik

Keywords Natural language processing · Statistical machine translation · Factored machine translation

1 Introduction

Factored machine translation [1] is an extension of phrase-based machine translation which uses lemma and other morphological information (*POS*, *gender*, *number*, *etc.*) with the surface forms. This can solve data sparsity problem by exploiting linguistic features of the words. Translation models that operate on more general representations, such as lemmas instead of surface forms of words, can draw on richer statistics and overcome the data sparseness problems caused by limited training data. Different factors contribute differently in the translation process. For a word, there can be various linguistic features present. For example, in a Hindi sentence “चूहा(chooha):mouse,” word is given, so its various factors would be *lemma*: चूहा(chooha), *pos*:NN, *gender*:m(male), *category*:n(noun), *number*:sg(singular), *etc.* For all the words in source and target sentences, these factors are generated and mapped from source to target while training.

Researchers have explored factored-based machine translation for various language pairs like English→Czech [6], Russian–English [7], Hindi–English [8], *etc.*

2 Factored Machine Translation: Background

Phrase-based translation model [3] translates from source to target language by mapping the small phrases or say chunks of words without considering any lin-

D. Banik (✉)

Kalinga Institute of Industrial Technology, Bhubaneswar, India

guistic (morphological, semantic etc.) information. This information is useful while translating the source text to the target text. Augmenting morphological information of a word in the translation process might be helpful in resolving data sparseness and disambiguation between words with similar lemma but different morphological information. Researchers extended the phrase-based statistical machine translation (PBSMT) approach by adding linguistic information [1] like lemma, POS, gender, number, etc. Lemma is an important factor which overcomes the problem of data sparseness. Multiple inflected forms can have the same lemma. For example, verb “go” can have different inflected forms like “go,” “goes,” and “going.” These all word forms have a base form, that is, the lemma “go.” POS and other morphological features are also used to make differences in word forms as per the context of that word. In the sentence “*She feels well,*” the word “*well*” is an “*adverb,*” but in sentence “*She fell into a well,*” word “*well*” is a “*noun.*” Factored machine translation [1] process is broken up into *translation* and *generation* steps.

- (I) Mapping source side factors to the factors of target side.
- (II) Generation of surface forms at target side using target side lemma and morphological information (POS, gender etc.)

Decoding in factored machine translation model is complex in terms of computation. This is because more than one phrase table may be present depending upon the number of factors and translation mappings between these factors. Beam search starts with empty hypothesis and keep on generating new hypothesis based on the available and appropriate translation options. Highest scoring hypothesis or translation candidate will be the best output according to the model.

3 Various Linguistics Factors

In the study of natural language processing, the linguistics factors lead to great role. We are turning the light on various factors along with their relevance and importance.

3.1 Lemma

Lemmatization usually helps in grouping of words with the use of a vocabulary and morphological analysis, normally aiming to remove inflectional endings and to return the base or dictionary form of a word, which is known as the lemma. If a word ‘पढते’(padahte) appears in a sentence, then its lemmatization would be ‘पढ’(padha). So, ‘पढ’(padha) is the root form or lemma for its inflectional forms ‘पढते’(padhate), ‘पढना’(padhana), ‘पढती’(padhati), etc. In a morphologically rich language like Hindi, many surface forms may have the same lemma. So, in factored translation model,

first translating the lemma and then combining it with its morphological information would generate appropriate surface form as the output.

3.2 Part of Speech (POS) and Category

A word may appear in different meanings which can easily be distinguished by their POS forms.

For example, if there are two sentences

“राम/NNP मीठा/JJ आम/NN खाता/VM है/VAUX I/SYM” and राम/NNP एक/QC “आम//JJ आदमी/NN है/VM I/SYM,” then

we can see the word “आम” is *Noun (NN)* in the first sentence while *adjective (JJ)* in the second.

Category is the fine-grained POS by which the word can be identified explicitly.

A Hindi sentence

“राम एक आम खाता है ।” will be tagged as “राम/n एक/adj आम/n खाता/v है/v ।/punc.”

3.3 Gender

In Hindi, there is an agreement between subject and verbs in terms of their gender. To form a syntactically correct sentence, gender identification is required. “f” tag represents “female” and “m” represents “male”.

For example, in the sentence

“लड़की/f पढ़ रही/f है/ I/_ (ladaki padha rahi hai),” gender of the word “लड़की(ladaki)” and “रही(rahi)” is *female*, and for sentence “लड़का पढ़ रहा है ।(ladaka padha raha hai),” gender of the word “लड़की(ladaki)” and ‘रहा(raha)’ is *male*. Here, “लड़की(ladaki) (subject) has agreement with “रही(rahi)” (verb), while “लड़का(ladaka)” (subject) has agreement with “रहा(raha)” (verb).

3.4 Number

Number is an important factor for syntactic and semantic knowledge transfer. It may be “singular (sg)” or “plural (pl)”. For example,

“सभी/pl बच्चे/pl किताब/sg पढ़ते/pl हैं/pl I/_.”

4 Data Preparation and Experiments with Various Factors

Table 1 shows the statistics of data used for experiments. For performing experiments, we need to follow some steps.

4.1 Pre-processing

Available corpora for ILCI [3] and HindEnCorp [4] are not in factored forms. We tokenize¹ and truecase² the English data using Moses tool, and for tokenizing Hindi sentences, we use Indic-nlp³ library. There is no need of truecasing for Hindi data because in Hindi no capital or small letter scenario exists. We drop the sentences with more than 80 tokens and with input–output ratio more than 9.

4.2 Factor Generation

After tokenization, truecase, and cleaning, we generate factors for Hindi and English parallel sentences. For generating factors of Hindi sentences, we use shallow parser⁴ which is trained with health and tourism domains. Hindi is a morphologically rich language, and for a Hindi word, we generate POS, lemma, gender, category, and number as factors. We use NLTK⁵ tool for generating factors of English sentences. We generate POS and lemma using NLTK’s POS tagger and lemmatizer.

Table 1 Statistics of dataset used for the experiments

Dataset	HindEnCorp			ILCI		
	Sentences	Tokens		Sentences	Tokens	
		English	Hindi		English	Hindi
Training	265,472	3,244,055	3,323,117	47,999	822,485	859,129
Development	1000	16,308	15,153	1000	16,063	16,386
Test	1000	14,746		1000	18,499	

¹<https://github.com/moses-smt/mosesdecoder/tree/master/scripts/tokenizer>.

²<https://github.com/moses-smt/mosesdecoder/tree/master/scripts/recaser>.

³https://github.com/anoopkunchukuttan/-indic_nlp_library.

⁴http://ltrc.iit.ac.in/showfile.php?filename=downloads/shallow_parser.php.

⁵<https://www.nltk.org>.

Table 2 The ablation study with various factors for Hindi → English

Factors	HindEnCorp		ILCI	
	BLEU	METEOR	BLEU	METEOR
surf POS lemma	15.64	23.65	18.48	29.29
surf POS lemmalcategory	15.55	23.78	18.52	29.42
surf POS lemmalgender	16.46	23.52	18.78	29.61
surf POS lemmalnumber	9.41	20.17	19.11	29.98
surf POS lemmalgender number	6.72	18.47	9.08	17.27

4.3 Factored Data Preparation for Training

Hindi is morphologically richer than English. Here, for our experiment in each English sentence, we generate POS and lemma for all the surface forms and append those factors with surface forms using pipe symbol (|) in the format of *surface|POS|lemma*. For example,

Original training example: “*planning policies enabled more dispersed pattern.*”

Factored training example: “*planning|NN|planning policies|NNS|policy enabled |VBN|enabled more|RBR|more dispersed|VBN|dispersed patterns|NNS|pattern .|SYM|.*”

For Hindi sentence, we generate lemma, POS, category, number and gender factors using shallow parser tool and we perform experiments by adding these factors one by one. For example:

Original training sentence: "और पानी के टैंक".

Factored training sentence: “*OralCC|Oral|n|mlsg pAn|INN|pAn|n|mlsg ke|PSP |kAl|psp|mlsg tEMka|NN|tEMka|n|mlsg .|SYM|.|punc|_|_*”

These factors are in order of *surface|POS|lemmalcategory|gender|number*.

4.4 Training

For training our factored translation model, we use Moses [2] toolkit. We use 4-gram language model with Kneser–Ney smoothing [9] using IRSTLM [10]. Alignment is done by GIZA++ [5] with grow-diag-final and heuristic. Tuning is done using Minimum Error Rate Training (MERT) [11]. For training, we make a different combination of factors at the source (Hindi) side that can be seen in Table 2 and words in target (English) sentence would be in the form of *surface|POS|lemma*. At translation step source (Hindi) lemma will be mapped to the English lemma, and other morphological information (POS, gender, number, category) would be mapped to the morphological information of target (English) side. Finally, at generation step target (English) side lemma and POS would generate the surface forms as the final output using 4-gram language model.

5 Results and Analysis: Ablation Study

Some factors are more important than the other. Importance of factors also depends on the particular characteristics of data. Therefore, factors can vary for the different datasets. For empirical evaluation, we use two datasets: HindEnCorp and ILCI corpus. From Table 2, we can see that for HindEnCorp dataset score is highest by using surface+POS+lemma+gender and lowest by using surface+POS+lemma+gender+number. Behavior changes with ILCI data as we can see that for ILCI dataset score is highest by using surface+POS+lemma+number and lowest by using surface+POS+lemma+gender+number. It is clear from our experiments that increasing number of factors may not improve the quality. It depends on types of factors and characteristics of data. For an example, the translation accuracy in terms of METEOR for ILCI and HindEnCorp dataset follows similar behavior with sur|POS|lemma|gender|number, sur|POS|lemma, and sur|POS|lemma|category (increasing order). This behavior proves the common importance of different factors for various datasets, but it may not be true for all cases. Translation accuracy in terms of BLEU score [12] and METEOR [13] with sur|POS|lemma|number achieves best accuracy for HindEnCorp dataset, whereas it achieves worse accuracy for ILCI corpus. Detailed statistics is shown in Table 2. The BLEU is the most popular machine translation (MT) evaluation metric where METEOR is improved with synonym. The main reason of this type of behavior is different influence of the factors over different datasets. Various translated outputs (from HindEnCorp) for different factors are noted down in Table 3. sur|POS|lemma|number provides best result in terms of the adequacy and the fluency. sur|pos|lemma|gender is syntactically correct and adequacy is somehow acceptable. Translated output with sur|POS|lemma, sur|POS|lemma|category, and sur|POS|lemma|gender|number is less adequate and not syntactically correct.

Table 3 Case study with different factor combinations

	Output
<i>Source</i>	इन में से कई कंपनियाँ अमरीका में थीं
<i>Transliteration</i>	in mein se kae kampaniyaan amareeka mein theen
<i>Gloss</i>	These in from many companies America in was
<i>Reference</i>	Many of these companies were in America
<i>sur POS lemma</i>	In this, many of the company was in the United States of America
<i>sur POS lemma category</i>	In this, many of the company was in the United States
<i>sur POS lemma gender</i>	Many of these companies, was in the United States
<i>sur POS lemma number</i>	Many of these companies were in the United States of America
<i>sur POS lemma gender number</i>	There have been many of these companies to go into the United States

6 Conclusions

In this chapter, we studied the influence of linguistic factors in a phrase-based machine translation. We performed experiments over two datasets *HindEnCorp* and *ILCI* using various factors (lemma, POS, gender, number, category). We kept the factors at target (English) side fixed as `surface|POS|lemma` while altered the combinations of factors at the source (Hindi) side. We found that involving morphological information of word forms during translation affects the output and helps in improving the performance. Experiments show that the effect of factor combinations is not the same for both the datasets. It is also seen that better translation cannot be guaranteed by having any large combination of factors during translation. It heavily depends on the nature (morphological enrichment, language similarity, etc.) of language pairs between which translations are being performed.

References

1. Koehn, Philipp, and Hieu Hoang. "Factored translation models." Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL). 2007.
2. Koehn, Philipp, et al. "Moses: Open source toolkit for statistical machine translation." Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions. 2007.
3. Jha, Girish Nath. "The TDIL Program and the Indian Language Corpora Initiative (ILCI)." LREC. 2010.
4. Bojar, Ondrej, Vojtech Diatka, Pavel Rychlý, Pavel Stranák, Vít Suchomel, Ales Tamchyna, and Daniel Zeman. "HindEnCorp-Hindi-English and Hindi-only Corpus for Machine Translation." In LREC, pp. 3550–3555. 2014.
5. Och, Franz Josef, and Hermann Ney. "A systematic comparison of various statistical alignment models." Computational linguistics 29.1 (2003): 19–51.
6. Bojar, Ondrej. "English-to-Czech factored machine translation." In Proceedings of the second workshop on statistical machine translation, pp. 232–239. 2007.
7. Huet, Stéphane, Elena Manishina, and Fabrice Lefèvre. "Factored machine translation systems for Russian-English." In Proceedings of the Eighth Workshop on Statistical Machine Translation, pp. 154–157. 2013.
8. Dungarwal, Piyush, Rajen Chatterjee, Abhijit Mishra, Anoop Kunchukuttan, Ritesh Shah, and Pushpak Bhattacharyya. "The IIT Bombay Hindi-English translation system at WMT 2014." In Proceedings of the Ninth Workshop on Statistical Machine Translation, pp. 90–96. 2014.
9. Kneser, Reinhard, and Hermann Ney. "Improved backing-off for m-gram language modeling." In 1995 International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 181–184. IEEE, 1995.
10. Federico, Marcello, Nicola Bertoldi, and Mauro Cettolo. "IRSTLM: an open source toolkit for handling large scale language models." Ninth Annual Conference of the International Speech Communication Association. 2008.
11. Och, Franz Josef. "Minimum error rate training in statistical machine translation." In Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1, pp. 160–167. Association for Computational Linguistics, 2003.

12. Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. "Bleu: a method for automatic evaluation of machine translation." In Proceedings of the 40th annual meeting of the Association for Computational Linguistics, pp. 311–318. 2002.
13. Banerjee, Satanjeev, and Alon Lavie. "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments." In Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization, pp. 65–72. 2005.

Damaged Units Return Investigation in Printer-Producing Industry Utilizing Big Data



Gali Nageswara Rao, Yagireddi Ramesh , Bondu Venkateswarlu, and Bosubabu Sambana 

Keywords Big Data · Defective · Hadoop · Manufacturing

1 Introduction

Manufacturers in developed countries must significantly increase productivity to remain competitive with manufacturers in developing countries. Productivity is sometimes measured in goods produced per hour or employee, and productivity is the value of goods paid divided by the weight of resources used [2]. It can be increased in several ways: reducing cost, increasing efficiency of raw material use, or identifying the defects.

Printers are one of the most popular computer peripherals in the world. It is widely used everywhere in various fields, such as organizations, factories, schools, colleges, and homes. The main reason the printer has gained so much popularity is because of its portability. The printer nowadays comes with the size of the mobiles. Printers accept the text or image from the computers and transfer that information pixel by pixel on the sheet of paper, and the printer languages can achieve this. Printer languages are nothing but the commands forwarded to the printer to tell

G. Nageswara Rao · Y. Ramesh
Department of Information Technology, Aditya Institute of Technology and Management,
Tekkali, Andhra Pradesh, India

B. Venkateswarlu
Department of Computer Science and Engineering, Dayanidhi Sagar University, Bengaluru, India

B. Sambana (✉)
Department of Computer Science and Engineering, Lendi Institute of Engineering and
Technology (A), Vizianagaram, Andhra Pradesh, India

Jawaharlal Nehru Technological University Kakinada, Kakinada, Andhra Pradesh, India

Department of Computer Science and Engineering, Indian Institute of Information Technology
Kota, Jaipur, Rajasthan, India
e-mail: 2021krpc1001@iiitkota.ac.in

the printer the document's format to be printed. Printer languages popular used are Postscript and printer control language. These commands manage the color, size, font size, etc. A wide range of printers is available concerning the brands, shape, size, capability of processing papers, etc.

HP Inc. is one well-known brand in the printer manufacturing industry. They were successful in delivering good-quality printers to the customer. HP Inc. is well known for manufacturing energy-efficient printers. HP company accounts for 20% of printers and other peripherals business. As the percentage is higher in manufacturing printers, they have many manufacturing locations worldwide that manufacture thousands of printers of various types and models. Data plays a vital role in the manufacturing industry as it may contain details of manufacturing locations, shipping data, warehouse data, customer information, etc. Big Data technologies come into the picture to maintain all the knowledge of the manufacturing locations, customer details, and various other information. Big Data helps the manufacturer to predict, foresee risk, understand their customer needs, and can also use to monitor defective products which are delivered to the customers. Big Data has become tremendously popular in many fields nowadays.

It has a wide range of applications in retail, healthcare, financial services, government, etc. As in this globalization time, there are some tough competitions between the manufacturing companies to provide the best quality products to the customers. As Big Data has a wide range of possibilities, it can be used in the manufacturing industry. Analyzing the Big Data in the manufacturing industry can reduce processing flaws, improve production quality, increase efficiency, and save time and money.

Big Data has many technologies such as Hadoop, Hive, Sqoop, Flume, and NoSQL. They can be used to store and analyze the data. This paper approaches for few of the technologies like Hadoop, Hive, Sqoop, and Flume to explore the defective units returned by the customers to the resellers.

(a) Hadoop

Hadoop is one of the frameworks which is written in java language. These frameworks are the collections of software utilities used to manipulate a large number of data in less time in the distributed environment. Hadoop was started by the Apache software foundation in the year 2006, and Hadoop is open-source. The Hadoop ecosystem consists of storage, and this storage is termed HDFS (Hadoop Distributed File System). The processing part in Hadoop is done by the Map Reduce programming, written in java language.

Hadoop can be configured with a single node cluster and a multi-node cluster. The single-node collection has only one machine, whereas the multi-node collection has multiple devices. On a single node cluster, all the daemons like Namenode, Datanode, task tracker, and job tracker run on the same machine, wherein the multi-node collection follows master-slave architecture. The device that acts as a master runs the name node daemon while the other machine acts as a slave machine, which runs the daemon like a data node.

(b) Sqoop

Sqoop is a tool used in transferring the data from the traditional relational database management system to the Hadoop distributed file system, and it is a command-line interface tool. In Sqoop, the moving of data occurs with the import command, and with the help of Sqoop, we can share the data from HDFS to MySQL with the export command used in Sqoop.

(c) Flume

A flume is a tool used for data ingestion for aggregating, collecting, and transporting a large amount of live data, such as events and log data stored in files from web servers to HDFS. The flume tool consists of agents, and it contains three components source, sink, and channel.

(d) Hive

Hive is a tool in data warehouse infrastructure to process structured data in the Hadoop ecosystem. Hive is used as a SQL type of query to do some of the Map-Reduce operations in Hadoop. Hive is mainly used to query and analyze the data present in the Hadoop distributed file system. Hive is designed for OLAP, and it uses SQL language known as HiveQL or HQL.

(e) Qlikview

Qlikview is one of the visualization tools. It helps visualize the various data from distinct origins. It supports multiple formats for connecting different sources, such as JDBC, odbc, and table files. The hive connection is even possible in the Qlikview, and Qlikview connects to the hive server, and we can execute the query in the Qlikview graphical interface.

This paper is about identifying and reducing the causes of defects in the printers by analyzing data from returned printer units. The general path of the team from “Factory to Customer” is “Manufacturing location – Shipped to Warehouse – stocked in Warehouse – shipped from Warehouse to Resellers (as per supply chain requirements) – from reseller to end customer” after which if a customer finds unit to be defective, will return the printer to the reseller.

The reseller reports the details of the defective unit back to the company for claims for refunds or restocking of the same inventory. From the data collated from resellers, individual printer unit info is available, from which the patch can be traced back to the manufacturing location.

Data from manufacturing locations, warehouses, and shipping companies are stored in an RDBMS. The data provided by these sources are in compressed format, i.e., they have information in thousands of records sent in one shot, either daily or weekly, or monthly. Data from resellers is live data that used to be stored in the same RDBMS, after which all the data used to be queried using available MySQL methodologies.

As the comprehensive data from all sources grew more extensive, the need for HDFS also arose, which was one of the basic needs for this project. Data had to be

migrated from RDBMS to HDFS, and also the live data from resellers was updated directly to HDFS. Analyzing patterns like the increase in defective units from one manufacturing location or reported from lots shipped by a particular shipping company will help find the probable root cause of defects.

2 Literature Review

As we know, Big Data is one of the fast-emerging techniques nowadays. As the data size is increasing at a rapid pace, the need for Big Data arises. Big Data consist of many technologies for various computation. Big Data consists of a massive volume of structured and unstructured data, so to make these data in the proper and organized form, Big Data technologies are used. As this paper is focused more on using Big Data technologies in the manufacturing industry, manufacturers in the manufacturing sector have vast data, and these data come from various resources like manufacturing locations and warehouses. The data continuously grows faster from these locations as the new products are manufactured. Big Data technology like Hadoop can be used to handle this data efficiently [1].



Fig. 1 Areas of greatest benefits for manufacturing/operations

Figure 1 shows how Big Data is used in the manufacturing industry. The survey was asked by the Tata Consultancy Services to manufacturers to rate how Big Data had helped them in multiple fields in the manufacturing industry [4]. Hadoop in the manufacturing industry is very rarely used. Hadoop in the manufacturing industry makes the handling of extensive data very easy. It can be used in the printer manufacturing industry to analyze the returned printer by the customer. HP Inc. holds 20% of its business on the printer and other peripherals [2]. It mainly covers the six groups: buyer groups, individuals, small businesses, large businesses, governments, and educational institutions.

Intel company has been using Big Data for its manufacturing process. The company has to test the chip whenever the chip comes onto the production line, and the chip has to go through the 19,000 test. So, by using Big Data for testing purposes, the company successfully reduced the test for quality assurance. This happens with predictive analysis using Big Data. Intel successfully analyzed the data from the manufacturing process to cut down test time and focus on specific tests. This implementation saved \$3 million in production costs for a single line of Intel processors [3].

Tata Consultancy Services nearly earns \$2 billion through business analytics. The company generates most of its income by manufacturing products as per the order. With the implementation of Big Data analytics, the company was able to analyze the behavior of repeat customers. The outcome is critical to understanding how to deliver goods quickly and profitably. Much of the analyses centered on making sure substantial contracts were in place. The company was also benefited as they were able to shift to lean manufacturing; lean manufacturing determines which products are viable and which ones need to be dumped [4].

3 Methodology

(a) Architecture (Fig. 2a):

(b) Implementation:

The project has three essential Big Data components:

1. Moving data from traditional RDBMS to HDFS (SQOOP).
2. Storing live data from resellers to HDFS (FLUME).
3. I am querying the data from HDFS using HIVE to provide useful analytical information for the decision-makers (HIVEally connected to Click View).

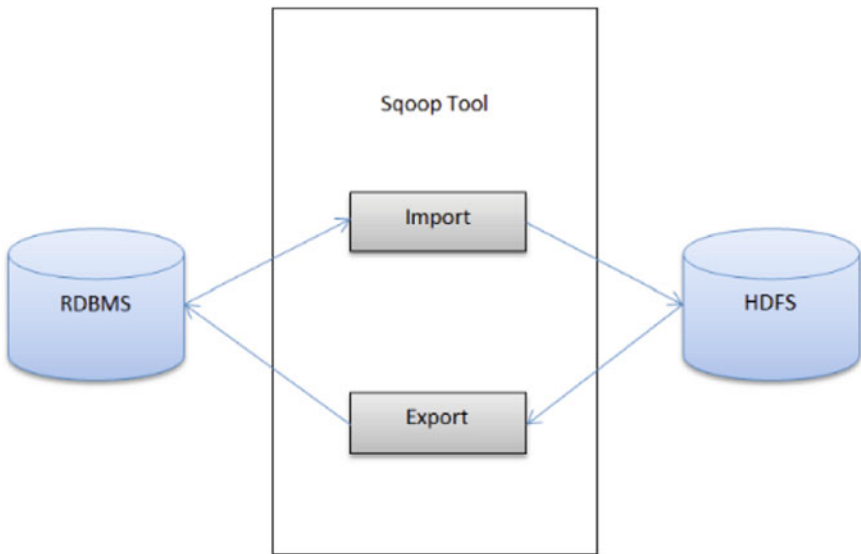
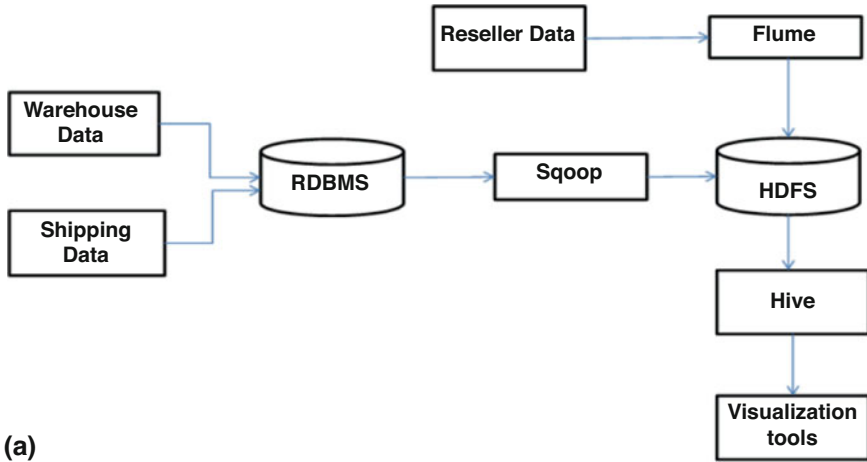


Fig. 2 (a) Architecture of HDFC. (b) Sqoop architecture

3.1 Moving Data from Traditional RDBMS to HDFS (SQOOP)

Firstly, the data is stored in MySQL database with 6 different tables mainly manufacturer_id, manufacturer_data, shipment_data, shipping_company_id, warehouse_data, and warehouse_id. The records that this table hold is transferred to

HDFS with the help of the Sqoop tool. The Sqoop is a tool that uses the command-line interface, so Sqoop consists of the import command that can be connected to the MySQL database and the target path that signifies where to store the data in Hadoop distributed file system [5]. Example of the Sqoop import command is as follows:

```
Sqoop import connect jdbc:mysql://192.168.30.1:8080/project-table manufacturer_data --username root -P --target-dir/Sqoop/import/manufacturer_data -m 1
```

The above command will connect to the machine with IP address 192.168.30.1 and the port no. on which the MySQL is running. Then it will extract the data from the project database, which has manufacturer_data as a table name, and push the data to the target directory. We can automate this process by using cronjob; it is a script to automate the import process of Sqoop. The hand takes the date and time and the approach to execute (Fig. 2b).

This process can be called batch mode as the data is already present in the database. Another technique that can take the data from the web server logs is live data [6].

3.2 Storing Live Data from Resellers to HDFS (FLUME)

Data is collected from resellers via a web portal made available to all resellers; the data from the web portal is in the form b server; with each log entry concerning each defective unit claim, the flume agent will send the data to the from the webserver to the server where Hadoop is installed. Another flume agent is configured in that server, receiving the data and putting it into the HDFS (Fig. 3).

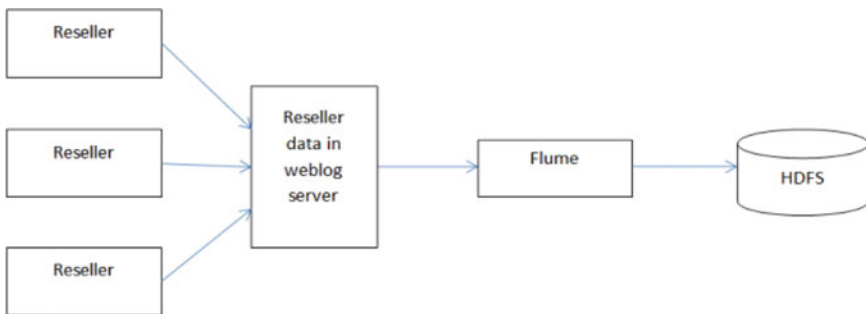


Fig. 3 Processing of flume once data is put into the HDFS, loads the data into the hive using the Regex serve

3.3 Querying the Data from HDFS Using HIVE to Provide Useful Analytical Information for the Decision-Makers

In this step, the data which is in HDFS is pushed into the HIVE. HIVE consists of the table or schema. The data is loaded in hive schema and queries the data to get useful analytical information—this information we can visualize with the help of many tools readily available in the market. The data thus stored in Hive from RDBMS using Sqoop and from Web Server using Flume is now available for querying to find useful analytical information. The decision-makers/stakeholders then use this information to make necessary changes like increasing the number of quality checks at the warehouse or reducing the number of days units are stored in separate warehouses to avoid defects from moisture collation, etc. The data is partitioned and bucketed for optimized querying performance [7].

4 Result Analysis

The project results in analyzing the defective units returned by the customers using some of the Big Data technologies. Using Big Data technologies, we can achieve the proposed objective with ease. The approach uses the data present in traditional RDBMS to process and analyze the data. Another method is to get the live data from the web server logs from the reseller web portal [8].

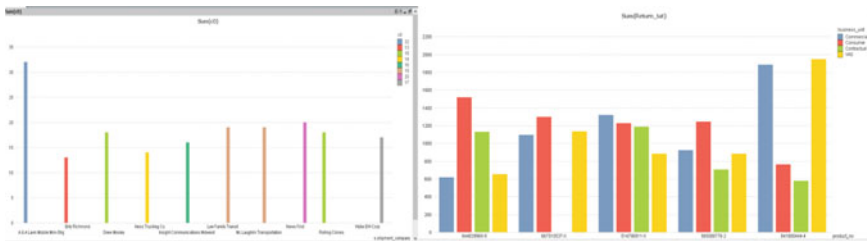


Fig. 4 (a, b) Analysis of the number of days the product is returned in a different business unit

Figure 4 represents the analysis of the number of days the product returns in a different business unit. The above figure serves the number of days on the y-axis and the various product numbers represented on the x-axis. The different colors in the bar represent the various business units in that particular product number. In product number “644039965-9,” the commercial business unit returns the product for 600 days. So similarly, the product number 841800444-4 had returned after 1945 days, so we analyze that this product is better than others [9].

4.1 Models with the Highest Defective Returns in Any Particular Business Unit

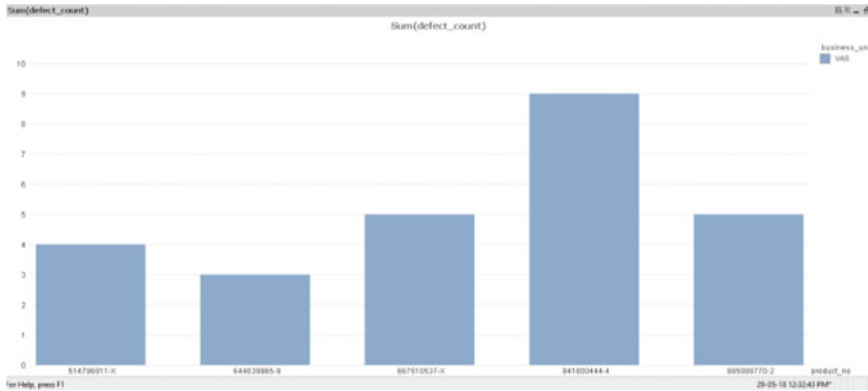


Fig. 5 Models with the highest defective returns in any particular business unit

Figure 5 represents the product numbers with the highest defective returns in a particular business unit. The above figure depicts the different product numbers on the x-axis and the count of the defects on the y-axis. This output is carried out on the VAS (value-added service) business unit. So the product number “514796911-X” returns the number of defects is 4 in the VAS business unit. So we can analyze that product number “841800444-4” returns the highest defective return, 9 in a “VAS” business unit [10].

4.2 Shipping Companies with High Defective Returns

Figure 5 represents the shipping companies with high defective returns. The above figure depicts the various shipping companies on the x-axis and the wrong counts which that particular shipping company had shipped on the y-axis.

The shipping company named “News First” had shipped the 20 defective pieces. Similarly, we can analyze that the shipping company “A &A Lavin Mobile Mini-Strg” had sent the highest faulty return, that is 32 as compared to other shipping companies [11].

4.3 A Most Recurring Issue for Any Particular Model

The Fig. 6 represents the most frequent issue for any specific product number. The figure consists of the x-axis representing the different defects and the y-axis representing the count of those defects.

The product numbers are repeated in numbers to get the single product number and calculate the number of faults through the returned products by the customer. For example, the figure shows that the product number “644039965-9” has five different defects.

Among the five defects, which are more in numbers are indicated with the assistance of the various bar heights with the counts. So we analyze that the deficiency “wifi printing takes too long” is 7 in the product number “644039965-9”, which is more than other defects.

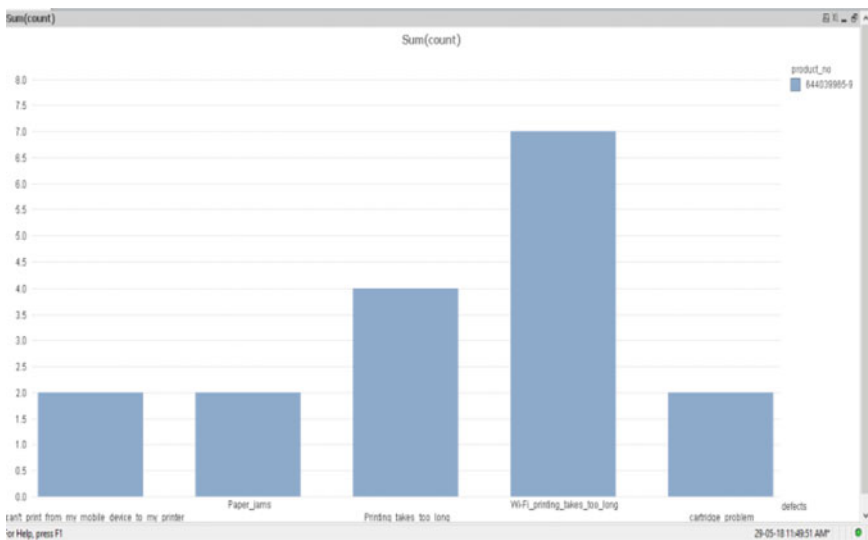


Fig. 6 A most recurring issue for any particular model

4.4 Warehouses with Max Returns for Particular Business Units

The Fig. 7 represents the warehouses with maximum returns for the particular business unit. The warehouse companies are responsible for the defective products

supplied to the end-user from a specific business unit. The x-axis serves the various warehouse companies, and the y-axis serves the number of product returns for particular warehouse companies. The warehouse company named “Kiehn group” returned the four defective pieces. So we analyze that the warehouse company “Torp Inc.” produces the highest bad pieces compared to other warehouse companies.

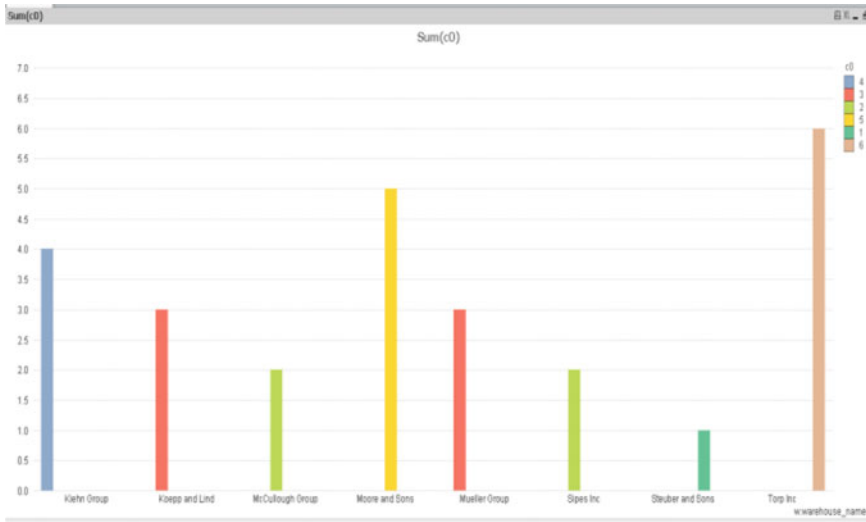


Fig. 7 Warehouses with max returns for particular business units

4.5 Models with the Lowest Return TAT

The Fig. 8 represents the models with the lowest return tat. The x-axis serves the various product numbers, and the y-axis serves the number of days it is returned to the reseller. For example, for the product number “667510537-X,” the number of days of return is 50. Similarly, we can analyze that the product number “9985088770-2” returns in fewer number days to the reseller.

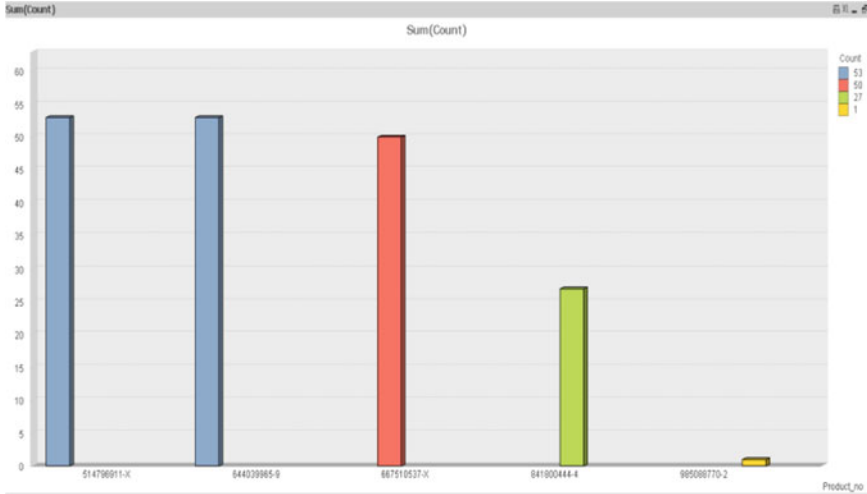


Fig. 8 Models with lowest return TAT

5 Conclusion

The analysis of the defective unit return analysis from the customer for the printer manufacturing industry is explicitly explained in this thesis report. The data from reseller data, which contains product number, serial number, etc., data can be analyzed to get the number of defects in the printer facing in more numbers. So, we can explore the data collected from various sources and can be diagnosed with the help of the Big Data technology Hadoop. The live data can also be fetched and analyzed with the help of the python scripts and using the tool flume.

Big Data has tons of features that can be used according to our own needs to retrieve and analyze the data per the requirement.

References

1. Introduction to manufacturing process Osama Mohammed Elmardi Suleiman Khayal, Nile Valley University, July 2017
2. Hilton, R.W., M.W. Maher, F.H. Selto, and B.J. Sainty. 2001. *Cost Management: Strategies for Business Decisions*. Canadian edition. Toronto, ON: McGraw-Hill Ryerson Limited
3. <https://www.slideshare.net/cloudera/manufacturing-cloudera-sessions-47003270>
4. An Analysis of the Printer Industry and Hewlett-Packard's Competitive Positioning
5. Time Value of Commercial Product Returns -V. Daniel R. Guide Jr.1, Gilvan C. Souza, Luk N. Van Wassenhove, Joseph D. Blackburn
6. Analysis of Web Server Logs Using Apache Hive to Identify the User Behaviour – B Shalem Raju, Dr. K. VenkataRamana

7. Mr.YogeshPingle, VaibhavKohli, ShrutiKamat, NimeshPoladia, (2012)—Big Data Processing using Apache Hadoop in Cloud System||, National Conference on Emerging Trends in Engineering & Technology.
8. <https://www.slideshare.net/cloudera/manufacturing-cloudera-sessions-47003270>
9. An Analysis of the Printer Industry and Hewlett-Packard's Competitive Positioning
10. <http://www.ingrammicroadvisor.com/data-center/4-big-data-use-cases-in-the-manufacturing-industry>
11. <http://sites.tcs.com/big-data-study/manufacturing-big-data-benefits-challenges>

Colorization of Grayscale Images: An Overview



Medha Wyawahare, Tejas Kolhe, Akshay Joshi, Yogesh Metkari,
and Kshitij Patil

Keywords Colorization · Image processing · Deep learning · Convolutional neural networks · Generative adversarial networks

1 Introduction

There is a growing interest in colorizing various types of grayscale images dating back before the discovery of the color camera. Historical pictures, natural images, and astronomical photography all can gain from image and video colorization. The process of image colorization is about adding plausible colors to grayscale images, to make them more realistic. This will help in gaining crucial information from many historical images, CCTV footage, medical images, videos, etc.

There are various types of colorization techniques presented till now showing great results. From a broad perspective, colorization is classified into three categories: hand coloring, semi-automated coloring, and automatic coloring. Hand coloring is the approach of manually adding color to a monochrome image. It is a time-consuming method. For colorizing whole video or classic movies it will be a tedious job. So, to avoid this, semi-automatic and automatic colorization comes into the picture. The recent advancement in the field of image processing, computer vision and deep learning [5] make the automatic colorization of images much easier.

Image colorization is an intelligent and sophisticated task. The semi-automatic tasks infer to the method in which human intervention is involved. Efforts have been taken to eliminate human intervention and make it a fully automatic process.

Colorizing a grayscale image entails providing three-dimensional (RGB) pixel values to an image that only varies along the axis. Luminance refers to the amount of light radiated by any color in an image. It is a metric for describing a

M. Wyawahare · T. Kolhe (✉) · A. Joshi · Y. Metkari · K. Patil
Department of Electronics and Telecommunication, Vishwakarma Institute of Technology, Pune,
India
e-mail: medha.wyawahare@vit.edu

color's perceived brightness. In picture editing, luminance is also used to combine brightness and change the ambiance of an image.

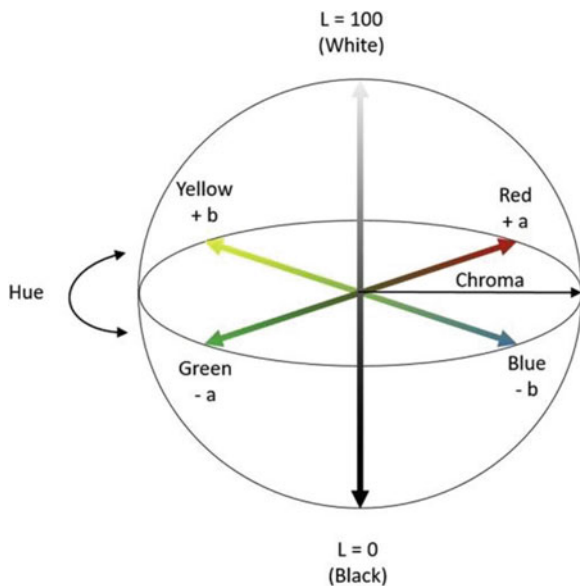
There are several different methods reported for colorizing the grayscale images. This paper focuses on comparing the effectiveness of some of these approaches. For this purpose, we have implemented them on a common dataset and compared the results.

2 Prerequisites

2.1 CIELAB Color Space

In all the reported approaches, the conversion of images in the LAB color space is a common step. The CIELAB color system, usually known as CIE L^* a^* b^* , illustrates the quantitative connection between colors on three axes: Luminance is represented by the L^* value, and chromaticity coordinates are represented by a^* and b^* [6]. In Fig. 1, L^* is shown on a vertical axis with values ranging from 0 (black) to 100 (white) on the color space diagram. The a^* value denotes a color's red-green component. On the b^* axis, the yellow and blue components are shown.

Fig. 1 CIELAB color space diagram [6]



This color space covers all the colors present in the human perception range. To color a grayscale image with plausible colors, we need all the possible colors for the pixels without interfering with the luminance of the original image. Hence, CIELAB

space will be useful. Most of the approaches revolve around converting images in CIELAB colorspace, performing feature extraction, and transferring chromatic and luminance values to the image in the same lab space.

2.2 Overview

The image colorization problem is more of an image enhancement problem than an image restoration problem. This problem statement is not new and has been in focus for the last few decades. Most early solutions presented were conventional machine learning approaches and mostly involved human intervention which reduced the time consumption significantly. Recently, many solutions are provided using deep learning which are automatic, i.e., involving no human intervention [11].

All the automatic methods have some common concepts of converting images in CIELAB color space, extracting features, mapping the luminance values and transferring the color information in the grayscale image, and then converting it back in RGB space. They all differ in the model structure, loss function, etc., through which they all achieve the task of image colorization.

3 Different Approaches

In this study, we have provided a detailed review of some approaches, implemented them, and compared the results we got on common datasets. Some of the following approaches have been reported.

3.1 Statistical Approaches

Transferring Colors to Grayscale Images

Adding chromatic value to a grayscale image has no fixed solution or algorithm, but this approach provides for minimum human interference [1]. Instead of coloring every component of an image this approach transfers the colors between the images based on similarities in luminance and texture information. Further, the procedure is enhanced by using rectangular swatches which allows user to match areas of two images. This technique can be applied to different types of images and video, provided that luminance and texture are significantly distinct.

As we know that colorizing a grayscale image is transferring the three-dimensional information to an image that varies along a single axis of luminance or intensity. This is the concept introduced by Reinhard et al. [2] in their paper. The authors state that we can transfer colors from one colored image to other.

The grayscale image is represented in one-dimensional distribution, so, it can match with colored images through the luminance channel only. To guide the exact matching areas for the colors to transfer we use pixel neighborhood or the texture around the pixel areas. The reason behind this is there can be more than one part in the image where the luminance values are the same but they might be representing different parts of the image. So, along with luminance matching author suggests looking for pixel neighborhood as well. Once a pixel is matched, the three-dimensional color information is transferred from source to target image, but the original luminance values are retained. The authors suggest the use of a few swatches to simplify the matching process in some problematic cases.

The general process for color transfer used here has the following steps as shown in Fig. 2.

Each image is converted into the lab color space. Then to select a small subset of pixels as samples, use jittered sampling.

Then scan the color image or source image for the best matching pixel using neighborhood statistics to transfer the color.

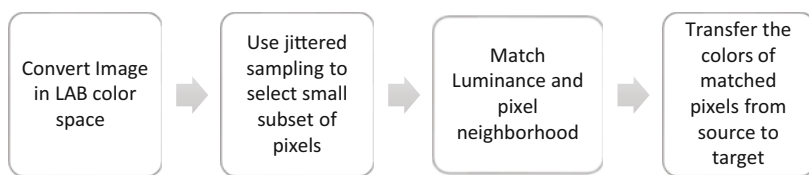


Fig. 2 Flowchart of the algorithm proposed by Welsh et al.

We have to take the weighted average of pixel neighborhood statistics and luminance values to find the best match for color transfer. It is also known as the global image matching technique.

The same color transfer technique can also be applied using swatches which will transfer the colors from the source swatch to the target swatch. Swatches are the areas users need to select to work on or do the calculations on. After coloring some pixels in the target swatch, we can use the colored pixels to color the remaining part of the image which is not colored. This method can be used for complicated cases where we are not able to match the exact pixels for color transfer. Figure 3 shows the result presented in the research.



Fig. 3 Results of Welsh et al. [1]

The advantage of using swatches is that we can color any region of the image according to the similarities in the texture of the source and target image. So texture and luminance play a major role here. But this can be achieved by compromising the automation of the process.

Automatic Colorization of Grayscale Images

Austin Sousa et al. [3] proposed automatic colorization of grayscale images approach using machine learning technique. This approach tries to provide exact or true color to the image rather than apply any plausible color to it. In their studies, the authors suggest two classes of colorization methods.

Based on image data, a color is assigned to a pixel according to its intensity.

Image is segmented into different regions to be assigned by a single hue. Figure 4 shows the steps followed in this method

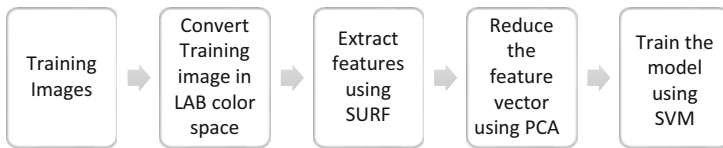


Fig. 4 Flowchart of the algorithm proposed by Austin Sousa et al.

In their studies, they analyze the color distribution of the training images provided, followed by the prediction of colorization of the single grayscale image per-pixel basis. In their approach, they use graph cut algorithm, which attempts to align color variation to intensity boundary.

All the training images are converted into lab color space before using them. The luminance channel is used to create a grayscale image from which features can be retrieved. The two 8-bit values (a,b) are used to represent color information, allowing for 2562 different colors. After that, using k-means, they reduce the color space to a reasonable range of colors, between 8 and 32, and use this color mapping for the final output. They select the small subset of training pixels randomly from each training image data to match the pixels.

In the step of feature extraction, the authors extract three different parameter classes, which are SURF (speeded up robust features), FFT, and localized mean and variance. A 20×20 window is used to obtain SURF descriptors for each pixel. The magnitude of the 2D FFT calculated across the same 20×20 grid is then added to this data.

Finally, they compute the mean and variance, resulting in a 786-point feature vector that is unmanageable. They reduce this set of features from 786 to 32 using PCA.

A Gaussian kernel is used for classification that consists of a series of support vector machines, one for each color bin. For each color bin, the array of SVMs conducts a binary classification, predicting whether a pixel is corresponding to color

or not. The margin of each SVM is used by the authors for the probability of each color being the true color. These probabilities are then employed by the graph-cut algorithm in post-processing.

To sum up, to colorize the grayscale images using this method, the authors suggest starting with calculating the identical 786-point feature vector for each pixel and in the training phase, reducing the dimension of this feature vector using PCA. This reduced feature vector is to be sent through the SVMs to extract the geometric data. For each of these feature vectors, we then calculate the margin which tells if the pixel is having correct color or not.

The output is then processed through a median filter to remove small regions of outlier colors in the image. In the study, using a compute cluster and MPI, authors performed many grid searches on a variety of test photos. The best-looking instance seems to coincide with the least colorization error in most situations; however, the selected parameters varied from image to image and failed to exhibit any trends. We can see in Fig. 5. that the model gets confused between sky and mountain because of low texture features.

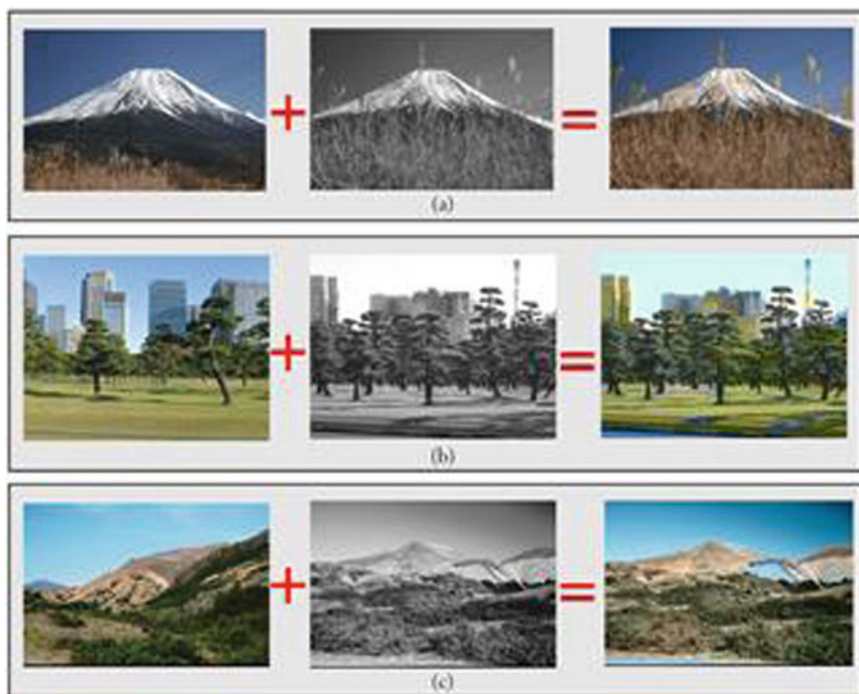


Fig. 5 Results in (a) and (b) are showing reasonable performance on landscape images. However, in some images like in (c), the model gets confused between sky and mountain because of low texture features [3]

3.2 *Deep Learning Approaches*

The deep learning approach drives the process toward making it more automatic. CNNs have shown great results in the field of computer vision by learning feature representation and replaced most of the feature engineering parts. We can train an end-to-end model on grayscale images to generate colored ones. With L2 loss over the images, the model can learn to generate images more similar to ground truth. But L2 imposes an averaging effect on the image resulting in dimmer or causing patches of different colors for the same area of the image [7].

Goodfellow et al. [13] proposed GANs which can be used to generate images. The loss function in adversarial learning chooses a single color to fill rather than averaging it, which gives better output than just using CNNs.

Colorization Using Encoder-Decoder Structure of CNNs

In this paper, Iizuka et al. [10] propose a data-driven CNN-based approach for applying colors to grayscale images, as shown in Fig. 6. This is a fully automated approach as well. The approach proposed by the authors uses the combination of global and local features which will be extracted from the training image data. The global features are extracted from the whole image which tells about the overview of the image like if the image is taken indoors or outdoors, whereas the local features, which are extracted from some image patches, tell about the objects or textures present in the image related to the global features. This helps the model to understand the plausible colors to be applied in the image [8, 9].

The local features extracted have a bias about applying some colors to the different objects present in the image due to the image prior, from global features. They select plausible colors by themselves like they will not apply colors of sky or grass to the image of global features indicating that the image is taken indoors.

For this purpose, the authors have proposed an architecture that extracts global and local features together and using them performs the final colorization.

The model consists of four different components: a low-level features network, a mid-level features network, a global features network, and a colorization network. These four components are part of a single network. The working of the model with these four components is explained as first a common set of low-level features are extracted and using these the mid and global level features are also extracted. These extracted features are fused and given as input to the colorization network which gives the final colored image as output.

The model proposed is a six-layer CNN network that receives the low-level features directly from the image. The weights are shared between the global and mid-level feature networks. After these, a four convolutional layer and three fully connected layer networks calculate the global features which result in a 256-dimensional feature vector, and two convolution layers to calculate the mid-level features are implemented which result in 256-channel features.

The fusion layer combines the global and local features, processing them through a single layer network.

These fused features are passed to the colorization network as input. The colorization network consists of upsampling and convolutional layers placed alternately. The upsampling uses the nearest neighbor technique, which is performed till the output is half the size of the original. After every upsampling layer, the output is twice as wide and twice as tall. The output layer of the colorization network is a convolutional layer with a sigmoid transfer function which gives the final chrominance as output. This chrominance is combined with the intensity of the image and results in the final colored image.

The authors highlight the flexibility of the approach as global features of one image can be combined with local features of another, which may result in a new variant of the image which will still be plausible. Other than colorization, as the model is extracting the features, it can also be used for classification.

According to the authors, the model gives the best performance when the image size is 224×224 , though it gives good results for varied sizes of input images. Because of these observations, if the input images are of different resolutions, rescaled images of the resolution must be used of 224×224 size.

They trained the model using MSE (mean squared error). Because of the sigmoid function, the output is normalized in the range $[0, 1]$. To calculate the MSE loss, the authors convert the output to the size of the colorization network output. This loss is back-propagated through the network.

They trained their model on the Places scene dataset [12], which consists of different types of images of scenes and locations from different places. Their model was evaluated in a user study and found to be natural 92.6% of the time.

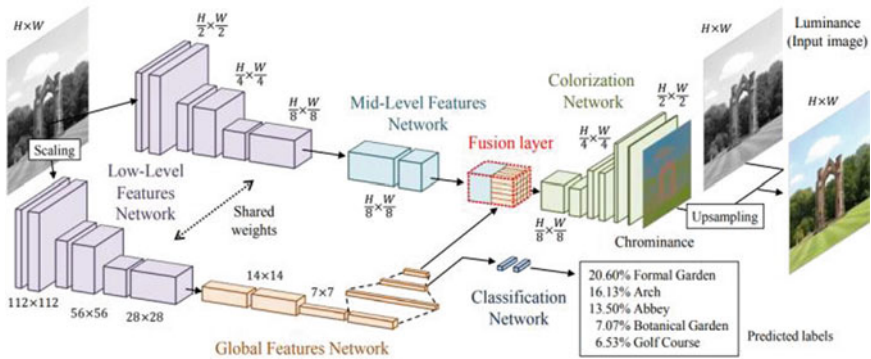


Fig. 6 Overview of automatic colorization model by Iizuka et al. [10]

Image-to-Image Translation with Conditional Adversarial Networks

Isola et al. [14] introduced conditional adversarial networks as a general purpose solution for image-to-image translation problems. They have given a common framework for all the different problems which include predicting pixels from pixels.

For image prediction problems, CNNs are generally considered as an approach. The authors find that CNNs learn to minimize a loss function like Euclidian distance between predicted and ground truth images, in a naïve approach, it will tend to produce blurry images. Creating a specific loss function and training a CNN model on it is a difficult problem. They suggest that we should make the model automatically learn the loss function which will produce appropriate outputs. This can be achieved by using GANs. GANs learn a loss that tries to classify if the output image is real or fake [14] and simultaneously train a generative model to minimize this loss.

In some earlier similar approaches, GANs are used with the unconditional setting [4]. In this paper, the authors suggest the use of conditional GANs (c-GANs) with a condition on input images and generate an output image.

The architecture proposed uses a “U-Net”-based generator and a convolutional “PatchGAN” classifier as a discriminator. We can see the proposed flowchart in Fig. 7.

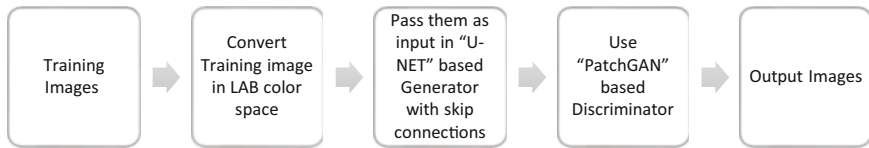


Fig. 7 Flowchart of the algorithm proposed by Isola et al.

The objective function of conditional GAN can be given as follows [14]:

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y} [\log D(x, y)] + [\mathbb{E}_{x,z} \log (1 - D(x, G(x, z)))] , \quad (1)$$

where G tries to minimize the objective and D tries to maximize it [13]. The authors explore that using this objective function with L1 distance rather than L2 will be beneficial as it produces less blurring.

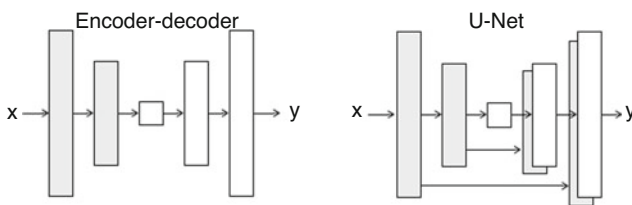


Fig. 8 Encoder-decoder structure without skip connections and U-Net structure with skip connections [14]

The network architectures of the generator and discriminator use modules of the form convolution-BatchNorm-ReLu. The generator architecture is designed around

the consideration that, both the input and output are both high-resolution but only differ in surface appearance, but have the same underlying structure. In the encoder-decoder structure, all the information is passed through all the layers. The low-level information is also transferred all the way from input to output, which can be directly passed to the output layer instead of going through all layers. To serve this purpose, the authors added a skip connection following the shape of “U-Net.” Similarly, all the skip connections are given between layer i and layer $n-i$ [14], as shown in Fig. 8.

The low-frequency correctness will be easy to handle because of the L1 loss. To model high-frequency, the authors find it sufficient to restrict to structure of image patches. The PatchGAN is designed to penalize the structure at the scale of patches. The discriminator classifies the $N \times N$ patches of the produced image as real or fake over the whole image. For evaluation of the generated images, they have used two different methods, first, run Amazon Mechanical Turk (AMT), and second, use object detection technique on the output.

To perform the analysis of objective function, the authors compared the results of the same images generated from normal cGAN, GAN, cGAN with L1, and GAN with L1 and found out that cGAN with L1 performs well and produces images closer to ground truth.



Fig. 9 cGAN produces grayish or desaturated results when not sure of a single plausible color [14]

Figure 9 shows some failed output showing some grayish or desaturated results. These are some of the drawbacks of using this methodology.

4 Methodology

4.1 Dataset and Implementation

The image datasets available are generally used for image segmentation, classification, etc. A standard dataset for colorization is not available. We need to convert the images first in grayscale then apply the algorithm and get the result. The datasets

generally available contain mixed images in them for example of vehicles, buildings, toys, etc. These objects can have any color and it would seem plausible. So, we cannot evaluate the actual performance of the models using this dataset.

Hence, we have used two open-source datasets one of which is a landscape dataset having natural landscapes of different types and seasons in it. Another one is of human faces dataset. Some images from both datasets are shown in Fig. 10. We have trained models on both datasets to compare the performance of all models. The landscape dataset is chosen because specific colors can only be considered plausible for natural images which reduce the chances of getting fake images. We have tried to use the models on the faces dataset as well to check if the model provides a similar result.



Fig. 10 The first four images are the sample of the faces dataset and the next four are from the landscape dataset

5 Results and Discussion

5.1 Metrics for Evaluation

Metrics used for evaluating colorization quality are standard metrics. These are generally used when a human would be determining the accuracy of the result. As all the models would be producing results having “plausible” results and not the same colors, it is not an easy task for humans to distinguish between real and fake and to see overall quality. Keeping these thoughts in mind, we will be using mathematical metrics like PSNR (peak signal-to-noise ratio) and SSIM (structural similarity).

Another reason to use specifically these two metrics is that metrics like MSE or PSNR give us the estimates about absolute errors but SSIM is a perception-based metric. It considers changes in the image as the perceived change in structural information and gives importance to perceptual information including luminance and contrast masking which is needed to be measured in colorization tasks.

Figure 11 shows some of the outputs of the landscape datasets for three different models and Fig. 12 shows the results of the same models applied on the faces dataset. The third and fourth column shows the output from the cGAN approach which we trained for two different number of epochs, i.e., 100 and 200. This produced significantly better outputs in some cases but did not improve much on failed cases. This conclusion is well supported by the metrics table given ahead. In the case of cGANs, the output is grayish or unsaturated due to the uncertainty of one single color. Some images did not improve because of too many details with different colors present in the original image. As seen visually, we cannot find robust outputs for the model suggested by [3]. We can see that CNN and cGAN approaches are producing plausible results which are not much indistinguishable from the original images. In those, cGAN has produced more robust outputs.

As suggested by the author that if there is very less difference in texture any color might be applied on the image patches, it can be seen in Fig. 13. The outputs of the algorithm given by Welsh et al. are shown in this image. Wherever the texture seems to match the source image texture, the color is transferred to the target patch. We tried to give exactly similar images as the source image and got the following results. Figure 13f is a failed case.

Table 1 depicts the comparison of performance of different approaches. The evaluation metrics used for performance evaluation are PSNR and SSIM [15–20]. From Table 1, it can be concluded that deep learning approaches suggested by [10, 14] produce better output when compared to others using metrics. Best outputs can be seen for the faces that the datasets produced by cGAN approach.

Summary

We have implemented the above-mentioned approaches and got the results. Table 2 is a summary of all these approaches. The advantages and limitations of the methods are mentioned by the authors in their study and we have ascertained them with our results as well.

6 Conclusion

Image colorization is an important research problem with significant real-life applications. Though important, it is a tedious task. Many different approaches have been suggested by different authors and we have presented four novel methods to colorize the image. We have come from semi-automatic to fully automatic methods



Fig. 11 Some outputs of the landscape dataset of models suggested by [3, 10, 14] compared with the original image (rightmost). Some of the outputs are failed cases as shown in the last two rows of outputs

which provide more “plausible” outputs. Deep learning approaches provide better output as compared to statistical models. We produced outputs for the given models and compared the results based on visual inspection as well as by using evaluation metrics. Based on our outputs and authors’ observations, we have compared all the models and provided a summary.

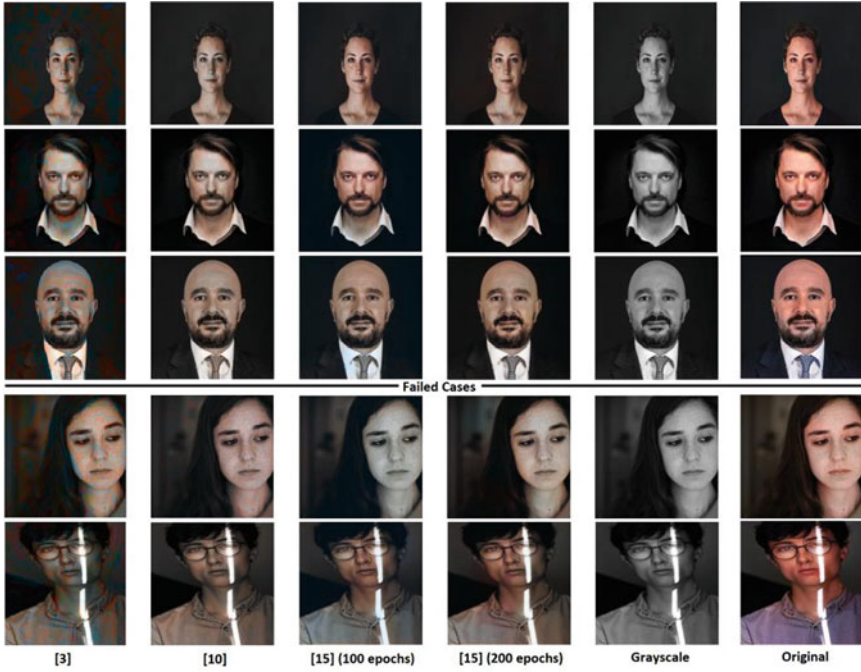


Fig. 12 Some outputs of faces dataset of models suggested by [3, 10, 14] compared with the original image (rightmost). Some of the outputs are failed cases as shown in the last two rows of outputs

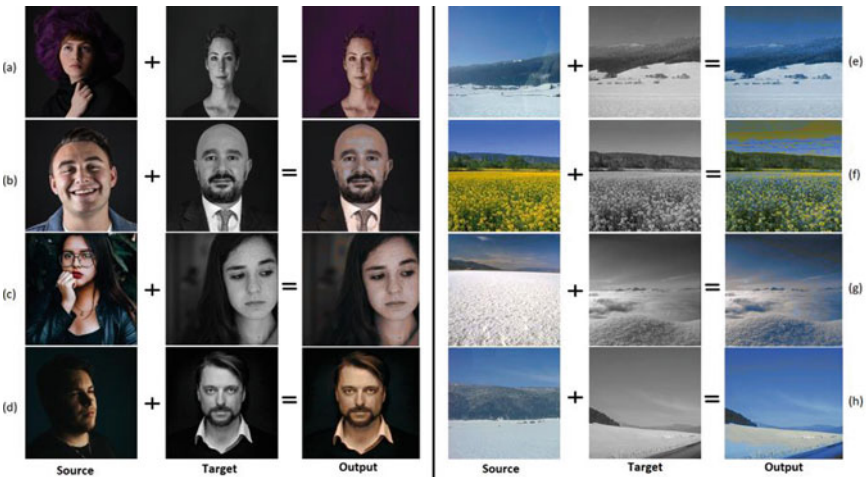


Fig. 13 In above figure, (a)–(d) show output for Faces dataset and (e)–(h) show output for landscape dataset of the algorithm suggested by Welsh et al. [1] for both the datasets

Table 1 Evaluation metrics values for each model for both datasets. (Higher the value for both the better)

Approach	Dataset	No. of epochs	PSNR	SSIM
Transferring color to grayscale images [1]	Both	–	27.8028	0.6656
Automatic colorization	Landscape	–	27.8864	0.5634
Of grayscale images [3]	Faces	–	27.9807	0.6347
Joint end-to-end learning of global and local	Landscape	30	29.1403	0.7961
Image priors for automatic image colorization [10]	Faces	30	30.4893	0.8705
Image-to-image translation with conditional adversarial networks [14]	Landscape	100200	28.827629.0681	0.75520.8098
	Faces	100200	30.952130.9829	0.82590.9315

Table 2 Summary of all the approaches, advantages, and limitations of using them, based on the authors' conclusions and our outputs

Title	Method	Advantages	Limitations
Transferring color to grayscale images [1]	Convert in lab color space and match texture for color transfer from colorful image to a grayscale image. While training converts images in color space using quantized k-means. Extract features using SURF algorithm, reduce the dimension using PCA, and train with SVM.	Fast and simple approach. When given a correct source image produces good "plausible" output. Provide great output on many categories of images like animals. The model gives a plausible result for many images and works exceptionally well on images it was trained on.	Need human intervention to give perfect matching source image for every target image or produce wrong output.
Automatic colorization of grayscale images [3]	Proposed an encoder-decoder structure using CNN. Compute low-level features and using them compute mid and high-level features. Pass this to the colorization network using the fusion layer and upsamples it.	The model can understand the colors and adapt them to the context of it. The model can also do style transfer meaning it can color the image using the context of another.	Does not differentiate between the pixels which may look the same locally but are from different areas globally for example in the landscape scenes in Fig. 3c.
Let there be color: joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification [10]	Conditional GANs are used in place of the encoder-decoder structure. U-Net with skip connections is used as a generator and convolutional PatchGAN classifier as a discriminator. The L1 loss function is proposed over the conventional L2.	Better output than the conventional encoder-decoder model with L2 loss. Produces images that are closer to the ground truth.	It is a data-driven method and thus will be able to colorize only those images that are having some common properties with the training set.
Image-to-image translation with conditional adversarial networks [14]			L1 loss produces an average grayish color when it is uncertain of any plausible color. In Fig. 8 failed case, we can see that at 100 epochs the image is still grayscale as it is uncertain of a single color.

References

1. T. Welsh, M. Ashikhmin, and K. Mueller, “Transferring color to greyscale images,” in 29th annual conference on Computer graphics and interactive techniques, 2002, pp. 277–280.
2. Reinhard, E. Ashikhmin, M., Gooch B. and Shirley P., “Color Transfer between Images”, IEEE Computer Graphics and Applications, September/October 2001, pp. 34–40
3. Austin Sousa, Rasoul Kabirzadeh, and Patrick Blaes, “Automatic Colorization of Grayscale Images”, CS229 <http://cs229.stanford.edu/proj2013/KabirzadehSousaBlaes-AutomaticColorizationOfGrayscaleImages.pdf>
4. Alex Meistrenko, “Automatic Recolorization of Movie Posters (Generative Modeling)”, CS231, <http://cs231n.stanford.edu/reports/2017/pdfs/302.pdf>
5. Ahmad S. Alhadidi, “An Approach for Automatic Colorization of Grayscale Images”, International Journal of Science and Research (IJSR) ISSN (Online): 2319-7064
6. https://www.researchgate.net/figure/The-CIELAB-color-space-diagram-The-CIELAB-or-CIE-L-a-b-color-system-represents_fig1_338303610
7. Qiwen Fu, Wei-Ting Hsu, Mu-Heng Yang, “Colorization Using ConvNet and GAN”, CS231, <http://cs231n.stanford.edu/reports/2017/pdfs/302.pdf>
8. F. Baldassarre, Diego González Morín, Lucas Rodés-Guirao, “Deep Koalarization: Image Colorization using CNNs and Inception-ResNet-v2”, ArXiv:1712.0340.
9. R. Zhang, P. Isola, and A. A. Efros, “Colorful image colorization,” in European conference on computer vision. Springer, 2016, pp. 649–666.
10. Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa, “Let there be color! joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification.”, ACM Transactions on Graphics, 35(4), July 2016.
11. Larsson, G., Maire, M., Shakhnarovich, G., “Learning representations for automatic colorization.”, European Conference on Computer Vision, Springer, 2016, pp.577–593.
12. Zhou, b., lapedriza, a., xiao, j., torralba, a., and Oliva, A., “Learning deep features for scene recognition using places database”. In NIPS, 2014
13. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Ward-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in Advances in neural information processing systems, 2014, pp. 2672–2680.
14. P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in IEEE conference on computer vision and pattern recognition, 2017, pp. 1125–113
15. Saeed Anwar, Muhammad Tahir, Chongyi Li, Ajmal Mian, Fahad Shahbaz Khan, Abdul Wahab Muzaffar, “Image Colorization: A Survey and Dataset”, arXiv:2008.10774v2 [cs.CV] 3 Nov 2020
16. A. Horé and D. Ziou, “Image Quality Metrics: PSNR vs. SSIM,” 2010 20th International Conference on Pattern Recognition, 2010, pp. 2366–2369, <https://doi.org/10.1109/ICPR.2010.579>.
17. Setiadi, D.I.M. PSNR vs SSIM: imperceptibility quality assessment for image steganography. *Multimed Tools Appl* 80, 8423–8444 (2021). <https://doi.org/10.1007/s11042-020-10035-z>
18. T. Zhao, K. Zeng, A. Rehman and Z. Wang, “On the use of SSIM in HEVC,” 2013 Asilomar Conference on Signals, Systems and Computers, 2013, pp. 1107–1111, <https://doi.org/10.1109/ACSSC.2013.6810465>.
19. M. Ü. Öziç and S. Özşen, “A new model to determine asymmetry coefficients on MR images using PSNR and SSIM,” 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), 2017, pp. 1–6, <https://doi.org/10.1109/IDAP.2017.8090201>.
20. P. Ye, J. Kumar, L. Kang and D. Doermann, “Unsupervised feature learning framework for no-reference image quality assessment,” 2012 IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 1098–1105, <https://doi.org/10.1109/CVPR.2012.6247789>.

Evolutionary Approaches Toward Traditional to Deep Learning-Based Chatbot



Arpan Maity, Srijita Guha Roy, and Debajyoty Banik

Keywords Natural language processing · Machine learning · Feed forward neural network · Chatbot · SVM · Confusion matrix · ANN · Decision tree · Tokenization · Stemming · Random forest classifier

1 Introduction

Chatbots help companies to address customer problems or queries and enhance the overall customer experience. Chatbots, also known as virtual conversational assistants, are the software program that allows the user to have real-time conversation with the software. These conversations can be in the form of texts or audio. Chatbots convert human language into digital data with the help of several machine learning and NLP techniques. Some of the renowned chatbots include Siri, Alexa, and Cortana which are developed by Apple, Amazon, and Microsoft, respectively. Artificial intelligence-based chatbots have the ability to understand and learn human language and not just follow commands like traditional chatbots. Three classification methods are commonly used by AI-based chatbots, which include pattern matches, natural language understanding (NLU), and natural language processing. Chatbots¹ can be divided into types: retrieval-based models and generative models.

In this paper, we have presented a retrieval-based chatbot model, which takes a user query and compares it with a set of predefined queries present in the language set of the model. Then, it selects one such tag of queries with the highest similarity score and returns a response which is paired with the chosen query. One response is

¹<https://www.chatbot.com/>

A. Maity (✉) · S. G. Roy · D. Banik
Kalinga Institute of Industrial Technology, Bhubaneswar, India
e-mail: 1905849@kiit.ac.in; 1905850@kiit.ac.in

chosen from the set of responses with the help of a random module function. These types of models work on the principle of directed flows or graphs. As a retrieval-based model simply picks a response from a given repository of query-response pairs, there are no chances of any grammatical error. Hence, the larger the language set that is fed into the model, the higher is the accuracy rate of the developed model. To implement our chatbot model, we used several machine learning algorithms and then carried out a comparative analysis among them. It was found that the feed forward neural network with two hidden layers provides the best accuracy score, and hence, this algorithm was used to develop our final model.

Further our paper has been divided as follows: Sect. 2 presents the basic concepts, Sect. 3 presents the motivation of our paper, Sect. 4 contains the literature review of previous related works, Sects. 5 and 6 contain the proposed work and algorithm used, respectively, Sect. 7 presents the UI design, Sect. 8 covers the result analysis, and Sect. 9 concludes the paper.

2 Key Concepts

Machine Learning In simple terms, Machine Learning is the ability of programs to adapt to new methods of solutions without following a set of rules. Depending on the type of labels available (continuous or discrete), ML model is of two types, regressor and classifier. Since our labels will be of discrete type, so we are using some of the popular classifier model for the comparison, like KNN, decision trees, random forest, SVM, and logistic regression.

Feed Forward Neural Network A feed forward neural network² is an artificial neural network where the data is only processed in one direction. Though the data may pass through multiple hidden layers between the input and output layer, it does not move backward, and hence, it never forms a cycle.

Natural Language Processing Natural language processing (NLP) deals with the interaction of computers and human language. NLP attempts to build such machines that understand and respond to natural human language such as text or voice messages and can respond with text and audio messages of their own. This technology can extract every nuance of the information present in a document and can even organize and categorize them accordingly.

Tokenization Tokenization is the one of the first steps for preprocessing the language set for any NLP model. It basically splits a group of text into smaller pieces, known as tokens. For example, a paragraph can be tokenized into sentences,

²<https://towardsdatascience.com/deep-learning-feedforward-neural-network-26a6705dbdc7>

and a sentence can be tokenized into words, and so on. After a chunk of text is tokenized, then the further processing is done on that tokenized text.

Normalization When a chunk of text is used for a NLP model, it needs to be normalized at first before any further preprocessing. The term normalization refers to performing some operations for bringing the entire text on the same level. Normalization helps in removing the extra spaces and the punctuation marks, converting the entire text into the same case (either uppercase or lowercase), converting digits or numbers to their equivalent words, etc. This helps in performing the further processing of the language set uniformly.

Stemming Stemming is a key concept that is used for NLP. In stemming, all the affixes from a word are removed to form the stem. The affixes include prefixes, suffixes, infixes, and circumfixes. For example, the word “waiting” is stemmed to “wait.”

Lemmatization Lemmatization³ is similar to stemming; the only difference between the two is stemming removes the affixes of a word, whereas lemmatization identifies the canonical form of the word. For example, if we try to stem the word “better,” it will not return any result, whereas using lemmatization, “better” can be changed into “good.” Lemmatization may be very efficient, but its implementation is much more complicated than that of stemming.

Bag of Words The bag of words is a particular format of representation of our language set which helps to simplify its contents. When each word is separated after tokenization, then the duplicate words are removed and that group of words are stored in a list. The bag of words does not consider the grammar or the order of words, but it mainly focuses on the number of occurrences of a particular word. Hence, the resulting bag of words consists of a set of words and their corresponding occurrences.

3 Motivation

“Predicting future is not magic, it’s Artificial Intelligence.” The work of a computer is to reduce a man’s effort in doing a work. The advent of AI reduced that effort further in an intelligent way. To reduce the human efforts, it is necessary to establish a very smooth and convenient communication between the human and the computer, as unlike humans, computers only understand in the form of 0 and 1. This led to the advancement of natural language processing, for better human-computer conversation. NLP reduced the need of human intervention in various areas likes websites for query processing, notifications, and many more. Today, anyone can get their queries answered, from related websites by the help of chatbots. But the

³<https://www.techslang.com/definition/what-is-lemmatization/>

task of NLP is just to convert simple human language to a form that machines can understand, but to apprehend the exact information that the user wants to access, ML and neural networks play a vital role there [11]. In other words, the ML/neural network models are the backbone of a NLP engine, and the better the ML models are, the better and accurate will be the query processing by NLP engine. So we worked on various popular ML and neural network models that are available, compared their results, and suggested a feed forward neural network algorithm, which is very simple to implement and would give accurate results.

4 Related Works

Mohammad Nuruzzaman et al. [1] presented a survey on the different techniques for chatbot implementation in the customer service industry with the help of deep neural networks. Their paper discusses the similarities, differences, and limitations of some existing chatbots. From their survey, it was inferred that 75% of the users have faced bad customer service, and the generation of long, informative, and relevant texts has always been a challenge for chatbots. Deep neural network is an important productive-based model to solve the response generation problem of chatbots. This survey mainly deals with 11 most famous chatbot applications and their functionalities and examines over 70 chatbot-related publications in the past 5 years. Supratip Ghose et al. [2] proposed the implementation of a dialogue-based natural language chatbot which is topic specific. This paper deals with the insights of a FAQ chatbot for college students. The chatbot takes natural language as its input, searches through its information set, and outputs the information of a student in natural language. They plot the data in the information repository in a connected graph, whose nodes contain the information while the links interrelate the nodes of information. For experimental purposes, they created three systems, a pure dialog-based system, a domain knowledge-based system, and another system hybrid of these two. The outcome of this experiment shows that domain-specific dialogue combined with conversational knowledge can produce maximum session of dialogue as compared to the general conversational dialogue.

Rafal Poświata et al. [3] created an annobot. Annobot is a chatbot that can annotate and create datasets based on human conversation. A more approachable and flexible interface has been developed through this natural conversation, for mobile devices especially. There are a huge range of applications of the annobot, such as classification of labels and labeling of data for binary, the data is also prepared for problems related to regression, and sets are created for machine translation, text synopsis, or answering questions. Some more interesting features of this chatbot are pre-annotation, learning online, active sampling, and real-time inter-annotator agreement. All these features make the annobot more useful and efficient compared to other chatbots.

Christopher D. Manning et al. [4] proposed a survey on understanding and predicting user dissatisfaction of a neural generative chatbot. In their paper, a

neural generative chatbot used as part of Chirpy Cardinal, an Alexa price socialbot, is used for a detailed study. It is found that unclear user expressions such as hallucinating, repetitions, ignorance, and unclearness are a major cause for most of the errors. However, sometimes the model makes reasoning errors even for precise information. Though the dissatisfaction expressed by the users depends on other factors as well such as the user's personal attitude, previously unaddressed dissatisfactions, etc. Finally, it was shown that to improve the dialogue system, a semi-supervised model was trained with dissatisfied user expressions to predict next-turn dissatisfaction.

Alaa Joukhadar et al. [5] proposed a model for Arabic dialogue recognition using a textual chatbot. They aim to recognize the dialogue entered by the user using Levantine Arabic dialect. They have used eight types of dialogue acts such as greetings, thanks, negate, and apology. They have used different machine learning techniques such as logistic regression, random forest classifier, extra tree classifier, and multi-nominal NB and SVM to detect the accurate dialogue act categories. Along with this, the voting ensemble method has also been used to derive the finest prediction of each model. Then, finally the results of different models were compared, and it was found that the SVM model has provided the best results with 2 g.

Jintae Kim et al. [6] proposed a two-step training and mixed encoding-decoding model for implementing a small dialogue corpus generative chatbot. When a huge amount of data is used for training a model, a generative chatbot can produce natural conversation interactions. But it becomes tough to collect large dialogue sets for training data except for a few languages such as Chinese and English. So, to overcome this problem, they used combinations of syllables and words. They used a two-step training which includes a pre-training using a large dialogue set and then a re-training using a small dialogue set. This method helped to deal with the out-of- vocabulary problem, and along with that, it also reduced the grammatical and semantic errors. Thus, the two-step training method made the chatbot much more error-proof, in comparison to training the chatbot with a single small dialogue corpus.

5 Proposed Work

In our work, we implemented a fully working chatbot, by using several popular ML techniques like KNN, decision tree classifier, SVM, random forest, and many more, and then compared the resultant [7] output with themselves and with a feed forward neural network with two hidden layers and a simple ANN model. The data was processed under tokenization and stemming algorithms, before converting them into bag of words. To implement the whole concept, we needed an example to work upon. So we used an imaginary restaurant, for whom the chatbot would work for the customer satisfaction. We did a survey in our friend circle, for the probable questions

that could be asked to a restaurant chatbot and included each and everyone of them. After that the dataset is passed through all the important processing engines, before feeding them into the ML models and the neural network model. The responses that the chatbot would give are also randomized, so that the interaction of the bot with human does not become monotonous. The details of each and every processes are given in Sect. 5.

5.1 Novelty

As we already discussed previously, a chatbot requires the combined implementation of NLP and ML/neural network models. So for the proper working of a chatbot, the proper implementation of underlying ML/neural network model is very important. Hence in this paper, we compared the performance of the chatbot, using some popular ML algorithms(stated above) as the underlying model. After that we suggested and worked with a feed forward neural network model (with two hidden layers), which gave much better results than the other popular ML models and then compared that model with an artificial neural network model, to bring a conclusive result.

5.2 Dataset Used

In this paper, we used two types of datasets. One made by us, for better fitting and better accuracy, and the latter was collected from an open-source website: <https://nextit-public.s3-us-west-2.amazonaws.com/rsics.html>.

We designed the dataset in json format, where everything was saved inside the viable “intents.” There is a tag for each kind of questions, and some example questions are given in “patterns,” and it’s expected answer is given in “response,” from where the output will be displayed in random fashion.

In case of the open-source dataset, we first converted it into json form manually and then fed it into our model, for optimum result

6 Evolutionary Algorithms

We have shown a glimpse of our algorithm in the proposed work in Sect. 4. The step-by-step procedure of the implementation of the algorithm is provided below:

- Our first aim was to develop a dataset, which will cover the maximum questions, and their varieties, so that our neural network model gets enough patterns to be feed with. We made a json file and named an entire set as intents.
- Then we chose several tags (or subjects) under which questions can be asked, like greeting, thank you, and goodbye.
- Inside those tags, we gave two sections, patterns and responses. Inside pattern, there is a set of questions, related to the tag, and in the responses, there are the answers of the question. In maximum cases, more than one answers are chosen, and while execution, the answers are displayed randomly, so that the human-bot interaction does not get boring.
- After that the whole dataset is passed through the NLP pipeline developed by us. The pipeline has five stages in total.
- In the first stage, the words are tokenized, that is each and every word is stored inside an array, as string. The punctuations are also treated as separate words and stored in the array.
- In the next stage, all the capital letters are converted to small letters, as entry of capital and small letter together could cause a mismatch in the pattern.
- After that, stemming is performed with each word. Stemming basically chooses the keyword, within the whole work, and eliminates the extra letters. By doing that recognition of patterns becomes much easier.
- In the next step, the punctuation marks are removed. Punctuation marks, though very much important in human conversion, or speech recognition, do not have a great role in pattern recognition.
- In the last stage of the pipeline, the tokenized, lowered and stemmed words are converted into bag of words. For making this, the array is compared with the total stemmed word dataset, and a list is made out of it, where 0 is put if the word is absent in the main set and 1 is added if it is present.
- Next we fed the bag of words to several ML models, i.e., KNN, decision trees, random forests, logistic regression, and SVM, so that we can compare the outputs with each of the ML techniques and also with some of the neural network techniques given below.
- After feeding the bag of words to the abovementioned neural network models, they passed to a neural network, for training. It is a feed forward neural network, with two hidden layers. The batch size is taken as 11 and the number of epochs is taken as 1500 for better accuracy. Here the input is the bag of words and the output is the specific tag.
- At last, the output is fed to a softmax, so that the output comes in the form of probability, so the result above a certain probability value can be chosen. In our algorithm, we chose the probability limit as 75%. Below that, an output of “I cannot understand” will appear.
- We had also passed it through a simple ANN method, so that it can be compared with the above mentioned feed forward neural network model.

7 UI Design

The user interface for our chatbot OreO has been developed using the Tkinter library of Python. Python provides many modules for developing a GUI (graphical user interface), out of which, Tkinter is the most widely used module. Tkinter is an open-source module released under a Python license. The name Tkinter is derived from Tk interface. It is available on most of the Unix platforms, including Windows and MacOS as well. Tkinter being the main module, there are several other modules which give Tk support, such as `tkinter.font`, `tkinter.colorchooser`, `tkinter.messagebox`, and `tkinter.ttk`.

In our code, a basic GUI has been created for our chatbot using Python. The user can type their query on the message entry box which is present at the bottom. The focus function has been used to always keep the message box in focus whenever the application is started, so that the user can directly start typing without clicking anywhere. For sending the message, the user can either hit the send button on the right of the message box or press the enter key on the keyboard. After a message is sent, it is deleted from the message entry box and displayed in the chat section, and a reply is received from our chatbot. Our chatbot also consists of a menu bar which includes three main menus – File, Options, and Help. Each of these menus consists of some submenus. The File menu is used to clear the chat and exit the application. The Options menu is used to change the font style and the color scheme of the application. And lastly, the Help menu displays information about OreO the Chatbot v1.0 and its developers.

8 Result Analysis

For the result analysis, we are taking the help of two important parameters:

- The execution time
- The overall accuracy

And after that, we also calculated the results of the softmax function, for which the feed forward neural network yielded the best overall result.

8.1 Execution Time

Unlike other machine learning algorithms, which involves [8] image processing and other heavy tasks inside their algorithms, in a chatbot algorithm, the model only deals with string, so the overall process becomes very fast. We looked at the execution time of the processes and noted it for each algorithm and found that the overall time was almost same for each of them. KNN, logistic regression, and

decision tree methods executed a little bit faster. Random forest and SVM methods were a bit slower than the previous, and the feed forward neural network method and the ANN one were the slowest. The reason for that was simple to understand, the simplicity of KNN, decision trees led them to give faster result than others, and the complexity of the neural network models made them slower

8.2 Accuracy

To measure the accuracy of each of the algorithms, we first selected 35 questions related to the chatbot we are implementing. Then we gave the questions as input to each and every method and noted the output tag they are predicting. On the basis of the output tags, the accuracy table is prepared in Table 1.

Table 1 Accuracy of various ML models

Algorithm name	Accuracy (in %)
KNN	71.4
Decision tree	81.0
Random forest	82.8
Logistic tegration	80.0
SVM	85.7
Artificial neural network	84.3
Feed forward neural network	94.4

Table 2 Softmax output for various statements

Statements	Probability
S1	0.999999166
S2	0.999998569
S3	0.999995449
S4	0.999999046
S5	0.532413244
S6	0.996549845
S7	0.999928496
S8	0.994595449
S9	0.992234254
S10	0.984595449

It can be understood very clearly that the feed forward neural network gave the best result among all [9], and among all the non-neural network methods, the SVM performed the best.

The main advantage that the feed forward neural network has over the others is the large number of parameters, like weight biases and a huge number of

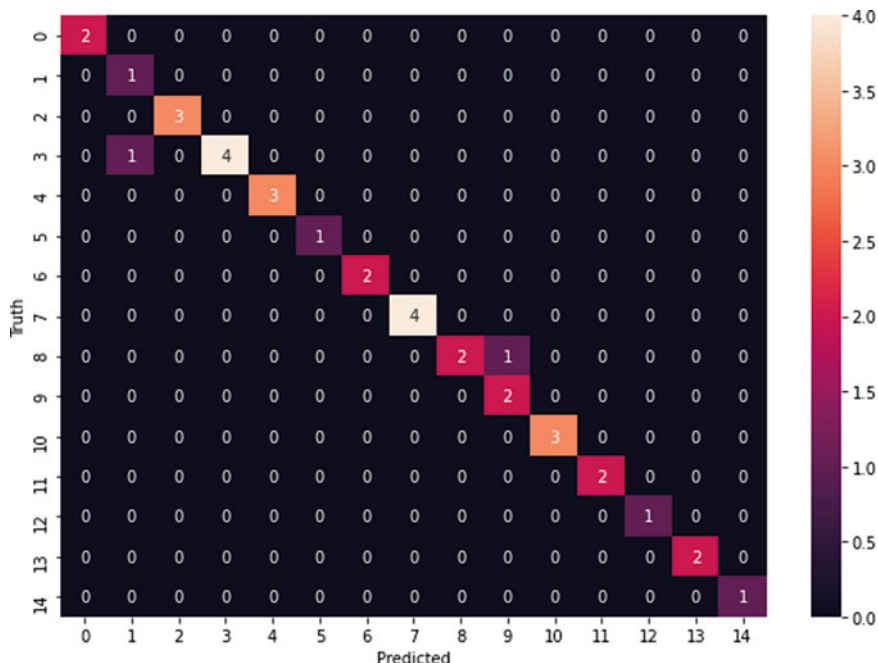


Fig. 1 Confusion matrix

interconnected nodes, so that they can fit highly complex datasets very easily and give very accurate results, and this result is very clearly visible also. In the paper “Arabic Dialogue Act Recognition for Textual Chatbot Systems” by Alaa Joukhadar [5] and his team, their accuracy maxed out at 91%, but our accuracy went up to 94% while using the feed forward neural network model. The accuracy is reflected in the table and the confusion matrix above in Fig. 1

8.3 Probability Achieved by Softmax Function

Previously, we had already seen that the feed forward neural [10] network is giving the best accuracy. We had also discussed the reasons behind that and the working of softmax function in Sect. 5. In Table 2, we have given the probability outputs of ten statements using softmax function. Here, we designed the model so that if the probability is greater than or equal to 0.75, then only it will give an answer according to the tag provided; otherwise it will say “I don’t understand.” Not only that the average accuracy of all the statements is 0.99+. So we can say that our feed forward neural network model with two hidden layers is performing very well.

9 Conclusion and Limitations

Modern-day's websites and applications are becoming more and more intelligent day by day by using several AI techniques, and a properly working chatbot is a must-have for an intelligent and automated website. The feed forward neural network model, used in our chatbot, is designed so that it will also perform just the same if we change or modify the dataset. So this model is upgradable, and this is the reason why we can change our chatbot from "a restaurant website helper" to any thing we want, by just changing the dataset. This modularity makes the model more suitable for real-world uses.

Talking about limitations, we can say that the major limitation that we observe here is the lack of more modern and up-to-date ML/NN techniques. This could make the whole model more accurate while consuming lesser time.

This will for sure be the main motivation behind our future work, for developing a smarter chatbot using more modern neural network technologies. GitHub link of our code is <https://github.com/arpan112233/Chatbot-Oreo>.

References

1. M. Nuruzzaman and O. K. Hussain, "A Survey on Chatbot Implementation in Customer Service Industry through Deep Neural Networks," 2018 IEEE 15th International Conference on e-Business Engineering (ICEBE), 2018, pp. 54–61, <https://doi.org/10.1109/ICEBE.2018.00019>.
2. S. Ghose and J. J. Barua, "Toward the implementation of a topic specific dialogue based natural language chatbot as an undergraduate advisor," 2013 International Conference on Informatics, Electronics and Vision (ICIEV), 2013, pp. 1–5, <https://doi.org/10.1109/ICIEV.2013.6572650>.
3. R. Poświata and M. Perelkiewicz, "Annobot: Platform for Annotating and Creating Datasets through Conversation with a Chatbot," 2020 27th International Conference on Computational Linguistics, Barcelona, Spain, December 12, 2020.
4. A. See and C. D. Manning, "Understanding and predicting user dissatisfaction in a neural generative chatbot", 2021 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue, July 29–31, 2021.
5. A. Joukhadar, H. Saghergy, L. Kweider and N. Ghneim, "Arabic Dialogue Act Recognition for Textual Chatbot Systems" 2019 The Second International Workshop on NLP Solutions for Under Resourced Languages, 2019.
6. J. Kim, Hyeon-Gu Lee, H. Kim, Y. Lee and Young-Gil Kim, "Two-Step Training and Mixed Encoding-Decoding for Implementing a Generative Chatbot with a Small Dialogue Corpus.," 2018 Workshop on Intelligent Interactive Systems and Language Generation (2ISNLG), Tilburg, The Netherlands, November 5 2018.
7. V. Hristidis, "Chatbot Technologies and Challenges," 2018 First International Conference on Artificial Intelligence for Industries (AI4I), 2018, pp. 126–126, <https://doi.org/10.1109/AI4I.2018.8665692>.
8. B. Setiaji and F. W. Wibowo, "Chatbot Using a Knowledge in Database: Human-to-Machine Conversation Modeling," 2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS), 2016, pp. 72–77, <https://doi.org/10.1109/ISMS.2016.53>.

9. M. Baez, F. Daniel, F. Casati and B. Benatallah, “Chatbot Integration in Few Pat- terns,” in IEEE Internet Computing, vol. 25, no. 3, pp. 52–59, 1 May–June 2021, <https://doi.org/10.1109/MIC.2020.3024605>.
10. M. Jovanović, M. Baez and F. Casati, “Chatbots as Conversational Healthcare Ser- vices,” in IEEE Internet Computing, vol. 25, no. 3, pp. 44–51, 1 May–June 2021, <https://doi.org/10.1109/MIC.2020.3037151>.
11. K. Chakraborty, S. Bhattacharyya and R. Bag, “A Survey of Sentiment Analysis from Social Media Data,” 2020 IEEE Transactions on Computational Social Systems, 2020, pp. 7–11.

Analysis of Machine Learning Algorithms for Detection of Cyberbullying on Social Networks



K. V. Soujanya and Omprakash Tembhurne

Keywords Cyberbullying · Machine learning · Analysis of algorithms

1 Introduction

The internet has become an essential component of everyone's life. The day of a person begins while using the internet. This also gave a boost to the growth of social media platforms such as Facebook, Twitter and Instagram. On the other side of the coin, these platforms have begun to increase the problem of cyberbullying. Social networks have spread widely, people find illegal and unethical ways to use these communities [1]. Cyberbullying on social media (OSNS) has become a major issue among teens. The main differences between bullying and cyberbullying are the fact that the internet can help cyberbullies to hide their identity [2].

The upward thrust of social networks inside the virtual area has brought about a brand new definition of friendships, relationships and social communications. One can also additionally have loads of buddies without even seeing their faces. Cyberbullying could have deeper and longer-lasting results in comparison to bodily bullying. Bullying can cause severe depression, low shallowness and there were instances of suicide among teenagers. Cyberbullying is well-studied trouble from a social perspective [3].

Twitter is a popular public real-time microblogging network where news is often displayed before appearing in official sources. Twitter usage skyrocketed thanks to the short message limit (currently 280 characters) and unfiltered feeds, especially

K. V. Soujanya
Department of Computer Science and Engineering, GVPCE(A), Visakapathanam, Andhra Pradesh, India

O. Tembhurne (✉)
Department of Computer Science and Engineering, School of Engineering, MIT ADT University, Pune, India
e-mail: omprakash.tembhurne@mituniversity.edu.in

during the event, with an average of 500 million tweets posted daily [5]. Twitter is one of the most recognizable sources of ongoing news and is also the dominant news media. It caused great damage by spreading gossip in advance [11].

In the proposed system, the tweets taken as input are pre-processed after removing the stop words and converting them into vectors. The machine learning algorithms use these vectors to classify whether a tweet is offensive or not offensive. Their results are compared based on accuracy, precision, recall and F1-Score. The algorithm with the highest values can be considered for model building.

2 Background and Related Work

Cyberbullying in online social networks (OSNS) has become a serious problem among adolescents. While a lot of research has been done with a focus on designing very accurate classifiers for automatic detection of cyberthreats in OSNS [2]. Cyberbullying, bullying through the use of information and communication technologies, is a global phenomenon. Extensive research into bullying helps educators to understand its impact. However, the gap between technological progress and the lack of studies on cyberbullying suggests that more research is needed to understand the extent of the bullying [4].

K.Reynolds, A. Kontostathis and L. Edwards proposed a model using a decision tree that records the curse and insult words within a post. This model gives 78.5% accurate results in detecting cyberbullying [1].

Franciska de Jong and Roeland Ordelman took 3,81,000 posts from My Space and applied SVM to differentiate the gender of the authors. In this research, women wrote 34% of messages while men wrote 64% of messages [3].

Tolga aricak took a survey of 269 students from sixth to tenth grade. In his survey, he came to know that 36.1% of students came across and were exposed to cyberbullying on the internet. Next, 23.7% of students came across cyberbullying through their cell phones. As a result, 35.7% of 269 students exhibited bullying behaviour and 23.8% of 269 students exhibited bully-victim behaviours. Only 5.9% of the 269 students were victims. More boys exhibited bullying victimization and behaviours than girls. In the face of cyberbullying, 25% of students said they reported incidents of cyberbullying to their peers and parents; 30.6% of 269 students said they found active solutions, such as blocking the bully. From this survey, the author concluded that using technology for communication has increased the cause of cyberbullying [4].

Monika Sharma proposed a model using random forest that helps in detecting cyberbullying and automatically forwarding the required information to the concerned authorities. She used Bayes Net, Logistic Regression, Random Forest, J48, Naïve Bayes and Multilayer Perceptron for comparison, and used F1-Score as a comparison metric [6].

According to Umadevi K S, SVM accuracy is mainly reliant on the sample size because of the very smooth improvement in accuracy as the sample size grows.

Count and TF-IDF Vectorizers create virtually identical trend lines, with TF-IDF providing a steadily increasing level of accuracy for every inch of sample size. When it comes to SVMs, TF-IDF seems to be the logical option. The fact that accuracy improves with sample size suggests that SVMs, when trained on bigger datasets, may yield better results than other models [8].

Irfan Kareem used news articles from print media and implemented TF-IDF for feature extraction together with seven machine learning algorithms. He compared the accuracy of these algorithms and concluded that KNN gives 70% accuracy and is the highest among these algorithms [10].

Ehesas Mia Mahir used social media content like tweet id, the source of tweet, and the contents of the tweet for the prediction of fake news using SVM, Naïve Bayes, RNN and LSTM algorithms. He got SVM as the best algorithm with better precision value and Naïve Bayes as the best algorithm with better accuracy. Based on the results, it can be seen that deep learning algorithms gave less accuracy when compared to Machine learning algorithms [11].

Brenda Irena took 50,610 tweet data and applied the decision tree C4.5 method and got an accuracy of 72.91% by splitting the data as 90% train data and 10% test data [12].

Dinakar K, R.Reichart and H.Lieberman used 4500 YouTube comments and applied a wide range of binary and multi-class classifiers. Their outcomes show that the binary individual classifier outperformed the multiclass classifier for the detection of textual cyberbullying to determine if it is in a range sensitive to topics like physical attributes, race and sexuality. The outcome was 66.7% accuracy in the detection of cyberbullying [13].

Rahul Ramesh Dalvi used the posts and tweets presented on the Twitter platform and implemented software that detects and prevents cyberbullying on Twitter. This model used two algorithms SVM and Naïve Bayes for training and testing the bullying content and attained an accuracy of 71.25% and 52.70% [14].

S Leitch and M Warren found several security issues associated with Facebook and stated that the personal information of a user can be misused by any person over the internet [15].

Yin D et al. took the content of online posts to determine whether or not the post was harassing using TF-IDF vectors applied by a support vector machine and formed groups containing cyberbullying using a rule-based algorithm [16].

Chisholm J F took the contents of online posts and formed clusters using TF-IDF vectors that contain cyberbullying using the rule-based algorithm [17].

Jaana Juvonen made an investigation among girls and adolescent females who are both victims and bullies. He described the risk of online activities among girls and increased awareness regarding cyber safety [18].

Trisna Febriana and Arif Budiarto present the process of developing a dataset that can be used to build a hate speech detection model. Using the Twitter API, more than one million tweets have been successfully gathered. Machine learning was used to do basic pre-processing and early research. The subject for each tweet was extracted using the latent Dirichlet allocation (LDA) method to examine if these topics were associated with debate themes. To calculate a polarity score for each tweet, pre-trained sentiment analysis was applied to the dataset. The number of good

and negative tweets among the 83,752 tweets included in the analysis phase is about equal [28].

Aamir Amin, Farhan Bashir Shaikhand and Mobashar Rehman investigate the link between the identified factors towards cyberbullying. The identification of factors and their relationships will provide a complete knowledge of the phenomena of cyberbullying. The researchers discovered 34 cyberbullying variables among university students. “Personal variables”, “Socio-cognitive factors”, “Psychological factors” and “Environmental factors” are the four primary groups of these elements. The use of concept maps to classify positive and negative aspects gives a clear image of cyberbullying factors and their link to cyberbullying perpetration. Researchers and policymakers can use this categorization to better understand the phenomenon of cyberbullying [29].

I-Hsien Ting and Wun Sheng Liou have presented a method based on SNM approaches. They gathered information from four of Taiwan’s most popular social networking sites. They gathered 100 postings for each website that were determined to be cyberbullying incidents, as well as 100 posts that were not regarded to be cyberbullying incidents. The information acquired was then analysed by extracting keywords, SNA measures and sentiments. The accuracy is approximately 0.79, and the recall is around 0.71, according to the evaluation results. This suggests that with this method, more than 70% of cyberbullying posts may be appropriately identified [30].

Weider D. Yu, Maithili Gole and Sowmya Prakash have provided a means of restricting cyberbullied incidences in a controlled messaging system and said detection is a key step, and prevention of the incidents will yield far more powerful results [31].

2.1 Machine Learning Models Used for Analysis

Decision Tree

The decision tree is a supervised machine learning algorithm that can be used for both classification and regression but is mostly used for classification. This algorithm contains decision nodes and leaf nodes. The decision nodes represent the decision rules with multiple branches and leaf nodes represent the outcome that doesn’t have any further branches.

This model takes the whole dataset into the root node and divides it into subsets by extracting the key features using the attribute selection measure (ASM). These subsets again generate decision trees repetitively, until there is no further scope for classification. It stores the predicted classes in the leaf nodes.

This model takes the whole dataset into the root node and divides it into subsets by extracting the key vectors based on information gain values for regression. The vector with the highest information gain value is taken as the root node. All the other subsets are repeatedly divided and categorized based on the information gain

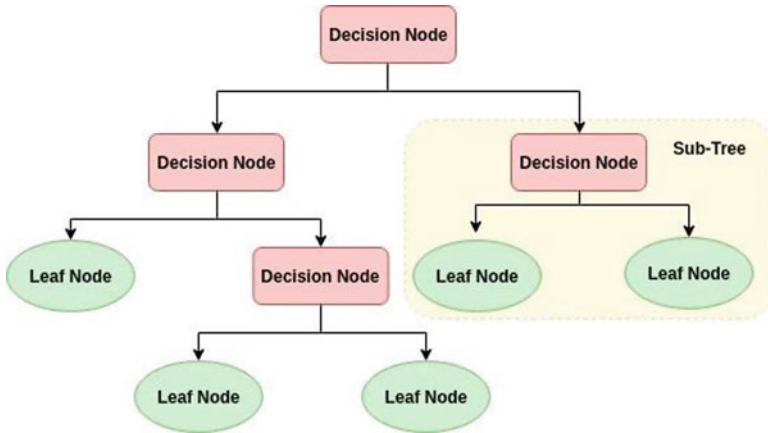


Fig. 1 Decision tree classifier

values. The end nodes store the output classes which are used for the classification [19] (Fig. 1).

Logistic Regression

Logistic regression is a machine learning algorithm that takes categorical values as input and converts them into probabilistic values which range between 0 and 1. It uses a sigmoid function for these conversions and forms an S-shaped curve for the data classification. This curve divides the data based on the probability value, such as above 0.5 the data are offensive while below 0.5 it is non-offensive. In some rare cases, the probability is 0.5, which states that the data is unclassified [20].

Natural language processing employs conditional random fields, a sequential data extension of logistic regression. You may use binomial, ordinal and multinomial logistic regressions. When the observed value of a dependent variable can only be one of two values: “0” or “1”, binomial or binary logistic regression is required. Multinomial logistic regression is the best method for analysing unordered data. Ordinal logistic regression works with ordered dependent variables.

The result is frequently recorded as “0” or “1” in binary logistic regression since this allows for the most obvious interpretation. To record a “1” for a successful outcome, the dependent variable is referred to as “instance”, “case”, and so on; to record a “0” for an unsuccessful outcome, the dependent variable is called “non-instance”. Probability of becoming a case based on the values of the independent variables is forecasted by using binary logistic regression (predictors). It is calculated by dividing the likelihood of a specific outcome being a case by the likelihood that it is not. Other regression analysis techniques use one or more predictor variables, which can be either continuous or categorical. Instead of using a continuous result, logistic regression uses one or a few categories to predict the outcome of a dependent variable (Fig. 2).

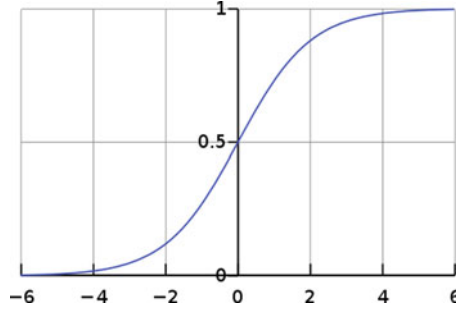


Fig. 2 Logistic regression

Random Forest

Random forest takes the complete dataset as input and divides it into various subsets randomly based on count vectors. It builds decision trees from these subsets. The prediction of each decision tree is considered and the class with the highest prediction is given for the test data [21].

In random forest algorithm, the classification process uses more than one “tree”. Each tree produces a classifier and these classifiers vote to decide the calculation that gets the most vote. This classification calculation is then used to characterize the huge amount of datasets. Hence considering the accuracy and leniency to use it with multiple datasets at a time [6] (Fig. 3).

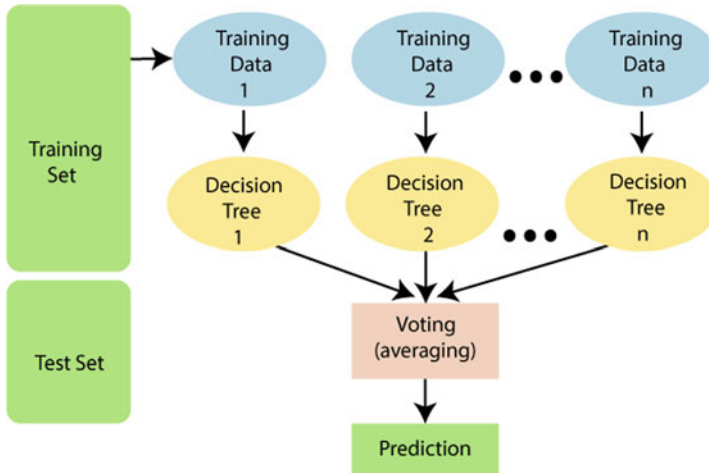


Fig. 3 Random forest classifier

Support Vector Machine

The parameter tweet is converted into vectors using the Count Vectorizer. These vectors are taken as input for training the support vector machine algorithm. This algorithm uses these vectors to generate hyperplanes iteratively until the best hyperplane is chosen. This hyperplane helps in classifying the dataset into two classes that are offensive or non-offensive. Though this algorithm takes a long time for training [22], it gives better accuracy when compared to other algorithms.

Support vector machines (SVMs), also known as support vector networks, are machine learning models that analyse data for classification and regression analysis to enhance accuracy in the field of machine learning. The kernel trick is a technique that SVMs employ to do non-linear and linear classification by implicitly turning their inputs into high-dimensional feature spaces. Because unlabelled data can't be utilized for supervised learning, an unsupervised learning approach is needed, in which the data is automatically sorted into categories and new data is mapped to these categories.

SVMs can be applied to a variety of real-world problems, but these are not limited to text and hypertext classification. SVMs have the potential to reduce the need for labelled training data in both inductive and transductive situations, making them useful in text and hypertext classification applications (Fig. 4).

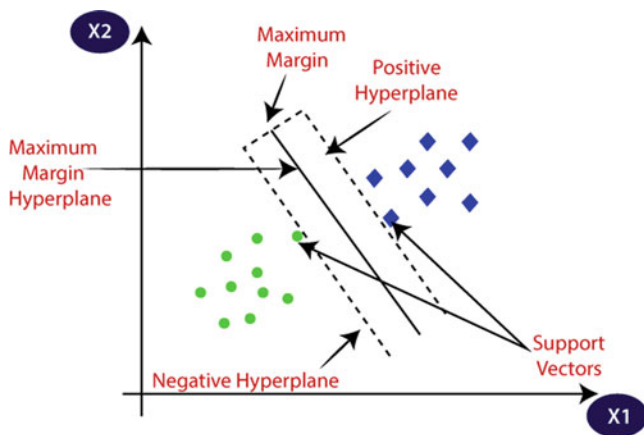


Fig. 4 Support vector machine

Bagging

It is an ensemble learning algorithm that uses the voting principle. This algorithm takes the dataset as input and picks a few random samples from it. These random samples are taken with replacement and bootstrapped each time the algorithm is

trained. Once the algorithm gets completely trained, it generates learners that are weak in nature. All of these learners come together to form a powerful learner who predicts outcome by counting the votes each learner has given them. The class with the highest votes is given as the output on the test data [23]. A combination classifier can be bagged because ensemble methods generate new versions of a classifier and classify by weighting the votes of each classifier [9].

The use of this technique may be subjected to overfitting if the dataset has not been pruned, but it may also be subjected to underfitting when the data set is exceedingly small, such as a decision stump that is only one level deep. We must remember that if an algorithm over or under fits to its training set, it will have difficulty generalizing to new datasets. This is why ensemble techniques are used to improve generalization performance to new datasets.

It uses a bootstrapping method to build a wide variety of samples. Selecting data points at random and replacing them is how this resampling approach creates various subsets of the training datasets. As a result, when you choose a training dataset data point, you may select that data point numerous times. Consequently, a value or instance appeared in the sample at least twice. Weak or basic learners are then employed to train these bootstrap samples, which are then trained in parallel with the rest of the bootstrap samples. To get a more accurate estimate, an average or a majority of the predictions are employed to generate a more accurate estimate. It is a regression strategy that takes the average of all the outputs predicted by each classifier and uses it to make decisions. When it comes to classification concerns, the class that receives the greatest number of votes is approved, which is known as hard voting or majority voting (Fig. 5).

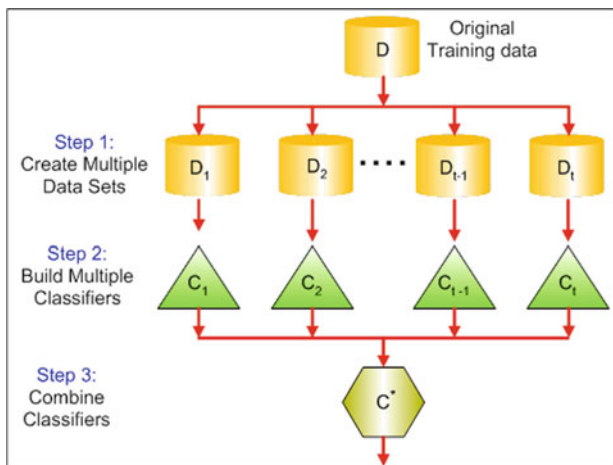


Fig. 5 Bagging classifier

Adaboost

Adaboost, also known as adaptive boosting, can be used to increase the performance of machine learning methods. This algorithm takes random samples from the dataset and builds learners by training the model. It then checks whether the predictions are accurate or not. Then it takes a few more random samples from the dataset along with the wrong predicted data from the previous model and updates learner values. This process continues until the whole dataset is completed.

All of these poor learners are then combined to form a powerful classification mechanism that helps improve the performance of the model. If any test data are given to this model, then it gives the class of the data as output [24] (Fig. 6).

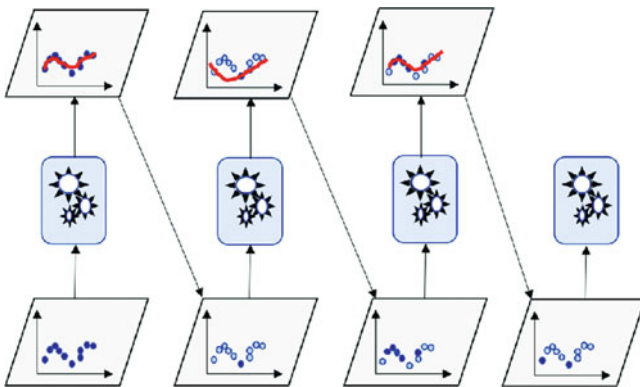


Fig. 6 Adaboost classifier technique

Naïve Bayes

Linear classifiers inclusive of Naïve Bayes are easy however extraordinarily efficient. The probabilistic version of Naïve Bayes classifiers is primarily based totally on the Bayes theorem and is referred to as naive because it assumes that the capabilities withinside the dataset are collectively independent. The capabilities are normally now no longer independent, however Naïve Bayes classifiers nevertheless have a tendency to carry out thoroughly below this unrealistic Ming. Naïve Bayes outperforms different effective classifier ranges while the pattern length is small [8].

This algorithm takes the dataset as input and uses the Bayes theorem to calculate the probabilities for each class label present in the dataset. The class label with the highest probability is given as output for the test data [25]. The formula for the Bayes theorem is as follows:

$$P\left(\frac{A}{B}\right) = \frac{P\left(\frac{B}{A}\right) P(A)}{P(B)} \tag{1}$$

Based on the Bayes’ theorem with strong assumptions regarding feature independence, Naïve Bayes classifiers are statistically known as probabilistic classifiers. When it comes to Bayesian networks, kernel density estimation is one of the simplest and most accurate (Fig. 7).

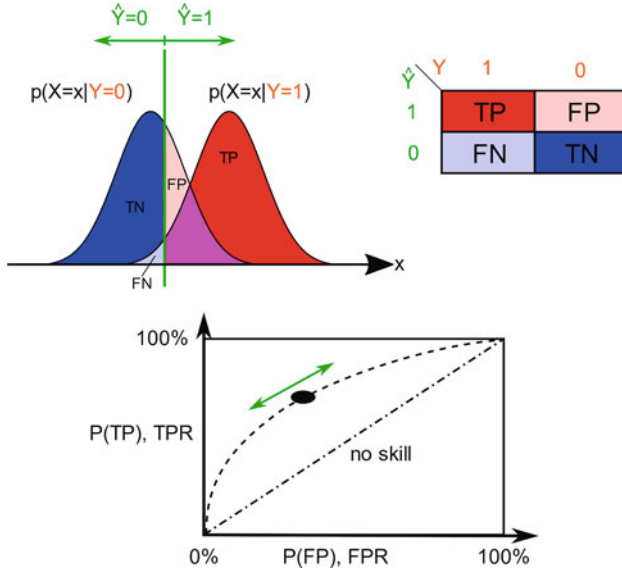


Fig. 7 Naïve Bayes classifier

Stochastic Gradient Descent

The stochastic gradient descent algorithm takes the count vectors as input and calculates the mean errors called epoch until the error rate is minimum. It updates the table after each calculation until no further calculation is needed.

The stochastic algorithm does not need to remember which examples were visited during the previous iterations, it can process examples on the fly in a deployed system. In such a situation, the stochastic gradient descent directly optimizes the expected risk, since the examples are randomly drawn from the ground truth distribution [26].

The stochastic gradient descent technique is used to increase the smoothness of an objective function via iteration. Gradient descent optimization may be regarded as a stochastic approximation since it uses an estimate of the gradient rather than

the true gradient (which is determined from the whole data set). This decreases the processing cost in high-dimensional optimization problems, allowing for quicker iterations in return for a lower convergence rate due to the lower calculation cost.

The principles of convex minimization and stochastic approximation theory were used to examine the convergence of stochastic gradient descent. When the target function is convex or pseudo convex, stochastic gradient descent converges to a global minimum with a high probability; when the goal function is neither convex nor pseudo convex, stochastic gradient descent converges to a local minimum with a high probability (Fig. 8).

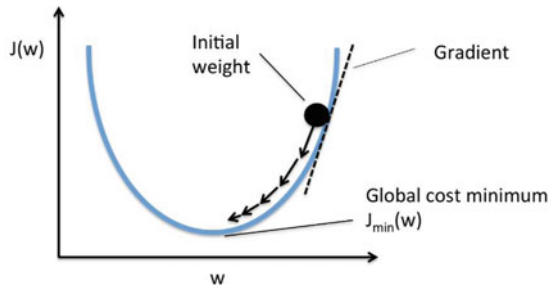


Fig. 8 Stochastic gradient descent classifier

K-Nearest Neighbour

K-nearest neighbours is one of the simplest machine learning algorithms based on the supervised learning technique. It can be used for both regression and classification, but it is mainly used for classification problems. It is also known as a lazy learning algorithm because it does not immediately learn from the training set, but instead saves the data set and performs an action on the data set at the time of predicting the major class. Here, K is the parameter related to the number of closest neighbours to include in the majority of the voting process [27].

The function is only approximated locally in k-NN classification, and all computation is postponed until the function evaluation is done. If the features are expressed in different physical units or at considerably different sizes, it is required to normalize the training data in order to improve the accuracy of this approach.

The technique of neighbour weighting may be used to both classification and regression problems, with the closer neighbours contributing more to the overall average than the farther neighbours. A collection of items is used to draw the neighbours if the class is known in the case of k-NN classification or the attribute value of an object is known when it comes to k-NN regression.

An Outlier of K-NN

A typical outlier score in anomaly identification is the distance to the k th nearest neighbour, which may be used to estimate local density. It is more likely that a query point is an outlier if it is far from the nearest k -NN. In comparison to more modern and advanced strategies, large-scale experimental research demonstrates that this outlier model and another traditional data mining strategy, the local outlier factor, perform rather well (Fig. 9).

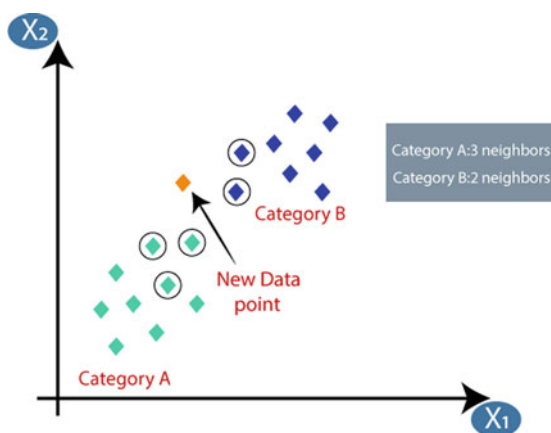


Fig. 9 K-nearest neighbour classifier

3 Materials and Methods

This section describes the dataset used for the detection of cyberbullying on Twitter, the proposed method to perform analytics on the selected dataset and discusses the evaluation metrics applied on the classification algorithms.

3.1 Dataset

The dataset with 11,091 records is taken as input along with another dataset with 24,784 records. These 2 datasets are combined to form a new dataset and tweets are distributed in the dataset as offensive and non-offensive. The new dataset has total 35,787 records in that 75% is given as training dataset and 25% is given as testing dataset input to the algorithms. The algorithms classify the data and detect the bullies [32].

3.2 Model Overview

Figure 10 describes the proposed model for cyberbullying detection that consists of pre-processing, classification and evaluation phases which are explained below.

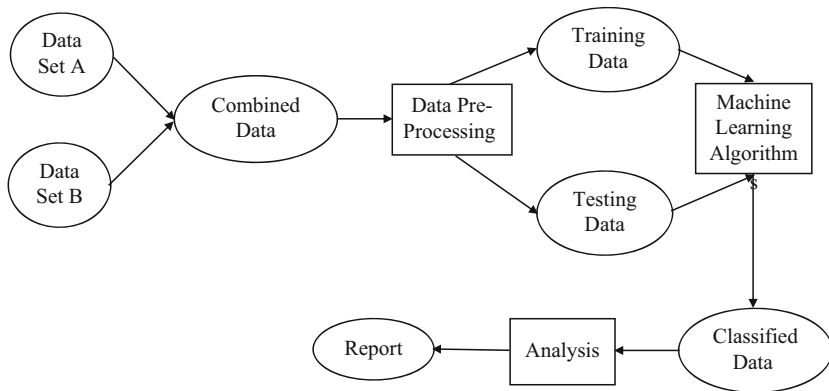


Fig. 10 System architecture

3.3 Pre-processing

The pre-processing is stop word removal from the dataset to reduce the noise present in the text data. This results in a clean tweet that can be given as an input for the classification algorithms.

3.4 Evaluation Metrics

The effectiveness of a proposed model can be determined by applying a few evaluation metrics to calculate how accurately the model can differentiate cyberbullying from non-cyberbullying. In this research, nine machine learning algorithms have been constructed, namely random forest, support vector machine, bagging classifier, naive Bayesian, k-nearest neighbour, logistic regression, decision tree, AdaBoost and stochastic gradient descent. So, to review these models, the standard evaluation metrics used by the research community are applied on them.

The most widely used metrics for evaluating the classifiers are as follows:

Accuracy

Accuracy is the proportion of correct classifications from the overall number of cases [7].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

Precision

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations [11].

$$Precision = \frac{TP}{TP + FP} \quad (3)$$

Recall

Recall is the ratio of correctly predicted positive observations to all observations in actual class [11].

$$Recall = \frac{TP}{TP + FN} \quad (4)$$

F1-Score The F1-Score is defined as a harmonic mean of precision and recall [11].

$$F1\ Score = \frac{2 * (Precision * recall)}{Precision + recall} \quad (5)$$

4 Results and Discussion

In this section, we have analysed the parameters (accuracy, precision, recall, F1-Score and prediction time) on the nine algorithms. The analysis is defined in Table 1 where the best and worst algorithms are analysed. In this, bagging and SGD are showing best accuracy on the given problem statement. The model is trained using an SGD classifier.

Table 1 Comparison of various machine learning algorithms with respect to the given problem statement

No.	Algorithm	Accuracy	Precision	Recall	F1-Score
1	Bagging classifier	0.927797	0.965493	0.923129	0.943836
2	SGD classifier	0.927462	0.961044	0.927211	0.943824
3	Logistic regression	0.926344	0.964089	0.922279	0.942721
4	Decision tree classifier	0.923438	0.952132	0.930272	0.941075
5	Linear SVC	0.916732	0.946599	0.92551	0.935936
6	Random forest classifier	0.910137	0.95112	0.910034	0.930123
7	Ada boost classifier	0.907567	0.972508	0.884354	0.926338
8	Multinomial NB	0.893372	0.901663	0.940306	0.920579
9	K neighbours classifier	0.857606	0.895161	0.887245	0.891186

The proposed model uses nine machine learning techniques that were set to achieve better accuracy. As per the given results, it can be inferred that the bagging classifier gave the best accuracy (92.77%) and F1-Score (94.38%) on the given dataset. Whereas SGD, logistic regression (92.63%) and decision tree (92.34%) had a fine difference in their accuracy where SGD (92.74%) got the accuracy which is somewhat better than the logistic regression (92.63%) and decision tree (92.34%) but less than bagging classifier (92.77%). Here, logistic regression achieved better precision (96.40%) while the decision tree achieved better recall (93.02%) values compared to SGD (96.10%, 92.72%) based on Fig. 11.

It can also be inferred that linear SVC (91.67%) and random forest (91.01) got similar accuracy values, but random forest (95.11%) got better precision value compared to linear SVC (94.65%). Though Adaboost classifier achieved less accuracy (90.75%), recall (97.25%) and F1-Score (92.63%) values, it has got the highest precision (97.25%) value among all nine algorithms based on Fig. 12.

Naïve Bayes (94.03%) achieved the highest recall value among all the nine algorithms based on Fig. 13. K-nearest neighbour algorithm achieved the least accuracy (85.75%); the bar chart of accuracy is presented in Fig. 11, which shows that K-nearest neighbour algorithm has the least accuracy.

From Fig. 11, it can be inferred that bagging classifier and SGD classifier achieve similar accuracies of 0.9277 and 0.9274 respectively compared to other algorithms.

From Fig. 12, it can be inferred that AdaBoost classifier achieves better precision of 0.972 individually when compared to other algorithms.

From Fig. 13, it can be inferred that multinomial NB achieves better recall value of 0.940 solely when compared to other algorithms.

From Fig. 14, it can be inferred that bagging classifier and SGD achieve similar and best F1-Score of 0.942836 and 0.943824 respectively when compared to other algorithms.

From Fig. 15, it can be inferred that bagging classifier got 41.75 the worst training time when compared to other algorithms.

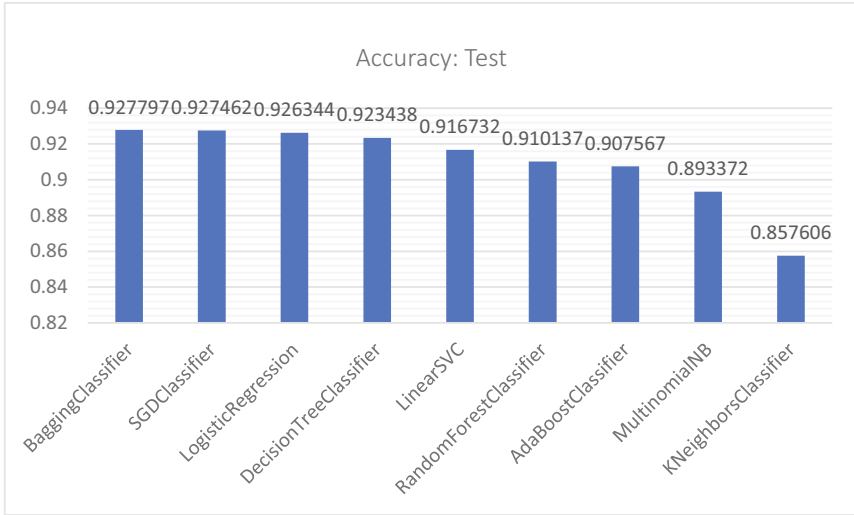


Fig. 11 Bar chart of accuracy with respect to the nine classifiers

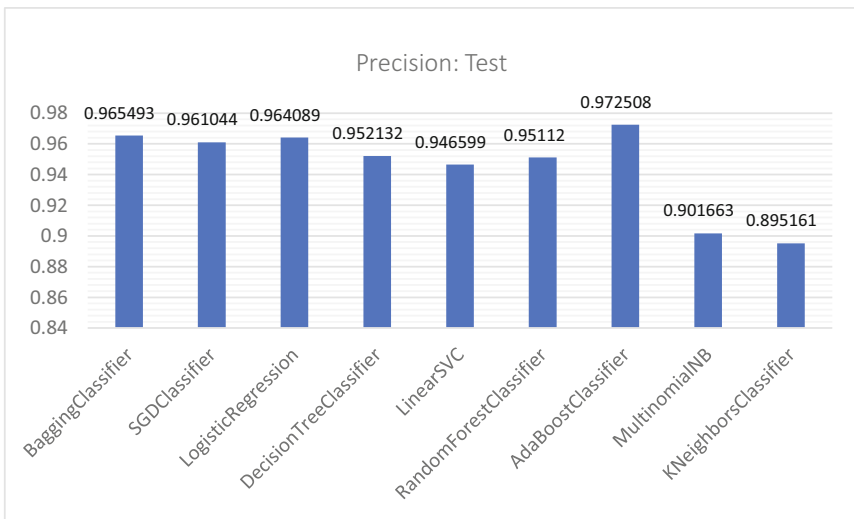


Fig. 12 Bar chart of precision with respect to the nine classifier

From Fig. 16, it can be inferred that the iterations are taken in a range 25 to 1000 at an interval of 25, 50, 100 and 200 respectively. It can be observed that the highest accuracy is achieved at 1000th iteration with a value of 0.9274.

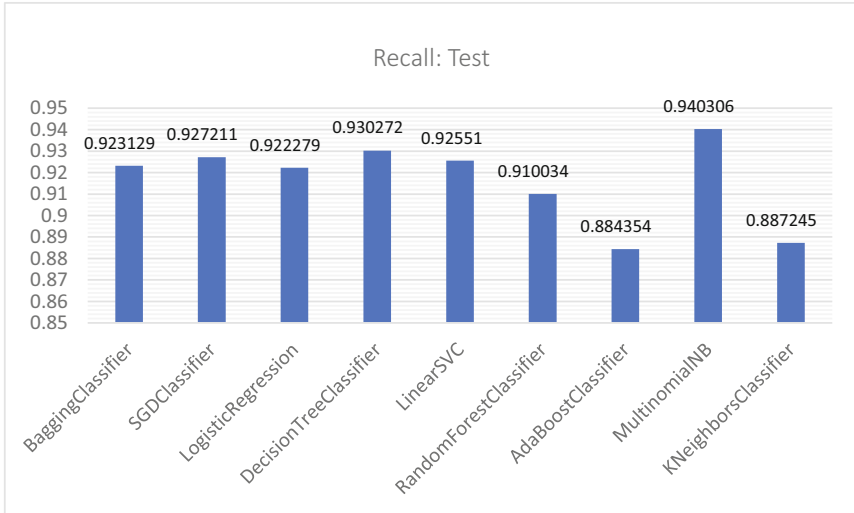


Fig. 13 Bar chart of recall with respect to the nine classifiers

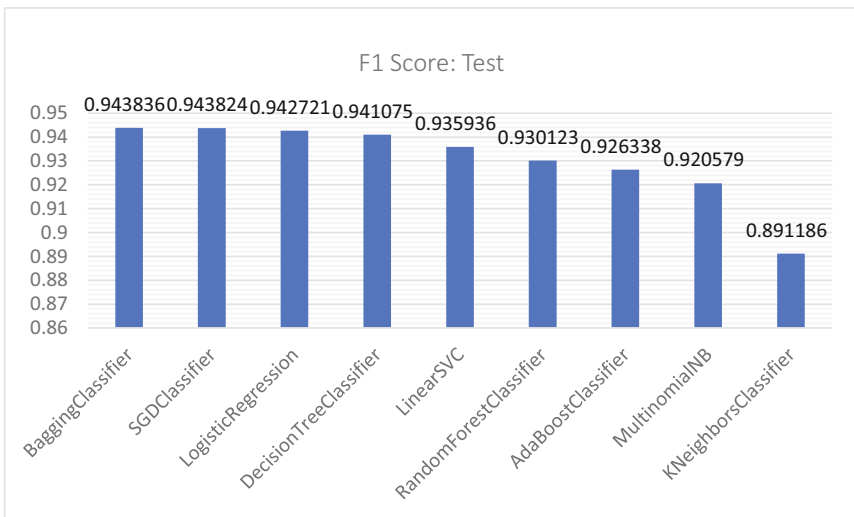


Fig. 14 Bar chart of F1-Score with respect to the nine classifiers

5 Conclusion

As per the analysis of the overall result, the stochastic gradient descent gave better performance compared to all other algorithms after taking different ratios of training and testing datasets. SGD gave 92.7% accuracy which is the highest among them.

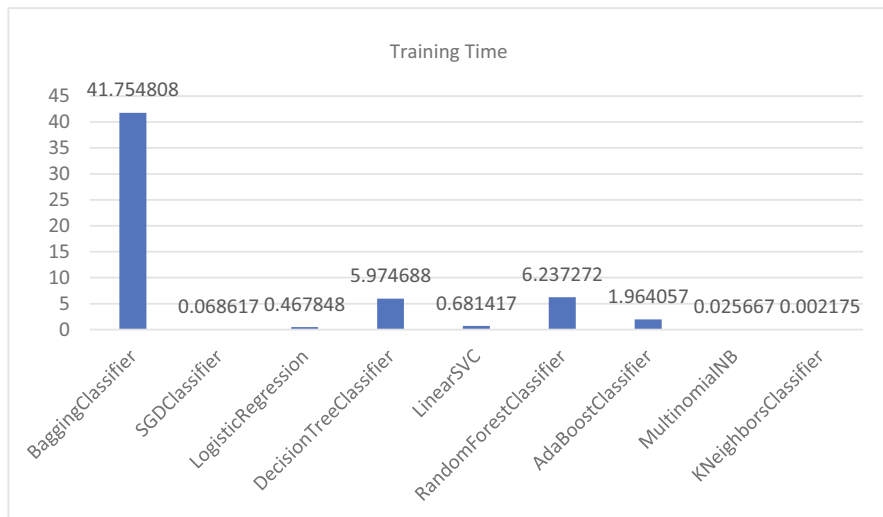


Fig. 15 Time complexity of algorithms

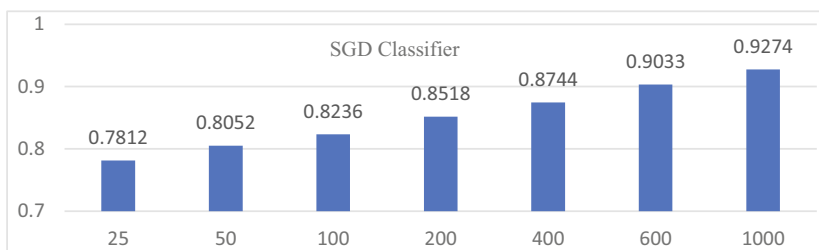


Fig. 16 Iterations of SGD classifier

Hence, SGD can be considered the best suitable model for the detection of cyberbullying among all nine algorithms. As per the analysis, we used an SGD classifier to train our model.

References

1. K.Reynolds, A. Kontostathis, L. Edwards (2011). Using Machine Learning To Detect Cyberbullying. 10th International Conference On Machine Learning And Applications And Workshops 2011.
2. Rahat Ibn Rafiq, Homa Hosseinmardi, Richard Han, Qin Lv, Shivakant Mishra. Scalable And Timely Detection Of Cyberbullying In Online Social Networks. *Sac '18: Proceedings Of The 33rd Annual Acm Symposium On Applied Computing*. April 2018 Pages 1738–1747.
3. Dadvar, M., de Jong, F. M. G., Ordelman, R. J. F., & Trieschnigg, R. B. (2012). Improved cyberbullying detection using gender information. In Proceedings of the Twelfth Dutch-Belgian Information Retrieval Workshop (DIR 2012) (pp. 23–25). Ghent University.

4. ARICAK, Prof. Dr. Osman Tolga, Siyahhan, Sinem, Uzunhasanoglu, Aysegul, Saribeyoglu, Sevda, Ciplak, Songul, Yilmaz, Nesrin, Memmedov, Cemil (2008). Cyberbullying Among Turkish Adolescents. *Cyberpsychology & behavior: the impact of the Internet. multimedia and virtual reality on behavior and society*.
5. Muneer, Amgad, Fati, Suliman (2020). A Comparative Analysis Of Machine Learning Techniques For Cyberbullying Detection On Twitter. *Future Internet*. Vol. 12, No. 11, P. 187, 2020.
6. T. Arora, M. Sharma, S. K. Khatri (2019). Detection Of Cyber Crime On Social Media Using Random Forest Algorithm. *Environment And Intelligent Control (PEEIC)*. Pp. 47–51.
7. Ahmad, M. Yousaf, S. Yousaf, M. Ahmad(2020). Fake News Detection Using Machine Learning Ensemble Methods. *Complexity*. Vol. 2020, Pp. 1-11.
8. K. Poddar, G. B. Amali D., K. S. Umadevi (2019). Comparison Of Various Machine Learning Models For Accurate Detection Of Fake News. *Innovations In Power And Advanced Computing Technologies (I-Pact)*. Pp. 1–5.
9. L. Kai, Z. Zhiping(2012). Using An Ensemble Classifier On Learning Evaluation For E-Learning System. *International Conference On Computer Science And Service System*. Pp. 538–541.
10. Irfan Kareem, S. M. Awan (2019). Pakistani Media Fake News Classification Using Machine Learning Classifiers. *International Conference On Innovative Computing (ICIC)*. Pp. 1–6.
11. Abdullah-All-Tanvir, E. M. Mahir, S. Akhter, M. R. Huq (2019). Detecting Fake News Using Machine Learning And Deep Learning Algorithms. *7th International Conference On Smart Computing & Communications (ICSCC)*. Pp. 1–5.
12. Brenda Irena, Erwin Budi Setiawan(2020). Fake News (Hoax) Identification On Social Media Twitter Using Decision Tree C4.5 Method. *Jurnal Resti (Rekayasa Sistem Dan Teknologi Informasi)*. Vol. 4, No. 4. Pp. 711-716.
13. Dinakar, K., Reichart, R., Lieberman, H. (2011). Modeling The Detection Of Textual Cyberbullying. *Social Mobile Web Workshop At International Conference On Weblog And Social Media*.
14. R. R. Dalvi, S. Baliram Chavan, A. Halbe(2020). Detecting A Twitter Cyberbullying Using Machine Learning. *4th International Conference On Intelligent Computing And Control Systems (ICICCS)*. Pp. 297–301.
15. S Leitch, M Warren(2009). Security Issues Challenging Facebook. In *Australian Information Security Management Conference*.
16. Yin, D., Xue, Z., Hong, L., Davison, B.D., Kontostathis, A., Edwards, L. (2009). Detection Of Harassment On Web 2.0. *Proceedings Of The Content Analysis In The Web 2.0 (Caw2.0) Workshop At Www2009*.
17. Chisholm, J.F. (2006). Cyberspace Violence Against Girls And Adolescent Females. *Annals Of The New York Academy Of Sciences* 1087. 74-89.
18. J. Juvonen, E. Gross(2008). Extending The School Grounds?-Bullying Experiences In Cyberspace. *Journal Of School Health*. Vol. 78. No. 9. Pp. 496-505.
19. Quinlan, J. R. (1986). Induction Of Decision Trees. *Machine Learning*. 1: 81–106.
20. Strother H. Walker, David B. Duncan (1967). Estimation of the Probability of an Event as a Function of Several Independent Variables. *Biometrika*. Volume 54. Issue 1-2. June 1967. Pages 167–179.
21. Ho, T.K. (1995). Random Decision Forest. *Proceedings of the 3rd International Conference on Document Analysis and Recognition*. Montreal. 14–16 August 1995. 278–282.
22. Bartlett, Peter, Shawe-Taylor, John (1998). Generalization Performance of Support Vector Machines and Other Pattern Classifiers. In Schölkopf, Bernard, Burges, Christopher J C, Smola, Alexander J (eds.). *Advances in Kernel Methods - Support Vector Learning*. MIT Press. Cambridge. USA.
23. Breiman, Leo (1996). Bagging predictors. *Machine Learning* 24. 123–140.
24. McCallum, A., Nigam, K. (1998). A Comparison of Event Models for Naive Bayes Text Classification. In *Working Notes of AAAI 1998. Workshop on Learning for Text Categorization*.

25. Robert E. Schapire(1996). Experiments with a New Boosting Algorithm. Machine Learning: Proceedings of the Thirteenth International Conference.
26. Bottou L. (2012). Stochastic Gradient Descent Tricks. In: Montavon G., Orr G.B., Müller KR. (eds) Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science. vol 7700. Springer.
27. T. Cover, P. Hart(1967). Nearest neighbor pattern classification. In IEEE Transactions on Information Theory. vol. 13. no. 1. pp. 21–27.
28. Febriana Trisna, Budiarto Arif (2019). Twitter Dataset for Hate Speech and Cyberbullying Detection in Indonesian Language. International Conference on Information Management and Technology (ICIMTech). 379–382. doi:<https://doi.org/10.1109/ICIMTech.2019.8843722>.
29. Bashir Shaikh Farhan, Rehman Mobashar, Amin Aamir (2020). Cyberbullying: A Systematic Literature Review to Identify the Factors Impelling University Students Towards Cyberbullying. IEEE Access, 8(), 148031–148051. doi:<https://doi.org/10.1109/access.2020.3015669>.
30. Ting I-Hsien, Liou Wun Sheng, Liberona Dario, Wang Shyue-Liang, Bermudez Giovanni Mauricio Tarazona (2017). Towards the detection of cyberbullying based on social network mining techniques. International Conference on Behavioral, Economic, Socio-cultural Computing (BESC).1–2. doi:<https://doi.org/10.1109/BESC.2017.8256403>.
31. Yu Weider D, Gole Maithili, Prabhuswamy Nishanth, Prakash Sowmya, Shankaramurthy Vidya Gowdru (2016). An Approach to Design and Analyze the Framework for Preventing Cyberbullying. IEEE International Conference on Services Computing (SCC). 864–867. doi:<https://doi.org/10.1109/SCC.2016.125>.
32. URL – <https://github.com/ZeeraKW/hatespeech/archive/master.zip> visited this site on 20 November 2020.

Sentiment Analysis of Political Tweets for Israel Using Machine Learning



Amisha Gangwar and Tanvi Mehta

Keywords Sentiment analysis · Public opinion · Politics · Machine learning · Twitter · Social networks · Israeli conflict

1 Introduction

With the rapid growth of online social networks (OSNs), communication platforms have become popular. This has helped large numbers of the population share, search, and interchange data and information without any concern about geographical distance [1]. The volume of data created through social media platforms, especially Twitter, is massive. Twitter is an online news and social networking site where people communicate in short tweets [2]. It has become the most popular social media platform, with millions of users posting tweets each day.

The data relating to public opinion has increased tremendously [3]. Identification of these opinions regarding political events or issues is essential to form international alliances, policies, and positions. The actions of government officials depend on these opinions, so they should keep an eye on this data to make future decisions [4]. Opinion polls have been the standard mechanism for collecting public opinion, often giving rise to several problems. It is not easy to interpret fine-grained analysis or learn the intentions, subjectivity, and motivations behind public opinion with polls [5]. All the aforementioned drawbacks make opinion polls not very well-grounded, and there is a requirement for advanced mechanisms to understand such public views.

The massive number of users, the diversified topics, and enormous volumes of posted tweets or content have resulted in social media becoming a rich source to predict the population's attitudes [6]. Mining or using advanced techniques for

A. Gangwar

Machine Learning Researcher at LearnByResearch, Bareilly, Uttar Pradesh, India

T. Mehta (✉)

Machine Learning Researcher at LearnByResearch, Pune, Maharashtra, India

political opinions may provide a faster, more accurate, and less expensive alternative to traditional polls [7]. Several research works have explored social media mining to analyze and predict public political opinions. However, these research works were mostly event-specific and used only relevant techniques to investigate the issue [8]. Furthermore, most of the studies depend on sentiment analysis to predict the user's feelings rather than the political opinion.

In politics, analyzing a sentiment depends on the side one stands by regardless of the users' views [9]. Hence, a sentiment analyzer may classify a tweet, phrase, comment, or idiom as "negative" or "positive." Furthermore, few researchers have analyzed current situations statistically instead of making predictions about public opinions [10]. This paper proposes an analytical and comparative study of the Twitter text dataset to examine the public political opinion in several countries toward the Israelis in the Palestinian-Israeli conflict. The data worked upon is dated May 2021. The proposed method uses the machine learning model based on support vector classifier (SVC), decision tree (DT), and Naïve Bayes (NB) algorithms, and the results show a comparative analysis of these algorithms.

In Sect. 2, a summary of the previous research works is presented. In Sect. 3, the techniques used in the proposed research are briefly described. Section 4 describes the implementation of the work. The analytical outcomes and evaluation are tabulated in Sect. 5. The conclusions and future scope are discussed in Sect. 6.

2 Literature Review

Bhawani Selvaretnam et al. in 2017 [11] designed an experiment that extracts the sentiments based on the subjects in tweets using NLP (Natural Language Processing). This was done in three steps: classification, semantic association, and polarity sort by identifying the grammatical relationship between subject and sentiment lexicon. SVM technique resulted in better accuracy.

Rajon Bardhan et al. in 2018 [12] presented sentiment analysis on user comments based on Natural Language Processing (NLP) used to generate datasets. The solution resulted in a data-driven experiment in the accuracy of finding the popular and high-quality video. The efficiency obtained through this approach was 75.44%.

Risky Novendri et al. in 2020 [13] presented a solution for analyzing viewers' comments on Netflix series episodes through sentiment analysis using the Naïve Bayes algorithm. The results were 81%, 74.83%, and 75.22% for accuracy, precision, and recall, respectively.

Hoda Korashy et al. in 2014 [14] has done a comprehensive survey of sentiment analysis. Most of the proposed research and algorithms are being investigated. They survey various methods like transfer learning emotion detection, including sentiment analysis.

Nhan Cach Dang et al. in 2021 [15] reviewed the study of sentiment analysis problems solved using deep learning, like sentiment polarity. They applied TF-IDF

and word embedding techniques to the datasets. The comparative analysis is done based on the experimental results. The research concluded that RNN is reliable.

Van Cuong Tran et al. in 2018 [16] explored the new approach based on a feature ensemble model containing fuzzy sentiment related to tweets by considering lexical and position of words by polarity. They experimented on actual data to study the performance of various algorithms.

Matheus Araújo et al. in 2014 [17] have compared eight sentiment analysis methods in the context of agreement and coverage. They developed a new way that combined best agreement and coverage results. They also presented iFeel, a free Web service that provides an open-access API through which to compare different sentiment analysis methods.

Angelika Maag et al. in 2019 [18] showed that sentiment analysis improves granularity at the aspect level. They explored aspect extraction and sentiment classification of target-dependent tweets using deep learning techniques. A comparative study of deep learning algorithms like CNN, LSTM, and GRU, for aspect-based analysis, was done.

Apoorv Agarwal et al. in 2018 [19] performed Twitter data sentiment analysis by introducing features like POS (Specific Prior Polarity) and explored the solution for feature engineering by introducing a tree kernel that completed the unigram baseline. In a feature-based approach, they concluded that important features combine the part-of-speech and prior-polarity of words.

Xiaoyu Qiu et al. in 2018 [20] has worked on improving word representation methods. This method integrated sentiment information into the traditional TF-IDF algorithm and generated weighted word vectors. The sentiment analysis was done using RNN, CNN, LSTM, and NB. The results obtained proved that the proposed method gave the highest accuracy of 92.18%.

Osama Abu-Dahrooj et al. in 2019 [21] studied data analysis models exploring country-level analysis and individual-level analysis of Palestine Tweets. The data was analyzed based on the activity and sentiments on the country and individual comments. They used a multi-level analysis technique for this approach.

Shahla Nemati et al. in 2021 [22] proposed a method for sentiment analysis using Attention-based Bidirectional CNN-RNN Deep Model (ABCDDM) on five datasets. Independent bidirectional layers like LSTM and GRU were used, ABCDDM did the extraction of context by applying attention mechanisms on the output of layers. Polarity detection was used to calculate the effectiveness of ABCDDM. The accuracy obtained by ABCDDM was 96.8%.

3 Techniques

The proposed techniques for Twitter Sentiment Analysis are briefly explained in this section. The techniques are as follows.

3.1 Support Vector Classifier (SVC)

SVC is applied for classification challenges. The data is segregated using the best decision boundary in this supervised algorithm, i.e., the hyperplane. It selects the extreme points called support vectors that help construct the hyperplane. It also has a positive hyperplane that passes through one (or more) of the nearest positive attributes and a negative hyperplane that passes through one (or more) of the nearest negative points [23]. The optimal hyperplane is where the margin, distance between positive and negative hyperplane, is maximum.

3.2 Decision Tree (DT)

The decision tree is not only applied for classification but also regression challenges. As the name suggests, this supervised algorithm uses the tree structure to depict the predictions that result from a series of element-based splits. It begins with the root node and ends with a decision made by leaves. It acquires basic terminologies like root nodes, leaf nodes, decision nodes, pruning, and sub-tree [24]. This technique possesses a bunch of if-else statements where it checks if the condition is true; if yes, it moves forward to the next node attached to that decision.

3.3 Naïve Bayes (NB)

Naïve Bayes, being a supervised algorithm, is based on Bayes Theorem and is applied for classification challenges. It is mainly used in text classification, including high-dimensional training datasets [25]. It helps build fast and accurate machine learning models and make quick predictions.

3.4 Bag of Words (BoW)

The BoW is a method of representing text data when modeling text with machine learning techniques. This model turns arbitrary text into fixed-length vectors by counting how many times each word appears. This process is known as vectorization. This simple implementation is used to solve problems related to language modeling and document classification by extracting text features. It offers much flexibility for customization on specific text data.

4 Methodology and Implementation

This section delineates the usual process of implementing sentiment analysis. The schematic representation of the proposed method is demonstrated in Figure 1. A brief discussion of the flow is done below:

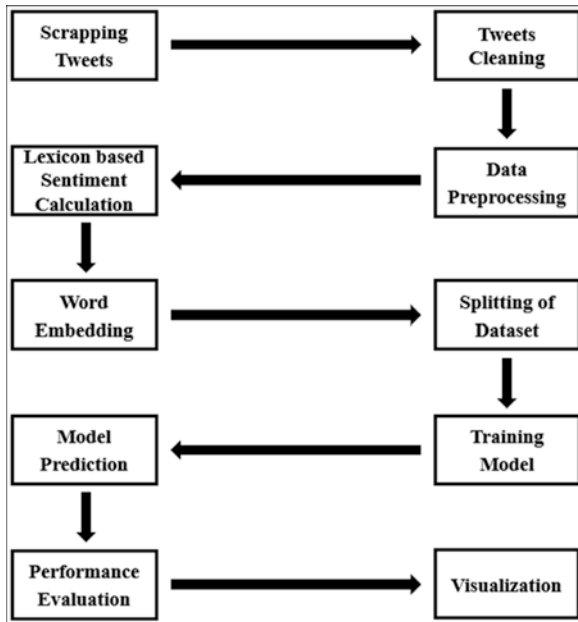


Fig. 1 Schematic representation of methodology

4.1 Dataset Scrapping

The dataset was scraped using Twitter Intelligence Tool (TWINT) from 6 May 2021 to 21 May 2021; 37,530 English tweets were extracted with hashtags such as #IsraelUnderAttack, #IStandWithIsrael, #WeStandWithIsrael, and #Israel-Palestineconflict. Figure 2 depicts the world cloud of our collected dataset.

4.6 *Splitting Dataset*

In this step, the required dataset gets bifurcated into training and testing sets. The training–the testing proportion is 80–20%, respectively.

4.7 *Training Model*

This step comprises implementing a classification model with the help of classifiers like SVC, DT, and NB.

4.8 *Model Prediction*

Various models like SVC, DT, and NB were used and classification reports and confusion matrix were predicted.

4.9 *Performance Evaluation and Visualization*

The accuracy metrics and confusion matrices analyzed the performance of various models. The classifier obtaining higher accuracy will be the most efficient one. The confusion matrices are plotted using a heatmap. Furthermore, a comparative analysis of these accuracies was done.

5 Experimental Results and Analysis

Figure 3 shows the subjective and polarity of the cleaned tweets, which suggests our dataset contains a great range of tweets.

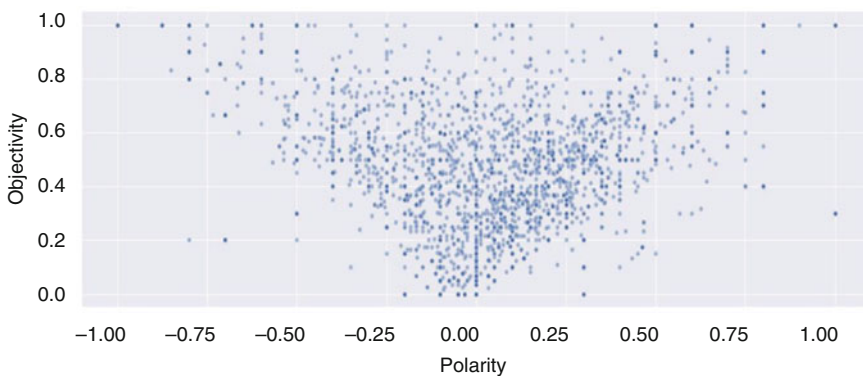


Fig. 3 Subjective versus Polarity plot

The results achieved from all the models are precision, recall, and F1 score. The tweets are distinguished into positive, neutral, and negative. These results with accuracies are tabulated in Tables 1 and 2. The graph of the accuracy of all models is demonstrated in Figure 4. It shows that NB obtains the highest accuracy of 93.21%.

Table 1 Comparative analysis of results

Models	Positive			Neutral			Negative		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Precision	Recall	F1 Score
SVC	0.92	0.79	0.85	0.91	0.97	0.94	0.81	0.82	0.82
DT	1.00	0.96	0.98	0.91	0.97	0.94	0.84	0.67	0.75
NB	1.00	1.00	1.00	0.98	0.90	0.94	0.73	0.93	0.82

Table 2 Comparative analysis of accuracy

Models	SVC	DT	NB
Accuracy	0.8976	0.9223	0.9321

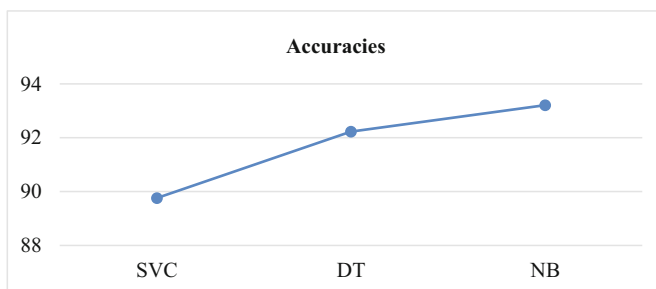


Fig. 4 Accuracy plot of models

The confusion matrix gives insights into the performance of machine learning models on a test set whose actual labels are known. In other words, it tells how much the model is confused about the loaded data set. The confusion matrices of SVC, DT, and NB are shown in Figures 5, 6, and 7, respectively.

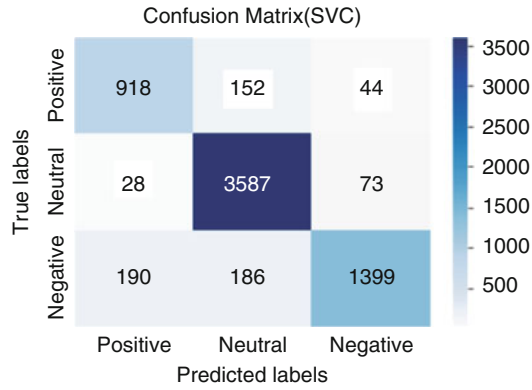


Fig. 5 Confusion matrix of SVC

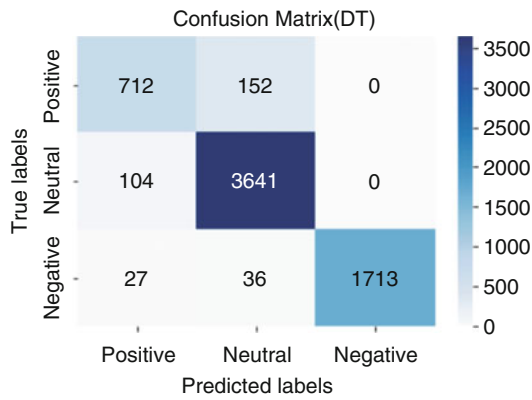


Fig. 6 Confusion matrix of DT

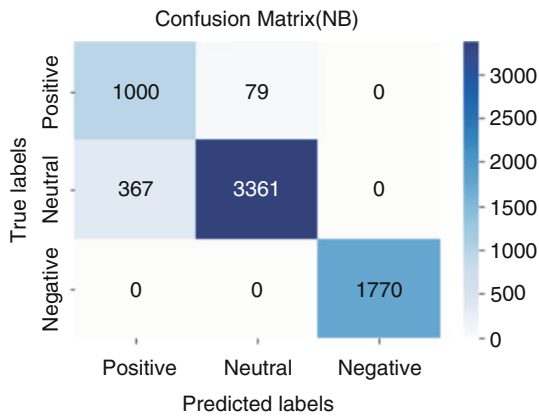


Fig. 7 Confusion matrix of NB

6 Conclusion

The research proposes the Twitter Sentiment Analysis of tweets about Israel toward Palestinian-Israeli conflict using advanced technology like machine learning. The techniques like support vector classifier (SVC), decision tree (DT), and Naïve Bayes (NB) were used to train the model. The result achieved was that Naïve Bayes (NB) obtained the highest accuracy of 93.21%. As part of this research, an attempt has been made to study and implement various algorithms to enhance the feasibility of sentiment analysis where massive amounts of data can be analyzed without much time consumption. The aim is to explore and perform future research on sentiment analysis using deep learning neural networks. This study, hence, studies all the features and elements, implements them, and makes an entirely accurate prediction. Therefore, this will overall contribute to making future decisions about political events.

Acknowledgments This research is sponsored by Learn By Research Organization, India.

References

1. Alamoodi, A. H. et al.; "Sentiment Analysis and Its Applications in Fighting COVID-19 and Infectious Diseases: A Systematic Review."; *Expert Systems with Applications*; 2021.
2. Feldman, Ronen; "Techniques and Applications for Sentiment Analysis: The Main Applications and Challenges of One of the Hottest Research Areas in Computer Science."; *Communications of the ACM*; 2019.
3. Rahmatika et al.; "The Effectiveness of Youtube as an Online Learning Media"; *Journal of Education Technology*; 2021.
4. Tafesse, Wondwesen; "YouTube Marketing: How Marketers' Video Optimization Practices Influence Video Views."; *Internet Research* 30.6; 2020.
5. Bozkurt et al.; "Cleft Lip and Palate YouTube Videos: Content Usefulness and Sentiment Analysis."; *Cleft Palate-Craniofacial Journal* 58.3; 2021.
6. Al-Sarraj et al.; "Bias Detection of Palestinian/Israeli Conflict in Western Media: A Sentiment Analysis Experimental Study."; *International Conference on Promising Electronic Technologies*; 2018.
7. Cambria, Erik; "Affective Computing and Sentiment Analysis."; *IEEE*; 2016.
8. Yadav, Ashima et al.; "Sentiment Analysis Using Deep Learning Architectures: A Review."; *Artificial Intelligence Review* 53.6; 2020.
9. Doaa Mohey et al; "A Survey on Sentiment Analysis Challenges."; *Journal of King Saud University - Engineering Sciences* 30.4; 2018.
10. Rudy, et al.; "Sentiment Analysis: A Combined Approach"; *Journal of Informetrics*; 2009.
11. Chong; "Natural Language Processing for Sentiment Analysis: An Exploratory Analysis on Tweets."; *ICALET*; 2014.
12. Bhuiyan, Hanif et al.; "Retrieving YouTube Video by Sentiment Analysis on User Comment."; *ICSIPA*; 2017.
13. Novendri, Risky et al.; "Sentiment Analysis of YouTube Movie Trailer Comments Using Naïve Bayes."; *Bulletin of Computer Science and Electrical Engineering* 1.1; 2020.
14. Medhat et al.; "Sentiment Analysis Algorithms and Applications: A Survey."; *Ain Shams Engineering Journal* 5.4; 2014.

15. Dang, Nhan Cach, María N. Moreno-García, and Fernando De la Prieta; “Sentiment Analysis Based on Deep Learning: A Comparative Study.”; *Electronics (Switzerland)*; 2020.
16. Phan, Huyen Trang et al.; “Improving the Performance of Sentiment Analysis of Tweets Containing Fuzzy Sentiment Using the Feature Ensemble Model.”; *IEEE Access* 8; 2020.
17. Gonçalves, Pollyanna et al.; “Comparing and Combining Sentiment Analysis Methods.”; *COSN 2013 - Association for Computing Machinery*; 2013.
18. Do, Hai Ha et al.; “Deep Learning for Aspect-Based Sentiment Analysis: A Comparative Review.”; *Expert Systems with Applications*; 2019.
19. Srivastava, Ankit et al.; “Sentiment Analysis of Twitter Data: A Hybrid Approach.”; *International Journal of Healthcare Information Systems and Informatics*; 2019.
20. Xu, Guixian et al.; “Sentiment Analysis of Comment Texts Based on BiLSTM.”; *IEEE Access* 7; 2019.
21. Al-Agha, Iyad, and Osama Abu-Dahrooj; “Multi-Level Analysis of Political Sentiments Using Twitter Data: A Case Study of the Palestinian-Israeli Conflict.”; *Jordanian Journal of Computers and Information Technology*; 2019.
22. Basiri, Mohammad Ehsan et al.; “ABCDM: An Attention-Based Bidirectional CNN-RNN Deep Model for Sentiment Analysis.”; *Future Generation Computer Systems*; 2021.
23. Vishal A. Kharde et al.; “Sentiment Analysis of Twitter Data: A Survey of Techniques”; *International Journal of Computer Applications*; 2017.
24. G. Gautam et al.; “Sentiment analysis of twitter data using machine learning approaches and semantic analysis”; *Seventh International Conference on Contemporary Computing (IC3)*; 2014.
25. Abdullah Alsaeedi et al.; “A Study on Sentiment Analysis Techniques of Twitter Data”; *International Journal of Advanced Computer Science and Applications*; 2019.

A Novel Approach for Real-Time Vehicle Re-identification Using Content-Based Image Retrieval with Relevance Feedback



N. Shankaranarayan  and S. Sowmya Kamath 

Keywords Vehicle re-identification · Content-based image retrieval · Relevance feedback · Deep neural models

1 Introduction

The growing number of applications connected to smart cities and autonomous driving systems highlights a need for addressing prevalent problems such as vehicle re-identification, similar vehicle retrieval, etc. The process of detecting multiple instances of a specific target vehicle given a gallery of a large number of vehicles is known as vehicle re-identification. The re-identification procedure produces a ranked list of gallery images for each query, with the gallery images sorted in order of most probable match to least probable match. Previously, onboard IoT sensor devices were used to solve vehicle re-identification challenges. However, as compared to methods based on sensors, techniques based on visual and temporal indications have gained a lot of traction in recent years because sensors are unreliable due to their susceptibility to environmental changes. Techniques based on a vehicle's visual characteristics, such as local and global features and spatiotemporal cues, have been increasingly popular in recent years.

Several challenges arise during the process of vehicle re-identification [1]. Inter-class similarity and inter-class difference are two challenges that every vehicle re-identification model must handle. Inter-class similarity can affect performance in cases where two different vehicles are hard to differentiate from each other. Intra-class differences manifest when different viewpoints of the same vehicle are unrelatable, i.e., the same car from different angles might look different. Apart from this, the problem of vehicle re-identification also suffers from issues such as

N. Shankaranarayan (✉) · S. S. Kamath

Department of Information Technology, National Institute of Technology Karnataka, Mangaluru, Karnataka, India

e-mail: sowmyakamath@nitk.edu.in

illumination, occlusion, noisy detection, etc. Illumination issues can arise when the color factor of the vehicle is affected by the illuminator, i.e., the same color may look different under varying illumination. Occlusion can result when an object blocks the target vehicle, causing issues due to hidden features of the vehicle, affecting the re-identification model's learning and prediction performance. The noisy detection issue is often prevalent when the detected area of interest contains more than just the vehicle, such as background noise.

The researchers across multiple domains [2–4] have tried to overcome the issue of inter-class similarity and intra-class difference by utilizing the local and global features present in an image. Global features help the model to learn features related to the overall appearance of the vehicle, such as color and type, while local features help the model to learn object-specific characteristics (such as visible damage, design patterns, and dashboard items in case of vehicles). Current vehicle re-identification models adopt a computation-heavy multi-branch architecture to learn global as well as local features, where some branches focus on global features of the vehicle, while the rest of the branches focus on the local features of the vehicle [8, 9]. Some models utilized complex post-processing techniques such as part-aware re-ranking, tracklet level verification, re-ranking, etc., along with the global and local feature extraction [5, 11]. Despite significant advancement, these models struggle in terms of real-world implementation because of their complexity and computational requirements. This cost of computation is caused mainly because of the two-branch architecture employed for extracting local/global features and complex post-processing techniques.

To address this, we propose a lightweight architecture that attempts to eliminate the need for detailed local feature extraction to perform re-identification. Instead of using a dedicated branch to learn local features, we attempt to learn salient local features with the help of relevance feedback technique, which has not been attempted so far for the vehicle re-identification problem, to the best of our knowledge. The remainder of this chapter is structured as follows. Section 2 presents a detailed discussion on the existing work. In Sect. 3, the proposed methodology is explained in detail with reference to all defined processes. Section 4 describes the various experiments performed and results highlighting the validity of the proposed approach, followed by the conclusion and future scope.

2 Related Work

Machine learning (ML) algorithms have been extensively adopted for addressing the re-identification problem, but the use of handpicked global and local features adversely affected their performance. Deep learning (DL)-based methods outperformed the previous ML-based models, as they use neural models to learn distinctive features that are used for re-identification. One of the earliest works on DL-based vehicle re-identification is by Wang et al. [6], where an orientation-invariant feature embedding module was proposed. The local features were calculated based on 20

key points extracted using a CNN model to extract the feature vectors and fuse them with the vehicle's global feature vector. They also adopted spatiotemporal regulations for making the re-identification process more accurate. Liu et al. [7] proposed a region-aware deep neural model, where the model jointly made use of global features, vehicle ID, model, and color to train. The fused global and local vector resulted in better discrimination and accuracy than the previous models. Zhao et al. [10] addressed the re-identification problem by combining a classification and detection model, with the first stage being classification followed by detection. The features from lower layers of the classification model were used as input to the single-shot detection method for detecting 21 classes of vehicle attributes. This enabled the model to identify smaller discrimination local features well.

Another notable work is proposed by Chen et al. [11], and they used deep pyramid feature learning, which utilizes multiple branches to take different scales of the input image and a fusion branch to fuse the information at different scales. This was based on the fact that resizing the image leads to the loss of essential discriminating data. Chen et al. [12] proposed a model based on enhanced multi-granularity network. It consisted of a backbone network pretrained on the ImageNet dataset and five other branches, namely H1, H4, H5, V4, and V5, to extract features on different levels. H1 is used for global feature extraction, and the other four branches divide the feature map horizontally and vertically for fine-grained local feature extraction. The model utilized softmax loss for classification and triplet loss for metric learning. Zhu et al. [13] proposed a model named VOC-ReID, which focused on the failure cases caused because of similar background and shape, taking the vehicle's orientation and camera similarity as a whole and reforming background/shape similarity as camera/orientation re-identification. The model included three parts, namely Vehicle RECT-Net, orientation RECT-Net, and camera RECT-Net. Vehicle RECT-Net checks the vehicle similarity, orientation RECT-Net checks the orientation similarity, and camera RECT-Net checks the camera similarity. Finally, the output of these three layers is fused together to form the final distance matrix. The model utilized circle loss and triplet loss.

Huang et al. [14] proposed a viewpoint-aware temporal attention model that utilized vehicle orientation and attributes such as vehicle type, brand, and color. The model first extracts frame-level appearance and structure features that are later concatenated and used to train a temporal attention model. The output of this model is used to train the re-identification model on concatenation with the previous structure features. The authors adopted metric learning with batch sample triplet loss and cross-entropy loss. In addition, re-ranking with soft decision boundary is applied as post-processing for result refinement. Jiang et al. [15] introduced a part-aware structure-based model. They utilized a part detector to find specific parts of the vehicle, which explicitly introduced prior knowledge on the structure of the vehicle. Two models were trained, a global level model and a part level model. The features generated by these two models are later concatenated, and post-processing steps such as part-aware verification, tracklet verification, and re-ranking are implemented.

The detailed review revealed several gaps and issues with existing models. Though these models perform really well in an experimental setup, a real-world implementation may prove to be challenging because of the computational requirements. Moreover, most of this vehicle re-identification assumes zero external interaction. Here, in our proposed model, we introduce the use of relevance feedback as a post-processing technique. We demonstrate the effectiveness of blind and explicit feedback on a vanilla vehicle re-identification model.

3 Methodology

The proposed methodology's system architecture is depicted in Figs. 1 and 2 represents the proposed model in the form of a flow chart. The proposed methodology has three main parts, namely, feature extraction, ranking based on similarity matching, and re-ranking based on relevance feedback. In the proposed model, given the query set, the model generates a ranked list of gallery images for each query, with the gallery images sorted in order of most probable matches to least probable matches. This ranked list is generated with the help of the Euclidean distance between the feature vectors of the query images and gallery images. The feature vectors of both the query set and gallery set are extracted in advance before Euclidean distance calculation. Furthermore, relevance feedback is implemented as a post-processing technique and is used to provide feedback on the current ranked list. This feedback is used to optimize and reformulate the query set to obtain a better result.

Dataset Specifics For experimental validation of the proposed approach, we use VERI-776 dataset. It contains a collection of 49,357 images from 20 cameras that represent 776 vehicle IDs. The dataset is divided into training and test sets. The training set contains 37,778 images with 576 vehicle IDs, while the test set contains 13,257 images with 200 vehicle IDs. The test set is, in turn, divided into query and gallery, where the query contains 1678 images with 200 vehicle IDs and 11,579 gallery images with 200 corresponding vehicle IDs.

Feature Extraction After resizing the query and gallery images, we train a ResNet50 model [1] with a combination of hard positive/negative mining triplet loss (computed as per Eq. 1) and cross-entropy loss utilizing 37,778 images with 576 vehicle IDs. The model is purposefully made to consist of only one feature extraction branch to be lightweighted and to demonstrate the effectiveness of the relevance feedback as a post-processing technique. For the purpose of testing, the last layer of the trained model is removed to capture just the feature vectors. A total of 2048 features are extracted from each image and stored in a database, which are then used to compute the similarity between the query and gallery images. The feature vectors from the query and gallery images are stored in order to compute the distance between them.

$$T(A, P, N) = \max(\|f(A) - f(P)\|^2 - \|f(A) - f(N)\|^2 + \alpha, 0) \quad (1)$$

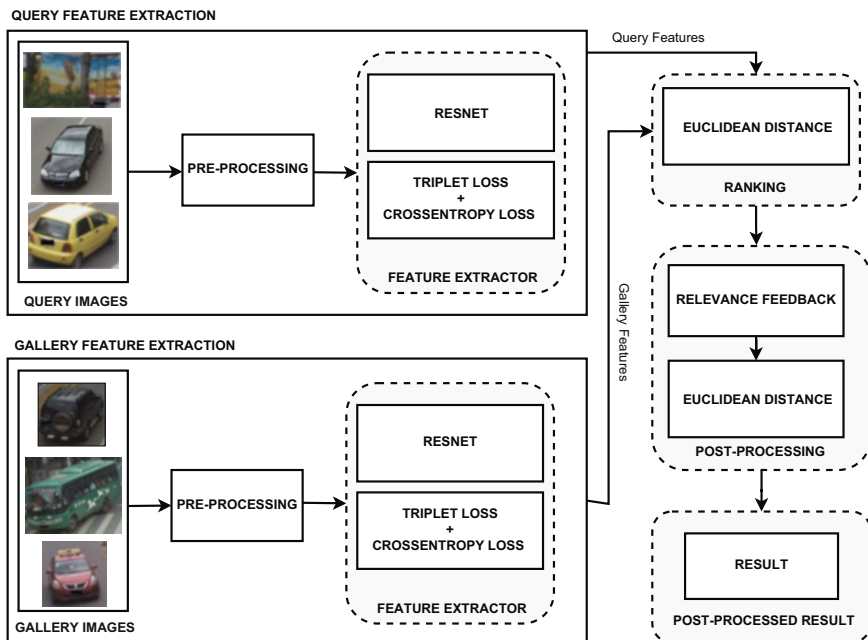


Fig. 1 Workflow of the proposed system

where A represents the anchor image, P represents the positive input that belongs to the same class as the anchor image, and N indicates the negative input that belongs to a different class. The triplet loss function aims to reduce the distance between the anchor A and the positive input P while increasing the distance between the anchor A and the negative input N .

Ranking Process The feature vectors obtained in the feature extraction phase are fed to the ranking algorithm to generate a ranked list of similar gallery images for every given query image. We use Euclidean distance to measure the similarity of the two vectors/images, i.e., a vector representing the query image and another vector representing a gallery image. The Euclidean distance is computed as per Eq. (2), giving the closeness between q and g , where q represents the query image and g represents the gallery image. The lower Euclidean distance between the query image and the gallery image indicates the high similarity among the features. Therefore, if the Euclidean distance is low, the corresponding image is ranked at the top among the retrieved output images.

$$E(q, g) = \sqrt{\sum_{i=1}^n (q_i - g_i)^2} \tag{2}$$

Relevance Feedback Process Relevance feedback enables the model to capture the results that are first returned from a query, collect relevance information, and then utilize that information to reconstruct a new query. The reconstructed query is expected to retrieve better results than the original query. In our work, we experiment with two kinds of feedback models, namely blind feedback and explicit feedback, to find the best fit for our problem.

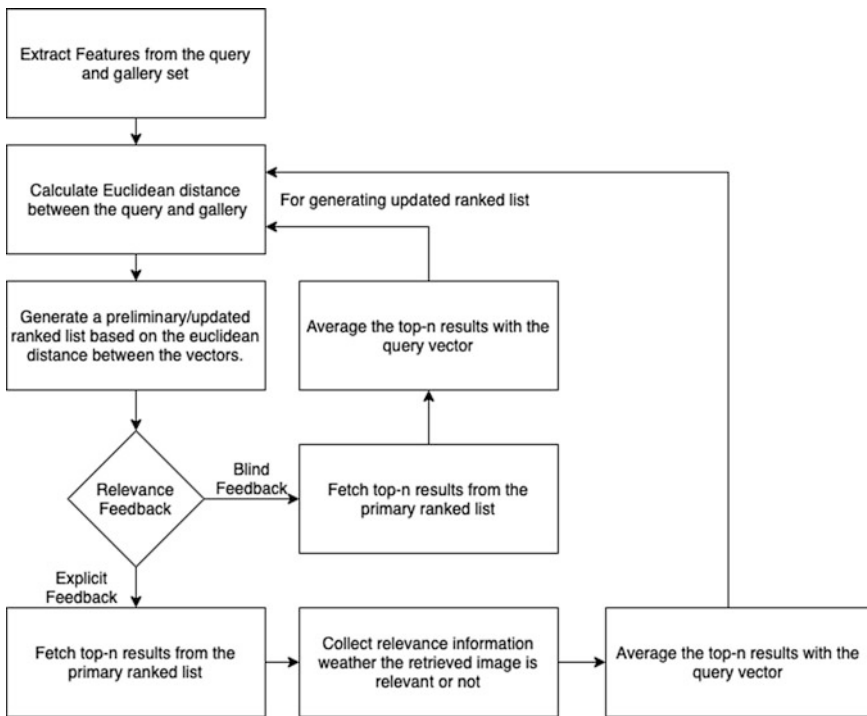


Fig. 2 Flowchart of the proposed model

1. *Explicit Feedback*: Explicit feedback utilizes information such as relevant and irrelevant results to reformulate the query. This is done by retrieving the feature vectors of the top n retrievals and averaging the feature vectors of the relevant retrieval results with the original query image’s feature vector.
2. *Blind Feedback*: Blind feedback automatically assumes the top n ranked results as relevant and reformulates the query. This is done by retrieving the feature

vectors of the top n results and averaging their feature with the original query image's feature vector.

4 Experimental Results and Evaluation

The proposed method is implemented using PyTorch and the experiments are carried over in a CUDA-based high-performance GPU cluster. The model is evaluated using the VeRi-776 dataset [9]. The test set consists of query images and gallery images, where the query contains 1678 images with 200 vehicle IDs and 11,579 gallery images with 200 corresponding vehicle IDs. Initially, the feature extraction model is used to extract feature vectors from the query and gallery images. Later, these features are fed to a ranking algorithm that utilizes Euclidean distance to produce a ranked list of gallery images for each query. This result is further used to reconstruct the query vector after collecting the corresponding relevance information, and the reconstructed query is used to retrieve better results. For the evaluation of the proposed technique, we utilize mean average precision (MAP) and rank-based accuracy.

Table 1 showcases the observations with reference to the experimentation conducted. Each row represents a varied experimental setup, and every column represents its corresponding performance. We experiment with a total of eight variations adapted from the base model in terms of the type of relevance feedback used and the number of Top- n results considered for retrieval evaluation. Out of the eight variations, three utilize blind feedback, and the others use explicit feedback. Here, the term Top- n represents the relevance feedback information taken from the Top- n results out of the preliminary ranking results to perform query reconstruction.

Table 1 Observed results for different models and relevance feedback methods

Model/approach	MAP	Rank-1	Rank-3	Rank-5	Rank-10	Rank-20
ResNet (without feedback)	26.5	61.6	74.1	79.9	86.3	91.6
ResNet + blind feedback (Top-1)	7.4	13.0	21.6	27.8	39.6	53.6
ResNet + blind feedback (Top-3)	2.0	1.0	2.4	3.6	8.3	17.5
ResNet + blind feedback (Top-5)	1.4	1.0	1.9	2.3	5.4	13.8
ResNet + explicit feedback (Top-1)	26.5	61.7	74.1	79.9	86.4	91.8
ResNet + explicit feedback (Top-3)	28.2	65.0	76.8	81.9	87.8	92.7
ResNet + explicit feedback (Top-5)	29.4	70.4	79.0	83.4	88.7	93.0
ResNet + explicit feedback (Top-10)	31.7	79.4	83.6	85.8	89.5	93.3
ResNet + explicit feedback (Top-20)	34.2	83.9	88.6	90.0	91.8	93.8

Figure 3 illustrates the mean average precision of all the experimental models, and Fig. 4 represents the rank-based performance of all the models. From the experimental results, it can be observed that the explicit feedback technique better suits our model and helps achieve a significant increase in performance, while the

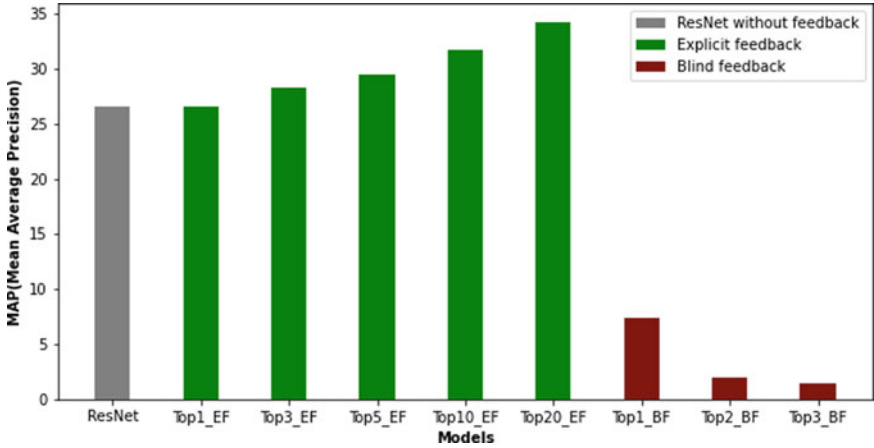


Fig. 3 Performance of various models in terms of mean average precision

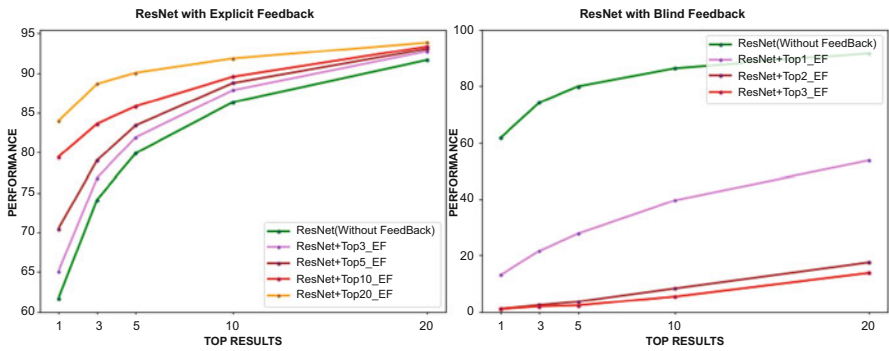


Fig. 4 Ranking performance of various models

blind feedback technique worsens the performance. A major improvement can be noticed with the rank-1 performance, and this proves the effectiveness of explicit feedback as a post-processing technique. The decrease in the model’s performance after the use of blind feedback is expected as the initial base model’s performance is not high enough. As a result of the low initial performance, the averaging of Top- n results makes the reconstructed query perform worse.

Figure 5 showcases a visual representation of the improvement in results after re-ranking to promote a better understanding of how relevance feedback affects the re-ranking process. Here, every row represents the result before re-ranking, and every column represents the top n results. Based on the experimental observation, we can conclude that the explicit feedback technique better suits models with low as well as high initial performance, while the blind feedback technique could be used on

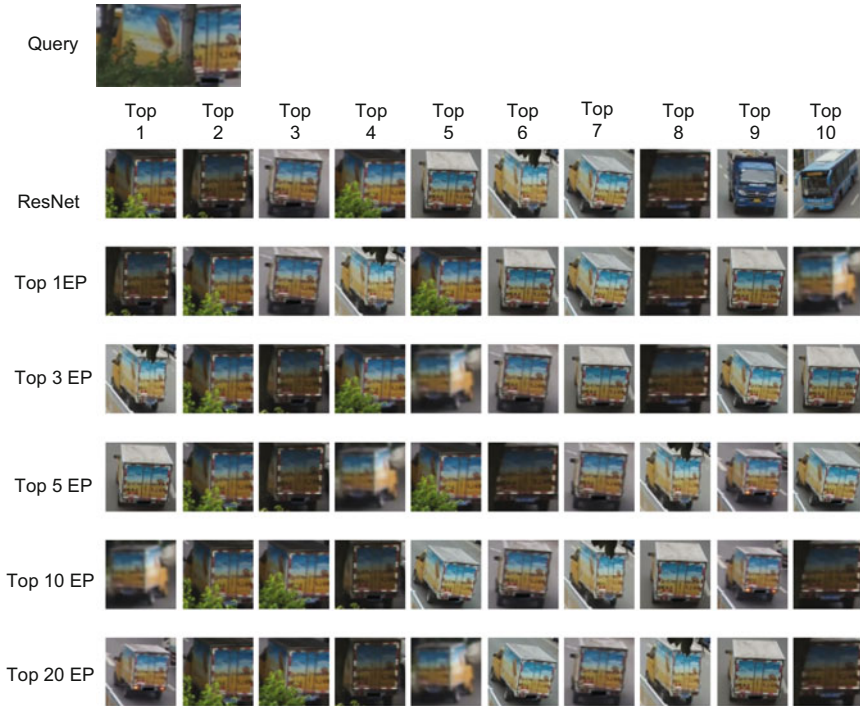


Fig. 5 Visual representation of the re-ranking process

models with better initial performance. The advantage of blind feedback is that it does not require user involvement, thus making the process continuous and faster.

5 Conclusion and Future work

In this chapter, a real-time approach to vehicle re-identification was presented, which utilizes a lightweight feature extraction model trained with a combination of triplet loss and cross-entropy loss. Furthermore, the use of relevance feedback as a post-processing technique was introduced. Two types of feedback techniques were experimented with, namely, blind feedback and explicit feedback. After re-ranking based on explicit feedback, a significant improvement was observed in the model performance in terms of both mAP and rank. We are able to observe a 7.7% increase in terms of mAP and 22.3% increase in terms of rank while considering Top-20 ranked images for relevance feedback. We also demonstrated the ineffectiveness of the blind feedback technique for models with low initial performance. As part of future work, we intend to further improve the proposed model by adapting better lightweight feature extraction models and the use of synthetic training data to increase the generalizability of the model.

References

1. He K, Zhang X, Ren S, Sun J: Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition 2016 (pp. 770–778).
2. Padmakumar, V, et al. “A robust approach to open vocabulary image retrieval with deep convolutional neural networks and transfer learning.” 2018 Pacific Neighborhood Consortium Annual Conf. and Joint Meetings (PNC). IEEE, 2018.
3. Karthik, K., and S. Sowmya Kamath. “A deep neural network model for content-based medical image retrieval with multi-view classification.” *The Visual Computer* 37, no. 7 (2021): 1837–1850.
4. Kumar, Niteesh, et al. “Sketch-Based Image Retrieval Using Convolutional Neural Networks Based on Feature Adaptation and Relevance Feedback.” *Intl. Conf. on Emerging Applications of Information Technology*. Springer, Singapore, 2021.
5. Xinchun Liu, Wu Liu, Tao Mei, Huadong Ma: PROVID: Progressive and Multimodal Vehicle Reidentification for Large-Scale Urban Surveillance. *IEEE Trans. Multimedia* 20(3): 645–658 (2018)
6. Wang Z, Tang L, Liu X, Yao Z, Yi S, Shao J, Yan J, Wang S, Li H, Wang X. Orientation invariant feature embedding and spatial temporal regularization for vehicle re-identification. In Proceedings of the IEEE Intl. Conf. on Computer Vision 2017 (pp. 379–387).
7. Liu X, Zhang S, Huang Q, Gao W. Ram: a region-aware deep model for vehicle re-identification. In 2018 IEEE Intl. Conf. on Multimedia and Expo (ICME) 2018 Jul 23 (pp. 1–6). IEEE.
8. Xinchun Liu, Wu Liu, Huadong Ma, Huiyuan Fu: Large-scale vehicle re-identification in urban surveillance videos. *ICME* 2016: 1–6
9. Xinchun Liu, Wu Liu, Tao Mei, Huadong Ma: A Deep Learning-Based Approach to Progressive Vehicle Re-identification for Urban Surveillance. *ECCV* (2) 2016
10. Zhao Y, Shen C, Wang H, Chen S. Structural analysis of attributes for vehicle re-identification and retrieval. *IEEE Transactions on Intelligent Transportation Systems*. 2019 Feb 18;21(2):723–34.
11. Chen Y, Zhu X, Gong S. Person re-identification by deep learning multi-scale representations. In Proceedings of the IEEE Intl. Conf. on computer vision workshops 2017 (pp. 2590–2600).
12. Chen Y, Jing L, Vahdani E, Zhang L, He M, Tian Y. Multi-camera Vehicle Tracking and Re-identification on AI City Challenge 2019. In *CVPR Workshops* 2019.
13. Zhu X, Luo Z, Fu P, Ji X. VOC-ReID: Vehicle re-identification based on vehicle-orientation-camera. In Proceedings of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops 2020 (pp. 602–603).
14. Huang TW, Cai J, Yang H, Hsu HM, Hwang JN. Multi-View Vehicle Re-Identification using Temporal Attention Model and Metadata Re-ranking. In *CVPR Workshops* 2019 Jun 16 (Vol. 2).
15. Jiang M, Zhang X, Yu Y, Bai Z, Zheng Z, Wang Z, Wang J. Robust vehicle re-identification via rigid structure prior. In Proceedings of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition 2021 (pp. 4026–4033).

Extractive and Abstractive Text Summarization Model Fine-Tuned Based on BERTSUM and Bio-BERT on COVID-19 Open Research Articles



Jhansi Lakshmi Durga Nunna, V. K. Hanuman Turaga,
and Srilatha Chebrolu

Keywords Natural Language Processing · Text summarization · Deep learning · Transfer learning · BERT · BERTSUM · Bio-BERT · COVID-19 · CORD-19

1 Introduction

Text summarization is a Natural Language Processing (NLP) text generation application. Text summarization is the procedure of producing a precis of the given textual content document. The generated text summary should possess the following properties: (i) readability, (ii) understandability, (iii) correctness, (iv) completeness, (v) non-redundant, (vi) conciseness, and (vii) coverage. The problem of summarizing text is categorized as abstractive and extractive. Abstractive Text Summarization (ATS) [23] is also known as paraphrasing and is the process of phrasing summary preserving the ideas and cognitive semantics of the original text document, whereas Extractive Text Summarization (ETS) [17] is also known as selection and combination and is the process of selecting and combining a few significant sentences from the text document. Recall-Oriented Understudy for Gisting Evaluation (ROUGE) metric [11] is used to assess the exception of the obtained textual content summary.

The task of obtaining ETS and ATS can be viewed as translating an input sequence of tokens or words or sentences represented as $X = (x_1, x_2, x_3, \dots, x_n)$ into another sequence of tokens or words or sentences represented as $Y = (y_1, y_2, y_3, \dots, y_m)$ where $m < n$ [22]. In the literature, many procedures had been proposed for textual content summarization. Seq2Seq for textual content

J. L. D. Nunna (✉) · V. K. H. Turaga · S. Chebrolu
Department of Computer Science and Engineering, National Institute of Technology Andhra Pradesh, Tadepalligudem, India
e-mail: mtcs2003@student.nitandhra.ac.in; vkhanumant.sclr@nitandhra.ac.in;
srilatha.chebrolu@nitandhra.ac.in

summarization had been mentioned within the literature [18, 24]. Encoder–decoder structure is broadly used to layout Seq2Seq models. In encoder, the input text is encoded into a hidden representation. This in turn is decoded and a text summary is generated. COPYNet [5] copy mechanism is embedded in the decoder RNN, and it selects and combines input subsequences. Pointer Softmax [6] is another Seq2Seq model. Here, encoder is a bidirectional RNN generating annotation vector. The decoder is a Gated Recurrent Unit (GRU) using a soft-alignment mechanism. Read Again [30] is a Seq2Seq model. In this approach, two variants of RNN, i.e., GRU and LSTM, are used. It makes use of a vocabulary dictionary for spotting small vocabularies and perceiving out-of-vocabulary (OOV) words. Convolutional neural network (CNN) [32] is also used as the Seq2Seq model for the problem of text summarization. Here, a hierarchical attention mechanism is used to initiate key words and key phrases. Kaichun Yao et al. [29] have introduced a dual encoding model to achieve text summarization. Here, an abstract summary is obtained. This model is built using bidirectional GRU-based RNN encoders. Haoran Li et al. [10] have proposed an ETS method for multi-modal data such as text, image, audio, and video. The authors have developed a multi-modal summarization corpus in both English and Chinese. Alexios Gidiotis et al. [4] have introduced a divide and conquer technique to obtain textual context summary of lengthy documents. This approach uses RNN and transformers. HH-ATS [28], in the first stage the knowledge-based hierarchical attention module, mimics rough reading. In the second stage, the multitasking learning module mimics active reading. And finally the Dual Discriminator Generative Adversarial Network (DD-GAN) mimics post-editing. These deep Seq2Seq models suffer from long-range dependencies. Other learning models such as reinforcement learning, transfer learning, transformer architecture, [26] and pre-trained language models for text summarization had been broadly mentioned within the literature. Bidirectional Encoder Representations from Transformers (BERT) [2], BART [9], PEGASUS [31], and Open AI’s Generative Pre-Trained Model (GPT) [20] are few among the pre-trained NLP models. Transformer architecture [26] addressed the issue of long dependencies. As transformers support parallel computation, utilization of GPU and TPU made the computation faster. The quality of the obtained text summary has also improved. Fine-tuning of BERT for ETS (BERTSUM) [12] is achieved by passing BERT outputs through a sequence of summary layers. T-BERTSum [14] is based on a neural topic model, BERT, and LSTM network. Ming-Hsiang Su et al. [25] have proposed an approach for variable-length ATS. The proposed approach is based on BERT, bidirectional LSTM, and BERTSUM. NEUSUM [33] for ETS was introduced. Here, the BERT model is used as the document encoder. Sentence extractor is achieved using two-layer RNN and layer normalization. CCTSenEmb [3] based on latent discriminative Gaussian topics and sentence embedding for document summarization was introduced. FLORAL [1] identifies the multimodality for the text summarization task with self-attentions. KTOPAS [8] approach for textual content summarization. This approach is based on Topic Knowledge Base (TKB). TKB obtains the semantic knowledge of the text data. The topic knowledge is given to convolutional sequence network models to obtain text summarization.

In [15], a summarization method based on contextualized embeddings obtained through the BERT was proposed for bio-medical text summarization. The proposed approach uses bio-medical knowledge bases. Akanksha Joshi et al. [7] have proposed the SummCoder method for obtaining ETS. SummCoder is a deep auto-encoder architecture. Here, the authors have introduced Tor Illegal Documents Summarization (TIDSumm) dataset. Balaanand Muthu et al. [16] have proposed Deep Learning Modifier Neural Network (DLMNN) supervised learner for text summarization.

Limitations of the related work include (i) inability to capture long distance patterns, (ii) lack of models on structured conceptual information, (iii) inability to process long documents with multiple topics, and (iv) generated summary does not match the facts of the source text document. In this chapter, *CORD-19* dataset is analyzed for the task of ETS and ATS. A hybrid NLP model has been proposed based on (i) BERT models pre-trained on bio-medical data and (ii) BERT models fine-tuned for the task of obtaining text summary. The proposed model is intended to work on bio-medical data for the task of generating text summary. The rest of the chapter is organized as follows. In Sect. 2, the following pre-trained NLP models: (i) BERT, (ii) BERTSUM for ETS, (iii) BERTSUMABS and BERTSUMEXTABS for ATS, and (iv) Bio-BERT are described. Section 3 discusses the proposed model. Section 4 deals with experimental analysis. And Sect. 5 concludes the chapter.

2 Related Work

This section describes language models based on transfer learning [19]. Transfer learning is the process of pre-training a model on unsupervised data and fine-tuning the model on supervised data for a specific downstream task.

2.1 BERT

BERT defines a unified architecture for various NLP tasks. BERT works in two stages. The first stage is known as pre-training and the second stage is known as fine-tuning. In pre-trained architecture, deep bidirectional pre-training on unlabeled text will be performed. Language model is obtained using a multi-layer bidirectional transformer encoder. Here, the input sequence is a sentence or a pair of sentences. The inputs are transformed to token embedding following WordPiece embeddings. WordPiece has 30,000 words vocabulary. Secondly, segment embedding is included to specify to which sentence this token belongs. Finally, position embedding is included to specify the position. Special tokens include classification token (CLS) and separator token (SEP). CLS represents classification task and SEP represents the separator token between two sentences. The output is the hidden vector of the input token.

Pre-trained architecture uses both Masked Language Model (MLM) and Next Sentence Prediction (NSP) for training different NLP tasks. MLM will mask a few tokens in the input sequence. The final hidden vectors of the masked tokens are given as input to the output layer and predict the masked token using softmax function. NSP is used to determine sentence relationships. BERT architecture is pre-trained to predict the succeeding sentence in the sequence using NSP. BERT is pre-trained on English Wikipedia and BooksCorpus. Fine-tuned architecture is supervised learning trained on labeled data. The fine-tuned architecture is initialized with the parameters obtained from the pre-training phase. During the fine-tuning phase, these parameters are fine-tuned according to the NLP task. BERT was initially proposed as $BERT_{base}$ and $BERT_{large}$. $BERT_{base}$ has $L = 12$, $A = 12$, and $H = 768$, and $BERT_{large}$ has $L = 24$, $A = 16$, and $H = 1024$, where L is the number of transformer encoders, A is the number of self-attention heads, and H is the number of hidden layers.

Multi-Head Attention (MHA) mechanism is used in each one of the transformer encoders of BERT. MHA is defined as follows:

$$MultiHead(Q, K, V) = Concat(head_1, \dots, head_h)W^O$$

where $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$, Q is query matrix, K is key matrix, V is value matrix, W^O , W_i^Q , W_i^K , and W_i^V are trainable parameters, and attention function is defined as

$$Attention(Q, K, V) = softmax(QK^T / \sqrt{d_k})V$$

where d_k is the dimension of the query matrix.

2.2 Extractive Text Summarization (ETS) BERTSUM

BERTSUM [12] is a variant of BERT for performing ETS. Here, input sequences and embeddings of BERT are modified to fit the problem of text summarization. Each sentence of the text document starts with a CLS token and ends with a SEP token. The embedding of CLS token represents the aggregated features of its corresponding sentence. The text document is represented as a sequence of tokens $X = w_1, w_2, \dots, w_n$. Sentence vectors are obtained from BERT. BERTSUM with inter-sentence transformer, layer normalization operation, multi-head attention operation, and position embedding function is applied on the sentence vector. Each token is assigned token, segment, and position embedding (PE).

$$PE_{(pos, 2i)} = \sin(pos/10000^{(2i/d_{model})}), \quad PE_{(pos, 2i+1)} = \cos(pos/10000^{(2i/d_{model})}),$$

where pos is the position, i is the dimension, and d_{model} is the dimensionality of input and output. These sentence vectors are given as input to the summarization layers.

$$\tilde{h}^l = LN(h^{l-1} + MultiHead(h^{l-1}))h^l = LN(\tilde{h}^l + FFN(\tilde{h}^l))$$

where $h^0 = PE(T)$, T denotes the sentence vector output by BERTSUM, LN is the layer normalization which overcomes the problem of covariance shift, l indicates the depth of the stack layer, and FFN is the Feed-Forward Neural network and is defined as $FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2$, where W_1 and W_2 are trainable parameters. The output from the summarization layers will indicate whether a sentence has to be part of the final text summary or not. BERTSUM uses simple classifiers, inter-sentence transformers, and RNN as classifiers. In BERTSUM with a simple classifier, the output is the sigmoid function of the linear combination of the sentence vectors. In BERTSUM with RNN, the LSTM layer is stacked on the BERT output layer, and layer normalization operations are also applied. BERTSUM has achieved state-of-the-art results on CNN/Daily Mail dataset.

2.3 BERTSUMABS and BERTSUMEXTABS for ATS

BERTSUMABS [13] is a variant of BERT for performing ATS. BERTSUMEXTABS [13] is another variant of BERT for performing ATS after performing ETS. BERTSUMABS and BERTSUMEXTABS are based on encoder–decoder frameworks. Here, the pre-trained BERTSUM is taken as the encoder and the 6-layered transformer is taken as the decoder. As the encoder is pre-trained and the decoder has to be trained, there will be mismatch, and as a result fine-tuning will become unstable. Two Adam optimizers are used separately for encoder and decoder. Optimizers set up unique warmup steps and learning rates. Fine-tuning of the encoder is carried out with a lower learning rate and uniform decay, such that the decoder will get stabilized. In BERTSUMEXTABS, fine-tuning is performed two stages. In the first stage, encoder is fine-tuned to achieve ETS. And in the second stage, encoder is fine-tuned to achieve ATS. Using ETS is found to improve ATS as both of them share information. BERTSUMABS and BERTSUMEXTABS are evaluated on XSUM, CNN/Daily Mail, and NYT datasets and found to achieve state-of-the-art performance.

2.4 Bio-BERT

Bio-BERT (BERT for Bio-medical TextMining) [21] is pre-trained on large bio-medical corpora to become effective in bio-medical NLP tasks. It is a variation of the BERT model developed by Korea University and Clova AI. The authors have

initialized Bio-BERT with weights of BERT, which was pre-trained on Wikipedia and BookCorpus, after that the authors have added PubMed and PubMed Central (PMC) full-text articles corpora to the original BERT. PubMed is a database of bio-medical citations and abstractions, whereas PMC is an electronic archive of full-text journal articles. Bio-BERT is also pre-trained using a Masked Language Model like BERT. In order to show the effectiveness of Bio-BERT in the bio-medical domain, it is fine-tuned on NLP tasks such as Named Entity Recognition, Relation Extraction, and Question Answering using bio-medical datasets. In this chapter, Bio-BERT is used to perform the text summarization since COVID-19 research articles contain medical terms. Using Bio-BERT will enhance the performance of any NLP tasks on bio-medical corpora.

3 Proposed Model for Text Summarization on COVID-19 Data

In the proposed model for text summarization, article-level encoder-based Bio-BERT is used to perform ETS and ATS. It will be able to encode the COVID-19 research articles and get the representations for the sentences. The ETS model was constructed on top of this encoder by piling up many inter-sentence transformer layers to catch article-level elements to extract the sentences. The abstractive model acquires an encoder–decoder framework by joining the pre-trained Bio-BERT encoder with introduced discretionary upsides of the transformer decoder. The training process that we utilized [13] will isolate the optimizers of the encoder and decoder. The proposed model also utilizes the two-phase approach where the encoder is fine-tuned twice, first with an ETS next with the ATS.

As shown in Fig. 1, hidden vector representations that we get from the pre-trained BERT will be given to the Bio-BERT. To get ETS, fine-tuning will be performed to Bio-BERT on COVID-19 data using BERTSUMEXT. For ATS, fine-tune the Bio-BERT on COVID-19 data using BERTSUMABS, and to enhance its performance, we used BERTSUMEXTABS. ETS tasks can be treated as a classification problem whether a sentence has a place within the summary. A few Inter-sentence Transformer Layers are piled up on top of BERT to catch extractive summary and are fine-tuned with BERTSUM. The proposed model utilizes Adam optimizer and Binary Cross Entropy to discover the loss for actual versus prediction. For the ATS task, we used encoder–decoder bodywork. The pre-trained encoder is BERTSUM and decoder is a six-layered transformer that is initialized arbitrarily. Here, a situation is raised where the encoder is pre-trained, while the decoder should be trained without having any preparation from scratch, and hence there is a befuddle between both. In order to overcome this issue, fine-tuning technique is used to separate the optimizers of BERTSUM and transformers. Here also Adam optimizers are used. To enhance the performance of ATS, a two-phase fine-tuning approach is utilized, i.e., encoder fine-tuned on both ETS and ATS.

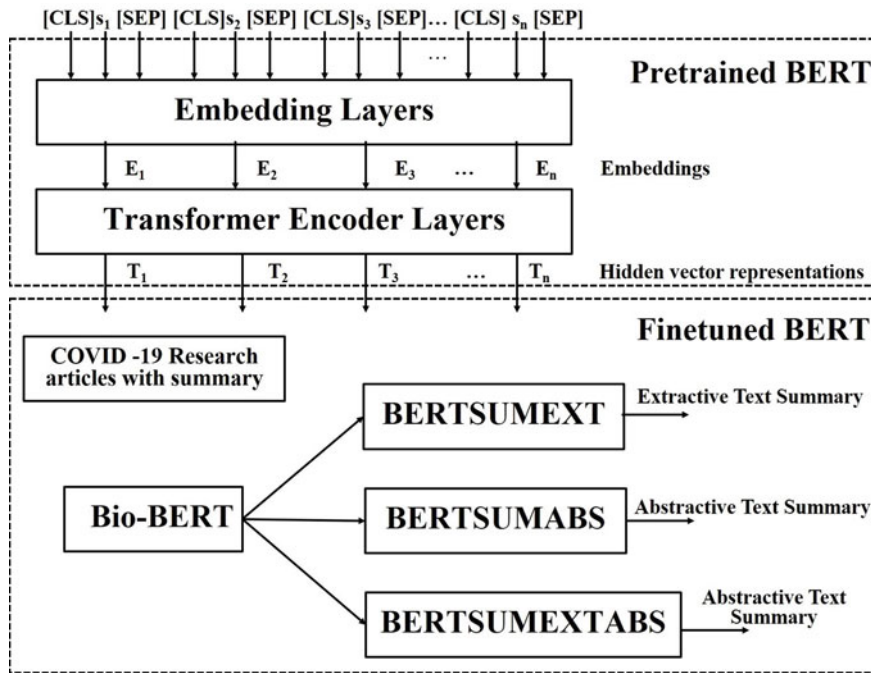


Fig. 1 Text summarization using Bio-BERT and BERTSUM for COVID-19 research articles

4 Experimental Analysis

This section describes the following: (i) CORD-19 dataset, (ii) ROUGE metric, and (iii) result analysis. COVID-19 Open Research Dataset (CORD-19) [27] is an openly available dataset. CORD-19 was developed by White House and a coalition of leading research groups, and they provided this dataset to the global research community to apply various techniques in NLP to give new insights in support of the fight against COVID-19. CORD-19 contains 247,231 research paper articles which are in json format having attribute names (keys in dictionary) paperid, title, author, abstract, body text, bib entries, and reference entries. Out of these 247,231 articles, 75,166 articles do not have an abstract. Finally, 172,065 COVID-19 research articles are available for training, testing, and validation. The following pre-processing steps have been applied on these research articles: (i) converted from json format to story format, (ii) sentence splitting and tokenization, (iii) formatted to simpler json format, and (iv) formatted to PyTorch files format. To represent each individual sentence, we use an external token [CLS] at the starting of each sentence which collects the features of the preceding sentence. To distinguish multiple sentences within the article, we used interval segment embeddings.

ROUGE [11] is a set of metrics used for automatic evaluation of text summarization. The metrics compare machine generated summary with human written. Machine generated summary is also called candidate summary. Human written summary is also called reference summary. Comparison is done by counting the number of overlapping one-gram, bi-gram, and n-gram words against the words of a reference summary. ROUGE-1 is used to measure the uni-gram overlaps between reference summary and candidate summary. ROUGE-2 measures bi-gram overlaps. ROUGE-L is computed using the longest common subsequence between reference summary and candidate summary. ROUGE-L considers each sentence as a sequence of words. Two summaries are similar if they have a common sequence of words. For the evaluation of generated summaries, Precision (P), Recall (R), and F-measure (F) for ROUGE-1, ROUGE-2, and ROUGE-L are computed as follows. The recall is used to know how many words of candidate summary are extracted from the reference summary.

$$R = (\text{Number of overlapping words}) / (\text{Total words in reference summary})$$

The precision is used to know how many candidate summary words are relevant.

$$P = (\text{Number of overlapping words}) / (\text{Total words in candidate summary})$$

F-measure will provide the complete information that recall and precision provide separately. $F - \text{measure} = ((1 + \beta^2)R * P) / (R + \beta^2 * P)$, where $\beta = 1$ for F_1 -measure.

Experiments were conducted, and the proposed hybrid model for text summarization was trained and evaluated on CORD-19 dataset. PyTorch and the BERT-base-uncased variant of Bio-BERT to implement the summarization tasks are used. Source and target data texts were tokenized with Bio-BERT word tokenizer. Trained the extractive model on GPUs provided by Google-colab for 2000 steps. Validation set is used to evaluate the model checkpoints for 1000 steps. The average results on the test set were reported using the loss on validation set. Predicted the summaries for new research articles by ranking the sentences based on the scores obtained from the proposed model. For the linear layers in the network architecture, a dropout mechanism to get abstractive summarization is used. For feed-forward layers and decoder, the hidden units are of size 768 and 2,048. With the accumulated gradient, the model was trained. For every 2500 steps, the model checkpoints were evaluated. The obtained ROUGE values are as specified in Table 1 and Fig. 2 shows the graphical representation of comparative results. Here, Bio-BERT- and BERT-based ROUGE-1, ROUGE-2, and ROUGE-L values are reported for ETS and ATS. For each ROUGE value, precision (P), recall (R), and F-measure (F) are computed. ROUGE values indicated in bold represent the highest values achieved under a category, and ROUGE values colored in blue represent the highest values achieved under all categories. The proposed Bio-BERT-based model has achieved the highest ROUGE-1 and ROUGE-L values for the task of ETS. The BERT-based model achieved highest ROUGE-2 values for the task of

ETS. For the CORD-19 dataset, extractive summarization has achieved the highest values in comparison with abstractive summarization and extractive–abstractive summarization.

Table 1 ROUGE values for CORD-19 dataset using Bio-BERT and BERT

Bio-BERT		Extractive		Abstractive		Extractive–abstractive	
		Bio-BERT	BERT	Bio-BERT	BERT	Bio-BERT	BERT
ROUGE-1	R	19.59	19.22	13.64	15.78	12.66	15.71
	P	43.61	43.29	41.72	39.27	39.05	40.54
	F	24.06	23.64	18.17	20.30	17.10	20.41
ROUGE-2	R	6.26	6.34	2.85	3.54	2.77	3.49
	P	13.53	13.88	8.97	9.39	9.07	9.47
	F	7.27	7.42	3.75	4.59	3.77	4.59
ROUGE-L	R	14.19	14.00	9.42	10.67	8.88	10.58
	P	31.45	31.44	29.47	26.15	27.64	27.05
	F	17.18	16.98	12.41	13.48	11.89	13.52

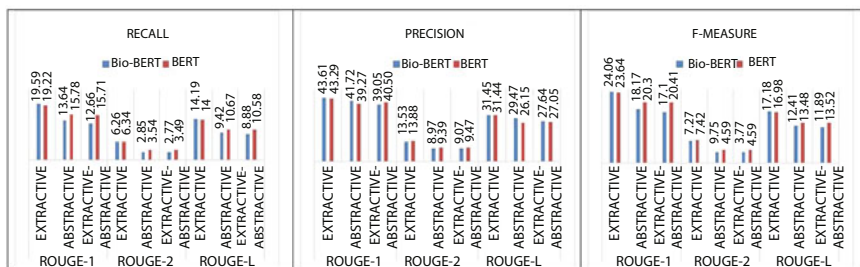


Fig. 2 Comparison of Bio-BERT and BERT ROUGE values for ETS and ATS

5 Conclusion

This chapter analyzes the NLP task of text summarization in the context of bio-medical data. The proposed bio-BERT-based BERTSUM, BERTSUMABS, and BERTSUMEXTABS were evaluated on CORD-19 dataset and are found to achieve better ROUGE-1 and ROUGE-L values for extractive summarization in comparison to BERT-based BERTSUM, BERTSUMABS, and BERTSUMEXTABS. For the CORD-19 dataset, extractive summarization has achieved the highest values in comparison with abstractive summarization and extractive–abstractive summarization. In the future work, the proposed model has to be evaluated on various

bio-medical textual data related to various diseases such as interstitial lung disease, Crohn's disease, congenital heart disease, Alzheimer's disease, and polycystic kidney disease. This further needs the development of the corpora.

References

1. Atri, Y.K., Pramanick, S., Goyal, V., Chakraborty, T.: See, hear, read: Leveraging multimodality with guided attention for abstractive text summarization. *Knowledge-Based Systems* **227**, 107152 (2021)
2. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: *NAACL-HLT, ACL, Volume 1*, pp. 4171–4186 (2019)
3. Gao, Y., Xu, Y., Huang, H., Liu, Q., Wei, L., Liu, L.: Jointly learning topics in sentence embedding for document summarization. *IEEE Transactions on Knowledge and Data Engineering* **32**(4), 688–699 (2020)
4. Gidiotis, A., Tsoumakas, G.: A divide-and-conquer approach to the summarization of long documents. *IEEE/ACM Trans. on Audio, Speech, and Lang. Processing* **28**, 3029–3040 (2020)
5. Gu, J., Lu, Z., Li, H., Li, V.O.: Incorporating copying mechanism in sequence-to-sequence learning. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1)*, pp. 1631–1640 (2016)
6. Gulcehre, C., Ahn, S., Nallapati, R., Zhou, B., Bengio, Y.: Pointing the unknown words. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1)*, pp. 140–149 (2016)
7. Joshi, A., Fidalgo, E., Alegre, E., Fernández-Robles, L.: Summcode: An unsupervised framework for extractive text summarization based on deep auto-encoders. *Expert Systems with Applications* **129**, 200–215 (2019)
8. Khanam, S.A., Liu, F., Chen, Y.P.P.: Joint knowledge-powered topic level attention for a convolutional text summarization model. *Knowledge-Based Systems* **228**, 107273 (2021)
9. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.: Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. pp. 7871–7880 (2020)
10. Li, H., Zhu, J., Ma, C., Zhang, J., Zong, C.: Read, watch, listen, and summarize: Multimodal summarization for asynchronous text, image, audio and video. *IEEE Transactions on Knowledge and Data Engineering* **31**(5), 996–1009 (2019)
11. Lin, C.Y.: ROUGE: A package for automatic evaluation of summaries. In: *Text Summarization Branches Out, ACL*, pp. 74–81 (2004)
12. Liu, Y.: Fine-tune bert for extractive summarization. *ArXiv* **abs/1903.10318** (2019)
13. Liu, Y., Lapata, M.: Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345* (2019)
14. Ma, T., Pan, Q., Rong, H., Qian, Y., Tian, Y., Al-Nabhan, N.: T-bertsum: Topic-aware text summarization based on bert. *IEEE Trans. on Computational Social Systems* pp. 1–12 (2021)
15. Moradi, M., Dorffner, G., Samwald, M.: Deep contextualized embeddings for quantifying the informative content in biomedical text summarization. *Computer Methods and Programs in Biomedicine* **184**, 105117 (2020)
16. Muthu, B., Cb, S., Kumar, P.M., Kadry, S.N., Hsu, C.H., Sanjuan, O., Crespo, R.G.: A framework for extractive text summarization based on deep learning modified NN classifier. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* **20**(3) (2021)
17. Nallapati, R., Zhai, F., Zhou, B.: Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In: *Thirty-First AAAI Conference on Artificial Intelligence* (2017)

18. Nallapati, R., Zhou, B., dos Santos, C., Gülçehre, Ç., Xiang, B.: Abstractive text summarization using sequence-to-sequence RNNs and beyond. In: Proceedings of The 20th SIGNLL CCNLL, pp. 280–290 (2016)
19. Pan, S.J., Yang, Q.: A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* **22**(10), 1345–1359 (2010)
20. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018)
21. Raffel, C., Shazeer, N.M., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv abs/1910.10683* (2020)
22. Rush, A.M., Chopra, S., Weston, J.: A neural attention model for abstractive sentence summarization. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 379–389 (2015)
23. Rush, A.M., Harvard, S., Chopra, S., Weston, J.: A neural attention model for sentence summarization. In: ACLWeb. Proceedings of the 2015 conference on empirical methods in natural language processing (2017)
24. See, A., Liu, P.J., Manning, C.D.: Get to the point: Summarization with pointer-generator networks. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Volume 1, pp. 1073–1083 (2017)
25. Su, M.H., Wu, C.H., Cheng, H.T.: A two-stage transformer-based approach for variable-length abstractive summarization. *IEEE/ACM Trans. on Audio, Speech, and Lang. Processing* **28**, 2061–2072 (2020)
26. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, u., Polosukhin, I.: Attention is all you need. In: Proceedings of the 31st International Conference, NIPS’17, pp. 6000–6010 (2017)
27. Wang, L.L., Lo, K., Chandrasekhar, Y., Reas, R., Yang, J., Eide, D., Funk, K., Kinney, R., Liu, Z., Merrill, W., et al.: Cord-19: The covid-19 open research dataset. *ArXiv* (2020)
28. Yang, M., Li, C., Shen, Y., Wu, Q., Zhao, Z., Chen, X.: Hierarchical human-like deep neural networks for abstractive text summarization. *IEEE Transactions on Neural Networks and Learning Systems* **32**(6), 2744–2757 (2021)
29. Yao, K., Zhang, L., Du, D., Luo, T., Tao, L., Wu, Y.: Dual encoding for abstractive text summarization. *IEEE Transactions on Cybernetics* **50**(3), 985–996 (2020)
30. Zeng, W., Luo, W., Fidler, S., Urtasun, R.: Efficient summarization with read-again and copy mechanism. *CoRR abs/1611.03382* (2016)
31. Zhang, J., Zhao, Y., Saleh, M., Liu, P.: Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In: International Conference on Machine Learning, pp. 11328–11339. PMLR (2020)
32. Zhang, Y., Li, D., Wang, Y., Fang, Y., Xiao, W.: Abstract text summarization with a convolutional seq2seq model. *Applied Sciences* **9**(8) (2019)
33. Zhou, Q., Yang, N., Wei, F., Huang, S., Zhou, M., Zhao, T.: A joint sentence scoring and selection framework for neural extractive document summarization. *IEEE/ACM Trans. on Audio, Speech, and Lang. Processing* **28**, 671–681 (2020)

RevCode for NLP in Indian Languages



Ashis Samal, Akash Sambhangi, and Charan Singh

Keywords RevCode · Lossless encoding–decoding · Indian languages · Low-resource NLP · Text classification

Abbreviations

BOW Bag of Words

LR Logistic Regression

SVM Support Vector Machines

TF-IDF term frequency-inverse document frequency

1 Introduction

The main challenge for anyone who is working with Indian languages is the diversity in languages. We have around 22 major languages in India, each with a different script. This makes it difficult for anyone who wants to read or understand the text, and it becomes overwhelming for someone to start working in Indian language NLP. Similarly, since each script has its own way to represent the text, this becomes a challenge for machine (algorithms) to learn and discover unique patterns for various NLP use cases in limited availability of data.

RevCode [3] addresses this challenge with a unique, pure consonant-based encoding keeping phonetic nature of Indian languages intact. For example, words like “नमस्ते” would be encoded to “namastE” and “ನಮಸ್ಕಾರ್”, encoded to “namaskAra.” This helps in positioning the vowels with consonants uniformly

A. Samal (✉) · A. Sambhangi · C. Singh
Reverie Language Technologies, Bengaluru, Karnataka, India
e-mail: ashis.samal@reverieinc.com; akash.sambhangi@reverieinc.com;
charan.singh@reverieinc.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
R. Misra et al. (eds.), *Machine Learning and Big Data Analytics*,
Springer Proceedings in Mathematics & Statistics 401,
https://doi.org/10.1007/978-3-031-15175-0_18

225

across 11 most used Indian languages. This common standard representation of text with normalization helps algorithms to discover patterns and converge faster with less data. This form of encoding is not captured by other text standards like Unicode. More details are shared in the experimentation section.

Lossless Encoding–Decoding In general, an encoded form of a text of any script can be either lossy or lossless. Lossy form of encoding cannot be decoded back to its initial form because of the loss of data during the encoding of the text, whereas in lossless encoding we can decode the encoded text back to its initial form of text. When we use transliteration to convert any Indian script to English for readability and if we try to convert it back to the original language, then it would not always produce the original text accurately.

RevCode works as a lossless encoding form where we can convert Indian Scripts to RevCode and back to its original script without any loss of information. This is useful in scenarios where our database systems are not designed to handle Unicode or UTF-8 data; in such cases, we can store the Indian scripts in database by converting it into roman script using RevCode. In further sections, let us see the actual demonstration of how RevCode works.

2 Dataset

2.1 IITP Product Review Hindi

For the demonstration of RevCode, we are considering IITP Product Review Hindi dataset[2]. It has two columns:

- *Text*—which consists of product review
- *Label*—which consists of the label of the text with three labels: positive, negative, and neutral

This dataset has been benchmarked on deep neural network models like IndicBERT, mBERT, and XLM-R.

From the above distribution, we can see that both positive and neutral reviews are similar in number, but the negative reviews are few, and hence this is an imbalanced dataset.

2.2 IITP Movie Review Hindi

For the demonstration of RevCode, we are also considering IITP Movie Review Hindi dataset[2]. It has two columns:

- *Text*—which consists of movie review

- *Label*—which consists of the label of the text with three labels: positive, negative, and neutral

This dataset has been benchmarked on deep neural network models like IndicBERT, mBERT, and XLM-R.

From the above distribution, we can see that both negative and neutral reviews are similar in number, but the positive reviews are more, and hence the dataset is imbalanced (Figs. 1 and 2).

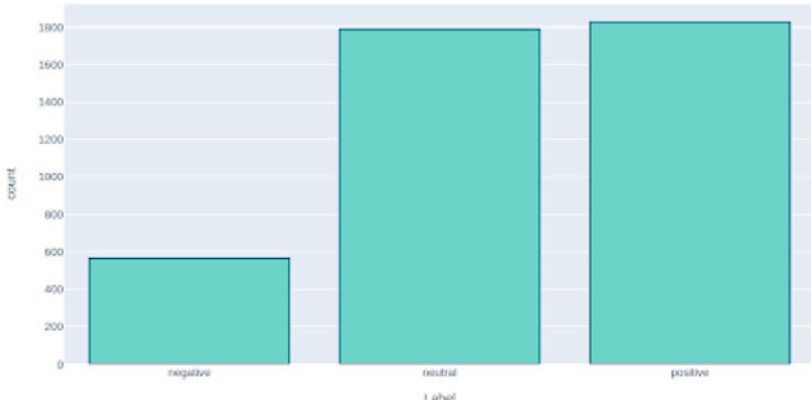


Fig. 1 Distribution of labels. This figure depicts the data distribution of IITP product review Hindi dataset [2]

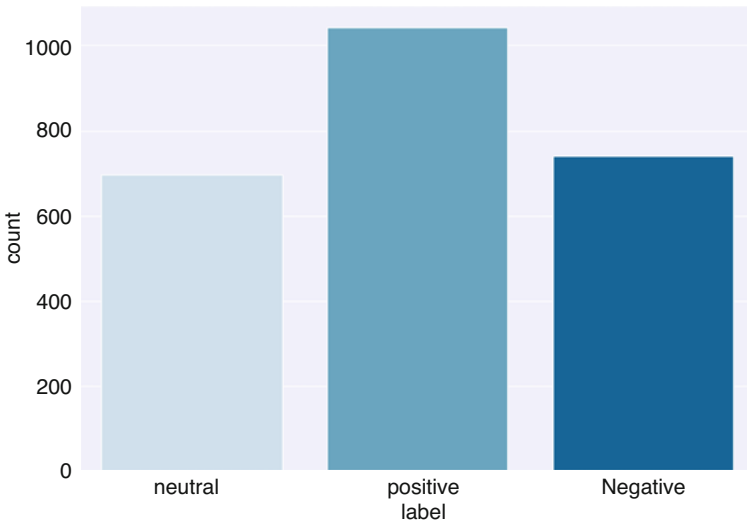


Fig. 2 Distribution of labels. This figure depicts the data distribution of IITP movie review Hindi dataset [2]

3 Experiment

We perform a detailed experimentation on the above dataset with and without using RevCode to show the impact of RevCode (Figs. 3 and 4).

The pipeline of this experimentation is:

1. **Normalization of text**—Reverie’s internal normalization process
2. **Text encoding using RevCode**
3. **Vectorizer**—Bag of Words (BOW)[10] and term frequency-inverse document frequency (TF-IDF)[11]
4. **Model Training**—Logistic Regression (LR) and Support Vector Machines (SVM)



Fig. 3 Pipeline without RevCode encoding. This figure depicts the text processing pipeline without RevCode encoding

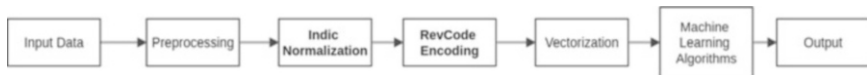


Fig. 4 Pipeline with RevCode encoding. This figure depicts the text processing pipeline with RevCode encoding

3.1 Normalization

Normalization is a process to bring uniformness into the text corpus, which helps to remove noise or inappropriate combination of characters, thus reducing the ambiguities in vocabulary. This process is crucial before we ingest the corpus for the training of algorithms.

It is the process of replacing multiple characters which actually are used to represent a single character with the actual character, examples being vowels written as combination of vowel and a matra (आ written as अ followed by ा, single matras being written as a combination of two or more matra (ेँ written as े followed by ँ), etc. This results in the reduction of number of unique tokens in the corpus, and

hence convergence of the learning becomes smooth. Normalization is done with the help of Unicode character mapping.

Example of Normalization in Devanagari script:

Without Normalization—हिन्दी

With Normalization—हिंदी

“Hindi” can be written as “हिन्दी” or “हिंदी,” both of which are correct but create inconsistency. Similarly, a person named “Nand” can prefer to type his name as “नन्द” or “नंद.”

Text Encoding Using RevCode After normalizing the text using Reverie’s internal normalization process, we encode the Hindi script to RevCode. Similar to this,

नमस्कार ->namaskAra

3.2 Vectorizer

We use BOW and TF-IDF vectorizer over advanced vectorizers like “word to vector” or “average word to vector” or an embedding layer which generates vectors is because most of the advanced techniques for vectorization models require large amounts of data to train a vectorizer which can perform well on a given language but this requires large datasets which are clean and verified, our aim is to create a pipeline with minimum resources and see if addition of RevCode alone can improve the model performance.

The same reason is also why we chose space separated words for tokenization and did not use any language-specific models like Byte pair encoding or Sentence-Piece tokenizer.

3.3 Model Training

For this experimentation, we are using Logistic Regression and Support Vector Machine (SVM) over more advanced models such as mBERT or IndicBERT][1] for text classification. The reason why we have considered machine learning algorithms instead of deep neural networks is to validate the extent of improvement that RevCode can provide to a given NLP task with minimum resources, and also using lighter models will result in lower latency.

Logistic Regression

Logistic Regression is a supervised machine learning algorithm which is used to calculate the probability of an outcome of a certain class with a given data point

as input[12]. However, Logistic Regression is used for binary classification, but it can be extended to solve multi-class classification problems. It is similar to Linear Regression, instead of predicting continuous value we predict probability of the class by applying sigmoid function (σ) to the continuous value, and it outputs the value in between 0 and 1 [8] (Figs. 5 and 6).

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

$$\text{LogisticRegression} : \hat{y} = \sigma(b_0 + b_1X)$$

Support Vector Machine

It is a supervised machine learning algorithm which is used for classification. In this algorithm, each data point is plotted in an n -dimensional space, where n is the number of features[6]. The algorithms find an optimal hyperplane which differentiates the classes significantly. This hyperplane is further used to categorize classes of data points[4]. We have to find a parameter W with given X_i as a set of input features such that

$$\text{Minimizes } \frac{1}{2} \|W\|^2, \text{ s.t. } y_i(W \cdot X_i + b) \geq 1, \forall X_i$$

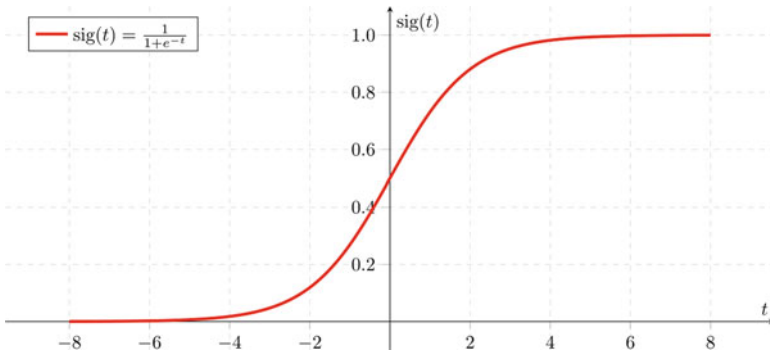
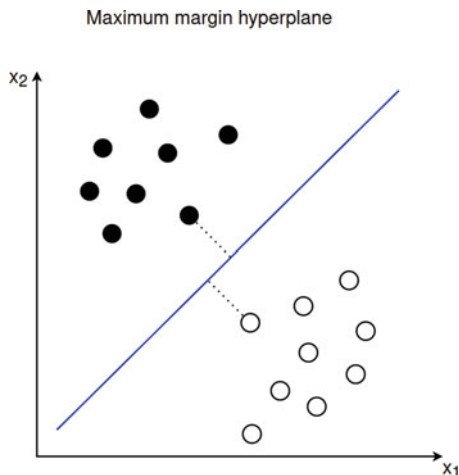


Fig. 5 Logistic regression (GeeksforGeeks, 2022) [8]

Fig. 6 Support vector machine (iNovex, 2021) [4]



3.4 Experiment Setup

The RevCode experiment will have different pipelines, all of which perform the same NLP task (in this case, it is IITP Product reviews classification in Hindi) with different component choices.

We have also performed the same experiment on a different dataset (IITP Movie Reviews) and the results are as follows:

3.5 Computation and Latency

In Fig. 7, we can see the model size and the latency of the model for single inference. Figure 7 shows the size and latency of the largest model from our pipelines. The size of the model is less than 1 MB, and the inference time is very minimal compared to heavy models with millions of parameters like BERT (Table 1).

4 Results

From the results, we can infer that the accuracy for this experiment setup is 71% on the test set (Table 2).

RevCode is also compared with mBERT[7], XLM-R[5], IndicBERT [1], and state-of-the-art algorithms in NLP[9], and the performance is similar to the traditional machine learning algorithms with the addition of RevCode normalization and encoding. A comparison is shown in Table 3.

Table 1 Results of IITP product reviews dataset. This table describes the results of different experiments

Exp. No.	Script type	Vectorizer used	ML algorithm	Normalization	RevCode encoded	Accuracy
ES-1	Devanagari	BOW	LR	No	No	67
ES-2	Devanagari	TF-IDF	LR	No	No	65
ES-3	Devanagari	BOW	SVM	No	No	66
ES-4	Devanagari	TF-IDF	SVM	No	No	63
ES-5	Devanagari	BOW	LR	No	Yes	69
ES-6	Devanagari	TF-IDF	LR	No	Yes	70
ES-7	Devanagari	BOW	SVM	No	Yes	69
ES-8	Devanagari	TF-IDF	SVM	No	Yes	68
ES-9	Devanagari	BOW	LR	Yes	No	65
ES-10	Devanagari	TF-IDF	LR	Yes	No	64
ES-11	Devanagari	BOW	SVM	Yes	No	64
ES-12	Devanagari	TF-IDF	SVM	Yes	No	63
ES-13	Devanagari	BOW	LR	Yes	Yes	69
ES-14	Devanagari	TF-IDF	LR	Yes	Yes	71
ES-15	Devanagari	BOW	SVM	Yes	Yes	69
ES-16	Devanagari	TF-IDF	SVM	Yes	Yes	68

Fig. 7 Latency

Model saved at model- 2021-08-03 13:02:38.155105
 Size of model is 879k
 Time taken for single inference 0:00:00.001916
 'Pipeline completed'

4.1 Key Observations

After experimenting it on different datasets, we have observed that:

1. RevCode + Normalization on Indian language scripts performs really well and on par with or better than some of the state-of-the-art models.
2. We have reduced the training effort by a scale of $10\times$ to $20\times$ for different text classification tasks. This includes data collection, training of tokenizer like BPE or SentencePiece, pre-training of language models, and fine tuning of models for specific tasks.

Table 2 Results of IITP movie reviews dataset. This table describes the results of different experiments [1]

Exp. No.	Script type	Vectorizer used	ML algorithm	Normalization	RevCode encoded	Accuracy
ES-1	Devanagari	BOW	LR	No	No	52
ES-2	Devanagari	TF-IDF	LR	No	No	52
ES-3	Devanagari	BOW	SVM	No	No	53
ES-4	Devanagari	TF-IDF	SVM	No	No	51
ES-5	Devanagari	BOW	LR	No	Yes	55
ES-6	Devanagari	TF-IDF	LR	No	Yes	53
ES-7	Devanagari	BOW	SVM	No	Yes	45
ES-8	Devanagari	TF-IDF	SVM	No	Yes	60
ES-9	Devanagari	BOW	LR	Yes	No	51
ES-10	Devanagari	TF-IDF	LR	Yes	No	51
ES-11	Devanagari	BOW	SVM	Yes	No	47
ES-12	Devanagari	TF-IDF	SVM	Yes	No	55
ES-13	Devanagari	BOW	LR	Yes	Yes	55
ES-14	Devanagari	TF-IDF	LR	Yes	Yes	53
ES-15	Devanagari	BOW	SVM	Yes	Yes	45
ES-16	Devanagari	TF-IDF	SVM	Yes	Yes	60

Table 3 Benchmarking with publicly available models

Dataset	mBERT	XLM-R	IndicBERT	RevCode+Norm		Without RevCode+Norm	
				SVM	LR	SVM	LR
IITP	74.57	78.97	71.32	69	71	66	65
IITM	56.77	61.61	59.03	60	55	51	52

- Another major benefit is the computation and memory footprint of models. The size of the above-trained models varies in the range of 1–2 MB, which reduces the memory by a scale of $50\times$ to $500\times$ compared to models mentioned in the above table (Table 3).
- All the abovementioned approaches will also lead to significant cost and time reduction in the text analysis project deployment.

5 Conclusion

RevCode and Normalization are very lightweight additions to a text NLP pipeline which results in improved accuracy even when paired with simple machine learning

algorithms. This proves to be a boon for all low-resource Indian language NLP tasks to improve performance on par with deep neural networks and data hungry models. We have also experimented with few other internal tasks in Telugu script and observed similar results.

6 Future Work

This RevCode-based encoding will be extended to train tokenizers and language models for different Indian languages. It can also be validated with other languages and tasks like named entity recognition, QnA.

Acknowledgments We express our gratitude to Reverie Language Technology for providing us the platform to conduct the research. We also thank all our guide and advisors (Bhupen Chauhan, Pranjal Nayak, and Vivekananda Pani) for their support throughout the experiments.

References

1. Indicbert. <https://github.com/AI4Bharat/indic-bert>.
2. Indicnlp. <https://indicnlp.ai4bharat.org/indic-glue/>.
3. Revcode. <https://github.com/reverieinc/RevCode>.
4. Support vector machine. https://www.inovex.de/wp-content/uploads/separating_hyperplanes-1024x461.png.
5. Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.
6. Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.
7. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
8. GeeksforGeeks. Logistic regression. <https://media.geeksforgeeks.org/wp-content/uploads/20190522162153/sigmoid-function-300x138.png>. [Online; accessed 23-March-2022].
9. Marie Francine Moens, Xuan-Jing Huang, Lucia Specia, and Wen-tau Yih. Findings of the association for computational linguistics: Emnlp 2021. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, 2021.
10. Wikipedia contributors. Bag-of-words model — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Bag-of-words_model&oldid=1060131791, 2021. [Online; accessed 23-March-2022].
11. Wikipedia contributors. Tf-idf — Wikipedia, the free encyclopedia. <https://en.wikipedia.org/w/index.php?title=Tf%E2%80%93idf&oldid=1071253989>, 2022. [Online; accessed 23-March-2022].
12. Raymond E Wright. Logistic regression. 1995.

Application for Mood Detection of Students Using TensorFlow and Electron JS



Marada Srinivasa Rao, Pasala Sandhya, Bosubabu Sambana ,
and Priyanka Mishra 

Keywords Tiny face detector · Face expression recognition · 68 points facial detection · TensorFlow · Electron Js · Face detection

Graphical Abstract

1 Introduction

We all depended on online platforms to learn and explore things during the pandemic. All our college/school activities are performed virtually through online applications like Zoom, Google meets, and Microsoft teams. The expanding notoriety of the Internet and progressions in the field of data innovation have, bit

M. Srinivasa Rao (✉)

Department of Information Technology, MVGR College of Engineering (A), Vizianagaram, Andhra Pradesh, India

P. Sandhya

Department of Computer Science and Engineering, Vignan's Institute of Information Technology (A), Visakhapatnam, Andhra Pradesh, India

B. Sambana

Department of Computer Science and Engineering, Lendi Institute of Engineering and Technology (A), Vizianagaram, Andhra Pradesh, India

Jawaharlal Nehru Technological University, Kakinada, Andhra Pradesh, India

Department of Computer Science and Engineering, Indian Institute of Information Technology Kota, Jaipur, Rajasthan, India

P. Mishra

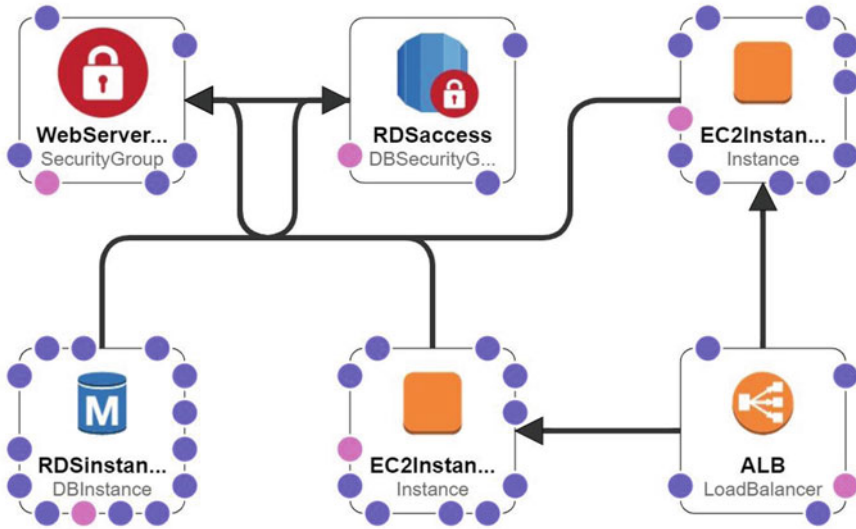
Department of Computer Science and Engineering, Indian Institute of Information Technology Kota, Jaipur, Rajasthan, India

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

R. Misra et al. (eds.), *Machine Learning and Big Data Analytics*,

Springer Proceedings in Mathematics & Statistics 401,

https://doi.org/10.1007/978-3-031-15175-0_19



by bit, expanded the utilization of online gatherings over a couple of years. It makes correspondence simpler, compelling, and proficient by utilizing web conferencing programming like Microsoft Teams and Cloud Meeting. However, many organizations still have difficulty making the most of online video conferencing, often due to various issues encountered during everyday use [1]. Lack of adequate bandwidth and network-related problems are probably the most common video conferencing issues users experience. Even though we have these many applications, we do not have any specific application for tracking the listener's mood.

Usually, these applications consume lots of data while switching on the video. The speaker has no idea whether the listener is listening or just left the place, turning off their camera. To overcome this problem, we developed an electron desktop application that runs parallelly along with other applications to track the listener's mood and helps in limiting the consumption of data [2]. To develop this application, we used the TensorFlow model, electron js, Axios for posting into the database, node express for backend, face API.js, and webcam js to access the image. So basically, it runs on the user end and makes API calls to the backend. So, let us have a clean look into the project's work.

1.1 Introduction to Electron.JS

Electron.js is a runtime structure that permits clients to use JavaScript to make work area pack applications, and the sky is the limit. Any web application you compose

can run on Electron.js. Similarly, any Node.JS application you collect can use this strategy. Electron JS uses straightforward JavaScript and other web advancements. It doesn't need local abilities except if you need to accomplish something progressed [3, 4]. It tends to be intended to work with a solitary program. Its document framework has a place with the Node.js API and is accessible for Linux, Mac OS X, and Windows.

Electron is primarily for JavaScript developers who want to build a desktop application. It utilizes npm modules which are broadly used in JavaScript. It comprises local menus for exchanges and warnings. The Windows installer doesn't need any arrangement. It additionally includes programmed updates and crash detailing utilizing Squirrel on Windows and Mac. Crash reports are submitted to a distant server for additional examination. Content following exercises, such as investigating and Chromium, deal with profiling.

The complex and monotonous piece of making a work area application is improving bundling, introducing, refreshing, offering help for local menus, warnings discoursed, and lastly, streamlining application crash announcing. Electron JS deals with practically these significant advances so that the client can zero in on the center of their application [5]. When we compose applications for internet browsers, we fundamentally contain code that will be executed on others' PCs. We do not know which programs our objective clients will utilize. It may be the most recent Chrome form or an obsolete rendition of Internet Explorer. So, we must choose the option to be moderate in the innovations we decide to execute and the sort of code we want to draft. When you construct an application with Electron, you are bundling explicit adaptations of Chromium and Node.JS, so we can depend on whatever usefulness is accessible in those variants.

1.2 Introduction to TensorFlow

TensorFlow is a free and open-source programming library for information stream and differentiable programming across various assignments. It is a representative mathematical library, likewise utilized for AI applications like profound learning and neural organizations. It is used by Google, both in research and underway. The Google Brain group created it for inward utilization, as it were [6]. On November 9, 2015, it was released under the Apache Licence 2.0.

Features

- (a) *Develops Machine Learning in the Web browser:* Involves adaptable and natural APIs in a Web program to assemble models without any preparation with the guide of the low-level JavaScript straight variable-based math library.
- (b) *Develops Machine Learning in Node.js:* Using Node.js runtime, we can execute local TensorFlow with a similar TensorFlow.js API.

1.3 Nginx Server

Nginx, articulated “engine-ex,” is an open-source web server. Since its underlying accomplishment as a web server, it is currently likewise utilized as an opposite intermediary, HTTP store, and burden balancer. Nginx is planned to give low memory use and high concurrence. Rather than making one more connection for each web interest, Nginx uses a unique, event-driven system where sales are dealt with in a singular string. With Nginx, one master association can deal with different worker processes. The controller server stays aware of the worker cycle while the experts do the veritable coping. Since Nginx is unique, workers can execute every sale simultaneously without discouraging various requests.

2 Facial Expression Recognition

Facial expressions give essential data about the feelings of an individual. Understanding facial expressions precisely are one of the problematic errands for relational connections. Programmed feeling discovery utilizing looks acknowledgment is presently a whole area of interest in different fields [7]. To distinguish the client’s disposition, regular AI calculations and profound learning strategies are utilized, and grouping exhibitions of each model are looked at.

Human face expression recognition is one of the most remarkable and testing undertakings in friendly correspondence. Feeling recognition is a significant area of work to work on the association between humans and machines. For the most part, face appearances are regular and direct means for individuals to convey their feelings and expectations. Face looks are the critical attributes of non-verbal correspondence. Feeling recognition is a significant area of work to work on the association between humans and machines. The intricacy of feeling makes the obtaining task more troublesome. Quondam works are proposed to catch feeling through unimodal systems like just looks or vocal information. All the more, as of late, the origin of the possibility of multimodal feeling acknowledgment has expanded the exact pace of the discovery of the machine [8]. Besides, profound learning procedures with neural organizations broadened the achievement proportion of devices regarding feeling acknowledgment [9]. Late works with deep learning strategies have been performed with various types of human conduct, for example, general media inputs, looks, body motions, EEG flags, and related brainwaves [10]. Still, numerous viewpoints around here to deal with to improve and make a robust framework will recognize and order feelings all the more precisely.

2.1 Detection and Characterization of Emotions Using Facial Features

The human face has particular and explicit qualities; in this way, it becomes hard to comprehend and distinguish looks. It is not difficult to determine the look of a specific individual in any picture arrangement. Assuming that we take a gander at robotized acknowledgment frameworks, in any case, the frameworks accessible are very lacking and unequipped for precisely distinguishing feelings [1]. The area of look ID has numerous significant applications. It is an intelligent device among people and PCs. The client, without utilizing the hand, can proceed with the looks. The look examination is on the variables, for example, pitiful, glad, disdain, shock, dread, and furious. This paper plans to identify faces from random pictures, remove facial elements (eyes and lips) and arrange them into six feelings (glad, dread, outrage, disdain, impartial, bitterness).

Depictions of facial muscles engaged with the feelings Darwin thought about all-inclusive. Feeling: Facial Description.

Dread: Opening of eyes, Mouth open, Lips withdrew, Eyebrows raised.

Outrage: Opening of eyes totally, Mouth packed, Nostrils expanded.

Disdain: Opening the mouth, lower lip down, Upper lip grew.

Satisfaction: Shimmering of eyes, Mouth moved back at corners, Skin under eyes crumpled.

Shock: Opening of eyes, Mouth open, Eyebrows raised, Lips jutted.

Trouble: Corner of mouth discouraged; Inner corner of eyebrows raised.

Euphoria: Upper lip raised; Nose labial overlay framed [7].

3 Proposed Model

3.1 Existed System

All our lives have been virtual due to pandemic situations; we are being managed only with web conferencing applications.

Some web conferencing applications are Zoom, GoToMeeting, Microsoft Teams, Webex, etc.

All these applications need access to our device cameras, microphones, the internet, etc.

3.2 Proposed System

Although there are many web conferencing applications, we do not have a particular application to detect the listener's mood without accessing cameras.

When using these online web conferencing applications, we need high-speed internet to access our cameras.

As everyone could not afford this high-speed internet, we came up with a solution where we could easily detect a person’s mood by minimizing the consumption of data. Figure 1 represents the flow of application launch by both teacher and student.

3.3 Application Model

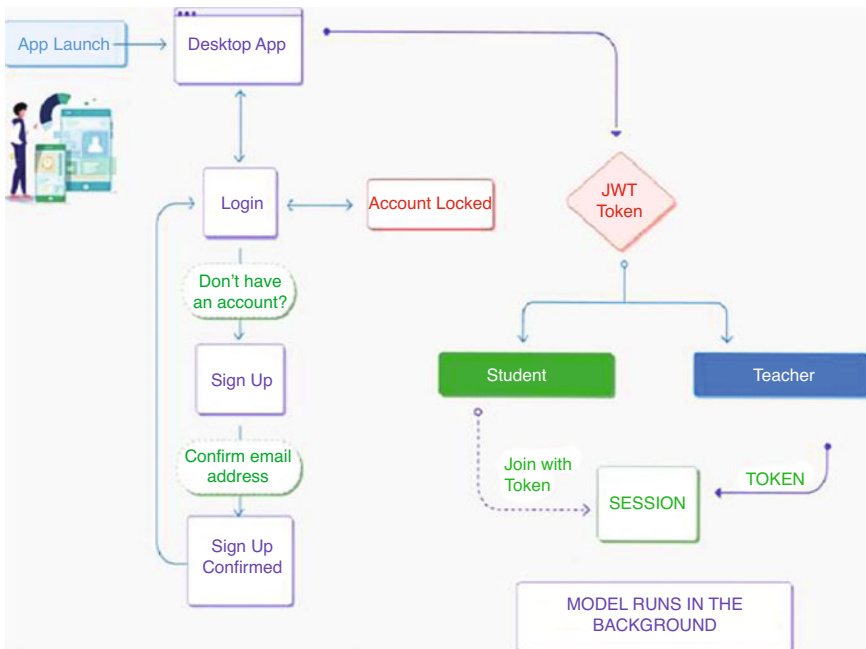


Fig. 1 Application flow model [6]

4 Algorithms

Tiny Face Detector: The Tiny Face Detector is a very performant, real-time face identifier, which is a lot quicker, more modest, and less asset-consuming, contrasted with the SSD Mobilenet V1 face locator. This model is incredibly portable and web well disposed of. Subsequently, it ought to be your GO-TO confront identifier on cell phones and asset-restricted customers. The size of the quantized model is just 190 KB [2].

Face Expression Recognition Model: The face appearance acknowledgment is quick and gives sensible precision. This model has a size of generally 310 kb, and it utilizes depth-wise separable convolutions and thickly associated blocks [3]. It has been prepared on an assortment of pictures from openly accessible datasets just as images scratched from the web.

68 Point Model: This bundle carries out an extremely lightweight, quick, and precise 68-point face milestone locator. The default model has a size of just 350 kb (face_landmark_68_model), and the miniature model is just 80 kb (face_landmark_68_tiny_model). The two models utilize the thoughts of profundity shrewd detachable convolutions just as thickly associated blocks [4]. The models have been prepared on a dataset of ~35 k face pictures named with 68 face milestone focuses. Figure 2 is an illustration of a 68 focused model.

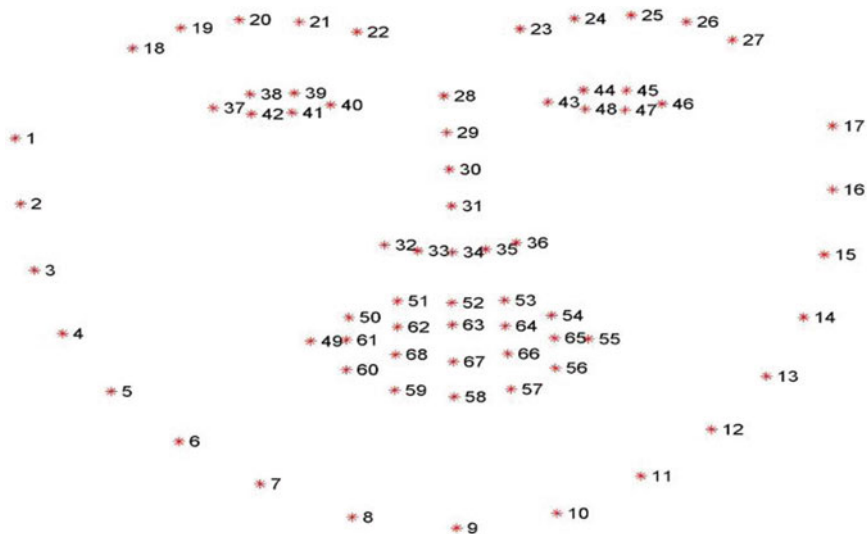


Fig. 2 68-point model [11]

There are, for the most part, two stages to recognizing face milestones in a picture which are given beneath:

- Point recognition: Face discovery is the primary technique that finds a human face and returns a worth in x,y,w,h , a square shape [5].
- Face milestone: After getting the area of a face in a picture, then, at that point, we need to toss focus on that square shape [6].

5 Results

Main Page

Once the user logs in successfully into our application, he will be redirected to the page that will track the user's mood. Here the user is a student. Figure 3 shows the spirit of students by using face detection.

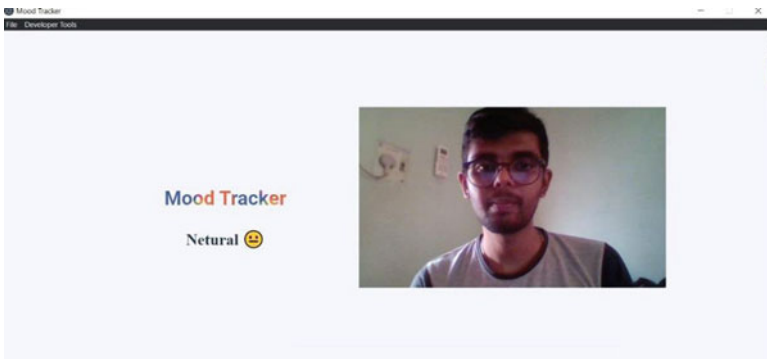


Fig. 3 Mood tracker page

Monitoring Student

On the other hand, if the user is a faculty, he needs to log in with the credentials, and after successful login, he will be redirected to the below page where he can check

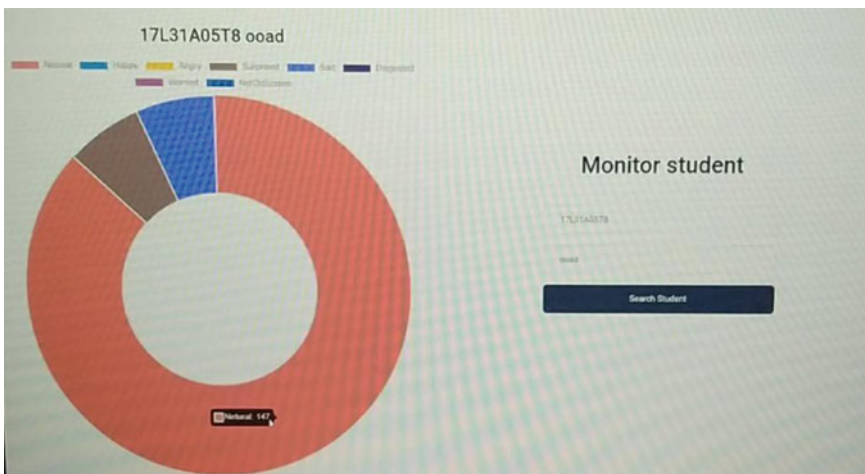


Fig. 4 Activities of a student

the listener's mood (i.e., student). Figure 4 shows the overall activities of a student in the form of a pie chart which is displayed to the faculty.

6 Conclusion

This project proposes an approach for building a real-time face emotion recognition desktop application using TensorFlow and electron js, which will run parallel with other online web conferencing applications. Through this project, we could minimize the consumption of data that will be consumed while switching on the camera in web conferencing. We could be able to represent the student's mood in the form of a pie chart by storing the facial expressions in the backend. The faculty can quickly identify the students who are not listening to the class or whether the student understands the course or not through their expressions of the student.

References

1. Charvi Jain, Kshitij Sawant, Mohammed Rehman, Rajesh Kumar. "Emotion Detection and Characterization using Facial Features." 2018 3rd International Conference and Workshops on Recent Advances and Innovations in Engineering (INCREASE).
2. Dolly renew, Neeta Tripathi. "An Efficient Method to Face and Emotion Detection." 2015 Fifth International Conference on Communication Systems and Network Technologies.
3. Ying-li Tian. "Evaluation of Face Resolution For Expression Analysis." 2004 Conference on Computer Vision and Pattern Recognition Workshop.
4. Yingli Tian., Kanade T., Cohn J.F. (2011) "Facial Expression Recognition." In: Li S., Jain A. (eds) Handbook of Face Recognition.
5. Piatkowska, Ewa. (2010) Facial Expression Recognition System.
6. Md. Forhad Ali, Mehenag Khatun, Nakib Aman Turzo. "Facial Emotion Detection Using Neural Network." 2020 International Journal Of Scientific and Engineering Research 11(8):1318-1325.
7. Turabzadeh, S.; Meng, H.; Swash, R.M.; Pleva, M.; Juhar, J. Facial Expression Emotion Detection for Real-Time Embedded Systems. *Technologies* 2018, 6, 17. <https://doi.org/10.3390/technologies6010017>.
8. M. Srinivasa Rao, Chinmaya Ranjan Pattanaik, Achyuth Sarkar, M. Ilayaraja, R. Pandi Selvam. (2020). Machine Learning Models for Heart Disease Prediction. International Journal of Advanced Science and Technology, 29(2), 4567 - 4580. Retrieved from <http://sersc.org/journals/index.php/IJAST/article/view/28464>.
9. Sekhar C., Rao M.S., Nayani A.S.K., Bhattacharyya D. (2021) Emotion Recognition Through Human Conversation Using Machine Learning Techniques. In: Bhattacharyya D., Thirupathi Rao N. (eds) Machine Intelligence and Soft Computing. Advances in Intelligent Systems and Computing, vol 1280. Springer, Singapore. https://doi.org/10.1007/978-981-15-9516-5_10.
10. Paweł Tarnowski, Marcin Kołodziej, Andrzej Majkowski, Remigiusz J. Rak, Emotion recognition using facial expressions, *Procedia Computer Science*, Volume 108, 2017, Pages 1175-1184, ISSN 1877-0509, 10.1016/j.procs.2017.05.025.
11. Amato, Giuseppe & Falchi, Fabrizio & Gennaro, Claudio & Vairo, Claudio. (2018). A Comparison of Face Verification with Facial Landmarks and Deep Features.

Using CNN Technique and Webcam to Identify Face Mask Violation



Bodduru Keerthana, Tata Rao Vana, Marada Srinivasa Rao,
Bosubabu Sambana , and Priyanka Mishra 

Keywords Transforming CNN · Features extraction · Images · Open CV · Face detection · Face recognition

1 Introduction

COVID-19 seems to be a pandemic that is quickly spreading over the globe. Even though this infection has a low number of deaths, dissemination and sickness rates are exceptionally high due to its intricate nature. COVID-19 is the hot new thing,

B. Keerthana

Department of Computer Science and Engineering, Gayatri Vidya Parishad College of Engineering (A), Visakhapatnam, Andhra Pradesh, India
e-mail: bkeerthana@gvpce.ac.in

T. R. Vana

Department of Computer Science and Engineering, Raghu Engineering College (A), Visakhapatnam, Andhra Pradesh, India

Jawaharlal Nehru Technological University, Kakinada, Andhra Pradesh, India

M. Srinivasa Rao (✉)

Department of Information Technology, MVGR College of Engineering (A), Vizianagaram, Andhra Pradesh, India

B. Sambana

Department of Computer Science and Engineering, Lendi Institute of Engineering and Technology (A), Vizianagaram, Andhra Pradesh, India

Jawaharlal Nehru Technological University, Kakinada, Andhra Pradesh, India

Department of Computer Science and Engineering, Indian Institute of Information Technology Kota, Jaipur, Rajasthan, India

P. Mishra

Department of Computer Science and Engineering, Indian Institute of Information Technology Kota, Jaipur, Rajasthan, India

either in villages or cities; people are afraid of the illness and are panicked due to the infection. If the disease is not handled early in nations like India, it will develop into a horrible situation with a high fatality rate. As more it is dealt with, the more significant for society's benefit of the entire, and especially for highly populated nations such as India. For example, see Fig. 1.

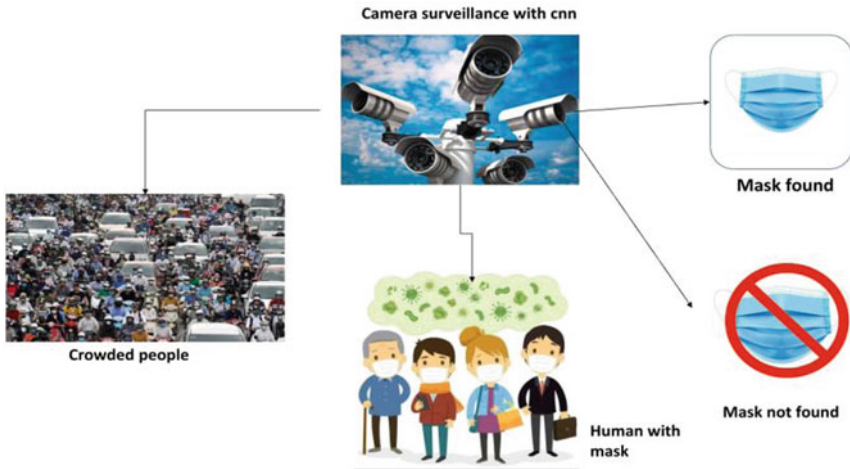


Fig. 1 People just waiting next to a heavily populated area, sporting facial masks, and maintaining public distance to escape spreading the corona viral infection

A. Motivation

An elevation in COVID-19 total illnesses has resulted in many losses. Coronavirus is growing significantly faster. As more than just a result, to accomplish efficiency and potency in detecting face masks and social distance, we have created a strategy that incorporates deep learning approaches with condition-characterized tasks. It is a safety mechanism for someone who does not use face masks or keep a social distance. This one will help minimize the virus's propagation while still protecting people's protection and reliability.

We need to handle the loss of social distance and people who do not wear masks. Manually monitoring the crowd is inefficient. To save lives, we need a better solution until everyone gets vaccinated. As a result, we created a system that uses deep learning to track social differences and identify people who are not wearing a mask or facial covering.

B. Objectives and Scope

The scope of this initiative is confined to shared spaces, like on a bridge or in a shopping complex, as is common with shut tv security footage. It is claimed that individuals only fill the observed location without cars or animals in evidence. So here is the deal: avoid crowds and large numbers of people. The rule does not apply to other types of visual media, like movies and music. We will use information recorded from stationary security cameras in real-life situations. We will continue to get better on previous studies in target tracking.

To come up with a strategy for contributing to the suppression of virus transmission by recognizing populations who seem to be:

- Establishing a healthy social space
- Implementing use of coverings in public places

As a result, we are designing a technology that uses a supervised learning technique to identify individuals who do not wear masks or build community blocks.

The output of something like the system can be seen with red and green hue frameworks as well as containers inside this framework of both the mask, such illustration, to represent whether either person wears a mask and would have been wearing a scarf, or perhaps the correctness of just that guess.

1.1 Related Work

An aspect of the project relevant to social distancing and face detection is outlined in this section. Our article can identify criminality and genuine audio notification issues [1, 2]. A cost-effective and efficient approach to AI is to provide a safe environment on the shop floor. There would be a highly complex distancing measurement device integrated. It leverages MoibleNetV2 as the built-in function for performance and employee and the template matching cascading for facial recognition. The distance measure is often used to measure is not whether people subscribe to separate channels. The image is still identifiable even while the entryway covers the participant's head [3, 4]. Can precisely classify the photos. Use either SI (MKS) or CGS as primary units. (SI units are encouraged.) English units may be used as secondary units (in parentheses). An exception would be English units as identifiers in trade, such as "3.5-inch disk drive."

Our recommended model is constructed by fine-tuning a deep learning model which has already been trained, and the following assumptions are the result used [5–8]. It is prepared and tested on a simulated masked face dataset (SMFD). To reduce the size of the dataset, image augmentation techniques have been used. The fewer data viewable, the better the model can learn and verify.

To categorize people that do not wear a face mask, using InceptionV3 well before the model has been used is proposed as a transfer learning-based technique. The model is far superior to the alternative a previously approved plan seeks toward 96

percentage-point consistency even during the pedagogical phase and 98 percentage-point correctness even during the trial phase [9].

2 Literature Review

This document uses a mixture of lightweight neural network-based ResNet50 and YOLOV3 (You Only Look Once) with a transfer learning technique to balance resource limitations and recognition accuracy in real-time video surveillance to detect people who wear face masks while maintaining a safe separation relationship [10]. Using OpenCV and ResNet50, researchers evaluate real-time streaming protocol (RTSP) video streams using neural networking models. We integrate current deep learning techniques using classic projective geometry to help fulfill real-time needs while maintaining high prediction accuracy. Violation notifications will be provided to the command center at state police headquarters if the individual is caught breaking the COVID-19 safety guidelines. It controls the solution, requires a mask, and follows social distance standards [11–14]. This model was created to run on a local computer, and the accuracy was between 80 and 90 percent.

Object detection and image processing are essential features in advanced technology development to detect images using surveillance cameras, drones, and other web-based technologies. Presented a road damage identification model in 2018. A deep learning algorithm is still used by collecting pictures with smartphones and the previous versions to identify traffic problems [15–18]. The F1 score evaluation technique provided 0.62 percentage as an outcome of this scientific work's evaluation. Using a deep learning method, the Hyeon-Cheol Shin, Kwang-II Lee, and Chang-Eun Lee proposed a method toward enlargement to detect coastal photos. The researchers used this augmentation methodology to extract more data.

This strategy was used to assist in the development of samples and the enhancement of deep neural networks. Amplification is a process of generating new retraining data from existing data [19]. Face detection is a prominent research topic nowadays. It utilizes a variety of methods proposed and Arduino-based autonomous face recognition system that can identify the streaming video and safeguard the security of both the intelligence system to estimate the facial; they deploy a Normally Servo controller Arduino Uno with Pan-Tilt motion and Open CV computer vision algorithms with the support of Haar classifying Algorithms. from live streaming data. Some common mistakes are as follows:

- It can only be used on its front, horizontal surface. Several co-face detectors with surf properties were another framework for feature detection presented by Li et al. Oro et al. offered a hear-like feature-based feature selection algorithm for high definition on the GTX470, which enhanced speed by 3.5 percent. However, they would utilize CUDA, a GPU programming toolkit for GPU acceleration.

- Unlike open CV, which is deployed in a range of technical aspects, it will be unable to solve the challenges of an imbalanced work, which was revealed while developing the viola jones face detection algorithm in graphics.
- Glass et al. (2006) discussed the importance of social differentiation and how, without antibodies or antiviral medicines, the risk of pandemic propagation can be gradually decreased by part is completed social isolation. The authors carried out a thorough study into it in both cities and suburbs to explain a decrease in the incidence of evolution. Luo Z examines how people with full-face or partial occlusion can be identified. This method divides the group into different groups: those who have their arms over their faces and who have their faces hidden by items [20, 21]. This method is inappropriate for our scenario, which necessitates detecting faces with their lips hidden by masks such as scarves or mufflers, handkerchiefs, etc.
- COVID-19 is a global problem that has infected nearly every nation. We used various methods to control and prevent the growth of this virus, including carrying a mask, frequently washing hands, utilizing sanitizers, and social distancing are all suggested.
- Researchers and scientists are working to find a technique to block the infection from spreading due to these problems. Abdellah Oumina, Noureddine El Makhfi, plus Mustapha Hamdi, three researchers who designed a methodology in 2020, created instructional transfer strategies to recognize masks. This research used machine learning approach (SVM) and K-nearest neighbor (KNN) machine algorithms to recover the characteristic, obtaining 94.1 percent accuracy employing mixed SVM and the MobileNetV2 model.

3 Methodology

(a) *Gathering and Planning Materials*

Our investigation paper utilizes the first camera, acquired online via YouTube and supplied through BriefCam and another surveillance video of the University City Centre. The first data comprises a Surveillance film of people crossing on pavements, whereas the second piece includes a video of pedestrians wandering in the University, Europe's crowded central downtown.

The University Market Town collections were used for many academic initiatives [22]. These are public databases that could be used for various science and design projects there in image recognition, image recognition, and other related areas.

This compilation is unique because it comprises video evidence from a publicly Surveillance camera set open for social protection. Several commuters in this clip are traveling or reacting in a natural, unplanned style. These ordinary citizens go out of their everyday routines on roadways or sidewalks [23–25]. Although these people were ignorant of the scientific studies, they were aware that cameras were watching

them, and this dataset was produced with their permission, thereby eliminating any ethical challenges.

(b) *Data Preparation*

The primary step in the research process would be to acquire video footage from the internet. Persons in a selected area are discovered by a camera mounted in video footage and calculate their lengths instantaneously with saving extra data. Moreover, this paper offers a novel method for identifying persons as assessing whether they should be infringing any social separation criteria. While detecting the interpersonal distances between the individuals in the video, their faces are detected utilizing facial detection to identify whether they will be wearing a mask or not. For example, see Fig. 2, which gives the entire working process of detection of face masks and social distance.

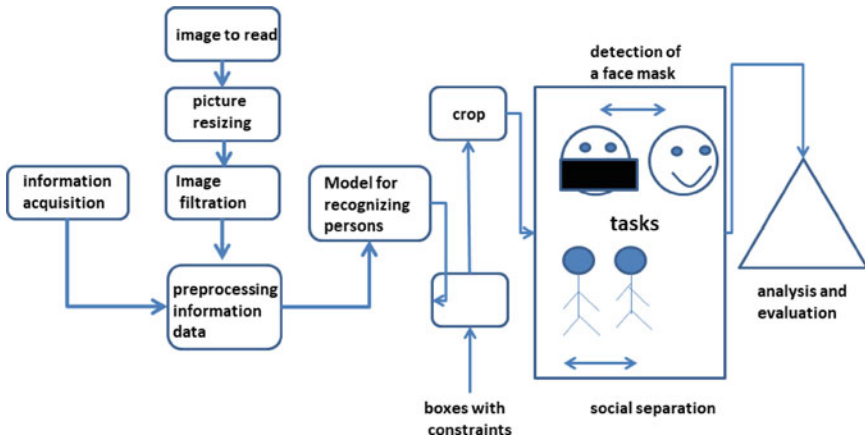


Fig. 2 (a) Flow diagram of the system detection of face mask and social distancing

Face Mask Detection

Face mask detection is split into two parts, the first one being face detection and the other of which would be face mask classifier. When someone is not wearing a mask, the surveillance screen will alert. The approach used to build the models is explained in this report.

(a) *Face Detection*

Facial recognition software face identification is made using a CNN-based deep learning model [25]. We chose this model because it has several advantages over the other models, including the capacity to detect faces at a low resolution [420×420]. Face size can also be seen using the MobileNetV2 model, with 94.2 percent accuracy. The base for face size recognition is 90×90 pixels. The output can be seen with the bounding box over the face, and then this crop is performed.

(b) *Information Gathering*

The face detection model is employed to train the face mask model. Custom datasets are a collection of actual pics of a person's face with and without the use of a face mask in our project. The dataset is composed of 3835 photo snaps that have been divided into two classes: with cover and without a body. With cover contains 1897 photos and without cover includes 1897 images. Initially, we collected approximately 5000 photo snaps, of which a small amount was rejected/deleted because of being blurred or not cleared. We can notice that data has been removed in this location. With both the help of the sklearn lib, each dataset is separated into two sections: 75 percent training data and 25 percent point test results test samples. The photos used for 896 images were used to test data.

(c) *Detection of Masks*

The face detector in OpenCV is based on the single shot multi-box detector (SSD) framework included in the MobileNetV2 architecture. We need to use the photo snap with object identification to get the bounding box (x, y) coordinates for an object, which, in this case, is a mask. Its Bolt Action Mobile Detection is a mixture of the R-CNN and YOLOv3 object detection technologies Google announced.

Detection of Social Distance

(a) *Person Detection*

The proposed model leverages the TensorFlow framework to detect people using ResNet50, a subclass of CNN models, and MobileNetV2. The ability to spot many items simultaneously is a critical element of this model. On the flip side, this model will demand more computations to get more accurate results. Especially compared to older models, GPU accelerating is enabled, allowing for speedier calculation. Eyes, nose, mouth, and other facial features from the input video.

(b) *Computing at a Distance*

For person detection in video, the model utilized in this application has a well MobileNetV2. The model will take the video frame as input and produce a list of coordinates in a rectangular bounding box for every person identified in the frame. [x-min, y-min, width, height] are the measurements of the rectangular bounding box.

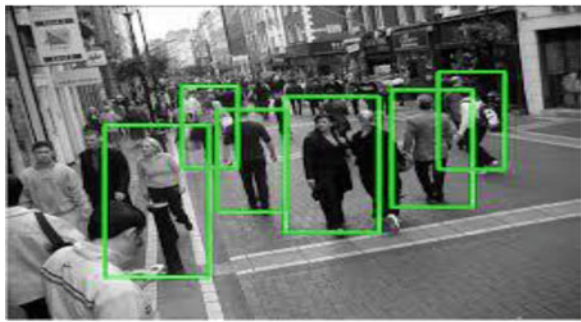
The resulting bounding box will have a centroid for every person in the video frame. The model would calculate the distance between two people by estimating the distance between two centroids. The Euclidean distance formula is used to calculate the distance between two centroids. The distance between any two real-valued vectors is calculated utilizing Euclidean distance. The person is not maintaining social distance if the computed distance is less than 3 feet. The person holds a safe space if the space is 6 feet or greater than 6 feet. For example, see Fig.3.

$$v = \sum_{i=1}^n \sum_{\substack{j=1 \\ j \neq i}}^n \|(d_{i,j})$$

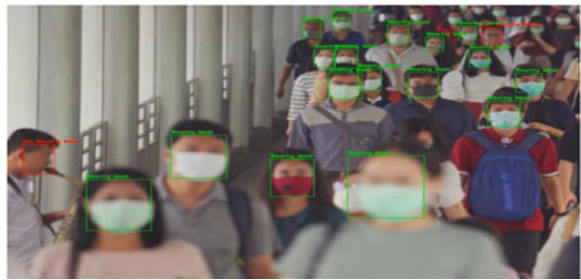
Utilizing real-world coordinates, $P = (p_1, p_2, p_n)$, the matching list of pedestrian distance D is easily found. $j = 1, 2, n = d_{min} = \min(d_i, j)$, $d_{min} = \min(d_i, j) = d_{min} = \min(d_i, j) = d_{min} = \min(d_i, j) = d_{min} = \min(d_i, j) = d_{min} = \min(d_i, j) = d_{min} = \min(d_i, j)$. We generate the two metrics for distancing dependent on the computed value of d_{min} .

$$(d_{i,j}) = \|P_i - P_j\|$$

Fig. 3 (a) First picture; (b) second picture



(a)



(b)

(c) *Limitations*

Lighting surroundings: The quality of the camera and the different lighting directly affect the detection of people. Uncontrolled background: Recognizing the subject and determining the distance in various climates.

4 Future Perspectives and Conclusion

COVID-19 is today a significant virus, and several countries are addressing this with a range of methods and techniques. A face mask is one of the protective measures used in this list. The fundamental concept driving our research project was to use the methodology of supervised learning classification CNN OpenCV is often used to recognize (try to label) whether or not such a person wears a mask. This scientific work's experimental results are divided into two phases. The outcome of the very first part of the study showed categorization as well as forecasting of an image database utilizing CNN, CNN-data strengthening, as well as MobileNet2V models, with 92 basis points exactness for CNN, 94-point profit accurateness for convolution advancement, and 98 percentage points exactness for MobileNet2V, including both.

Individuals who already have good relations or know one another and move together would be regarded as over the cultural convention distance parameter in this study. Some say that social buffering should be observed in public locations, while others argue it is essential. In conclusion, this is one of the report's weaknesses, which can be applied to future research. Its birds-eye view capability also is not utilized in this research. However, it could use in subsequent analysis. People are required to wear double face masks in some locations where virus transmission is prevalent, and it may be capable of finding whether a person is wearing a double face mask in the future. This approach was suggested with an easy way for people who do not wear a face mask but do not maintain social distance to be reported to officials by email. As a future enhancement, we will be able to accurately predict when it would become busy, allowing an appropriate heat map to be shown.

References

1. Wang Chen, Horby Peter W, Hayden Frederick G, Gao George F. A novel corona virus outbreak of global health concern. *The Lancet*. 2020;395(10223):470–473. doi:[https://doi.org/10.1016/S0140-6736\(20\)30185-9](https://doi.org/10.1016/S0140-6736(20)30185-9).
2. Matrajt L, Leung T. Evaluating the effectiveness of social distancing interventions to delay or flatten the epidemic curve of coronavirus disease. *Emerg Infect Dis*. 2020
3. <https://www.who.int/emergencies/diseases/novel-coronavirus-2019/advice-for-public>
4. <https://www.cdc.gov/coronavirus/2019-ncov/prevent-getting-sick/cloth-face-cover.html>
5. <https://www.who.int/news-room/detail/03-03-2020-shortage-of-personal-protective-equipment-endangering-health-workers-worldwide>
6. M. Young, *The Technical Writer's Handbook*. Mill Valley, CA: University Science, 1989.
7. Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao, "Yolov4: Optimal speed and accuracy of object detection", 2020, arXiv:2004.10934
8. M. Srinivasa Rao, Chinmaya Ranjan Pattanaik, Achyuth Sarkar, M. Ilayaraja, R. Pandi Selvam.(2020). Machine Learning Models for Heart Disease Prediction. *International Journal of Advanced Science and Technology*, 29(2), 4567 - 4580. Retrieved from <http://sersc.org/journals/index.php/IJAST/article/view/28464>.

9. Sekhar C., Rao M.S., Nayani A.S.K., Bhattacharyya D. (2021) Emotion Recognition Through Human Conversation Using Machine Learning Techniques. In: Bhattacharyya D., Thirupathi Rao N. (eds) Machine Intelligence and Soft Computing. Advances in Intelligent Systems and Computing, vol 1280. Springer, Singapore. https://doi.org/10.1007/978-981-15-9516-5_10.
10. S. Susanto, F. A. Putra, R. Analia and I. K. L. N. Suciningtyas, "The Face Mask Detection For Preventing the Spread of COVID-19 at Politeknik Negeri Batam," *2020 3rd International Conference on Applied Engineering (ICAE)*, 2020, pp. 1–5, DOI: <https://doi.org/10.1109/ICAE50557.2020.9350556>.
11. Joseph Redmon, Ali Farhadi, "YOLOv3: An Incremental Improvement", 2018, arXiv:1804.02767.
12. Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN," *IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPR Workshop)*, 2020.
13. Hui, "YOLOv4", Medium, 2020. [Online]. Available: <https://medium.com/@jonathanhui/yolov4-c9901eaa8e61>
14. Zhanchao Huang, Jianlin Wang, "DC-SPP-YOLO: Dense Connection and Spatial Pyramid Pooling Based YOLO for Object Detection," 2019, arXiv:1903.08589.
15. Sangdoon Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo, "CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features," 2019, arXiv:1905.0489
16. Aldila, D., Khoshnaw, S., Safitri, E., Anwar, Y., Bakry, A., Samiadji, B., Anugerah, D., GH, M., Ayulani, I. and Salim, S., 2020. A mathematical study on the spread of COVID-19 considering social distancing and rapid assessment: The case of Jakarta, Indonesia. *Chaos, Solitons & Fractals*, 139, p. 110042.
17. Christian, M., Loutfy, M., McDonald, L., Martinez, K., Ofner, M., Wong, T., Wallington, T., Gold, W., Medeski, B., Green, K., and Low, D., 2004. Possible SARS Coronavirus Transmission during Cardiopulmonary Resuscitation. *Emerging Infectious Diseases*, 10(2), pp. 287-293.
18. Deore, G., Bodhula, R., Udpikar, V. and More, V., 2016. Study of masked face detection approach in video analytics. *2016 Conference on Advances in Signal Processing (CASP)*.
19. Fong, S., Dey, N. and Chaki, J., 2020. An Introduction to COVID-19. *Artificial Intelligence for Coronavirus Outbreak*, pp. 1-22.
20. Glumov, N., Kolomiyetz, E. and Sergeev, V., 1995. Detection of objects on the image using a sliding window mode. *Optics & Laser Technology*, 27(4), 241-249.
21. Prateek Khandelwal, Anuj Khandelwal, Snigdha Agarwal, Deep Thomas, Naveen Xavier, Arun Raghuraman. Using Computer Vision to enhance Safety of Workforce in Manufacturing in a Post COVID World. arXiv 2005.05287 11/05/2020.
22. Jignesh Chowdary G., Punn N.S., Sonbhadra S.K., Agarwal S. (2020) Face Mask Detection Using Transfer Learning of InceptionV3. In: Bellatreche L., Goyal V., Fujita H., Mondal A., Reddy P.K. (eds) *Big Data Analytics. BDA 2020. Lecture Notes in Computer Science*, vol 12581. Springer, Cham. https://doi.org/10.1007/978-3-030-66665-1_6
23. Madhura Inamdar, Ninad Mehendale. Real-Time Face Mask Identification Using Facemasknet Deep Learning Network. SSRN 3663305 12/02/2020
24. M. M. Rahman, M. M. H. Manik, M. M. Islam, S. Mahmud and J.-H. Kim, "An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network," *2020 IEEE International IoT, Electronics and Mechatronics Conference (IEMTRONICS)*, 2020, pp. 1-5, DOI: <https://doi.org/10.1109/IEMTRONICS51293.2020.9216386>.
25. R. Padilla, C. F. F. Costa Filho and M. G. F. Costa. Evaluation of Haar Cascade Classifiers Designed for Face Detection. *World Academy of Science, Engineering and Technology International Journal of Computer and Information Engineering Vol:6, No:4*, 2012

A Review on Internet of Things-Based Cloud Architecture and Its Application



D. Dakshayani Himabindu, Keesara Sravanthi, Raswitha Bandi,
and Bharathi Panduri

Keywords Internet of Things · Cloud computing · IoT cloud · MQTT · Sigfox sensors

1 Introduction

Nowadays emerging technologies are playing a key role in our daily activities. Some of them are Internet of Things, blockchain, artificial intelligence, and robotics. More than 9 billion ‘Things’ (physical objects) are connected through the Internet. In the future, the number is likely to rise to a whopping 50 billion [1]. Internet of Things consists of four main components. They are low-power embedded systems, cloud computing, big data availability and connecting of networks. In IOT low-power utilization and high performance are contrary factors which play an important role in design of electronic systems. Cloud computing is used because IOT devices are huge and data collected is massive, and so to store the data, we need a storage server. Hence cloud computing is used and plays a vital role. The data is trained to discover flaws like electrical or errors found within the system. Availability of big data where IOT depends on sensors and real-time devices as the electric devices are throughout all fields, usage of these devices cause a huge change in usage of big data. Networking connection is used to communicate and Internet connection

D. D. Himabindu (✉) · K. Sravanthi · R. Bandi
Department of IT, VNRVJIET, Hyderabad, Telangana, India

B. Panduri
Department of IT, GRIET, Hyderabad, Telangana, India

is necessary where every physical object will be represented by the IP address. IP naming is given to limited devices as growth in devices increased naming procedure is not applicable further. So researchers are proceeding for alternate naming system to signify physical object individually [1, 2]. Here we are going to discuss about Internet of Things along with IOT cloud. Internet of Things makes objects very intelligent and increases their potential for active interaction. We identify the objects and perceive the data around them in IOT. These objects interact with the server and store the information, and if we want to access the data, this can be done using the Internet. In order to store the information which is necessary for these objects, we require a large storage capacity that will become complicated for organizations to provide as it will become expensive to purchase more physical machines and provide space to store them to rectify this problem. We use cloud architecture as it provides large storage capacity. In order to manage and deploy the IOT applications, we use the cloud computing techniques. Earlier before cloud IOT applications are used to run locally using the fog computing and edge computing as they are sufficient, but when large amount of storage capacity is required, then these computing techniques are not useful and cost spent will be more for storage. Hence to sort out this problem cloud computing techniques have come to existence to store IOT applications. Now, we discuss about the architecture and applications of IOT cloud which explains how the data flow happens between different layers and how in each layer data is made more and more efficient for analysis and insights. There are various IOT applications which are using cloud architecture, for example, smart appliances, smart home hub, smart locks, self-driving cars and smart security systems [3].

In IOT standardization we use different techniques like M2M, Contiki, LiteOS, RPMA and Sigfox. Machine-to-machine service layer is embedded both with hardware and software, for example, microwave oven which is hardware embedded with the software. Contiki is an open-source operating system used for the IOT microcontrollers that require low power. LiteOS is also an open-source software that works similar like Linux operating system which is used for wireless sensor networks. RPMA takes the standard ownership to connect the objects of Internet of Things. Sigfox is a low-power technology for both IOT and machine-to-machine communications. The IOT device we done various investigations on Wireless capability, function process, Interoperability, Security providence for storage, Fast boot capacity, Categorization of Devices, System Bandwidth, Control the Cryptographic, Managing the power [4]. We will discuss about the IOT device architecture which use both the network and the cloud the architecture explained in four stages. The first stage is about the different networks like the actuators we use and the wireless sensor networks. Stage two consists of the sensor data aggregation system and conversion of analog to digital data is. Third stage is the system analytics and processing using edge IT, and final stage, stage four, is where the entire data is stored in the data centre where cloud acts as the data centre and management of the data is done.

1.1 Related Work

There are some existing IOT applications which are using cloud architecture for storing the information. The IOT cloud architecture consists of IOT cloud applications, IOT integrated middleware, databases, security patches and other software algorithms, and analytic engine where the interface of secure communications is developed between the IOT device and the cloud and is done by the MQTT, COAP, AMQP, Websockets, etc. We will connect the IOT device and monitor it continuously with the help of the IOT cloud applications. These will also help the consumers with the issues they face and resolve them, and these applications are loaded with APIs and interface which will pull and push the information/data/commands to and from the applications and sensor nodes/devices. The above services are used in the IOT cloud architecture. The usage of IOT cloud has a great impact on performance of IOT applications. We have several IOT cloud applications which were discussed previously like smart buildings, smart farming, smart parking lot, intelligent transportation, smart homes and many more. In Saber Talari et al. (2017), smart cities based on Internet of Things are proposed. Smart buildings use sensors for power management to continuously monitor the power; these results will give an idea on usage of electricity and water consumption where sensors are used to detect the usage of water and automatically turn on/off of the lights, and here the sensors which we use will gather data and analyse the data and maintain the record of the data which will help us in developing the smart cities. In Chungsan Lee et al. (2016) smart parking system using IOT is proposed where ultrasonic sensors are used and connected to smartphones to find parking space availability and also to detect location of the vehicle. In Lu Hou et al. (2016) intelligent transportation is proposed where GPS is used to locate vehicle and sensors and cameras are used to know the traffic congestion of the vehicles, and the data collected and analysed is stored in IOT cloud and through this technology traffic information will also be available to pedestrians. In Prem Prakash Jayaraman et al. (2016) smart farming is proposed which used phenonet which involved a variety of crop studies that are conducted by means of IOT technologies which include sensors, cameras, smartphones and data analytics.

Abdulsalam Yassine et al. (2018) proposed about IOT smart homes that use sensors, metering services and different appliances that collect and analyse data through big data analytics using fog computing and cloud computing. Alessio Botta et al. (2015) proposed about cloud IOT applications and explained that the data and applications are different drivers which are used for the purpose of communication and the cloud can improve the IOT application's performance.

2 IOT Cloud Architecture

IOT cloud applications consist of interfaces or APIs which are used to send or receive information between the IOT sensors and consumer apps. We use IOT integrated middleware like MTQQ cluster to get data from sensors and transmit the data into cloud services to process it. The MTQQ is used for communication between the IOT devices and the end users as it is a broker protocol used for subscribing transportation of messages and the packets will be published with the help of these broker. For the purpose of publication we will uniquely define a topic and the packets will be published using broker but for subscription we use MTQQ client by using these at a time multiple topics can subscribed and for server implementation we use open source in node that means we use MTQQ connections to improve the performance of MQTT servers. The IOT cloud uses different types of NoSQL databases like MongoDB, Cassandra, CouchDB, Hypertable, Redis, Riak, Neo4j and Hadoop HBase which are used to store the data, and here the data is stored in rows and the columns same as RDBMS (Relational Database Management System). NoSQL databases are used for managing the data based on the modern demands. Rest API endpoints are the points where URL of a service is included, and each endpoint is used to access resources for carrying the functions. API runs by sending the request to server and receiving the response from server. Analytic engine consists of RDBMS, artificial intelligence algorithms, and machine learning algorithms which are used to integrate IOT-based applications or various business apps. There are different types of IOT cloud application services like sensor nodes of IOT and interface of cloud development, user IOT application developments, database management, analytics in sensor nodes of IOT and interface of cloud development which provides secured communication between IOT devices and cloud applications using MQTT, and the continuous monitoring of IOT cloud application is done using IOT device, and IOT device also alerts the users about issues and resolve them. Safe handshake mechanism is implemented for communicating or transferring the data between sensor nodes and services. User IOT applications development here user pays a key role where user access assigned IOT devices after login and end users will be given roles and responsibilities that are maintained by IOTCloud applications. IOT devices are managed and stored to cloud where it is mapped to users so that we can provide security by avoiding non-permitted access and finally end users can also control, monitor and access parameters and processes which are stored inside IOT cloud. In Database Management secure design is maintained which ensure no loss in Data, optimization is used to maintain scalability and managing device data for monitoring of devices which are deployed and the analytic phase is used for developing and integrating machine learning algorithms with IOTcloud for the growth of business. Historical data is processed when we enable IOT cloud which predicts behaviour patterns that in turn benefits the user. In Fig. 1, the process of the IOT cloud application is shown [4, 5].

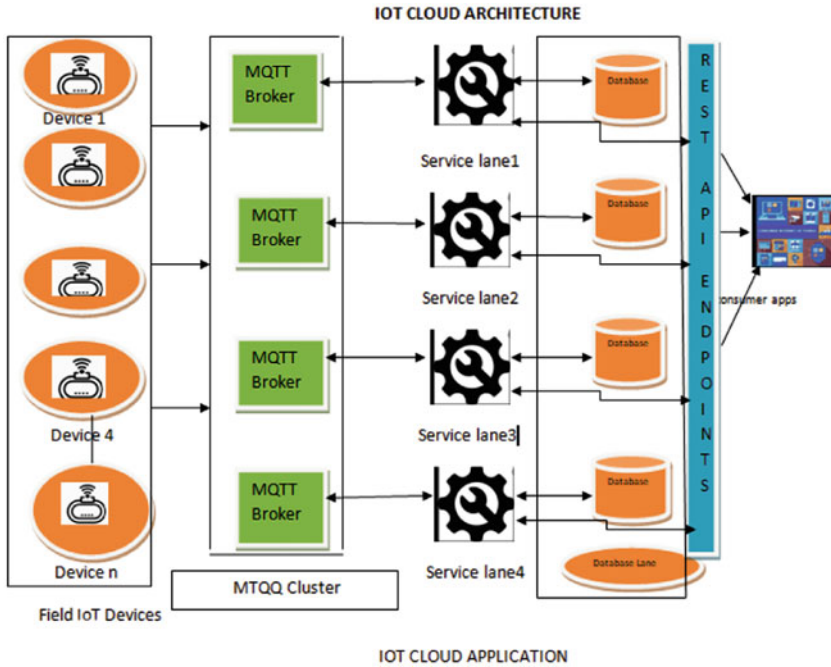


Fig. 1 IOT cloud application

3 IOT Cloud Applications

Before going to discuss about various IOT cloud applications, we will go through the services of the IOT cloud as IOT cloud should fulfil the requirements of all IOT applications using the services that are provided by IOT cloud. Different applications will be offered for the end users, managers and programmers. We can access the services using the browsers and smartphones whenever we require and from anywhere in the world. IOTCloud are classified into three categories [5]. Web applications where IOTCloud provides services by deploying WebPages using HTTP servers and these WebPages are developed with the help of Hyper text Markup Language and CSS is used to develop static pages and JavaScript is used to page where a page is taken to forward for next page [6, 7]. In IOT cloud web applications are developed for managers in order to manage interface, monitor the data of the IOT devices like timing tasks and control IOT devices with owner’s permissions for debugging. Mobile apps in smartphone play a crucial role in communication as it helps people in their daily life to access the Internet using different kinds of apps. And for the ease of users, these apps are developed in such a way that they can be accessed both in Android and iOS platforms like Facebook, etc. [8, 9]. To improve the applications and the services provided by Internet of Things,

we use a software development kits (SDKs). IOT uses two types of SDKs. One is Android and the other is Azure. We have many different types of IOT applications that use cloud architecture. We are going to discuss these applications in details and how they are using the cloud architecture and the benefits they can get by using this architecture. Here I will explain about the various applications which use IOT cloud architecture, as shown in Fig. 2.

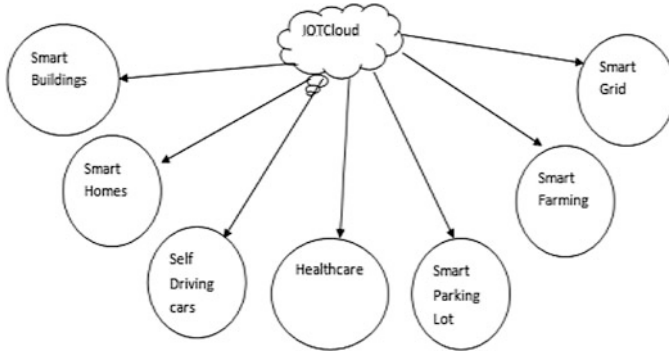


Fig. 2 Application using IOT cloud architecture

3.1 Smart Buildings

Smart buildings will adjust to both the inside and outside environmental change to provide comfort to occupants where the automated system infrastructure of a building is cheaper, simpler, autonomous and wireless. Sigfox sensors can be quickly installed all over the building to monitor the HVAC, boilers, light, power, fire and security. These sensors allow us to control and monitor all the operational parameters such as indoor air quality, temperature, occupancy, humidity and door openings. The machinery like lifts and escalators which are heavy in size and weight can also be fitted using sensors to trigger and check the breakdowns and to be proactive to do the required service to overcome the problem when a fault is detected. If there is water leakage, it can also be detected using IoT sensors. The advantage of using these sensors is that they can run on the same battery, thus eliminating the hassle of manual maintenance and control. We will get fire alerts and smoke alerts via fire safety using Internet equipment where time and manpower is wasted by doing continuous checkings. Alerts will be triggered by smoke detectors which will not be heard by people or ignored by them. These detectors will be connected to the Sigfox network which can send real-time alerts to keep status and battery level alive. By these alerts security providers will be alerted and will respond quickly, and they will also set up backup alarms and provide

protection which is powered by Sigfox IoT network. Smart security solutions are also used to hold security guards accountable and increase the reliability of alarm system. Instead of competing with traditional systems, smart devices are used to complement existing platforms and provide a cost-effective way to help increase the effectiveness of existing system services. By collecting the consumption data effortlessly we can put an end to time spent and money used on manual on-site metre readings and processing of data. Once activated, metres will transmit data immediately using the Sigfox public network, without pairing or configuring systems [10]. You can monitor infrastructure by detecting leaks and by activating and deactivating the services. Without wasting the time we can call refuse collection service and automatically sent the data to cloud to trigger refuse collection request [11]. A simple temperature sensor and fire alarm can be added and triggered as appropriate. This offers convenience and safety and cuts down on collection requests. Refer to Table 1 for these information [12].

3.2 Smart Parking Lot

Parking is considered as the major problem in cities where population is high compared to towns and villages. So we should provide extra services to solve this problem which will be useful for parkers and also for administrators. We are offering a survey on the relevant technologies and IoT with parking in smart city [13]. From the past decade, cars are manufactured having some advanced features which include radio, automatic door locks, movie player, navigating systems, and so on. Now using IoT we can link them up with smartphone apps to find availability of parking, reservations of parking [14] and online payment mode [15]. We can secure this parking lot information using the emerging technology called blockchain. Techniques like sensor network, positioning system and image detection are also used. Table 2 explains what are the types of sensors used in parking lots in different countries like underground sensors, RFID, etc. [16].

3.3 Smart Farming

Nowadays lots of services are provided by various technologies like IoT, artificial intelligence and other technologies. Some of them are used for farming which will increase growth and reduce manpower as well as increase the quality of product. IoT plays a key role in product growth and provides better measures compared to regular farming, and the technologies offered are sensors, software solutions used for the purpose of connecting IoT platforms, connectivity's used and mobile devices along with GPS locations using robotic tractors along with different processing facilities

Table 1 Usage of IOT cloud by smart buildings

Scenario	Use cases	Device type	People population	Energy consumption	Cost	Throughput	Latency	Mobility	Reliability	Security
Smart buildings	Water metering	Sensors Metres	Large	Low	Low	Low	High	Fixed	Medium	Low
	Residential monitoring	Sensors	Few	Low	Low	Low	Low	Fixed	High	High
	Secure electrical panels	Sigfox	Few	Low	Low	Low	High	Fixed	High	High
	Smoke and fire using the Internet	Detectors	Large	Low	Low	Low	High	Fixed	Medium	Low

Table 2 Smart parking applications and technologies used in various areas

Smart parking application	Country	Sensors/technology used
Park.ME	Austria, Germany	Underground sensors
SmartParking	New Zealand	Underground sensors, RFID
ParkMe	Japan, US, UK, Germany, Brazil	Underground sensors
ParkAssist	US	M4 smart sensors, LPR
SpotHero	US	Underground sensors
EasyPark	Canada	Underground sensors
PaybyPhone	France	Underground sensors
ParkMobile	US	Underground sensors
AppyParking	UK	Magnetometer
EasyPark Group	Sweden, Denmark, etc.	Transactional data and crowdsourcing
Parker	US	Underground sensors, machine vision
ParkFi	US	Magnetometer
Best Parking	US	Underground sensors
Parkopedia	US, Germany, Sweden, etc.	Predictive analytics, Underground sensors
SFPark	US	Underground sensors
Open Spot	US	Crowdsourcing

and we can also use data streaming for other solutions. By using all these, a farmer can always monitor the field condition and can act according to it and follow using the steps like first observing the problem, then diagnosing the damage and what cause it and taking a decision to make action for that problem. We use two types of farming: One is precision farming and the other is livestock farming . In precision farming the changes in the field are measured and will act according to it, while in livestock farming, we use various smart farming techniques which will make cultivation easy for the farmer [17].

We use different ways to use IoT in farming to make agriculture more efficient. These include using sensors which collect data to improve the quality of soil, growth of the crop and health of the cattle and provide information of climate conditions. It also helps control the risk in production by predicting future problems. We can also reduce the cost as there will be a decrease in manpower and predict losses in harvesting the crop by knowing the weather conditions [18].

The monitoring of weather conditions is done by using some IoT devices like Smart Elements and allMETEO, Greenhouse automation is also done by using the Farmapp and GreenIQ which are used to find the humidity, lighting and soil conditions of the crops. The management of the crops is done by the devices like Arable and Semios, Monitoring the cattle is done by placing sensors to the cattle which will monitor their health condition like the using Cowlar. CropX is used to determine soil condition and finally drones are used to monitor and perform operations like spraying pests (Fig. 3). Table 3 shows the sensors used in smart farming technologies [19].

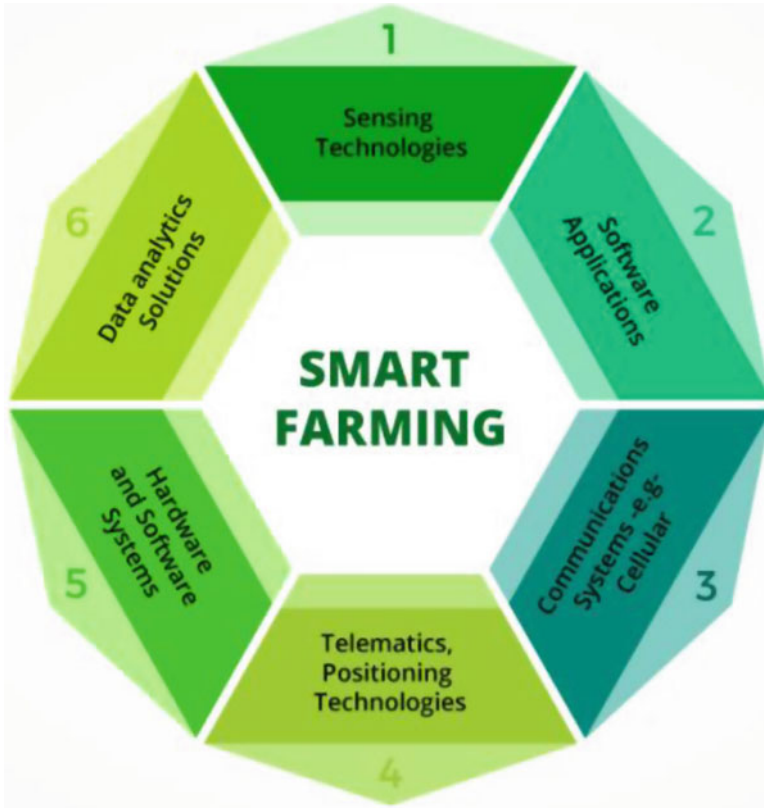


Fig. 3 Smart farming technologies

4 Conclusion

In this review paper, I addressed some of the IoT applications that use cloud architecture. These applications are very useful to the end users. With this new technology, various experiments were conducted with an objective to evaluate the performance of the application servers in the proposed IoT cloud. For example, in smart agriculture we have smarter and efficient farming methods which will increase the growth of the crop. All these aspects are mentioned in this paper briefly and cloud computing implementation is explained thoroughly and other applications are specified where IoT-based architecture is used in all these applications. Based on the following data I can conclude that IoT applications will become efficient to use by using cloud architecture. More research is being conducted to solve challenges which are faced while using IoT cloud.

Table 3 Sensors used in IoT-based agriculture

Sensor/system	Target/placed				Considered purpose/parameters							
	Plant	Equipment	Soil	Weather	Yield	Temp	Moisture	Location/tracking	Wing	Pollution/Co ²	Water	Fruit/stem size
Loup 8000i [138]		✓			✓		✓					
XH-M214 [139]			✓				✓					
Ag Premium Weather [140]		✓		✓		✓		✓				
FI-MM [141]	✓											✓
PYCNO [142]			✓	✓		✓					✓	
MP406 [143]			✓			✓						
DEERE 2630 [144]		✓			✓		✓					
Sol Chip Com (SCC) [145]				✓						✓	✓	
SenseH2TM [146]	✓								✓	✓		
DEX70 [147]	✓											✓
Piccolo ATX [148]		✓						✓				
CI-340 [149]	✓						✓			✓		
Wind Sentry 03002 [150]				✓					✓			
AQM-65 [151]				✓						✓		
POGO Portable [152]			✓			✓					✓	

References

1. L. Hou et al., “Internet of Things Cloud: Architecture and Implementation,” in *IEEE Communications Magazine*, vol. 54, no. 12, pp. 32–39, December 2016, doi: <https://doi.org/10.1109/MCOM.2016.1600398CM>.
2. C. Lee, Y. Han, S. Jeon, D. Seo and I. Jung, “Smart parking system for Internet of Things,” *2016 IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, USA, 2016, pp. 263–264, doi: <https://doi.org/10.1109/ICCE.2016.7430607>.
3. Jayaraman PP, Yavari A, Georgakopoulos D, Morshed A, Zaslavsky A. *Internet of Things Platform for Smart Farming: Experiences and Lessons Learnt*. Sensors (Basel). 2016 Nov 9;16(11):1884. doi: <https://doi.org/10.3390/s16111884>. PMID: 27834862; PMCID: PMC5134543.
4. M. Ayaz, M. Ammad-Uddin, Z. Sharif, A. Mansour and E. M. Aggoune, “Internet-of-Things (IoT)-Based Smart Agriculture: Toward Making the Fields Talk,” in *IEEE Access*, vol. 7, pp. 129551–129583, 2019, doi: <https://doi.org/10.1109/ACCESS.2019.2932609>.
5. Ademir F. da Silva, Ricardo L. Ohta, Marcelo N. dos Santos, Alecio P.D. Binotto, A Cloud-based Architecture for the Internet of Things targeting Industrial Devices Remote Monitoring and Control, *IFAC-Papers On Line*, Volume 49, Issue 30, 2016, Pages 108–113. <https://doi.org/10.1016/j.ifacol.2016.11.137>.
6. A. Chowdhury, G. Karmakar, J. Kamruzzaman, A. Jolfaei and R. Das, “Attacks on Self-Driving Cars and Their Countermeasures: A Survey,” in *IEEE Access*, vol. 8, pp. 207308–207342, 2020, doi: <https://doi.org/10.1109/ACCESS.2020.3037705>.
7. Z. Zhou, H. Yu and H. Shi, “Human Activity Recognition Based on Improved Bayesian Convolution Network to Analyze Health Care Data Using Wearable IoT Device,” in *IEEE Access*, vol. 8, pp. 86411–86418, 2020, doi: <https://doi.org/10.1109/ACCESS.2020.2992584>.
8. L. Liu, Y. Liu, L. Wang, A. Zomaya and S. Hu, “Economical and Balanced Energy Usage in the Smart Home Infrastructure: A Tutorial and New Results,” in *IEEE Transactions on Emerging Topics in Computing*, vol. 3, no. 4, pp. 556–570, Dec. 2015, doi: <https://doi.org/10.1109/TETC.2015.2484839>.
9. Luigi Atzori, Antonio Iera, Giacomo Morabito, *The Internet of Things: survey*, *Computer Networks*, Volume 54, Issue 15, 2010, Pages 2787–2805, ISSN 1389-1286, <https://doi.org/10.1016/j.comnet.2010.05.010>.
10. Goel, S.S., Goel, A., Kumar, M. *et al.* A review of Internet of Things: qualifying technologies and boundless horizon. *J Reliable Intell Environ*7, 23–33 (2021). <https://doi.org/10.1007/s40860-020-00127-w>.
11. <https://www.embitel.com/iot-cloud-application-development>
12. Butpheng, C.; Yeh, K.-H.; Xiong, H. Security and Privacy in IoT-Cloud-Based e-Health Systems—A Comprehensive Review. *Symmetry* **2020**, *12*, 1191. <https://doi.org/10.3390/sym12071191>
13. A. Alsaidi and F. Kausar, “Security Attacks and Countermeasures on Cloud Assisted IoT Applications,” *2018 IEEE International Conference on Smart Cloud (SmartCloud)*, New York, NY, USA, 2018, pp. 213–217, doi: <https://doi.org/10.1109/SmartCloud.2018.00043>.
14. A. Hameed and A. Alomary, “Security Issues in IoT: A Survey,” *2019 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)*, Sakhier, Bahrain, 2019, pp. 1–5, doi: <https://doi.org/10.1109/3ICT.2019.8910320>.
15. R. K. Dwivedi, S. Singh and R. Kumar, “Integration of Wireless Sensor Networks with Cloud: A Review,” *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, Noida, India, 2019, pp. 114–119, doi: <https://doi.org/10.1109/CONFLUENCE.2019.8776968>.
16. Keesara Sravanthi, Burugari Vijay Kumar, Prabha Selvaraj, P. Kanmani, Amit Kumar Tyagi, “Preserving Privacy Techniques for Autonomous Vehicles”, *International Journal of Emerging Trends in Engineering Research*, vol. 8, no. 9, pp. 5180–5190, Sep 2019, <https://doi.org/10.30534/ijeter/2020/48892020>.

17. P. Eugster, S. Kumar, S. Savvides and J. J. Stephen, "Ensuring Confidentiality in the Cloud of Things," in *IEEE Pervasive Computing*, vol. 18, no. 1, pp. 10–18, Jan.-March 2019, doi: <https://doi.org/10.1109/MPRV.2018.2877286>.
18. Y. Chen, "Challenges and opportunities of internet of things," 17th Asia and South Pacific Design Automation Conference, Sydney, NSW, Australia, 2012, pp. 383–388, doi: <https://doi.org/10.1109/ASPDAC.2012.6164978>.
19. Keesara. Sravanthireddy, "Blockchain-Enabled Decentralization Service for Automated Parking Systems," *Opportunities and Challenges for Blockchain Technology in Autonomous Vehicles*, edited by Amit Kumar Tyagi, et al., IGI Global, 2021, pp. 51–63. <https://doi.org/10.4018/978-1-7998-3295-9.ch004>

Prediction of Maneuvering Status for Aerial Vehicles Using Supervised Learning Methods



Abhishek Gupta, Sarvesh R. Thustu, Riti R. Thakor, Saniya A. Patil, Raunak Joshi, and Ronald Melvin Laban

Keywords Flight maneuvering · Binary classification · CatBoost

1 Introduction

The maneuvers are very important metrics for aerial vehicle trajectories. The maneuvers measure the ability of the highly coordinated movements of aerial as well as ground vehicles. The aerial vehicles require this principle at a very important level, and the detection of the maneuver can prove to be important for aerial simulations for trajectory calculation. Machine learning can be leveraged for such type of problem, as finding the state of maneuvering is a binary classification [1] problem. Machine learning in the area of supervised learning can help tackle this problem. Since supervised algorithms are taken into consideration, the constraint of using a parametric or nonparametric model is of not an utter importance. For classification, we decided to consider a variety of algorithms. We targeted Linear [2], Distance Measure [3], Discriminant Analysis [4], and Boosting Ensemble Method [5]. The linear models have a wide variety but decided to select Logistic Regression [6, 7] as it is the primordial algorithm used for binary classification problem. For distance measure, we considered working with K-Nearest Neighbors [8, 9], which is a nonparametric supervised learning method. For discriminant analysis, we considered Linear Discriminant Analysis [10–13], and for boosting ensemble method, we considered CatBoost [14–17] algorithm. These algorithms are varied and will give varied results for maneuver prediction of trajectory-based data.

A. Gupta (✉) · S. R. Thustu · R. R. Thakor · S. A. Patil · R. M. Laban
St. John College of Engineering and Management, Palghar, India
e-mail: abhishekguptal@sjcem.edu.in; sarveshtl@sjcem.edu.in; rititl@sjcem.edu.in;
saniyal@sjcem.edu.in; ronaldl@sjcem.edu.in

R. Joshi
University of Mumbai, Mumbai, India

2 Methodology

2.1 Data Preparation

The first thing is preprocessing the data in a conducive form for the machine learning algorithms. The required elements of the data typically focus on the Latitude, Longitude, Altitude, and Timestamps of all the aerial vehicles traveling from one point to destination point. Such kind of data can be used for conversions required for one single data that works for algorithms. The maneuver can be calculated using many Latitude, Longitude, and Altitude. These are very influencing parameters for many features of dataset used in this project. Latitude gives measure of North–South location from the center of the Earth and is measured in degrees. The symbol used for representing latitude is ϕ . Equator is the very line drawn horizontally, which represents center of the Earth. Latitude is 0° at Equator and 90° at both the North and South poles. Longitude on the other hand gives lines from North to South pole in a continuous manner. These lines are also called Meridians. These are denoted by λ symbol. The prime meridian is calculated from Greenwich, England where it is considered as 0° and runs all the way to $+/- 180^\circ$, from East to West keeping track of Prime Meridian. The altitude is denoted by the α symbol and ranges from 0° to 90° . The n th discrete difference is required for using Latitude, Longitude, and Altitude for calculation of maneuver. Along with maneuver, the parameters we require in data are maximum altitude, vertical acceleration, horizontal speed of plane, and distance. These will make the major features for determining the maneuver factor. The formula for vertical acceleration is calculated using mean of α , that is, altitude. The data consist of records of over 6000 files for calculation which have numerous coordinates which are calculated for parameters for dataset. The formula for horizontal speed of the plane is given by

$$S_p = \sqrt{\phi^2 + \lambda^2} \quad (1)$$

This formula gives speed which then individually is multiplied with an arbitrarily initialized variable value that represents speed of the plane as per the standards. Later, a mean of the values is taken to achieve horizontal speed of the plane. The distance can be calculated using

$$D = \sqrt{(\phi_i - \phi_j)^2 + (\lambda_i - \lambda_j)^2} \quad (2)$$

Equation 2 gives the distance, and this is an important explanatory variable that is a feature influencing the maneuver. The maximum altitude is calculated by finding the maximum value of the α . Finally, we calculate the maneuver. First, we arbitrarily denote a threshold for maneuver. The values falling above the maneuver are considered, and others are labeled as zero. The formula can be denoted as

$$M = \begin{cases} 1, & \text{if } \operatorname{argmax}|\alpha| > T \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Equation 3 states that Maneuver is calculated with a condition; the maximum absolute value of the α needs to be greater than the declared threshold T , and in such case the maneuver is given as 1. If the condition is not followed, it is 0. Using all the equations, we calculated the values for over 6000 records of aerial vehicles and used this data for machine learning algorithms.

2.2 Linear Model

The Linear Model that we consider is Logistic Regression. It is linear parametric supervised learning method that uses a logit function for prediction. The logit function is a smoothening curve-based function which is also known as sigmoid. The formula for sigmoid can be given as

$$\sigma(Z) = \frac{1}{1 + e^{-Z}} \tag{4}$$

where Z is an equation that is parametric. This function is similar to Linear Regression function which is used for continuous value predictions. The Linear Regression function is enforced smoothness using sigmoid function for discrete value predictions. This function is given as

$$Z = \beta_0 + (\beta_1 * X_i)$$

where β_0 is regression constant and β_1 is regression coefficient representing one feature set.

$$\sigma(Z) = \frac{1}{1 + e^{\beta_0 + (\beta_1 * X_i)}} \tag{5}$$

This is a linear equation and later translates to many features as required and gives the dependent variable. It yields a threshold with certain probability ratio. This can be represented in formula as

$$\hat{y} = \begin{cases} 1, & \text{if } \hat{y} \geq T \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

Equation 6 gives the representation prediction variable. \hat{y} is considered as the prediction variable. If it is greater than or equal to the threshold specified, it yields 1, else 0.

2.3 Distance Measure Model

The algorithm that we used for this type is K-Nearest Neighbor which is nonparametric supervised learning method. This algorithm takes an arbitrarily initialized K value which will be used to draw associations for all the data points from the data with respect to it. Usually, the value of K is considered as odd value and more than 3. It then calculates the similarity value of all the points under observation. This is done using some distance measure metric like Euclidean distance. The formula for Euclidean distance is given as

$$d(p, q) = \sqrt{\sum_{i=1}^n (p_i - q_i)^2} \quad (7)$$

Similarly, different types of distance metrics can be used. This helps the algorithm to determine the new sample based on K observations. Usually, distribution of the data does not matter in Nearest Neighbors. Applied dimensionality reduction techniques for Nearest Neighbors sometimes help increase the accuracy of the algorithm.

2.4 Discriminant Analysis

The algorithm we considered for Discriminant Analysis is Linear Discriminant Analysis. These algorithms use Dimensionality Reduction and works for co-variance matrix. The estimates of the co-variance matrix are known as Scatter Matrix which is a semi-definite matrix. These scatter matrices are of 2 types, between class scatter and within class scatter. The formula is given as

$$LDA = \frac{(M_1 - M_2)^2}{S_1^2 + S_2^2} \quad (8)$$

where numerator is between class scatter and denominator within class scatter. The equation needs to be expressed in the terms of W and needs to be maximized. After maximization of the numerator, the scatter matrix becomes $W^T \cdot S_b \cdot W$ and the denominator becomes $W^T \cdot S_w \cdot W$. The equation is given as

$$LDA = \frac{W^T \cdot S_b \cdot W}{W^T \cdot S_w \cdot W} \quad (9)$$

Furthermore, Eq. 9 after differentiating with respect to W, eigenvalue, and eigenvector [18] in a generalized manner is achieved.

2.5 Boosting Ensemble

The Ensemble Learning is a branch of machine learning that focuses on combining weak learners to yield a result equivalent to or better than a strong learner. The boosting has major subdivisions known as Bagging [19] and Boosting. The bagging is known as Bootstrap Aggregation and Boosting uses the set of weak learners known as stumps and combines them to achieve the best possible result. The boosting methods include many algorithms out of which we considered CatBoost. This is known as Categorical Boosting and is an advancement in the edition of boosting algorithms.

3 Results

3.1 Precision

This is a type of metric that defines the number of positive classes predicted from the total number of positive classes. The foundation of the precision [20] starts from Confusion Matrix [21]. The elements of confusion matrix are True Positives, True Negatives, False Positives, and False Negatives. The formula for precision is given by

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

The precision has 2 types viz, Macro Averaging and Weighted Averaging. The macro averaging is where the precision does consider the unweighted averages, whereas the weighted averaging considers the every class while averaging.

Table 1 Macro and weighted averaging precision

Algorithm	Macro	Weighted
Logistic regression	85%	83%
KNN	81%	85%
LDA	82%	85%
CatBoost	85%	88%

The results from Table 1 give a detailed set of macro averaging and weighted averaging values for all the algorithms used, but these values are not the fixed indication to derive a proper inference about the performance of the algorithm. For this, we will have to include more metrics that will insights about learning process. Recall is the next metric we consider.

3.2 Recall

This is a metric that focuses on the positive predictions out of all the predictions. Recall [20] also works considering the elements of confusion matrix. The formula for recall is given by

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

Just like precision, the recall also has the macro average and weighted average process.

Table 2 Macro and weighted averaging recall

Algorithm	Macro	Weighted
Logistic regression	66%	82%
KNN	79%	85%
LDA	78%	86%
CatBoost	81%	88%

Table 2 gives the metrics, and it derives that Logistic Regression performs the worst. The indication is done using variation in macro and weighted averaging system. The least calculated variation for macro and weighted is in CatBoost.

3.3 F₁-Score

The F₁-Score [20] is a metric of accuracy that is calculated using precision and recall. The metric clearly gives assumption of the performance of the learning models. The formula for F₁-Score is given by

$$F_1 = 2 * \left(\frac{P * R}{P + R} \right)$$

This in terms of confusion matrix elements can be represented as

$$F_1 = \frac{TP}{TP + \frac{1}{2} * (FP + FN)} \tag{12}$$

Below is given a table that gives the F₁-Score for all the algorithms used, viz. Logistic Regression, K-Nearest Neighbors, Linear Discriminant Analysis, and CatBoost (Table 3).

Table 3 F_1 -score

Algorithm	Score
Logistic regression	82%
KNN	85%
LDA	86%
CatBoost	88%

The F_1 -Score gives the accuracy, and as per the results, CatBoost is able to give the best performance followed by LDA, followed by KNN, and finally by Logistic Regression.

3.4 Receiver Operating Characteristics and Area Under Curve

Receiver Operating Characteristic abbreviated as RoC [22] is performance measurement metric for classification models considering thresholds. It is formed using True Positive Rate and False Positive Rate [23]. But the RoC is not sufficient, Area Under Curve abbreviated as AUC is essential for representing the measure of separability. The value of AUC is plot between 0 and 1. Closer the AUC to 1, better is the algorithm. The elements required for the graph need to be explained. True Positive Rate is also known as sensitivity and also known as recall. The formula is as same as that of recall. For False Positive Rate, one cannot directly calculate the value of it. One needs to calculate specificity first, which is represented by the formula

$$Specificity = \frac{TN}{TN + FP}$$

and now this can be used to find the False Positive Rate with formula as

$$FPR = 1 - Specificity \tag{13}$$

Now, assuming this notion, we have implemented the RoC and AUC for all the classifiers used in this chapter

Figure 1 gives the graphical representation of the classifiers used. The red diagonal is the threshold; any algorithm that falls below it is not considered a good algorithm. The inference can be drawn from the graph that CatBoost is the best classifier for the maneuver classification.

4 Conclusion

The data for aerial vehicles are allocated at a very high rate and can be definitely used to find the maneuver based on certain parameters. This is a binary classification

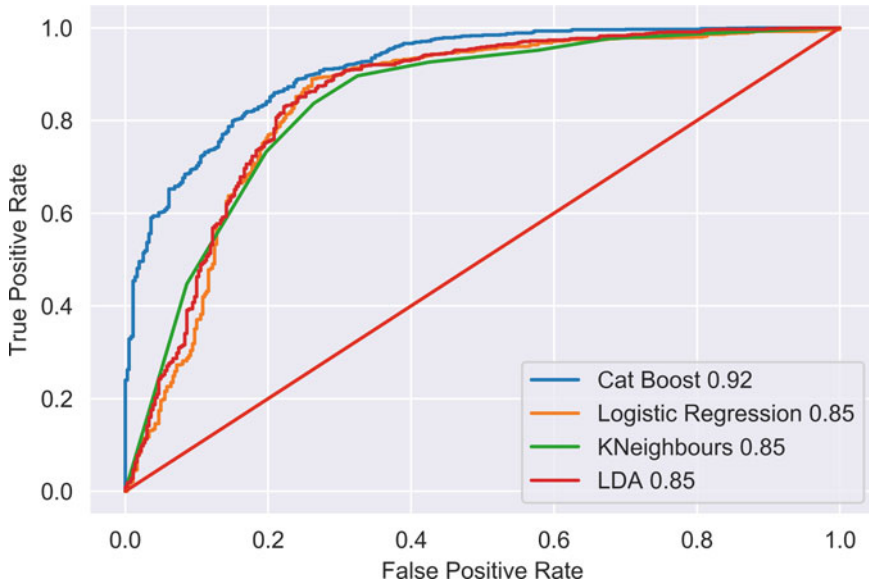


Fig. 1 RoC with AUC

problem and can be managed by leveraging machine learning. We wanted to provide the approach to tackle the problem in every single aspect of machine learning. Since it is binary classification, using supervised learning methodology was straightforward approach, in which we considered working with Linear Model, Distance Metric Model, Discriminant Analysis, and Boosting Ensemble Model. We used the best algorithm of each single approach that suits the best for our data. We used Logistic Regression for linear, KNN for distance metric, Linear Discriminant Analysis for discriminant analysis, and CatBoost for boosting ensemble method. We provided the algorithms with metrics from different facets and proved the potential of every algorithm, and we successfully proved that maneuver detection for aerial vehicle data is possible. Obviously, many more improvements in future can be done in future, and we would be glad if someone else referred our work.

References

1. Kumari, R., Srivastava, S.K.: Machine learning: A review on binary classification. *International Journal of Computer Applications* **160**(7), 11–15 (Feb 2017). <https://doi.org/10.5120/ijca2017913083>, <http://www.ijcaonline.org/archives/volume160/number7/27084-2017913083>
2. Generalized linear models. *Journal of the Royal Statistical Society. Series A (General)* **135**(3), 370–384 (1972), <http://www.jstor.org/stable/2344614>
3. Pandit, S., Gupta, S.: A comparative study on distance measuring approaches for clustering. *International Journal of Research* **2**, 29–31 (2011)

4. Ramayah, T., Ahmad, N.H., Halim, H.A., Zainal, S.R.M., Lo, M.C.: Discriminant analysis: An illustrated example. *African Journal of Business Management* **4**, 1654–1667 (2010)
5. Opitz, D., Maclin, R.: Popular ensemble methods: An empirical study. *Journal of artificial intelligence research* **11**, 169–198 (1999)
6. Cramer, J.S.: The origins of logistic regression (2002)
7. Chung, M.K.: Introduction to logistic regression (2020)
8. Guo, G., Wang, H., Bell, D., Bi, Y., Greer, K.: KNN model-based approach in classification. In: Meersman, R., Tari, Z., Schmidt, D.C. (eds.) *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*. pp. 986–996. Springer Berlin Heidelberg, Berlin, Heidelberg (2003)
9. Taunk, K., De, S., Verma, S., Swetapadma, A.: A brief review of nearest neighbor algorithm for learning and classification. In: 2019 International Conference on Intelligent Computing and Control Systems (ICCS). pp. 1255–1260 (2019). <https://doi.org/10.1109/ICCS45141.2019.9065747>
10. Gaber, T., Tharwat, A., Ibrahim, A., Hassaniien, A.: Linear discriminant analysis: a detailed tutorial. *AI Communications* **30**(2), 169–190 (2017). <https://doi.org/10.3233/AIC-170729>, <http://usir.salford.ac.uk/id/eprint/52074/>
11. Li, C.: Fisher linear discriminant analysis (2014)
12. Shashoa, N.A.A., Salem, N.A., Jleta, I.N., Abusaeeda, O.: Classification depend on linear discriminant analysis using desired outputs. In: 2016 17th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA). pp. 328–332 (2016). <https://doi.org/10.1109/STA.2016.7952041>
13. Gupta, A., Soni, H., Joshi, R., Laban, R.M.: Discriminant analysis in contrasting dimensions for polycystic ovary syndrome prognostication. arXiv preprint arXiv:2201.03029 (2022)
14. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A.V., Gulin, A.: CatBoost: unbiased boosting with categorical features (2019)
15. Dorogush, A.V., Ershov, V., Gulin, A.: CatBoost: gradient boosting with categorical features support (2018)
16. Ibrahim, A.A., Ridwan, R.L., Muhammed, M.M., Abdulaziz, R.O., Saheed, G.A.: Comparison of the CatBoost classifier with other machine learning methods. *International Journal of Advanced Computer Science and Applications* **11**(11) (2020). <https://doi.org/10.14569/IJACSA.2020.0111190>, <https://doi.org/10.14569/IJACSA.2020.0111190>
17. Gupta, A.M., Shetty, S.S., Joshi, R.M., Laban, R.M.: Succinct differentiation of disparate boosting ensemble learning methods for prognostication of polycystic ovary syndrome diagnosis. In: 2021 International Conference on Advances in Computing, Communication, and Control (ICAC3). pp. 1–5. IEEE (2021)
18. Denton, P., Parke, S., Tao, T., Zhang, X.: Eigenvectors from eigenvalues: a survey of a basic identity in linear algebra. *Bulletin of the American Mathematical Society* **59**(1), 31–58 (2022)
19. Breiman, L.: Bagging predictors. *Mach. Learn.* **24**(2), 123–140 (Aug 1996). <https://doi.org/10.1023/A:1018054314350>, <https://doi.org/10.1023/A:1018054314350>
20. Powers, D.M.W.: Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation (2020)
21. Ting, K.M.: Confusion matrix. In: *Encyclopedia of Machine Learning* (2010)
22. Bradley, A.P.: The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognit.* **30**, 1145–1159 (1997)
23. van Ravenzwaaij, D., Ioannidis, J.P.A.: True and false positive rates for different criteria of evaluating statistical evidence from clinical trials. *BMC Medical Research Methodology* **19** (2019)

HRescue: A Modern ML Approach for Employee Attrition Prediction



Rudresh Veerkhare, Parshwa Shah, Jiten Sidhpura, and Sudhir Dhage

Keywords Employee attrition · Federated learning · SHAP analysis · Gradient boosting · ADASYN

1 Introduction

Employees are the key asset of any organization. Each company grows by cumulative efforts by all its employees. Matching the right talent with the correct job position is crucial for the HR department. Each company invests significant time and money in employee screening, interviews, hiring, onboarding, training, benefits, and retention policies. However, in this competitive world, the demand for highly skilled employees has skyrocketed, thus giving employees lucrative opportunities. Retaining proficient employees is becoming difficult day by day which has given rise to a phenomenon called attrition, employees leaving the company through resignation, retirement, or some other personal reasons.

According to [1], 31% of new hires have left the job within six months. Attrition is detrimental to the entire organization. There are various reasons for employee attrition such as not being excited about the work assigned, not having enough growth opportunities, inadequate compensation, not enough salary hikes, and superficial and toxic relationships with their managers and colleagues. Not getting adequate leaves and quality family time reduces their motivation to work.

These authors “Rudresh Veerkhare, Parshwa Shah, and Jiten Sidhpura” contributed equally to this work.

R. Veerkhare (✉) · P. Shah · J. Sidhpura · S. Dhage
Department of Computer Engineering, Sardar Patel Institute of Technology, Mumbai, India
e-mail: rudresh.veerkhare@spit.ac.in; parshwa.shah@spit.ac.in; jiten.sidhpura@spit.ac.in;
sudhir_dhage@spit.ac.in

So, each organization should take care of employees' professional growth, work-life balance, and physical and mental health to reduce attrition.

We are now a part of a data-driven economy, where data speaks more than anything. Various factors influence the attrition rate, and companies can get a lot of insights about employee attrition from their employee data. Accordingly, companies can introduce various benefits, policies, healthcare programs, workshops, and volunteering activities to enhance employee morale and thus retain their employees. Retaining employees eventually creates a win-win situation for both the company and its employees.

The objective of this research study is to utilize machine learning models to predict whether an employee working in an organization will continue to work or leave the organization. We have also focused on explaining the predictions of our classification models to determine how input data features affect attrition.

We have followed three approaches in our proposed methodology with the input dataset. The methodology flow diagram is mentioned in Fig. 1. In the first approach, we have an original imbalanced dataset. In the second approach, we have applied ADASYN oversampling technique to balance the dataset. After data preprocessing, classifier models were used to classify attrition class. Finally, we evaluate the model and perform model prediction analysis using SHAP values. In the third approach, we used federated learning approach where we train the model on multiple clients and perform aggregation to get a robust global model.

The structure of the chapter is as follows: Sect. 2 describes the literature survey, where we discussed the existing approaches. In Sect. 3, we have proposed our methodology. Section 4 contains information about Model Prediction Analysis. Section 5 demonstrates the experimental setup. In Sect. 6, we have presented our results. Finally, in Sect. 7, we conclude our work.

2 Literature Survey

Saeed Najafi-Zangeneh et al. [2] proposed a machine learning framework for employee attrition prediction. They have used the popular IBM HR dataset for evaluating their proposed methodology. Since the number of features in the dataset is very high, the authors have proposed max-out algorithm as their feature selection method. After that, they trained a Logistic Regression model on these newly selected features. The authors used F1-Score for evaluating their model because the IBM dataset is imbalanced and achieved an accuracy score of 0.81 and an F1-Score of 0.56. Finally, they trained their model on multiple bootstrapped datasets, followed by computing the average and standard deviation of model parameters to validate their stability.

Shikha N. Khera and Divya [3] trained a Support Vector Machine (SVM) model to predict the employee attrition rate specifically for the Indian IT industry. They gathered the HR data from three Indian IT firms and developed a dataset containing 1650 employees. Their dataset was imbalanced, with around 83.87% of the data

belonging to the class in which employee is working for the organization (active). The remaining data belong to the class where the employee has left the organization (inactive). Originally there were 22 features in the dataset, but after applying the feature selection technique, there were 19 features considered. The authors then explained why they selected SVM over other models such as logistic regression, K-Nearest Neighbors, Random Forests, and Decision Trees. Their final model gave an accuracy score of 85% in predicting attrition.

Priyanka Sadana and Divya Munnuru [4] conducted surveys with current employees and alumni of an IT company and then created a dataset with 1649 records. For exploratory data analysis, they determined relationships between the target variable and the predictor variables in the dataset. Their dataset was imbalanced, and they solve that problem with the help of Synthetic Minority Oversampling Technique (SMOTE) was used to balance the data. Because of encoding categorical data, the dimensionality of the dataset increased. The authors thus used Principal Component Analysis (PCA) for feature reduction. Later, they trained multiple models such as Logistic Regression and Random Forest. The authors focused more on the Recall metric for evaluating their models. Random Forest after hyperparameter tuning gave a Recall score of 93%.

Sarah S. Alduayj and Kashif Rajpoot [5] have proposed various machine learning techniques to predict employee attrition. The publicly available “IBM Watson Analytics dataset” was used. It contained 32 features and 1470 employee data with 1233 “no” and 237 “yes” attrition categories. They removed two features and converted non-numerical data to numeric. They tried three different techniques. After trying each approach, they applied fivefold cross-validation and feature selection. First, they applied machine learning models directly to the imbalanced dataset. The maximum accuracy and F1-Score were 87.1% and 0.503, respectively, given by Quadratic SVM. Second, they used Adaptive Synthetic Sampling (ADASYN) approach to balance the dataset by oversampling the minority “yes” class. The best F1-Score was 0.967, given by KNN ($K = 1$). The best F1-Scores were 0.829, 0.806, and 0.909 for 2, 3, and 12 features, respectively, by Random Forest. Third, they manually undersampled the data with 237 observations of each class. The best F1-Score was 0.74 by Quadratic SVM. In the end, they achieved an F1-Score of 0.909 with Random Forest after selecting 12 out of 29 features.

Praphula et al. [6] tested different machine learning algorithms to predict employee attrition. The study was done to find different factors influencing the attrition rate and their possible solutions. The publicly available “HR management dataset” was used. It contained 10 features and 14,000 records. All records were divided into 10 business departments. As a part of exploratory data analysis, the authors performed variable identification, univariate and Bi-variate analysis, and correlation. The rows with missing values were removed. The dataset was split into training and testing sets. They tried three machine learning algorithms like Support Vector Machine with Radial Basis Function Kernel, Decision Tree, and Random Forest. These algorithms were applied to 4 departments with more than 1000 employees with evaluation parameters like Precision, Recall, and F1-Score.

Random Forest Classifier performed the best with precision 0.99, 0.97, 0.98, and 0.99 for Sales, Technical, Support, and IT departments, respectively.

Francesca Fallucchi et al. [7] proposed a machine learning approach for employee attrition using objective features. The authors have used the IBM HR dataset for their study. In data preprocessing, redundant features are removed, categorical features are label encoded, and numerical features are scaled using a standard scaler. They conducted a feature correlation analysis to determine feature importance as well as an extensive descriptive analysis, taking Attrition as a target feature. Multiple predictive models are used based on methods such as the Bayesian method, SVM, decision trees, and Logistic Regression. The training set contains 70% of the data, while a test and validation set contains the remaining 30%. Also, the authors used Holdout and cross-validation for the evaluation of models. Among all models, Gaussian Naive Bayes gave the best F1-Score of 0.446 and the lowest false positive rate of 4.5% along with the highest true positive rate of 72%.

Usha et al. [8] proposed various machine learning techniques to predict employee attrition. They used publicly available “IBM HR Analytics dataset.” It contained 35 features and 1470 employee data with 1233 “no” and 237 “yes” attrition categories. They used the Weka tool to apply machine learning. The irrelevant and redundant columns are discarded. Weka tool is used to convert dataset CSV file to ARFF (Attribute-Relation File Format) for its processing. They applied Decision Tree and Naive Bayes classifier giving an accuracy of 82.449% and 78.8435%, respectively, on tenfold cross-validation. Then, they applied K-Means and Expectation Maximization clustering algorithms giving a correct classification rate of 57.2789% and 55.102%, respectively.

Rachna Jain and Anand Nayyar [9] have proposed to use gradient boosting-based methods such as XGBoost and GLMBoost. The authors have performed a thorough exploratory analysis and feature engineering. IBM dataset was used for modeling. The correlation of attributes was analyzed to filter given features. Also, a class imbalance problem is not addressed. Finally, XGBoost model gave an accuracy score of 89%

Marc Guirao [10] has used the IBM HR Analytics dataset to predict employee attrition with the help of machine learning. The author has extensively used KNIME open-source data analytics software. Dataset was split into 80% for training and 20% for testing. Being an imbalanced dataset, the minority class was upsampled using Synthetic Minority Oversampling Technique (SMOTE) algorithm during training. The author then trained four machine learning models such as Naive Bayes, Random Forest, Logistic Regression, and Gradient Boosting. Finally, the best score was given by Random Forest on the imbalanced test set with an accuracy score of 0.89 and an F1-Score of 0.59.

To summarize the literature survey, the IBM HR dataset is severely imbalanced. To tackle it, some of the papers have oversampled the minority class. Most of the existing approaches are predicting employee attrition using the centralized approach with the help of classical machine learning algorithms such as Logistic Regression, Decision Trees, and Random forest.

3 HRescue Methodology

3.1 Dataset Description

IBM's data scientists have created this publicly available dataset named IBM HR Analytics [11]. The dataset contains 1470 records and 35 features where the deciding feature about employee attrition is the "Attrition" column containing "Yes" and "No" labels. Feature datatypes are shown in Table 1. The dataset has an issue of class imbalance because 1233 records are from the "No" category and 237 are from the "Yes" category. There were no missing values in the entire dataset. All employees in the dataset are aged above 18. The dataset has details about employees' education, marital status, previous work history, and more. EmployeeCount, Over18, and StandardHours are some features from the dataset that were discarded as they contained only one unique value. EmployeeNumber feature had unique values for all the records, so we ignored it as it did not provide any meaningful information.

3.2 Dataset Balancing

Dataset is severely imbalanced, having only 237 positives and 1233 negative samples for the target class Attrition as shown in Fig. 2. Class Imbalance makes it difficult for a machine learning model to generalize better on given data. So, to tackle this problem, we are using Adaptive Synthetic (ADASYN) sampling [12] to upsample the minority class. ADASYN creates synthetic points by using the density distribution of the minority class. Categorical features are encoded using OneHotEncoding for upsampling and then inversely transformed after upsampling, to retain the original form of the dataset. After upsampling, our balanced dataset has 1160 Attrition and 1233 Non-Attrition samples.

3.3 Predictive Models

Gradient Boosting [13] is the technique to combine the power of many small learning models (learners). Generally, decision trees are considered learners for this technique. All the trees are connected sequentially with each subsequent decision tree learning from the errors made by the previous decision tree. While making the final prediction, every decision tree contributes. Learning rate (*alpha*) and *n_estimators* (the number of trees) are important hyperparameters in the gradient boosting algorithm. Learning rate defines the rate at which the model learns, a model trained with a low learning rate makes it better but requires more computations. Using a high number of trees can make the model overfit.

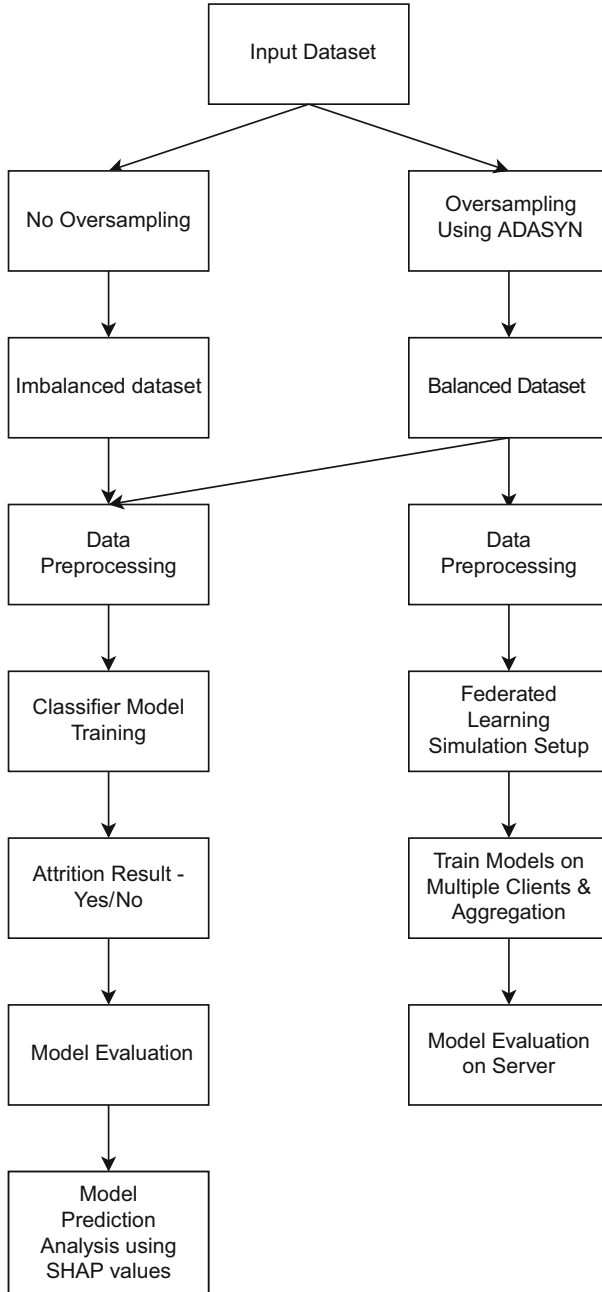


Fig. 1 HRRescue methodology flow diagram

Table 1 IBM HR dataset description

Feature	Type ^b	Feature	Type ^b
Age	N	MonthlyIncome	N
BusinessTravel	C	MonthlyRate	N
Daily rate	N	YearsWithCurrentManager	N
Department	C	YearsSinceLastPromotion	C
DistanceFromHome	N	YearsInCurrentRole	C
Education	C	YearsAtCompany	N
EducationField	C	WorkLifeBalance	C
EmployeeCount	N	TrainingTimesLastYear	N
EmployeeNumber	N	TotalWorkingYears	N
EnvironmentSatisfaction	C	StockOptionLevel	C
Gender	C	StandardHours	N
HourlyRate	N	RelationshipSatisfaction	C
JobInvolvement	C	PerformanceRating	N
JobLevel	C	PercentSalaryHike	N
MaritalStatus	C	OverTime	C
JobRole	C	Over18	C
JobSatisfaction	C	NumCompaniesWorked	N
Attrition ^a	C		

^a Target Variable

^b (N = Numerical, C = Categorical)

XGBoost

It is an implementation of gradient boosting created by Tianqi Chen [14]. It is a Stochastic Gradient Boosting with both L1 and L2 regularization. It automatically handles missing values and supports parallel tree construction. The data science community widely uses XGBoost because of its high execution speed and model performance.

CatBoost

It is a gradient boosting tree-based model with out-of-box support for categorical features. Because of this, there is no need to perform any preprocessing on categorical features. The algorithm converts the categorical features into numerical values with the help of target statistics. It is designed by Yandex [15] and has applications in the fields of self-driving cars, weather prediction, and more. The CatBoost algorithm has performed better than other state-of-the-art gradient boosting algorithms such as XGBoost, LightGBM on Epsilon, Amazon, and other standard machine learning datasets.

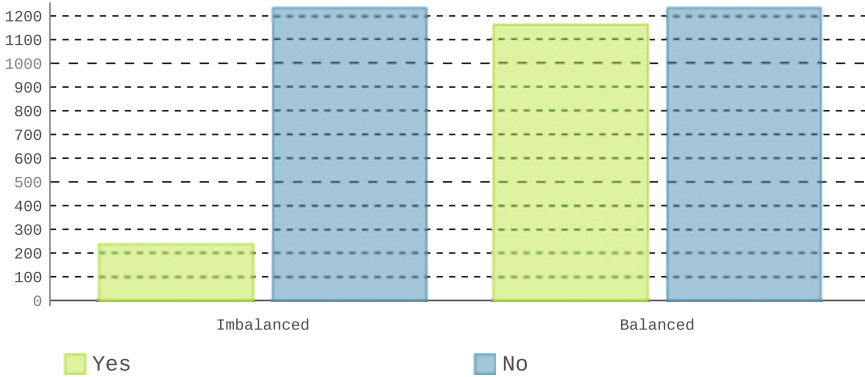


Fig. 2 Balanced vs imbalanced dataset sample counts

3.4 Model Training

We have conducted three experiments in our methodology, one on the imbalanced dataset and the other two on the balanced dataset, sample count comparison for both datasets is given in Fig. 2. In all the experiments, 70% of the dataset is used for training, remainder 30% is for validation.

Imbalanced Dataset

Due to the class imbalance problem, it was difficult for the models to classify the minority class samples. To tackle this issue, we added extra weight in the Logloss (Binary Crossentropy) loss function when it makes an error while predicting the minority class sample with categorical features in the dataset and XGBoost supporting only numerical data. These features were converted to numerical format using the LabelEncoder Class from the sklearn's preprocessing module while training with XGBoost. In the case of CatBoost, no preprocessing is performed on such categorical data. To avoid overfitting, we also used a high value of L2 regularization in our models.

Balanced Dataset

A balanced dataset does not contain a class imbalanced problem, so there are no extra weights used in Logloss function. The same preprocessing as imbalanced dataset is used.

3.5 *Federated Learning Experiment*

Federated machine learning [16] is a distributed ML approach addressing data privacy issues. In this approach, client data never leave the client device, rather an ML model is trained on the edge device, i.e., on the client device. The model parameters of the trained model are then securely transferred over the network. Model parameters from multiple models of various clients are aggregated using algorithms like FedAverage [16] to create a robust global model.

Having a global system to predict employee attrition is challenging because employee data is sensitive to corporations. This can be addressed by using the federated machine learning approach. Using this method, corporations can train local models on their data. Then, aggregate all these models to make a global model. This would enable utilizing a large amount of employee data from different corporations.

Dataset

For this experiment, we are using the same size samples of the IBM dataset. Before sampling, the dataset is balanced using ADASYN, and Principle Component Analysis (PCA) is performed on the data. In a real-world implementation, each client would have their own data.

Simulation Setup

For the simulation, we are using five clients each having an equal amount of data. After each round of training, model parameters are sent over the network to the server, and then the server aggregates them using FedAverage. This new model is then evaluated on test data and distributed to clients for the next iteration. We are using Logistic Regression and a simple neural network for this experiment.

Logistic Regression

This is a simple linear model used to classify linearly separable data. In our experiment, sklearn implementation is used and internal parameters such as coefficient and intercepts are transferred over the network for aggregation. The global model achieved an accuracy of 87% on the validation data.

Neural Network

The network consists of two hidden layers with 16 and 8 neurons, respectively, that use ReLU as an activation function. An output layer consists of a single neuron with a sigmoid as an activation function. This network has 641 trainable parameters, which are transferred over the network. Models are trained for 50 iterations. The final global model achieved an accuracy of 89% on the validation set.

3.6 Evaluation Metrics

We have used the Logloss function for our classification problem in our research study. As shown in Eq. (1), the Logloss value is calculated as the negative summation of the actual label (y_i) times log of the probability of class 1 (p_i) and (1—actual label (y_i)) times log of the probability of class 0 ($1 - p_i$), divided by the total number of samples (N).

$$\text{Logloss} = -\frac{1}{N} \sum_{i=1}^N (y_i \log(p_i) + (1 - y_i) \log(1 - p_i)) \quad (1)$$

Because of the imbalanced nature of the dataset, Accuracy does not provide information about how accurate the model is for predicting the minority class samples. Hence, we have used the F1-Score to evaluate our models. This metric as shown in Eq. (2) is harmonic mean of Precision and Recall.

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (2)$$

4 Model Prediction Analysis

In any business, the reason for prediction by a model is equally important as the model predicting accurately. The Human Resources (HR) department of the company can take appropriate policies or decisions if they know why a particular employee has left. For this, we have computed SHAP (Shapley Additive Explanations) [18] values of the CatBoost model trained on the original imbalanced dataset. SHAP values are used to explain why a model has predicted for the given sample of features in a certain way. From Fig. 3, the features displayed on the left of the summary plot are in the decreasing order of feature importance. OverTime

is the most important feature for prediction, while PerformanceRating is the least important. From the right side of the plot, we can see that as the value of a particular features increases it is marked in red or blue if it decreases. From the OverTime feature, we conclude that when the value of the OverTime feature is high (1), it makes the predictions toward Attrition Class and Non-Attrition class if it is low (0). Similarly, we can derive how each feature impacts model prediction. It can help the respective HR department to determine the common factors causing attrition. Calculating SHAP values of a single example can find reasons driving employees to leave the firm at an individual level. It would even be more beneficial to the HR department. In Fig. 4, it can be seen which features may have caused for an employee to continue or leave the job.

5 Experimental Setup

We performed our proposed methodology extensively on Google Collaboratory. Our environment used Python (v3.7.12), scikit-learn (v1.0.2), CatBoost (v1.0.4), and XGBoost (v0.90). Both CatBoost and XGBoost models have many hyperparameters to tune. To efficiently get the best results, we have used popular hyperparameter optimization frameworks Optuna (v2.1.0) and Hyperopt (v1.2.7). While training, the model that gave the best performance on the evaluation dataset is selected in all the experiments. For simulations of federated learning, experiments are carried out on a local machine having Python (v3.8.0) and Flower federated framework [17] (v0.17.0).

6 Results

In this section, we have discussed the performances of our models with previous work on both the balanced and the imbalanced dataset. We have compared the accuracy and F1-Score measures with the existing approaches. In both versions of the dataset, CatBoost performed slightly better than XGBoost. From Table 2, we can say that our models have outperformed by a margin of 15% in the F1-Score on the imbalanced dataset. As shown in Table 3, on the balanced dataset, CatBoost gave an F1-Score of 0.94 and XGBoost gave 0.93, respectively. The authors of [5] got an F1-Score of 0.97 with KNN ($K = 1$), but they have mentioned that their model was overfitted.

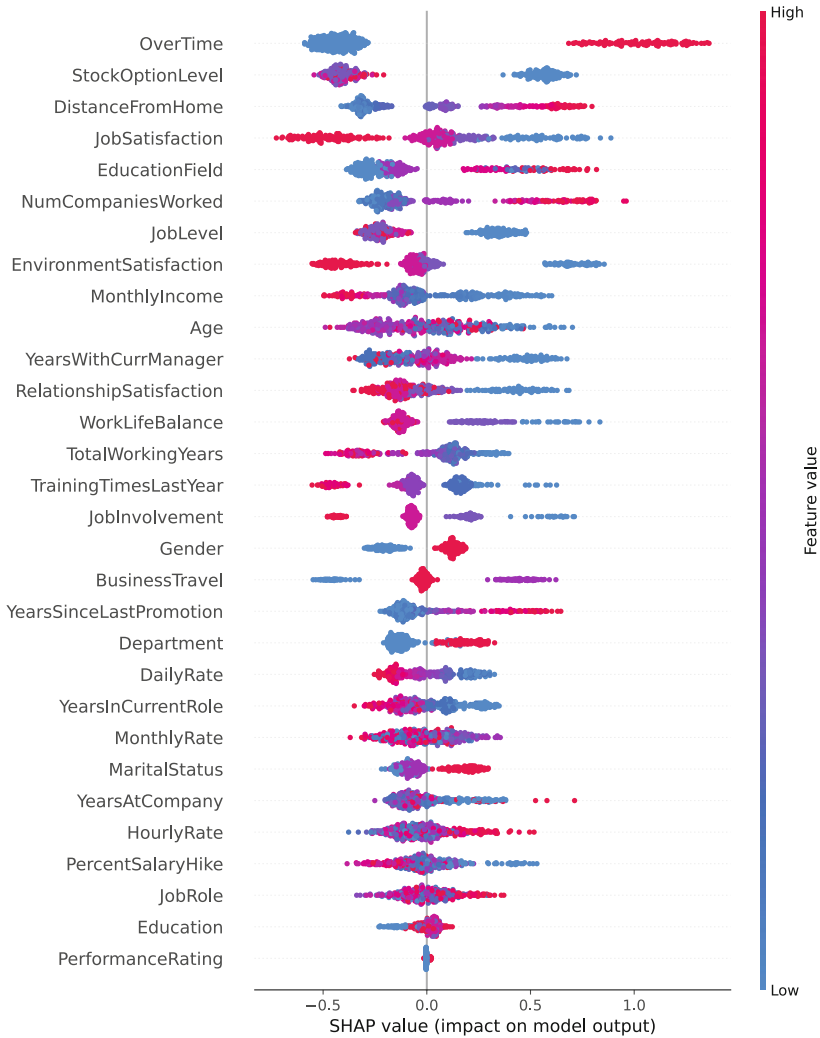


Fig. 3 SHAP values summary plot imbalanced dataset using CatBoost model

7 Conclusion and Future Scope

Employee attrition is one of the biggest obstacles for companies. We have performed 3 experiments as a part of our methodology. Firstly, we have trained our CatBoost and XGBoost classifier models on an imbalanced dataset. The results obtained are an F1-Score of 0.69 and an accuracy of 90%. Secondly, we trained our XGBoost and CatBoost classifier models on the balanced dataset. The results obtained are an F1-Score of 0.94 and an accuracy of 94%. Finally, we applied the federated learning



Fig. 4 SHAP values of individual test sample

Table 2 Results and comparison of F1-score and accuracy on imbalanced dataset

Paper	Model	Accuracy	F1-score
[5]	Linear SVM	0.87	0.37
	Quadratic SVM	0.87	0.5
	Cubic SVM	0.84	0.46
	Gaussian SVM	0.87	0.34
	Random forest	0.86	0.27
	KNN(K = 1)	0.83	0.08
	KNN(K = 3)	0.84	0.01
[7]	Gaussian NB	0.83	0.45
	Bernoulli NB	0.85	0.38
	Logistic regression	0.88	0.45
	KNN	0.85	0.15
	Random forest	0.86	0.19
	Decision tree	0.82	0.35
	SVM	0.86	0.17
	Linear SVM	0.88	0.36
[2]	Logistic regression	0.78	0.53
	Logistic regression (feature selection)	0.81	0.56
[8]	Decision tree	0.83	–
	Naive Bayes	0.81	–
[9]	XGBoost	0.89	0.60
[10]	Logistic regression	0.79	0.52
	Naive Bayes	0.83	0.49
	Gradient boosting	0.88	0.58
	Random forest	0.89	0.59
Ours	CatBoost	0.90	0.69
	XGBoost	0.90	0.68

technique to the balanced dataset. We hope that this solution creates a win-win situation for employers and employees. As a part of future work, we would like to create a more robust model trained on a large, realistic, and balanced dataset. We can add new features to the dataset after taking feedback from employees who left.

Table 3 Results and comparison of F1-score and accuracy on balanced dataset

Paper	Model	F1-score	Accuracy
[5]	Linear SVM	0.78	0.78
	Quadratic SVM	0.88	0.88
	Cubic SVM	0.93	0.93
	Gaussian SVM	0.91	0.91
	KNN(K = 1) ^a	0.97	0.97
	KNN(K = 3)	0.93	0.93
	Random forest	0.93	0.92
Ours	Catboost	0.94	0.94
	XGBoost	0.93	0.93

[5] have mentioned that their model was overfitted

References

- Bamboohr Blog, <https://www.bamboohr.com/blog/onboarding-infographic/>. Last accessed 7 Feb 2022
- S. Najafi-Zangeneh, N. Shams-Gharneh, A. Arjomandi-Nezhad, and S. Hashemkhani Zolfani, "An Improved Machine Learning-Based Employees Attrition Prediction Framework with Emphasis on Feature Selection," *Mathematics*, vol. 9, no. 11, p. 1226, May 2021, <https://doi.org/10.3390/math9111226>.
- Khera, Shikha N. "Predictive Modelling of Employee Turnover in Indian IT Industry Using Machine Learning Techniques." *Vision* 23, no. 1 (March 2019): 12–21. <https://doi.org/10.1177/0972262918821221>.
- P. Sadana and D. Munnuru, "Machine Learning Model to Predict Work Force Attrition," 2021 6th International Conference for Convergence in Technology (I2CT), 2021, pp. 1–6, <https://doi.org/10.1109/I2CT51068.2021.9418140>.
- S. S. Alduayj and K. Rajpoot, "Predicting Employee Attrition using Machine Learning," 2018 International Conference on Innovations in Information Technology (IIT), 2018, pp. 93–98, <https://doi.org/10.1109/INNOVATIONS.2018.8605976>.
- Jain, P.K., Jain, M. & Pamula, R. Explaining and predicting employees' attrition: a machine learning approach. *SN Appl. Sci.* 2, 757 (2020). <https://doi.org/10.1007/s42452-020-2519-4>
- Fallucchi, F.; Coladangelo, M.; Giuliano, R.; William De Luca, E. Predicting Employee Attrition Using Machine Learning Techniques. *Computers* 2020, 9, 86. <https://doi.org/10.3390/computers9040086>
- Usha, P.M.; Balaji, N. Analysing employee attrition using machine learning. *Karpagam J. Comput. Sci.* 2019, 13, 277–282.
- R. Jain and A. Nayyar, "Predicting Employee Attrition using XGBoost Machine Learning Approach," 2018 International Conference on System Modeling & Advancement in Research Trends (SMART), 2018, pp. 113–120, <https://doi.org/10.1109/SYSMART.2018.8746940>.
- Predicting Employee Attrition with Machine Learning, Knime Blog, <https://www.knime.com/blog/predicting-employee-attrition-with-machine-learning>. Last accessed 14 Mar 2022
- IBM HR Dataset, Liu: Attrition, (2020). <https://doi.org/10.5281/zenodo.4323396>
- Haibo He, Yang Bai, E. A. Garcia and Shutao Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), 2008, pp. 1322–1328, <https://doi.org/10.1109/IJCNN.2008.4633969>.
- Friedman, J. H. (2002). Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4), 367–378.

14. Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16). Association for Computing Machinery, New York, NY, USA, 785–794. <https://doi.org/10.1145/2939672.2939785>
15. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31.
16. McMahan, B., Moore, E., Ramage, D., Hampson, S. & Arcas, B.A.y.. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data. Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, in Proceedings of Machine Learning Research 54:1273–1282 Available from <https://proceedings.mlr.press/v54/mcmahan17a.html>.
17. Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Parcollet, T., de Gusmão, P. P., & Lane, N. D. (2020). Flower: A friendly federated learning research framework. arXiv preprint arXiv:2007.14390.
18. Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.

Using Machine Learning to Detect Botnets in Network Traffic



Shambhavi Rai, S. A. Sajidha, V. M. Nisha, and B. Mahalakshmi

Keywords Botnet · Botnet detection · IoT · DDoS · CTU-13

1 Introduction

Now a new definition of currency has emerged in this era of the Internet. Today we have started treating data like currency. And why should we not when everything has been digitized. We give the name of cyber attack to the tampering or theft done with this currency. These attacks are done on the confidentiality, integrity, and availability of the currency and the assets that generate them. And as the digitalization is increasing, in the same way the data generated from this digitalization, i.e., the currency as well as manipulations with this currency. The thing to note is that more than 80% of cyber attacks that occur on the Internet every day are launched by botnets. Therefore, it becomes very important to know what botnets are and how an organization can be saved from them [1]. Botnets [2] as the name speaks for itself are a network of robots and the botmaster of that botnet is the one who programs and operates them all. First, the botmaster writes a program of a small virus or worm and then hacks many systems through the network with this virus or worm [3]. Now all these hacked systems act like sleeper cells. All these are now the weapon of that botmaster who can on his one signal take down the server or service of any company or organization in a few minutes, which breaks one of the three fundamental principles of information security, i.e., availability. Because most network connections follow TCP 3-way handshake. Now the end user has to send a last synchronization acknowledgment to accept the syn request sent from the server after a connection request has been made by the client so that the connection between the server and the end user is finally established and transfer of information becomes possible. However, in case of attack on the availability principle of CIA

S. Rai (✉) · S. A. Sajidha · V. M. Nisha · B. Mahalakshmi
School of Computer Science and Engineering, VIT University, Chennai, India
e-mail: sajidha.sa@vit.ac.in; nishavm@vit.ac.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
R. Misra et al. (eds.), *Machine Learning and Big Data Analytics*,
Springer Proceedings in Mathematics & Statistics 401,
https://doi.org/10.1007/978-3-031-15175-0_24

295

triad, the end user never sends back the acknowledgment and the server keeps on waiting for the acknowledgment on that port and therefore the ports are not available to the legitimate users. We call this type of attack Dos, i.e., Denial of Service Attack. When many end users do the same attack together, the server is completely exhausted with so many ports waiting for the acknowledgment request for a long time. We call this Distributed Attack DDoS, i.e., Distributed Denial of Service Attack [1, 4, 5]. By infecting millions of computers with viruses, the botmaster gets the servers of big companies down through this DDoS attack on their signals [6], which has to bear heavy losses to the customer's trust, company's reputation, and business continuity. Therefore, it has been considered as the most dangerous of all the dangers that the Internet faces today. To save the organization from this danger, a lot of extensive research has been done, so that three methods have come out. The first is a method of detecting anomalies in network behavior. In this, bots are identified by changes in network traffic flow due to their network latency and unusual ports. The second is flow-based method. It identifies botnets by using flow-based features such as it identifies a group of packets whose source and destination IP, source and destination port, protocol type, etc., attributes are the same. The third and last method is the graph-based method using either static or dynamic graphs. In this, time-varying graphs are decomposed or those graph communities are monitored over time. And the similarity of the graph is calculated on their degree distribution and diameter. Since botnets encrypt their commands and use sophisticated cloaking techniques, the precious-based method is not better at detecting bots. The existing version of the flow-based method relies on statistical computation of the features of the flow traffic. Also, graph-based methods and all other methods depend on rules. Meaning there should be a predetermined rule against which botnets can be detected. But since botnets are masters at disguise, these methods are not so effective. To identify and avoid this changing behavior of botnets, we used machine learning models in this research to detect botnets so that servers, firewalls, or other security devices could train themselves by using the best model out of all these machine learning models. And identify those botnets and protect themselves from them. Since botnets repeatedly attack in disguise, machine learning and artificial intelligence can predict their changing behavior to some extent, and so in this research, we used many different ML models to determine and find out their ability to detect botnets so that we can identify which model and algorithm would be best to predict this changing behavior and cloaking.

2 Literature Review

Most methods of detecting botnets are focused on their command and control protocols and their structures. But if botnets change their structure and protocol, then these methods are of no use again. Hence, there is a need for a method that can detect the ever-changing network behavior of botnets. Here we have given a brief overview of some of the research done on detecting botnets.

2.1 Botnet Detection Based on Anomaly

The method presented in [7] is a network-based anomaly detection model which takes a snapshot of network behavior with the help of autoencoders to detect any anomaly. It was mainly focused on detecting botnets in IoT devices and it can instantly and accurately detect attacks as and when they are launched from the compromised IoT devices. Next with the help of *DTRAB* [8] method, we can prevent DDoS attacks by detecting those attacks which are encapsulated through encryption and are against the application-level protocol using a distributed detection mechanism which can detect anomalous events as soon as they occur. Anomaly-based detection techniques are based on rules. But a predefined set of rules will be unable to capture the changing behavior of botnets.

2.2 Botnet Detection Based on Flow

The authors of [9] proposed *Hanabot* that detects botnets on the basis of flow records and activities taking place in the host and is very effective in detecting them at an early stage. The evaluation showed low false positives and high accuracy. In [10], the authors proposed *ARGUS* (Audit Record Generation and Utilization System) which could successfully capture botnets by similar behavior and communication patterns of a group of hosts. Flow-based methods depend on the computation of the statistical features of the flow traffic. Due to this, it is able to detect the effect of bots only on individual links. Actually, flow-based methods compare each traffic flow with each other to find out the malicious traffic. By changing the variable length encryption or packet structure, botnets can easily avoid being detected by this method.

2.3 Botnet Detection Based on Graphs

In [11], the authors proposed a graph-based feature clustering that has been suggested to detect botnets. In this method, with the help of self-organizing maps, clusters are created within the network of nodes according to the features. By this model, botnets are isolated into smaller clusters, and larger clusters contain normal nodes. This method was verified using CTU-13 and ISCX dataset and it showed higher efficiency in bot detection despite their varying behavior compared to previously known methods. However, all the detection approaches are based on rules, that is, on the basis of a predetermined rule only botnets can be detected through the graph and hence are not much effective for detecting next-generation botnets.

3 Methodology

3.1 Overview

This research aims to check for the accuracy of different machine learning classifiers in detecting botnet-based intrusion detection. In this research, we have used the following ML classifiers to determine their botnet detection capability:

- *Decision Tree*

It is a hierarchical algorithm that follows divide and conquer strategy and represents the entire dataset into a finite union of local regions defined by some measures. It can also be represented as a set of if then rules.

- *Logistic Regression*

Logistic regression is based on likelihood estimation where we try to learn the probability of given classes with logit transformation of probability which ultimately leads to the sigmoid function which gives us a curve shaped like the letter s and ranging from 0 to 1.

- *Support Vector Machine*

This algorithm creates a decision boundary to segregate n-dimensional dataset into classes so that to enable proper placement of new data points in the future.

- *Gaussian Naive Bayes*

This algorithm is a variant of Naive Bayes algorithm. It is applied on a dataset that follows Gaussian normal distribution and is continuous in nature.

- *K-nearest Neighbors*

This is a clustering algorithm that groups an unlabeled dataset into k clusters. This algorithm works by calculating centroid and associating each cluster with a centroid value iteratively.

3.2 Dataset

The first condition for detecting a botnet is getting real-time traffic. Most of the available datasets are either derived from a simulated environment or are faked traffic, making it impossible for botnets to be properly detected in real time. So here we have used the 11th scenario of CTU-13 dataset [12]. In 2011, CTU University located in the Czech Republic captured botnet traffic and it is known as the CTU-13 dataset. This dataset has 3 class labels – botnet, normal, and background traffic. This dataset is made up of 13 scenarios in which a different malware example was used for each scenario as mentioned in Fig. 1. Searching online resources on the Internet or opening mail, etc., comes under normal traffic. Background traffic was generated so that it could obscure the presence of botnet traffic (refer Table 1 and Fig. 2 to know about the actual traffic distribution). All the data was stored in a pcap (packet capture) file which is processed and converted into netflow files so

that the differentiation between client and server can be done well. Figure 3 shows decomposition of network traffic as per the network protocol. Beginning with the data preprocessing, the CTU-13 dataset has integer, float, object, and categorical values. Some columns such as that of IP addresses have large cardinality and some columns such as that of type of service fields have low cardinality. So to make it compatible with machine learning training and prediction, preprocessing is a must. So for vectorizing the categorical values we have used one hot encoding (refer Fig. 4), then for converting the labels to numbers we have used label encoding. High cardinality columns and those which cannot be converted to numbers were dropped and few columns were scaled from range 0 to 1. Next, we had 15 features in the original dataset and after performing the preprocessing the column number increased to 32 and training any ML model on 32 columns may not give accurate performance so we used various techniques to reduce the number of features and took into account only those features that were really important. Again, now there is a huge imbalance in the dataset as the % of botnet traffic is much lower than that of background traffic (refer Table 1) [13]. And therefore we used different data balancing techniques such as undersampling and oversampling to balance the data. Then finally on the preprocessed and balanced data we applied machine learning models by splitting them into training and test data and finally calculating their accuracies using different python libraries and modules as mentioned in Fig. 4.

Table 2 – Characteristics of the botnet scenarios. (CF: ClickFraud, PS: Port Scan, FF: FastFlux, US: Compiled and controlled by us.)

Id	IRG	SPAM	CF	PS	DDoS	FF	F2P	US	HTTP	Note
1	✓		✓							
2	✓	✓	✓							
3	✓		✓							
4	✓			✓				✓		
5		✓		✓	✓				✓	UDP and ICMP DDoS. Scan web proxies.
6				✓						Proprietary CAC. RDP.
7				✓					✓	Chinese hosts.
8				✓						Proprietary CAC. Net-BIOS, STUN.
9	✓	✓	✓	✓						
10	✓				✓			✓		UDP DDoS.
11	✓				✓			✓		ICMP DDoS.
12							✓			Synchronization.
13		✓		✓					✓	Captcha. Web mail.

Fig. 1 Dataset scenario description.

4 Results and Discussion

Here in Table 2 we can see the accuracy, precision and recall for all the ML classifiers that we used to check their accuracy in detecting botnets. The most notable among all is the decision tree classifier with highest accuracy and highest precision. This has a much better accuracy than many traditional approaches as we discussed because those approaches were dependent on a predetermined set of rules which could detect only certain types of attacks, say suppose attack using http protocol or spamming attacks on web. They cannot detect attacks where botnets

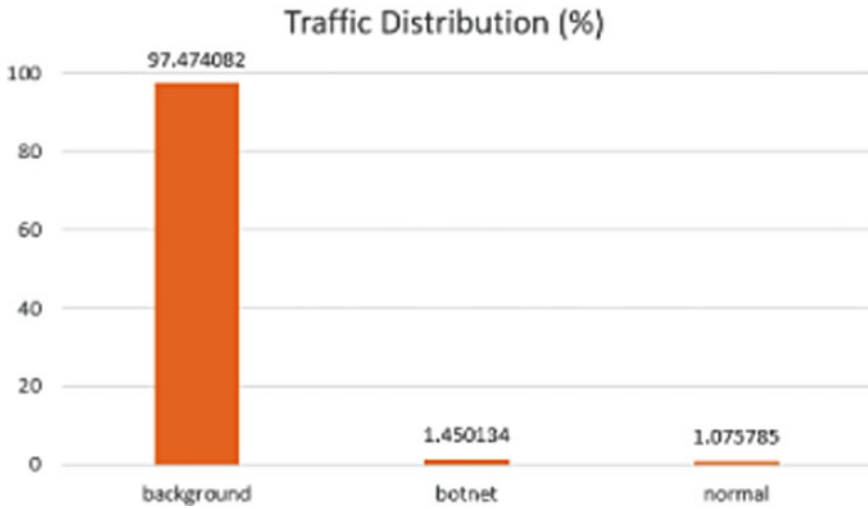


Fig. 2 Traffic frequency distribution

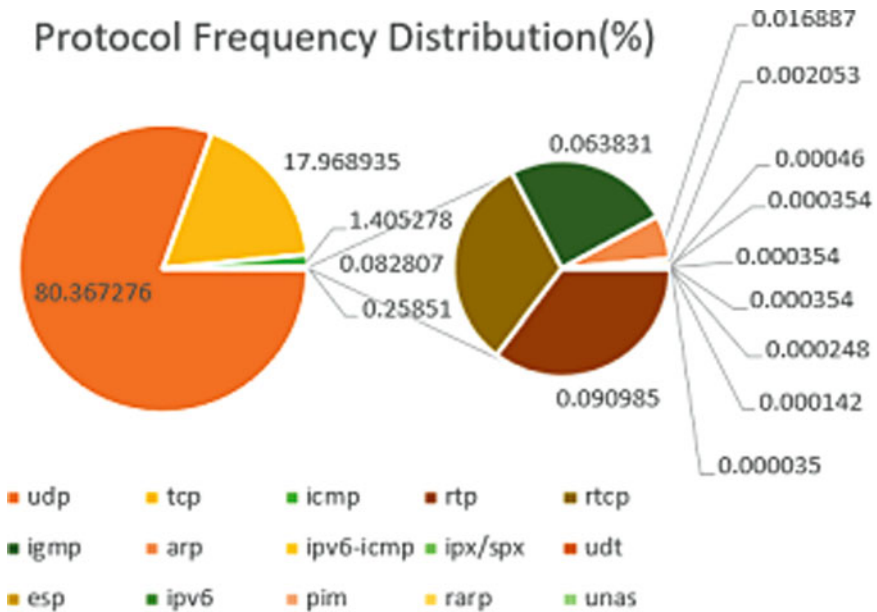


Fig. 3 Protocol frequency distribution

encrypt their commands or change their network flow behavior and hence we need ML classifiers which detect botnets based on prediction with learning on a real-time dataset. Also the least effective classifier is the Gaussian Naive Bayes classifier with an accuracy of only 72% and a recall lesser than 0.7 which specifies

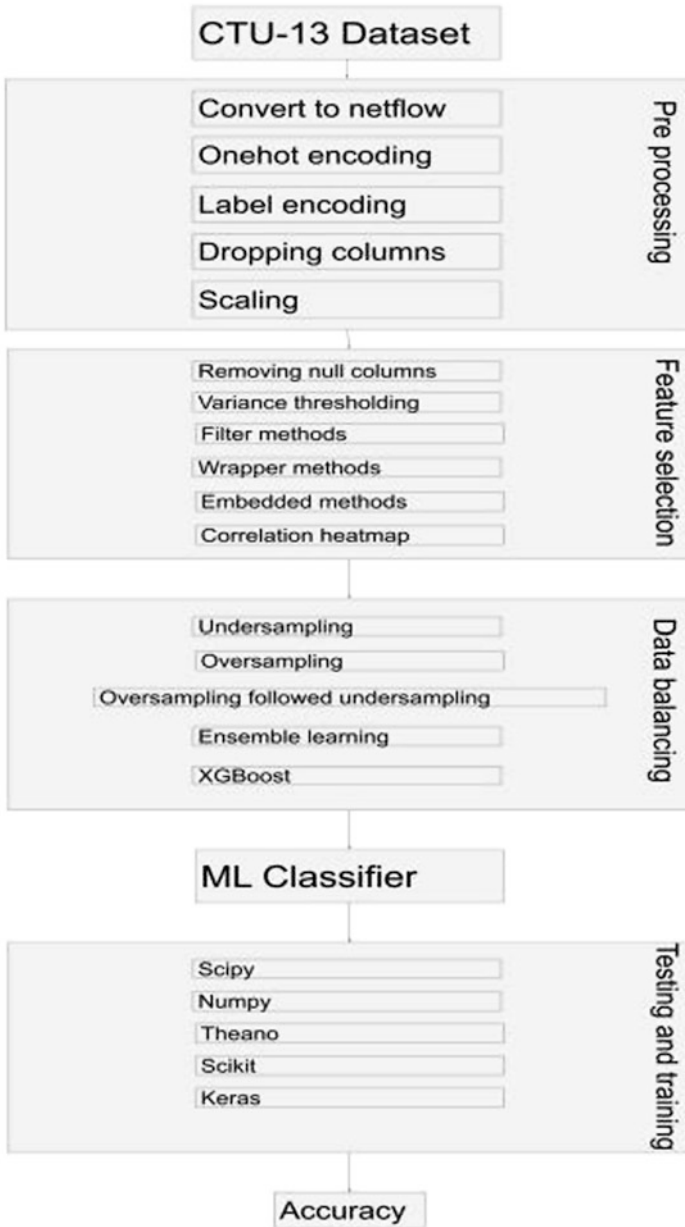


Fig. 4 Proposed model

Table 1 Dataset diversity distribution.

Scenario	Background flows (%)	Botnet flows (%)	Normal flows(%)	Total flows
1	97.47	1.41	1.07	2,824,636
2	98.33	1.15	0.5	1,808,122
3	96.94	0.561	2.48	4,710,638
4	97.58	0.154	2.25	1,121,076
5	95.7	1.68	3.6	129,832
6	97.83	0.82	1.34	558,919
7	98.47	1.5	1.47	114,077
8	97.32	2.57	2.46	2,954,230
9	91.7	6.68	1.57	2,753,884

its ineffectiveness in detecting botnets as compared to other machine learning classifiers. And hence we can compare and contrast the relative effectiveness of different machine learning classifiers using Table 2.

Figure 5 shows the probability distribution curve for the normal (denoted in green) and botnet (denoted in red) traffic as classified by decision tree. According to the graph from -2 to $+1$ and again from $+2$ to $+5$ on x-axis and from $+2$ to $+10$ on y-axis comprises the region of normal traffic while the remaining region is botnet intruded traffic. Future data points lying on the green region will be marked as normal traffic and those lying on the red region will be marked as botnet intruded traffic. Also we can see some false positives and false negatives in the graph.

Figure 6 shows the probability distribution curve for the normal (denoted in green) and botnet (denoted in red) traffic as classified by K-nearest neighbor. According to the graph the green region which is from $+2$ to $+4$ on x-axis and $+4$ to $+6$ on y-axis and again more from -2 to $+1$ on x-axis and $+2$ to $+10$ on y-axis comprises of normal traffic and the remaining red region comprises of botnet intruded traffic. Here also we can see some false positives and false negatives.

Figure 7 shows the probability distribution curve for the normal (denoted in green) and botnet (denoted in red) traffic as classified by logistic regression. According to this classifier there is no separate region for normal traffic and almost all traffic are found to be botnet intruded leaving no space for normal traffic and some false negatives.

Figure 8 shows the probability distribution curve for the normal (denoted in green) and botnet (denoted in red) traffic as classified by Gaussian Naive Bayes. Here we can see a vertical parabolic green region with center at (0.5) consisting of all normal traffic and the remaining red region comprising all botnet intruded traffic. This graph also contains some false negatives and few false positives.

Figure 9 shows the probability distribution curve for the normal (denoted in green) and botnet (denoted in red) traffic as classified by support vector machine using a linear kernel. Here again there is no region specified by the classifier for normal traffic and therefore according to the classifier all the traffic will be classified as botnet intruded according to the dataset with few false negatives.

Also a comparative analysis of all the different machine learning classifiers we used is given in the form of a column bar graph in Fig. 10 and we can infer that among all the classifiers decision tree has better accuracy, precision, and recall. In [1], the accuracy received from the decision tree classifier was 99.89% with 100% precision and 100% recall. However, it was tested using parts of the UNSW-NB15 dataset which is very old.

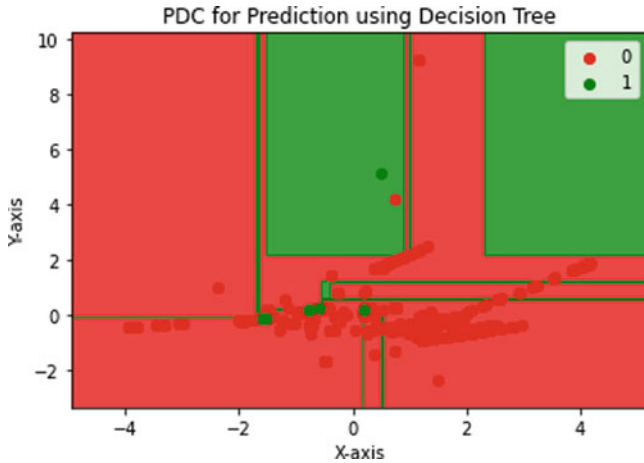


Fig. 5 PDC for prediction using decision tree classifier

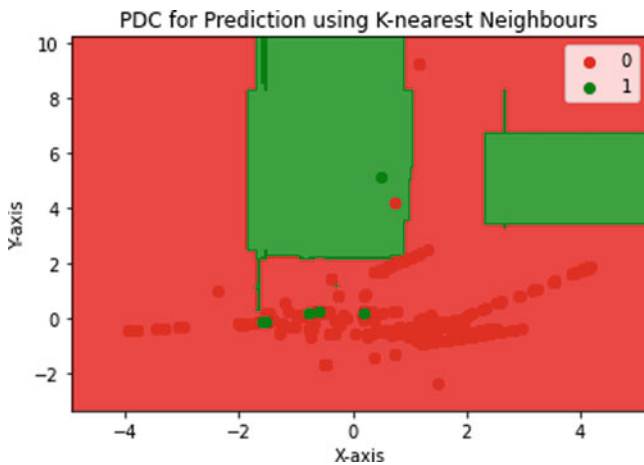


Fig. 6 PDC for prediction using K-nearest neighbors classifier

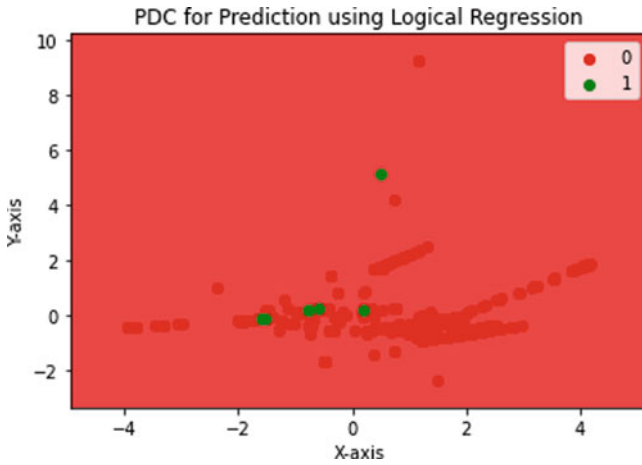


Fig. 7 PDC for prediction using logistic regression classifier

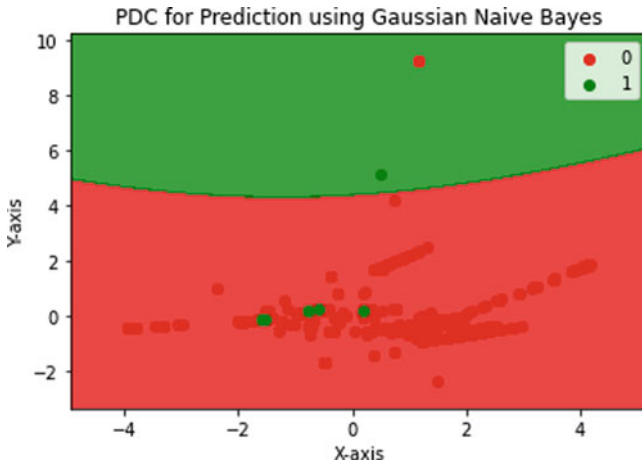


Fig. 8 PDC for prediction using Gaussian Naive Bayes classifier

5 Conclusion

Nowadays, life without the Internet cannot be imagined. While the Internet offers immense benefits, it also is subject to fatal security risks. While the Internet offers immense benefits, it also is subject to fatal security risks and among all, particularly that of botnets, it is not an easy task to detect botnets. Even if a single bot is detected in the same network, it is very difficult to detect the entire botnet. Among all methods that are proposed to detect botnets, classification and clustering ML models which are based on supervised and unsupervised approaches respectively are the best. As it is important to focus on changing behavior of botnets instead of using a

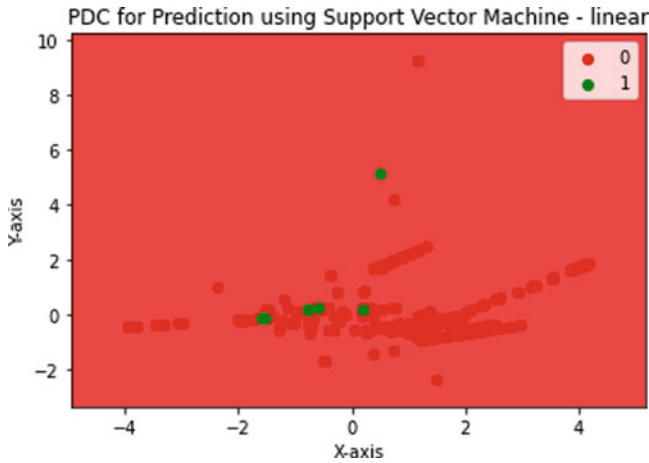


Fig. 9 PDC for prediction using support vector machine classifier with a linear kernel

A Comparison of different ML classifiers

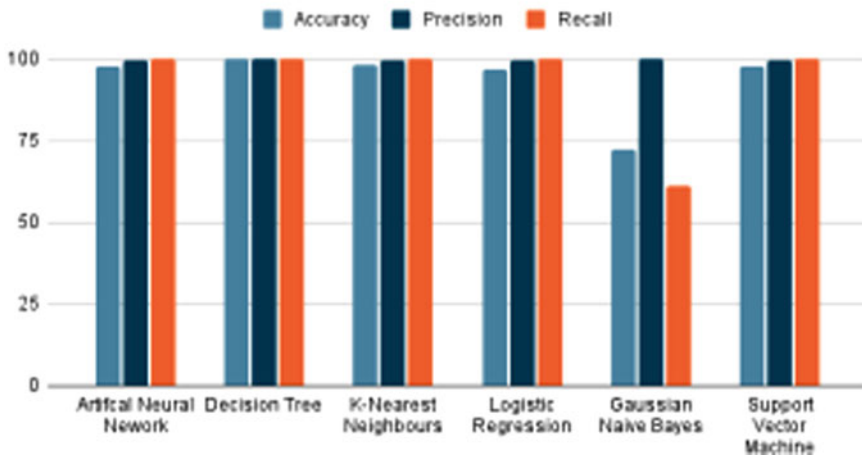


Fig. 10 A comparison of different ML classifiers used

particular value or range of features, hence machine learning shows higher accuracy in detecting botnets than traditional approaches. Here in this research we verified the detection accuracy of some of the popular ML models using CTU-13 dataset which is the largest dataset that contains bot-labeled nodes along with normal and background traffic. And among all the six different machine learning classifiers we tested, decision tree was found to be highly accurate with an accuracy of 99.90%. But as a matter of fact, neither the traditional method nor the ML models were able to achieve 100% accuracy. Therefore, there has to be a lot more research to put a stop to organizations losing reputation and increasing Internet security vulnerabilities.

Table 2 Accuracy, precision, and recall for different ML classifiers

S. No.	ML models	Accuracy (%)	Precision	Recall
1.	Artificial neural network	98.01	99.880881477069	100.0
2.	Decision tree	99.91	99.960254372019	99.980123235937
3.	K-nearest neighbors	98.19	99.880881477069	100.0
4.	Logistic regression	96.97	99.861055974593	100.0
5.	Gaussian Naive Bayes	72.25	100.0	61.1608030212681
6.	Support vector machine	98.04	99.861055974593	100.0

References

1. Mustafa Alshamkhany and et.al., Botnet Attack Detection using Machine Learning, 2020 14th International Conference on Innovations in Information Technology (IIT), 17-18 Nov. 2020. DOI: 10.1109/IIT50501.2020.9299061
2. S. S. C. Silva, R. M. P. Silva, R. C. G. Pinto, and R. M. Salles, "Botnets: A survey," *Comput. Networks*, vol. 57, no. 2, pp. 378–403, 2013, doi: <https://doi.org/10.1016/j.comnet.2012.07.021>.
3. S. Amina, R. Vera, T. Dargahi, and A. Dehghantaha, "A Bibliometric Analysis of Botnet Detection Techniques."
4. A. Lohachab and B. Karambir, "Critical Analysis of DDoS — An Emerging Security Threat over IoT Networks," vol. 3, no. 3, 2018.
5. R. Hallman, J. Bryan, G. Palavicini, J. Divita, and J. Romero-mariona, "IoDDoS — The Internet of Distributed Denial of Service Attacks : A Case Study of the Mirai Malware and IoT-Based Botnets IoDDoS —The Internet of Distributed Denial of Service Attacks A Case Study of the Mirai Malware and IoT-Based Botnets," no. April 2017.
6. A. Al-nawasrah and S. Arabia, A Survey of Fast Flux Botnet Detection With Fast Flux Cloud Computing, vol. 10, no. 3, 2020.
7. Y. Meidan et al., "N-BaIoT-Network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Comput.*, vol. 17, no. 3, pp. 12–22, 2018, doi: <https://doi.org/10.1109/MPRV.2018.03367731>.
8. Fadlullah ZM, Taleb T, Vasilakos AV, Guizani M, Kato N. DTRAB: combating against attacks on encrypted protocols through traffic-feature analysis. *IEEE/ACM Trans Netw (TON)*. 2010;18(4):1234–47.
9. S. Almutairi, S. Mahfoudh, S. Almutairi, and J. S. Alowibdi, "Hybrid Botnet Detection Based on Host and Network Analysis," *J. Comput. Networks Commun.*, vol. 2020, 2020, doi: <https://doi.org/10.1155/2020/9024726>.
10. Argus (audit record generation and utilization system); 2016. <http://www.qosient.com/argus> Accessed 21 Feb 2016.
11. S. Chowdhury et al., "Botnet detection using graph-based feature clustering," *J. Big Data*, vol. 4, no. 1, Dec. 2017, doi: 10.1186/s40537-017-0074-7.
12. S. Garcia et al., "An empirical comparison of botnet detection", in *Computers and Security*, vol. 45, pp. 100–123, Sep. 2014, doi: <https://doi.org/10.1016/j.cose.2014.05.011>
13. Vishwakarma, Anand Ravindra, "Network Traffic Based Botnet Detection Using Machine Learning"(2020). Master's Projects.917. DOI: <https://doi.org/10.31979/etd.4nd6-m6hp>

Reducing Peak Electricity Demands of a Cluster of Buildings with Multi-Agent Reinforcement Learning



Manoj Kumar Balwant, Sai Rohan Basa, and Rajiv Misra

Keywords Demand side management · Demand response · Building energy management system · CityLearn · Multi-agent reinforcement learning

1 Introduction

Affordable and clean energy is one of the 17 sustainable development goals identified by the United Nations that aims to ensure access to affordable, reliable, sustainable, and modern energy for all [3]. One out of ten people still lacks electricity in rural areas of the developing world. At the same time, today, energy is the main contributor to climate change which contributes 73% to human-caused greenhouse gasses. Population growth, rapid urbanization, and increased ownership of personal appliances are leading factors in increasing energy demand. This phenomenon is more noticeable during hot and humid climates where more use of air conditioning makes electricity demand and prices high. This results in peaks in electricity demands and sometimes causes electricity blackouts in many parts of the country.

Relatively low utilization of generation and transmission networks means that there is significant scope for demand-side energy management (DSM) that could offer peak shaving and grid stability. The DR program is one promising aspect of DSM to optimize peak electricity demand. It relies on consumers to either change their consumption behavior during peak hours or change the time of energy-intensive activities from peak hours to off-peak hours or use on-site renewable energy during peak times [8]. The DR programs control electricity demands through different mechanisms that can be classified into two main categories: price-based and incentives-based. In the price-based program, grid operators indirectly control customers' appliances by varying electricity prices that eventually affect customers'

M. K. Balwant (✉) · S. R. Basa · R. Misra

Department of Computer Science and Engineering, Indian Institute of Technology, Patna, India
e-mail: 1801cs14@iitp.ac.in; rajivm@iitp.ac.in

electricity bills. Due to the time-varying prices, customers make smart decisions by shifting some loads from high price periods to low price periods. This offers customers saving some money while the grid operator benefits from reduced peak demand. However, the price-based programs sometimes cause undesirable peaks of demand during low electricity prices. The incentive-based DR program pays incentives to customers (end users) to shift their electricity usage voluntarily during peak demand periods or power supply emergencies as requested by grid operators. Generally, the customers voluntarily change their consumption patterns. However, in some cases, there is a penalty on customers for not meeting the requirements.

The increase of distributed energy storage systems (i.e., batteries, heat pumps, water heaters) and distributed energy generation resources (i.e., PV panels) in buildings provides an opportunity to effectively reduce peak energy demands. Distributed energy sources (DESS) include distributed generators (DGs), renewable energy sources (RESs), and/or energy storage systems (ESSs) such as batteries and EVs. As time advances, buildings become smarter and more independent in terms of their own DES. The recent advances and applications in microgrid (MG) technologies integrate DES such as DGs, RES, and ESS into the power grid. A microgrid consists of distributed energy sources (DESS) with a low-voltage network to meet the local power demand and decentralization of power supply. An MG exploits DES and can operate in parallel to the main grid to satisfy high variations of load demands. It is anticipated that the future smart grid will be dominated by many autonomous MGs collaborating with each other [7]. However, the stochastic behavior of these DES still results in random variations of load demands on both the supply and demand sides. The problem of uncertainty of the load demand and DES can be addressed through the real-time scheduling of MG with advanced control strategies.

The peak energy demands can be significantly reduced by advanced control strategies that offer energy flexibility in buildings by exploiting DES and DR programs. Significant advances have been made in Reinforcement Learning (RL) in recent years due to the tremendous success of various sequential decision-making problems, including games of Go and poker, robotics, and autonomous driving. These applications involve the participation of more than one agent, which naturally falls under the purview of the multi-agent RL (MA-RL) domain [16]. An MA-RL-based controller can effectively coordinate DES as well as energy use among different buildings in a Smart City to flatten high peaks of electricity demand. This chapter used MA-RL-based decentralized controller to explore coordinated energy management for reducing the peak electricity demand of a cluster of buildings.

2 Related Work and Contribution

A recent review [8] reported that 70% of papers have an optimization objective to reduce the energy cost of a cluster of buildings by incorporating price-based DR program, and they achieved an average of approximately 18% reduction in total

energy cost across different studies. The past five studies shifted certain appliances and charged ESS to peak renewable energy generation and discharged ESS during peak electricity demand. They reported a reduction of approximately 20% in total energy consumption. Several studies used classical programming methods, including linear programming, integer linear programming, nonlinear programming, and mixed-integer linear programming, for cost saving at both building and district levels for a cluster of buildings [8]. Unlike linear programming to solve the unique optimal solution, heuristic algorithms converge faster to a solution but may not be an absolute optimal solution. Heuristic algorithms utilized in achieving energy cost and consumption savings include particle swarm optimization, simulated annealing, and genetic algorithm. Model predictive control (MPC) requires a linear/nonlinear model of buildings and uses optimization to find solutions to optimal control problems while satisfying a set of constraints. Mariano-Hernández et al. [11] reported several studies on building energy management using MPC, which includes ensuring thermal comfort with minimal energy utilization, predictive whole building heat and moisture, optimal control of cooling and heating activities, reducing energy demand costs of a group of buildings connected to heat pumps, and minimizing the energy consumption and the energy cost of the building HVAC. However, the greatest challenge in MPC is to design a detailed model of buildings that can describe the building energy dynamics [9]. The approximate model cannot achieve what is achievable in reality.

Recently, MA-RL-based controllers have been used in coordinated control of DES for building energy management. The peak energy demands of a cluster of buildings can be addressed with MA-RL through a centralized controller or decentralized controllers. The centralized controller considers information about the current state of all buildings in the cluster, while a decentralized controller works on a single building [4]. Most studies in the literature presented single agent-based RL that works greedily and independently of other buildings, ignoring opportunities provided by coordinated control to reduce the peaks [12]. A recent work [13] exploited a “Soft Actor Critic” (SAC)-based centralized (single-agent) RL controller that manages thermal storage of a cluster of buildings to flatten cluster load profile while optimizing energy consumption. The proposed controller considers a static electricity price through the design of a price-sensitive reward function. The controller is trained/tested on CityLearn, an OpenAI gym environment, and compared against a manually optimized rule-based controller. Another study [2] used a single agent centralized RL controller along with utilizing an incentive-based DR program to control thermal storage of multiple buildings to optimize the energy use. Kathirgamanathan et al. [10] also applied a single-agent centralized RL based on SAC on the CityLearn environment for optimizing peak demands. It has limitations of the manual reward shaping that limits the generalization ability of the RL controller. Vazquez-Canteli et al. [15] introduced an MA-RL controller called MARLISA for improving peak demands in microgrids, and it is evaluated on the CityLearn environment. This approach utilizes the concept that each agent predicts their own future electricity consumption and shares this information with another agent following a leader–follower schema. Another research [5] proposed an MA-

RL controller called *MARLISA_{DACC}* for improving peak energy demands based on decentralized actor–critic RL method MARLISA [15] but used a centralized critic. They demonstrated their decentralized actors and centralized critic based architecture on the CityLearn environment for coordinated demand response. This work explores MA-RL-based decentralized controller to facilitate coordinated energy management for reducing the peak electricity demand of a cluster of buildings. A novel reward function is proposed that is sensitive to the DR program. The MA-RL controller is evaluated on a new benchmark environment CityLearn [14].

3 Simulation Environment: CityLearn

We have evaluated our method on CityLearn [14], an OpenAI Gym environment. It contains five different climate zones, each with nine buildings. The buildings have diverse load profiles, which include DHW, chilled water (for sensible cooling and dehumidification), solar photovoltaic arrays, air to water heat pumps, and electric heaters. The environment contains the precomputed energy load of each building on an hourly basis which includes heating loads, cooling loads, non-shiftable appliances loads, and solar generation. The environment provides simulated hourly data of buildings and allows the controller to manage the charging and discharging of thermal storage devices installed in buildings. The cooling energy is supplied by air to water heat pumps while heating energy is supplied by electric heaters. In addition, some buildings are installed with PV systems to produce energy on-site. The environment is specifically designed for the standardized evaluation of different RL models for the reduction in peak energy demands in smart cities. Users can select up to 30 different state variables, which include current weather conditions, future weather forecasts, the state of charge of the different energy storage devices, and solar radiation forecasts. The environment allows us to control multiple thermal storage devices, including chilled water tanks, hot water tanks, and batteries within a heterogeneous cluster of buildings. It offers the possibility to easily implement both centralized and decentralized MA-RL controllers. The proposed MA-RL controller is tested on the CityLearn environment considering the maximum and minimum charging rate of energy storage devices and ensuring the cooling, and DHW demands are always satisfied.

4 Design of the Multi-Agent RL Controller

In this work, we used an MA-RL controller where each building has an individual agent that shares limited information with other agents (as shown in Fig. 1). We used soft actor–critic (SAC) architecture [6, 15] for centralized training and decentralized execution. Each agent has one actor and one critic. The actor is trained using its

own state and action. While the critic is trained using the state and action of all agents in a centralized way. It incorporates three key components: an actor–critic architecture that employs a separate network for policy and value function, an off-policy formulation that reuses past experiences along with the benefit of learning from fewer samples, and entropy maximization to enable efficient exploration with stable training. The actor–critic architecture consists of an actor and a critic network where the actor-network maps the current state of an environment to the probability values of each action in the action-space. In contrast, the critic network evaluates these actions by mapping them to Q-values from its state.

The SAC algorithm learns three different functions: the actor (policy), the critic (soft Q-function), and the value function. The value function is given by Eq. 1.

$$\begin{aligned} V(s_t) &= \mathbb{E}_{a_t \sim \pi_\theta} [Q(s_t, a_t) - \alpha \log \pi_\theta(a_t | s_t)] \\ &= \mathbb{E}_{a_t \sim \pi_\theta} [Q(s_t, a_t)] + \alpha \mathbb{E}_{a_t \sim \pi_\theta} [\log \pi_\theta(a_t | s_t)] \\ &= \mathbb{E}_{a_t \sim \pi_\theta} [Q(s_t, a_t)] + \alpha H \end{aligned} \quad (1)$$

In the above value function, $H \geq 0$ is the Shannon entropy of the policy π_θ . The policy with zero entropy leads to deterministic policy, and policies with non-zero entropies allow more randomized action selection. The goal of the SAC agent is to learn the optimal stochastic policy π^* that maximizes the expected long-term reward and long-term entropy as given by Eq. 2.

$$\pi^* = \arg \max_{\pi_\theta} \sum_{t=0}^T \mathbb{E}_{(s_t, a_t) \sim \rho_\pi} [r(s_t, a_t) + \alpha H(\pi_\theta(\cdot | s_t))] \quad (2)$$

where $(s_t, a_t)_\pi$ is the state–action pair that is sampled from the agent’s policy and $r(s_t, a_t)$ is the reward corresponding to a state–action pair. The added entropy term allows the agent to behave as randomly as possible while maximizing the returns. The critic networks update their parameters by minimizing the expected error J_Q between the predicted Q-values and the ones calculated through iteration.

$$J_Q = \mathbb{E}_{(s_t, a_t) \sim D} \left[\frac{1}{2} \left(Q_\theta(s_t, a_t) - \left(r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim p} [V_{\bar{\theta}}(s_{t+1})] \right) \right)^2 \right] \quad (3)$$

Here, $\alpha \in (0, 1)$ is a hyperparameter called temperature that controls the importance of the entropy term. The value of $\alpha \in 1$ leads to uniformly random behavior. While $\alpha \in 0$ would ignore the entropy term completely, the agent would focus on maximizing return without exploration. This leads to an almost deterministic policy. The temperature α used in this work is set to a constant value of 0.2. In the following subsections, the state- and action-space are presented, along with a description of the reward function design.

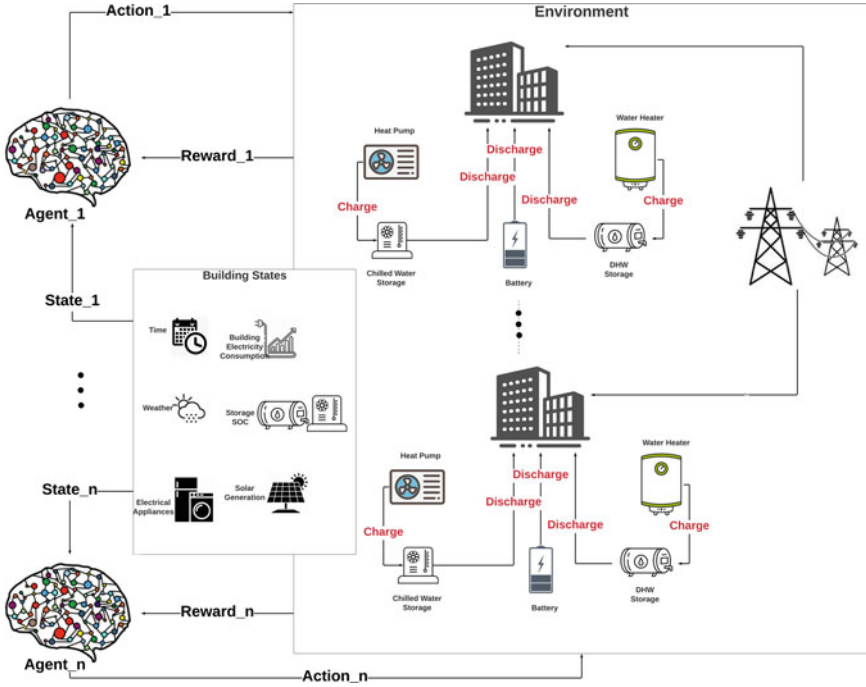


Fig. 1 MA-RL controller for reducing peak electricity demand of a cluster of buildings

4.1 Action-Space

In this work, we are dealing with a cluster of nine buildings where each building contains at most three types of storage to be controlled (as shown in Fig. 1). This results in three actions that have different targets to be achieved through controlled actions. The first type of action is controlling the storage of domestic hot water that is charged by an electric heater or discharged to fulfill the domestic hot water demand of a residential building. The second action is controlling the storage of chilled water that is charged by the heat pump to store chilled thermal energy or discharged to fulfill the cooling load of residential buildings. The third action is controlling the storage of electricity that is charged to store electricity or discharged to release electricity at an appropriate time to meet the electrical demand of residential buildings.

Each building has different storage capacities of a chilled water tank, hot water tank, and battery. The action-spaces are normalized values between -1 and 1 , where -1 represents fully discharged, and 1 represents fully charged. Considering the fact that full charging or discharging is not possible in a single time step (set to 1 hour), the action-spaces are bounded between intervals $[-1/C, 1/C]$, where C is the thermal energy capacity. For instance, to implement more realistic charging and discharging,

if the capacity C of a storage device is equal to 3, then its action-space is bounded between $[-0.33, 0.33]$, considering the fact that complete charging and discharging lasts for three hours.

4.2 State-Space

The environment is observed by RL agents to select among available actions. In this work, the environment is represented by the state-space, which consists of mainly three components: weather, district, and building states. Outdoor temperature and direct solar radiation are selected under the weather component as they strongly influence building loads for cooling. The district component contains variables that have the same values for all buildings over time. It includes months, days (from 1 to 8, which includes holidays), hours (from 1 to 24), and daylight savings status. The building component contains variables related to electricity production (photovoltaic system) and consumption of the buildings (non-shiftable load). They are specific to each building because each building has a different energy production capacity (PV) and consumption pattern. The different variables selected for state-space are shown in Table 1.

Table 1 State-space variables

Variable group	Variable	Unit
Weather	Outdoor temperature	°C
	Outdoor temperature forecast (6 h, 12 h, 24 h)	°C
	Indoor temperature	°C
	Indoor relative humidity	%
	Outdoor relative humidity	%
	Outdoor relative humidity forecast (6 h, 12 h, 24 h)	%
	Direct solar radiation	W/m ²
	Direct solar radiation forecast (6 h, 12 h, 24 h)	W/m ²
	Diffuse solar radiation	W/m ²
	Diffuse solar radiation forecast (6 h, 12 h, 24 h)	W/m ²
District	Total load	kW
	Total load	KW
	Hour of day	∈ {1 – 24}
	Day of the week	∈ {1 – 8}
	Month	∈ {1 – 12}
Building	Non-shiftable load	KW
	Solar generation	W/m ²
	Cooling storage SOC	∈ [0,1]
	DHW SOC	∈ [0,1]
	Daylight savings status	∈ {0, 1}

4.3 Reward

In this work, the reward structure is designed based on DR program. Inspired by the Austin (Texas) electricity tariffs [1], the three different cases are considered while designing the reward function: off-peak, mid-peak, and on-peak period. These off-peak, mid-peak, and on-peak periods are not based on time; instead, it is based on the net electricity consumption of the district (all buildings), which represents the lowest, moderate, and highest electricity demand of buildings. The reward structure for each building in each time step is governed by Eq. 4.

$$R_{MA-RL}^i = \begin{cases} E_i^2, & \text{if } E_{district} < offPeakLoad_{thres} \text{ and } E_i < 12, \\ -1 * E_i^2, & \text{if } offPeakLoad_{thres} \geq E_{district} < onPeakLoad_{thres} \text{ and } E_i \geq 12 \\ -1 * E_i^3, & \text{if } E_{district} \geq onPeakLoad_{thres} \text{ and } E_i \geq 22 \end{cases} \quad (4)$$

where $E_{district}$ is the net electricity demand of the district and E_i is the net electricity demand of building i at each time step. The $offPeakLoad_{thres}$ and $onPeakLoad_{thres}$ are the off-peak and on-peak electricity demands of the district, whose values are 110 KW and 200 KW, respectively.

We give three different rewards to the agent based on the type of peak period. We have identified two thresholds $offPeakLoad_{thres}$ and $onPeakLoad_{thres}$ to determine peak periods. During the off-peak period, i.e., when the electricity demand of entire buildings (or the district) is less than 110 KW and if a building i is consuming less electricity, i.e., less than 12, then we give a positive reward of the square of the energy demand of the building i . During the mid-peak period, i.e., when the energy demand of the district is greater than 110 KW but less than 200 KW and if a building i is consuming more electricity, i.e., more than 12 KW, then we give a negative reward of the square of the energy demand of the building i . During the on-peak period, i.e., when the energy demand of the district is greater than 200 and if a building i is consuming significantly more electricity, i.e., more than 22 KW, then we give a negative reward of the cube of the energy demand of the building i .

5 Results

We compared the performance of the MA-RL with the proposed reward function R_{MA-RL} with that of MARLISA $R_{MARLISA}$ [15]. Figure 2 illustrates the net electricity consumption of the entire district (a set of 9 buildings) of climate zone 5 over a period of 4 years for these two methods. In the plot, the blue line shows the electricity demand without PV generation and without any storage. The orange line represents the electricity demand with PV generation but without storage. The green line and red lines represent electricity demand with PV generation and with

the storage control method based on the reward function $R_{MARLISA}$ and R_{MA-RL} (proposed reward), respectively. In the plot, the energy consumption during the initial phase is more in our R_{MA-RL} method which is represented by the red line. This is because the model is taking random actions. After learning the environment well in the initial stage, our MA-RL method is performing better than MARLISA, which is represented by the green line in terms of reducing peak electricity demands.

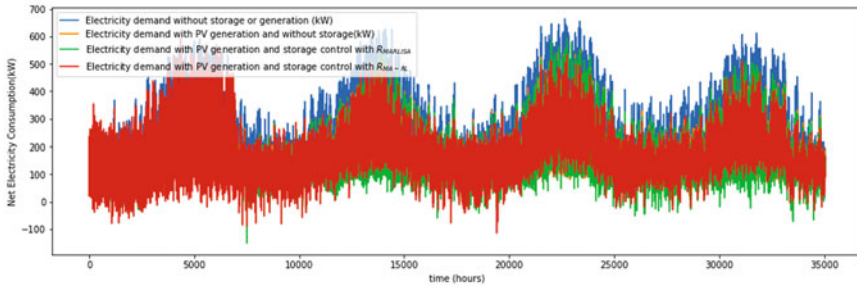


Fig. 2 Net electric consumption of the entire district in climate zone 5 over a period of 4 years

Figure 3 shows a comparative representation of net electricity consumption of the district in the summer season of the last year for the two methods $R_{MARLISA}$ and our R_{MA-RL} . The orange line shows the electricity demand with PV generation and without storage, and it handles the peak demands better than the electricity demand without any PV generation or any storage, as represented by the blue line. The electricity demand with PV generation and with storage control using $R_{MARLISA}$ or R_{MA-RL} (represented by green and red lines, respectively) optimizes the peak demands better compared to the electricity demand with PV generation and without any storages. From the graph, it can be seen that the electricity demand and peaks are less in our R_{MA-RL} method (represented by the red line) when compared to $R_{MARLISA}$ (represented by the green line). Figure 4 shows the comparison of net electricity consumption of the district in the winter season of the last year for the two methods. From the graph, it is clear that in the winter season also, the electricity demand with PV generation and with storage control using our R_{MA-RL} method, which is represented by the red line, optimizes the peak demands better when compared to the baseline method $R_{MARLISA}$, which is represented by the green line.

6 Conclusion

In this chapter, we explored the possibility of incorporating a DR program into an MA-RL controller to flatten the peak demands of a cluster of buildings. Furthermore, an MA-RL controller based on SAC is employed to address the

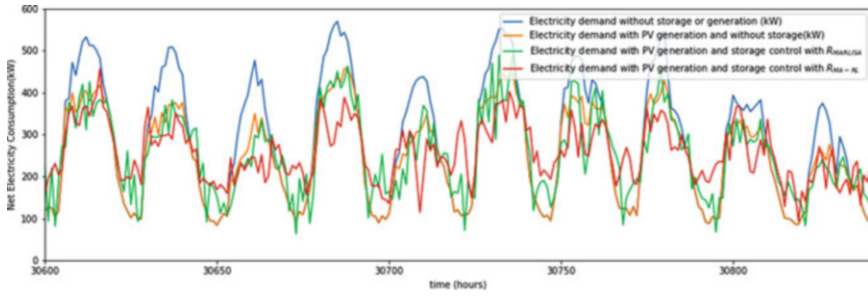


Fig. 3 Net electric consumption of the entire district in climate zone 5 over the last year summer season

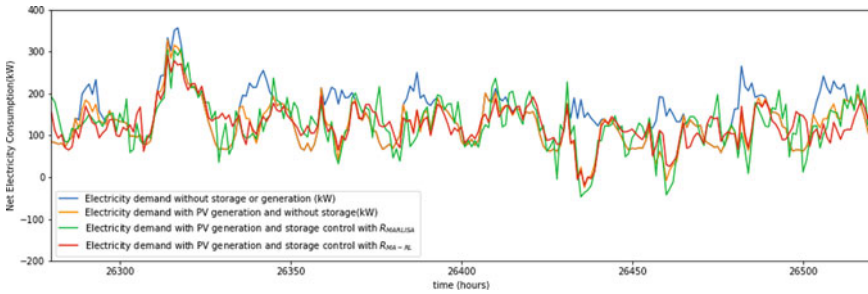


Fig. 4 Net electric consumption of the entire district in winter season over last year in climate zone 5

coordinated control of DES with the DR program. We presented a novel reward function based on the DR program and investigated the application of the MA-RL controller to improve peaks in energy demands. Our MA-RL controller consists of multiple decentralized agents, and they effectively coordinate control of DES in buildings for managing supply and demand. The performance of the proposed reward function is evaluated against the baseline RL MARLISA. Our results show that the proposed MA-RL controller with a novel reward function outperforms the baseline method. Our future work focuses on incorporating a dynamic pricing-based DR program into the MA-RL controller.

References

1. MAustin Energy electricity tariff pilot programs. <https://austinenergy.com/ae/>. Accessed: 2010-09-30
2. Deltetto, D., Coraci, D., Pinto, G., Piscitelli, M.S., Capozzoli, A.: Exploring the potentialities of deep reinforcement learning for incentive-based demand response in a cluster of small commercial buildings. *Energies* **14**(10), 2933 (2021)
3. Desa, U., et al.: Transforming our world: The 2030 agenda for sustainable development (2016)

4. Dhamankar, G., Vazquez-Canteli, J.R., Nagy, Z.: Benchmarking multi-agent deep reinforcement learning algorithms on a building energy demand coordination task. In: Proceedings of the 1st International Workshop on Reinforcement Learning for Energy Management in Buildings & Cities, pp. 15–19 (2020)
5. Glatt, R., Silva, F.L.d., Soper, B., Dawson, W.A., Rusu, E., Goldhahn, R.A.: Collaborative energy demand response with decentralized actor and centralized critic. In: Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation, pp. 333–337 (2021)
6. Haaroja, T., Zhou, A., Abbeel, P., Levine, S.: Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: International conference on machine learning, pp. 1861–1870. PMLR (2018)
7. Ji, Y., Wang, J., Xu, J., Fang, X., Zhang, H.: Real-time energy management of a microgrid using deep reinforcement learning. *Energies* **12**(12), 2291 (2019)
8. Kaspar, K., Ouf, M., Eicker, U.: A critical review of control schemes for demand-side energy management of building clusters. *Energy and Buildings* **257**, 111,731 (2022)
9. Kathirgamanathan, A., De Rosa, M., Mangina, E., Finn, D.P.: Data-driven predictive control for unlocking building energy flexibility: A review. *Renewable and Sustainable Energy Reviews* **135**, 110,120 (2021)
10. Kathirgamanathan, A., Twardowski, K., Mangina, E., Finn, D.P.: A centralised soft actor critic deep reinforcement learning approach to district demand side management through CityLearn. In: Proceedings of the 1st international workshop on reinforcement learning for energy management in buildings & cities, pp. 11–14 (2020)
11. Mariano-Hernández, D., Hernández-Callejo, L., Zorita-Lamadrid, A., Duque-Pérez, O., García, F.S.: A review of strategies for building energy management system: Model predictive control, demand side management, optimization, and fault detect & diagnosis. *Journal of Building Engineering* **33**, 101,692 (2021)
12. Pinto, G., Piscitelli, M.S., Vázquez-Canteli, J.R., Nagy, Z., Capozzoli, A.: Coordinated energy management for a cluster of buildings through deep reinforcement learning. *Energy* **229**, 120,725 (2021)
13. Pinto, G., Piscitelli, M.S., Vázquez-Canteli, J.R., Nagy, Z., Capozzoli, A.: Coordinated energy management for a cluster of buildings through deep reinforcement learning. *Energy* **229**, 120,725 (2021)
14. Vázquez-Canteli, J.R., Dey, S., Henze, G., Nagy, Z.: CityLearn: Standardizing research in multi-agent reinforcement learning for demand response and urban energy management. arXiv preprint arXiv:2012.10504 (2020)
15. Vazquez-Canteli, J.R., Henze, G., Nagy, Z.: Marlisa: Multi-agent reinforcement learning with iterative sequential action selection for load shaping of grid-interactive connected buildings. In: Proceedings of the 7th ACM international conference on systems for energy-efficient buildings, cities, and transportation, pp. 170–179 (2020)
16. Zhang, K., Yang, Z., Başar, T.: Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control* pp. 321–384 (2021)

Virus Texture Classification Using Genetic Algorithm and Pre-trained Convolutional Neural Networks



Chandra Mohan Bhuma

Keywords Virus texture classification · TEM images · Pre-trained CNNs · Genetic algorithm

1 Introduction

Images of certain nature contain repetitive patterns and persistent structure which is known as texture. There are many other definitions of texture. Texture is used in classification and recognition of objects. Identification of type of virus is important in medicine and biology. Compared to light microscopic images, TEM images are able to offer higher resolutions and hence they are more informative than light microscopic images. Virus texture can be described by using traditional Gabor filters, image moments, Haralick features, GLCM features, and local binary pattern (LBP).

This paper is organized as follows. A brief review of pre-trained CNNs and the feature extraction from them is presented in Sect. 3. Details of the dataset is given in Sect. 4. Proposed algorithm is illustrated in Sect. 5. Simulations and results are given in Sect. 6. Conclusions are given in Sect. 7.

2 Related Works

Kylber et al. [1] have demonstrated the power of two local two global texture descriptors in identifying the type of virus in the virus texture dataset. Loris Nanni et al. [2] used quinary coding of LBP variants, and a mean accuracy of 80.7% was reported. Dos Santos et al. [3] have used ensemble of texture descriptors, i.e., LBP,

C. M. Bhuma (✉)

Department of ECE, Bapatla Engineering College, Bapatla, India

local ternary pattern, dense local binary pattern, and other three descriptors for virus classification. A combination of LBP and multi-scale PCA (principal component analysis) was used by Wen et al. [4] and obtained an accuracy of $86.2 \pm 2.01\%$.

J Y Ren and X J Wu [5] have used CovD (covariance descriptor) for virus texture dataset and the reported accuracy was $79.4 \pm 3.3\%$. K X Chen et al. [6] have extended the concept of CovD and have shown improved accuracy compared to [5].

Backes and Mesquita [7] have used a fusion approach and fused the features obtained using randomized neural network (RNN), Haralick features, Gabor wavelets, Fourier, DCT, Gray Level Dependence Matrix (GLDM), fractal descriptors and lacunarity. In a recent work, Geus et al. [8] used convolution neural networks (CNN), i.e., ResNet, DenseNet, InceptionV3, and SqueeNet. They have trained the CNNs for 50 and 100 epochs and have shown the superiority of the CNNs over traditional texture descriptors. In this work also, we demonstrate that the features extracted from the selected pre-trained CNNs can be concatenated and much higher improved accuracies can be obtained. Since there are more than 600+ CNN architectures available as of now, it is growing due to the tremendous hardwork by deep learning researchers. Best pre-trained CNNs are selected from a pool of feature matrices which are pre-computed.

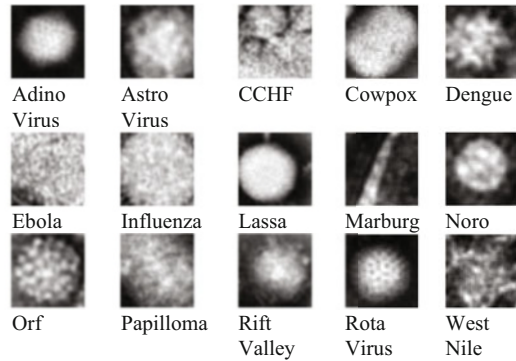
3 Pre-trained Networks and Feature Extraction

In recent times, CNNs have been extensively used by many researchers for image classification problems. With the given dataset, the CNNs can be trained from scratch and inference is done with the trained CNN. This is laborious and may take longer times based on the size of the dataset. Transfer learning approach permits us to utilize the trained CNN weights for another dataset. It is possible to freeze certain layers and utilize the remaining layers for training. With minimal effort, features can be extracted from the pre-trained CNNs after passing the images through the CNN. This is called feature extraction. In this work, this approach is used. All the pre-trained networks used in this work are the CNNs trained on ImageNet [9] dataset. One network is trained on CUB-200dataset [10]. A summary of the used pre-trained CNNs is given in Table 1.

4 Dataset Description

The virus texture dataset v.1.0 [1] is a dataset comprising TEM images of 15 types of viruses. With 100 images per class, 15 classes, this dataset is a 1500 images dataset. All the images are of 41×41 gray-scale images. Shape of the virus ranges from icosahedral to highly pleomorphic. Both 8- and 16-bit images are available in the dataset. In this work, 8-bit images are considered and are shown in Fig. 1.

Fig. 1 Sample images of each category from the virus texture dataset



5 Proposed Algorithm

Features are extracted from the last pooling layer of the pre-trained networks. A total of 34 pre-trained networks as given in Table 1 are used in this work. At first, the dataset is resized and normalized with appropriate mean and standard deviation as per the requirement of the selected pre-trained network and dataset on which it was trained. Features matrices are obtained for all the pre-trained networks. Selection of the pre-trained network is done using genetic algorithm [11]. As such any meta-heuristic algorithm can be used for this. Features from the selected networks are concatenated. Appropriate train and test split or K-fold cross-validation is applied. Train feature vector is given to a classifier. Test feature vector is given to the trained classifier and predicted class labels are obtained. Mean classification accuracy is computed. The error between the ideal mean classification accuracy (100%) and the obtained mean classification accuracy from the selected pre-trained networks is the fitness function. This fitness function is minimized using GA. The process continues until a minimum error is obtained or terminated with a prescribed time limit. The best pre-trained networks feature matrices are obtained. Since the feature vector dimension increases with concatenation, suitable dimensionality reduction techniques can be employed for faster inference. Steps of the proposed algorithm are illustrated in Fig. 2. The global objective function and local objective function minimization as the number of epochs changes are given in Fig. 3. Various other plots demonstrating the GA process [12] utilized in the proposed algorithm are shown in Fig. 3.

6 Simulations and Results

Extensive collection of various pre-trained CNNs based on Pytorch framework is available in “osmrgithublibrary” [13]. Even though there are many pre-trained networks available in the sandbox of [13], we have selected only 34 pre-trained

CNNs. In the proposed work, the best pre-trained networks selected by GA are ResNet101, ECA101d, DPN131, and CondenseNet. The feature vector sizes are 2048, 2048, 2688, and 2064. After concatenation, the feature vector size increases to 8848. In fact, as can be seen from Table 2, the performance of passive aggressive classifier and linear SVC is superior to ridge classifier. Performance of the tree-based and ensemble-based classifiers is not encouraging.

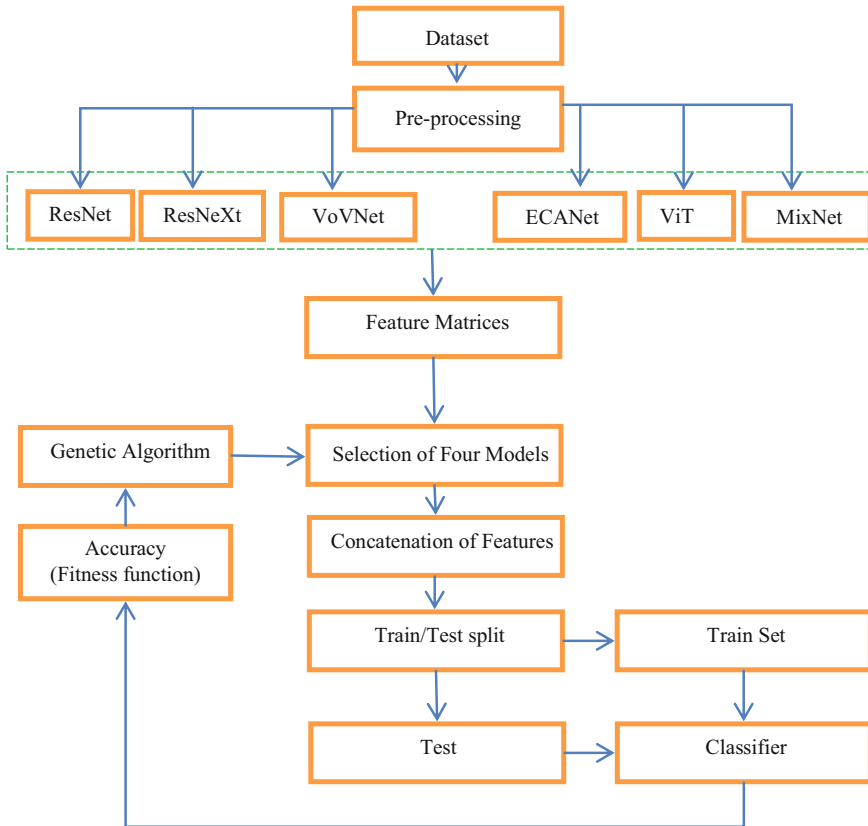


Fig. 2 Block Diagram of the proposed algorithm

The proposed model is more general as indicated by the mean classification accuracies achieved with both tenfold CV and fivefold CV. Hence, there is no overfitting problem. To ensure this, the performance of the algorithm is assessed by repeating the tenfold CV as shown in Fig. 4. Even for 15 repeats, the mean classification accuracy is around 95%. Further, the role of K value in the cross-validation is studied and is presented in Fig. 5. K is varied from 2 to 30 and the mean classification accuracy obtained is around 95%. All the above said validations are applied by considering a linear SVC classifier.

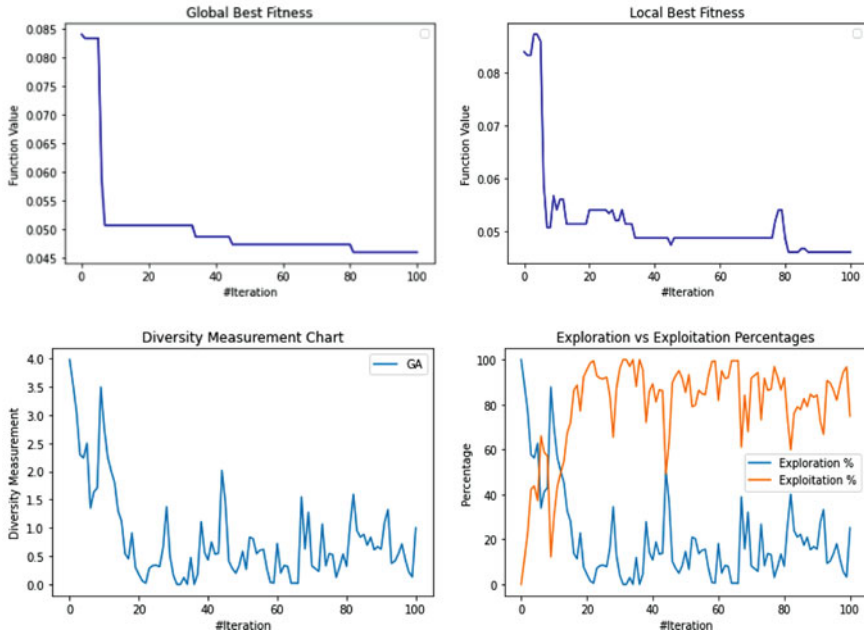


Fig. 3 Various plots illustrating the GA minimization of the fitness function

To assess the model performance on each class, confusion matrix is obtained by selecting an 80% train and 20% test split of the data. With a linear SVC classifier, the confusion matrix obtained is shown in Fig. 6. Per class metrics, i.e., precision, recall and F1 score are computed and presented in Table 3. Precision of Dengue and Marburg virus is the least compared to the other virus types. Except Dengue, West Nile, and Marburg virus, all the other 12 viruses are classified with a precision and recall of more than 0.9. It is to be noted that the majority of the viruses are classified with a precision and recall of 1.0. As the feature vector size is large, dimensionality reduction can be applied. Various dimensionality reduction techniques are available in the literature. Performance of the proposed algorithm after applying six dimensionality reduction techniques is given in Table 4 by considering only 100 components. Just with 100 components only, a mean classification accuracy of more than 90% is obtained with PCA and SVD. Performance of the proposed algorithm is compared with the state-of-the-art techniques on virus dataset and is given in Table 5. Compared to the works of [7] and [8] which are more recent once, the proposed algorithm offers much higher mean classification accuracy and is more general without any overfitting problem.

Table 1 Various pre-trained CNNs used in this work

CNN architecture	Variants used
Dual Path Networks (DPN) [14]	Dpn107, Dpn98, Dpn92, Dpn68, Dpn131
Attention Inspiring Receptive-fields Network [15]	Eairv2
Cross Stage Partial Network (CSPNet) [16]	CSPDarkNet53
Variety of View Network (VoVNet) [17]	ESEVovnet39b
Vision Transformer (ViT) [18]	ViTsmall16x224, ViTlarge16x224 ViTbase16x224, ViTbase32x224 ViTbase32x384, ViTbase16x384 ViTlarge32x384, ViTlarge16x384
Residual Network (ResNet) [19]	ResNet12, ResNet14, ResNet50 ResNet101, ResNet12(CUB), RegNet
Miscellaneous Networks used	PyramidalNet, PeeleNet, BAMNet, WideResNet, ShakeResNet NASNet, DLANet, CondenseNet, MixNet, XceptionNet, HRNet, ECANET

Table 2 Various classifiers performance for tenfold and fivefold validations

Classifier	Tenfold CV Accuracy (%)	Fivefold CV Accuracy (%)
Logistic Regression Classifier	95.0	94.7
Ridge Classifier	93.1	92.5
SGD Classifier	93.7	93.7
Passive Aggressive Classifier	95.2	95.06
KNN Classifier	87.1	86.5
Decision Tree Classifier	57.1	57
Extra Tree Classifier	45.6	46.1
Linear SVC Classifier	95.4	95.2
SVC Classifier (RBF)	93.1	92.6
Gaussian Naïve Bayes Classifier	65.1	61.6
Adaboost Classifier	12.7	14.8
Bagging Classifier	77.6	78.06
Random Forest Classifier	87.4	86.4
Gaussian Process Classifier	6.6	6.6

7 Conclusion

An algorithm for accurately classifying the virus texture images of TEM nature is proposed in this work. Several pre-trained networks are used in the work. Feature matrices from the pre-trained networks are computed. Concatenated features are used in the classification process. Using genetic algorithm, the best pre-trained networks features are selected and their features are concatenated. Minimization of the error between the ideal (100%) and predicted mean classification accuracy is achieved with GA. Performance of the several classifiers is assessed and reported. The feature vector size increases with the concatenation. Hence, dimensionality

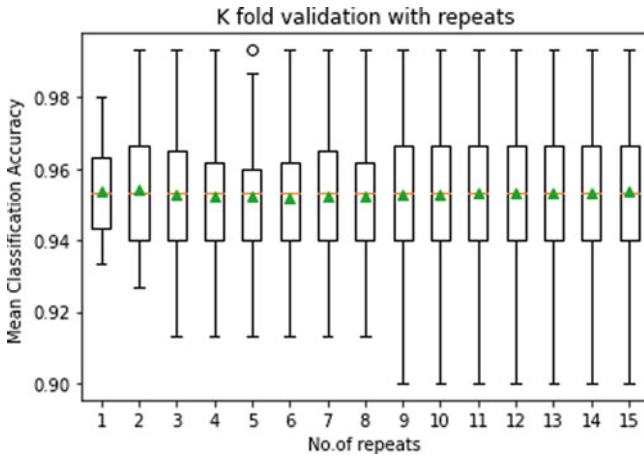


Fig. 4 K-fold validation with repeats

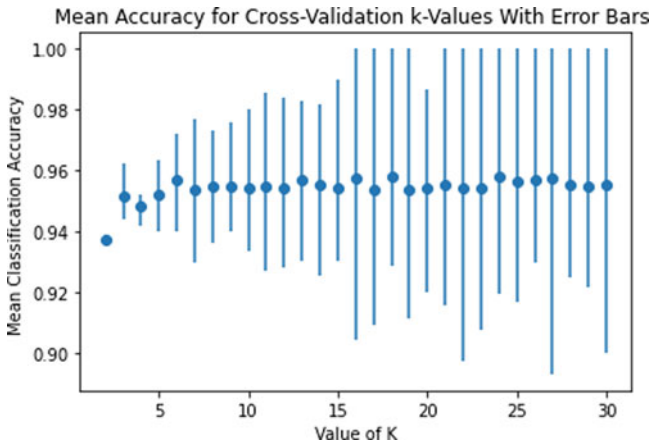


Fig. 5 Mean classification accuracy vs. K

reduction techniques can be employed for faster evaluation. With 100 components chosen, six dimensionality reduction algorithms are compared in this work. The method proposed is more general and as such there is no overfitting problem. Extensive simulations are carried using K-fold cross-validation and the simulations are repeated to demonstrate the model generalization capability. Various other metrics used in assessing the classifiers ability are also reported. Compared to many works reported in the literature, the proposed approach offers higher mean classification accuracies. With a tenfold and fivefold cross-validation, mean classification accuracies achieved are 95.40% and 95.20% respectively.

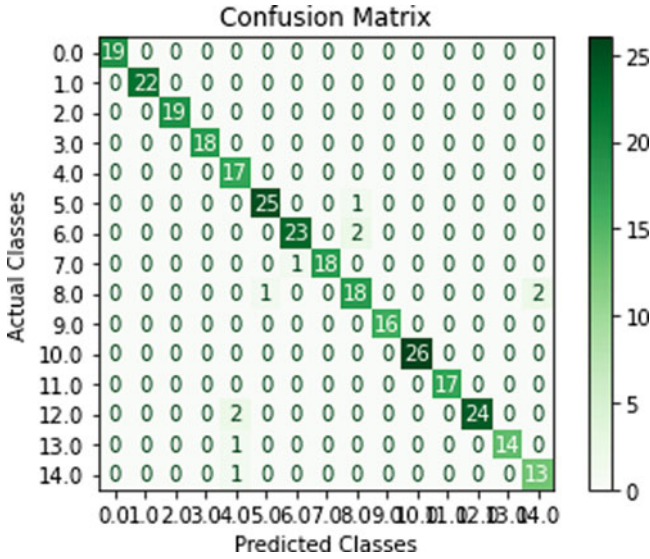


Fig. 6 Confusion matrix with 80%train and 20%test split (linear SVC classifier) (0-Adino Virus, 1-Astro Virus, 2-CCHF, 3-Cowpox, 4-Dengue, 5-Ebola, 6-Influenza, 7-Lassa, 8-Marburg, 9-Noro Virus, 10-Orf, 11-Papilloma, 12-Rift Valley, 13-Rota Virus, 14-West Nile)

Table 3 Classwise metrics

Type of Virus	Precision	Recall	F1 score
AdinoVirus	1.00	1.00	1.00
AstroVirus	1.00	1.00	1.00
CCHF	1.00	1.00	1.00
Cowpox	1.00	1.00	1.00
Dengue	0.81	1.00	0.89
Ebola	0.96	0.96	0.96
Influenza	0.96	0.92	0.94
Lassa	1.00	0.95	0.97
Marburg	0.86	0.86	0.86
NoroVirus	1.00	1.00	1.00
Orf	1.00	1.00	1.00
Papilloma	1.00	1.00	1.00
Rift Valley	1.00	0.92	0.96
Rotavirus	1.00	0.93	0.97
West Nile	0.87	0.93	0.90

Table 4 Effect of dimensionality reduction

Technique	No. of components	Tenfold CV %Accuracy	Fivefold CV %Accuracy
Principle component analysis (PCA)	100	90.0	88.1
Singular value decomposition (SVD)	100	90.8	89.7
Linear discriminant analysis (LDA)	100	89.1	88.9
Isomap embedding	100	79.7	77.9
Locally linear embedding	100	85.3	84.9
Modified locally linear embedding	100	59.1	62.6

Table 5 Comparison of the proposed method with the existing works

Methods	Database type	Description	Mean classification accuracy(%)
Proposed	8 bit	Concatenation of features ResNet101,ECA101d, Dpn131,CondenseNet	95.40% (tenfold CV) 95.20% (fivefold CV)
A.R.Backes and Mesquita [7]	16 bit	Fusion of Haralick, Gabor wavelets, Haralick, Fourier, DCT	87.27 ± 1.65%
A.R.de Geus et al. [8]	8 bit	CNN training	89.00 ± 2.3% (DenseNet)
Z.Wen et al. [4]	16 bit	LBP, PCA	86.2 ± 2.01%
F. L. C. dos Santos et al. [3]	16 bit	Ensemble of descriptors LBP,LTP,DLBP,RLBP	85.70%
Jie-Yi Ren et al. [5]	8 bit	Covariance descriptor	79.40%
Chen, K.X. et al. [6]	8 bit	Euclidean space to semi-positive definite manifold	77.93%

References

1. Kylberg G., Uppström M., and Sintorn I.-M. Virus Texture Analysis Using Local Binary Patterns and Radial Density Profiles In Proceedings of the 16th IberoAmerican Congress on Pattern Recognition (CIARP), LNCS-7042, pp. 573-580, 2011.
2. Loris Nanni et al., “Virus Image Classification using Different Texture Descriptors”, The 14th Interntional Conference on Bio informatics and Computational Biology, (BioComp’13) July, 22-25, 2013. Los Vegas, USA
3. L. C. dos Santos, M. Paci, L. Nanni, S. Brahnam, and J. Hyttinen, “Computer vision for virus image classification,” *Biosystems Engineering*, vol. 138, pp. 11 – 22, 2015, innovations in Medicine and Healthcare.
4. Z. Wen, Z. Li, Y. Peng, and S. Ying, “Virus image classification using multi-scale completed local binary pattern features extracted from filtered images by multi-scale principal component analysis,” *Pattern Recognition Letters*, vol. 79, pp. 25 – 30, 2016.
5. Jie-Yi Ren, Xiao-Jun Wu, “Vectorial approximations of infinite-dimensional covariance descriptors for image classification”, *Computational Visual Media*, Vol. 3, No. 4, December 2017, 379–385.
6. J KX Chen et al., “More About Covariance Descriptors for Image Set Coding: Log-Euclidean Framework based Kernel Matrix Representation”, *IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 27-28, October, 2019, Seoul, South Korea.
7. A. R. Backes and J. J. de Mesquita Sá Junior, “Virus Classification by Using a Fusion of Texture Analysis Methods,” *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2020, pp. 290-295, doi: <https://doi.org/10.1109/IWSSIP48289.2020.9145325>.
8. A. R. de Geus., A. R. Backes., and J. R. Souza., “Variability evaluation of cnns using cross-validation on viruses images,” in *Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 4: VISAPP, INSTICC*. SciTePress, 2020, pp. 626–632.
9. ImageNet, <http://www.image-net.org>
10. Welinder P., Branson S., Mita T., Wah C., Schroff F., Belongie S., Perona, P. “Caltech-UCSD Birds 200”. California Institute of Technology. CNS-TR-2010-001. 2010.
11. GA - Genetic Algorithm . Holland, J. H. (1992). *Genetic algorithms*. Scientific american, 267(1), 66-73.
12. Thieu Nguyen, “A collection of the state of the art Meta Heuristic Algorithms in Python: MealPy”, March, 2020, Zenodo, doi: <https://doi.org/10.5281/zenodo.3711948>.
13. <https://github.com/osmr/imgclsmob>
14. Yunpeng Chen and Jianan Li and Huaxin Xiao and Xiaojie Jin and Shuicheng Yan and Jiashi Feng, “Dual Path Networks”, arXiv: 1707.01629, 2017.
15. L. Yang, Q. Song, Y. Wu and M. Hu, “Attention Inspiring Receptive-Fields Network for Learning Invariant Representations,” in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 6, pp. 1744-1755, June 2019, doi: <https://doi.org/10.1109/TNNLS.2018.2873722>.
16. Alexey Bochkovskiy and Chien-Yao Wang and Hong-Yuan Mark Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection”, arXiv: 2004.10934, 2020
17. Youngwan Lee and Joong-won Hwang and Sangrok Lee and Yuseok Bae and Jongyoul Park, “An Energy and GPU-Computation Efficient Backbone Network for Real-Time Object Detection”, arXiv: 1904.09730, 2019.
18. Alexey Dosovitskiy et al., “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, arXiv: 2010.11929, 2020.
19. He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep residual learning for image recognition.” In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778, 2016.

Fog Computing-Enabled Internet of Things for Resource Optimization



Meenaxi M. Raikar  and Meena S M 

Keywords MQTT · Raspberry Pi · Publish · Subscribe · Edge device

1 Introduction

The Fog-enabled Internet of Things (IoT) architecture is used for resource optimization. The agriculture sector use case is considered for providing service to the farmers. The farmer can enable the services using the mobile application. The lightweight protocol is chosen for communication between the devices for energy-efficient transmission of the data. The network topology consists of heterogeneous nodes such as things and edge devices. The nodes in the network are power constraint devices; hence, the lightweight protocol is opted for the communication. The Message Queue Telemetry Transport (MQTT) is the lightweight protocol that is used for communication between the edge devices such as Raspberry Pi. The MQTT protocol uses publish-subscribe communication model. It is assumed that the service provider will have the infrastructure for facilitating the different services mentioned.

The recommendation by experts regarding which crop to be grown in the field based on region, season, and climate conditions is available to the farmers. Advice related to the appropriate time for sowing the seed to get better yield is facilitated on request through mobile. This paper outlines the use of IoT devices such as rain sensor, soil moisture sensor, and temperature sensor to sense the agricultural data and store into edge device database. The stored data is monitored through big data analytics, and prediction is performed using data mining techniques. The energy efficiency is computed in terms of the packet size and the round trip time for the transmission of the data between the devices.

M. M. Raikar (✉) · Meena S M
K. L. E. Technological University, Hubballi, India
e-mail: mmraikar@kletech.ac.in; msm@kletech.ac.in

The organization of the paper: In Sect. 2, related work on Fog IoT services is presented. In Sect. 3, the architecture for Fog IoT taking precision agriculture as a use case is described. The energy model for the Fog IoT devices is discussed in Sect. 4. The distance between the Fog IoT nodes, a factor to reduce latency, is presented in Sect. 5. The result analysis and conclusion is presented in Sects. 6, and 7, respectively.

2 State of the Art in Fog-Enabled IoT Services

In [1], the system that collects soil properties and stores it in cloud for further analysis is mentioned. It also speaks about building a scalable sensor data analysis for smart farming with commercially available IoT devices which reduces maintenance cost and provides data analytics resulting in increased crop yield. In [2], cloud-based IoT application for precision agriculture with three-layer architecture is described. The top layer deals with collecting information and applying necessary agricultural actions. The next layer connects the top layer to IoT. The last layer deals with data storage and processing. AgroTick, a novel hybrid system for smart agriculture, is discussed in [3]. AgroTick is an Internet of Things (IoT) system with a mobile interface that was created using technological modules such as cloud computing, embedded firmware, hardware, and big data analytics. AgroTick is built to increase agricultural efficiency, develop a well-connected farming network, and provide a knowledge-sharing platform for farmers. In [4–7], the lightweight protocol, Message Queue Telemetry Transport (MQTT), is the communication model used. It has opened its way in many sectors since its invention in 1999.

The different cloud deployments and services available for the Internet users are presented in [8]. The various applications of IoT such as smart parking, waste management, and home automation are described in [9]. The power efficiency for IoT devices is enhanced using the software-defined networks (SDN) architecture [10].

The energy efficiency of the IoT devices is improved using load balancing and fault tolerance techniques [11]. The IoT services of the smart city are described in [12]. The 6LoWPAN and CoAP protocol in development of the applications are presented in [13]. The security issues are concerns in the IoT network presented in [14, 15].

3 Fog Computing-Enabled IoT Architecture

The Fog computing-enabled IoT architecture for precision agriculture using lightweight protocol is shown in Fig. 1. The objective is to provide different service for farmers that aid in precision agriculture such as:

1. Irrigation and fertigation as a service
2. Crop-related decision support system as a service
3. Recommendation as a service
4. Pest control as a service

The farmers have to subscribe for the services by registration process. The metering and billing module at the service provider (SP) side keeps track of the unit amount of resource utilized by the farmers.

Irrigation and Fertigation As a Service: This service is used by farmers to irrigate and fertigate the land. The controlled amount of water and fertilizers are supplied to the plants based on the different parameters such as soil moisture content, temperature, and rainfall in the region.

Crop-Related Decision Support System As a Service This service aids the farmer in making decisions related to the type of crop to be sowed or the suitable time for sowing in the region.

To facilitate this service at the SP side, big data analytics technique are applied as shown in Fig. 1.

Recommendation As a Service The farmers who have subscribed to this service get recommendation related to the crop. The data related to each farmers land is collected using sensors and stored at the data center for analysis. The data mining techniques are applied for recommendation to the farmers.

Pest Control As a Service This service enables the farmers to combat the loss in yield due to pests and rodents.

The automated fertigation unit is installed at the service provider location. The metering unit aids in determining the amount of unit consumed by the farmers for accounting and billing. The virtualization technique is applied for efficient utilization of resources at the data centers. The virtual private network (VPN) component enables the user to have private network for security reasons.

These services are deployed at the edge device closer to the sensors to minimize the latency. In the Fig. 1, end devices could be the mobile, laptop, or the tablet. The services could be accessed using a browser or an application. The service response time could be greatly enhanced by deploying these on the edge devices closer to the sensors. In the next section, the energy efficiency model for fog IoT architecture is presented.

4 Energy Model for the Fog IoT Devices

The evolution of Fog computing-enabled Internet of Things is at its peak as represented in Gartner's predictions in the recent days. Here, the things connect to the Internet anytime, anywhere providing the ubiquity to the users. These things are

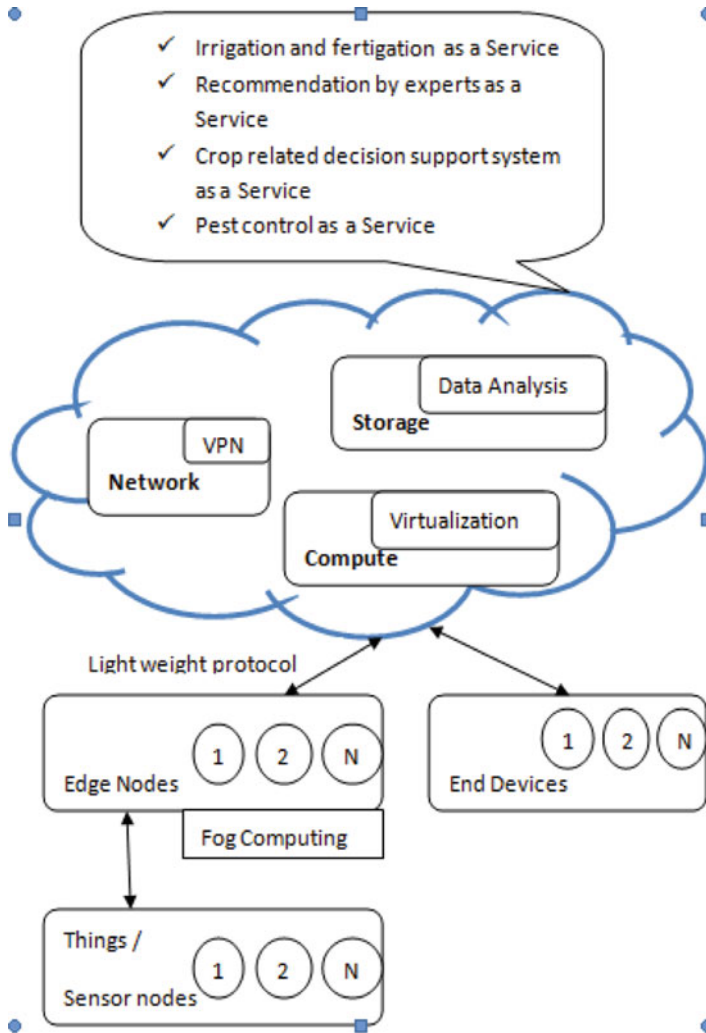


Fig. 1 Fog-enabled IoT architecture using lightweight protocol

fueled by energy sources. Hence, energy efficient utilization of the IoT resources is an important research domain. With the invention of Industrial IoT, the need for low power technology is increasing tremendously. Thus, the power consumption model for IoT applications plays an important role in network resource optimization. A trade-off is to be achieved between cost incurred, network lifetime, and the power consumption of the devices. The power efficiency model for IoT devices is presented in [16]. The effect of power consumption based on the traffic pattern in IoT devices of smart city applications is described in [17].

In an IoT application, energy is consumed during data acquisition, processing, and communication. The alternate to save energy is to use energy harvesting technologies, which is out of the scope of this paper. The analysis of energy in the life cycle of an IoT application is presented here. The objective is to minimize the power consumption of the IoT applications. In the literature, system level conservation of energy is described vastly. The focus of this paper is mainly from the perspective of energy consumption during communication between the devices.

In an IoT application, the pattern observed is data acquisition by the sensors/things, processing performed by the controller, and the transmission of the data. Based on this pattern, the power consumption of the IoT application is broken into four different blocks. The first block being the power consumed at the system/device level (P_{SYS}) to perform the operations at the operating system level. The second block being the power consumption during the data acquisition by the sensors/things (P_{DACQ}). The third block is power consumed for processing (P_{PROC}) and finally the power utilized for communicating between the networking devices (P_{COMMN}). The mathematical representation of the observed pattern in IoT application is given in Eq. (1).

$$P_{NDEV} = P_{SYS} + P_{DACQ} + P_{PROC} + P_{COMMN} \quad (1)$$

With the main objective being the power conservation during communication, the lightweight protocol is used in the development of the IoT applications. The payload in case of lightweight protocol such as MQTT for IoT applications is 2 bytes. Since the IoT devices are battery powered, the objective is to minimize the power consumed during transmission for increasing the network lifetime.

4.1 Power Consumption Model for Data Communication

The average power consumed during communication is represented as energy required to send a message (E_M) and the time interval between consecutive messages (T_M) as given in Eq. (2).

$$P_{COMMN} = \sum_{k=1}^n \frac{E_M^{(k)}}{T_M^{(k)}} \quad (2)$$

where n is the number of messages sent during the experimental period and k varies from 1 to n . The factors that influence E_M are based on whether it is a wired medium or wireless medium for transmission, embedded chip type on the sensor/thing and the period for every transmission.

The IoT applications can be categorized into two types such as periodic and event trigger-based reporting. In case of periodic reporting such as temperature monitoring, the value for $T_M^{(k)}$ is constant. Therefore, Eq. (2) is rewritten as:

$$P_{COMMN} = \frac{1}{T_M^0} \sum_{k=1}^n E_M^{(k)} \quad (3)$$

Here, T_M^0 is constant time interval between consecutive reporting instances.

4.2 Power Consumption Model for Data Acquisition

The IoT applications are classified into two types such as periodic reporting and nonperiodic or event trigger-based reporting. The power consumption for the data acquisition is given by Eq. (4):

$$P_{DACQ} = \begin{cases} P_{DACQ-1-SAMPLE} \cdot SN_1 \dots (\text{periodic}) \\ P_{DACQ-1-SAMPLE} \cdot SN_2 \cdot Prob(E) \end{cases} \quad (4)$$

where SN_1 and SN_2 are the number of samples for periodic and event trigger-based reporting, respectively. In case of the event trigger-based reporting, the probability of occurrence of an event (E) is taken into consideration.

4.3 Power Consumption Model of a System/Device

The different states of a networking device are active, idle/sleep, transmit, and receive. The power consumption of a networking device (P_{SYS}) is given as:

$$P_{SYS} = P_{active} + P_{sleep/idle} + P_{transmit} + P_{receive} \quad (5)$$

The power consumption during each of these states is presented in this section.

Data Transmission The power consumption for transmission of data in an IoT environment, considering periodic transmission, is represented as:

$$E_{DataTx} = P_{DataTx}(mW) \times T_M^0(ms) \quad (6)$$

Data Reception Similarly, the power consumed to receive the data is given as:

$$E_{DataRx} = P_{DataRx}(mW) \times T_{Rx}(ms) \quad (7)$$

where $T_{Rx}(ms)$ is the duration for data reception.

Active/Sleep Modes Based on the active and sleep modes of the networking device, power consumption is modelled as:

$$P_{active+sleep} = P_s \times T_s + (T_{total} - T_s) * P_{active} \quad (8)$$

where T_{total} and T_s are the total time spent and time spent in sleep mode, respectively.

The power consumption of an IoT device can be reduced by increasing the sleep modes. Based on the deviation of the sensed data, the sleep modes can be increased for prolonged network lifetime. The deviation of the sensed data is computed as:

$$Deviation_of_sensed_data = \frac{\sqrt{\sum (y_i - \mu)^2}}{\mu} \quad (9)$$

where y_i is the sensed data and μ is the mean computed for “ n ” sensed data values. μ is computed as $\sum_{i=1}^n y_i/n$. The data reduction is performed by monitoring the deviation in the data sensed. The voluminous data, variety of sensors, and velocity of data generated are the aspects to be considered in Fog computing.

4.4 Power Consumption Model for Data Processing

The power consumed for data processing depends on the number of operations performed such as arithmetic operations. It depends on the hardware architecture chosen for the deployment of the IoT applications.

In this section, the different units of power consumption in the IoT applications are outlined. With the proposed architecture, the latency for the services is reduced which aids in decreasing the power consumed.

5 Distance Between Nodes in Fog IoT

Assuming there exists “ M ” routers between the source and destination, the end-to-end delay (d_{E-to-E}) is given by Eq. (10), when the queuing delay is negligible:

$$d_{E-to-E} = M * (d_{proc} + d_{trans} + d_{prop}) \quad (10)$$

where d_{proc} is the processing delay, d_{trans} is the transmission delay, and d_{prop} is the propagation delay [18]. If the link length is doubled from d to $2d$, the propagation delay is $d_{prop} = 2 * d_{prop}$. The propagation delay is proportional to the distance

between the nodes. Hence, the Fog computing reduces the end-to-end delay when the distance between the data gathering nodes and edge devices is reduced.

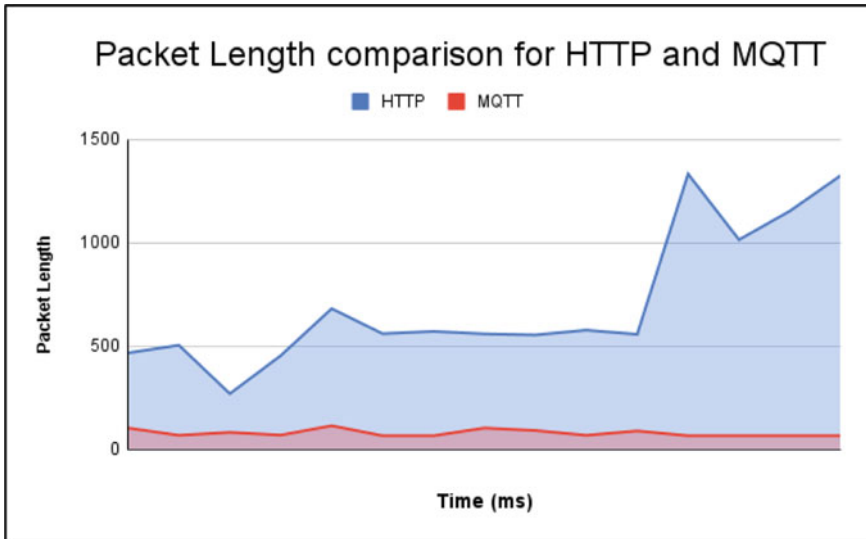


Fig. 2 Comparison of packet length for HTTP and MQTT protocol

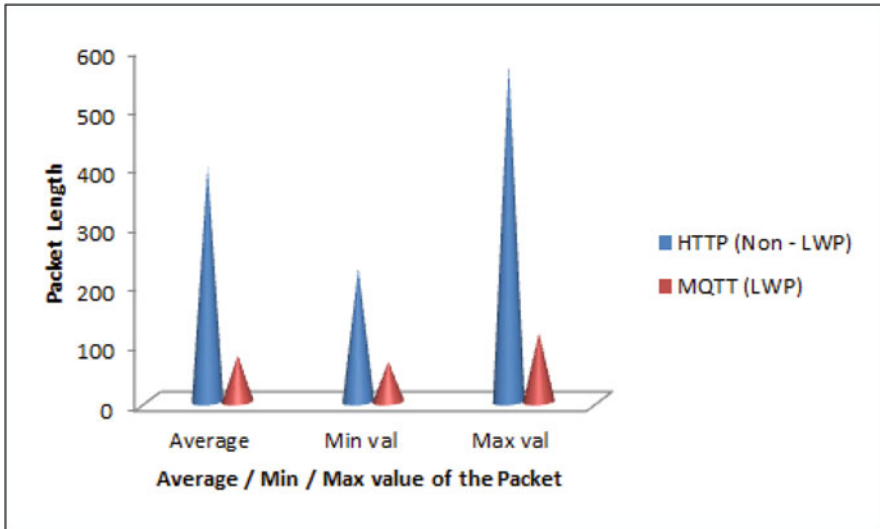


Fig. 3 Packet length for lightweight protocol and non-lightweight protocol

6 Result Analysis

The Raspberry Pi 3 is used as the end node at the fog computing layer for aggregation, computation, and processing. The lightweight protocol is used for data transmission between the nodes. MQTT protocol is chosen as the lightweight protocol for transmission of the sensor-captured data. The “publish-subscribe” communication model is applied in MQTT. The mosquito broker is utilized to send the sensor-captured data between the subscriber (farmers) and publisher (SP).

The mosquito broker is configured to function as a broker on Raspberry Pi system. An instance is created on broker with the port number. The topic with name as “IrrigationFertigation” is created. The subscribers connect to the broker using this topic. The message to “ON” the irrigation/fertigation unit is sent using this topic.

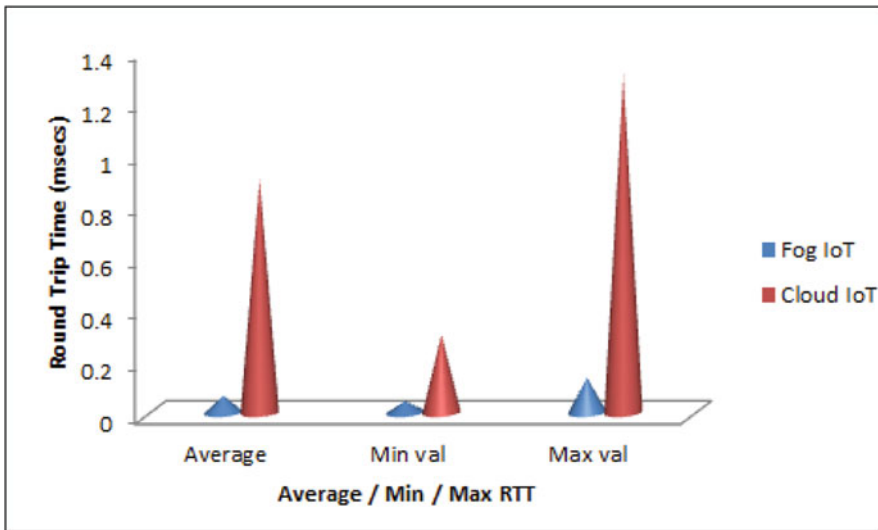


Fig. 4 RTT comparison of Fog IoT and Cloud IoT architecture

The Wireshark tool is used to capture the packet information. The filter is applied to compare the packet length during transmission using “http” and “mqtt” protocol. The area covered with blue represents the packet length for HTTP, and the red color presents the packet length for MQTT as shown in Fig. 2. The other lightweight protocol is CoAP (Constrained Application Protocol) used for power-constrained devices. In Fig. 3, the average, minimum, and maximum packet length of non-lightweight protocol – Hyper Text Transfer Protocol (HTTP) – and lightweight protocol (MQTT) is represented. In Fig. 4, the round-trip time comparison for Cloud IoT and Fog IoT architecture is presented.

7 Conclusion

The Fog-enabled IoT architecture proposed reduces the round-trip time as the edge devices are closer to the sensor devices. The case study presented benefits the farmer to deploy precision agriculture techniques in the farm for increasing the crop yield. This model could be scaled to the entire rural area. The lightweight protocol is selected for service providing, because it aids in lowering the latency and achieving higher throughput.

References

1. S. Rajeswari and et al., “A smart agricultural model by integrating IoT, mobile and cloud-based big data analytics”, 2017 International Conference on Intelligent Computing and Control (I2C2), Coimbatore, 2017, pp. 1–5.
2. A. Khattab and et al., “Design and implementation of a cloud-based IoT scheme for precision agriculture,” 2016 28th International Conference on Microelectronics (ICM), Giza, 2016, pp. 201–204.
3. S. Roy et al., “IoT, big data science & analytics, cloud computing and mobile app based hybrid system for smart agriculture,” 2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON), 2017, pp. 303–304, <https://doi.org/10.1109/IEMECON.2017.8079610>.
4. W. R. Heinzelman and et al., “Energy-efficient communication protocol for wireless microsensor networks,” Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, Maui, HI, USA, 2000, pp. 10 pp. vol.2.
5. P. R. Deshmukh and D. Bhalerao, “An implementation of MQTT through the application of warehouse management system for climacteric fruits and vegetables”, 2nd International Conference on Communication and Electronics Systems (ICES), Coimbatore, 2017, pp. 844–849.
6. J. Velez and et al., “IEEE 1451-1-6: Providing common network services over MQTT,” 2018 IEEE Sensors Applications Symposium (SAS), Seoul, 2018, pp. 1–6.
7. M. M. Raikar and et al., “Blend of Cloud and Internet of Things (IoT) in agriculture sector using light weight protocol”, 2018 IEEE ICACCI, Bangalore.
8. M. M. Raikar, P. Desai, M. Vijayalakshmi and P. Narayankar, “Augmenting Cloud concepts learning with Open source software environment,” 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, 2018, pp. 1405–1411. <https://doi.org/10.1109/ICACCI.2018.8554826>
9. M. M. Raikar, P. Desai, V. M and P. Narayankar, “Upsurge of IoT (Internet of Things) in engineering education: A case study,” 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, 2018, pp. 191–197. <https://doi.org/10.1109/ICACCI.2018.8554546>
10. N. Kaur and S. K. Sood, “An Energy-Efficient Architecture for the Internet of Things (IoT),” in IEEE Systems Journal, vol. 11, no. 2, pp. 796–805, June 2017. <https://doi.org/10.1109/JSYST.2015.2469676>.
11. Farzad Kiani, “A Survey on Management Frameworks and Open Challenges in IoT,” Wireless Communications and Mobile Computing, vol. 2018, Article ID 9857026, 33 pages, 2018. <https://doi.org/10.1155/2018/9857026>.
12. A. Zanella, N. Bui, A. Castellani, L. Vangelista and M. Zorzi, “Internet of Things for Smart Cities,” in IEEE Internet of Things Journal, vol. 1, no. 1, pp. 22–32, Feb. 2014. <https://doi.org/10.1109/JIOT.2014.2306328>

13. Izal, Mikel et al. "Computation of Traffic Time Series for Large Populations of IoT Devices" *Sensors* (Basel, Switzerland) vol. 19, 1 78. 26 Dec. 2018, <https://doi.org/10.3390/s19010078>.
14. M. M. Raikar and S. M. Meena, "SSH brute force attack mitigation in Internet of Things (IoT) network: An edge device security measure," 2021 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC), 2021, pp. 72–77, <https://doi.org/10.1109/ICSCCC51823.2021.9478131>.
15. M. M. Raikar and M. S M, "Vulnerability assessment of MQTT protocol in Internet of Things (IoT)," 2021 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC), 2021, pp. 535–540, <https://doi.org/10.1109/ICSCCC51823.2021.9478156>.
16. B. Martinez, M. Montón, I. Vilajosana and J. D. Prades, "The Power of Models: Modeling Power Consumption for IoT Devices," in *IEEE Sensors Journal*, vol. 15, no. 10, pp. 57775789, Oct. 2015. <https://doi.org/10.1109/JSEN.2015.2445094>.
17. A. Ikpehai, B. Adebisi and K. Anoh, "Effects of Traffic Characteristics on Energy Consumption of IoT End Devices in Smart City," 2018 Global Information Infrastructure and Networking Symposium (GIIS), Thessaloniki, Greece, 2018, pp. 1–6. <https://doi.org/10.1109/GIIS.2018.8635744>.
18. Kurose, J. F., & Ross, K. W. (2001). *Computer networking: A top-down approach featuring the Internet*. Boston: Addison-Wesley.

Network Media Content Model in the Era of Smart Devices



Adapa Venkateswara Rao, Molli Srinivasa Rao, and J. Durga Prasad Rao

Keywords Information technology · Information society · Smartphones · User-generated content · Communication networks

1 Introduction

We have been living in a society dominated by social media for more than a decade. The technological revolution has had a profound impact on how content is produced and shared. This has been a constant state for traditional media, institutions, individuals, and groups. This new scenario is hard to predict in the medium-term. This technological revolution allows media content to flow across different environments via its users based upon technical, logistical, and social conditions [4, 8, 12, 31]. New communicative models are required to explain the new logic of production and consumption that is dominated by intelligent devices, whether they are mobile, fixed, or portable. They also have the ability for personal production, distribution, and consumption.

This “massive personalized blast of content” [27] or mass self-communication [6] should be investigated simultaneously as technology [7]. Research challenges are significant for social media and digital technologies. The concepts, models, theories, and models are evolving to meet this new reality from a communication point of view. The paradigm of personalized interaction has replaced the paradigm of unidirectional, massive communication. This super structural phenomenon calls for new research in the areas of media agenda [30], content effects [29], automated ethics [11], and brand communication strategies [14], among other things. New concepts of communication are possible in all these cases. Technological dependence

A. Venkateswara Rao (✉) · M. Srinivasa Rao
Dadi Institute of Engineering & Technology, Anapakalli, Visakhapatnam, India

J. Durga Prasad Rao
Shri Shankaracharya Mahavidyalaya Bhilai, Bhilai, India

and constant rethinking of old models call for a global and theoretical approach to new models of production, distribution, and consumption of media content.

This article presents a proposal that combines theoretical models of communication flows with a general explanation from a technical-practical perspective. The theory behind the features of new communication users, the content created by them, is based on the model of mass self-communication and network structures [5, 6]. The classic phenomenon of media convergence has been reviewed to see if it can be applied to the future evolution of technology. This is the key to the digital revolution.

We also present three flow models or diagrams in which we plan to condense the convergence of technical and theoretical proposals during the three phases of the current information society. These diagrams are a first step toward a comprehensive communicative modelling that explains how information is created, distributed, consumed, and reused today [9] and, in context of Web Science and Network Science [26], understood as an interrelationship of independent systems [26].

2 Network Society

The information society that Manuel Cas tells (2000) and Armand [16], among others, anticipated and described in great detail at the turn of the century is already a reality that has materialized in three different stages [9].

With the birth of information technologies at the beginning of the century, the concept of the Network Society emerged, understood as an interconnection of documents on a planetary scale and in real time through search engines and electronic mail, although with important limitations, mutations for the exchange of audiovisual materials. Web 1.0 was developed, thanks to the rapid increase in data download speeds over the Internet, the democratization of the PC and the laptop, and the rapid digital literacy of the population in work, cultural, and administrative fields. The growing exchange of documents of all kinds (audiovisuals are incorporated) and the emergence of smartphones and smart devices (usable and intuitive) progressively lead to a true network of people connected by multiple criteria through different platforms—social, cultural, labor, political, intimate, etc.—in what has come to be called Web 2.0.

Today's large global networks (Facebook, YouTube, LinkedIn, Twitter. etc.) have fostered multiple associations among their members according to very different criteria (social, cultural, labor, geographic, linguistic, national, etc.), and that [28], when mentioning the Forrester Survey [13], reduces to entertainment, career, and family. To the extent that a large majority of society in developed countries has constant and unlimited access to these social platforms, virtual communities have embraced the old social communities and have been reproduced according to multiple and mutable criteria. The trend is to create a network of networks capable of absorbing the real life of society.

These contents are mediated to the extent that there is a technology that determines the type of production, distribution, and reception. However, it is about

content that is different from that of traditional media. Although these contents may be produced by external agents, the entire process is determined by the construction of the meaning that the community will make by virtue of the criteria that constitute it. This produces an automatic filter phenomenon within the community. To identify this filter, it is necessary to identify the boundaries of the network that direct the rules of the game and the discourse [19].

Thus, the networks are configured for some reason that gives rise to the dynamic filter process and that will determine the criteria by which their members will make comments, add and edit graphic and audiovisual material, share an article in the press, etc. This control depends on a kind of group self-awareness about the relevance of a comment, information, link, or photo in the course of the group and is manifested through the acceptance, evaluation, edition, or reproduction of the shared information. Therefore, each “social” network (or of any kind) is just a macro network that operates under great filtering criteria and that includes thousands of multidimensional subnets whose reason for being is those sub-filters. Therefore, the contents end up being disseminated on the Internet and being able to adopt very different meanings and scopes, depending on the technical conditions that configure their social distribution and the particular interpretation of the message that the individuals that make it up throughout said process.

In short, we are faced with a network structure that ranges from the most personal environment to the most global. This structure exists due to multiple integrating criteria, which guide the production and interpretation of its contents.

3 User-Generated Content

The concept of mass communication has been superseded by a conceptualization based on the receiver, and it is better to talk about receivers whose needs and interests only need the appropriate tools to express themselves and establish contact with other people and groups. As technology made a more real response possible (immediate, economical, sophisticated, and precise), the receiver gradually became a participatory receiver until reaching simultaneous reception and emission functions [10].

This creates a sort of collective productive intelligence [25] because the content changes permanently depending on the contribution of each member, as is the case, for example, with some collaborative models such as Wikipedia and wikis. The content consumed and produced by users can be of three types:

- Pre-existing content, on many occasions, canned for the Internet: it can range from classic mass television content posted on YouTube to small fragments of programs streaming on the web of a media conglomerate.
- Content created ad hoc by YouTubers or network influencers (network-making power): normally through the monitoring of great personalities or entities within the networks.

- Content produced by the members of the network written, graphic, or audio-visual, with a smart device and with applications that allow you to share it immediately.

The new users of communication are themselves a source of content, since the features that define them (sociodemographic criteria, tastes, attitudes, values, etc.) are very important for network managers, that is, brands, companies, or institutions. These agents intervene in this process either by making existing networks more dynamic (switchers) or by creating new ones (network-making power) [5], giving rise to a new operating logic among market players: “the already close relationship between content producers, advertisers, and consumers has become even more intimate” [28].

In sum, technical capacities, economic interests, and social needs constitute the framework criteria for the “consumption and production” of content in the defined network structure. The result is a network socialization at the service of the communicative products that are produced and consumed within it. These products energize and make a certain network evolve based on their contribution to its theme, that is, based on their ability to be: valued (the “likes,” the “participate,” etc.), commented and recommended, edited or filtered (shared, disseminated, sent, etc.), and produced (depending on the previous content, which will once again give the chance to be valued, commented on, etc.).

Therefore, we recover the filter here as a configuration criterion for the “user networks.” The constant generation of content on social networks, in what Dylo and McCluskey have called “customization” of information, defines the users of these networks and builds them in reality [2].

4 Media Convergence

The crisis of the traditional television model together with the technological explosion and the development of the Internet in the framework of the aforementioned Network Society have marked a new converging model of media and communication actors [3, 32]. Current devices supply multiple functionalities on a personal and virtual platform—specific applications to produce text messages, images, sounds, video, internal and cloud storage, recording, and editing of audio and images in high definition. This phenomenon has made it possible to modify the unidirectional communicative and informative model, typical of television, radio, and fixed and cellular telephony. Thanks to these devices, the receiver becomes an active agent in creating and exchanging new digital content. This change has forced the different conventional media to reinvent themselves to compete in a new digital market, more open, collaborative, and diversified in its functions and tasks.

This integrative approach has its first precedent in the term media convergence and refers to the fusion of digital platforms intended to provide cable and satellite television services [21]. When “multimedia” emerged, Salvaterra and Popov [22] articulated a model based on four dimensions—business, technological, professional, and communicative—which included the user for the first time in the global process of news production. These dimensions had a direct impact on the design of communication media interfaces. Little by little, the online or web versions were created, and a process of profound change began in the way of addressing the public, consuming the contents and participating in a gear that finally ends up modifying the format, style, meaning, and impact of the messages. The written press was the first to adapt to the web environment and complement the textual information with audiovisual files. Soon, television channels had to quickly offer content online and “on demand” to provide greater visibility and flexibility to their programs and avoid being overtaken by platforms such as YouTube or Netflix. Radio ended up tracing a similar path with the appearance of streaming stations and podcast files, while, following the model of pay television, it gradually became an alternative for all tastes and countries. From this, online discography business models such as Spotify or Deezer were derived. In addition, new narrative and business formulas have emerged: interactive stories to be consumed through mobile phones such as *The Imp* and *Angus and Cheryl* [17] or stories that have migrated from open television to the web or vice versa such as *Enjuto Mojamuto* [18].

At a technological level, the standardization of Android, iOS, and Windows Phone operating systems on mobile phones and tablets has allowed access to hundreds of applications through virtual stores. Apple, Windows, Google, or Amazon offers countless free and paid resources capable of capturing and recreating digital content in any format [9, 23, 24] text (SMS, WhatsApp, Telegram, Twitter, and Line), image files, and multiple calls with video. Regarding photography and still images, Instagram and DHQ Solutions allow retouching and including texts in photographs and sharing them on social networks. The applications for the treatment of sound and video are also very varied: Video editor and Trimmer are just some examples. Through them we have everything we need to produce content and disseminate it socially through specialized groups, forums, and blogs, practically at no cost.

5 Evolution of Network Models

Behind the evolution of media content flows is a series of key technological changes. The slow introduction of the Internet and the web during the 1990s led to the appearance of the first user-generated content, but it was not until 2005 that own media production became widespread and the first phase of Web 2.0 emerged (blogging, forums, podcasting, first wikis, and video/photo sharing sites). At a

technological level, this first phase is characterized by: (1) a rapid expansion of personal computers (in June 2008 the figure of one billion was reached worldwide); (2) a dynamic increase in computing speed (practically doubling every 2 years, keeping Moore’s Law in force) and storage capacity; and (3) an increase in capacity (from 128 Kb/s of ISDN to 0.25–20 Mbit/s downlink of ADSL).

We develop the graphic model following the first stage of network communication, which we will call User Interaction, presenting the titles of the chart boxes in quotation marks (see Fig. 1).

5.1 First Stage: “User Interaction”

The user creates his own content or retrieves it from the network and is ready to exchange files for different purposes: information, entertainment, and persuasion, which are produced based on this textual information within the framework of the different software or Web 2.0 platforms. These messages are uploaded on websites, blogs, or forums via the Internet or are stored on hard drives to later be sent via email to individual users or massive groups. This communicative process produces an interaction that ended with simple feedback from the reception or directly with no response from its creator.

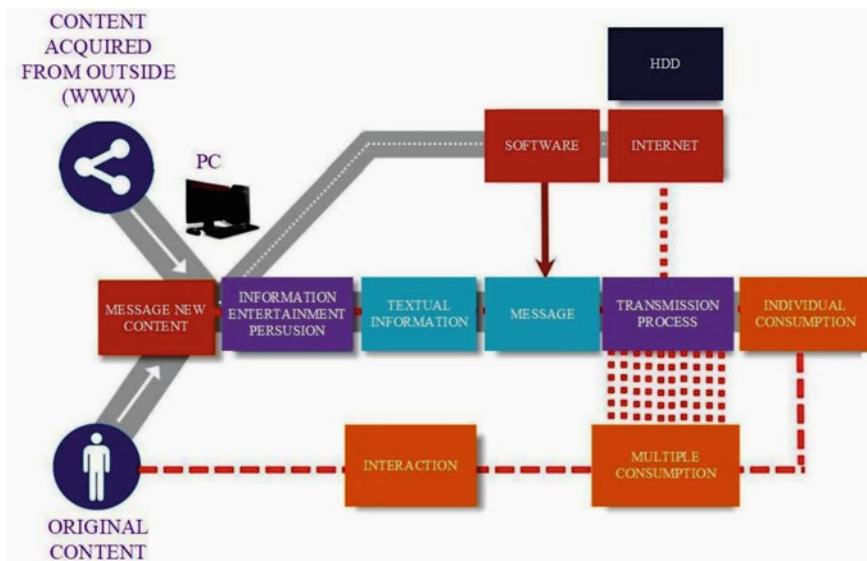


Fig. 1 Analysis of case 1: user interaction

5.2 *Second Stage: “User-Generated Content (UGC)”*

In a second phase of Web 2.0 (2005–2010), social networks (Facebook) take center stage, microblogging appears (Twitter), and sites for sharing images and videos (YouTube, Flickr, etc.) expand. This stage is still linked to the personal computer, but now it is oriented preferentially to portable computers compared to desktop computers, because of a significant increase (in wireless connections) and second-generation mobile phones. Both types of devices have clearly separate functions, since the limited data transfer capacity of cell phones (80–100 kbit/s on 2.5G cell phones) makes applications unattractive (navigation, multimedia, etc.). However, the previous success of SMS showed that the future lay in portability. Indeed, the end of the first decade of the century is characterized by a spectacular increase in connectivity thanks to the deployment of 3G networks (HSDPA allows a download speed of up to 14.0 Mbit/s) and the generalization of wireless access to broadband connections from virtually anywhere.

Likewise, hardware for mobile devices (low-consumption microprocessors, etc.) is being intensively developed. All this is the ideal context for Apple to launch the first iPhone in 2007, which represents the beginning of the era of smartphones. These small computers multiply and transform the use of applications characteristic of Web 2.0, thanks to new features: (1) connectivity (ubiquitous broadband), (2) hardware (high-performance computing with low consumption, high-resolution multi-touch screens), (3) GPS, (4) high-quality digital photography, and (5) virtually unlimited storage space (see Fig. 2).

At this stage, the user is now fully aware of the existence of the “cloud,” so he/she uses any media content as a data source to produce messages. Smart devices are now capable of managing substantial amounts of data for the production, dissemination, and communication of text messages through applications.

These messages can be stored on the platform or returned to the “cloud.” In both cases, they form part of “Web 2.0,” which manifests itself as a configuration criterion for the community of users to whom the message is addressed. However, they themselves (including the original user) consume it and regenerate it (“CGU”) iteratively (they edit it, comment on it, value it, rearrange it, manipulate it, remake it, copy it, imitate it, etc.) with applications within the framework of the “network” cooperatively and in a loop. All these operations do not occur all at once but as the message is spread socially in each community or network of users. In any case, this in-process content can always return to become part of the cloud and enter the loop of other different or higher- or lower-level networks (Fig. 2).

5.3 *Third Stage: “Intelligent Communication Society (ICS)”*

The development and use of applications for smart devices has now reached a frenetic pace (in July 2013, the application distribution platform for smart devices

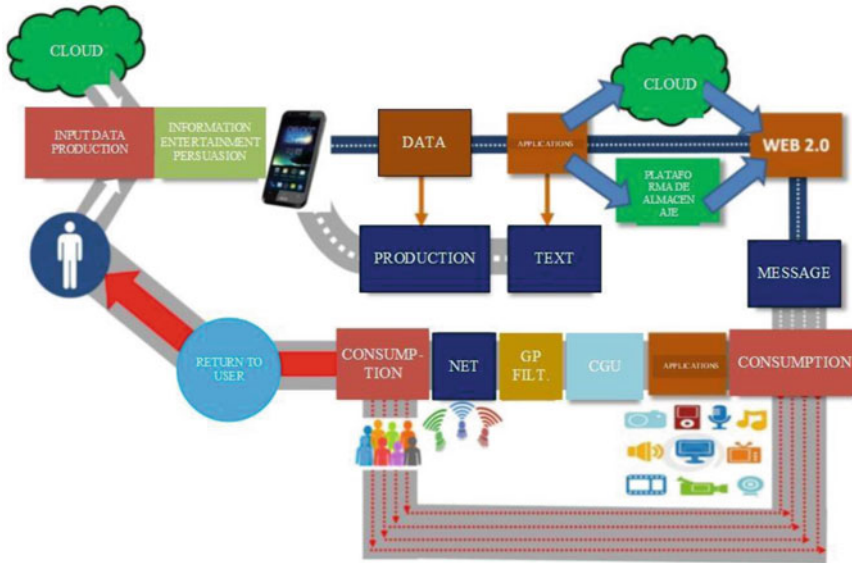


Fig. 2 Analysis of case 2: user-generated content communication process

Google Play exceeded one million applications and a total of more than 50,000 million application downloads) [20]. The technological evolution of devices, cloud computing [1], and the technological ecosystems created by large companies (application markets, development environments, the system operating Android) offer the most favorable conditions for the expansion of smart devices and their applications.

The volume of content according to the previous model multiplies very quickly. Ninety percent of the data we have has been generated in the last 2 years alone. Its discriminated, intelligent, and semantic exploitation is what is known as Big Data analysis. However, it is not easy to make predictions for the future in such a dynamic context. The technological evolution of smart devices is characterized by (1) an improvement in connectivity (greater bandwidth and coverage), (2) better screens and new forms of human-machine interaction, (3) better batteries and lower consumption, and (4) larger, faster, and cheaper storage. In this sense, head-mounted displays (HMD, e.g., Google Glass), wearables (smart watches, smart bands), Internet of Things (IoT), advanced robotics (chat bots), and autonomous vehicles or drones are part of the semantic or intelligent network (Web 3.0), characterized by networks of users that make up intelligent functional systems integrated in the activities of the human being, for which the miniaturization and automation of gadgets and the man-machine interaction. Therefore, this third stage is characterized by permanent data flows linked to the user’s day-to-day life thanks to a total connection of information and content from the “cloud” and having a higher signal quality and speed (“data”) on the smart device.

In the future, smart devices will be fully integrated into the user in their different routine activities, being able to interact with automatic systems while conducting any other task. Applications are becoming the tool to recreate received content or view it through devices. This makes the different actors in the process operate on the basis of the available content or else produce new text content using the applications to generate or edit it. Therefore, unlike the CGU, where we could more easily define the roles and location of the actors in the content production process, in the SCI these become more diffuse and certainly interchangeable. The user simultaneously performs the “producer” function and the “consumer” function of textual information generated by other participants in the process—lower part of the figure—as a receiver and as a link in the chain of meaning (Fig. 3).



Fig. 3 Analysis of case 3: intelligent communication society (ICS)

This is visualized in the two levels of flow that are interconnected through the applications as channels for sending the message or as data for a new story in any format (text, image, sound, video, or a combination of all individually or collaboratively (cross funding)) and its subsequent sending to other users or possible reprocessing producers (upper part of the figure). Therefore, we are faced with a continuous process of circulation of original messages, acquired or new (reworked), by the different users who make the message to be received and later recreated

by other users participating in the network or by the group that introduces them intentionally in the circuit.

Intelligence viewed broadly, this flow model can make the management and interaction of the individual with other individuals, social groups, administrations (management), or abstract entities a reality and with the automatic systems that govern the different routines: cities, intelligent (optimization of schedules, traffic management, weather, route optimization, etc.), smart homes (alarms, appliances, etc.), virtual work lathes, etc. Web 3.0 is the intelligent interconnection of all data that define our lives at the service of the systems in which we inevitably or voluntarily integrate and that serve our “interests” in an intelligent or meaningful way (semantics)

6 Conclusion

This article attempts to model the evolution of media content distribution in the digital age. Two phases are proposed, plus a third that is beginning to be established. Specifically, we have focused on the receiver as the core of communication processes according to the paradigm shift observed by Jianjia et al. [15] from broadcasting or mass communication to personalized communication. The first diagram explains the progressive conversion of a traditional receiver into a digital native user. The second diagram has been characterized by socialization on a global scale and in all areas of life, which has led to the personalized massification of media content according to the new online logic. Also, the third phase adds the intelligent and automatic management of network data at the service of fully individualized users. In the first phase, the Internet was the agent of change; in the second, it was the smart device; and in the third, integrated automatic devices will take center stage, be it in the form of glasses, chips incorporated into our bodies, oral intelligent navigation systems, all kinds, navigation systems incorporated into means of transport, etc.

This article theoretically explores the double perspective (communicative and technical) proposed by D'Arrigo et al. [9], who conceives the evolution of the network in three stages. The prospective approach to a current communicative model requires a theoretical review both from the most social or human proposals and from information technology itself. In line with Dylko and McCluskey, only in this way we can establish business, political, sociocultural, or even scientific strategies. Also, only in this way can new fields of study be generated. In particular, future lines of work in this area should follow the following lines: the quality and informative standards of products produced in some or all of their phases by non-institutionalized agents, currents of opinion (agenda) and the frames created by this new media reality, the formation of virtual communities, brand strategies on social networks, moral and ethical issues according to the perceived risk, the acceptance of technology and the attribution of responsibility (when they occur), and serious crises in a context in which most agents in the network are automatons and in which

the production, distribution, and even consumption of content is dominated by these systems.

It is therefore necessary to think about the new communication society based on the media content that conveys it, with the aim of promoting greater efficiency in the general interest of all. The accumulation of knowledge around current and future communication flows can have important social, economic, cultural, and moral applications.

Naturally, it is convenient to apply methodological designs different from the conventional ones to be able to study this bidirectional and dynamic interaction between users and to be able to build communicative theories that explain these new processes. This essay-type work only serves to point out working hypotheses and relevant variables of the knowledge problem. However, at present, initiatives capable of delimiting unfamiliar problems and new objects of study are needed. In this sense, the paradigm of the effects of the media continues to be applicable today, although considering the new forms that the message takes depending on the new conditions of reception. It would no longer be a unidirectional or bidirectional process but rather a process based on the automated interaction of its users, starting from a controlled initiatory message and measuring the perceptive and social effects that are produced as said interaction progresses.

References

1. Ahmed, S., & Gil-Lopez, T. (2022). Incidental news exposure on social media and political participation gaps: Unraveling the role of education and social networks. *Telematics and Informatics*, 68, 101764. <https://doi.org/10.1016/j.tele.2021.101764>
2. Apuke, O. D., & Omar, B. (2021). The ethical challenges and issues of online journalism practice in Nigeria: What do professionals and academics think? *Technology in Society*, 67, 101713. <https://doi.org/10.1016/j.techsoc.2021.101713>
3. Balmaceda, M., Högselius, P., Johnson, C., Pleines, H., Rogers, D., & Tynkkynen, V.-P. (2019). Energy materiality: A conceptual review of multi-disciplinary approaches. *Energy Research & Social Science*, 56, 101220. <https://doi.org/10.1016/j.erss.2019.101220>
4. Becken, S., Stantic, B., Chen, J., & Connolly, R. M. (2022). Twitter conversations reveal issue salience of aviation in the broader context of climate change. *Journal of Air Transport Management*, 98, 102157. <https://doi.org/10.1016/j.jairtraman.2021.102157>
5. Begany, G. M., Martin, E. G., & Yuan, X. (Jenny). (2021). Open government data portals: Predictors of site engagement among early users of Health Data NY. *Government Information Quarterly*, 38(4), 101614. <https://doi.org/10.1016/j.giq.2021.101614>
6. Ben Slama, S. (2022). Prosumer in smart grids based on intelligent edge computing: A review on Artificial Intelligence Scheduling Techniques. *Ain Shams Engineering Journal*, 13(1), 101504. <https://doi.org/10.1016/j.asej.2021.05.018>
7. Bruning, P. F., Alge, B. J., & Lin, H.-C. (2020). Social networks and social media: Understanding and managing influence vulnerability in a connected society. *Business Horizons*, 63(6), 749–761. <https://doi.org/10.1016/j.bushor.2020.07.007>
8. Cheng, W. W. H., Lam, E. T. H., & Chiu, D. K. W. (2020). Social media as a platform in academic library marketing: A comparative study. *The Journal of Academic Librarianship*, 46(5), 102188. <https://doi.org/10.1016/j.acalib.2020.102188>

9. D'Arrigo, P., Greco, A., Franchina, A. G., Ingala, S., Milluzzo, R. P., Calderone, D., Agnello, F., Legnazzi, M., Occhipinti, G., Scalia, L., Spagnolo, M., & Capodanno, D. (2021). Determinants of Popularity and Natural History of Social Media Accounts in Interventional Cardiology. *JACC: Cardiovascular Interventions*, *14*(6), 720–721. <https://doi.org/10.1016/j.jcin.2021.01.021>
10. Denson, T. F., Dixon, B. J. W., Tibubos, A. N., Zhang, E., Harmon-Jones, E., & Kasumovic, M. M. (2020). Violent video game play, gender, and trait aggression influence subjective fighting ability, perceptions of men's toughness, and anger facial recognition. *Computers in Human Behavior*, *104*, 106175. <https://doi.org/10.1016/j.chb.2019.106175>
11. Drouin, M., Sprecher, S., Nicola, R., & Perkins, T. (2022). Is chatting with a sophisticated chatbot as good as chatting online or FTF with a stranger? *Computers in Human Behavior*, *128*, 107100. <https://doi.org/10.1016/j.chb.2021.107100>
12. Han, Z., Handfield, R. B., Huo, B., & Tian, Y. (2022). Effects of power use in buyer–supplier relationships: The moderating role of communication. *Industrial Marketing Management*, *102*, 45–57. <https://doi.org/10.1016/j.indmarman.2022.01.001>
13. Hegg, J. C., Middleton, J., Robertson, B. L., & Kennedy, B. P. (2018). The sound of migration: exploring data sonification as a means of interpreting multivariate salmon movement datasets. *Heliyon*, *4*(2), e00532. <https://doi.org/10.1016/j.heliyon.2018.e00532>
14. Jackson, R. L. (2021). The neural correlates of semantic control revisited. *Neuro Image*, *224*, 117444. <https://doi.org/10.1016/j.neuroimage.2020.117444>
15. Jianjia, H., Gang, L., Xiaojun, T., & Tingting, L. (2021). Research on collaborative recommendation of dynamic medical services based on cloud platforms in the industrial interconnection environment. *Technological Forecasting and Social Change*, *170*, 120895. <https://doi.org/10.1016/j.techfore.2021.120895>
16. Lei, X., & Rau, P.-L. P. (2021). Effect of relative status on responsibility attributions in human–robot collaboration: Mediating role of sense of responsibility and moderating role of power distance orientation. *Computers in Human Behavior*, *122*, 106820. <https://doi.org/10.1016/j.chb.2021.106820>
17. Liu, D. (2022). Research on the analysis method of digital media art communication based on 3D image recognition. *Displays*, *72*, 102149. <https://doi.org/10.1016/j.displa.2022.102149>
18. Liu, Q., Wu, J., Zhou, Z., & Wang, W. (2022). Smartphone use can modify the body schema: An ERP study based on hand mental rotation task. *Computers in Human Behavior*, *128*, 107134. <https://doi.org/10.1016/j.chb.2021.107134>
19. Maleh, Y. (2021). IT/OT convergence and cyber security. *Computer Fraud & Security*, *2021* (12), 13–16. [https://doi.org/10.1016/S1361-3723\(21\)00129-9](https://doi.org/10.1016/S1361-3723(21)00129-9)
20. Maurya, S., Lakhera, G., Srivastava, A. K., & Kumar, M. (2021). Cost analysis of amazon web services – From an eye of architect and developer. *Materials Today: Proceedings*, *46*, 10757–10760. <https://doi.org/10.1016/j.matpr.2021.01.669>
21. Nasution, N. K. G., Jin, X., & Singgih, I. K. (2022). Classifying games in container terminal logistics field: A systematic review. *Entertainment Computing*, *40*, 100465. <https://doi.org/10.1016/j.entcom.2021.100465>
22. Popov, J. (2020). Boundary crossing and identity re-negotiation in internships: The integrative, future-oriented and transformational potential of interns' identity project. *Learning, Culture and Social Interaction*, *24*, 100383. <https://doi.org/10.1016/j.lcsi.2020.100383>
23. Riquelme, I. P., Román, S., & Cuestas, P. J. (2021). Does it matter who gets a better price? Antecedents and consequences of online price unfairness for advantaged and disadvantaged consumers. *Tourism Management Perspectives*, *40*, 100902. <https://doi.org/10.1016/j.tmp.2021.100902>
24. Sama, A. J., Matichak, D. P., Schiller, N. C., Li, D. J., Donnally, C. J., Damodar, D., & Cole, B. J. (2021). The impact of social media presence, age, and patient reported wait times on physician review websites for sports medicine surgeons. *Journal of Clinical Orthopaedics and Trauma*, *21*, 101502. <https://doi.org/10.1016/j.jcot.2021.101502>

25. Troise, C., Corvello, V., Ghobadian, A., & O'Regan, N. (2022). How can SMEs successfully navigate VUCA environment: The role of agility in the digital transformation era. *Technological Forecasting and Social Change*, *174*, 121227. <https://doi.org/10.1016/j.techfore.2021.121227>
26. Turnwald, B. P., Perry, M. A., Jurgens, D., Prabhakaran, V., Jurafsky, D., Markus, H. R., & Crum, A. J. (2022). Language in popular American culture constructs the meaning of healthy and unhealthy eating: Narratives of craveability, excitement, and social connection in movies, television, social media, recipes, and food reviews. *Appetite*, *172*, 105949. <https://doi.org/10.1016/j.appet.2022.105949>
27. Vollero, A., Yin, J., & Siano, A. (2022). Convergence or divergence? A comparative analysis of CSR communication by leading firms in Asia, Europe, and North America. *Public Relations Review*, *48*(1), 102142. <https://doi.org/10.1016/j.pubrev.2021.102142>
28. Vukadin, A. (2019). Chapter 3 – Why organise information about transmedia? In A. Vukadin (Ed.), *Metadata for Transmedia Resources* (pp. 69–101). Chandos Publishing. <https://doi.org/10.1016/B978-0-08-101293-2.00003-3>
29. Weston, D. (2021). Gatekeeping and linguistic capital: A case study of the Cambridge university undergraduate admissions interview. *Journal of Pragmatics*, *176*, 137–149. <https://doi.org/10.1016/j.pragma.2021.02.002>
30. Wilkinson, S., Hojckova, K., Eon, C., Morrison, G. M., & Sandén, B. (2020). Is peer-to-peer electricity trading empowering users? Evidence on motivations and roles in a prosumer business model trial in Australia. *Energy Research & Social Science*, *66*, 101500. <https://doi.org/10.1016/j.erss.2020.101500>
31. Xie, X., Tian, H., Zhou, B., & Li, K. (2021). The life-cycle development and cause analysis of large diameter shield tunnel convergence in soft soil area. *Tunnelling and Underground Space Technology*, *107*, 103680. <https://doi.org/10.1016/j.tust.2020.103680>
32. Xiong, L., Ning, J., & Dong, Y. (2022). Pollution reduction effect of the digital transformation of heavy metal enterprises under the agglomeration effect. *Journal of Cleaner Production*, *330*, 129864. <https://doi.org/10.1016/j.jclepro.2021.129864>

Sentimental Analysis of Stock Market via Twitter



S. V. S. S. Lakshmi, T. Vineetha Sai Durga, N. V. L. Tejaswi,
and A. Yeshwanth Sai Kumar

Keywords Sentimental analysis · Naive Bayes · Stock markets · Moods · Web scraping · Matplotlib library · Machine learning · Twitter

1 Introduction

Sentimental analysis is an important part of NLP as well as machine learning; it is essential in order to determine the particular “mood” for a given dataset. As the name suggests, sentimental analysis is about predicting the mood or the result of a dataset and using this result to understand certain decisions or opinions. It is used on textual data, i.e., it is used on data that are sentences in any language that have a certain meaning, such as questions, statements, and so on. There are three possible results to sentimental analysis; they are the following:

- Positive polarity
- Negative polarity
- Neutral polarity

These polarities determine the feelings behind a particular textual dataset: positive polarity means that the sentiments that are determined here are positive, which can be shown by words such as like, love, glad, and so on; negative polarity describes the negative sentiments that are shown by the statements and can be given by words such as no, not, dislike, and so on.

Neutral polarity is given for the neutral statements, i.e., the statements where it is not obvious whether the feelings conveyed are positive or negative.

For example, there is a text A which states that “I don’t think I like this new Thai restaurant”; if sentimental analysis is performed on it, it is realized by the system

S. V. S. S. Lakshmi (✉) · T. Vineetha Sai Durga · N. V. L. Tejaswi · A. Yeshwanth Sai Kumar
Department of Computer Science and Engineering, Anil Neerukonda Institutes of Technology
and Sciences, Visakhapatnam, India

that the sentiments that are being described here are negative. If there is a text B that states “I love yellow,” the sentiments are positive.

In the present day, we are surrounded by massive amounts of data, all of which can be used as feedback in order to better and advance changes. This is the reason why many businesses, educational institutions, and global markets rely on sentimental analysis to understand their customers better.

Predictions of the stock market have always been a huge research field as the market depends on it; positive feedback, negative feedback, all of it adds up to the mood of a particular stock of the day; it may also hint at future prices and data. A lot of investments are made based on predictions and form the whole basis of the global market.

1. A bank in South Africa used AI-powered sentimental analysis to extract data from 2 million texts that were taken from social media via hashtags, and they used this data to see the sentiments in the manner of aspects of the company such as branch, fees, online, etc. to maximum accuracy [2].
2. Sentimental analysis is used by many companies in order to deal with the language barrier; it was used by an Arabic healthcare company in order to analyze data in Arabic while having the capacity and speed to process around 12 million surveys to maximum accuracy [9].

2 Related Research

The vast development in technology leads to developing new tools and models for sentimental analysis. The studies said that a large number of the population follow social media (mainly Twitter) to predict customers’ attitude toward the company and daily market mood [4]. The reason the sentimental analysis is vastly used is that it performs prediction by analyzing the tweets collected from Twitter. Sentimental analysis is also used in banking, healthcare, and government sectors [5] (Fig. 1).

3 Literature Review

The most effective way to perform sentimental analysis on real-time data is to make use of a large data, i.e., using a Twitter dataset of tweets consisting of around 1,600,000 rows * 6 columns, in order to truly train the machine with all kinds of tweets and results that will occur at the end of the sentimental analysis. Each tweet is trained as such using the Naïve Bayes method, and it is done so with the entire dataset. After training the dataset, the results are checked by choosing a random tweet, and the accuracy can be tested. This paper proposes a visual tool that provides the mood of the market of the day as well as daily reports, data on Indian and foreign investors, and so on [1].

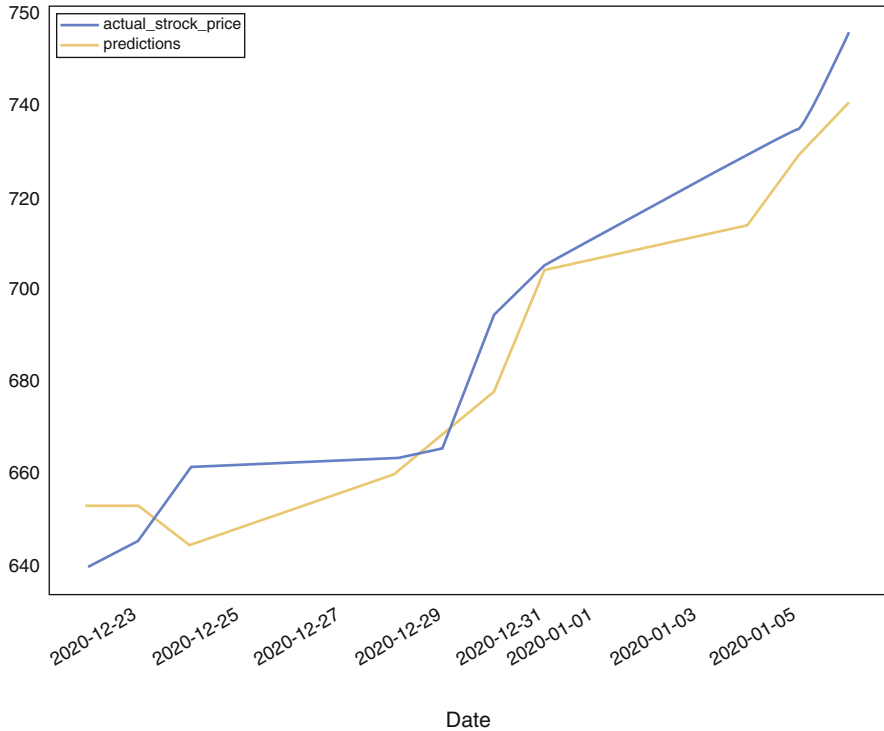


Fig. 1 A chart that displays the prediction vs the actual stock price data in real time

Sentimental analysis can be used in order to make future predictions for the market as well as the current state of the market, and therefore it is essential for anyone who might want to invest in the market. A large dataset ensures that all the outliers and unusual data can be trained as well. Many large companies can use this data as essential information regarding the future of the company while treating it as feedback from the customer (Fig. 2).

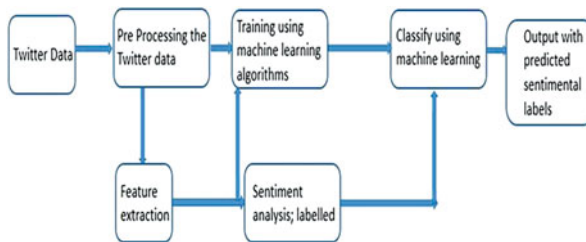


Fig. 2 Description of the proposed idea of the sentimental analysis system via tweets

4 Proposed Methodology

In this paper, the proposed idea is to provide a method with which we can predict the mood of the market for a given day using Tweets from Twitter. These tweets are collected in real time [8]. It can be used as a prediction model with which investors and companies alike can make predictions regarding the mood and price of the market, therefore, making good decisions. It will be developed using NLP as well as machine learning algorithms.

5 Data Collection

Data collection is done by using real-time tweets that are extracted from Twitter using the Twitter API [3]. The collection of the tweets is done in real time by making use of keywords. Keywords such as #Stock, #Mood, #Sensex, #Intraday, #In Trade, and #Investment and many more such words are used in order to filter the tweets that are needed for training the model to perform sentimental analysis [8]. This process not only gives us data about the stocks but also provides us with information regarding the market as well as the companies whose stocks come into picture. The keywords should be chosen with much thought and must be directly related to the stock market and its buzzwords with extreme bias. These tweets will be filtered using the keywords and, therefore, will include the names and details of the various public sector companies.

6 Data Preprocessing

The good data [3] is undoubtedly more important than good models and for which the quality of the data is more important. Due to weekends and the public holidays, the stock market is not in function. Due to this reason, the data collection remains incomplete. Stock data generally follows a concave function. If the stock value on a day is “a” and the next value is “b” with some missing values in between. The first missing value is approximated to $(b + a)/2$, and the same process is used to fill the gaps.

Since the tweets are taken from the Twitter API, it consists of many acronyms and unnecessary data like pictures and URLs, punctuations, etc. The cleanup of data occurs in multiple steps, wherein each step cleans up and polishes the data till it can be used to feed the training model. They are the following:

- The first step is to remove all punctuations, i.e., commas, full stops, exclamation marks, etc., that need to be removed as they are unnecessary data elements that will only confuse the training model. In tweets, as the data size is small, the punctuations are limited, so they are removed first.

- Tokenization is the next step and is the continuation of the removal of punctuations; the data is split into words and removes everything else but the words in the intended language.
- Removal of stop words is the next step where not all the data are relevant to sentimental analysis; they do not provide any input, so they are removed.
- All the words are converted to lower case letters.
- Stemming is the last step where all the conjugates of the words are transformed in the dataset into reliable data that can be fed to the training model.

7 Word List Generation

Word list generation is an integral part of the process of sentimental analysis. It can be done by the preparation of a personalized word list which satisfies all the needs of the problem; it can be based on POMS which stands for Profile of Mood Stocks questionnaire, which is known for covering all important requirements of the problem. Its main idea is to make a person rate their own mood by asking a certain number of questions, usually the number of questions is 65. The rating scale for POMS is from 0 to 5, 0 indicating the least stress and 5 indicating the most stress. The answers are based on the main six POMS columns, where each column indicates a particular mood. These six moods are depression, anger, tension, vigor, confusion, and fatigue. The 65 answers are mapped onto this graph, and analysis can then be performed based on the most commonly occurring answers to the given scale.

8 Implementation of the Result Data from Sentimental Analysis

As proposed in this paper, the result of the sentimental analysis will be in form of 0 and 1 s which will indicate the polarity and the statistics of the given dataset and the individual data element. Therefore, only having this result has no purpose or use, so it must be implemented into a tool or site where the results can be formulated into information that can be useful to people who want to interact and know more about the stock market.

So, in addition to the basic sentimental analysis, we propose a tool that will take this entire result and data from the sentimental analysis done using the Naïve Bayes classifier, which can be implemented using any basic programming language, and formulate it in a way that it can predict the mood of the market on a given particular day.

The tool will be a user friendly tool that can predict the mood of the market based on the result of the sentimental analysis and can be used by laymen who know not

much about the stock market but want to make investments and learn more. This tool can make predictions in real time, which makes it a valuable asset. This is the main feature of the tool. However, it also contains various essential features that will make it a one of a kind tool that can be used by everyone regardless of knowledge.

- A visualization feature which works as an aesthetic user interface as well displays the necessary data as simply and neatly as possible. This will be achieved using python and its various libraries such as Matplotlib, NumPy, pandas, and many more. It is essential that the tool is user friendly.
- This tool also has the feature to give the sentiments of the foreign and domestic investors' mood for the particular day. These include the investments that are made or will be made, how it affects the market, and other important information.
- It will contain the feature which gives the user options to query days of data of indexes as well as the prices of stocks, all of which can provide information that can be used for future predictions as well as change in investments for a particular day.
- It will also contain the feature in which the user can also query stock information if they wish to do so, as it is essential to know more about where a certain is going and to who.
- The most important feature will be the daily reports feature, which allows the users to check the purchases, changes, and other such important data.
- Most of these features can be executed using web scraping with python as well as its libraries.
- Graphical representation of the market is also an important feature of this tool.

9 Model Algorithm

Naïve Bayes classifier is one of the most efficient machine learning algorithms that are used. It can be defined as a probabilistic classifier algorithm that comes from the Bayes' theorem. It is used to classify objects or data. It uses probability in order to classify the data, i.e., it works on the relationship and independence among the columns of a given set of data. The main theorem behind Naïve Bayes classifier is that the probability of a particular event can or will change depending on the data that will be added to the expanding dataset.

Unlike the Bayes' theorem, Naïve Bayes theorem is not a singular algorithm, but it is made up of various theorems belonging to the same family of ideas. The accuracy of the Naïve Bayes increases the more it is trained, i.e., the more the data that goes through the classifier algorithm, the more accurate the classifier gets.

The algorithm can be explained by using its most common application as an example, i.e., the spam filter. At first instance, the classifier is given a dataset filled

with mails and then based on the data, it is asked to sort the spam mail from the important mail. At first instance, the classifier, not knowing what truly differentiates spam mail to important mail, may fail, but as more and more mail data is fed, it can finally accurately differentiate the spam mail to the important mail. This is how Naïve Bayes algorithm can be used.

This is how it can be essentially used to perform sentimental analysis of data. As more and more data is fed to the classifier, it can finally differentiate between positive, negative, and neutral polarity of the given dataset, with considerable accuracy. Using tweets only simplifies the process, as the tweets are usually small in size and direct to the point, so, with the help of keywords, words associated with sentiments, the Naïve Bayes approach can be used to accurately predict the sentiment of a given dataset.

As it is based on the Bayes theorem, its mathematical equation is essential to provide accurate results:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Formula 1: The main Bayes mathematical formula

where

- P(AB) is the posterior probability of class (A, target) given predictor (B, attributes).
- P(A) is the prior probability class.
- P(B|A) is the likelihood which is the probability of predictor given class.
- P(B) is the prior probability of predictor.

10 System Architecture

This paper proposes the following basic architecture (Fig. 3):

11 System Requirements

11.1 Hardware Requirements

Processor: Intel CORE i3 and above (recommended because of high processing speed)

Memory: 250 GB ROM, 2 GB RAM (data is downloaded and stored in hard disk)

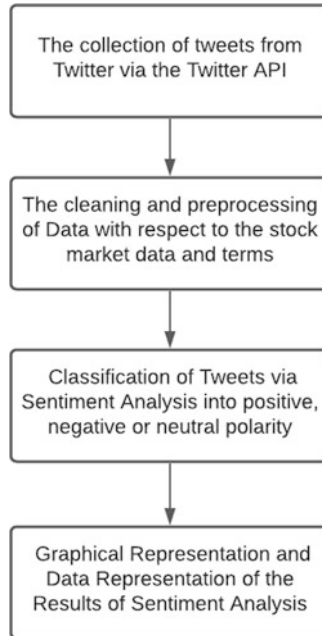


Fig. 3 The flow chart representing the architecture of the proposed system

11.2 *Software and Tools*

Operating Systems: Ubuntu 16.4, Windows

Programming Languages: Python

APIs: Python, Libre Office, Machine Learning, Natural Language Processing (NLP), Twitter API

12 Datasets

1. Tweets collected in real time as and when required
2. Stock information collected using magazines and articles as well as using the market analysis data that is available to the public

13 Evaluation Methods

1. Measure correlation between volume of tweets vs. change in stock price, sentiment of tweets vs. change in stock price, volume of news articles vs. change in stock price, sentiment of news article vs. change in stock price.

2. Checking accuracy by checking whether or not sales and purchases were done as predicted and the effect on the mood of foreign and domestic investors [8].

14 Conclusion

Sentimental analysis is an integral part of modern technology where the abundance of data only ensures that information can be taken from it and is put into bettering technology and certainly our lives. Sentimental analysis can analyze data and give necessary feedback for the given data which can be utilized by companies in order to either advertise or better their products and so forth [6]. The usage of sentimental analysis in stocks has only increased in value, as it was very difficult to predict the mood of the market. Now, using this method, the future of the market can be analyzed, and necessary actions can be taken in order for profits. The calculations are done in such a manner that a particular day at the market has a lot of positive polarity buzz, and then it can safely be assumed to be a positive day to invest in the market [7]. Similarly, if the market has a much more of a negative polarity, decisions can be made accordingly. The most is to be done in order to make sure that the result is as accurate as possible. The more the data is fed to the model, the more the accuracy rates increase.

References

1. Saurabh Singh, "Twitter Sentiment Analysis Using Machine Learning", IJRCSEIT, 2020
2. K L Shreyas Kumar, Akshaya K M, Suhas S, "Sentimental Analysis of Twitter for Stock Market Investment", International Research Journal of Engineering and Technology, 2019
3. Twitter sentimental analyze dataset from Kaggle
4. www.nseindia.com
5. www.investopedia.com
6. Xing Fang, Justin Zhan, Sentimental analysis using product review data, SpringerOpen, 2015
7. Shri Bharathi. Sv, Angelina Geetha, Sentimental analysis for effective stock market prediction, InternationalJournalofIntelligentEngineeringandSystems 10(3):146–154, 2017
8. www.dsji.com
9. Vishal, A. Kharde, S.S. Sonawane, Sentiment Analysis of Twitter Data: A Survey of Techniques, International Journal of Computer Applications (0975 – 8887), 2016

Detection and Classification of Tumor Tissues in Colorectal Cancer Using Pathology Images



Ponnarasee B. K and Lalithamani N

Keywords Medical image analysis · Pathology images · AI models · Image processing · Transfer learning

1 Introduction

The colorectal disease is positioned second among the most widely recognized passing causing malignant growths around the world. The World Health Organization assessed that the subsequent driving reason for death before 70 years old in 112 of 183 nations is malignant growth [1].

The meaning of disease occurrence and mortality is becoming overall quickly. Colorectal malignant growth is the second-driving demise causing disease after lung and bosom malignant growth, separately. In 2020, more than 1.9 million new instances of colon disease and 935,000 deaths were recorded [2]. Overall, colorectal malignant growth positions third in frequency and second in mortality. The indicative cycle is trying because of scant analytic assets in all well-being frameworks, distinguished utilizing colonoscopy, which is viewed as an intrusive system.

Colorectal disease is analyzed by different cycles, for example, symptomatic colonoscopy, proctoscopy, biopsy, endorectal MRI, CT examines, and so forth; the best strategy for analysis is endoscopy, where the assessments permit limiting the growth and its sorts in the digestive organ for histological assessment. Screening and earlier recognition of colorectal malignant growth are very treatable. Early-stage detection of growth tissue type is a decent method for treating the disease at a prior stage or eliminating that specific tissue before it is maneuvered toward the thick disease tissues.

Ponnarasee B. K · Lalithamani N (✉)

Department of Computer Science and Engineering, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore, India

e-mail: cb.en.p2cse20022@cb.students.amrita.edu; n_jalitha@cb.amrita.edu

The biopsied tests are examined under the magnifying lens by a very much concentrated pathologist with numerous long periods of preparing experience. Experiences procured from the minute review are all-around detailed and given to particular specialists in worry of additional therapy on disease patients. Computerization of this manual pathologist concentrates on process utilizing picture handling, and a man-made consciousness model can provide analysis with great exactness in less calculation time. A pathologist will want to get boundaries about the cancer tissue from sectioned pictures and characterization of the sort of tissue development utilizing AI models. Earlier stage recognition and tissue development investigation in colorectal disease make ready for more outrageous therapy than different tumors and bring down death rates when thought about an obtrusive technique.

The order of pathology cancer tissue pictures utilizing computerized reasoning models does not just work on the exhibition and productivity of the arrangement but additionally permits doctors to make early, helpful choices as far as a clinical treatment. As per the development of tissues and cores in the body, they watch out for different kinds of growth tissue classes of colorectal disease. They are significantly characterized into various distinct sorts: fat tissue and bodily fluid, stroma and muscle, colorectal disease, epithelial tissue, and stomach malignant growth epithelial tissue. This examination comprises an efficient investigation of pathology in-age examination and transfers learning, where profound learning and AI models are joined for the arrangement of pathology pictures [3].

2 Literature Survey

Xing et al. [4] in their exploration portray the significance of profound learning in microscopy picture examination, difficulties, and possible future patterns in microscopy picture investigation involving profound learning and current profound learning accomplishments in identification, division, and grouping of microscopy picture investigation syntax.

Kather et al. [5] in their research study detail how the instability potential of micro-satellite using deep learning methods in determining the respond of patients toward the immunotherapy is analyzed. The convolutional neural network was trained as a tumor detector for STAD and CRC. The various steps of tile separation and sorting among MSI and MSS, scaling are processed. Finally, the network model was trained to classify MSI vs. MSS.

Lizuka et al. [6] describe the importance of AI in pathological diagnosis services. Further, they applied and demonstrated that deep learning models provide high promising potential of deployment in histopathological diagnosis of colonic tumors through the research on classifying gastric and colonic tumors using biopsy histopathological whole slide images of stomach and colon with convolution neural network and recurrent neural network.

Sena et al. [7] portray that picture naming is a tedious undertaking that is inclined to botch because of human interruption. The review on profound learning algorithmic approach to perceive four unique phases of disease tissue advancement illustrated that the brain network with appropriate preparation can give a precise marking to colon malignant growth with worked on quality and speed in clinical picture analysis with exactness more prominent than 80%. The model assembled includes Adam Optimizer, and Softmax is picked for misfortune work estimation. This model can help pathologists lessen the weight and recurrence of blunders.

Du et al. [8] in their review proposed a model which learns the essential elements of CNN techniques and beats high-quality elements and naturally distinguished epithelial and stromal districts in the bosom. His exploration is with bosom malignant growth and colorectal pathology pictures. What's more; he likewise showed that colon cancers could be separated from growth tissue utilizing an organization architecture layer approach with results that were 84% precise.

Morkunas et al. [9] in their study provided a grading system that classifies tumor cell abnormalities and predicts development in the automated identification of tumor tissue using colorectal cancer whole slide images. The parameters which are used for grading the tissues in this research are color and texture analysis. They also discussed several possibilities to achieve epithelium-stroma classification by utilizing clarified superpixels to prepare AI models. On the other hand, to get ready convolutional mind association, stamped superpixels are used to create square picture patches by moving fixed-size window around each superpixel centroid. The proposed procedure deals with the course of ground truth data combination and should restrict the time spent by a talented expert to perform manual remark of whole slide pictures.

3 Findings

As indicated by the Indian Association of Pathologists and Microbiologists, there are just 4500 pathologists authorized with enrollment. The discoveries from the pathology study on histology picture assume an imperative part in the treatment and further procedure with medico. Pathology audits are not normalized; it might change starting with one then onto the next. No examination has been done as such far to get to the kind and degree of mistakes and the subsequent impact on malignant growth care on-premise of the pathology survey concentrate on the report. Varieties found in the digitalization of pathology examples like luminance variety, stain variety, mark variety, and miss outs [10]. Still, restrictions in the accessibility of appropriate datasets individual to specific malignant growth illnesses are present.

AI models can be coordinated and planned by the attainable quality of datasets. The preparation and approval of specific information can be applied with a norm and by and large precision kept up with for all examples. Picture handling techniques will give an extraordinary degree of progress in the picture as per the articles like division, shading standardization, improvement, and so on. By coordinating picture handling and AI models, we are ready to robotize the investigation discoveries or investigation of the pathology disease picture and give a period and assets oversaw mechanized arrangement with great precision.

4 Proposed Model

The architecture for the detection and classification of tumor tissues in colorectal cancer systems consists of both image processing and artificial intelligence model integrated to build an automated model. Figure 1 depicts the different stages involved in this architecture. The histology colorectal cancer images undergo several image processing techniques such as image enhancement, stain normalization, and nuclei segmentation. Later, the segmented image is impacted by the model drive. In this proposal, the transfer learning approach is used to develop the AI model. The VGG16 model is mapped out as the feature extractor. The data is spilt into training and testing data of about 80% and 20%, respectively. The extracted features from the VGG16 model are fed to several machine learning algorithms for training and testing. Random forest, support-vector machine, and artificial neural network are a portion of the AI model that has been applied. Finally, a comparative analysis of these models is carried out to figure out the best-performing model.

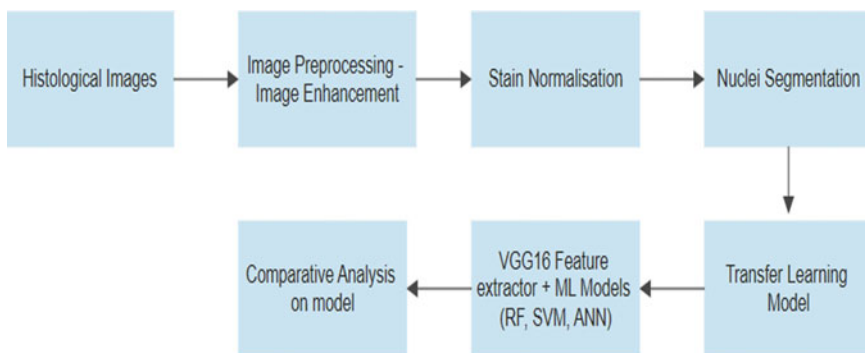


Fig. 1 Colorectal cancer screening and tissue categorization architecture

5 Methodology

5.1 Histological Images

These are the biopsy specimen that are fixed onto the glass slides and are digitally recorded after the observation. The microscopic tissue patch is stained with hematoxylin and eosin. The dataset contains 1500 patches of images categorized into loose nontumor tissue (LNT), dense nontumor tissue (DNT), and tumor tissues (TT) [11]. The categories are almost balanced equally in numbers. All images are 512×512 px at $0.5 \mu\text{m}/\text{px}$.

5.2 Image Processing

Improvement in the nature of the pathology pictures clears a method for removing highlights in a better manner. Along these lines, picture overhaul is applied here. In this iteration [12], the Contrast-limited adaptive histogram equalization (CLAHE), deals with the foundation of above amplification of the contrast and the more modest locals in the images are applied rather than whole image. In digitalized pathology pictures luminance channel is concerned. So CLAHE functions admirably on these shading pathology pictures. The first versus upgraded pathology fix picture is portrayed in Fig. 2.

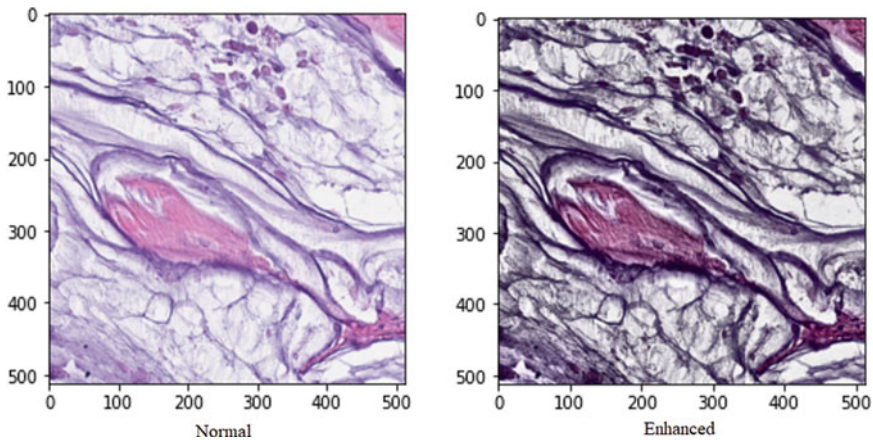


Fig. 2 Normal patch image versus enhanced patch image

5.3 Stain Normalization

The pathology pictures are stained with hematoxylin and eosin (H and E) stains. H is the blue stain that holds nuclei insights, and E is the pink stain that holds other cell substances. Using stain normalization, H and E stains are separated using separating color channels. H stain which holds nuclei features that are used for further process in this study is depicted in Fig. 3.

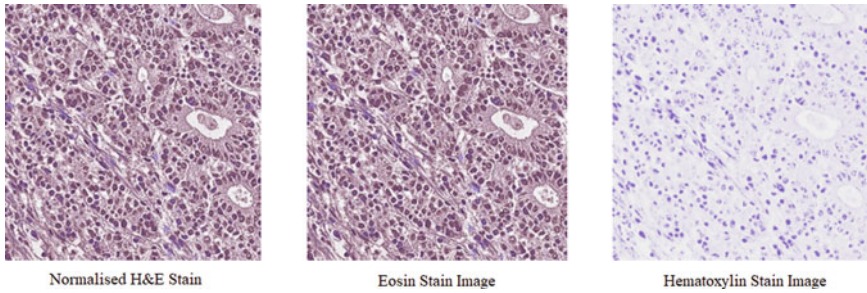


Fig. 3 Normalized stain content patch image, eosin stain content patch image, and hematoxylin stain content patch image

5.4 Nuclei Segmentation

The tumor tissues are classified into types and grades on basis of the cell or tissue growth. The segmentation of the nuclei in the pathology images provides an easy, time-saving, and good analysis for the pathologist. Otsu and watershed segmentation is applied for nuclei segmentation [13, 14]. Otsu's calculation performs bunching in light of the limit of the picture to binary picture as displayed in Fig. 4.

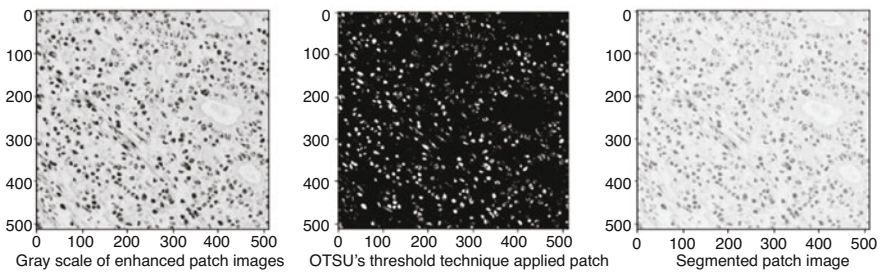


Fig. 4 Gray scale of enhanced patch images, Otsu's threshold technique applied patch, and segmented patch image

The nuclei segmentation algorithm is characterized as per the following:

1. On-premise to the edge, the resulting pixel is set apart as white (foreground), and assuming it is more noteworthy or equivalent to the limit, then, at that point, it is set apart as dark (foundation). The white spots in Fig. 4 address the core's cell division in the pathology pictures.
2. The foreground characteristics are fed into the watershed algorithm so that overlapping cells can be segmented and analyzed.
3. Watershed algorithm works on the principle of marker attributes that holds local minima in a gradient of the image [15]. The obtained segmented image through watershed segmentation technique is depicted in Fig. 4. Here, we are able to interpret the image into an analysis such that according to the textures the mapped area shows the nuclei manipulation at the mass cluster. So, this tissue type can be interpreted as tumor tissue class.

5.5 *Transfer Learning*

In our study, the integration of the pretrained deep learning model of VGG16 and machine learning algorithms is applied [16, 17]. The VGG16 is defined as a feature extractor that does not include dense layers in its structure; the segmented images are rescaled to values between 0 and 1. The extracted features are fed to the machine learning models for the classification of tumor tissue types in colorectal cancer [18].

5.6 *Machine Learning Models*

The pathology image dataset is separated into train and test data with respect to 80% and 20%. The train instances are utilized to prepare the ML models, and test instances are utilized to test the prepared ML model.

Support Vector Machine

The SVM model also known as a large margin classifier can be applied when the number of features is high. The model will in general dissect and group the instances. The work based on mapping the information into a higher aspect space is finished using Kernel capacities. Doyle et al. introduced a quantitative mechanized framework for separating the non-harmful and dangerous pictures of bosom cancer and afterward grouped the destructive pictures into the low-and high-grade values [19]. The extracted features can be accounted to colorectal cancer tissues classification

Random Forest

The random forest model builds n number of trees as per the defined parameters. Every tree gives a unit vote, and the final decision is termed according to the most probable class label. The fast and robust model doesn't possess an overfitting problem [20].

Artificial Neural Network

ANN model comprises various neuron layers such as an input layer, output layer, and hidden layer. The parameters such as activation rate and weights are defined over the connected layers during training of the model. Forward and backward propagation takes place for training the model, and classification is performed. We identified and compared the differences in the diagnostic histological images [21]. On the other hand, to plan organization, checked super pixels are used to deliver square picture patches by moving fixed-size windows around each super pixel centroid. The proposed technique chips away at the course of ground truth data arrangement and should restrict the time spent by a skilled expert to perform manual remarks of whole slide pictures

5.7 Metrics for Evaluation

The metrics considered for estimating the classification algorithm performance are as follows:

1. Confusion matrix: A lattice portrayal of the expected results that depict the exhibition of the arrangement on a bunch of test instances.
2. They are related as follows: true positive (TP), predicted true and true in existence; true negative (TN), predicted false and false in existence; false positive (FP), predicted true and false in existence; false negative (FN), predicted false and true in existence.
3. Accuracy: Accuracy denotes the percentage of the rightly classified class. Accuracy is obtained by $\text{Accuracy} = (\text{TP} + \text{TN})/\text{Total}$.
4. Precision and recall: Recall gives us the true positive rate (TPR), which is the ratio of true positives to everything positive. Precision and recall are given by $\text{Precision} = \text{TP}/\text{predicted yes}$; $\text{True Positive Rate} = \text{TP}/\text{actual yes}$.

6 Result and Discussion

This research proposed the solution which provides an automated model to detect and classify the tumor tissue types in colorectal cancer. The pathologist is able to

obtain properties of the pathology images from the segmented image and region props result obtained for the patch image. The biopsy pathology patch image contains various tissue and cell elements. The classification of tumor tissue type is based on the content, patterns identified in the patch image. If the pathologist needs any further insights on the pathology image, they can use the segmented image for further study.

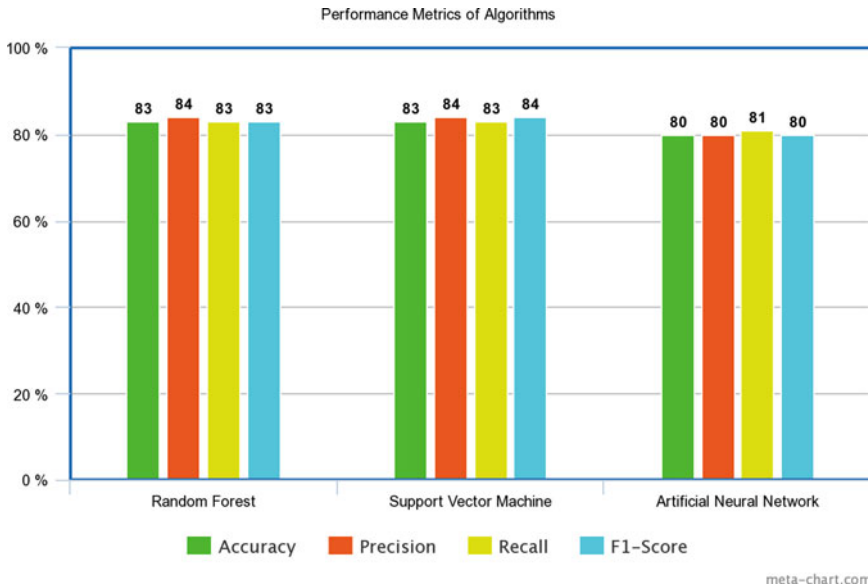


Fig. 5 Execution proportion of random forest, support-vector machine, and artificial neural network

The quantity of pathology images used for training and testing is 1200 and 300, respectively. The test data of about 104, 86, and 110 images are taken randomly from each type of tumor tissue DNT, LNT, and TT, respectively. In 104 DNT images, 81 are correctly classified; out of 86 images, 75 are correctly classified as LNT; out of 110 images, 94 are correctly classified; and the remaining are mistakenly classified into other classes in random forest algorithm. The random forest and SVM algorithm give accuracy performance of about 83%, and ANN is about 80%. Figure 5 depicts the performance metrics. Three-classification algorithm performs with good accuracy. RF and SVM are superior than ANN. The pathology image is uploaded. The model predicts the classification of that particular tumor tissue of colorectal cancer. The result obtained is depicted in Fig. 6. The label TT describes the label tumor tissue.

```
The Colorectal Cancer Tumor Tissue Type is : ['TT']
<matplotlib.image.AxesImage at 0x13595ce3d90>
```

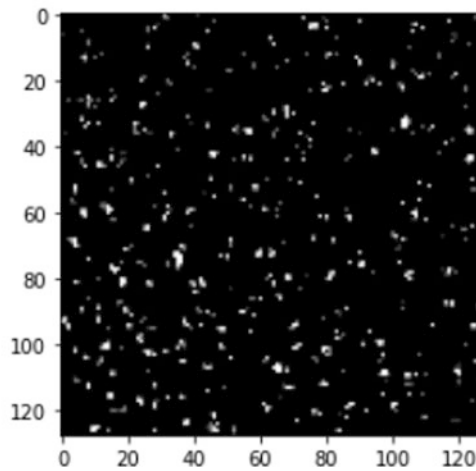


Fig. 6 Predicted category of the tumor tissue image

7 Conclusion and Future Discussion

By this method, colorectal cancer tumor tissue types can be identified at the prior stage in minimal time, and the manipulation of the cells can be controlled and treated if the tissue is classified appropriately without any delay. The automation in the detection and classification of tumor tissue in colorectal cancer diagnosis using transfer learning technique provides an improvised classification of tumor tissues. This approach can fundamentally uphold the exact location and grouping of growth tissues in colorectal disease with negligible calculation exertion. Future work can be drawn out by developing a web application that manages all digitalized pathology images in the database management; the AI model can be integrated with the web application; it will be an end-to-end automated application. The pathologist will be able to classify, detect, segment, and analyze the features extracted and possible mislabeled classification, and statistics visualization can be obtained from the single platform.

References

1. Siegel, Rebecca L., et al. "Colorectal cancer statistics, 2020." *CA: a cancer journal for clinicians* 70.3 (2020): 145–164.
2. Mazidimoradi A, Hadavandsiri F, Momenimovahed Z, Salehiniya H. "Impact of the COVID-19 pandemic on colorectal cancer diagnosis and treatment: a systematic review". *Journal of Gastrointestinal Cancer*. 2021 Nov 29:1–7.

3. Cui M, Zhang DY. Artificial intelligence and computational pathology. *Laboratory Investigation*. 2021 Apr;101(4):412–22.
4. Xing, Fuyong, et al. “Deep learning in microscopy image analysis: A survey.” *IEEE transactions on neural networks and learning systems* 29.10 (2017): 4550–4568.
5. Kather, Jakob Nikolas, et al. “Deep learning can predict microsatellite instability directly from histology in gastrointestinal cancer.” *Nature medicine* 25.7 (2019): 1054–1056.
6. Iizuka, Osamu, et al. “Deep learning models for histopathological classification of gastric and colonic epithelial tumours.” *Scientific reports* 10.1 (2020): 1–11.
7. Sena, Paola, et al. “Deep learning techniques for detecting preneoplastic and neoplastic lesions in human colorectal histological images.” *Oncology Letters* 18.6 (2019): 6101–6107.
8. Du, Yue, et al. “Classification of tumor epithelium and stroma by exploiting image features learned by deep convolutional neural networks.” *Annals of biomedical engineering* 46.12 (2018): 1988–1999.
9. M. Morkunas, P. Treigys, J. Bernatavičienė, A. Laurinavičius, G. Korvel, Machine learning based classification of colorectal cancer tumour tissue in whole-slide images, *Informatica* 29 (1) (2018) 75–90.
10. Corvò A, Westenberg MA, Wimberger-Friedl R, Fromme S, Peeters MM, van Driel MA, van Wijk JJ. Visual Analytics in Digital Pathology: Challenges and Opportunities. In *VCBM 2019* (pp. 129–143).
11. J. N. Kather, Histological images for tumor detection in gastrointestinal cancer (Feb. 2019). <https://doi.org/10.5281/zenodo.2530789>.
12. Sengoz N, Yigit T, Ozmen O, Isik AH. Importance of Preprocessing in Histopathology Image Classification Using Deep Convolutional Neural Network. arXiv preprint arXiv:2201.09867. 2022 Jan 24.
13. Hayakawa, Tomohiro, et al. “Computational nuclei segmentation methods in digital pathology: a survey.” *Archives of Computational Methods in Engineering* 28.1 (2021): 1–13.
14. L. S. Nair, R. P. R. G. Sugathan, K. V. Gireesh and A. S. Nair, “Mitotic Nuclei Detection in Breast Histopathology Images using YOLOv4,” 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), 2021, pp. 1–5, <https://doi.org/10.1109/ICCCNT51525.2021.9579969>.
15. V. K. Srinivasalu, K. Pavithran, et al., Biomarkers in colon cancer and its clinical implications, *Journal of Current Oncology* 3 (2) (2020) 66.
16. Saber A, Sakr M, Abo-Seida OM, Keshk A, Chen H. “A novel deep-learning model for automatic detection and classification of breast cancer using the transfer-learning technique”. *IEEE Access*. 2021 May 11;9:71194–209.
17. Jayaram K., Gopalakrishnan P., Vishakantiah J. (2022) Abstract and Image Analysis of High-Temperature Materials from Scientific Journals Using Deep Learning and Rule-Based Machine Learning Approaches. In: Kumar A., Senatore S., Gunjan V.K. (eds) *ICDSMLA 2020. Lecture Notes in Electrical Engineering*, vol 783. Springer, Singapore. https://doi.org/10.1007/978-981-16-3690-5_43.
18. R. P S and R. V, “Identification of Colorectal Cancer in pathological images Using CNN Algorithm,” 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC), 2021, pp. 1358–1363, <https://doi.org/10.1109/ICESC51422.2021.9532919>.
19. C. Kaushal, S. Bhat, D. Koundal, A. Singla, Recent trends in computer assisted diagnosis (cad) system for breast cancer diagnosis using histopathological images, *IRBM* 40 (4) (2019) 211–227.
20. B. M. Boban and R. K. Megalingam, “Lung Diseases Classification based on Machine Learning Algorithms and Performance Evaluation,” 2020 International Conference on Communication and Signal Processing (ICCSP), 2020, pp. 0315–0320, <https://doi.org/10.1109/ICCSP48568.2020.9182324>.
21. Altini, Nicola, et al. “Multi-class Tissue Classification in Colorectal Cancer with Handcrafted and Deep Features.” *International Conference on Intelligent Computing*. Springer, Cham, 2021.

Challenges Encountered in the Implementation of Machine Learning in the Healthcare Industry



Rita Roy , Subhodeep Mukherjee , Manish Mohan Baral ,
Ajay Kumar Badhan, and Marada Ravindra

Keywords Healthcare · Machine learning · Challenges · Deep learning · Algorithm

1 Introduction

According to Russell Reynolds and Associates, international healthcare costs, which are estimated to be between \$6 and \$7 trillion, are expected to exceed \$12 trillion in just 7 years [1]. Internationally, in the United States, this trend is exemplified. The total healthcare cost in the United States has risen to \$3 trillion, increasing 5.3% [1]. The substantial disparity between healthcare costs is reflected in this federal funding increase. The present business is going through some critical changes, and the first is the shift to patient-focused consideration. Authoritative changes have shifted away from medical clinic-based review and toward deterrent and outpatient-based thinking as medication approaches the zenith of its consumerist evolution [2].

The inability to set up a decent connection between the well-being supplier and the patient is a principal issue in the model above and the past medical services framework. Patients' fulfilment with care, drug reaction, and probability to plan follow-up arrangements are impacted by how a current doctor speaks with them.

R. Roy (✉)

Department of Computer Science and Engineering, GITAM Institute of Technology, GITAM (Deemed to be University), Visakhapatnam, Andhra Pradesh, India

S. Mukherjee · M. M. Baral

Department of Operations, GITAM SCHOOL OF BUSINESS, GITAM (Deemed to be University), Visakhapatnam, Andhra Pradesh, India

A. K. Badhan

Department of Computer Science, Lovely Professional University, Phagwara, Punjab, India
e-mail: ajay.27337@lpu.co.in

M. Ravindra

Infosys, Hyderabad, Telangana, India

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

R. Misra et al. (eds.), *Machine Learning and Big Data Analytics*,

Springer Proceedings in Mathematics & Statistics 401,

https://doi.org/10.1007/978-3-031-15175-0_31

Specialists now have numerous patients to guarantee that they are solid even after the performance [3]. Specialists are likewise being approached to see a more critical number of patients in a more limited period. Specialists should utilize innovation to accomplish their objective of patient-focused consideration. Information across areas is filling dramatically in the present associated world. New and inventive approaches to convey and remove significance arise [4]. Human mediation has been utilized to parse through information; this is wasteful and leaves many personal examples in the information unseen [5].

This paper addresses the challenges faced by the healthcare sector in the adoption of ML. In healthcare, ML is becoming more broadly used, and it is helping health professionals and patients in a variety of ways. The most common applications of ML in healthcare are hospital billing automation, decision support, and the development of healthcare guidelines. As a result, ML aids in storing, selecting, and reformatting data. The effectiveness of ML-based automatic detection and diagnosis systems has been comparable to that of a skilled radiologist.

The rest of the paper is arranged as follows: section two discusses the survey, section three discusses the various methods, section four presents the result and discussion, and section five provides the conclusion.

2 Literature Review

2.1 *Machine Learning in Healthcare*

The medical services industry is one region where AI will have broad social ramifications [6]. AI might be the response to bringing down medical care costs and further developing the patient-specialist relationship. AI and large information arrangements can be utilized for different well-being-related purposes, incorporating helping specialists grow more customized solutions and therapies for patients and assisting patients in deciding when and whether they need to plan follow-up arrangements [7]. A considerable measure of information has become accessible in the medical services field. This incorporates EMRs that contain both organized and unstructured information [8]. Organized well-being information will be data that is easy to order in a data set; it might incorporate an assortment of measurements and classifications, like patient loads, temperatures, and surprisingly conventional manifestations, for example, migraines and stomach torment [5, 9]. Most clinical information is unstructured, comprising notes, reports, release rundowns, photos, and sound and video accounts [10]. A supplier and a patient discussion are hard to evaluate and sort since it is profoundly customized and can go differently [11, 12]. For instance, the conversation and information for two patients with a similar virus strain will fluctuate contingent upon their experience, the specialist's experience, and, surprisingly, how the patient depicts their indications [13]. Large, organized information makes up about 20% of existing EMRs, while unstructured information makes up 80%.

Because medicine is stored, modern ML methods must focus on coordinating and forming relationships between large quantities of unstructured, raw data [14]. The capacity to gather and investigate this information for an enormous scope will assist with utilizing AI advances in the medical care field [15]. Numerous fake information advances exist for organized details right now; in any case, just a minor level of existing trailblazers is centered around organized information, zeroing in on the stories found in the medical care industry. When utilized accurately, AI can help specialists make exceptional close determinations, select the best medications for their patients, recognize patients at great danger of helpless medicine results, and further develop patients' generally speaking well-being while at the same time bringing down costs [16]. AI has been displayed to diagnose patients and recognize more inclined to repeat infections. Moreover, almost 90% of trauma center visits can stay away [17].

2.2 Challenges Faced by Machine Learning in the Healthcare Sectors

Large Volume of Data (LVD)

ML is a term that refers to a group of computational models that are incredibly time-consuming. Fully connected multilayer neural networks are a good example, as they require estimating a large number of network parameters [6]. ML is a term that refers to a group of computational models that are incredibly time-consuming. Fully connected multilayer neural networks are a good example, as they require estimating many network parameters [15].

H1: LVD creates a challenge for the adoption of ML in healthcare.

Quality of Data (QD)

In contrast to other areas where data is clean and structured, healthcare data is highly heterogenous, ambiguous, noisy, and incomplete [18]. Training a good ML model with these large and diverse data sets requires several factors to be considered [3].

H2: QD creates a challenge for the adoption of ML in healthcare.

Temporality (TEM)

Diseases constantly evolve and change in unpredictable ways over time [19]. Designing ML methods to manage temporal healthcare data is a critical component that will create new solutions.

H3: TEM creates a challenge for the adoption of ML in healthcare.

Domain Complexity (DC)

The diseases are highly diverse, and we still don't know everything there is to know about most of their causes and progression [20]. Furthermore, in a real-life clinical situation, the number of patients is generally limited, and we can't ask for as many as we want [21].

H4: DC creates a challenge for the adoption of ML in healthcare.

Interpretability (INT)

Although machine models in several applications have been efficient, they are often treated as black boxes [17, 22, 23]. Indeed, model interpretability (i.e., the determination of which phenotypes drive forecasting) is essential to persuade medical workers to take the measures suggested by the predictive system (e.g., a medical prescription).

H5: INT creates a challenge for the adoption of ML in healthcare.

3 Research Methodology

The respondents were selected from the employees who had IT information of their healthcare firms. The experiences of the respondents were considered during the questionnaire distribution. The survey was conducted in an online mode. The survey was conducted primarily on the private hospitals of India. The respondents were having knowledge of ML and its applications in the healthcare sector. The respondents were asked permission for the survey; only then the questionnaire was sent to the respondents. The questionnaire clearly mentioned that the collected information will not be used for other purposes except this study. The questionnaire was sent to 556 respondents out of which only 367 respondents returned the filled questionnaires. However, out of 367 questionnaires, only 328 respondents' responses were used for the study. After data cleaning in SPSS 20.0, the usable responses used for the analysis were 328. The single-factor Harman test is performed to check the biasness of the data, and the result is 33.107% which is below the recommended value of 50% [24]. Table 1 shows respondents' distribution based on gender, respondent's current position, and years of online experience.

Table 1 Demographics of the respondents

Item	Frequency	Percentage
Gender		
Male	226	69
Female	102	31
Years of experience using online services		
<3 years	142	43
3–5 years	97	30
>5 years	89	27
Respondents' current position		
Doctors	144	44
IT staff	184	56

Source: Results obtained from the survey of the respondents

4 Data Analysis

To test the reliability of the data in all constructions, Cronbach alpha was utilized [25]. All dimensional scales must exceed the recommended 0.70 level [26]. For all components, also composite reliability (CR) was measured. Table 2 shows the values of Cronbach alpha and composite reliability. It is measured by its ability to produce better results for internal coherence [27].

Table 2 Cronbach's alpha and composite reliability

Constructs	Cronbach's alpha	Composite reliability
LVD	0.836	0.889
QD	0.845	0.887
TEM	0.733	0.851
INT	0.863	0.896
DC	0.789	0.907

Source: Results obtained from SPSS software

Exploratory factor analysis (EFA) is performed. Kaiser-Meyer-Olkin (KMO) value for the current research is 0.862. The minimum level set for this statistic is 0.60 [28]. The rotated component matrix is significant for interpreting the results displayed in Table 3.

Table 3 Rotated factor matrix

Pattern matrix					
	Component				
	1	2	3	4	5
LVD1			0.833		
LVD2			0.774		
LVD3			0.791		
LVD4			0.866		
QD1		0.774			
QD2		0.841			
QD3		0.868			
QD4		0.771			
TEM1					0.858
TEM2					0.883
TEM3					0.677
DC1				0.866	
DC2				0.955	
DC3				0.797	
INT1	0.822				
INT2	0.839				
INT3	0.865				
INT4	0.777				

Source: Results obtained from SPSS software

A sensible critical plan to survey the legitimacy of an action to foster convergent validity (CV). CV is how much a test measures the idea or advancement expected to evaluate. For the most part, CV is attempted by assessing the evaluations’ relationship from a couple of scales [29]. Table 4 shows the construct correlation and average variance extracted (AVE). No cut-off describes CV [30].

Table 4 Construct correlation and AVE

	AVE	Variance extracted between factors				
		INT	QD	LVD	DC	TEM
INT	0.829	1				
QD	0.812	0.673	1			
LVD	0.817	0.677	0.663	1		
DC	0.868	0.720	0.706	0.711	1	
TEM	0.805	0.667	0.653	0.658	0.701	1

Source: Results obtained from AMOS software

Hence, there are no validity concerns, and further analysis is performed utilizing the SEM approach. Figure 1 shows the SEM model for the challenges of ML in healthcare. It shows five independent variables and one dependent variable. This model tested the five challenges that are taken for the study. Five proposed hypotheses were accepted for the study. The independent variables taken in the study were LVD, *large volume of data*; QD, *quality of data*; TEM, *temporality*; DC, *domain complexity*; and INT, *interpretability*. The dependent variable taken for the study was CAML, challenges for adoption of ML in healthcare sector. The results of the path estimates are shown in Table 5. The results show the five hypotheses that P-value supports [31]. The multi-field correlations (R2) help measure how easily a regression line estimates actual data points between 0 and 1 [32]. The 67.3% variance of CAML can be explained by the model proposed.

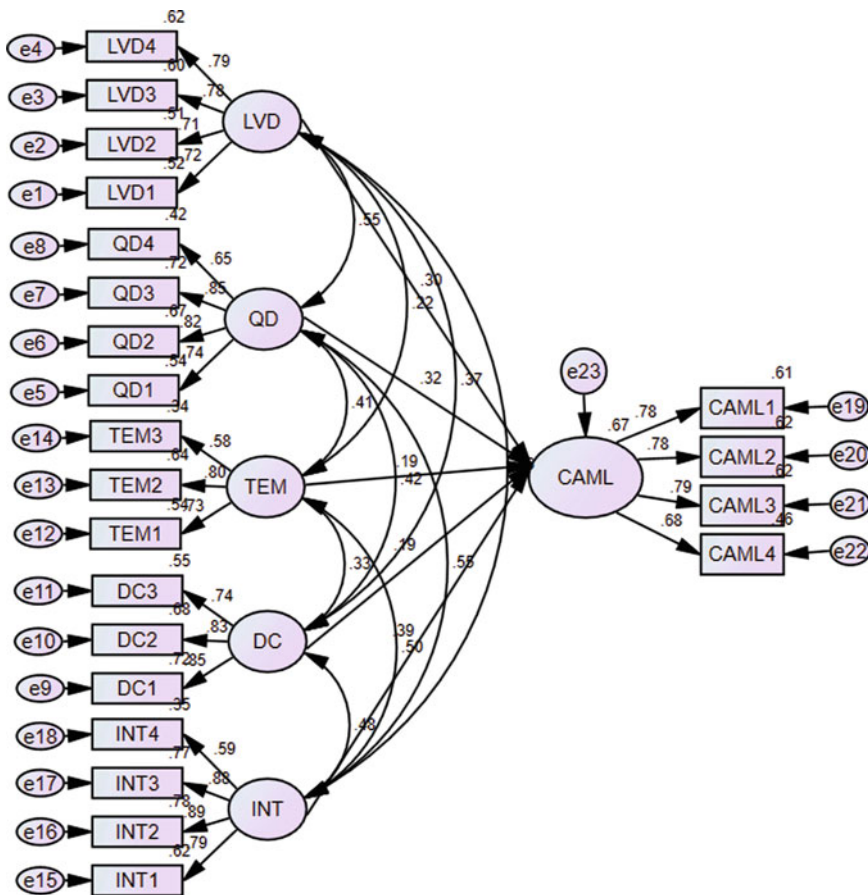


Fig. 1 The structural model

Table 5 Structural model results

	Estimate	SE	CR	P	Hypothesis
CAML ← LDV	0.300	0.081	3.703	0.00	Supported
CAML ← QD	0.323	0.062	5.209	0.00	Supported
CAML ← TEM	0.186	0.053	3.509	0.00	Supported
CAML ← DC	0.190	0.041	4.634	0.00	Supported
CAML ← INT	0.394	0.049	8.041	0.00	Supported

Source: Results obtained from AMOS software

5 Discussion

Although artificial intelligence and technology in ML are promising to contribute to improved patient outcomes and lower the cost of healthcare, using them effectively requires expertise and experience in managing massive data sets and tools for extracting the correct information to respond to the most challenging issues of healthcare [14]. Few professionals and scientists in the analysis have profound expertise, and even less have healthcare experiences with artificial intelligence and learning technologies [33]. So, for now, you have to team with experts who understand this kind of questions to get meaningful results for healthcare problems [23]. Health discovery and positive health outcomes often result in causality or something that happens [34, 35].

Interoperability and integration are key factors that distinguish academic research from practical ML applications, and this is what all healthcare facilities are focusing on right now. The patient is at the center of ML in healthcare. Consequently, we must be cautious about how transparent we are in addressing some of these ML remedies back at the health facilities and how much doctors can understand [36]. Compared to predicting what is likely to happen in a well-understood field, predictor modelling requires a different task and specific instruments to understand the causality [37]. To achieve cause-specific analysis, experts need to concentrate on the particular problem that needs to be addressed, namely, the health expertise [13].

With the help of ML and deep learning models, AI has revolutionized the study of image diagnosis in healthcare. MRI scans are an essential application of AI in clinical diagnosis. ML has begun to take over the complex analysis of MRI scans and has substantially simplified the process. Medical conditions such as heart disease and diabetes necessitated the use of ML. Many AI-powered wearables are being developed to monitor a person's health and notify when devices detect anything unexpected or unusual. According to a recent study, the advancement of ML-based virtual nurses has risen as an engine for physician assistants [18]. According to a recent study, virtual nursing assistants could be worth up to \$20 billion by 2027 [10]. A virtual nurse monitors patients' conditions and follows up on treatment [38]. ML played an essential role in healthcare decision-making. Still, AI has also improved businesses by studying customer needs and assessing any potential risks that a company may face. An impactful use case of ML in decision-making is robotic

surgery, which can reduce errors and variations and eventually help increase the efficiency of your surgeons.

6 Conclusion

This research paper aims to investigate the challenges and impact of ML in the Indian healthcare industry. A structured literature review was conducted, and some challenges were identified. In the Indian healthcare industry, a questionnaire was being developed for survey-based research. People working in the healthcare sector or hospitals were the intended audiences. EFA and SEM techniques are used for analysis. The outcome demonstrated that all hypotheses were accepted.

References

1. Bhardwaj, R., Nambiar, AR., Dutta, D.: A study of machine learning in healthcare. In 2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC) (2017)
2. Mukherjee, S., Baral, MM., Venkataiah, C., Pal, SK., Nagariya, R.: Service robots are an option for contactless services due to the COVID-19 pandemic in the hotels. *Decision*. (2021)
3. Shameer, K., Johnson, KW., Glicksberg, BS., Dudley, JT., Sengupta, PP.: Machine learning in cardiovascular medicine: are we there yet?. *Heart*. (2018).
4. Miotto, R., Wang, F., Wang, S., Jiang, X., Dudley, JT.: Deep learning for healthcare: review, opportunities and challenges. *Briefings in bioinformatics*. (2018).
5. Carleo, G., Cirac, I., Cranmer, K., Daudet, L., Schuld, M., Tishby, N., Vogt-Maranto, L., Zdeborová, L.: Machine learning and the physical sciences. *Reviews of Modern Physics*. (2019).
6. Liakos, KG., Busato, P., Moshou, D., Pearson, S., Bochtis, D.: Machine learning in agriculture: A review. *Sensors*. (2018).
7. Holzinger, A., Goebel, R., Palade, V., Ferri, M.: Towards integrative machine learning and knowledge extraction. In *Towards integrative machine learning and knowledge extraction* (2017)
8. El Naqa, I., Murphy, MJ.: What is machine learning?. In *machine learning in radiation oncology* (2015).
9. de la Torre, J., Marin, J., Ilarri, S., Marin, JJ.: Applying machine learning for healthcare: A case study on cervical pain assessment with motion capture. *Applied Sciences*. (2020).
10. Nayyar, A., Gadhavi, L., Zaman, N.: Machine learning in healthcare: review, opportunities and challenges. *Machine Learning and the Internet of Medical Things in Healthcare*. (2021).
11. Zhang, XD.: A matrix algebra approach to artificial intelligence.
12. Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y., Zhu, H., Gao, M., Hou, H., Wang, C.: Machine learning and deep learning methods for cybersecurity. *Ieee access*. (2018).
13. Mustafa, A., Rahimi, Azghadi M.: Automated machine learning for healthcare and clinical notes analysis. *Computers*. (2021).
14. Jordan, MI., Mitchell, TM.: Machine learning: Trends, perspectives, and prospects. *Science*. (2015).
15. Emmanuel, M., Milena, C., Alexandre, CC., Hamilton, V., Derennes, T., André, I., Franck, VM., Turcotte, S., Samuel, K., Tang, A.: Deep learning workflow in radiology: a primer. *Insights into Imaging*. (2020)

16. Libbrecht, MW., Noble, WS.: Machine learning applications in genetics and genomics. *Nature Reviews Genetics*. (2015).
17. Ferdous, M., Debnath, J., Chakraborty, NR.: Machine learning algorithms in healthcare: A literature survey. In 2020 11th International conference on computing, communication and networking technologies (ICCCNT) (2020).
18. Loh, BC., Then, PH.: Deep learning for cardiac computer-aided diagnosis: benefits, issues & solutions. *MHealth* 3: 45–45.
19. Bakator, M., Radosav, D.: Deep learning and medical diagnosis: A review of literature. *Multimodal Technologies and Interaction*. (2018).
20. Zhou, L., Pan, S., Wang, J., Vasilakos, AV.: Machine learning on big data: Opportunities and challenges. *Neurocomputing*. (2017).
21. Dhillon, A., Singh, A.: Machine learning in healthcare data analysis: a survey. *Journal of Biology and Today's World*. (2019).
22. Manogaran, G., Lopez, D.: A survey of big data architectures and machine learning algorithms in healthcare. *International Journal of Biomedical Engineering and Technology*. (2017).
23. Saravagi, D., Agrawal, S., Saravagi, M.: Opportunities and challenges of machine learning models for prediction and diagnosis of spondylolisthesis: a systematic review. *International Journal of Engineering Systems Modelling and Simulation*. (2021).
24. Mukherjee, S., Chittipaka, V.: Analysing the adoption of intelligent agent technology in food supply chain management: an empirical evidence. *FIIB Business Review*. (2021).
25. Baral, MM., Verma, A.: Cloud Computing adoption for healthcare: an empirical study using SEM approach. *FIIB Business Review*. (2021).
26. Nunnally, JC.: *Psychometric theory* 3E. Tata McGraw-hill education; (1994).
27. Henseler, J., Ringle, CM., Sinkovics, RR.: The use of partial least squares path modeling in international marketing. In *New challenges to international marketing* (2009).
28. Pal, SK., Mukherjee, S., Baral, MM., Aggarwal, S.: Problems of big data adoption in the healthcare industries. *Asia Pacific Journal of Health Management*. (2021).
29. Pal, SK., Baral, MM., Mukherjee, S., Venkataiah, C., Jana, B.: Analyzing the impact of supply chain innovation as a mediator for healthcare firms' performance. *Materials Today: Proceedings*. (2021).
30. Moerdyk, AP.: *The principles and practice of psychological assessment*. Van Schaik; (2009).
31. Kline, RB.: *Principles and practice of structural equation modeling*. Guilford publications; (2015).
32. Kline, RB.: *Assumptions in structural equation modeling*. (2012).
33. Roy, R., Giduturi, A.: Survey on pre-processing web log files in web usage mining. *Int. J. Adv. Sci. Technol*. (2019).
34. Rad, NM., Marchiori, E.: Machine learning for healthcare using wearable sensors. In *Digital Health* (2021).
35. Al-Shedivat, M., Dubey, A., Xing, EP.: Contextual Explanation Networks. *J. Mach. Learn. Res.*. (2020).
36. Dev, D. R., Badhan, A. K., and Roy, R.: A Study of Artificial Emotional Intelligence for Human – Robot Interaction, *J. Crit. Rev.*, (2020).
37. Injadat, M., Moubayed, A., Nassif, AB., Shami, A.: Machine learning towards intelligent systems: applications, challenges, and opportunities. *Artificial Intelligence Review*. (2021).
38. Sharma, A., Singh, P., Dar, G.: *Artificial Intelligence and Machine Learning for Healthcare Solutions. Data Analytics in Bioinformatics: A Machine Learning Perspective*. (2021).

Performance Evaluation of Deep Learning Architectures for Recognition of Moisture in Dried Coconut Copra



K. Padma Vasavi, G. Srinivasa Rao, S. Hanumantha Rao, M. Prema Kumar, and P. Ravi Kumar

Keywords Copra · Moisture · Classifier · SqueezeNet · AlexNet

1 Introduction

Coconut, scientifically known as *Cocos nucifera* [1] is cultivated in India since ancient times. India is the second largest producer of coconut in the world, contributing to 24.24% in total production and occupying one-third of total area under cultivation across the globe [2]. The coconut tree is believed as “Kalpataru” by the Indians as every part of the tree is useful to the humankind in one or the other way. One such useful product from the coconut tree is “copra” which is an endosperm of the coconut and is a rich source of edible oil. The coconut copra is more commonly used in baking, cooking, and beauty products for moisture retention of the skin and as a food for the farm animals. To make copra, the coconuts are broke opened, and, after the water is drained, the coconut kernels are dried either by exposing to sunlight or by heating in a kiln. However, the traditional sun drying can be used only on sunny days, and it required a minimum of seven to 8 days to completely dry the copra. Further, in the traditional sun drying process, the farmers will spread the coconut shells on cement floors, on soil along the roadside, or on the roof tops. As a result of this unhygienic process, there will be a huge product loss due to uneven drying and growth of fungus on the coconut shells [3]. Another most important technique for drying the coconuts is to dry them in kilns. In kiln drying, the coconuts are exposed to flame by placing them on the racks just above the fire. As a result, polycyclic aromatic hydrocarbons (PAH) are deposited on the kernels, which is hazardous to health as PAH is carcinogenic and causes cancer. Further, as the copra is directly subjected to fire, it results in discoloring of copra and gives

K. Padma Vasavi (✉) · G. Srinivasa Rao · S. Hanumantha Rao · M. Prema Kumar · P. Ravi Kumar
Shri Vishnu Engineering College for Women, Bhimavaram, India
e-mail: padmavasaviece@svecw.edu.in; hanumanth.s@svecw.edu.in;
mpremkumar@svecw.edu.in; Ravikumar_tnk2@svecw.edu.in

them a brown color which reduces the food grade quality of the copra. Biomass solar dryers are the alternatives for drying the copra without degrading its quality [4]. The biomass dryers consist of a burning chamber and a flue gas chamber. The flue gas chamber is befitted with flow pipes to enable the heat transfer between flue gas and the fresh air which result in the removal of moisture content in the copra. This entire system is retrofitted to a solar dryer, which supplies hot air through the holes to dry the coconut shells. The biomass solar dryers are found to be the most efficient system for drying the copra with minimum moisture content without degrading the quality of the copra [5]. However, as the coconuts are dried as batches either in the kilns or in the biomass solar dryers, there is every possibility that a good number of copra shells stealthily come out with moisture content not suitable for good food grade quality. Further, these copra shells with moisture content may develop fungus, and spread it across the entire batch resulting in huge yield loss. So, researchers have come up with systems that assess the moisture content of the copra after getting dried either using conventional solar drying or in kilns and biomass dryers. A few devices like copra moisture tester are available in the market which measure the moisture content by probing the needle into copra. The measurement is done on one-to-one basis which is time conssecuming [6]. Madan Kumar et al. used an online near-infrared spectroscopic system to measure the moisture content in the copra from a far proximity [7]. This system is found to be working very efficiently under laboratory conditions. However, this system is bulky and is costly. It also requires trained personnel to operate the instrument. So, there is a need for a low-cost portable system which considers a batch of copra shells to measure the moisture content, classifies them according to the percentage of moisture present in them, and grades them according to the quality based on the moisture content. This paper proposes to develop a computer vision-based Copra Moisture Detection System which implements deep learning architectures to classify the copra according to the moisture content present in the coconut shell. The rest of the paper is organized as follows: The details of the deep learning architecture used to classify the copra are presented in Sect. 2, the results and analysis are discussed in Sect. 3, comparative analysis with different deep learning architectures in determining the moisture content is presented in Sect. 4, and finally Sect. 5 concludes the paper.

2 Copra Moisture Classification System

The concept diagram for the proposed Copra Moisture Classification System (CMCS) is shown in Fig. 1. The concept diagram of the proposed Copra Moisture Classification System (CMCS) is shown in Fig. 1. The working of CMCS is divided into two phases: training phase and testing phase. In the training phase, a dataset consisting of copra images with different moisture levels is presented to the deep neural network implemented in the graphics processing unit (GPU). The deep neural network extracts the features of the copra and classifies them into two classes which come under suitable and unsuitable categories. During the testing phase, the camera

captures the image of the copra and passes it to the GPU. The DNN implemented in the GPU examines the features of the copra and takes a decision to accept or reject the copra for further processing.

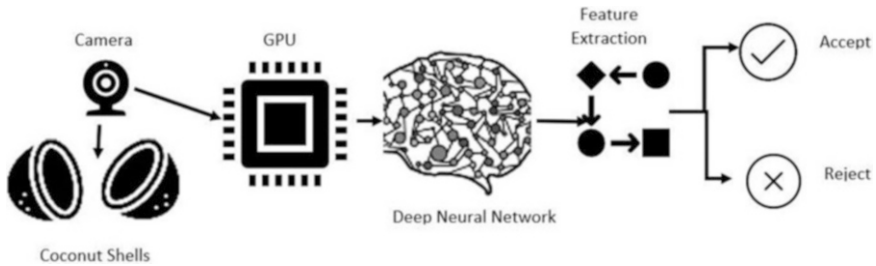


Fig. 1 Concept diagram of Copra Moisture Classification System (CMCS)

As a benchmark dataset for the copra images is not available in the literature to the best of author’s knowledge, we have made our own database of 300 images of copra shells with different moisture contents. The database is created by collecting the copra images from various Internet sources. As the database, thus, collected is comprising of images with different sizes and resolutions, the images in the datastore are resized to $224 \times 224 \times 3$, which is the size required by the input layers of both AlexNet and SqueezeNet. Further, the background of the copra images is removed by background subtraction. Then, the preprocessed dataset is augmented with copra images subjected to random shifting scaling and rotation. Then, a zero-center normalization is performed on the dataset to make the mean equal to zero and the standard deviation equal to one. This process is to ensure the entire dataset is uniform which helps the neural network to learn at a faster rate and improve the classification accuracy. A sample of the images in the database is shown in Fig. 2.



Fig. 2 Random samples in the Copra Image Database

The proposed Copra Moisture Classification System is implemented by using transfer learning technique on two different deep neural network architectures: the AlexNet and the SqueezeNet.

2.1 The AlexNet

Simple convolution neural network architectures like “LeNet” exhibit efficient classification performance for low databases consisting of thousands of labelled images. However, their classification and recognition efficiency decrease for large databases with hundreds of thousands to millions of labelled images. Further, they are hardly suitable for images with high resolution. AlexNet performs well in such scenarios as it is an extremely deep neural network model. The architecture of AlexNet is shown in Fig. 3. The AlexNet architecture consists of five convolution layers followed by three fully connected layers. A ReLU layer is connected after every convolution layer and fully connected later. The final fully connected layer is connected to a softmax layer which categorizes the copra into one of the two classes. Further, a dropout layer is connected to the first and second fully connected layers to eliminate overfitting [8].

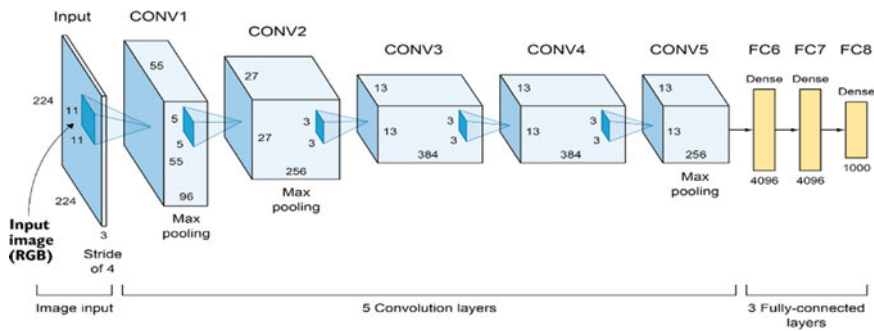


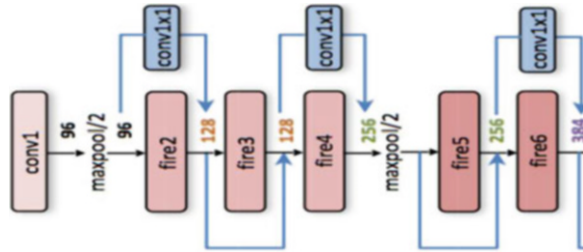
Fig. 3 Architecture of AlexNet [9]

Along with the AlexNet, a SqueezeNet architecture is also used classify the copra moisture. The details of the SqueezeNet are presented below.

2.2 The SqueezeNet

Though AlexNet is a wonderful architecture, in terms of its classification accuracy, it requires enormous number of parameters for classification. This makes it difficult to deploy the AlexNet architecture on hardware processor boards like FPGA and Raspberry Pi, because of their limited storage space. SqueezeNet offers a solution for this problem, by giving the same accuracy as AlexNet and reducing the parameter size by fifty times that of the AlexNet, with the help of an intelligent convolution module called “Fire Module.” The architecture of SqueezeNet is shown in Fig. 4.

Fig. 4 SqueezeNet architecture [10]



As mentioned earlier, the heart of the SqueezeNet is the fire module, consisting of a Squeeze convolution layer with 1 X1 filters connected to an expand layer which is a combination of 1 X1 and 3 X3 filters. The use of 1 X1 filter as opposed to 3 X3 filters in AlexNet reduces the number of parameters by nine times. The fire module is shown in Fig. 5. The architecture of SqueezeNet consists of a regular convolution layer followed by eight fire modules [9]. The last fire module is connected to another convolution layer which is connected to a soft max layer that gives the two labelled classes of copra.

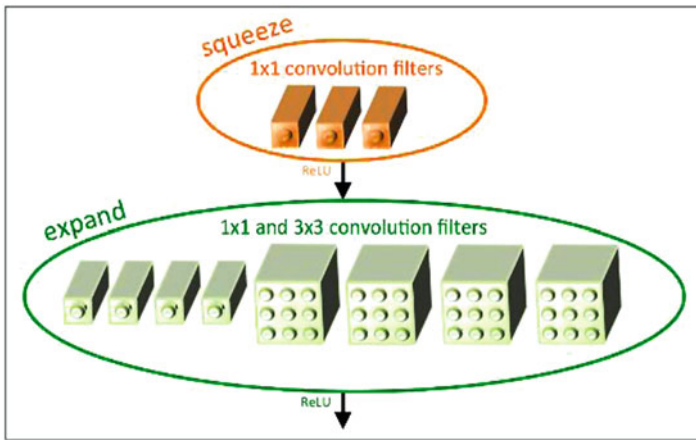


Fig. 5 Fire module in SqueezeNet

Till now the implementation details of proposed Copra Moisture Classification system are explained in detail; the next section presents the results obtained in copra classification using the proposed system.

3 Results and Discussion

As discussed earlier, transfer learning with benchmark architectures of AlexNet and SqueezeNet is done to classify the copra according to the moisture content present. The Alex Net can classify 1000 classes, and our application requires only 2 classes.

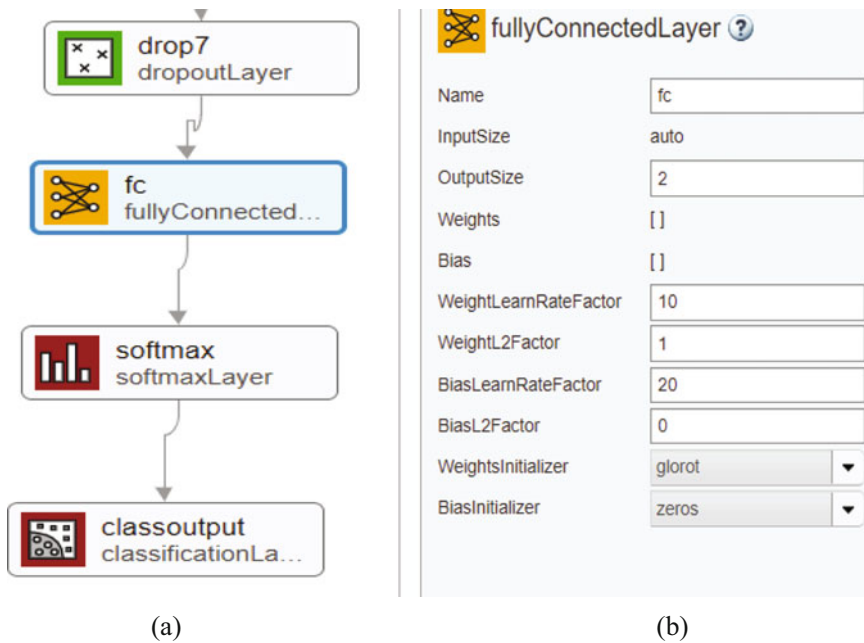


Fig. 6 (a) New layers in AlexNet; (b) fully connected layer with two classes

Therefore, the last three layers of the AlexNet—fully connected layer with 1000 classes, softmax layer, and the classification layer—are removed and are replaced with fully connected layer with two classes: new softmax layer and a new classification layer as shown in Fig. 6a and b.

Similarly, for SqueezeNet, the final convolution layer and classification layer are replaced by convolution layer with two filters and classification layer with two classes as shown in Fig. 7.

We have prepared a dataset of 300 images for each class of copra and augmented the data by shifting through -90° and 90° scaling the images from 1 to 2. Some sample images of each class of copra are shown in Fig. 8a and b, respectively.

Both the deep neural network architectures, AlexNet and SqueezeNet, were trained on Nvidia (TM) GeForce GTX GPU with 16 GB of memory. For each architecture, the training was done with an initial learning rate of 0.0001. The early layers are trained with a slower rate than the layers that are near to the output. This combination of learning rate settings results in fast learning only in

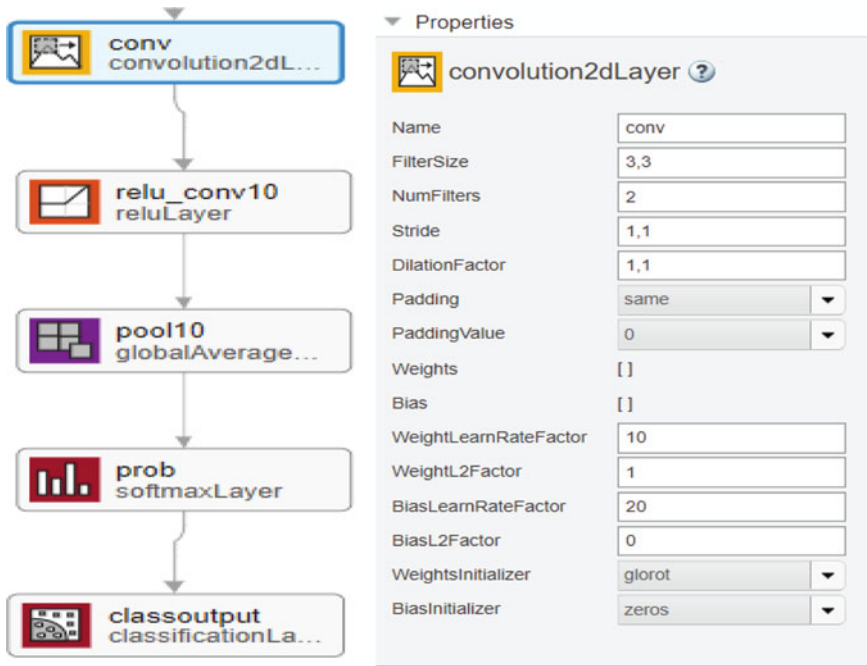


Fig. 7 (a) New layers in SqueezeNet, (b) 2D convolution layer with two classes

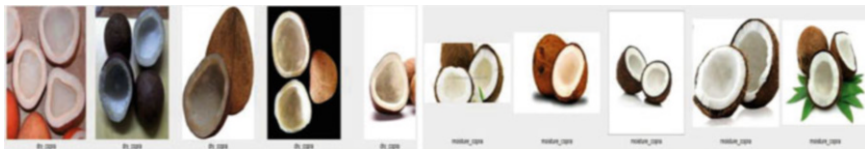


Fig. 8 (a) Dry copra, (b) moisture copra

the new layers and slower learning in the other layers during transfer learning. One of the advantages of transfer learning is that the deep neural network need not be trained for a greater number of epochs, where an epoch is defined as full training cycle on an entire dataset. For each network, training is done with three different solvers: stochastic gradient descent with momentum (SGDM), RMSProp, and ADAM, keeping the other training options fixed.

The results of training the architecture with AlexNet using stochastic gradient descent with momentum, RMSProp, and ADAM solvers is shown in Fig. 9. A validation accuracy of 94.74% is obtained with SGDM solver, a validation accuracy of 92.98% is obtained with RMSProp solver, and a validation accuracy of 96.49% is obtained with ADAM solver for the AlexNet architecture.

From the confusion matrix, the parameters—accuracy, precision, and recall—are calculated that determine the performance of a classifier. The confusion matrix is

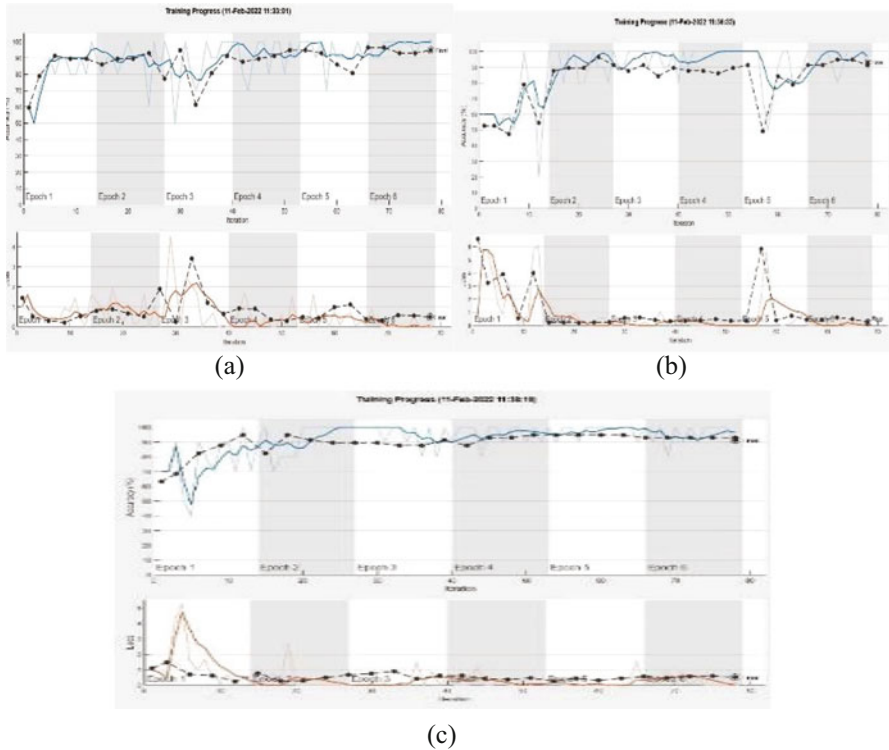


Fig. 9 Training AlexNet with (a) SGDM solver (b) RMSProp solver (c) ADAM solver

defined as an $N \times N$ matrix to evaluate the performance of a classifier. The columns of the confusion matrix represent the actual values of the target variable, and the rows represent the predicted values of the target variable.

The diagonal cells correspond to observations that are correctly classified. The off-diagonal cells correspond to incorrectly classified observations. Both the number of observations and the percentage of the total number of observations are shown in each cell. The last column on the right of the plot, called precision, shows the percentages of all the samples predicted to belong to each class that classified correct and wrong. The bottom most row of the plot shows the percentages of all the examples belonging to each class that classified correct and wrong which are called recall and false-negative rate. The overall accuracy is given by the bottom right cell of the plot. The cell in the bottom right of the plot shows the overall accuracy. The confusion matrices plotted for copra classification with AlexNet using SGDM, RMSProp, and ADAM solvers are shown in Fig. 10.

The results of training the architecture with SqueezeNet using stochastic gradient descent with momentum, RMSProp, and ADAM solvers are shown in Fig. 11. A validation accuracy of 98.25% is obtained with SGDM solver, a validation accuracy

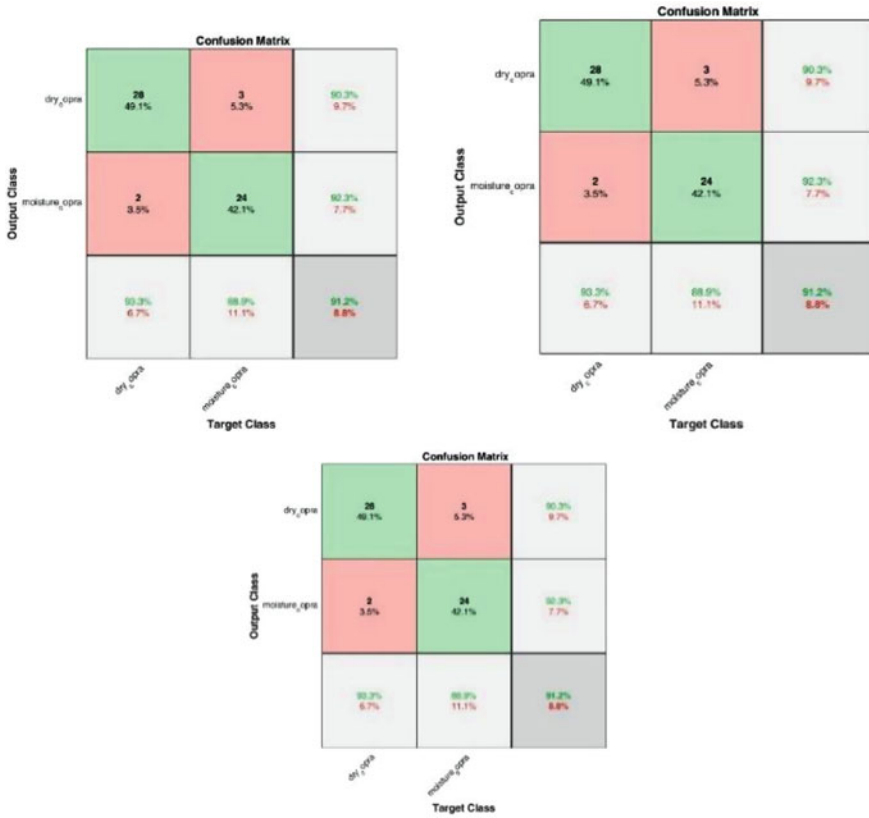


Fig. 10 Confusion matrix of AlexNet with (a) SGDM solver, (b) RMSProp solver, (c) ADAM solver

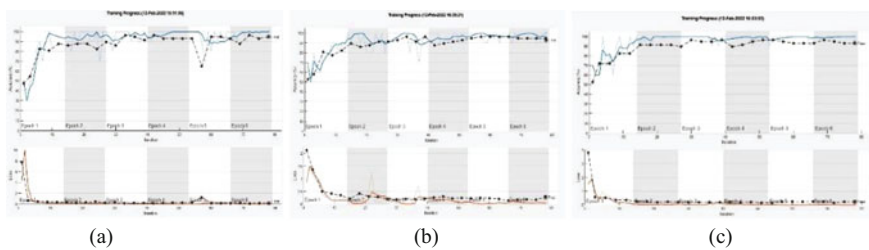


Fig. 11 Training SqueezeNet with (a) SGDM solver, (b) RMSProp solver, (c) ADAM solver.

of 87.4% is obtained with RMSProp solver, and a validation accuracy of 92.98% is obtained with ADAM solver for the SqueezeNet architecture.

The confusion matrices plotted for copra classification with SqueezeNet using SGDM, RMSProp, and ADAM solvers are shown in Fig. 12.

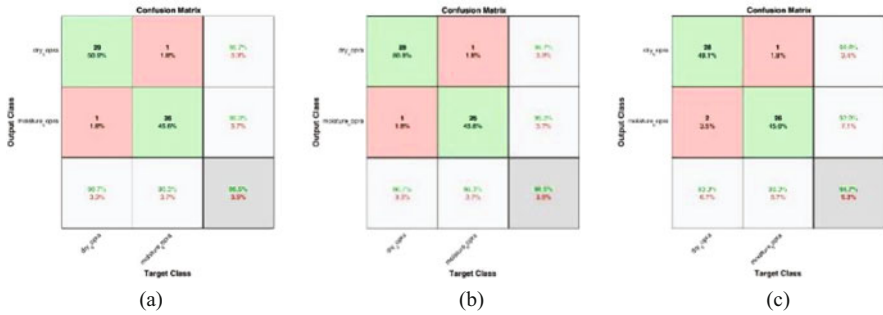


Fig. 12 Confusion matrix of SqueezeNet with (a) SGDM solver, (b) RMSProp solver, (c) ADAM solver

The performance of the architecture is tested by randomly giving test images from the test dataset, and the result of classifying the copra with appropriate labels is shown in Fig. 13.

Fig. 13 Testing the architecture for classification



Till now, the experimental set-up and results obtained for copra moisture classification using AlexNet and SqueezeNet are discussed in detail. The performance comparison of these two networks for copra classification is presented in the next section.

4 Comparative Analysis

The proposed copra moisture classification system is developed by using transfer learning with two different architectures: the AlexNet and the SqueezeNet by using three different solvers like SGDM, RMSProp, and the ADAM solver, keeping the

other hyperparameters like learning rate and the activation function at fixed values. The training data size is selected as 96% of total dataset to improve the classification accuracy.

Table 1 Training options

S. No.	Parameter	Value
1	Mini batch size	10
2	Initial learning rate	0.0001
3	Maximum epochs	6
4	Training data size	96% of total data
5	Testing data size	2% of total data
6	Validation data size	2% of total data
7	Shuffle data	Every epoch

The performance evaluation of the two architectures is made w.r.t accuracy, precision, recall, and execution time and is presented in this section. The comparisons are made by fixing the training options as given Table 1 for both the deep neural network architectures and for all the three solvers.

The performance comparison of the deep neural networks for copra moisture classification w.r.t accuracy, precision recall, F1 score, and execution time are given in Table 2, where:

$$\text{Precision} = (\text{True Positive}) / (\text{True Positive} + \text{False Positive}) \tag{1}$$

$$\text{Recall} = (\text{True Positive}) / (\text{True Positive} + \text{False Negative}) \tag{2}$$

$$\text{F1 Score} = 2 (\text{Precision} * \text{Recall} / (\text{Precision} + \text{Recall})) \tag{3}$$

Table 2 Performance comparison of copra moisture classifiers

Parameter	AlexNet			SqueezeNet		
	SGDM	RMSProp	ADAM	SGDM	RMSProp	ADAM
Accuracy in %	94.74	92.98	92.98	98.25	89.4	92.98
Recall in %	50	50	50	98.1	87	96.3
Precision in %	100	100	100	98.3	89	96.8
F1 score	66.67	66.67	66.67	98.2	88	96.5
Execution time in seconds	35	32	32	26	36	36

A graph plotted between the parameters chosen for performance evaluation and different deep neural networks with different solvers is shown in Fig. 14. From the comparison table of Table 2 and the comparison chart shown in Fig. 14, it is observed that the SqueezeNet with stochastic gradient descent solver is showing

better performance when compared with other methods as its performance is better with respect to accuracy, F1 score, and execution time. Though the precision exhibited by AlexNet for different solvers is a 100%, its performance with respect to F1 score, recall, and execution time is inferior compared to SqueezeNet with any solver. So, it is understood that SqueezeNet with SGDM solver is the better choice for copra moisture classification.

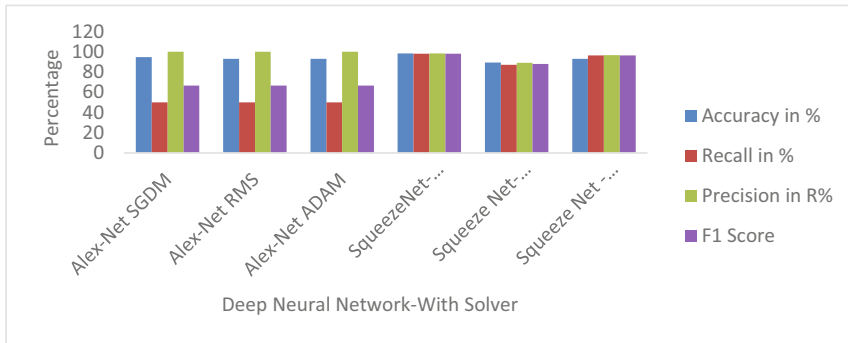


Fig. 14 Performance comparison chart

5 Conclusion

A computer vision-based Copra Moisture Detection System which implements deep learning architectures to classify the copra according to the moisture content present in the coconut shell is proposed in this paper. The proposed system is implemented by transfer learning with AlexNet using three different solvers and SqueezeNet with three different solvers. The performance of these two networks is compared in terms of accuracy, precision, recall, and execution time. It is observed that the SqueezeNet with stochastic gradient descent solver is showing better performance when compared with other methods as its performance is better with respect to accuracy, F1 score, and execution time. Though the precision exhibited by AlexNet for different solvers is a 100%, its performance with respect to F1 score, recall, and execution time is inferior compared to SqueezeNet with any solver. So, it is understood that SqueezeNet with SGDM solver is the better choice for copra moisture classification.

References

1. Arunachalam, “2-Coconut”, Genomics of Cultivated Palms,2012, pp. 13–27, Handbook of Coconut Development Board 2019–20
2. Thiruchelvam Thanaraj, Nimal D.A., Dharmasena, Upali Samarajeewa, “Comparison of quality and yield of copra processed in CRI improved kiln drying and sun drying”, Journal of Food Engineering Volume 78, Issue 4, February 2007, Pp. 1446–1451
3. Sachidananda Swain, Din M, Chandrika R, Sibnarayan Dam Roy, “performance-evaluation-of-biomass-fired-dryer-for-copra-drying-a-comparison-with-traditional-drying-in-subtropical-climate”, Journal of Food Process Technology 2014,
4. J Deepa, P Raj Kumar & T Armuganadhan, “Quality Analysis Of Copra Dried At Different Drying Air Temperatures”, International Journal of Agricultural Science and Research (IJASR) ISSN(P): 2250-0057; ISSN(E): 2321-0087 Vol. 5, Issue 4, Aug 2015, 1–6
5. https://www.kett.co.jp/products_en/hx-120/
6. Madan Kumar Lakshmanan, Tharmalingam Chinnu, Gopal Arvamuthan, “Near-infrared reflectance spectroscopy based online moisture measurement in copra”, Journal of Food Process Engineering, John Wiley,2020.
7. Krizhevsky, Alex; Sutskever, Ilya; Hinton, Geoffrey E. (2017-05-24). “ImageNet classification with deep convolutional neural networks”, *Communications of the ACM*. **Pp:** 84–90.
8. Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, Kurt Keutzer, “Squeeze Net: Alex Net-level accuracy with 50x fewer parameters and <0.5MB model size”, *ar Xiv preprint ar Xiv:1602.07360* (2016).
9. Alex Krizhevsky Ilya Sutskever Geoffrey E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, Communications of the ACM Volume 60, Issue 6 June 2017 pp 84–90 <https://doi.org/10.1145/3065386>
10. Amir Gholami, Kiseok Kwon, Bichen Wu, Zizheng Tai, Xiangyu Yue, Peter Jin, Sicheng Zhao, Kurt Keutzer, “SqueezeNext: Hardware-Aware Neural Network Design”, arXiv:1803.10615v2 [cs.NE] 27 Aug 2018

Training Generative Adversarial Networks (GANs) Over Parameter Server and Worker Node Architecture



Amit Ranjan and Rajiv Misra

Keywords Generated adversarial networks (GANs) · Deep learning · Distributed networks

1 Introduction

Artificial intelligence (AI) has been on the ascent in the previous decade, primarily because of the involvement of Deep Neural Networks and Big Data. Generative Adversarial Networks (GANs) [1] are a deep generative model that have accomplished enormous achievements in a variety of generative applications like image to image, image to text, so on [2–5, 11, 13]. GANs study the nature of distribution of input data and extract data points from the distribution, which aims to produce original looking real data. It is based on a min-max game theory and generate output using deep neural network (DNN) techniques. In GANs, one DNN is known as generator which produces data samples and the other DNN called the discriminator which tries to differentiate among the generator's data with the real data. The interaction between the generator and discriminator brings about improving the DNN weights with the end goal that the generator's produced data seems to appear the same as the real data.

Usually, AI techniques need to assemble training data into a central database for training to be conducted. Like most of the machine learning models, GANs need huge dataset to perform their corresponding learning tasks. These days, it is common for service providers to gather huge amounts of data in a central database like their data center and the training stage takes place in those data centers. Such datasets are usually gathered from worker nodes (end users) of the system and kept in a central data center for being utilized by a central workspace or cloud for learning

A. Ranjan (✉) · R. Misra

Department of Computer Science and Engineering, Indian Institute of Technology Patna, Patna, Bihar, India

e-mail: amit_1921cs18@iitp.ac.in; rajivm@iitp.ac.in

activities. However, depending on a central workplace requires strong computational capacities and may result in significant delays. On the other hand, these central data centers suffer from the risk of external attacks. Another issue is the ongoing rise of guidelines restricting data movement beyond the international boundaries. In numerous situations, for example, in healthcare and financial applications, the information is confidential and distributed over various workers because of protection concerns that will not be transferable to a central database through open communications. Such difficulties persuade parallelism and the requirement for parameter server and worker node architecture, i.e., multi-worker learning for GANs. How to securely retrieve data from these heterogeneous devices to properly train models has become an open research issue.

A recent study consists of geo-based machine learning techniques [6], in which data obtained from multiple data centers remain in the same place, but it is difficult to meet the time delay in case of a single central database. Many deep learning models are hence to be adjusted in this setup. Few ongoing research incorporate many generators and discriminators with the aim of improving GAN integration, but they do not intend to use distributed data.

One of the most encouraging solutions for these issues is training over the parameter server and worker node setup (used in this chapter). In the parameter server and worker node architecture, more than one worker can learn GAN parameters in a respectful way by sharing certain types of data with every worker node simultaneously maintaining the security of their database. The objective of every worker in this GAN training arrangement is to train themselves to produce rich quality data samples which are almost similar to the real ones. This architecture can decrease the communication overhead and computation shortcoming of training central GAN models which makes them more efficient in huge environments with more workers. The parameter server and the worker node architecture are fundamentally a set of techniques and methods used to add a learning component to other devices which are generally associated with network linking. In this way, the resources of different devices can be utilized to speed up the training and in some cases empower the training on huge databases. It can likewise help in maintaining local laws and regulations, as worker nodes can hypothetically be said to be spread all over the world.

2 Literature Survey

Many conveyed distributed frameworks for machine learning models have been proposed to encourage parallelism [7, 8, 18, 19]. With the introduction of model parallelism in [7], the authors looked at the problem of deep learning model training with billions of parameters utilizing countless CPU centers. The authors in [7] built a software framework named DistBelief, which can use computer clusters and thousands of devices to train huge models. And inside this architecture, they built two large-scale training algorithms for appropriate distributed learning, i.e., first,

Downpour SGD, which is an asynchronous stochastic gradient descent method that supports a large number of model imitation, and second, Sandblaster, which is an architecture that upholds a wide range of group processes optimization.

As of late, federated learning (FL) in [8] was presented as a successful distributed learning process that permits various worker nodes to independently train a global model using their own database and link with the training update to a central server that aggregates the worker node weights while training the global model. The authors in [8] developed a high-quality model, while the training data are always distributed to a large number of worker clients each with inconsistent and moderate network connections. In [8], a new learning algorithm for a similar setup is proposed, where each worker in every cycle independently registers for an update to the current model based on local data and transmits the updated parameters to a central server, where the worker updates are set up to create a new global model. In any case, the methods in [7, 8] just as subsequent meetups on federated learning focus on derivation models and do not refer to any generative models.

In [9, 10, 12, 14], the authors researched the utilization of distributed structures that consider different GAN architecture that includes two different DNNs (one is a generator and the other one is a discriminator). In [9], a new method to train Generative Adversarial Networks (GANs) is proposed which contains a combination of generators to overcome the problem of model collapse. Their primary instinct is to utilize multiple generators, rather than utilizing a solitary one as in the original GAN. The generators generate the samples designed that seem to originate from the similar distribution as the training data, while the discriminator decides if the samples are real data or produced by the generators, and the classifier determines the generated sample created by which generator. Also, in [10], the Generative Multi-Adversarial Network (GMAN) framework is proposed that extends GANs to different discriminators. The structures utilized in GMAN can be dependably trained without tampering the original objective. Different discriminators are utilized to settle the learning cycle however not to learn from numerous datasets. While in [12] a solitary discriminator is associated with different generators so as to get familiar with numerous modalities of a dataset and to address the mode collapse issue. In [12], MAD-GAN architecture is proposed which takes a close look at original Generative Adversarial Networks (GANs) and its conditional variants to address the notable issue of mode breakdown. In [14], a new way to distribute GAN is trained using non-i.i.d. data named Forgiver-First Update, which states mainly two points firstly request workers to train their own discriminator with the data residing in their local place and secondly update generator to mislead “forgiving” discriminators who take samples that are produced as very real. With input as non-i.i.d. data, their fundamental purpose is learning a distribution, which includes all class input data, while the data are stored in every worker’s local space. Be that as it may, jobs in [7, 9, 10] look at the central database that all workers can access. In addition, the arrangements proposed [8] can create communication overhead because the workers require to update the DNN-trained parameters to the parameter server at each cycle. Additionally, none of the GAN structures in [9, 10, 12, 14] are completely decentralized and only require a centralized generator or centralized

discriminator. Moreover, GAN architecture solutions included in [9, 10, 12, 14] do not consider the calculation of the power and storage capacity of workers.

A new generative training method using an integrated training framework is proposed in [15]. Their approach trains the generative models distributed to workers, while the models trained by each worker are integrated into a single integrated and multifunctional model in the center.

3 Methodology and Architecture

3.1 Generative Adversarial Networks

The clarity of GAN benefit introduced earlier in [1] is that their learning phase is unsupervised in nature, e.g., there are no descriptive labels involved in training the model from the dataset. The basic GAN model has two parts: first one is the generator G and the second is discriminator D . The input fed into the generator is actually a random noise vector. The discriminator gets data as input from two sources: the output generated by the generator and raw data from the dataset containing real data. The main objective of discriminator is to determine the original data source, from where it is originating. Toward the start of training stage, generator produces data in a specific distribution manner and discriminator rapidly figures out the way to separate the produced data from generator with the real data coming from the database. After a certain epoch, the generator figures out how to create information that is nearer to the probability distribution of the data residing in the dataset. If it is the case where discriminator is unable to distinguish the generated data, which suggests that the generator got trained in learning the data distribution from the database.

More technically, as mentioned in [1], suppose a training dataset X , with x an element in dataset, follows a probability distribution P_{data} . Furthermore, as proposed in [1], the learning involved while training GAN can be seen as a two-player min-max game between generator G and discriminator D with the **value objective function** $V_{GAN}(G, D)$ given in Eq. (1).

$$V_{GAN}(G, D) = E_{x,y \sim p_{data}(x,y)}[\log(D(x, y))] + E_{x \sim p_{data}(x), z \sim p_z(z)}[\log(1 - D(x, G(x, z)))] \quad (1)$$

Overall loss function:

$$L1(G) = E_{x,y \sim p_{data}(x,y), z \sim p_z(z)}[\|y - G(x, z)\|_1] \quad (2)$$

Combined Objective Function:

$$G^* = \arg \min_G \max_D = V_{GAN}(G, D) + \lambda L1(G) \quad (3)$$

In order to generate output with promising accuracy, we add an additional L1 loss term that captures image pixel reconstruction error, Eq. (2), and combine it with the global $V_{GAN}(G, D)$ loss term, Eq. (1), to form the final objective function. The combined objective function is given in Eq. (3), where λ is selected as a hyper-parameter.

3.2 Parameter Server and Worker Node Architecture

We have developed a framework for learning the GAN model which contains one parameter server node that interacts with “N” worker nodes in a convenient way over the network. We arranged “N” worker nodes, each with its own data with meter samples that possess similar probability distribution of P_{data} . These local datasets reside in their local space (for example, will not be sent over the network). One assumption made that local datasets are i.i.d. in every worker node; in other words, there is no bias in distributing data on a specific worker node. The parameter server and worker node framework use worker nodes for the parallel processing, while the central parameter server manages shared parameters updated by the workers. This approach targets to train a similar model for all workers utilizing their shared information and to sync their training outcomes with the parameter server in every iteration in order to update model parameters.

3.3 Architecture

When it comes to the construction of our GAN model architecture, generator and discrimination are two different neural networks that are firmly integrated. To train the GAN model in our framework, which includes a server-side generator and the discriminator that is distributed over a lot of worker nodes. Our training follows the framework of the parameter server and worker node, especially where workers perform multiple local iterations in each cycle before communicating with the server, rather than sending minor updates. A simple block diagram of parameter server and worker node architecture is shown in Fig. 1. To eliminate a portion of the load from the server-side, discriminators are held by worker nodes only. Every worker begins with its own discriminator based on certain parameters. It may be possible to have different neural network architecture and different initial weights on every worker node, but for our convenience we assumed it to be the same. The purpose of GANs is to train a generator running on a parameter server, with a lot of

discriminators residing on worker nodes. It is basically a many-versus-one game in which the generator monitors all discriminators, i.e., the generator tries to produce data that are considered authentic by all workers. Workers utilize their in-place data to distinguish between the generated one and the actual one.

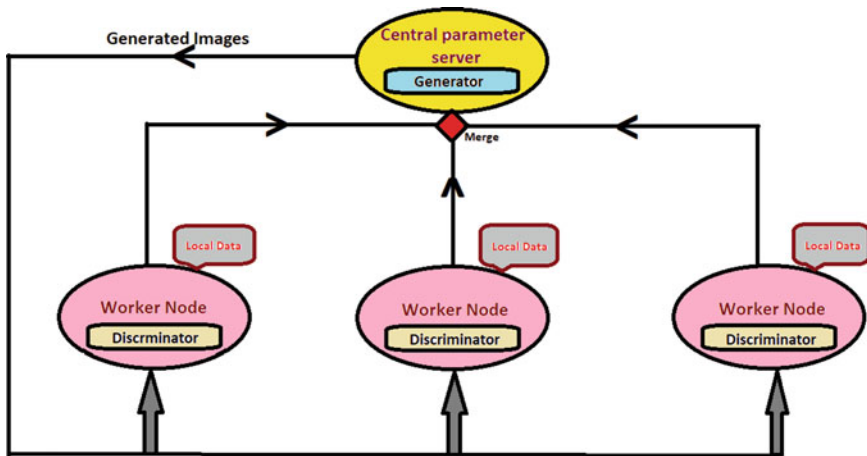


Fig. 1 Block diagram of training of GANs over parameter server and worker node framework

4 Experiments and Results

4.1 Dataset

The dataset used for training our model was the CIFAR-10[16] benchmark dataset. CIFAR-10 dataset is used as computer vision benchmark dataset utilized for object detection. It comprises 60,000 32×32 colored pictures containing one of 10 item classes, with 6000 pictures for every class. CIFAR-10 is made out of a training set of 50,000 RGB pictures of 32×32 pixels; each of them belongs to the given 10 classes: plane, vehicle, fowl, cat, deer, dogs, frog, horse, ships, and truck. CIFAR-10 has a test dataset of 10,000 pictures. It is one of the most broadly utilized datasets for AI research.

4.2 Simulation Setup

All our tests were performed using the PyTorch framework with the tensor backend. We simulated worker node and servers on GPU-based servers equipped with two

Intel processors with 12GB of RAM and two GPUv2 @ 2.2GHZ with 8 GB of RAM each. This simulation setup takes GAN training like actual distributed deployment. For testing the speculation, the parameter server and the worker node architecture were executed over a bunch of 3 sets of machines, one of which was a generator, and the other two were discriminators. The basic GAN network was a straightforward Deep Convolutional GAN [17] with 5 layers of architecture for every discriminator and generator. The architecture of the generator (on the server-side) and the discriminator (on worker nodes) used in our proposed framework consists of five convective layers in each generator and descriptor neural network. A 100-dimensional noise vector is given as an input to the generator. For the first layer, we have taken a stride of 1 and padding zero, followed by batch normalization and activation function as ReLU. After the first layer, we have 4 consecutive layers and all four layers have a kernel size of 4, with a stride value of 2 and a padding of 1. Instead of using ReLU, we used Tanh activation function in the last convolutional operation. The final image generated by the generator is $64 * 64 * 3$ dimensions. A simple convolution operation block diagram of generator and discriminator is shown in Fig. 2a and b.

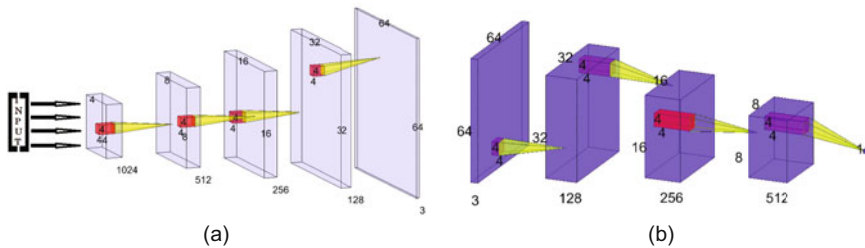


Fig. 2 (a) Block diagram of generator over parameter server and (b) block diagram of discriminator over worker node

4.3 Results

For every simulation, each class was divided into 3 different chunks and given to every worker node for training purposes. Thus, as an outcome of this, no data point was shared between the two workers. The division was made with the goal that none of the workers could get more data of a similar class compared to the other worker node. Results obtained after simulation are mentioned in Table 1, and for visualization of generated images compared with real image w.r.t. epoch, iteration is shown in Fig. 3a, b, and c. A comparison between the final generated output and the real image from the CIFAR-10 dataset is shown in Fig. 4.

Table 1 Results obtained from training GANs over parameter server and worker node architecture

Sl No.	Measures	Proposed architecture
1	No. of parameter server	1
2	No. of worker node	2
3	No. of epoch	24
4	Optimizer	Adam
5	Batch size	64
6	Learning rate	0.0002
7	Loss function	BCE
8	Data skew	0%
9	Accuracy	71%

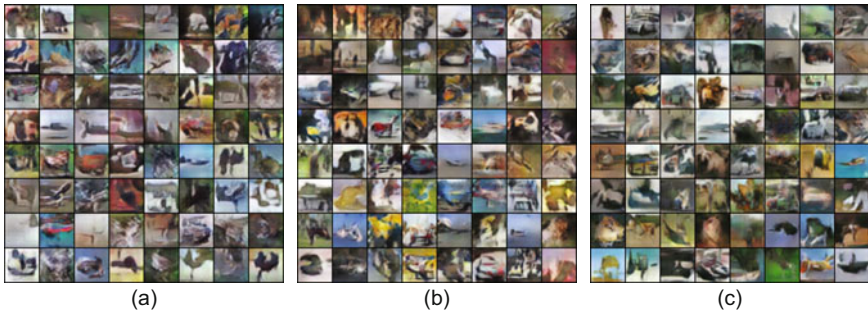


Fig. 3 (a) Generated image at Epoch 8, (b) generated image at Epoch 16, and (c) generated image at Epoch 24

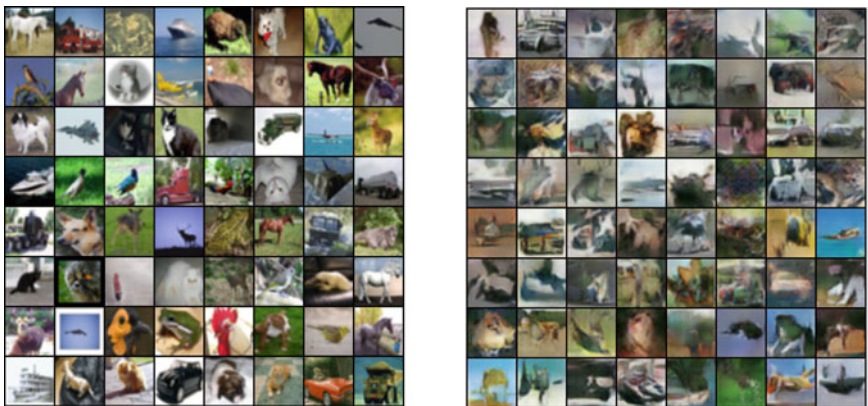


Fig. 4 Comparing the real image (left) with final generated output (right)

5 Conclusion and Future Work

GAN performance under our framework is satisfactory. But it is hard to say that this model does not have any defect. The fundamental shortcoming of this methodology

that we have assumed that the workers possess a similar kind of data distribution in their local space and because of this some real-world applications are beyond the scope of this architecture.

Future work involves training our model with non-i.i.d. data and also extending our framework in various GAN applications where data privacy became one of the major hurdles while training any deep learning model in a cost-efficient manner.

References

1. Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." In *Advances in neural information processing systems*, pp. 2672–2680. 2014.
2. Ranjan, Amit, Debanshu Lalwani, and Rajiv Misra. "GAN for synthesizing CT from T2-weighted MRI data towards MR-guided radiation treatment." *Magnetic Resonance Materials in Physics, Biology and Medicine* pp. 1–9. 2021.
3. Mirza, Mehdi, and Simon Osindero. "Conditional generative adversarial nets." *arXiv preprint arXiv:1411.1784* (2014).
4. Deb, Sagar Deep, et al. "Facial Expression Classification using Multi-Scale Histogram of Oriented Gradients." *International Journal of Image Processing and Pattern Recognition* 6.1 (2020): 5–13.
5. Ranjan, Amit, et al. "Generating novel molecule for target protein (SARS-CoV-2) using drug-target interaction based on graph neural network." *Network Modeling Analysis in Health Informatics and Bioinformatics* 11.1 (2022): 1–11.
6. Hsieh, Kevin, Aaron Harlap, Nandita Vijaykumar, Dimitris Konomis, Gregory R. Ganger, Phillip B. Gibbons, and Onur Mutlu. "Gaia: Geo-distributed machine learning approaching LAN speeds." In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, pp. 629–647. 2017.
7. Dean, Jeffrey, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc’aurilio Ranzato et al. "Large scale distributed deep networks." In *Advances in neural information processing systems*, pp. 1223–1231. 2012.
8. Konečný, Jakub, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. "Federated learning: Strategies for improving communication efficiency." *arXiv preprint arXiv:1610.05492* (2018).
9. Hoang, Quan, Tu Dinh Nguyen, Trung Le, and Dinh Phung. "Multi-generator generative adversarial nets." *arXiv preprint arXiv:1708.02556* (2017).
10. Durugkar, Ishan, Ian Gemp, and Sridhar Mahadevan. "Generative multi-adversarial networks." *arXiv preprint arXiv:1611.01673* (2018).
11. Ranjan, Amit, et al. "Transfer Learning Based Approach for Pneumonia Detection Using Customized VGG16 Deep Learning Model." *International Conference on Internet of Things and Connected Technologies*. Springer, Cham, 2021.
12. Ghosh, Arnab, Viveka Kulharia, Vinay P. Namboodiri, Philip HS Torr, and Puneet K. Dokania. "Multi-agent diverse generative adversarial networks." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8513–8521. 2018.
13. Deb, Sagar Deep, et al. "A multi model ensemble based deep convolution neural network structure for detection of COVID19." *Biomedical Signal Processing and Control* 71 (2022): 103126.
14. Yonetani, Ryo, Tomohiro Takahashi, Atsushi Hashimoto, and Yoshitaka Ushiku. "Decentralized Learning of Generative Adversarial Networks from Non-iid Data." *arXiv preprint arXiv:1905.09684* (2019).

15. Fan, Chenyou, and Ping Liu. "Federated Generative Adversarial Learning." arXiv preprint arXiv:2005.03793 (2020).
16. A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.
17. Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv preprint arXiv:1511.06434 (2015).
18. Yang, Chao-Tung, et al. "An energy-efficient cloud system with novel dynamic resource allocation methods." *The Journal of Supercomputing* 75.8 (2019): 4408–4429.
19. Verma, Vinod Kumar, et al. "Next-generation Internet of things and cloud security solutions." *International Journal of Distributed Sensor Networks* 15.3 (2019): 1550147719835098.

Handwritten Digit Recognition Using Neural Network with Gabor Filter for Information Fusion



Akshay Kumar and Brindha Murugan

Keywords Machine learning · Image processing · Pattern recognition · Computer vision

1 Introduction

Image processing and machine learning are domains in engineering and computer science that are continuously advancing. In the broader domains of pattern recognition and computer vision, multidimensional image processing and analytics is now a key area of study and innovation. Pattern recognition is involved with the categorization of items into classes, particularly by machines, such as digit recognition in image analysis of digital and handwritten documents. This pattern recognition approach predicts objects using both image processing and machine learning. During the course of the past decade, electronic document management systems have brought many benefits to society. Software tools such as word processors, computer-aided design (CAD) software packages, drawing programs, and markup languages are widely used in creating, storing, and retrieving documents in format that can be understood by the human computer. In the given format, edits can be made easily to any document, it can be copied to a paper, or it can be distributed and published electronically over global networks [1]. Additional processing tools and utilities are used in situations where the document is in a machine legible or an easy-to-understand format. These tools include researching keywords or patterns in long documents, applying optimization algorithms or simulations on things like designing electronic circuits, or improving the visual quality of pages, books, or photos by removing noise that can be the result of years of deterioration [2]. However, when the document is on paper, none of the

A. Kumar (✉) · B. Murugan
Department of Computer Science and Engineering, National Institute of Technology,
Tiruchirappalli, India
e-mail: brindham@nitt.edu

abovementioned tasks can be performed by a computer. There are a large number of paper documents that need to be processed and handled for a multitude of reasons. A rough estimate states that more than 250 billion USD is spent around the world to extract information from paper documents annually, and this number is only one portion of the cost which is from skilled human labor. Only after this process of data entry can extraction be automated or computerized and significant cost savings be seen. Apart from this, the amount of data that is being brought online is also increased exponentially. In each of the applications, one of the significant priorities is to get the data enclosed in the documents and also to store the extracted data in a digital format. This is one of the fundamental motivations of a document analysis systems working on digital basis. The most important component of electronic document analysis systems is optical character recognition (OCR). The answer can be found at the crossroads of pattern recognition, image processing, and machine learning. Despite extensive research, the state of the art in OCR has just recently reached the stage of partial utilization. Handwriting recognition is only partially successful, especially for separated and neatly hand-printed characters and words with a limited lexicon. Throughout more than three decades of sustained work, the recognition of freestyle handwriting as shown in Fig. 1 remains a study topic.

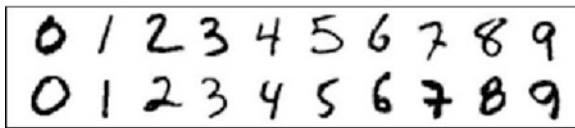


Fig. 1 Freestyle handwritten digits

2 Related Work

Purkaystha et al. [1] propose a convolutional neural network model for recognizing handwritten Bengali characters.

They developed a convolutional neural network with two convolution layers, three densely connected layers, and the dense layer softmax function as the last layer. The first layer analyzes the entire image for a 5×5 receptive field. It is then passed through the ReLu activation mechanism after scanning. This layer's output is then sent to a max pooling layer with a size of 2×2 . This output is then sent to a second convolutional layer with 64 3×3 kernels. The same function ReLu and pooling of size 2×2 are used as in the first layer. This layer's output is actually employed in the categorization process. They were 98.66% accurate with numerals, 94.99% accurate with vowels, 91.23% accurate with alphabets, and 89.93% accurate with all Bengali characters. However, the limitation is that there are some errors in their prediction due to misleading in the dataset.

Roohi and Behnam [2] had investigated the performance of deep convolutional neural network on the Persian handwritten character dataset. In this work, two types of CNN were used to see if they could outperform the standard technique in the PHCR problem. The first network is a single convolutional neural network (SCN), which is based on the CNN structure (LeNet-5). The bagging concept was applied to CNN with a range of network settings to extend it into ensemble CNN. This study found that when an ensemble CNN model was used, the accuracy was 97.1%, and when a single CNN was used, the accuracy was 96.3%. However, the shortcoming of the work is that not every instance of ensemble CNN method outperforms a simple neural network. Also, the training time for this model is not optimal.

Teow [3] presented a work on how to bridge the gap on understanding the mathematical structure and the computational implementation of a CNN using minimal model. They adopted a hybrid CNN with two networks as a starting point. The feature learning network and the classification network are the two networks. The goal of a fully connected network is to learn visual features without supervision. Feature learning using high-level representation will synthesize the learned picture feature. Finally, the classification will use this high-level image to conduct image recognition. They employed the MNIST dataset for handwritten digit recognition, which contains 70,000 images of handwritten digits, 60,000 of which are used for training and 10,000 for testing. Each number is a grayscale image with a size of 28×28 pixels. With epoch 10, the minimum model performs 97.3%, while the extended minimal model performs 98.50%. However, the limitation of this method is it needs considerable amount of training time to give comparable performance to more optimized methods.

Dhande and Kharat [4] presented the work related to recognition of cursive English handwriting. The letters in a word are linked together in cursive English handwriting. As a result, segmenting and extracting features from cursive English script is extremely challenging. They employed horizontal and vertical projection methods for segmentation to overcome this challenge. For feature extraction, the convex hull technique is utilized, and SVM is used for recognition and classification. However, the model is very slow and inaccurate in comparison to neural networks.

Thangamariappan and Pamila [5] propose a system using simple multilayer perceptron (MLP) as their neural network model. There are 784 input neurons in the neural network model. There are two hidden layers in use. Hidden layer 1 uses 512 neurons, while hidden layer 2 uses 256 neurons. The activation function for the hidden layers is ReLu, while the activation function for the output layer is softmax. The model is fitted using the MNIST data set. There are 70,000 handwritten digits in the MNIST data collection. There are 60,000 photos used as training examples and 10,000 images used as testing examples in this study. Special Database 1 and Special Database 3 are combined in the MNIST database. The Special Database 1 is made up of digits written by children in school. Employees' digits are stored in Special Database 3. This model has a test accuracy of 98.5% and a test score of 88.3%. However, this method incorporates large preprocessing overhead with no significant impact on accuracy.

Bautista [6] investigated the accuracy and precision of the proposed system by cross-examining the values solved using their system with the values solved manually. Because the accuracy and precision of optical character recognition were directly influenced by the feature extractor and classifier, the authors explored using the combination of projection histogram and support-vector machine as a feature extractor-classifier for handwritten characters (SVM). There were three stages to the model: the preprocessing or feature extraction stage, followed by the SVM-based recognition stage, and the solving equations and accuracy measurement as the final stage. The SVM was trained with linear, polynomial, and RBF kernels, with 90 training photos per character (a total of 5580 images), and a database containing the unique features that represent each character was constructed. However, the limitation is that the system is undertrained and thus needs more training data to achieve comparable accuracy.

3 Methodology

The methodology used in this paper is to split the complete detection into two parts. Preprocessing on the input image and prediction of handwriting in the image; i.e., for each input, the complete model performs the following task: (1) applying preprocessing algorithm on input image and (2) applying neural network algorithm for creating model and predicting handwriting. Both methodologies are being discussed one by one.

3.1 Preprocessing Algorithm

Various preprocessing steps are used throughout the paper.

Flip and rotate image: The input dataset which the proposed model is using is having images of size 28×28 . All these images are vertically flipped and 90° anticlockwise rotated. To convert them into original image, system needs to flip them and then rotate them 90° clockwise. Image before and after modification are as shown in Fig. 2.

Normalizing image: Prior to delivering photos as input to a deep learning neural network model for training or assessment, the pixel values in the images must be transformed. Typically, images would have to be scaled prior to model training and saved in the scaled representation in memory or on storage.

RGB to grayscale conversion: When user is giving an image that may be of RGB type, for processing, it is always preferred that the image should be in black and white format. In order to convert the RGB image to black and white, the system first has to convert it into a grayscale image. This grayscale image is basically an image where each pixel is represented in terms of an amount of light.

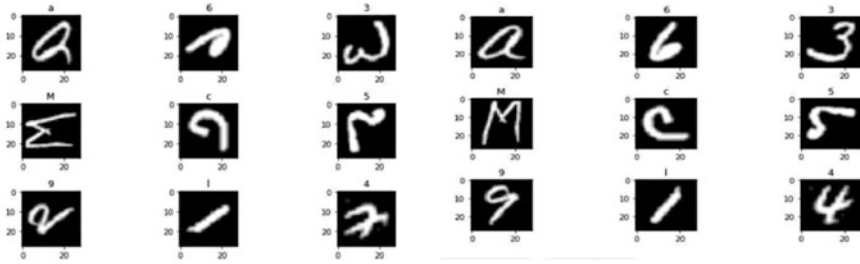


Fig. 2 Original image and rotated+flipped image

Grayscale to black and white: After getting grayscale image, the system will convert it into black and white image. The threshold value for this conversion is set to 127.

Information extraction from image: It is done using Gabor filters on the image. The Gabor filter extraction of features procedure begins with a two-dimensional Gabor filter that is applied to each image separately. Gabor’s uncertainty principle governs the procedure, stating that the product of frequency resolutions and time must be larger than a given constant. This principle might help to choose better orientations and frequencies. Filter bank of two Gabor filters is used for extracting mean and standard deviation. Figure 3 shows the filter bank used.

Passing through filter bank: Each digit image is passed through the proposed filter bank to create the fused projection input for the training model. Figure 4 displays the image after passing through filter bank.

3.2 Prediction Algorithm

Once the input image gets divided into individual letter image, the next task is to predict the digit present in that image. For doing that, first training of model is done. This model will give output as the predicted number present in the image. For training the model, “artificial neural network” is used.

An ANN is made up of three or more interconnected layers. Figure 4 shows sample neural network architecture. The first layer is made up of input neurons, which send data to deeper layers, which then send final output to the final output layer. Every layer serves as an input and output layer for the ANN, allowing it to comprehend more complicated things. Each layer is made up of units that use a number of transformations to adapt information from one layer to the next. Each of these layers is referred to as a neural layer collectively. The neural layer’s units attempt to learn about the data collected by weighing it according to the ANN’s internal framework. These rules enable units to provide an altered result, which is subsequently sent to the following layer as an output. Additionally, the ANN uses backpropagation, a process through which errors are taken into account when it

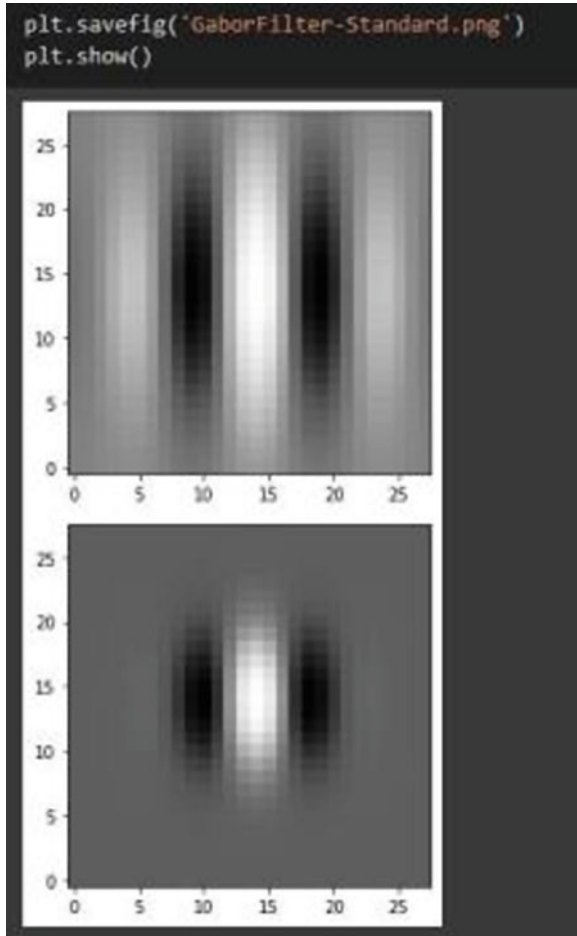


Fig. 3 Filter bank

comes time to adjust its results based on past errors. During the supervised training phase, the ANN classifies any inaccurate output as an error via backpropagation, and then the information is passed backward. Each weight is adjusted in accordance to how much they contributed to the inaccuracy. The model which is used in the proposed work consists of parameters like layers, output shape after every layer, and number of parameters for training.

Dataset For our model, we are using Modified National Institute of Standards and Technology dataset which is also known as MNIST dataset. MNIST handwritten digit dataset: total classes = 10. All these datasets contain numbers of images of size 28×28 . Among all, MNIST by class is used for training the model. It contains 60,000 training and 10,000 testing images. In the training set, it is further dividing

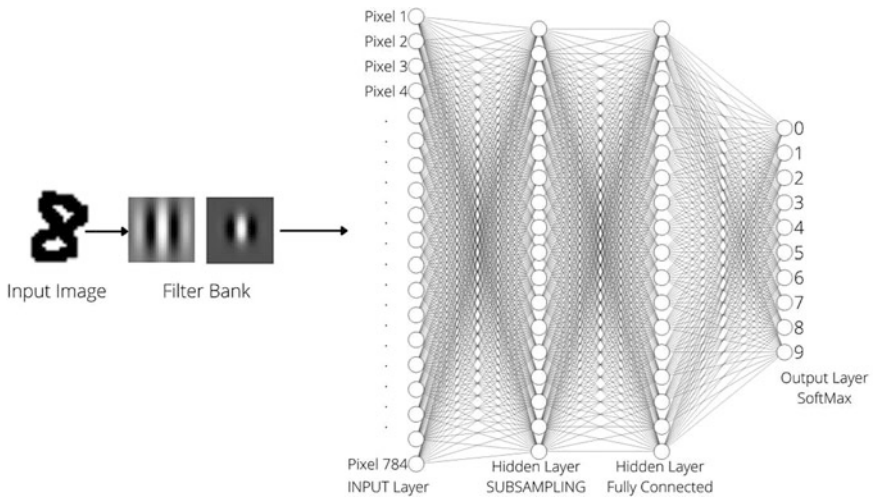


Fig. 4 Proposed architecture

into 75% training and 25% validation dataset. The dataset is unbalanced, i.e., the number of images in each class is not uniform. There are total 62 classes in which dataset is distributed. These classes are from 0–9 (10), A–Z (26), and a–z (26).

4 Results and Outputs

In this section, it is discussed about the working of the model. There is also information on how the model performs the task of handwritten digit recognition and also how effectively it produces results. The section elaborates on how this model compares with existing systems in place for a similar problem statement.

Table 1 Accuracy loss and performance (in time)

Algorithm	True classified digit (in percentage)	False classified digit (in percentage)	Time taken for prediction (in seconds)
Neural Network	97.38	2.62	0.56
CNN	98.37	1.63	3.62
Support Vector Machine	95.88	4.12	1.29
Naïve Bayes	90.83	9.17	3.45
KNN	86.58	13.42	0.55

Training the Model The model is being trained or compiled using Adam optimizer with the features extracted from Gabor filter bank. These features are used to feed forward into the network and to backpropagate in case of misclassification. Major purpose of the model being trained according to proposed method is to reduce loss and increase accuracy without overfitting of the model. Table 1 displays accuracy and loss according to epoch. It is evident from the table that with each consecutive epoch the accuracy of the model can be seen increasing, and the loss conversely reduces. This goes on to show that with each training epoch the model improves itself.

Note The saturation point in accuracy metric with reference to epochs is when by training the model for another epoch brings no significant change in accuracy. This training for another epoch only serves to add additional computational overhead to the training phase of the model. Figure 5 shows saturation curve of accuracy with increase in epochs.

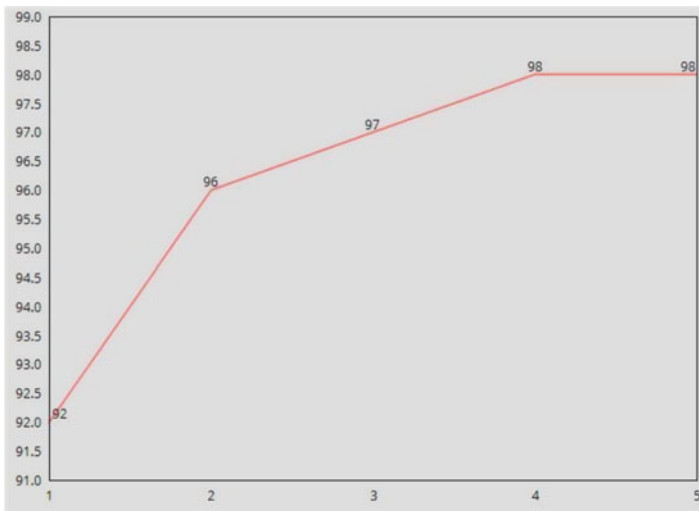


Fig. 5 Accuracy vs. epoch graph

4.1 Accuracy and Loss Graph

Accuracy is the measure of how correctly the model is able to predict digits in the training dataset. Loss on the other hand tells how many false positives the model produces in the process of training, validation, and testing. It is evident that, upon

increasing the number of epochs, the accuracy increases while the loss decreases as the model gets more robust.

4.2 Comparison with Other Models

At last, comparison of the model is shown with other digit recognition models. Figure 6 displays the comparison between performance of neural networks with other algorithms upon various metrics such as accuracy, F1 score, precision, and recall. Also, the time taken for a classifier to give results is key. Figure 7 shows the

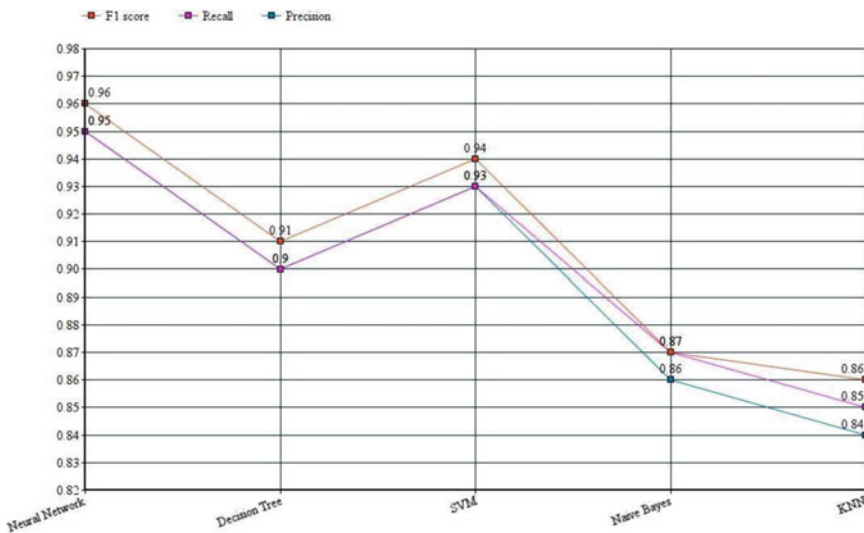


Fig. 6 Comparison with other models

comparison in performance of these algorithms. The performance of the proposed neural network model is compared with state-of-the-art convoluted neural network and with other classification algorithms. This is only a measure of how correctly or incorrectly the algorithm is able to identify any digit given to it for classification. Although we can see that CNN shows better accuracy, the trade-off with prediction time is very high. Thus, we can conclude that the proposed method is superior in performance to all other algorithms.

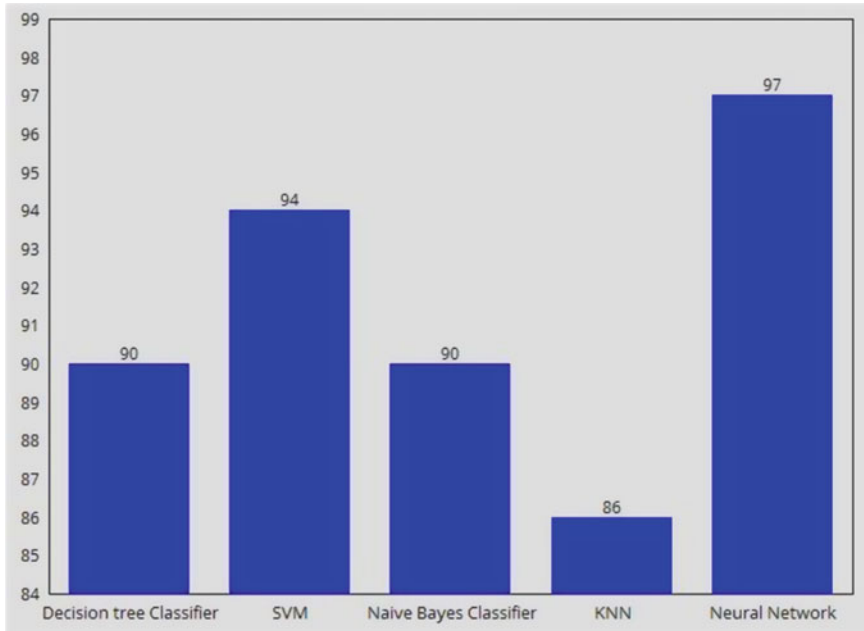


Fig. 7 Accuracy comparison with other models

5 Conclusion

Handwriting recognition is a vast topic for research due to different parameters like variation in handwriting, cursive writing, and large number of regional languages. Due to time limitation and also the goal of creating a semifunctional handwritten digit recognizer for demonstration purposes, just a few experiments were conducted to compare the various settings. An existing dataset of handwritten digits was used to generate a list of 700 digits per class. This collection was divided into two parts: a training set of examples and a test set of questions. Previous studies found a maximum accurate categorization score of 92% on the identical test set (percentage of correctly recognized digits). Using digit components, this statistic was boosted to 98%, which was judged suitable for use in a demonstration. Each digit class was represented by one network, for a total of ten networks in these studies. The best classification results for a selection of digit sizes are provided in this table for a range of fragment sizes using minimal distance classification. From the experiments, we can conclude that as the number of epochs increases, accuracy of model increases and loss decreases. Also, if the number of samples in training is high for certain character, then accuracy for prediction of that character is higher. The prediction of word is having better accuracy if we apply image preprocessing algorithms. Also, it is verified that accuracy would increase if the number of hidden layers is increased,

but it will take more time for training. For the proposed model, an accuracy of up to 97% is obtained for detection of numbers.

References

1. Bishwajit Purkaystha, Tapos Datta, Md Saiful Islam, "Bengali Handwritten Character Recognition Using Deep Convolutional Neural Network", 2017 20th International Conference of Computer and Information technology (ICCIT), (2017)
2. Samad Roohi, Behnam Alizadehashrafi, "Persian Handwritten Character Recognition Using Convolutional Neural Network," 10th Iranian Conference on Machine Vision and Image Processing, (2017)
3. Matthew Y.W. Teow "Understanding Convolutional Neural Network Using A Minimal Model for handwritten Digit Recognition", IEEE 2nd International conferences on automatic and intelligent system, kota kinabalu, sabah, Malaysia, (2018)
4. Pritam Dhande, Reena Kharat "Recognition of cursive English handwritten characters", International Conference on Trends in Electronics and Informatics (ICEI), Tirunelveli, India,(2017)
5. P. Thangamariappan, J. C. Miraclin Joyce Pamila "HANDWRITTEN RECOGNITION BY USING MACHINE LEARNING APPROACH", International Journal of Engineering Applied Sciences and Technology, 2020 Vol. 4, Issue 11, ISSN No. 2455-2143, Pages 564-567, (2020)
6. Bautista SA, Vishnu, Navata, Aldrich H.N, Timothy S, Justine D, Edison A. Roxas, "Recognition of Handwritten Alphanumeric Characters using Projection Histogram and Support Vector Machine", 8th IEEE International Conference Humanoid, Nanotechnology, Information Technology Communication and Control, Environment and Management (HNICEM) The Institute of Electrical and Electronics Engineers Inc, (2015)

FAFOC: Fog-Based Energy-Efficient Clustering Technique for Wireless Sensor Networks



R. Dayana and G. Maria Kalavathy

Keywords Fog computing · WSN · Clustering · Fuzzy logic · Artificial flora · Energy efficiency · Network lifetime

1 Introduction

In general, the proficient deployment of Internet of Things (IoT) examines the atmosphere and performs specific operations. Hence, the responsibility of a computing method which manages the IoT devices is highly effective. The IoT devices are mainly composed of sensors, and a collection of ubiquitous sensors forms the wireless sensor network (WSN). Initially, cloud computing (CC) depends upon the WSN and is utilized for managing the sensor data [1]. Transmission and computation of massive data which is passed through CC resulted in various complexities like delayed service response time and additional energy requirement. Moreover, the domains of a WSN has been developed extensively and demands for practical computations and data transmission which finally enhances the IoT management task [2]. A novel computing method has been introduced, named as fog computing; in this process, the volume of data produced by a sensor is maximized, and regular data processing becomes a complicated issue.

Here, fog computing is a technique applied for data management and performs communication with the nearby field using a sensor. Since the devices in fog computing applies near-field communication, the response speed becomes robust and rapid when compared to CC. Besides, the sensor count assigned to a device is minimum than CC. Under the assumption of data aggregation, the power consumption can be reduced which has been caused because of the elimination

R. Dayana (✉)

Department of CSE, Jeppiaar Institute of Technology, Chennai, India

G. Maria Kalavathy

Department of CSE, St. Joseph's College of Engineering, Chennai, India

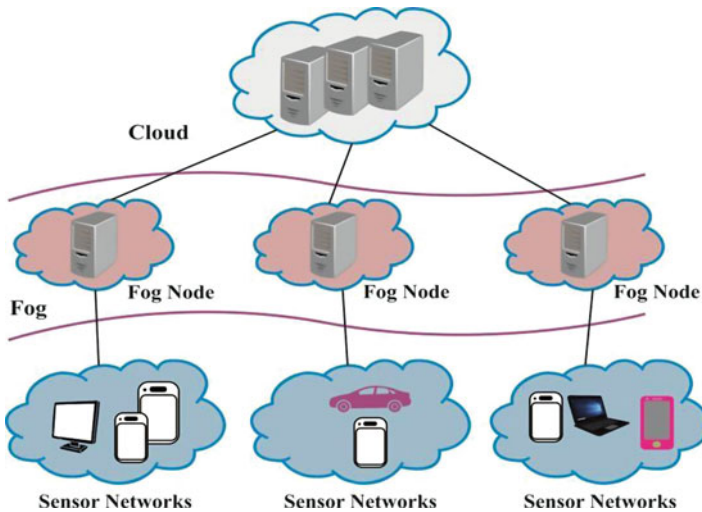


Fig. 1 Architecture of fog-based WSN

of redundant data transmission among nearby devices in a sensor node. On the other hand, the clustering-based hierarchical process offers massive benefits [3–5]. Furthermore, fog computing services are data development, computation, and transmission, as depicted in Fig. 1. Besides, a cloud server is helpful in extracting features from server on the basis of data type when it was actually created. Followed by, a server contains set of rules, pattern analysis, as well as inference for data saving.

Data that is computed is fed into a data owner or authorized server. At this point, data used for a service user has been saved in the form of plain text and transmitted to observing server so that it can be applied in future. Data distribution and applications are the predictable operations. Data which is produced in a sensor might be an ineffective one for the user; hence, a delegation function is to assign a centralized manager for accessing the data, and revocation function is applied to eliminate the authentication for accessing the data as the user required.

Fog computing [6] is defined as a virtual environment which offers computation, storage, and networking service among a device as well as CC data center. Therefore, it has not been placed uniquely at the edge of network. The operations like computation, storage, and networking are considered to be the building blocks of cloud and fog. The cloud layer is considered to be the main theme of fog computing that computes data virtualization, examination, and machine learning (ML) and upgrades the rules and patterns from fog layer's proxies. The proposed fog-based WSN involves a set of two nodes, namely, advanced nodes, normal nodes, and some FNs. The existence of advanced nodes makes the network heterogeneous, which results in maximum energy efficiency of the network. Though the fog based

WSN offers several benefits, there is a need to develop an energy-efficient clustering technique to select the CHs and organize clusters.

This paper presents a new fuzzy with artificial flora optimization-based clustering (FAFOC) algorithm for CHs selection to attain maximum energy efficiency in fog-based WSN. The proposed FAFOC algorithm involves three major stages, namely, node initialization, tentative CH selection, and final CH selection. Initially, the nodes undergo random deployment, and initialization process takes place. Then, the FAFOC selects the CHs in two stages, namely, tentative CH selection and final CH selection. At the initial stage, all the nodes in fog-based WSN undergoes fuzzy logic-based tentative CH selection (FL-TCHS) process which elects a set of tentative nodes based on residual energy (RE), distance to base station (DBS), and node centrality (NC). Then, the tentative CHs execute the artificial flora optimization-based final CH selection (AFA-FCHS) process to select the final CHs. Once the CHs were effectively chosen, the nearby nodes join the CHs and forms clusters. Finally, the cluster members (CM) send the data to CHs, which then forward it to the fog nodes subsequently to cloud. The experimental validation of the proposed FAFOC algorithm is carried out, and the results are examined under several dimensions.

2 Related Works

Energy application management of WSN has gained massive attention from research communities. The developers have examined all other solutions from the application of another power sources, named as solar energy [7], and some other research have focused on the development of power-aware protocol models [8, 9]. Using the objective of enhancing the entire network operation effectively, fog computing is mainly applied for extending the CC model to the network edge [10]. Fog computing is mainly deployed for addressing the issues over mobility, geography, and delay requirements of IoT systems [11]. Because of the demand while installing massive interconnected devices, energy maintenance and self-recovering applications are the two important developing attributes of fog computing environments.

In Al-Fuqaha et al. [12], the major features behind network mechanisms of a fog computing have been examined such as position, distribution, reliability, density of machines, mobility support, real time, and standardization. In addition, the network structure in fog computing devices should be independent and effective with respect to power application as well as network resilience. Due to the network requirements, IEEE 802.15.4 ZigBee standard from ZigBee has inspired the concentration of developers. Diverse researchers are effective and provide the advantages of ZigBee in various Smart City as well as Smart Farming domains. In Lee and Wang [13], the researchers have determined the connection, packet loss rate, and transmission throughput from indoor research region. A hybrid network model, ZigBee/5G, is presented in [14].

The developers have demonstrated efficiency of combining diverse protocols which aims at limiting the energy application. Followed by, the developers from [15] have determined and related the power energy utilization of ZigBee vs. the Advanced and Adaptive Network Technology (ANT). Based on the survey, ANT performs well in ZigBee with respect to power application executed by using sleep and wake technology. Hence, it is stated that, an appropriate management of entire system, radio, and sensors, the energy application addressed by ZigBee systems are reduced gradually. The final outcomes make ZigBee a tremendous environment to learn the advantages of combining energy application management models. Moreover, the extensive application of ZigBee tends to make greater platform for developing Smart City as well as Smart Farming applications. Recently, Piromalis and Arvantis [16] established a reliable hardware structure for WSN and actuator networks which has to be applied in Smart Farming domains. Likewise, the structure of sensor node concentrates on the energy application as well as network resilience methodologies are considered as advanced innovations on fog computing. It refers that the newly developed models takes the structure of sensor node, with main characteristics of fog computing model, especially geographical position, delay, as well as energy provisioning.

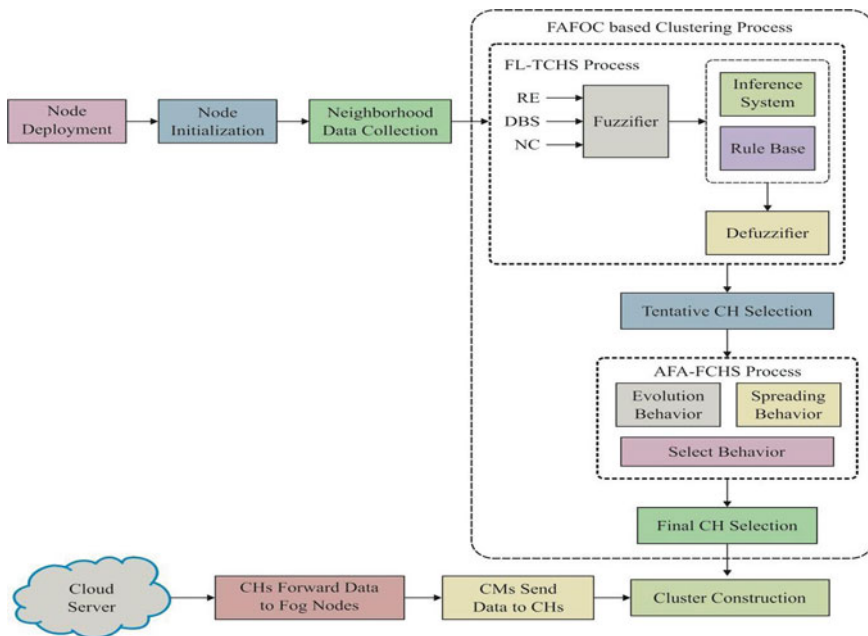


Fig. 2 Block diagram of FAFOC algorithm

3 The Proposed FAFOC Algorithm

The working process involved in the FAFOC algorithm is demonstrated in Fig. 2. The proposed FAFOC algorithm involves three major stages, namely, node initialization, tentative CH selection, and final CH selection. At the beginning stage, the nodes are randomly deployed and initialized, and information collection about nearby nodes takes place. Then, the FL-TCHS process gets executed and selects the tentative CHs. Afterwards, the tentative CHs performs the AFA-FCHS process and elect the final CHs. Once the CHs were chosen, the data transmission process begins where the data from CMs to cloud takes place via CHs and fog nodes in the network.

3.1 Fuzzy Logic-Based Tentative CH Selection Process

In this section, the tentative CHs were selected using fuzzy logic with the use of three input parameters, namely, RE, DBS, and NC [17].

- (i) *Residual Energy*: It plays a significant role for selecting a node as CH which spends maximum energy when compared with CM. A CH node gathers data from members, accumulate the data gathered, and transmits to sink node or BS. Hence, competing energy level is essential for a CH in order to execute the predefined events.
- (ii) *Node Centrality (NC)*: Total count of one-hop adjacent nodes inside R_c of a node is named as node degree. NC is a factor that calculates the node's position in the intermediate location over its neighbors. Lower NC measure offers greater opportunity of selecting a node as CH:

$$NC = \frac{\sqrt{\sum_{i=1}^{ND} dist_i^2 / ND}}{Ntk_Dimension} \quad (1)$$

In Eq. (3), ND (node degree) corresponds to count of neighbors inside a communication radius R_c of a node, and Ntk-Dimension value is “ M ” in $M \times M$ field area, as well as $dist_i^2$ implies a distance with i th neighbor node which refers that, in $100\ m \times 100\ m$, Ntk-Dimension is 100, and, in $200\ m \times 200\ m$ field area, Ntk-Dimension is 200.

- (iii) *Distance to BS*: The power conservation increases while transmitting data and also the distance among a transmitter and receiver nodes. From the point of power conservation, the distance among CH and BS has to be reduced:

$$Distance\ to\ BS = \frac{d_i}{\alpha \cdot Ntk_Dimension}, \quad (2)$$

$$\alpha = \frac{d_{max}}{Ntk_Dimension}$$

In Eq. (4), d_i refers to the distance among node i and the BS , d_{\max} denotes higher distance among a node in a system and BS , and A signifies a network dimension constant.

3.2 AFA-Based Final CH Selection Process

Once the tentative CHs are chosen, they compete for final CH selection using AFA. The AFA method is composed of four fundamental units, namely:

- Original plants
- Offspring plants
- Position of plants
- Propagation distance

Initially, the original plant means that the crops are ready for distributing the seeds. Secondly, the offspring plant is considered as the seed of original plant, and it is not suitable for spreading seeds in specific duration. Thirdly, the position of the plant refers the actual localities of the plant. Finally, propagation distance defines the distance of seeds spreading. It is composed of three main behavioral patterns like:

- Evolution
- Spreading
- Select

At the first stage, evolution behavior describes the possibility of time taken by a plant for adapting with the ecological behavior. Secondly, the spreading behavior defines the distribution of the seed. Finally, the select behavior refers that flora might stay alive or becomes expired because of the atmosphere [18].

Evolution Behavior

The original plants distribute seeds over the circle with radius named as propagation distance, which has been developed from propagation distances of the parent as well as grandparent plants.

$$d_j = d_{1j} \times \text{ran}(0, 1) \times c_1 + d_{2j} \times \text{rand}(0, 1) \times c_2 \quad (3)$$

where d_{1j} and d_{2j} denotes the propagation distance of the grandparent and parent plants, c_1 and c_2 defines the learning coefficients, and $\text{rand}(0,1)$ stands for autonomous and uniformly distributed within $(0,1)$.

A novel grandparent propagation distance is as follows:

$$d'_{1j} = d_{2j} \tag{4}$$

The novel parent propagation distance is meant to be a standard deviation (SD) among the positions of original plant as well as offspring plant:

$$d'_{2j} = \frac{\sqrt{\sum_{i=1}^N (i,j - P'_i)^2}}{N} \tag{5}$$

Spreading Behavior

Initially, the AFA has produced the data randomly from original flora with N solutions, where N plants are existed in flora. The location of the original plant is represented by matrix P_i , where i denotes a dimension and j refers to the flora plant count.

$$P_{i,j} = \text{ran}(0, 1) \times d \times 2 - d \tag{6}$$

where d indicates the higher limit area and $\text{rand}(0,1)$ denotes the array of arbitrary values which are distributed uniformly from (0,1) .

The location of the offspring plant can be produced on the basis of a propagation function in the following:

$$P'_{i,j \times m} = D_{i,j \times m} + P_{i,j} \tag{7}$$

where m refers count of seeds which a single plant propagates, $P'_{i,j \times m}$ means a location of offspring plant, $P_{i,j}$ stands for location of original plant, and $D_{i,j \times m}$ signifies a random value with the Gaussian distribution of mean 0 and varianced_j . When none of the offspring plant existed, then a new plant would be produced on the basis of Eq. (6).

Select Behavior

The active plant can be computed using survival probability as given in the following:

$$p = \left| \frac{\sqrt{F(P'_{i,j \times m})}}{F_{\max}} \right| \times Q_x^{(j \times m - 1)} \tag{8}$$

where $Q_x^{(j \times m - 1)}$ is Q_x to the power of $(j \times m - 1)$ and Q_x implies the selective probability. It can be viewed that the fitness of offspring plants are far away from

Table 1 Parameter setting

Parameters	Value
Area	$100 \times 100 \text{ m}^2$
E_0	0.5 J
Node count	500
E_{elec}	50 nJ/bit
ϵ_{fs}	10 pJ/bit/m ²
ϵ_{mp}	0.0013 pJ/bit/m ⁴
Packet size	4000 bits

actual plant. Q_x estimates the searching ability capability of the method. Q_x has to be maximum for the problem which is simple to enter the local optimal solution. F_{max} refers to the higher fitness in flora, and $(P'_{i,j \times m})$ denotes the fitness of j -th solution.

Roulette wheel selection approach has been applied for determining whether the plant is active or not. The main objective of this model is to “accept according probability”; which means that there may various instances and it is composed of a potential value. Therefore, selection is completely based on value of potential score. The maximum the score, then the higher the accepting probability becomes.

4 Performance Validation

In this section, an extensive experimental analysis is carried out, and the results are examined under diverse aspects. The parameter settings involved in the simulation process is shown in Table 1.

4.1 Network Lifetime Analysis

Figure 3 shows the network lifetime analysis of the FAFOC model in terms of FND, HND, and LND. The figure portrayed that the FAFOC model has delayed the death of the first as well as the last node in the network. It is noted that the FND of the FAFOC model is 391 rounds, whereas the FND of FEAR, P-SEP, QACMOR, and AFA is 75, 198, 88, and 111 rounds, respectively. At the same time, it is observed that the FAFOC model has attained the HND of 1315 rounds, whereas the HND of FEAR, P-SEP, QACMOR, and AFA is 455, 984, 625, and 711 rounds, respectively. At last, it is exhibited that the FAFOC model has delayed the LND to 1989 rounds, whereas the LND occurs at the earlier rounds of 995, 1612, 544, and 1329 by FEAR, P-SEP, QACMOR, and AFA algorithms.

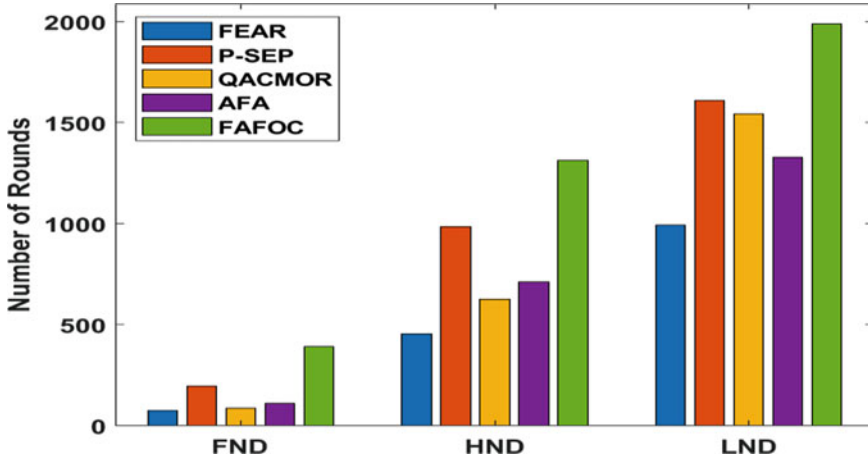


Fig. 3 Network lifetime analysis of FAFOC model

Figure 4 observes the performance of the FAFOC method in terms of PDR under varying node count. The figure exhibited that the FEAR model has depicted PDR over the compared approaches. Concurrently, the QACMOR and AFA models have illustrated certainly higher PDR over the FEAR model. However, the P-SEP model has tried to demonstrate moderate PDR. Yet, the highest PDR has been achieved by the projected FAFOC algorithm. For instance, under the node count of 50, the PDR by the presented FAFOC model is 0.847, whereas the PDR by FEAR, P-SEP, QACMOR, and AFA is 0.777, 0.677, 0.731, and 0.747 correspondingly. Also, under the node count of 500, the PDR by the projected FAFOC model is 0.727, while the PDR by FEAR, P-SEP, QACMOR, and AFA is 0.657, 0.512, 0.527, and 0.567, correspondingly.

Figure 5 showcases the performance of the FAFOC method with respect to average residual energy under varying number of rounds. The figure demonstrated that the FEAR method has shown minimum average residual energy over the compared methods. At the same time, the QACMOR and AFA methods have illustrated certainly superior average residual energy over the FEAR model. However, the P-SEP model has tried to demonstrate moderate average residual energy. However, the highest average residual energy has been attained by the presented FAFOC algorithm. For instance, under the round of 500, the average residual energy by the presented FAFOC model is 0.489J, while the average residual energy by FEAR, P-SEP, QACMOR, and AFA is 0.277J, 0.4875J, 0.3925J, and 0.3425J, respectively. Similarly, under the round number of 1500, the average residual energy by the proposed FAFOC model is 0.35J, whereas the average residual energy by FEAR, P-SEP, QACMOR, and AFA is 0J, 0.225J, 0.1J, and 0.09J, correspondingly.

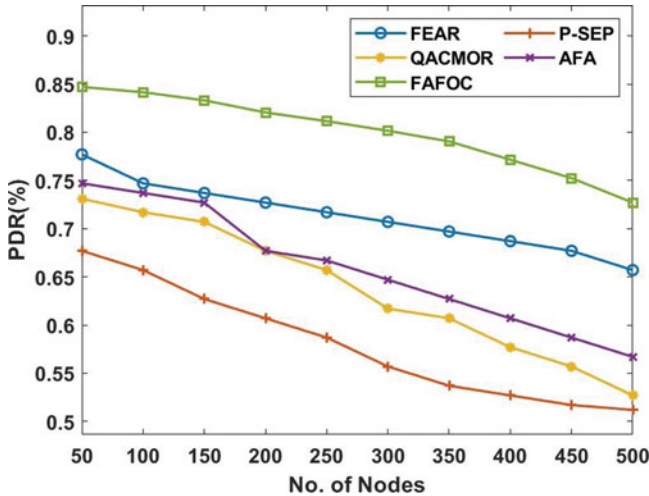


Fig. 4 PDR analysis of FAFOC model

Figure 6 depicts the performance of the FAFOC method with respect to packet loss under varying node count. The figure demonstrated that the FEAR model has shown the highest packet loss over the compared methods. At the same time, the QACMOR and AFA models have shown certainly lower packet loss over the FEAR model. However, the P-SEP model has tried to demonstrate minimum PDR. However, the low packet loss has been achieved by the projected FAFOC method. For instance, under the node count of 50, the packet loss by the projected FAFOC model is 0.02, while the PDR by FEAR, P-SEP, QACMOR, and AFA is 0.2, 0.11, 0.056, and 0.04, correspondingly. Likewise, under the node count of 500, the PDR by the presented FAFOC model is 0.14, but the packet loss by FEAR, P-SEP, QACMOR, and AFA is 0.489, 0.275, 0.26, and 0.22, respectively.

5 Conclusion

This paper has developed a new FAFOC algorithm for CHs selection to attain maximum energy efficiency in fog-based WSN. Initially, the nodes are randomly deployed and initialized, and information collection about nearby nodes takes place. Then, the FL-TCHS process gets executed and selects the tentative CHs. Afterwards, the tentative CHs performs the AFA-FCHS process and elect the final CHs. Once the CHs were chosen, the data transmission process begins where the data from CMs to cloud takes place via CHs and fog nodes in the network. An extensive experimental analysis is carried out, and the results are examined under

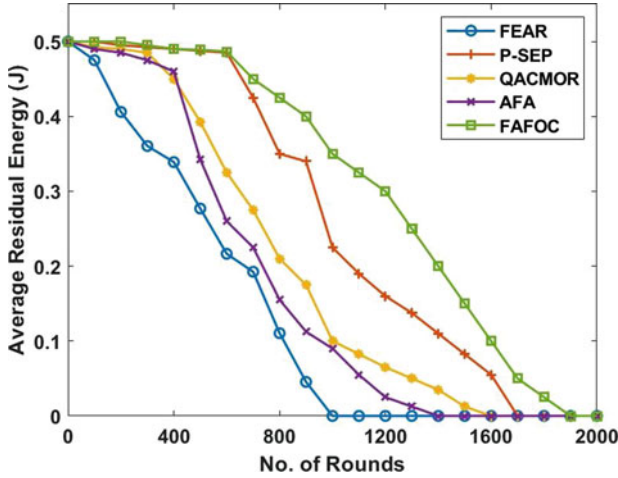


Fig. 5 Average residual energy analysis of FAFOC model

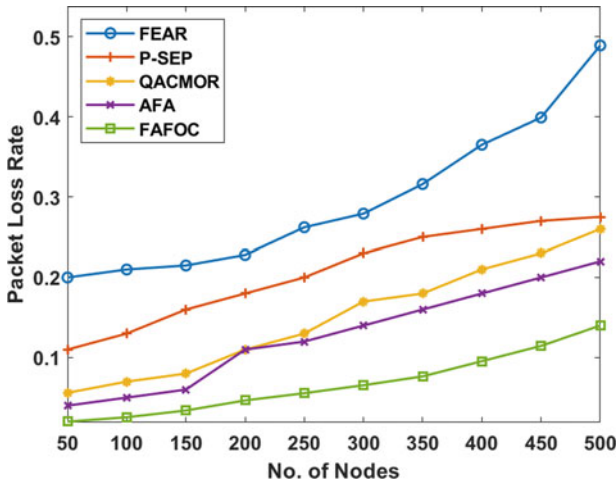


Fig. 6 Packet loss rate analysis of FAFOC model

diverse aspects. The experimentation outcome ensured the effective performance of the FAFPC algorithm in terms of energy efficiency, network lifetime, packet loss, PDR, and delay. As a part of future work, the network lifetime can be lengthened by the use of bio-inspired algorithm-based routing protocol to select the optimal forwarding node set.

References

1. Doukas, C.; Maglogiannis, I. Bringing IoT and cloud computing towards pervasive healthcare. In Proceedings of the 2012 Sixth International Conference Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), Palermo, Italy, 4–6 July 2012; pp. 922–926.
2. Stojmenovic, I.; Wen, S. The Fog computing paradigm: Scenarios and security issues. In Proceedings of the 2014 Federated Conference Computer Science and Information Systems (FedCSIS), Warsaw, Poland, 7–10 September 2014; pp. 1–8.
3. Krishnamachari, L.; Estrin, D.; Wicker, S. The impact of data aggregation in wireless sensor networks. In Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops, Vienna, Austria, 2–5 July 2002; pp. 575–578.
4. Lindsay, S.; Raghavendra, C.S.; Sivalingam, K.M. Data gathering in sensor networks using the energy delay metric. In Proceedings of the 15th International Parallel & Distributed Processing, 23–27 April 2001; IEEE Computer Society: Washington, DC, USA, 2001; p. 188.
5. Tan, H.O.; Korpeoglu, I. Power efficient data gathering and aggregation in wireless sensor networks. *ACM Sigmod Record* 2003, 32, 66–71.
6. Arkin, H.R.; Diyanat, A.; Pourkhalili, A. MIST: Fog-based data analytics scheme with cost-efficient resource provisioning for IoT crowdsensing applications. *J. Netw. Comput. Appl.* 2017, 82, 152–165.
7. W. B. Pramono, P. Setiawan, and Firdaus, “Solar power supply for ZigBee wireless sensor network,” in 2016 International Seminar on Application for Technology of Information and Communication (ISemantic), pp. 336–340, Semarang, Indonesia, 2016.
8. T. Matsui and H. Nishi, “ECORS: energy consumption oriented route selection for wireless sensor network,” in 2016 IEEE 14th International Conference on Industrial Informatics (INDIN), pp. 1044–1049, Poitiers, France, 2016.
9. C. Pérez-Garrido, F. González-Castaño, D. Chaves-Díeguez, and P. Rodríguez-Hernández, “Wireless remote monitoring of toxic gases in shipbuilding,” *Sensors*, vol. 14, no. 2, pp. 2981–3000, 2014.
10. F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, “Fog computing and its role in the internet of things,” in MCC ‘12 Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, pp. 13–16, New York, NY, USA, 2012.
11. D. Spirjakin, A. Baranov, A. Karelin, and A. Somov, “Wireless multi-sensor gas platform for environmental monitoring,” in 2015 IEEE Workshop on Environmental, Energy, and Structural Monitoring Sy
12. A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, “Internet of things: a survey on enabling technologies, protocols, and applications,” *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.
13. J.-S. Lee and Y.-M. Wang, “Experimental evaluation of ZigBee based wireless networks in indoor environments,” *Journal of Engineering*, vol. 2013, Article ID 286367, 9 pages, 2013.
14. J. Mu, “An improved AODV routing for the ZigBee heterogeneous networks in 5G environment,” *Ad Hoc Networks*, vol. 58, pp. 13–24, 2017.
15. S. Gharghan, R. Nordin, and M. Ismail, “An ultra-low power wireless sensor network for bicycle torque performance measurements,” *Sensors*, vol. 15, no. 5, pp. 11741–11768, 2015.
16. D. Piromalis and K. Arvanitis, “SensoTube: a scalable hardware design architecture for wireless sensors and actuators networks nodes in the agricultural domain,” *Sensors*, vol. 16, no. 8, p. 1227, 2016.
17. Balakrishnan, B. and Balachandran, S., 2017. FLECH: fuzzy logic based energy efficient clustering hierarchy for nonuniform wireless sensor networks. *Wireless Communications and Mobile Computing*, 2017.
18. Cheng, L., Wu, X.H. and Wang, Y., 2018. Artificial flora (AF) optimization algorithm. *Applied Sciences*, 8(3), p.3.

Evaluation of Supervised Classifiers for Fake News Detection Using Twitter Dataset



Vinita Nair and Jyoti Pareek

Keywords Lemmatization · Stemming · Supervised classification · Twitter

1 Introduction

On account of worldwide COVID-19 outbreak, government and public health institutions announced social distancing and stay-safe-at-home guidelines. People were more confined to home, which resulted in a tremendous need of digital media as a source of healthcare, information, shopping, entertainment, and education and to maintain social connections [10]. People were more inclined toward social media as a source of information, social connections, and well-being. Excessive usage of social media as a source of communication and information resulted in surged propagation of fake news. Any social media information appeared to be more reliable in the era of crisis. A survey was conducted by New York University and France's University Grenoble Alpes which concluded that misinformation received six times as many likes, shares, and interactions as legitimate news articles [4]. The uncurbed growth in misinformation was due to fear, panic during pandemic, low digital literacy, and high Internet penetration for media of communication and entertainment, resulting in over usage of social media platforms.

According to the survey conducted in 138 countries, social media produced the largest amount of misinformation accounting to 84.94%, whereas overall Internet services produced 90.5%, combined, which contributed for most of the COVID-19 misinformation [4, 6]. Social media platforms, namely, Facebook and Twitter, strongly follow the protocols to control the propagation of misinformation by mechanisms, namely, flagging the tweets and usages of fact-checkers to control the propagation among other digital users.

V. Nair (✉) · J. Pareek

Department of Computer Science, Gujarat University, Ahmedabad, India

e-mail: vinitanair@gujaratuniversity.ac.in; jspareek@gujaratuniversity.ac.in

Our paper discusses fake news detection model using a set of preprocessing and vectorization techniques further evaluating with supervised classifiers. The paper is distributed into sections such as Sect. 2 describing the existing work done in misinformation and fake news identification using supervised learning classification. Section 3 elaborates the entire workflow comprising of tweet scrapping, normalization, preprocessing, and vectorization. Section 4 represents classification using supervised classifiers. Section 5 summarizes evaluation and comparative results. Section 6 discusses the conclusion and future scope of research.

2 Related Work

The following section presents the work done by various researchers in the field of misinformation and fake news detection.

Agrawal et al. in [16] discussed machine learning and NLP techniques like bag-of-words, n-grams. They did the supervised classification for labeled dataset with five classifiers and compared the result using precision, F1, and recall and a comparison metric. The researchers concluded SVM model the best among others, generated the highest values for precision, 0.62; recall, 0.62; and F1 score, 0.61.

Espinosa et al. in [8] suggested a model for detecting fake news which involved identification of features based on content and context-based features extracted from tweets. The former may allow us to study the linguistic features extracted from the user-written text, and the latter is about user characteristics like network propagation of the text, user reactions. They applied tokenization, stop words removal, and tweet aggregation for 300 users' tweets. The extracted features were psychological, linguistic, twitter actions, and headline analysis data. The random forest classifier outperformed among all five classifiers, namely, SVM, logistic regression, KNN, decision tree, with accuracy, 0.68; precision, 0.7; and recall, 0.64.

Fersini et al. in [2] have proposed a solution to identify whether the given tweet is fake or not for English and Spanish languages. They have considered main characteristics for analyzing the tweets, namely, stylometry, personality, emotions, and feed embeddings. Their proposed model achieved accuracy of 60% and 72% for English and Spanish dataset, respectively. They also exploited their model using Myers-Briggs Type Indicator (MBTI) for studying the personality trait of the user based on the text.

Cardaioli et al. in [9] presented and implemented a framework for profiling fake-news spreaders by exploiting behavioral features, namely, personality and stylometry. They have compared various classification methods for supervised machine learning models combining the Big Five personality and stylometric features. After the evaluation and comparison, it was found that random forest performed the best among all others. Generated accuracy: English corpus, 0.6750; Spanish corpus, 0.7150; and average score between the two corpora, 0.6950.

3 Methodology

Our proposed model aims to classify tweets either real or fake based on supervised classification techniques. The implementation methodology commencing from tweets collection to generation of final outcome is represented (see Fig. 1). Each step involved in the entire process is described as below:

3.1 Tweet Extraction

The foremost step was acquiring the credentials for authorized usage of Twitter API using API key, API secret, access token, and access token secret. Followed by authorization, Python-based tweepy library was used for tweet extraction through Twitter API. Table 1 summarizes a few extracted tweet composites. For our experiment, we have considered only two tags, namely, text, describing the tweet posted by a user, and lang, describing the language of the tweet. We downloaded tweets related to COVID-19 outbreak in the English language.

Table 1 Listing of few tweet composites with description

Tweet tag	Description
created_at	Date of creation of tweet with timestamp
Id	The twitter ID which refers to the user who posted the tweet
Text	Textual description of the tweet which may consist of alphabets, number, hashtags, punctuations, user mentions, URL, and many more
Source	It points to the URL, which refers to the origin the text or URL been mentioned in the tweet
favorite_count, retweet_count	Number of favorites represents the popularity of the tweet Number of times the tweet is reposted on the wall
Lang	Represents the language of the tweet text (we have considered “lang” =” en”:- English for our study and experimentation)
User	The user’s entire profile consists of information such as user_id, name, screen_name, location, url, description, verified, follower_count, friends_count, created_at, and many more.
Entities	List of entities such as URLs, hashtags, user mentions, symbols
geo_enabled	Boolean value when true represents location shared by user consisting of latitude and longitude

3.2 *Preparing Dataset – Normalization and Preprocessing*

For experimentation, we have used the training corpus provided by PAN 20 [3] sectioned in two languages English and Spanish. We have considered an English language corpus which comprises 300 files; each file represents a unique user, containing 100 tweets per user: word-length of 140 characters, language of tweet. All the tweets are labeled with an identifier 0 or 1 representing the tweet as fake or real, respectively [8]. The corpus contained masked URLs, links, hashtags, and user mentions. Table 2 summarizes training and testing corpus used for experimentation.

Table 2 Training and testing corpus summary

Dataset	Source	Normalized	No. of users (unique)	Total count of tweets
Training	PAN 20	URLs, links,	300	30,000
Testing	Twitter API	hashtags, and user mentions	2087	6000

The tweets are raw, noisy, and unstructured in nature, which is not suggestive for direct usage in machine learning algorithms for feature extraction. Henceforth, the raw and unstructured text needs to be cleaned, normalized, and preprocessed. In our experiment, we have undergone a two-step mechanism for preparing our testing dataset prior to the classification and prediction.

- The first step is normalizing the scrapped tweets to a pattern mapping the training data set used for the experimentation. The tweets were obfuscated with regard to URLs, mentions, usernames, emojis, and hashtags with certain capital-lettered keywords using RegEx of Python as represented in Table 3.
- The second step is preprocessing of obfuscated text using various NLP libraries including the technique of lemmatization and stemming as depicted in Fig. 1. Undergoing the two-step mechanism – normalization and preprocessing of text – generates better results when compared using the different supervised classifiers.

3.3 *Tweet Preprocessing*

We have used scikit-learn [5], an enriched machine learning library in Python basically used for classification, regression, clustering, dimensionality reduction, model selection, and preprocessing. Python provides various rich NLP libraries like NLTK, SpaCy, Gensim, TextBlob, and many more; we have used NLTK library for preprocessing of tweets. Preprocessing includes tokenization, removal of digits, punctuation, special characters, and stop words. We have applied and compared the

Table 3 Tweet tokens with applied regular expression

Tweet tokens	Regular expression (Python)
Hashtags	hashtag_pattern = r"(?!\#+[\w_]+[\w'_-]*[\w_]+)"
Emoticons	#emoticons u"\U0001F600-\U0001F64F" #symbols & pictographs u"\U0001F300-\U0001F5FF" #transport & map symbols u"\U0001F680-\U0001F6FF" #flags u"\U0001F1E0-\U0001F1FF"
URLs	url_pattern = r"http[s]?://(?:[a-z] [0-9] [\$@.&+;] [!*\(\),] (?:%[0-9a-f][0-9a-f]))+"
User mentions	mention_pattern = r"(?:@[w_]+)"
Punctuation	r"[^{\w}\s]"
Uppercase	r"[^{\w}\s]"
Numbers	`\d+
White spaces	r'\s\s+'
HTML	html_pattern = r'<[^>]+>'
Original data	RT @dohanews: @jakethdyer @cocobelladoodle As of October 4, new guidelines will allow travelers from Qatar to enter England with no quarantine requirements #COVID-19 https://t.co/4wWFY7iDZP
Normalized data	RT #MENTION#: #MENTION# #MENTION# As of October 4, new guidelines will allow travelers from Qatar to enter England with no quarantine requirements #EMOJI# #HASHTAG# #URL

results of both NLP text preprocessing techniques – lemmatization and stemming which helps in morphological analysis of words being used as represented in Table 4.

Table 4 Text preprocessing techniques – lemmatization and stemming

Technique	Description	Example
Lemmatization	Lemmatization uses vocabulary and morphological analysis of word and try to remove inflectional endings, thereby returning words to their dictionary form [1]	Leaves } Leaf Leafs }
Stemming	Stemming uses removal of derivational suffixes as well as inflections (i.e., suffixes that change the form of words and their grammatical functions) so that word variants can be combined into the same roots or stems [1]	Leaves → Leav Leafs → Leaf

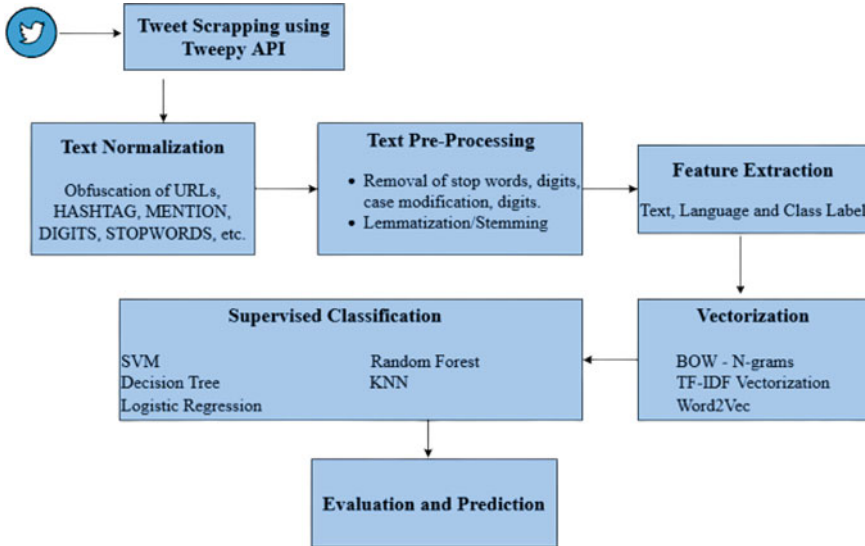


Fig. 1 Tweet extraction, classifier, and prediction model

3.4 Feature Extraction

Prior to classification, the normalized and preprocessed tweets need to be vectorized. There are various techniques used for text vectorization. We have considered TF-IDF, bag-of-words (both implemented using NLTK), and word embedding techniques, namely, Word2Vec. (implemented using Genism). BOW – countvectorizer simply provides term count and treats all inputted words the same providing low semantic information. TF-IDF – Term Frequency Inverse Document Frequency method represents number of times a term appears in a document to inverse document frequency of the term across a set of other documents, providing higher semantic information comparatively. Word2Vec is one of the efficient word embedding methods providing more semantic and syntactic similarity in correlation with other terms in a document. Word2Vec is an amalgamation of two techniques, namely, CBOW and skip gram [13, 14]. We have applied lemmatization and stemming prior to vectorization and compared the results individually.

4 Classification

Supervised learning is a task-driven approach composed of labeled data to train the machine learning algorithm [11]. Supervised learning can be categorized into classification and regression. The former uses an algorithm to recognize and classify specific objects into the categories being labeled and trained. The latter uses an

algorithm to draw and understand correlation between dependent and independent variables. For our experimentation, we have considered only one feature, text for each tweet, and we have applied classification algorithms using scikit-learn [5] libraries of Python. Most popularly used supervised classification algorithms are support-vector machine, decision tree, k-nearest neighbors, random forest, and logistic regression [7, 11–13].

5 Evaluation and Result

Evaluation is performed using binary classification; training dataset were labeled as “real” or “fake” represented as 1 and 0, respectively. To evaluate the performance of supervised classification, we have considered a confusion matrix. The model for confusion matrix in represented in Table 5, which can be elaborated as: true positive (TP), when predicted fake texts are actually annotated as fake; true negative (TN), when predicted true (real) text are actually annotated as true (real); false negative (FN), when predicted true(real) text are actually annotated as fake text; and false positive (FP), when predicted fake text are actually annotated as true (real) [15].

Table 5 Model for confusion matrix

		<i>Predicted class</i>	
		<i>Class = yes</i>	<i>Class = no</i>
<i>Actual class</i>	<i>Class = yes</i>	True positive	False negative
	<i>Class = no</i>	False positive	True negative

Accuracy, precision, recall, and F1 score is considered as the evaluation metrics for result analysis. The evaluation metrics are implemented for TF-IDF vectorization, bag-of-words, and word embedding techniques, namely, Word2Vec as described in Table 6. Accuracy, precision, recall, and F1 score can be calculated using the follow formulae as listed below:

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{True Negative} + \text{False Positive} + \text{False Negative}} \tag{1}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \tag{2}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \tag{3}$$

$$\text{F1 score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

The results are presented with respect to each supervised classification model as shown in Figs. 2, 3, 4, 5, and 6, after applying text preprocessing techniques lemmatization, and stemming followed different vectorization techniques.

5.1 Logistic Regression

For logistic regression classification, BOW after stemming performed well compared to other TF-IDF and Word2Vec with precision of 0.781, recall 0.758, F1 score 0.71, and accuracy 77%. From Fig. 2, we can observe that stemming outperformed lemmatization.

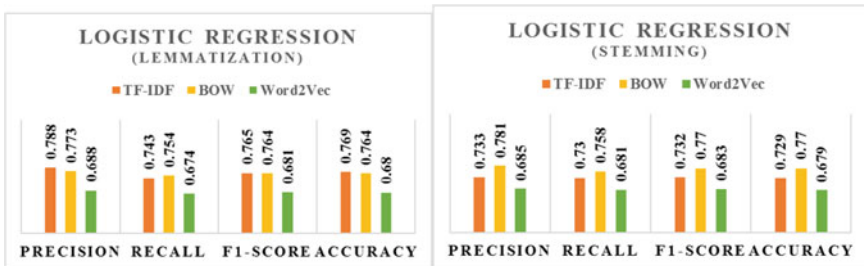


Fig. 2 Comparing vectorization techniques for logistic regression

5.2 Random Forest

For random forest classification, BOW after stemming performed well compared to other TF-IDF and Word2Vec with precision of 0.798, recall 0.744, F1-score 0.771, and accuracy 77.6%. From Fig. 3, we can observe that stemming outperformed lemmatization.

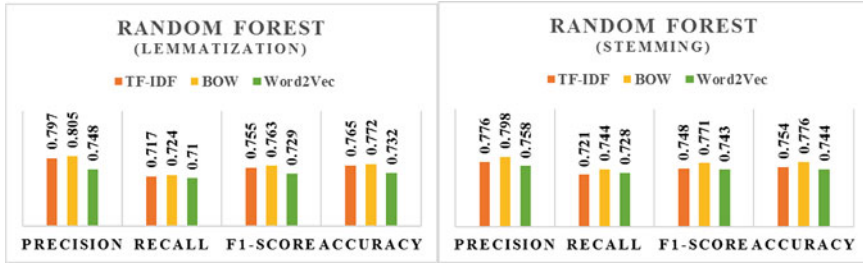


Fig. 3 Comparing vectorization techniques for random forest

5.3 Support-Vector Machine

For support-vector machine classification TF-IDF after lemmatization performed well compared to other BOW and Word2Vec with precision of 0.799, recall 0.732, F1 score 0.765, and accuracy 77.1%. From Fig. 4, we can observe that lemmatization outperformed stemming.

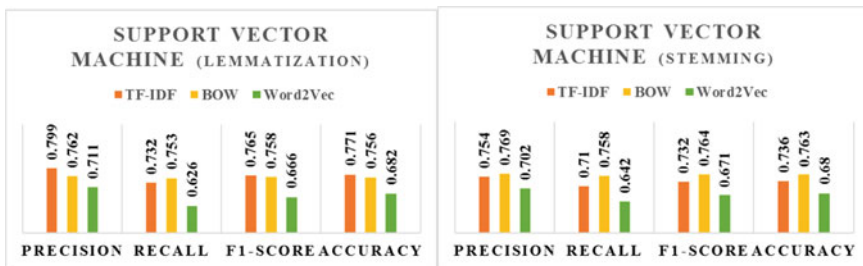


Fig. 4 Comparing vectorization techniques for support-vector machine

5.4 Decision Tree

For decision tree classification, BOW after lemmatization performed well compared to other TF-IDF and Word2Vec with precision of 0.764, recall 0.777, F1 score 0.771, and accuracy 77.6%. From Fig. 5, we can observe that lemmatization outperformed stemming.

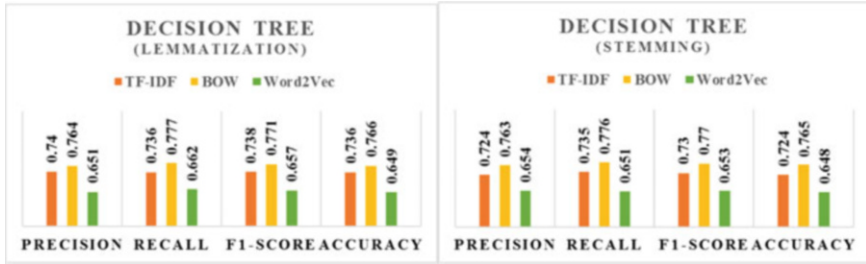


Fig. 5 Comparing vectorization techniques for decision tree

5.5 K-Nearest Neighbor

For k-nearest neighbor classification, Word2Vec after stemming performed well compared to other TF-IDF and BOW with precision of 0.733, recall 0.724, F1 score 0.729, and accuracy 72.6%. From Fig. 6, we can observe that stemming outperformed lemmatization.

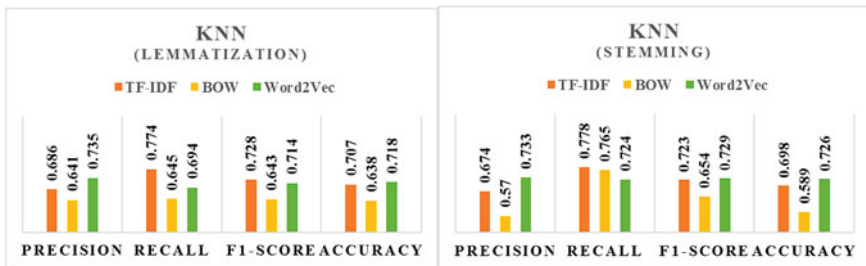


Fig. 6 Comparing vectorization techniques for KNN

Table 6 Comparing accuracy for vectorization models

Classification model – random forest	Accuracy		
	TF-IDF (%)	BOW (%)	Word2Vec (%)
Lemmatization	76.5	77.2	73.2
Stemming	75.4	77.6	74.4

After comparing the accuracy for all supervised classifiers with different pre-processing techniques, namely, lemmatization and stemming, and vectorization methods, it can be observed that the random forest classification model delivered the best results with stemming and BOW. KNN performed the worst for each feature

extraction method after stemming technique. Table 7 summarizes confusion matrix for random forest with stemming and BOW.

Table 7 Confusion matrix for random forest

		<i>Predicted values</i>		
		<i>Positive</i>	<i>Negative</i>	<i>Totals</i>
<i>Actual values</i>	<i>Positive</i>	1315 (TP)	453 (FN)	$P = (TP + FN) = \text{Actual total positives}$
	<i>Negative</i>	333 (FP)	1399 (TN)	$N = (FP + TN) = \text{Actual total negatives}$
<i>Totals</i>		<i>Predicted total positives</i>	<i>Predicted total negatives</i>	

6 Conclusion

The paper describes a model for comparative evaluation of varied supervised classifiers using precision, recall, F1 score, and confusion matrix. The model also covers different preprocessing techniques, namely, lemmatization and stemming, and compares further with different vectorization models, namely, BOW, TF-IDF, and Word2Vec. The results shown from Figs. 2, 3, 4, 5, and 6 shows that BOW with stemming worked well for logistic regression and random forest. While TF-IDF with lemmatization proved better for SVM and decision tree. Word2Vec with stemming showed good results for KNN. Based on the results shown in Table 6, it is found that random forest model outperformed SVM, KNN, decision tree, and logistic regression with precision of 0.798, recall 0.744, F1 score 0.771, and accuracy 77.6%. Table 7 summarizes the performance ratio for fake news prediction achieved by random forest classifier. In future, one can experiment with more preprocessing techniques apart from TF-IDF, BOW, and Word2Vec to improve the accuracy of the model, for fake news identification and prediction for social media platforms like Twitter.

References

1. Balakrishnan, Vimala and Ethel Lloyd-Yemoh. "Stemming and lemmatization: A comparison of retrieval performances." (2014), <https://doi.org/10.7763/LNSE.2014.V2.134>
2. Elisabetta Fersini, Justin Armanini, and Michael D’Intorni, "Profiling fake news spreaders: stylometry, personality, emotions and embeddings" in Notebook for PAN CLEF 2020.

3. Francisco Rangel; Paolo Rosso; Bilal Ghanem; Anastasia Giachanou, "Profiling Fake News Spreaders on Twitter" dataset for PAN 2020 on February 29, 2020. <https://doi.org/10.5281/zenodo.4039435>
4. <https://brandequity.economictimes.indiatimes.com/news/research/misinformation-on-facebook-gets-way-more-engagement-than-news-study/86000388>
5. <https://scikit-learn.org/stable/>
6. <https://www.firstpost.com/india/how-heavy-internet-usage-and-poor-digital-literacy-made-india-worlds-top-source-of-misinformation-on-covid-19-9966471.html>
7. Iqbal H. Sarker, "Machine Learning: Algorithms, Real-World Applications and Research Directions", *SN Computer Science (2021) 2:160*, <https://doi.org/10.1007/s42979-021-00592-x>
8. María S. Espinosa, Roberto Centeno, and Álvaro Rodrigo, "Analyzing User Profiles for Detection of Fake News Spreaders on Twitter" Notebook for PAN at CLEF 2020.
9. Matteo Cardaioli, Stefano Ceconello, Mauro Conti, Luca Pajola, and Federico Turrin, "Fake News Spreaders Profiling Through Behavioural Analysis", Notebook for PAN at CLEF 2020
10. Minh Hao Nguyen, Jonathan Gruber, Jaelle Fuchs, Will Marler, Amanda Hunsaker, and Eszter Hargittai, "Changes in Digital Communication During the COVID-19 Global Pandemic: Implications for Digital Inequality and Future Research", *2020 SAGE Journals*<https://doi.org/10.1177/2056305120948255>
11. Mohammed M, Khan MB, Bashier Mohammed BE. Machine learning: algorithms and applications. CRC Press; 2016
12. R. Joshi and R. Tekchandani, "Comparative analysis of Twitter data using supervised classifiers," *2016 International Conference on Inventive Computation Technologies (ICICT)*, 2016, pp. 1-6, <https://doi.org/10.1109/INVENTIVE.2016.7830089>
13. S. Chowdhury and M. P. Schoen, "Research Paper Classification using Supervised Machine Learning Techniques," *2020 Intermountain Engineering, Technology and Computing (IETC)*, 2020, pp. 1-6, [10.1109/IETC47856.2020.9249211](https://doi.org/10.1109/IETC47856.2020.9249211)
14. Sharma, Aditya & Daniels, Alex. (2020). Tweets Sentiment Analysis via Word Embeddings and Machine Learning Techniques.
15. Shu, Kai & Sliva, Amy & Wang, Suhang & Tang, Jiliang & Liu, Huan. (2017). Fake News Detection on Social Media: A Data Mining Perspective. ACM SIGKDD Explorations Newsletter. 19. <https://doi.org/10.1145/3137597.3137600>
16. Vasu Agarwal, H.Parveen Sultana,Srijan Malhotra, Amitrajit Sarkar, "Analysis of Classifiers for Fake News Detection", *INTERNATIONAL CONFERENCE ON RECENT TRENDS IN ADVANCED COMPUTING 2019*. Procedia Computer Science 165 (2019) 377–383 <https://doi.org/10.1016/j.procs.2020.01.035>

Analysis of Pest Recognition Using Lightweight CNN



C. Nandhini and M. Brindha

Keywords Deep learning · Pest classification · DenseNet · EfficientNet

1 Introduction

Agriculture is essential to humanity's survival and economic well-being since it feeds an ever-increasing population. On the other hand, crop output seems to be dropping due to climate change and the scarcity of farming land. The vast agricultural land of a country decides its social and economic prosperity. Agriculture is the primary source of employment for the majority of the population. Most of the business of a country relies on agricultural products. One of the most significant factors affecting agricultural crop productivity is the plant diseases, pest and insects. Pests are tiny, destructive creatures that attack crops and hamper agricultural yield by destroying them. Insects are also small creatures, but some of the insects are beneficial as they help in pollination of flowers and pest control, while others are considered as pests. The lack of farmer knowledge in identifying specific pathogen attacks complicates the control of the diseases. A study on accessing the farmer's knowledge about insects and plants disease reveals that only 16% of farmers can recognize insects. Providing farmers with suitable training can help them better identify and control crop infections. However, the lack of substantial expertise makes training impractical for huge farmers. Precise identification of insect pests makes it easier to take timely preventative measures to avoid financial losses. The availability of automatic system for identifying pests more accurately boosts the overall crop productivity and quality.

Artificial intelligence (AI) has been used in various applications like medical, agriculture, computer vision etc., to benefit society. Deep learning has benefited from recent advances in memory and computation capability that helps in analysing

C. Nandhini (✉) · M. Brindha
National Institute of Technology, Tiruchirappalli, India
e-mail: cn.nandhini@gmail.com; brindham@nitt.edu

massive amounts of data. The growth of deep learning with convolutional neural networks (CNNs) has made a revolution in computer vision tasks such as classification, detection, segmentation, etc. The deep neural network architectures like LeNet, AlexNet, VGG and GoogleNet obtain more extraordinary performance on ImageNet datasets compared to traditional methods, inaugurating the success of deep learning for computer vision. AI applications for agriculture include soil categorization and crop selection based on water availability, predicting crop disease and thus increasing crop production, continuous monitoring for yield forecasting etc. The main contribution in this chapter can be summarized as follows:

- To develop a framework for pest recognition based on supervised deep learning architecture
- To investigate the feature extraction, using different deep learning architectures
- To present a comparative analysis of each deep learning architecture in terms of accuracy, precision and recall

2 Related Work

Damage to crops occurs due to natural calamities such as inadequate or more rainfall, flood, and due to less nutrient soil, the crops may be affected by several disease and pests. So, spotting the pests in the agriculture field at the earlier stage is an essential task for the farmers, to take adaptive protective measures. Recent years, many researches based on deep learning techniques are proposed, focussing on agricultural tasks like weed identification, leaf disease identification, pest detection, crop type classification, yield prediction and fruits counting. Literature [1–4] shows that deep learning techniques are used to analyse diseases in rice, wheat, maize, tomato, apple, tea, pear, peach, grapevine etc. Mohanty et al.[1] trained AlexNet and GoogleNet based disease detection for 26 types of disease on various crops on PlantVillage dataset. Shijie et al. [2] use a transfer learning model based on VGG16 to detect tomato pests and diseases achieving an average accuracy. DeChant et al. [5] trained full-scaled images by dividing a single image into many smaller images. Liu et al. [3] use a combination of AlexNet and GoogLeNet for apple disease identification. By analysing the past work, deep learning algorithms have increased the performance of pest categorization considerably. The SACNN proposed by Zeng et al. [6] uses self-attention network to learn key features which increase the precision in identifying the crop disease. Verma et al. [4] implemented capsule networks for classification of potato diseases using the field images. Duong Linh et al. [7] use pre-trained EfficientNet and Mixnet for automatic fruit recognition. Chen et al. 2020[8] use the pre-trained VGGNet and Inception module to identify plant leaf disease in rice and maize crops. Based on the observation, better performance can be obtained using pre-trained CNN, and hence the proposed work analyses the different lightweight model for pest identification.

3 Deep Neural Networks to Recognize Pests

This section presents the related technologies as well as the proposed architecture for developing an expert system to recognize pests in a systematic manner.

3.1 Convolutional Neural Network

Convolutional neural networks (CNNs) were designed exclusively for image processing, and they attempt to capture spatial relationship while minimizing the number of parameters. CNN comprises CNN layer that acts like a filter and used to extract features, pooling layers to reduce the width and height and fully connected layer to combine all the features and create the output classification labels. Nowadays, the dimension of feature maps and the layer depths are increased to extract the fine-grained features.

DenseNet Inspired by the success of DenseNet[9], the proposed work uses DenseNet architecture to efficiently learn both the high- and low-level spatial features. DenseNet allows layers to concatenate feature maps from the preceding layers. This way, feature maps of all preceding layers are used as input. It can effectively alleviate the difficulty of training very deep networks and strongly enhance the performance by reusing features from the initial layers in all subsequent layers.

EfficientNet The EfficientNet[10] model is developed by Google AI, which is based on simple and extremely effective compound scaling algorithms. In general, EfficientNet models outperform existing CNNs like AlexNet[11], GoogleNet[12] and MobileNetV2[13] in terms of accuracy and efficiency. To the best of the authors' knowledge, no other work has used EfficientNet for transfer learning in the context of Pest classification. EfficientNet improves the CNN's performance by scaling the width, height and the number of channels using an appropriate scaling factor. EfficientNet uses the compound scaling method that reduces the feature map size while scaling up the width in subsequent layers that helps in capturing detailed information.

MobileNetV3 Mobile models have become increasingly efficient as a result of the use of more efficient building components. MobileNetV1[14] included depth-wise separable convolutions, which isolate the feature generation method from the spatial filtering. The linear bottleneck and inverted residual structure were introduced in MobileNetV2[13], which preserves a compact representation at the input and output while internally extending to a higher dimensional feature space to boost the expressiveness of nonlinear per channel operations. There are two MobileNetV3 models: MobileNetV3-Large and MobileNetV3-Small. These models are designed

for scenarios with high and low resource usage and are chosen based on platform-aware Neural Architecture Search (NAS) and NetAdapt for network integration.

3.2 Dataset

This chapter uses a large-scale open-source dataset named IP102[15] for insect pest recognition. Specifically, it contains around 75,000 images belonging to 102 categories. The dataset is split into three independent sets: the training set, the validation set and the test set. The training set accounts for 60% of the total dataset (45,095 images), the validation set accounts for 10% (7508 images) and the test set accounts for 30% (22,619 images). Figure 1 shows the sample images in the IP102 dataset. The images under this dataset have hierarchical structure; the top level is based on economic crop or field crop. Paddy, wheat and rice belong to field crops, while lemon belongs to economic crops. Next level, the pests that are affecting a particular crop are grouped together.

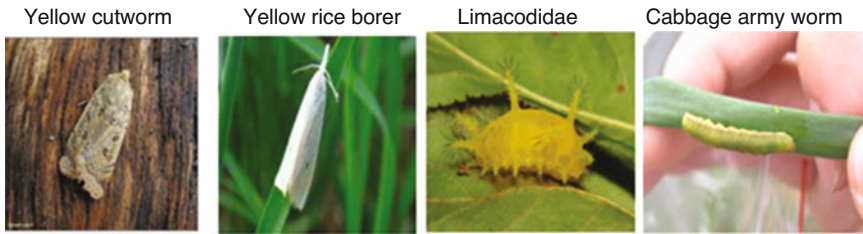


Fig. 1 Sample images from IP102 dataset

3.3 System Architecture

The proposed system architecture is depicted in Fig. 2, and it consists of the learning and deployment phases.

Pre-processing The images in the training set are transformed to increase the model's generalization capability. This transformation helps to recognize target objects of different sizes, contrast, varied angles etc. The images are resized to width and height equivalent to 224. Then, images are random rotated by 10-degree, horizontal flip. The mean and standard deviation of pixels from training images are calculated and used for the standardization of the images.

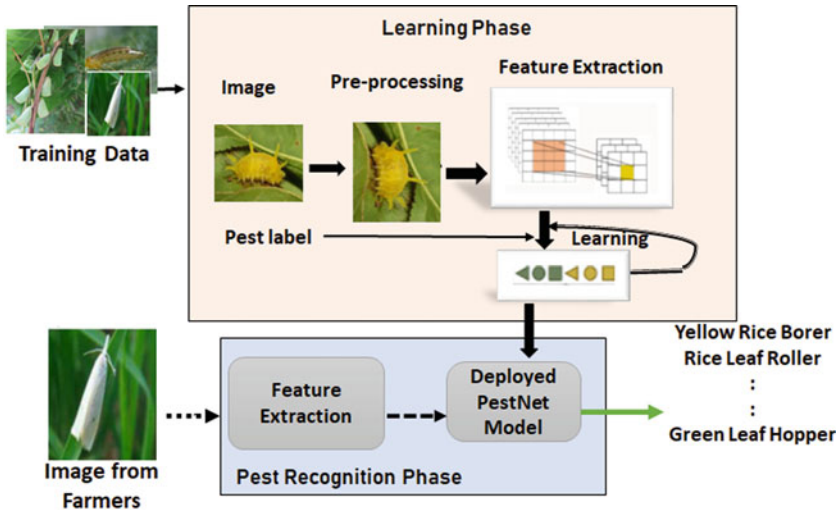


Fig. 2 Proposed pest recognition system architecture

Feature Extraction Transfer learning has been developed as an efficient method for exporting knowledge from a knowledgeable source domain to the target domain. The feature extraction phase uses the pre-trained weights learned from ImageNet datasets. The output of this phase is feature vectors, which along with the pest labels are given to the learning phase. As the dataset is large, fine-tuning is done by freezing this layers, and training happens only in the learning phase.

Learning The learning module is used for classification purpose. It usually comprises fully connected layers. The features extracted from previous layer are given to the fully connected layers, in which all of the neurons in the previous layer are totally connected. The output of the learner is the model that contains the weights and biases that are going to be deployed in the cloud or mobile devices and are used to classify any pest images from the agriculture field.

Deployment Phase Before deployment, quantization can be done to reduce the model file size by 50% and accuracy will not be affected. The final quantized model can be deployed in the cloud server or used as a mobile app to predict the pest classes. If the model is deployed in cloud, the size of the model is not a significant issue, and however the inference speed is affected by the user network capacity.

4 Experiments

The experiments are conducted using Intel Core i7-9700K CPU with the NVIDIA GeForce RTX 2080Ti with RAM of 16 GB with 3.60 GHz processor and the implementation are done using PyTorch. The state-of-the-art models such as

EfficientNet-B0, DenseNet-121, MobileNetV3-Small and MobileNetV3-Large are explored for feature extraction. The weights for the CNN in feature extraction layers use pre-trained weights, while the fully connected layers in the learning phase are initialized with random weights.

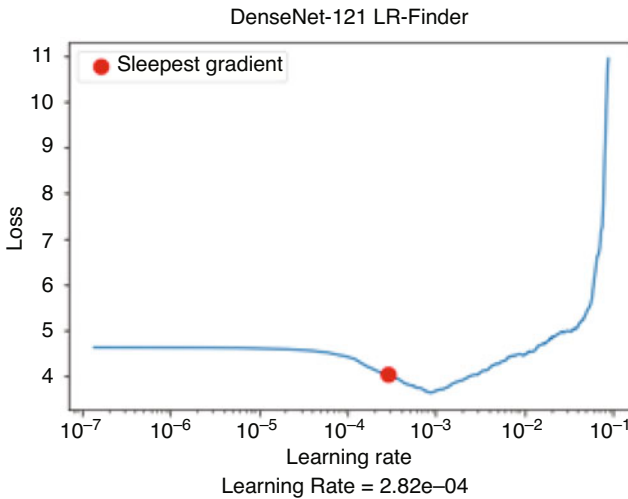


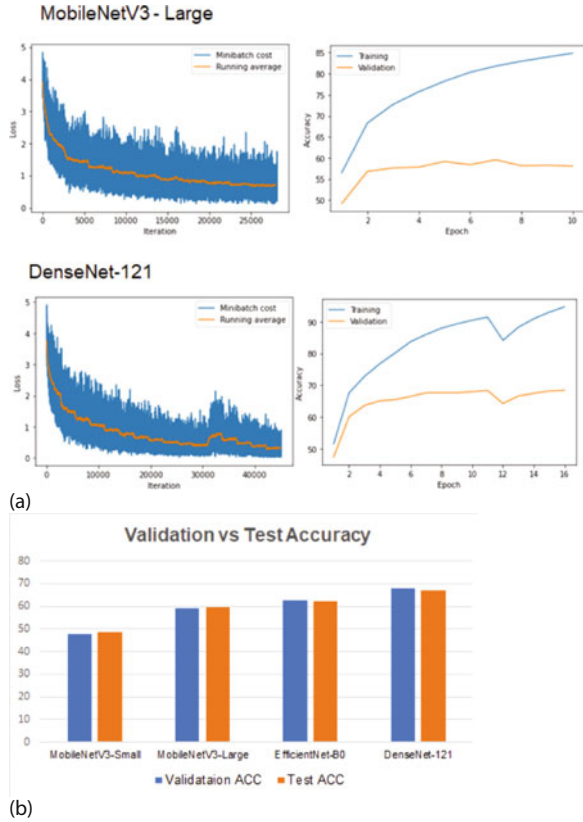
Fig. 3 Learning rate optimization using cyclic learning rate

The hyperparameters used are as follows: dropout rate of 0.25 and learning rate are decided using cyclic learning rate [16] with a batch size of 16. Figure 3 shows the learning rate graph obtained while optimizing the learning rate for DenseNet-121 based model. The range for learning rate is set as $(2.8 \times 10^{-4}, 2.8 \times 10^{-3})$ for DenseNet-121 based classifier. Similarly, the optimal learning rate is identified for all the models under consideration. The network is trained for 100 epochs, and early stopping is used to stop the training when there is no loss convergence for 10 epochs. Figure 4a shows the training and validation loss graph during training of MobileNetV3-Large and DenseNet-121 based models. The similar pattern in both validation and training loss confirms that there is no overfitting during learning. As the transfer learning is used for feature extraction, the network is trained within 10 to 20 epochs depending on the network architecture.

4.1 Performance Evaluation

This section provides insights into performance metrics that are commonly adopted for evaluating the performance of the pest recognition model. In the test set, for all N images, the corresponding class is denoted by actual classes $AC = \{P_1, P_2, P_3, \dots, P_N\}$

Fig. 4 (a) Loss vs epochs and (b) test vs validation accuracy



and the predicted set of classes is given as $PC = \{C_1, C_2, C_3 \dots C_N\}$. Table 1 represents the performance evaluation on the test set. The various performance metrics are listed below:

Accuracy

The ratio of successfully categorized pests images to the total number of pest classes in the test set.

$$Accuracy = \frac{\sum_{i=1}^N AC_i \cap PC_i}{\sum_{i=1}^N |AC_i|} \times 100\% \tag{1}$$

From Fig. 4b, it is clear that test and validation accuracy are approximately equivalent, which indicates that the network is able to generalize very well on the unseen data. DenseNet-121 based model has the highest accuracy of all the models.

Precision

Precision represents the percentage of the correct predictions made by the model out of the total predictions.

$$Precision = \frac{\sum_{i=1}^N AC_i \cap PC_i}{|PC_i|} \quad (2)$$

Recall

The recall ratio represents the percentage of correct predictions out of the actual labels detected by the model.

$$Recall = \frac{\sum_{i=1}^N AC_i \cap PC_i}{|AC_i|} \quad (3)$$

F1 Score

The recall ratio represents the percentage of correct predictions out of the actual labels detected by the model.

$$F1Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (4)$$

Table 1 shows the precision, recall, F1 score and the number of parameters for learning head in terms of thousands. MobileNetV3-Small is having almost one-third of DenseNet-121 parameters. From Table 1, it is observed that DenseNet-121 based model has given the highest accuracy values compared to other models. Reusing the features from the previous layers is considered as an effective approach for the pest classification. MobileNetV3-Small is the architecture with the lowest number of parameters but giving less accuracy.

Table 1 Performance evaluation on test set

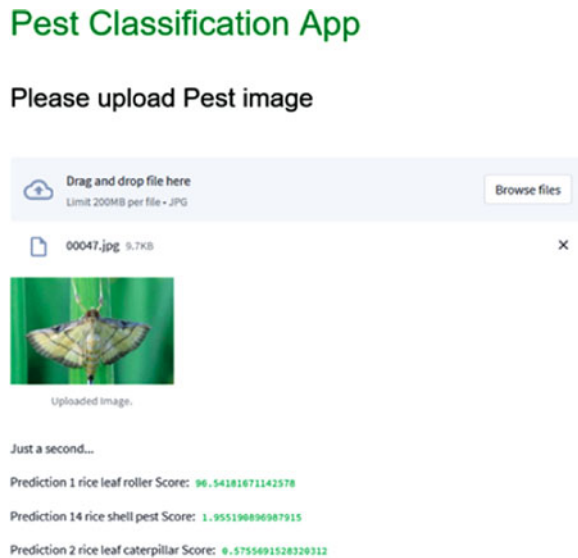
Model	Accuracy (%)	Precision	Recall	F1 score	#Params(Th) learner
MobileNetV3-Small	48.73	41.63	32.08	32.99	58.85
MobileNetV3-Large	59.80	54.75	48.69	49.10	98.02
EfficientNet-B0	62.48	59.12	50.01	51.05	130.66
DenseNet-121	67.05	56.69	57.58	56.02	104.55

4.2 Model Quantization and Deployment

Table 2 Effect of quantization on model size

Model	Model size(MB) Model size(MB)	Quantized model size(MB)	Quantized model accuracy (%)
MobileNetV3-Small	10.54	5.52	50.14
MobileNetV3-Large	22.53	14.70	61.37
EfficientNet-B0	21.97	17.74	64.71
DenseNet-121	32.94	29.55	66.60

Fig. 5 Web application deployment



Model quantization is a technique for reducing inference time and memory requirements. Model quantization is the process of storing network weight tensors with integer precision rather than floating point precision. The quantized model performs all operations on integers, and the model size is reduced by approximately four times, resulting in a reduction in memory. In comparison to floating point,

integer computation is faster, resulting in faster inference. The quantized model is well suitable for mobile and web based applications. In the proposed approach, the model is trained in floating point and then dynamically quantized using PyTorch. The fully connected layers after the feature extraction phase are quantized. The quantized model was validated to ensure that the network's performance would not be compromised. Table 2 shows the size of the model, its quantized version size and the accuracy using quantized model. The quantized version of MobileNet based models has reduced its size nearly by half. DenseNet-121 based model needs more memory than other models. So, it is suitable for deploying in cloud based web servers. EfficientNet-B0 and MobileNetV3-Large have a similar number of parameters and the model file sizes are also the same. EfficientNet-B0 has the similar configuration as MobileNetV3-Large but has a higher accuracy of approximately 62.5% accuracy and hence can be used for mobile devices. To provide with real-time implementation, the quantized model is deployed as a web application using Streamlit library of PyTorch. Figure 5 shows the deployment of web application. The quantized model is able to identify the rice leaf roller pest with a confidence score of 96.5, which demonstrates the prediction capability of the deployed model.

4.3 Comparison Analysis

The performance is compared with several state-of-the-art models, and the results are shown in Fig. 6 which demonstrates the effectiveness of the proposed approach. Most of the accuracy and trainable parameters for the state-of-the-art models for comparison are taken from [17]. As observed from Fig. 6, the proposed transfer learning based approach for feature extraction provides the significant performance improvement with a less number of trainable parameters.

5 Conclusion

The proposed work analyses effective solutions for pest recognition using well-defined and evolved deep neural network classification techniques. The performance of the approach has been validated using the IP102 dataset. The experimental results show that using DenseNet and EfficientNet based models greatly enhances overall prediction accuracy. Even the simplest network architectures, such as EfficientNet-B0 and DenseNet-121, achieve good prediction performance. This chapter tries to find the compact model that can be used on as a stand-alone mobile application that can help farmers to identify the pest at the earlier stage and thus improves the overall crop production. The final model to be considered for the mobile app deployment was a EfficientNet-B0 fine-tuned model, and DenseNet-121 based model can be

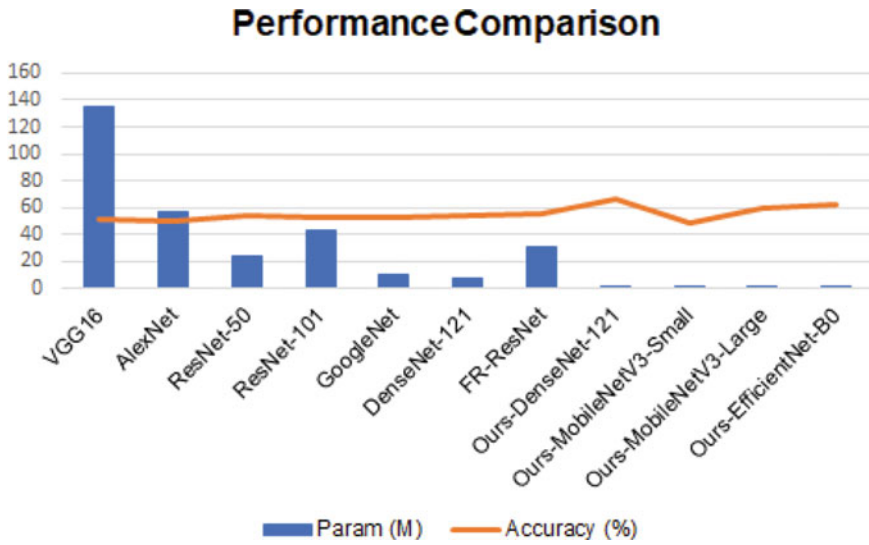


Fig. 6 Performance with the state-of-the-art

considered for cloud server deployment. For the future work, the pest recognition and pest detection for particular crop will be analysed.

References

1. Sharada P Mohanty, David P Hughes, and Marcel Salathé. Using deep learning for image-based plant disease detection. *Frontiers in plant science*, 7:1419, 2016.
2. Jia Shijie, Jia Peiyi, Hu Siping, et al. Automatic detection of tomato diseases and pests based on leaf images. In *2017 Chinese automation congress (CAC)*, pages 2537–2510. IEEE, 2017.
3. Bin Liu, Yun Zhang, DongJian He, and Yuxiang Li. Identification of apple leaf diseases based on deep convolutional neural networks. *Symmetry*, 10(1):11, 2018.
4. Shradha Verma, Anuradha Chug, and Amit Prakash Singh. Exploring capsule networks for disease classification in plants. *Journal of Statistics and Management Systems*, 23(2):307–315, 2020.
5. Chad DeChant, Tyr Wiesner-Hanks, Siyuan Chen, Ethan L Stewart, Jason Yosinski, Michael A Gore, Rebecca J Nelson, and Hod Lipson. Automated identification of northern leaf blight-infected maize plants from field imagery using deep learning. *Phytopathology*, 107(11):1426–1432, 2017.
6. Weihui Zeng and Miao Li. Crop leaf disease recognition based on self-attention convolutional neural network. *Computers and Electronics in Agriculture*, 172:105341, 2020.
7. Linh T Duong, Phuong T Nguyen, Claudio Di Sipio, and Davide Di Ruscio. Automated fruit recognition using efficientnet and mixnet. *Computers and Electronics in Agriculture*, 171:105326, 2020.
8. Junde Chen, Jinxiu Chen, Defu Zhang, Yuandong Sun, and Yaser Ahangari Nanehkaran. Using deep transfer learning for image-based plant disease identification. *Computers and Electronics in Agriculture*, 173:105393, 2020.

9. Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.
10. Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.
11. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.
12. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.
13. Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.
14. Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.
15. Xiaoping Wu, Chi Zhan, Yu-Kun Lai, Ming-Ming Cheng, and Jufeng Yang. Ip102: A large-scale benchmark dataset for insect pest recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8787–8796, 2019.
16. Leslie N Smith. Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE, 2017.
17. Fuji Ren, Wenjie Liu, and Guoqing Wu. Feature reuse residual networks for insect pest recognition. *IEEE Access*, 7:122758–122768, 2019.

Early Prediction of Alzheimer's Disease Using Ensemble Learning Models



Divjot Singh and Ashutosh Mishra

Keywords Alzheimer's disease (AD) · Machine learning (ML) · Gradient boosting machine (GBM)

1 Introduction

New diseases are discovered every year, and their cures are made in parallel. From the development of first vaccine to the cure made for corona virus, we have come a long way successfully. This doesn't mean that every disease present in current time can be detected and treated permanently. There are many diseases for which there is no cure available. AD is a neurodegenerative disease which has affected millions of people on the planet, and yet there is no permanent cure for it. We can try to detect AD at an early stage so that we can reduce the progression rate of the patient and patient can live a longer life. Various methods like PET scan, MRIs, blood sampling, behavioral study, sleeping patterns, and other MCI-related features are observed to detect whether a person has Alzheimer's disease or not. Collecting all this heterogeneous data and combining it manually can take ages to get to the result accurately. Machine learning and its modern techniques have proved to be effective in predicting this answer with high precision. The features that help us in predicting the result more accurately are used with different classification models. So, the primary goal of our study is to use machine learning and its techniques to detect AD at an early stage.

D. Singh (✉) · A. Mishra

Computer Science and Engineering Department, Thapar Institute of Engineering and Technology, Patiala, India

e-mail: ashutosh.mishra@thapar.edu

2 Literature Survey

Testing the brain and obtaining an MRI don't mean that we can be sure that a person has Alzheimer's or not. The better way to be sure is if we see a pattern in a number of images of the same kind. Going through all those thousands and millions of images is very difficult and very time-consuming. So, there should be a way by which we can automatically classify if they lie in the same category or not such that if the patient is Alzheimer's positive or not. This is where machine learning and its modern techniques come into the picture. Machine learning (ML) is the study of computer algorithms that improve automatically through experience. Machine learning algorithms build a model based on sample data, known as training data, to make predictions or decisions without being explicitly programmed to do so. To understand the work done in this field, an in-depth study is done in Table 1 as literature survey.

Table 1 Literature survey

Year	Author	Objective	Method	Outcome	Limitations
2021	Ghoraani et al.	Detection of AD and MCI patients by observing gait features using SVM	SVM	It proves the potential of ML and gait assessments together for detection at an early stage Accuracy: 78%	Accuracy is lower than the other using the cognitive score Due to more focus on gait features, it is less trustworthy
2020	Ammar et al.	Detecting AD at an early stage by studying language-related features	SVM	AD affects linguistic abilities first, which is proved to help in early detection Accuracy: 91%	Language impairment alone doesn't confirm Alzheimer's disease
2020	Bringas et al.	Collection of sensor data for identifying the stage of AD using deep learning	CNN model	It can classify all the three stages successfully Accuracy: 90.91%	CNN fails if dataset features vary Patients had to carry smartphones everywhere Data sequences are too long

(continued)

Table 1 (continued)

Year	Author	Objective	Method	Outcome	Limitations
2020	Liu et al.	To combine hippocampal segmentation results using multitask deep CNN and DenseNet for determining the class of person	Multitask deep learning 3D densely connected CNN (DenseNet)	Outperforms the normal CNN methods Accuracy: 88.9% (AD vs. NC) Accuracy: 76.2% (MCI vs. NC)	Data loss is in abundance Less accuracy Characterization of learned features relevant to disease is not optimal
2020	Haider et al.	Prediction of AD using paralinguistic acoustic features from Dementia Bank Pitts spontaneous speech dataset	SVM KNN LDA Decision tree	The acoustic features of spontaneous speech are important for AD detection Accuracy: 78.7%	Clustering the original acoustic features and then extracting new features based on these clusters, one loses the ability to access the contribution of original features to the model’s predictions
2020	Bi et al.	Detection of AD using MRI images and unsupervised deep learning	Unsupervised CNN	One slice Accuracy: 95.52% (AD vs. MCI) Accuracy: 90.6% (MCI vs. NC) TOP data Accuracy: 97.01% (AD vs. MCI) Accuracy: 92.6% (MCI vs. NC)	With a greater no. of samples, the accuracy dropped by 2% The chances of dragging outliers can cause trouble when size and density vary
2020	Janghel and Rathore	To improve the limitations of ML models using preprocessing and then classifying data using CNN	SVM K means Decision tree CNN	FMRI accuracy: 99.95% PET accuracy: 73.96%	Time-consuming PET is expensive, and diabetic patients need more caution while scanning

(continued)

Table 1 (continued)

Year	Author	Objective	Method	Outcome	Limitations
2020	Li et al.	Spatial and time-varying information obtained from 4D fMRI data to be processed using the 4D DL model	Deep learning model (C3D – LSTM)	4D fMRI preserves the spatial and temporal information more significantly LSTM accuracy: 92.11%	4D implementation is complex and is more prone to information loss
2020	Lopez-Martina et al.	Developing a novel DL method to detect AD using magnetoencephalography	2D CNN	F1 score = 91%	Ignored the MSME and demographic variables
2020	Uysal and Oztur	Volumetric reduction in the hippocampus is an important indicator for predicting AD at an early stage	Logistic regression K-nearest SVM DT RF Gaussian naive	The left hippocampus related to analytical thinking skills can make more accurate estimates KNN: 80% GNB: 82%	Ignored right hippocampus while calculating GNB Primary focus on hippocampus only
2020	Kumari et al.	Identifying normal and AD brain aging using ML techniques	Fuzzy clustering CNN	It outperforms the KNN model by a vast majority Accuracy: 90.25%	Fuzzy clustering requires more iterations for refining a result thus time-consuming
2020	Padole et al. (Member, IEEE)	To outperform traditional ML models using coarse graph and GCNN for AD prediction	G-CNN	Outperforms state-of-the-art methods both in general coarsening and as pooling operator Accuracy: 92%	The value of the scaling parameter is not optimal because it is selected empirically

(continued)

Table 1 (continued)

Year	Author	Objective	Method	Outcome	Limitations
2020	El- Sappagh et al.	To apply multimodal multitask modeling techniques to classify AD progression	Ensembled CNN BiLSTM	It investigates the joint prediction of multiple regression and multiclass classification variables from multiple longitudinal and BG data Accuracy: 92.62%	BiLSTM is costly Training data accurately is a bit hard
2020	GU et al.	To develop a DL model for AD prediction using clinical and MRI data	Auto-encoders and improved deep learning algorithm	SD reduced by 45% forecasting the model to be more reliable and efficient Accuracy: 90%+	Overfitting and underfitting prediction are unclear Auto-encoders can ignore valuable data by pruning
2020	Liu et al.	To detect AD using proton magnetic resonance spectrography	Gradient boosting decision tree	Using H-MRS successfully detects the biomarkers for EOAD patients	The H-MRS technique is very complex and is rarely used
2019	Ahmed1 et al.	Improving traditional ML methods by reducing overfitting and predicting AD using landmark features of the brain	CNN and patch-based approach	More effective on the same kind model used earlier Accuracy: 90%	Only focused on the hippocampal area and ignored the rest of the biomarkers
2019	Jiménez-Mesa et al.	To overcome the challenge of classifying multiclass problems automatically using MCI data	Multiclass classification using t- test	This method is coherent with recent findings in CAD of AD 67% accurate than present multiclass methods	High redundancy. Limited data makes it prone to bias Cross- validation technique created a mismatch between training and final fitting

(continued)

Table 1 (continued)

Year	Author	Objective	Method	Outcome	Limitations
2019	Raza et al.	Prediction of AD using inertial sensors to track daily work activities and classifying the data using deep learning	CNN (AlexNet) SVM	It shows that physical movements when captured accurately can help in early detection Smartphone: 98.1% Multisensory: 98.3%	Sensor data is always prone to errors Patients have to deal with sensors all over the body and smartphone all the time
2019	Ge et al.	To study the different tissue areas like gray matter, white matter, and cerebrospinal fluid for AD prediction	Multiscale CNN	Using two-stream brain regions (GM and CSF) gives more accuracy Randomly partitioned data accuracy average: 98.29% Subject separated partitioned dataset: 89.51%	MCI subjects were ignored.
2019	Ju et al.	Processing fMRI and clinical data to predict AD at an early stage	Auto-encoders	Standard deviation decreased by 51.23% 20% increase in classification Increase in prediction accuracy by 31.21%	It was carried out on a limited dataset of ADNI
2019	Rohini and Surendran	To reduce the risk of development in AD patients by detecting at an early stage	Ensemble learning	Volumetric reduction and thickness are the first changes that happen in AD patients Accuracy: 89%	Many wrongly classified data points within the range and outliers were misclassified
2018	Hojjati et al.	To combine the sMRI and rs-fMRI data to identify the patients using multi-modality model	SVM	The combination of data produces more accurate results than the single modality approach Accuracy: 96.9%	Using an SVM classifier increases the chances of error as it leaves the outlier data

(continued)

Table 1 (continued)

Year	Author	Objective	Method	Outcome	Limitations
2018	Salvatore and Castiglioni	Designing a multi-label classification model to classify people into four different classes	Multi-label classifier	MMSE and hippocampus-related features are a must for early detection of AD Accuracy: 90%	Multi-grouping into binary classification could be better as some comparisons gave very little accuracy
2018	Nanni et al.	Detection of AD at an early stage using MLNech dataset	SVM Gaussian process classifier Ensemble learning	A unique method to convert high dimension data into a low dimension data Accuracy: 55%	Feature selection is not up to mark Dummy and fake data present in actual data
2018	Liu et al.	To overcome the limitations of predefined features of MRI by using landmark-based features	Deep multi-instance learning	ADNI accuracy: 91.09% MIRIAD accuracy: 92.75%	Limited training subjects were used Many clinical scores of each subject were ignored
2017	Zhang et al. (Member, IEEE)	Diagnosis of AD from landmark features by studying longitudinal structural MR image	SVM Landmark-based feature extraction method	Avoids time-consuming steps of nonlinear registration and tissue segmentation Deals with the problem of inconsistent scan numbers Accuracy: 94%	Longitudinal analysis is more complex than cross-sectional analysis
2017	Previtali et al.	To propose an automated approach for classifying AD using MRI	SVM	New feature extraction technique Orient FAST and Rotated BRIEF helps in identifying useful features to work on Accuracy: 99%	Risk of classifier overfitting Limited approach as it depends only on training data
2017	Gao et al.	Early prediction of AD using CT images	CNN	Combining 2D and 3D CNN outperforms the 2D model Accuracy: 87.6%	Complex model design Manual quantification of images can produce errors

(continued)

Table 1 (continued)

Year	Author	Objective	Method	Outcome	Limitations
2016	Beheshti et al.	To develop a computer-aided design by ranking the features of sMRI data	SVM	Feature extraction is necessary to improve the results Accuracy: 92.48%	SVM classifier can wrongly classify the results excluding data points as outliers
2016	Land and Schaffer	To diagnose AD at an early stage using speech and demographic features	Bayesian Naïve SVM GRNN Oracle	Accuracy = 97%	The sensitivity of features toward Bayesian naïve is very limited
2013	Ayhan et al.	Using ML methods and PET scan to detect AD	Naïve Bayes SVM	Feature selection using correlation is important for correct classification	PET scan is dangerous if the subject has diabetes Other kinds of biomarkers were not included

SVM classifier model is used in [1] to classify the dataset given to it. Gait features and MCI features were studied to get a result. Detection of Alzheimer's at an early stage depends on the different functionalities of the brain, and SVM can classify the patients accordingly [2]. focused on paralinguistic acoustic features. Speech impairment is one of the symptoms of AD. It can be used for early detection, and by applying KNN, SVM, LDA, and decision trees, the patients of AD can be identified. Clustering does reduce the accuracy of the model, but, still, it is one of the unique ways of diagnoses [3]. noticed that most of the researches is done on fMRI datasets. This paper combined fMRI data with sMRI data and performed a multi-modality method using an SVM classifier. Both combined data can be used in a better way to detect AD at an early stage [4]. emphasized on different features of data. Choosing features is as important as choosing relevant data so the main focus of this paper is to choose relevant features using a feature extractor called Oriented FAST and Rotated BRIEF and then classifying using an SVM classifier. MRI data is the key feature in identifying AD patients and [5] used that to identify the patients of AD using structural MRI data. A computer-aided design model was developed using an SVM classifier, and the features were selected and ranked according to their importance.

With a deep learning model, an algorithm can determine on its own if a prediction is accurate or not through its neural network [6]. observed that, using the unsupervised CNN method with CAD, early diagnosis can be done. MCI features are studied and are processed through the CNN model which learns from the training of the dataset. Unsupervised CNN has its limits as it cannot give the same accuracy when the data size is large [7]. performed both machine learning and

deep learning techniques on the same dataset. CNN outperforms KNN with a great margin while detecting AD at its early stage [8]. conducted research that focused on the hippocampus area using the CNN approach. The hippocampus shrinks at more than the constant rate during Alzheimer's condition. The focus on hippocampal features only gave high accuracy. According to [9], there are many features that MRI data fails to capture, and the predefined data has some shortcomings. The shortcoming can be overcome using land-marking features. Two different kinds of datasets are applied to the deep learning model but failed to produce an impact as the training set is limited as mentioned in the paper [10]. used computerized tomography (CT) images to diagnose AD at an early stage. A combination of 2D CNN and 3D CNN produced a complex model using 2D images along spatial axis direction and 3D images as segmented blocks. It is a unique method to visualize a 3D image of the brain to get a better understanding of brain features. The real-time data is more useful in every possible aspect and its importance is given by [11]. A multimodal multitask deep learning model based on real-time data is used for the detection of Alzheimer's.

Ensembled CNN and BiLSTM methods are used to investigate the joint prediction of multiple regression and multiclass classification variables from multiple longitudinal and BG data. This model is an advanced step toward the prediction of Alzheimer's disease at its early stages [12]. predicts the AD at its early stage so that the complications that occur at later stages lead to inevitable results. This paper targets the parameters like shrinkage of the brain with abnormal rate, thickness, and volumetric reduction as key features. According to the paper, these changes are the first identification markers that help in identifying AD at an earlier stage.

[13] used voting method to classify the model into four different classes. A multi-label classifier was used to classify MMSE and hippocampus features. It was observed that the hippocampus plays important role in the early prediction of AD. Due to the multi-classes and different voting schemes, the accuracy of the model got reduced [14]. used T-1 MRI data with reduced features to be processed on an ensemble model to predict the AD at its early stage. The results were compared with different machine learning models, and multi-classification was done. The early signs that an AD patient develops is the change in coordination of brain systems, and [15] studied one of those features. Speech impairment is common among AD patients, and, when combined with demographic features, the results stated that these features can be used to detect AD at an early stage. Bayesian naïve classifier with ensemble learning makes a strong model, but sensitivity of data toward Bayesian naïve is not up to mark. Auto-encoders method when used by [16] along with an improved deep learning algorithm gained an accuracy of over 90 percent, thus reducing the standard deviation by 45%. Auto-encoders do cause overfitting and underfitting problems, but it proves that deep learning can play a crucial role in the reliable and efficient detection of the problem. Detection at an early stage is very important, and MEG comes in handy according to [17]. Using 2D-CNN, the brain activities were observed as AD affects the brain. Although the demographic variables and MSME observations were not considered, still they can be trusted.

According to [18], computer-aided diagnosis can help in differentiating between Alzheimer's disease patients and normal people, and, when this approach is used with multiclass classification, the accuracy of the model increased many times even though there was high redundancy and limited data [19]. [19] observed the hippocampus area of the brain and concluded that volumetric reduction of the hippocampus is an important indicator for early-stage prediction of AD. According to the paper, left hippocampus area provides much more information on AD.

[20] implemented a novel way to study brain functions. The tissue area's gray matter, white matter, and cerebrospinal fluid behavior after observation can be used as a marker to detect AD at an early stage. Gait features are one of the first things to get affected, and in [1] journal it was observed that gait features when studied using an SVM classifier can help in the early diagnosis. Still, dependency on only gait features is not enough. So, the results in Positron Emission Tomography (PET) scan and fMRI, when ran through CNN and other machine-learning classifiers by [21], were very satisfying, achieving the accuracy of 99.95% for fMRI.

[2] focused on paralinguistic acoustic features. Speech impairment is one of the symptoms of AD. It can be used for early detection, and, by applying KNN, SVM, LDA, and decision trees, the patients of AD can be identified. Clustering does reduce the accuracy of the model, but, still, it is one of the unique ways of diagnoses.

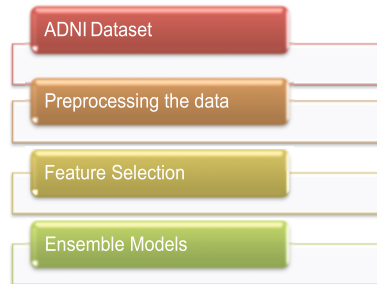
Machine learning and sensors are a great pair and [22] used real-time data collected from sensors to detect the AD. The collected data when applied with CNN was able to identify all the three stages of Alzheimer's. Although patients had to carry smartphones everywhere with sensors attached to their bodies, still, real-time data can help us diagnose AD much faster. The collection of real-time data produces very valuable results, and [23] used this idea to study the daily movement data of a person. The idea is to study the difference between the inertial movements of a normal and an AD patient. Classification of results is done using deep learning models and machine learning techniques. People living in different areas of the world have different kinds of physical and genetic features. They develop different immunity levels [24]. [24] used landmarks feature to study Alzheimer's disease. Using an SVM classifier based on the longitudinal structural MRI, the classification was done successfully. Auto-encoders method implemented by [25] helps in adjusting data according to filters to give us better results, and, when the ADNI dataset was used, there was a drop in standard deviation in great number. Costing the computation time, the accuracy was improved.

3 Methodology and Implementation

Classification of data means grouping a dataset into several predefined classes. When the data is very large, processing that amount of data manually is very tiresome and time-consuming. There needs to be a method that can identify the classes accurately in a very short time. Machine learning and its techniques are designed to work on that kind of large data and give results accordingly. The

stepwise approach from initial stage to final stage of the study has been shown in Fig. 1 as workflow diagram.

Fig. 1 Workflow Diagram



3.1 Dataset Selection

There are several ways to collect data for AD. MRIs, PET scans, and collecting MCI data are the most popular ways of collecting it. Every test specifies a different aspect and different trait of the brain. Some researchers collect data on their own, and some work on pre-available data. Instead of focusing on a specific type of data, several types of data can be combined to create a heterogeneous data so that a diverse data can be created. Alzheimer’s Disease Neuroimaging Initiative (ADNI) is a multisite study of different traits of the human brain that contain the data of healthy people, Alzheimer’s patients, and those who might develop AD in the future. ADNI datasets have been used in thousands of research papers which prove that ADNI is an ideal choice for the dataset. It can be used by anyone and is easily available. The dataset contain 90,000 rows and 19 columns. First 18 columns store different types of information related to the brain, while the 19th column defines class. There are two classes: positive and negative. Thirty percent of the data is used for the testing purpose. Also refer to appendix.

3.2 Preprocessing of Data

Sometimes, the data we obtain is raw. The data may have some kind of noise in it. There are chances that the data may have some unwanted values in several columns, or maybe some columns are empty, and there are no values inserted in it. Some false data may be present which is not similar to the values inserted earlier. So, there is a need to solve such kind of data noise problems as it may affect the accuracy of the model. Preprocessing is used to solve such issues. Preprocessing library is imported from sklearn, and the data is prepared to be used in a model.

3.3 *Feature Selection*

Collection of different types of data has several different features. It is not necessary that every feature contributes positively to the study. Some features are less useful in model, but they affect the accuracy of the model. It is important to identify those features and eliminate them so that we can get better results by using only valid features. It is very hard to check which features are valid or not by any traditional approach. There are many ways to perform feature selection. One of the best ways is to use the Weka tool for feature selection. Weka inputs data and gives output in a very short time. Features can be classified based on ranks, greedy method, or few best attributes. There are different feature selection models like wrapper method and Boruta.

Boruta is a famous technique used for feature selection. By setting a threshold value through randomized shadow features, different attributes can be rejected or accepted. Features that score above the threshold are accepted, while the rest of the attributes are rejected.

The Wrapper Method works as a feature selection method by following the greedy approach. The evaluation criteria for the selection of features depend on precision, recall, and F1 parameters. It has the forward selection and backward elimination processes which iterate to produce the desired result.

After analyzing these feature selection methods, top six features were selected which contributed maximum to the model. These six features are as follow:

- X raw
- Y raw
- X
- Y
- R
- Log R ratio

3.4 *Proposed Ensemble Model*

There are many classical machine learning models that can be used to classify the dataset, but applying them individually can cause many issues. Among the common problems that a model can face are bias and variance errors. If a model has high bias, that means it is missing important trends and is not performing how we want. Overfitting is very common when the variance of model is high. In high variance, the model overfits on training model. So, we need a model that keeps underfitting and overfitting under control.

Ensemble learning model is the combination of different machine learning models that can give an optimal solution when combined together, in simple language, consulting different experts regarding work and then creating an optimal solution by combining the solution given by each one of the experts.

Ensemble learning models are constructed using a weak learner or a base learner. The base learner is generated through training data. The most common base learner is a decision tree. They are like foundations and are called weak learners because they have high bias or variance. As soon as bias or variance is reduced, it becomes a strong learner.

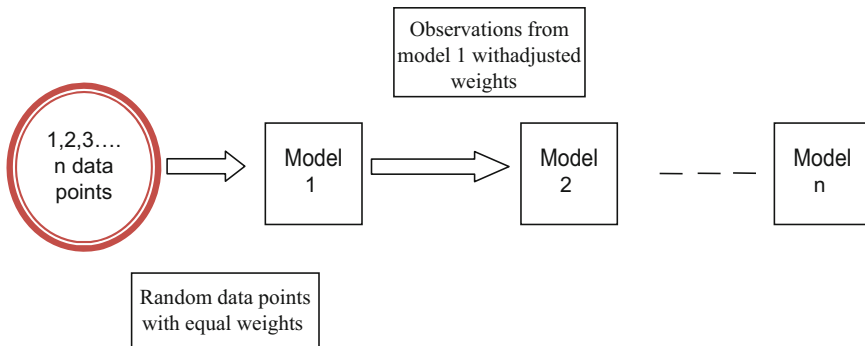


Fig. 2 Boosting mechanism

Boosting is the most prominent algorithm that is used. It handles the bias and variance errors by learning from the shortcomings from the prior model and adjusting the weights.

As in Fig. 2, let’s say we have n data points, and we want to create a strong learner. We randomly select some data points and assign them to model 1. Initially, the weights assigned to them are taken to be equal. Model 1 is a weak learner, so it will misclassify some observations. These observations are sent to model 2, but the weights are adjusted for the wrongly classified observations. Similarly, observations from model 2 are sent to model 3 with adjusted weights and so on until the errors are minimized, and the model becomes a strong learner.

Adaboosting is a famous technique that works on the boosting principles. It works on weak learners which are also known as decision stumps. Single split decision trees are used as decision stumps. Weights are assigned to the observations. The weights are modified for the classes which are hard to classify. For those observations, more weight is added. On the other hand, the value of weight is reduced for observations that can be classified easily. Focusing on every weak learner, the stress is put on the hard to classify observations. The final result is concluded by taking the average of the weighted outputs from all individual weak learners.

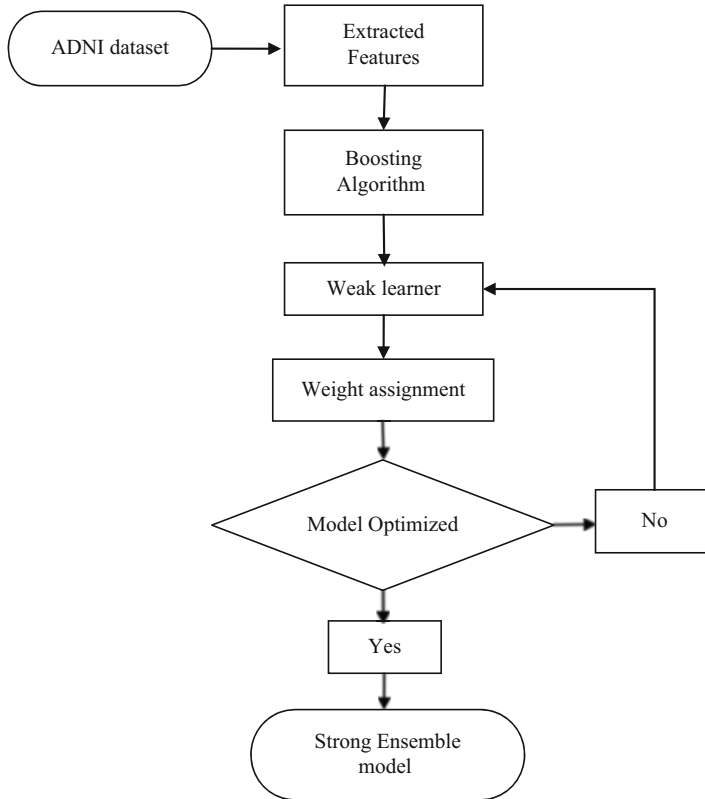


Fig. 3 Flowchart diagram for the implementation of boosting mechanism

3.5 Gradient Boosting Machine Algorithm

Gradient boosting machine algorithm is a combination of Adaboosting and weight minimization. At every step, the weights are adjusted according to the loss functions and are used as input in next model to develop a strong learner. Gradient boosting is divided into two parts: a weak learner and an additive component. Trees are generally considered as weak learners, and the values output by a tree are added to the model so the errors can be reduced.

To form a strong learner, several weak learners are combined. The weak learners are the decision trees with single split. These decision trees are joined in a sequence where every tree tries to reduce the error value of the previous tree by adjusting the weights according to the different instances. The learning speed of boosting technique is generally slow. The reason is its sequential connection. Taking the output of previous tree as input, every tree tries to improve the efficiency of the model by reducing the error. The results are then combined to give a strong learner as an output. The diagrammatical representation for the working of boosting mechanism is shown in Fig. 3.

Loss function is used for the calculation of residuals. Additive component means the different trees defined over the time are added to the model. This doesn't affect the existing trees, and their values are not altered.

Hyperparameters tuning is crucial for comparing the results of different models. The performance of a model depends upon different hyperparameters. In gradient boosting algorithm, the most used hyperparameters are learning rate and $n_estimators$. The speed at which a model learns is known as learning rate which is represented by α . The slower the model learns, the more robust and accurate it will be. To balance the execution time-related issue, the number of trees is increased for the training of a model. It is also denoted by $n_estimator$. So, the learning rate and $n_estimator$ are inversely proportional to each other.

3.6 Extreme Gradient Boost or XGBoost

XGBoost and Gradient Boosting Machine algorithm are both ensemble learning techniques that work on boosting the weak learners. However, XGBoost performs better than GBM because of some of its unique features.

1. Parallelization – It provides the facility to change the order of the nested loops, and this switching of loops is key function to make the base learners strong.
2. Tree pruning – It follows a depth first approach due to which the pruning of trees starts from backward. It uses a max_depth function for it.
3. Hardware optimization – This algorithm is designed to use the hardware in an optimal way. It is able to fit the big data frames that can't be fit in the memory normally and can execute the programs quickly.

4 Results

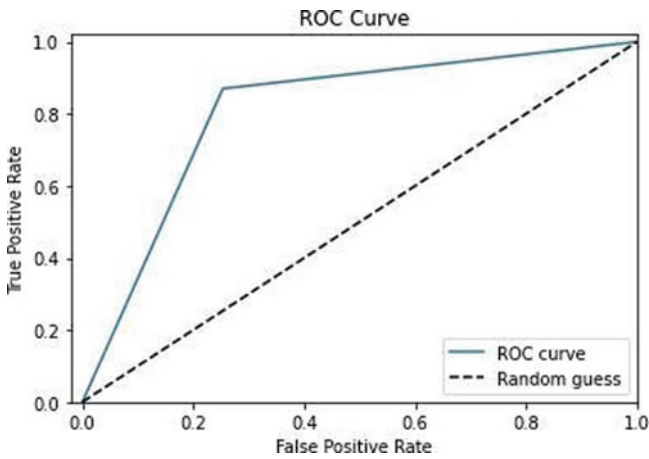
Ensemble learning models are used to overcome the problems that machine learning models face. Ensemble models are efficient enough to control bias and variance of the model and provide us with a model that has low bias and low variance. Boosting is a crucial algorithm of ensemble learning and is used to reduce the loss by significant margin. ADNI dataset is used with XGBoost and Gradient Boosting Machine algorithms. XGBoost model outperformed GBM model by a very small margin. It can also be concluded that the learning rate plays a crucial role in the efficiency of the model. The results obtained after applying the model have been summarized in Table 2.

The receiver operating characteristic curve or ROC curve is designed to display the true positive rate (TPR) and false positive rate (FPR). TPR is also known as recall. It is defined as true positive out of actual positive values, and FPR is defined as false positives by total negative values. The area under the curve defines the accuracy of ROC; the farther the curve is from the threshold value, the more accurate

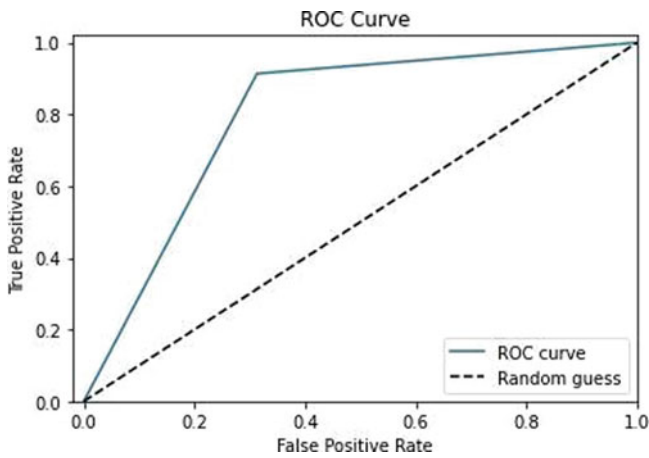
Table 2 Confusion matrix and performance measures for ensemble models

Model	TP	TN	FP	FN	Accuracy	Precision	Recall
<i>XGBoost</i>	13,961	8501	3124	1414	82.40	0.84	0.69
<i>GBM</i>	13,202	8830	2994	1974	81.60	0.82	0.75

the model is. The ROC curve values for the GBM and XGBoost are shown via Graph 1 and Graph 2, respectively. The XGBoost graph shows better ROC values than GBM and scores the value above 0.9.



Graph 1. ROC for GBM



Graph 2. ROC for XGBoost

5 Conclusion and Future Scope

From the results above, it is concluded that XGBoost performed better than GBM. It has also been concluded that every feature from dataset do not contribute for a better result. Only selected number of features is useful that can be used to predict the AD at an early stage. It makes feature selection an important step to attain a good accuracy of the model.

In future, this dataset can be used on other deep learning and machine learning methods. Also, some additional dataset can be used on these models to make the prediction of AD easier and earlier so that lives of patients can be saved.

Acknowledgments. <https://ida.loni.usc.edu/>

Appendix

Dataset attributes	Data values				
GC score	0.9299	0.7877	0.8612	0.8331	0.9466
Allele1 – top	3	1	1	3	4
Allele2 – top	3	1	1	3	4
Allele1 – forward	3	4	1	2	4
Allele2 – forward	3	4	1	2	4
Allele1 – AB	2	1	1	2	2
Allele2 – AB	2	1	1	2	2
Position	139,026,180	139,046,223	219,793,146	219,797,929	219,783,037
GT score	0.8931	0.7824	0.8318	0.8117	0.9126
Cluster Sep	1	0.9871	0.514	0.717	0.9218
Theta	0.94	0.043	0.063	0.991	0.994
R	0.897	2.087	0.359	1.251	1.088
X	0.078	1.955	0.327	0.018	0.01
Y	0.82	0.132	0.032	1.234	1.078
X raw	358	6040	1064	217	209
Y raw	3581	691	201	4029	5108
B allele freq	0.9731	0	0.0213	0.9971	1
Log R ratio	-0.1095	0.1914	-0.5333	0.0402	0.2318
Class	1	1	1	1	1

References

1. B. Ghoraani, L. N. Boettcher, M. D. Hssayeni, A. Rosenfeld, M. I. Tolea, and J. E. Galvin, "Detection of mild cognitive impairment and Alzheimer's disease using dual-task gait assessments and machine learning," *Biomed. Signal Process. Control*, vol. 64, no. July 2020, p. 102249, 2021, doi: [10.1016/j.bspc.2020.102249](https://doi.org/10.1016/j.bspc.2020.102249).
2. F. Haider, S. De La Fuente, and S. Luz, "An Assessment of Paralinguistic Acoustic Features for Detection of Alzheimer's Dementia in Spontaneous Speech," *IEEE J. Sel. Top. Signal Process.*, vol. 14, no. 2, pp. 272–281, 2020, doi: <https://doi.org/10.1109/JSTSP.2019.2955022>.
3. S. H. Hojjati, A. Ebrahimzadeh, A. Khazaee, and A. Babajani-Feremi, "Predicting conversion from MCI to AD by integrating rs-fMRI and structural MRI," *Comput. Biol. Med.*, vol. 102, no. September, pp. 30–39, 2018, doi: <https://doi.org/10.1016/j.compbiomed.2018.09.004>.
4. F. Previtali, P. Bertolazzi, G. Felici, and E. Weitschek, "A novel method and software for automatically classifying Alzheimer's disease patients by magnetic resonance imaging analysis," *Comput. Methods Programs Biomed.*, vol. 143, pp. 89–95, 2017, doi: <https://doi.org/10.1016/j.cmpb.2017.03.006>.
5. I. Beheshti, H. Demirel, F. Farokhian, C. Yang, and H. Matsuda, "Structural MRI-based detection of Alzheimer's disease using feature ranking and classification error," *Comput. Methods Programs Biomed.*, vol. 137, pp. 177–193, 2016, doi: <https://doi.org/10.1016/j.cmpb.2016.09.019>.
6. X. Bi, S. Li, B. Xiao, Y. Li, G. Wang, and X. Ma, "Computer aided Alzheimer's disease diagnosis by an unsupervised deep learning technology," *Neurocomputing*, vol. 392, pp. 296–304, 2020, doi: <https://doi.org/10.1016/j.neucom.2018.11.111>.
7. R. Kumari, A. Nigam, and S. Pushkar, "Machine learning technique for early detection of Alzheimer's disease," *Microsyst. Technol.*, vol. 26, no. 12, pp. 3935–3944, 2020, doi: <https://doi.org/10.1007/s00542-020-04888-5>.
8. S. Ahmed *et al.*, "Ensembles of Patch-Based Classifiers for Diagnosis of Alzheimer Diseases," *IEEE Access*, vol. 7, pp. 73373–73383, 2019, doi: <https://doi.org/10.1109/ACCESS.2019.2920011>.
9. M. Liu, J. Zhang, E. Adeli, and D. Shen, "Landmark-based deep multi-instance learning for brain disease diagnosis," *Med. Image Anal.*, vol. 43, pp. 157–168, 2018, doi: <https://doi.org/10.1016/j.media.2017.10.005>.
10. X. W. Gao, R. Hui, and Z. Tian, "Classification of CT brain images based on deep learning networks," *Comput. Methods Programs Biomed.*, vol. 138, pp. 49–56, 2017, doi: <https://doi.org/10.1016/j.cmpb.2016.10.007>.
11. S. El-Sappagh, T. Abuhmed, S. M. Riazul Islam, and K. S. Kwak, "Multimodal multitask deep learning model for Alzheimer's disease progression detection based on time series data," *Neurocomputing*, vol. 412, pp. 197–215, 2020, doi: <https://doi.org/10.1016/j.neucom.2020.05.087>.
12. M. Rohini and D. Surendran, "Classification of Neurodegenerative Disease Stages using Ensemble Machine Learning Classifiers," *Procedia Comput. Sci.*, vol. 165, pp. 66–73, 2019, doi: <https://doi.org/10.1016/j.procs.2020.01.071>.
13. C. Salvatore and I. Castiglioni, "A wrapped multi-label classifier for the automatic diagnosis and prognosis of Alzheimer's disease," *J. Neurosci. Methods*, vol. 302, no. January, pp. 58–65, 2018, doi: <https://doi.org/10.1016/j.jneumeth.2017.12.016>.
14. L. Nanni, A. Lumini, and N. Zaffonato, "Ensemble based on static classifier selection for automated diagnosis of Mild Cognitive Impairment," *J. Neurosci. Methods*, vol. 302, pp. 42–46, 2018, doi: <https://doi.org/10.1016/j.jneumeth.2017.11.002>.
15. W. H. Land and J. D. Schaffer, "A Machine Intelligence Designed Bayesian Network Applied to Alzheimer's Detection Using Demographics and Speech Data," *Procedia Comput. Sci.*, vol. 95, pp. 168–174, 2016, doi: <https://doi.org/10.1016/j.procs.2016.09.308>.
16. H. Guo and Y. Zhang, "Resting State fMRI and Improved Deep Learning Algorithm for Earlier Detection of Alzheimer's Disease," *IEEE Access*, vol. 8, pp. 115383–115392, 2020, doi: <https://doi.org/10.1109/ACCESS.2020.3003424>.

17. M. Lopez-Martin, A. Nevado, and B. Carro, "Detection of early stages of Alzheimer's disease based on MEG activity with a randomized convolutional neural network," *Artif. Intell. Med.*, vol. 107, no. January, p. 101924, 2020, doi: <https://doi.org/10.1016/j.artmed.2020.101924>.
18. C. Jimenez-Mesa *et al.*, "Optimized one vs one approach in multiclass classification for early alzheimer's disease and mild cognitive impairment diagnosis," *IEEE Access*, vol. 8, pp. 96981–96993, 2020, doi: <https://doi.org/10.1109/ACCESS.2020.2997736>.
19. G. Uysal and M. Ozturk, "Hippocampal atrophy based Alzheimer's disease diagnosis via machine learning methods," *J. Neurosci. Methods*, vol. 337, no. November 2019, p. 108669, 2020, doi: [10.1016/j.jneumeth.2020.108669](https://doi.org/10.1016/j.jneumeth.2020.108669).
20. C. Ge, Q. Qu, I. Y. H. Gu, and A. S. Jakola, "Multi-stream multi-scale deep convolutional networks for Alzheimer's disease detection using MR images," *Neurocomputing*, vol. 350, pp. 60–69, 2019, doi: <https://doi.org/10.1016/j.neucom.2019.04.023>.
21. R. R. Janghel and Y. K. Rathore, "Deep Convolution Neural Network Based System for Early Diagnosis of Alzheimer's Disease," *Irbm*, vol. 1, pp. 1–10, 2020, doi: <https://doi.org/10.1016/j.irbm.2020.06.006>.
22. S. Bringas, S. Salomón, R. Duque, C. Lage, and J. L. Montaña, "Alzheimer's Disease stage identification using deep learning models," *J. Biomed. Inform.*, vol. 109, no. July, p. 103514, 2020, doi: <https://doi.org/10.1016/j.jbi.2020.103514>.
23. M. Raza, M. Awais, W. Ellahi, N. Aslam, H. X. Nguyen, and H. Le-Minh, "Diagnosis and monitoring of Alzheimer's patients using classical and deep learning techniques," *Expert Syst. Appl.*, vol. 136, pp. 353–364, 2019, doi: <https://doi.org/10.1016/j.eswa.2019.06.038>.
24. J. Zhang, M. Liu, L. An, Y. Gao, and D. Shen, "Alzheimer's disease diagnosis using landmark-based features from longitudinal structural MR images," *IEEE J. Biomed. Heal. Informatics*, vol. 21, no. 6, pp. 1607–1616, 2017, doi: <https://doi.org/10.1109/JBHI.2017.2704614>.
25. D. Pan, Y. Huang, A. Zeng, L. Jia, and X. Song, "Early Diagnosis of Alzheimer's Disease Based on Deep Learning and GWAS," *Commun. Comput. Inf. Sci.*, vol. 1072, no. 1, pp. 52–68, 2019, doi: https://doi.org/10.1007/978-981-15-1398-5_4.

Cattle Identification from Muzzle Print Image Pattern Using Hybrid Feature Descriptors and SVM



Amanpreet Kaur, Munish Kumar, and M. K. Jindal

Keywords Muzzle pattern · SIFT · SURF · ORB · SVM

1 Introduction

A cattle biometric is a pattern identification-based system. It is acquiring extension because of its variety of uses and its uses to address visual appearance, conduct investigation, and address dairy cattle biometric highlights [13]. Biometric features give an enormous impact on identification techniques for animals and other species for gaining development in the computer vision-based methodologies for representing and identifying cattle [12]. In recent years, cattle identification has extensively been used for applications such as cattle registration, traceability, tracking, health trajectory, and behavioral analysis and for breeding associations with computer vision and machine learning approaches [13]. Cattle identification also plays a considerable role in false insurance claims, missed and swapped cattle, and managing livestock of cattle and for safety policies of animals [13, 22]. Hence, cattle identification is very essential for the ownership of cattle and the traceability process [15]. The traditional cattle identification methodologies have been classified into three categories, namely, (i) PIM, permanent identification method; (ii) SIM, semipermanent identification method; and (iii) TIM, temporary identification method. PIM techniques include ear tattoos, ear notches, and freezing

A. Kaur (✉)

Department of Computer Science and Applications, Guru Nanak College, Sri Muktsar Sahib, Punjab, India

M. Kumar

Department of Computational Sciences, Maharaja Ranjit Singh Punjab Technical University, Bathinda, Punjab, India

M. K. Jindal

Department of Computer Science and Applications, Panjab University Regional Centre, Sri Muktsar Sahib, Punjab, India

for different cattle identification. SIM includes techniques like ID collar and ear tags, TIM approaches applied signaling technique RFID (radio-frequency identification), and dye-based technique like sketching. The above mentioned traditional cattle identification methodologies have certain boundaries and limitations for the identification of cattle: noninvasiveness, cost-effectiveness, affordability, problems of tracking, physical harm, loss of tags (it faded with weather or get damaged), false insurance claims, missed and swapped cattle, and health trajectory. RFID requires a management-based software system for the identification of cattle or other animals [1, 16]. These artificial marking methods can be duplicated and fraud, and cattle manipulation are possible. The classical identification system based on PIM, SIM, and TIM have several reasons that attain our focus to shift to the new paradigm for cattle identification based on computer vision approach to identify cattle based on their muzzle point pattern image [10–12, 15].

From the literature reviewed, it is found that each breed of the cattle has distinct muzzle point images, and it is most suitable for the identification of individual animals. Moreover, the authors reported that muzzle dermatoglyphics from various breeds of cattle have distinct muzzle patterns: beads, ridges, granola, and vibrissae are used for the identification of cattle. The beads are irregular structures, and their shape is similar to the islands. The ridges have a structure that is similar to rivers [3, 17]. These silent features of extracted features of muzzle point are discriminated and more accurate to identify cattle [18].

1.1 Major Contributions of the Research Work

In this paper, the proposed muzzle point identification system considered a muzzle point image pattern of cattle as a primary biometric feature to identify cattle, as muzzle point image pattern has rich texture information based on skin texture and distinct features of beads and ridges present in muzzle point pattern images. To predict cattle from muzzle point images, few challenges such as poor illumination, poor-quality images are considered to be a barrier for identification of cattle. The proposed muzzle point identification system mitigates the artifacts by using texture descriptor-based algorithms SIFT and SURF and ORB. The rest of this paper is organized as follows: Section 2 presents literature reviews in the field for the identification of cattle. In Sect. 3, characteristics of muzzle point pattern images of cattle has been discussed. Section 4 describes the descriptions of proposed muzzle point pattern-based recognition approach that presents the feature extraction and classification algorithms for recognition of cattle. Finally, Sect. 5 illustrated the performance analysis.

2 Related Work

In the emerging field of research in computer vision, cattle biometrics is getting more proliferation for identification of individual animals, natural habitat, health trajectory, safety policies, and food production system. In a similar direction, Barry et al. [4] proposed a framework with Euclidean distance by achieving 98.85% accuracy. Noviyanto and Arymurthy (2013) in the proposed approach used SIFT on 160 muzzle patterns with equal error rate (EER) value of 0.0167 [18]. Awad (2013) proposed a cattle identification-based framework in which scale-invariant feature transform (SIFT) approach has been used to detect and localize the interesting points for the matching procedure. The authors worked on the dataset of 15 cattle with 90 images (6×15). For this identification, hybrid technology is adopted in which random sample consensus (RANSAC) algorithm and SIFT output remove the outlier's points to reach a more powerful and robust feature. The resultant value from the experimental evaluation is 93.3% [1]. Kusakunniran et al. (2020) have used support-vector machine (SVM) classifier is applied with histogram intersection. The author achieved 95% accuracy [14]. Nurtanio et al. (2020) used SIFT and eliminated features with incompatibility using the RANSAC approach and achieved 93.05% accuracy [19]. Sian et al. (2020) have explored the individual identification of cattle on a single breed with an SVM classification of 95.3% accuracy [21]. Shojaeipour et al. (2021) implemented deep transfer learning by proposing a two-stage YOLOv3-ResNet50 algorithm with 99.13% accuracy for the identification system [23]. Kaur et al. (2022) in this proposed system authors have used Shi-Tomasi, SURF, and SIFT features along with MLP classifier, decision tree classifier, and random forest classifier. Their proposed system has reported a recognition accuracy of 83.35% [9].

3 Muzzle Point Image Pattern

The muzzle point image can be gathered into a database in two forms: (1) muzzle print images that can be lifted on the piece of paper and (2) capturing muzzle image pattern with the camera. Both methods have drawbacks, as in the first method it is very difficult to gather a large amount of data. It's a time-consuming process due to the noncooperative nature of cattle. The second method to capture a muzzle image is easy but may have some artifacts, such as blurriness, noise, and poor image quality. In this emerging field of image processing, two features are considered as global and local features. Global features are derived from the whole image by one vector. Global features describe the image in a generalized form. It includes contour representation, shape descriptor, and texture feature. Certain examples of global descriptors are Gabor, HOG (Histogram Oriented Gradients), PHOG, GIST (Gradient information), invariant moments, and shape matrices. Local features are patterns or small image patches derived from the image. Local features describe the key points in the image. It represents the texture in an image patch. It includes points,

edges, corners, blobs, and ridges. A few examples of local features are SIFT, SURF, LBP, BRISK, FREAK, Harris corner detector and Shi-Tomasi corner detector, Gradient Location-Orientation Histogram (GLOH), and Pyramidal Histogram of Visual Words (PHOW). From the literature reviewed, it is analyzed that global features are used for applications such as object detection – image matching – image stitching, and classification, whereas local features are used for applications such as object recognition, face recognition, and image retrieval. So, based on the literature review, we have identified that we will use local features for individual cattle recognition systems [2, 8]. The authors of the proposed system have used local texture features on the muzzle point pattern images. The artifact of muzzle point pattern image is grouped based on distinct features available in the muzzle defined as beads and ridges [3, 6, 7, 17]. The authors have also stated that individual cattle have distinct pattern-like minutiae points in human fingerprints for recognition of the individual identity of cattle [24, 25].

3.1 Materials and Methods

3.1.1 Data Acquisition

The dataset of the muzzle point image pattern of 930 images of cattle is prepared with 16 megapixels camera to obtain quality muzzle images used for breed classification. The images in cattle dataset were captured from Punjab farms of four breeds: Holstein-Friesian (HF1) breed, Rathi breed, Jersey breed, and Sahiwal breed. The image acquisition was performed indoors and outdoors, and the dataset contains all types of images. After taking the images, the interest area of cattle was segmented manually from the acquired images of cattle. This region of interest is defined between two nostrils of cattle as shown in Fig. 1. This research contributes an emerging perspective to researchers in this field, veterinarians, and scientists for cattle identification using cattle biometrics.

3.1.2 Data Screening

Many of the improper images for the cattle identification were also included in the acquired images because of light sources and other factors as depicted in Fig. 2. As a first step for the identification, it is necessary to select the proper images with no light reflection and high sharped images. These images with light reflection and without light reflection are manually inspected are shown in Fig. 3.

Fig. 1 Region of interest

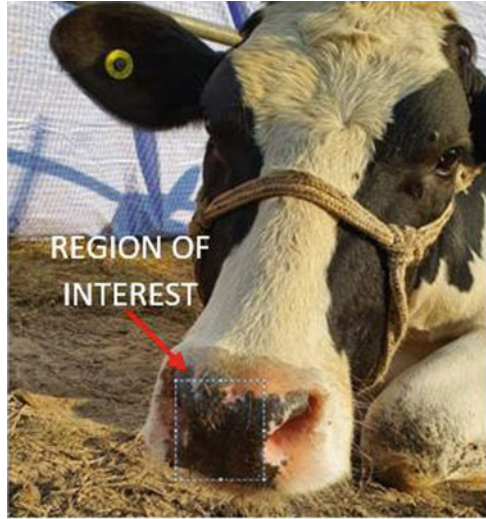


Fig. 2 Improper images

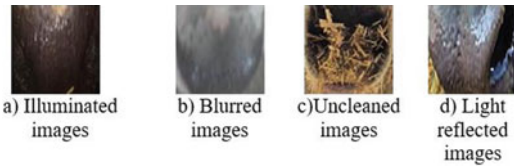


Fig. 3 Muzzle pattern images



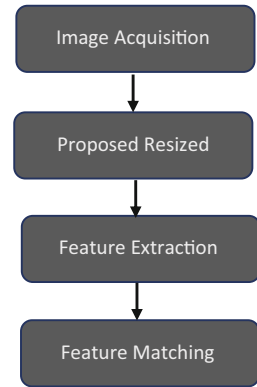
4 Proposed Muzzle Point Pattern-Based Recognition Approach

The proposed cattle recognition system is illustrated in Fig. 4. The working of the proposed cattle recognition system consists of the following steps to recognize the breed of cattle. For the input ROI images, the steps are, namely, (i) no preprocessing application to enhance the muzzle images, (ii) segmentation of muzzle point images pattern, (iii) feature extraction (bead and ridge features) from the segmented muzzle point image pattern with general-purpose local feature extraction algorithms. With the features descriptors extracted from muzzle pattern images, matching was performed. In this study, algorithms were implemented with Python and open CV.

4.1 Image Resizes

The difference in sizes may degrade the identification performance of the system. Therefore, [10, 15, 18] also resized the images to a fixed size of 400×400 , 300×300 , and 300×400 pixels, respectively. If the images are resized to become large and small, information loss may occur that can reduce the accuracy. Therefore, the image was resized to 64×64 pixels depending upon the original size of the image.

Fig. 4 Block diagram of proposed system



4.2 Feature Extraction Algorithm

There are general-purpose feature extraction algorithms such as SIFT, SURF, and ORB proved to be very suitable for the object recognition system. For cattle muzzle, point recognition algorithms were applied to extract the unique muzzle pattern images features. These local feature extraction algorithms consist of key points detectors. These detectors find the key points from the images, and the descriptor describes those key points of an image.

4.2.1 Scale-Invariant Feature Transform (SIFT)

SIFT is an object recognition algorithm proposed by David Lowe (1999) that detects, describes, and matches local features. SIFT overcomes the image degradation factors, and it is considered to be better for scaled images and invariant to transformations. SIFT is a local descriptor that extracts unique features from an input image. SIFT produces the output as a set of feature vectors obtained from an image.

The operational block of SIFT feature extraction consists of these steps:

- **Scale-space extrema detection:** Here, input image is scaled with different scaling factors using Gaussian filters to detect key points. DoG (difference of Gaussian) occurs to take multiple scales from Gaussian blurred images. DoG images are obtained after comparing to its eight neighbors at the same scale and depending upon the pixel value, i.e., maximum or minimum candidate point is selected from compared pixels.
- **Key point localization:** Among the too many key points that are not stabled, key point localization is achieved by DoG to perform accurate location, scale, and ratio. DoG responses high on corners, edges, and blobs; here, edges are rejected from key points as they do not have enough contrast, so it cannot be localizable using Hessian matrix.
- **Orientation assignment:** Here, Gaussian smoothed image at the key point scale is used to describe key points invariance to image rotation. Here, an orientation histogram of 36 bins is formed where each bin covers 10 degrees.
- **Key point descriptor:** Here, the feature descriptor vector for each key point is computed. An orientation histogram is created on 4×4 pixels with 8 bins each over a 16×16 region around each key point, since $4 \times 4 = 16$ histograms each with 8 bins and finally computed as $16 \times 8 = 128$ vector elements.
- **Key points matching:** Here, two images are matched using key points. Matching is done using brute force matcher and k-tree to increase the speed. The matching accuracy decreases over large databases as SIFT features are highly distinctive.

SIFT recognition algorithm is a highly distinctive, relatively easy to extract features. Its computation takes time as a histogram of gradients is performed on each pixel and is very cost-effective.

4.2.2 Speeded-Up Robust Feature (SURF)

SURF (Speeded-Up Robust Feature) was introduced by Bay et al. (2006), conceptually similar to SIFT feature descriptor to localize the interest points and comparison of images [5]. It is a local feature detector used for tracking and object recognition that also focuses on the spatial distribution of gradient information. SURF recognizes objects under different scales that introduce the scale-space concept. A scale space is a function that is used to find the maximum and minimum value across all possible scales; this scale space is created by applying box filters. For scale-space SURF implemented, LOG with a box filter of size 9×9 filter calculates the detector based on the fast Hessian matrix approach to find interest points and choose the one with maxima. To achieve rotational invariance and reproduce the orientation for interest points, SURF uses Haar wavelet filters. All the responses of Haar wavelets calculated within the window in both vertical and horizontal are summed to yield a new vector using the Gaussian function. SURF is faster than SIFT as it produces a 4×4 subregion of length 64 as the SIFT descriptor is 128 in length.

4.2.3 Orient FAST and Rotated BRIEF

ORB was introduced by Ethan Rublee et al. (2011) as an alternative to SIFT and SURF feature extraction algorithms; both algorithms are patented where ORB is free to use, better than SURF, and faster than it. ORB builds on the concept of FAST (Features from Accelerated and Segment Test) to find key points and BRIEF (Binary Robust Independent Elementary Features) descriptor by converting to binary features vector to represent an object; here, each key point is 128–512-bit string. ORB has good performance and low cost when compared to SIFT and SURF [20].

The operational block of ORB feature extraction consists of these steps:

- FAST finds the position of key points.
- Select the n best key points.
- Add direction of points in intensity centroids.
- Extract features using BRIEF.
- Use the greedy algorithm to find low correlative pixel blocks.
- Receive 256-bit descriptor called rBRIEF.

4.3 Classification Algorithms

Image classification is a prime phase to identify images based on extracted features. The basic goal is to classify an image and assign specific labels to it. There are a number of machine learning image classifiers based on supervised and unsupervised classifiers available like SVM (support-vector machine), k -nearest neighbors, random forest, and naive Bayes.

4.3.1 Support-Vector Machine (SVM)

SVM is a supervised machine learning algorithm for analysis and classification introduced by Vapnik et al. (1997) performed on both linear data and nonlinear data. SVM algorithm is used to solve real-world problems like image classification, text categorization, image segmentation, and hand-written characters by assigning labels to the objects or characters. SVM machine learning is also used for a wide variety of biological and other science applications like automatic classification of microarray genes, tumor samples, classifying objects as diverse as protein and DNA.

Basically, there are four concepts required for SVM classification:

1. The separating hyperplane to form a set of hyperplanes to identify the best one.
2. The maximum-margin hyperplane to choose a hyperplane whose distance from it to the nearest data points on each side is maximized so that classification cannot be missed.
3. The soft margin is used when data is not linearly separable.

4. The SVM kernel functions to convert non-separable problems to separable problems used in nonlinear data.

5 Experimental Validation and Performance Analysis

The analysis is measured to evaluate the performance of cattle identification; for this experimental work, the authors have considered their own dataset of 930 cattle images of 4 breeds [9]. For this experimental work, 70% of all datasets are used for training data and 30% of the dataset for testing. Moreover, the results achieved using the dataset partitioning strategy are analyzed by using a threefold cross-validation dataset partitioning algorithm. Table 1 illustrates the classifier-wise recognition accuracy and experimental reports delineating that a hybrid feature extraction algorithm has achieved high accuracy, i.e., 90.00% with linear SVM classifier.

Table 1 Recognition accuracy for cattle identification

Feature extraction techniques	Partitioning strategy Training 70%: testing 30%	Threefold cross-validation
SIFT (8)	81.24%	85.50%
SURF (8)	70.74%	70.90%
ORB (8)	76.65%	83.45%
SIFT (8) + SURF (8)	83.84%	86.68%
SIFT (8) + ORB (8)	87.35%	87.54%
SURF (8) + ORB (8)	85.29%	85.46%
SIFT (8) + SURF (8) + ORB (8)	90.00%	90.22%

Moreover, the authors have also applied a threefold cross-validation dataset partitioning strategy. Experimental results based on a threefold cross-validation technique are also depicted in Table 1 with 90.22% accuracy.

References


1. Awad, A. I. (2016). From classical methods to animal biometrics: A review on cattle identification and tracking. *Computers and Electronics in Agriculture*, 123: 423-435.
2. Bakour, K., & Ünver, H. M. (2021). VisDroid. Android malware classification based on local and global image features, bag of visual words and machine learning techniques. *Neural Computing and Applications*, 33(8):3133-3153.
3. Baranov, A.S., Graml, R., Pirchner, F., et al. (1993). Breed differences and intrabreed genetic variability of dermatoglyphic pattern of cattle, *Animal Breeding Genetics*, 110, (1-6):385-392.

4. Barry B., Gonzales-Barron U. A., McDonnell K., Butler F., & Ward S. (2007). Using muzzle pattern recognition as a biometric approach for cattle identification. *Transactions of the ASABE*, 50(3):1073-1080.
5. Bay H, Tuytelaars T, & Van Gool L (2006). Surf: Speeded up robust features. *European conference on computer vision*, 404-417.
6. Bello, R. W., Olubummo, D. A., Seiyaboh, Z., Enuma, O. C., Talib, A. Z., & Mohamed, A. S. A. (2020). Cattle identification: the history of nose prints approach in brief. *Proceeding of international Conference Series: Earth and Environmental Science* 594(1): 012026.
7. Jang D. H., Kwon K. S., Kim J. K., Yang K. Y., & Kim, J. B. (2020). Dog Identification Method Based on Muzzle Pattern Image. *Applied Sciences*, 10(24):8994.
8. Kabbai, L., Abdellaoui, M., & Douik, A. (2019). Image classification by combining local and global features. *The Visual Computer*, 35(5):679-693.
9. Kaur, A., Kumar, M., & Jindal, M. K. (2022). Shi-Tomasi corner detector for cattle identification from muzzle print image pattern. *Ecological Informatics*, 101549. <https://doi.org/10.1016/j.ecoinf.2021.101549>
10. Kumar, S., Chaube, M. K., & Kumar, S. (2021). Secure and Sustainable Framework for Cattle Recognition Using Wireless Multimedia Networks and Machine Learning Techniques. *IEEE Transactions on Sustainable Computing*.
11. Kühl, H. S., & Burghardt, T. (2013). Animal biometrics: quantifying and detecting phenotypic appearance. *Trends in ecology & evolution*, 28(7):432-441.
12. Kumar, S., Singh, S. K., Abidi, A. I., Datta, D., & Sangaiah, A. K. (2018). Group sparse representation approach for recognition of cattle on muzzle point images. *International Journal of Parallel Programming*, 46(5):812-837.
13. Kumar, S., Singh, S. K., & Singh, A. K. (2017). Muzzle point pattern-based techniques for individual cattle identification. *IET Image Processing*, 11(10): 805-814.
14. Kusakunniran W, Wiratsudakul A, Chuachan U, Kanchanapreechakorn S, Imaromkul T, Suksriupatham N, & Thongkanchorn K. (2020). Biometric for Cattle Identification using Muzzle Patterns. *International Journal of Pattern Recognition and Artificial Intelligence*, 2056007.
15. Kusakunniran, W., Wiratsudakul, A., Chuachan, U., Imaromkul, T., Kanchanapreechakorn, S., Suksriupatham, N., & Thongkanchorn, K. (2021). Analysing muzzle pattern images as a biometric for cattle identification. *International Journal of Biometrics*, 13(4): 367-384.
16. Lowe, D. G. (1999). Object recognition from local scale-invariant features. *Proceedings of the seventh IEEE international conference on computer vision 2*: 1150-1157.
17. Mishra S., Tomer O.S., Kalm E. (1995). Muzzle dermatoglyphics: a new method to identify bovines. *Asian Livestock*, 91-96.
18. Noviyanto, A., Arymurthy, A.M. (2013). Beef cattle identification based on muzzle pattern using a matching refinement technique in the SIFT method', *Comput. Electron. Agric.*, 99:77-84.
19. Nurtanio I, Areni IS, Bugiwati SR, Bustamin A & Rahmatullah M (2020). A Portable Cattle Tagging Based on Muzzle Pattern. *International Journal of Interactive Mobile Technologies*, 14(13).
20. Rublee E., Rabaut V., Konolige K, Bradski G. (2011) ORB: an efficient alternative to SIFT or SURF, *Proceedings of the IEEE International Conference on Computer Vision*, 2564-2571.
21. Sian C, Jiye W, Ru Z, & Lizhi Z (2020). Cattle identification using muzzle print images based on feature fusion. *Proceeding of Conference Series: Materials Science and Engineering* 853(1):012051
22. Singh, P., Devi, K. J., & Varish, N. (2021). Muzzle Pattern Based Cattle Identification Using Generative Adversarial Networks. *In Soft Computing for Problem Solving*, 13-23.
23. Shojaeipour, A., Falzon, G., Kwan, P., Hadavi, N., Cowley, F. C., & Paul, D. (2021). Automated Muzzle Detection and Biometric Identification via Few-Shot Deep Transfer Learning of Mixed Breed Cattle. *Agronomy*, 11(11):2365.

24. Tharwat A, Gaber T, & Hassanien, A.E. (2015). Two biometric approaches for cattle identification based on features and classifiers fusion. *International Journal of Image Mining*, 1(4):342-365.
25. Vapnik, V., Golowich, S. E., & Smola, A. (1997). Support vector method for function approximation, regression estimation, and signal processing. *Advances in neural information processing systems*, 281-287.

Evaluation and Detection of Breast Cancer Using Data Mining Models



B. S. Panda, Satyabrata Patro , and Jyotirmaya Mishra

Keywords Breast cancer · Data mining · Logistic regression · Neural network · Naïve Bayes

1 Introduction

The disease is developed from internal tissues of the female breast. The signs of breast cancer may contain a tumor in the breast, a change in breast size and shape, swelling on some part of the breast, roughness of the skin, and liquid coming from the nipple or a red or scaly patch of skin on a newly inverted nipple. These symptoms are mainly distant spread of the breast cancer; there may be a chance to breathing problem, sometimes bone pain, swollen lymph nodes, or yellow skin [1].

The breast cancer may be caused by family history, being a female, obesity, alcoholism, lack of physical exercise, radiation exposure, hormonal imbalance during menopause, older age, primary age at first menstruation, inherited genes, having children late in life or not at all, and having a personal history.

The reports, investigations, and tests used to identify breast cancer are as follow:

- Breast test: The doctor will test and identify both breast sizes and lymph nodes in the armpit, feeling any pain or other abnormal symptoms.
- Mammogram test: The very important test is mammogram that is an X-ray of the normal breast. It is a common way to test and identify the breast cancer. If a mammography reveals any anomalies or irregularities is detected in the

B. S. Panda (✉)

Department of CSE, Raghu Engineering College, Visakhapatnam, Andhra Pradesh, India
e-mail: panda.bs@raghuengcollege.in

S. Patro · J. Mishra

GIETU, Gunupur, Odisha, India
e-mail: satyabrata.patro@giet.edu; jyoti@giet.edu

mammogram test, then the doctor may advice to get another mammogram test to further investigate the abnormality.

- **Ultrasound test:** The ultrasound test creates images for detecting breast carcinoma within the body. It may be diagnosis whether a new lump is a fluid-filled cyst or a solid mass. Maximum breast lumps are noncancerous cysts or benign.
- **Biopsy:** In this test, doctors use a specific needle to remove tissue or fluid from the infected area. The infected cells are tested in a microscope and examined the symptoms of breast cancer. A biopsy is a proper diagnostic that can find out if the particular cell is cancerous.
- **Magnetic resonance imaging:** In recent research, this test may locate small lesions missed by the mammography. It uses radio and magnet waves to develop pictures of the internal parts of the breast. This test usually may not detect early indication of cancer which is very easily found on a mammogram.

2 Proposed System

In this system, it is to develop a data mining model [2] which has been trained on real examples and has been tested to prove its efficacy. This system eliminates the scope for human error and vastly reduces expenses by automating the diagnosis. Furthermore, this system will serve to vastly improve cancer diagnosis rates and thus facilitate early detection and treatment, saving countless lives.

The objective of this system is to vastly improve the diagnosis accuracy of breast cancer [3], eliminating extenuating costs and the proclivity for human error while providing a cheap, accessible, and available system to automate the process. There are six files used, and five of these files are Python notebooks which preprocess the dataset and build the data mining models. The other file is a text file listing the dependencies of the paper [4], to provide easy initialization first data should preprocessing be done.

3 Proposed Dataset

The proposed dataset, BC Wisconsin, in Table 1 is downloaded from the repository of UCI ML for diagnosis and predicts that the cancer is in early stage or advanced stage. The dataset contains 569 observations of 32 attributes.

3.1 Sample Dataset

The following features are present in the dataset in three forms (mean, standard error, and worst):

Table 1 Sample dataset of Breast Cancer Wisconsin

ID	Diagnosis	Radius mean	Texture mean	Perimeter mean	Area mean	Smoothness mean	Compactness mean	Concavity mean
842,302	M	17.99	10.38	122.8	1001	0.1184	0.2776	0.3001
842,517	M	20.57	17.77	132.9	1326	0.08474	0.07864	0.0869
84,300,903	M	19.69	21.25	130	1203	0.1096	0.1599	0.1974
84,348,301	M	11.42	20.38	77.58	386.1	0.1425	0.2839	0.2414
84,358,402	M	20.29	14.34	135.1	1297	0.1003	0.1328	0.198
843,786	M	12.45	15.7	82.57	477.1	0.1278	0.17	0.1578
844,359	M	18.25	19.98	119.6	1040	0.09463	0.109	0.1127
84,458,202	M	13.71	20.83	90.2	577.9	0.1189	0.1645	0.09366
844,981	M	13	21.82	87.5	519.8	0.1273	0.1932	0.1859
84,501,001	M	12.46	24.04	83.97	475.9	0.1186	0.2396	0.2273
845,636	M	16.02	23.24	102.7	797.8	0.08206	0.06669	0.03299
84,610,002	M	15.78	17.89	103.6	781	0.0971	0.1292	0.09954
846,226	M	19.17	24.8	132.4	1123	0.0974	0.2458	0.2065
846,381	M	15.85	23.95	103.7	782.7	0.08401	0.1002	0.09938

- Radius [define space from middle to points average]
- Texture [define SD in gray scale value averages]
- Perimeter [define points on the “perimeter” mean]
- Area [define mean of the area distance]
- Smoothness [define deviation in radius length]
- Compactness [define $\text{perimeter}^2/\text{area} - 1.0$ value]
- Concavity [define the value of the contour]
- Concave points [define the count of concave portions of the contour]
- Symmetry [define the summary of symmetry]
- Fractal dimension [define coastline approximation – 1]

4 Methodology

4.1 RF (*Random Forest*)

A random decision forest is a ML technique used to solve classification and regression problems. Random forest utilizes a technique that combines many classifiers to produce results to complex tasks, and it consists of DT (decision tree). Random forests construct MDT (multiple decision trees) and combine to get exact and accurate prediction results. The random forests are randomly choosing the training dataset and build the classification and prediction of the distinct or unique random trees [5].

The random forest method contains more number of unique decision trees that builds as an ensemble. Each unique tree in the random forest divides the class prediction, and it builds the accurate model’s prediction.

An increased similarity uncorrelated trees (models) function as a group, and it will form any number of the unique component models (trees).

In decision tree, the minimum correlation model between the nodes are made as the key like how the savings with small correlations (similar as bonds and stocks) combine together to build a selection that is bigger than the total of its items. Uncorrelated tree models can build ensemble results that are more exact than any of the unique predictions. The main motive for this perfect result is that the models protect every new form of their unique errors.

4.2 LR (*Logistic Regression*)

Logistic regression (LR) is a ML technique that is a simple method that requires a logistic model to build a dependent binary data value, though maximum complicated extensions developed. The LR analysis, a logit regression (or logistic regression), is analyzing with the arguments of a logistic design model (binary regression).

Statistically, a logistic binary method contain a dependent attribute with two probable values, such as true/false which is denoted by the pointer attribute, where the two attributes are considered as “1” and “0” [6].

In this approach, it predicts the odds-ratio to build many explanatory attributes (the model of the logistic-odd) with the attribute named as “1” being the linear arrangement of unique or many unique attributes (“observations”); the unique attributes may be both a binary attribute (“indicator value”) and any real value (“continuous value”). The equivalent possibility of the attribute named as “0” can differ between “1” (positively the value “1”) and “0” (positively the value “0”). The LR model converts odds ratio to possibility ratio to estimate every event with relation between probabilities and the features.

The logistic regression becomes very popular in online marketing to predict the sales percentages. It prevents risk factors in healthcare to identify the diseases and is also used very much in weather forecasting to predict rainfall or snowfall. Furthermore, it forecasts whether a loan borrower in banking will default to pay back the loan amount or not depend on his annual income and past defaults [7].

4.3 NB (Naïve Bayes)

The NB classifiers are a group of classification techniques [8] depending on the Bayes’ theorem. The classifier naïve Bayes is not a single technique but a bunch of techniques where maximum of them part a common method, is every couple of structures is classified as unique of one and other.

The naïve Bayes hypothesis is used in every structure builds an independent or same contribution to the results. With combination to the data values, this method can be easily assumed independence between predictors [9].

Next, each technique is given the equal importance (weight), for instance, knowing simply only the humidity and temperature can’t predict the result exactly. Nothing of the variables is unrelated and expected to be causing equally to the result [10].

Bayes’ classifier predicts the probability of an incident arising given the probability of another incident that has already been completed. Bayes’ classifier is built mathematically as the following equation:

$$P(X|Y) = \frac{P(Y|X) P(X)}{P(Y)}$$

The classifier attempting to predict the probability of an event X specific an event Y is true accordingly the event is also called as confirmation.

4.4 NN (Neural Network)

A NN is a collection of links of networks or neurons; in a digital era, an ANN (artificial neural network) is combined with artificial nodes or neurons. So, the NN [11] is either an artificial neural network or a biological neural network, made up of actual biological neurons for simplifying AI (artificial intelligence) tasks. The networks of the biological neuron are established as weights. A confident weight returns an excitatory links, while destructive values return inhibitory links. Each reaction is altered and given more weight. These actions will be denoted by a straight arrangement. In conclusion, a beginning task controls the amplitude of the result, which means a suitable range of result is frequent among 1 and 0 [12].

Neural networks (NN) can be used in variety of applications. Assuming responsibility for ANN [13] (artificial neural networks) is a useful technique include the wide-range of classes below:

- Regression analysis or function approximation, containing time series estimate and displaying
- Classification, containing sequential decision-making and pattern recognition [14]
- Data processing, containing clustering, filtering, compression, and blind signal separation [15]

5 Result and Discussions

To obtain the results, the applied technique here is Python programming. Python is an interpreter, general-purpose, high-level language. It was mainly developed for code accessibility, and its debug allows programmers to write less lines of code.

5.1 *Random_forest.ipynb*

The random forest model runs a bunch of decision trees at once. All the errors in every tree compensate for the other errors; hence, there is no need for feature scaling. In the field of healthcare, it is used to classify the exact combination of components in medicine and to identify a patient's medical history to identify the breast cancer.

```
y = df.diagnosis.value
x_data = df.drop(['diagnosis'], axis = 1)
#Ttrain and test dataset to be split
x_train, x_test, y_train, Y_test =train_test_split
# Classification of Random Forest
From sklearn.ensemble import(rp_classification)
rp=rpclassifier(n_estimators = 1000, state =1)
rp.fit(x_train, y_train)
```



```
acc=rp.score(x_test, y_test)*100
print ("randomforest accuracy score :format(acc))
```

The accuracy count of random forest algorithm is 93.57%.

5.2 *Logistic_regression.ipynb*

```
#Training and testing sets to be split
#set the test-size 0.3
#means 70% is for training and 30% for test
x=df
y=df['diagnosis']
x_train, x_test, y_train, Y_test = train_test_split
(x.y, test_size=0.3, random_state=40)
cols=df.coloumns.drop('diagnosis')
f='diognasis'+.join(cols)
print(formula)
```

This model has given us an accuracy of 96.5%.

5.3 *Naïve_bayes.ipynb*

This theorem is a probabilistic model, not a distance-based one. Hence, we do not feature scale it.

```
y = df.diagnosis.value
x_data = df.drop(['diagnosis'], axis = 1)
#Train and test data to be split
x_train, x_test, y_train, Y_test = train_test_split
(x.y, test_size=0.3, random_state=40)
from sklearn.naive_bayes import GaussianNB
nb=GaussianNB()
nb.fit(x_train, y-train)
acc = nb.score(x_test,y_test)*100
print("accuracy of Navie Bayes: format(acc))
```

The accuracy of naïve Bayes obtained is 94.74%. It is not the best but better than the regression-based models.

5.4 *Neural_network.ipynb*

The neural network is the most sophisticated machine learning algorithm and is so promising; it has branch of deep learning. Neural networks area collection of layers is filled with the previously seen perceptions each interconnected with the next in network.

In order to create and train our neural network to use of the Keras library which runs on TensorFlow backend. This library is a high-level API providing functions to construct and train our neural networks.

```
#Intialising the ANN
classifier = Sequential()
#include input-layer
classifier.add(Dense(output_dime=16,init 'uniform',
  activation='relu', input_dime=30))
classifier.add(Dropout(p=0.1))
#include the hidden-layer
classifier.add(Dense(output_dime=16,init 'uniform',
  activation='relu'))
#include the output-layer
classifier.add(Dense(output_dime=1,init 'uniform',
  activation='sigmoid'))
classifier.compile(optimizer='adam',
  loss='binay_crossentropy', metrics=['accuracy'])
```

The initial accuracy is 69.24%, and each time it improves, we store the parameters in a weights.hdf4 file.

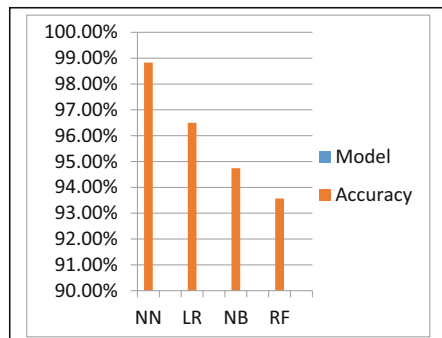
```
filepath = "weights.hd4"
check_point = ModelCheckpoint(fp,monitor= 'value', verbose=1,
  save_bestonly=True)
tboard=Tensor_Board(log_dir='./logs',
  write_graph =True, write_image=True)
callback_list = [checkpoint, tboard]
```

Here, the accuracy has improved to 98.83%, but no further the neural network has attained the highest accuracy we have seen so far, which is also the most sophisticated.

Finally, let us tabulate the results and performances in Tables 2 and 3.

Tables 2 and 3 Result analysis

Rank	Model	Accuracy
1	Neural Network	98.83%
2	Logistic Regression	96.50%
3	Naïve Bayes	94.74%
4	Random Forest	93.57%



6 Conclusion

This paper serves as an efficient, effective, and accurate replacement to existing medical practices used for diagnosing cancer tumors. Mammograms, biopsies, and MRI scans are all traditional techniques plagued by a myriad of problems, the most alarming of them being the rate of misclassification, which is too high when dealing with a life or death disease.

Through our data mining models, this is provided an alternative tool for medical personnel to make use of. This system not only greatly mitigates the human error present in traditional techniques but is also a tool of convenience for both the doctor and the patient. By automating the entire process, the expenses needed for the diagnoses are vastly diminished, making it far more affordable for the patients.

This is concluded by saying that the field of medicine, with its orthodox practices fraught with the threat of human error and misdiagnosis, is an excellent field for the ever-versatile field of machine learning to lay its mark on. By providing highly effective and efficient alternate systems, data mining models might place us on the precipice of a great medical evolution.

References

1. Y.S. Hotko, [2013] "Male breast cancer: clinical presentation, diagnosis, treatment, Exp. Oncol", 35 (4)303–310.
2. BS Panda, SS Gantayat, A Misra, [2015] "Rough set rule-based technique for the retrieval of missing data in malaria diseases diagnosis" in Computational Intelligence in Medical Informatics, Springer, Singapore pp. 59–71.
3. S. Malvia, S.A. Bagadi, U.S. Dubey, S. Saxena, [2017] "Epidemiology of breast cancer in Indian women, Asia Pac. J. Clin. Oncol", 13 (4),289–295.
4. Shallu, Rajesh Mehra, [2018]"Breast cancer histology images classification: Training from scratch or transfer learning", ICT Express 4, 247–254.
5. V. Anji Reddy, Badal Soni, [2020] "Breast cancer identification and diagnosis techniques, in: Machine Learning for Intelligent Decision Making", Springer.
6. ShwetaKharya, SunitaSoni, [2016] "Weighted naive bayes classifier: A predictive model for breast cancer detection", Int. J. Comput. Appl. 133 (9), 32–37.
7. Qiao Pan, Yuanyuan Zhang, Dehua Chen, Guangwei Xu, [2017] "Character Based Convolution Grid Neural Network for Breast Cancer Classification", IEEE, p. 31.
8. SanaUllah Khan, Naveed Islam, Zahoor Jan, IkramUd Din, Joel J.P.C. Rodrigues, [2019]"A novel deep learning based framework for the detection and classification of breast cancer using transfer learning", in: Pattern Recognition Letters, Elsevier.
9. Dumitru, D., [2009], "Prediction of Recurrent Events in Breast Cancer using the Naive Bayesian Classification," Annals of the University of Craiova-Mathematics and Computer Science Series, 36(2), 92–96
10. Qinghua Huang, Yongdong Chen, Longzhong Liu, Dacheng Tao, Xuelong Li, [2020]"On combining by clustering mining and boost for breast tumor classification", IEEE Trans. Knowl. Data Eng. 32 (4), 728–738.
11. Rani, K. U., [2010], "Parallel Approach for Diagnosis of Breast Cancer Using Neural Network Technique," International Journal of Computer Applications, 10(3), 1–5.

12. Huang, M. L., Hung, Y. H., and Chen, W. Y., [2010], "Neural Network Classifier with Entropy Based Feature Selection on Breast Cancer Diagnosis," *Journal of Medical Systems*, 34(5), 865–873.
13. Karabatak, M., and Ince, M. C., [2009], "An Expert System for Detection of Breast Cancer Based on Association Rules and Neural Network," *Expert Systems with Applications*, 36(2), 3465–3469.
14. R.D.H. Devi, M.I. Devi, [2016] "Outlier detection algorithm combined with decision tree classifier for early diagnosis of breast cancer", *Int. J. Adv. Engg. Tech./Vol. VII/Issue II/April-June* 93–98.
15. M Varma, BS Panda, [2019] "Comparative analysis of Predicting Diabetes Using Machine Learning Techniques" in *Journal of Emerging Technologies and Innovative Research (JETIR)*, pp. 522–530.

Lung Cancer Disease Prediction Using Machine Learning Techniques



Selvani Deepthi Kavila, S. V. S. S. Lakshmi, Rajesh Bandaru, Shaik Riyaz, and Neelamsetty Sai Venkata Rushikesh

Keywords Machine learning techniques · Support-vector machine · Logistic regression · Non-small cell lung cancer · Minor lung cancer

1 Introduction

Cancer is one of the leading causes of nonfatal deaths. According to numerous studies, lung cancer is the leading cause of death worldwide. One of the main reasons to suffer from lung cancer is due to the habit of smoking. Approximately 80% of lung cancer patients have a smoking habit. There are some cases where a nonsmoker can also have lung cancer by exposure to radon, second-hand smoke, air pollution, and workplace exposures to asbestos, diesel exhaust, or certain other chemicals. Early detection of lung cancer is critical for improving survival rates. Because of the presence of blood vessels and lymph fluid in lung tissue, cancer cells spread quickly [4]. It is difficult to diagnose lung cancer because symptoms appear in the later stages, and it is nearly impossible to save a person's life in the earlier stages.

S. D. Kavila (✉) · S. V. S. S. Lakshmi · S. Riyaz · N. S. V. Rushikesh
Department of CSE, Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam,
India
e-mail: lakshmi.cse@anits.edu.in; shaikriyaz.20.csm@anits.edu.in;
nsaivenkatarushikesh.20.csm@anits.edu.in

R. Bandaru
Department of CSE, GITAM (Deemed to be University), Visakhapatnam, India

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
R. Misra et al. (eds.), *Machine Learning and Big Data Analytics*,
Springer Proceedings in Mathematics & Statistics 401,
https://doi.org/10.1007/978-3-031-15175-0_41

501

As a result, it is difficult to recognise symptoms of lung cancer until it has spread to other parts of the body. If people go for an early diagnosis, they can reduce the demise phase to administer suitable treatment within the specified time. In short, cancer is the uncontrolled growth of abnormal cells that attack surrounding tissues. Machine learning is an excellent solution to this problem because algorithms can learn faster in a much larger patient population than any set of clinical technology, allowing algorithms to provide more accurate predictions.

A regression-based approach is preferred for this study as the classification models may not be more effective than regression models. The classification models are not considered more effective because the results obtained are continuous but not discrete. Additionally, lung cancer is divided into two subtypes: non-small cell lung cancer (NSCLC) and minor lung cancer (SCLC). This article will focus on patients with non-small cell lung cancer (NSCLC), as it is a complex and challenging disease to treat. Numerous factors must be considered while diagnosing and treating SCLC and NSCLC. There are several approaches for lung cancer detection. One of them is to construct and create a classification and prediction model without utilising SVM or LR algorithms. Data mining approaches frequently include the acquisition of information from significant databases. It has firmly established a foothold in every pasture, including healthcare. It was critical in extracting private information from medical websites.

The mining process encompasses more than data analysis; it also includes classification, clustering, mining union regulation, and prediction. When cancer spreads, a person may develop symptoms in other parts of the body. Its signs are used to determine the disease's severity [12]. The dataset is taken from Kaggle website which is used as the input to the method explored in this study. After that, the dataset is separated into training and a testing set. It trains and tests the data in order to forecast the findings and their correctness. The train set is used to train the model, while the test set is used to predict the outcome and accuracy of the prediction. After training and testing the model, the user can enter his or her matching values in the user interface and then anticipate whether or not he has cancer. Thus, using machine learning to the task of locating lung cancer data enables the reduction of extremely lengthy standards, which benefits both clinicians and patients. To forecast survival outcomes, we use support-vector machines and logistic regression.

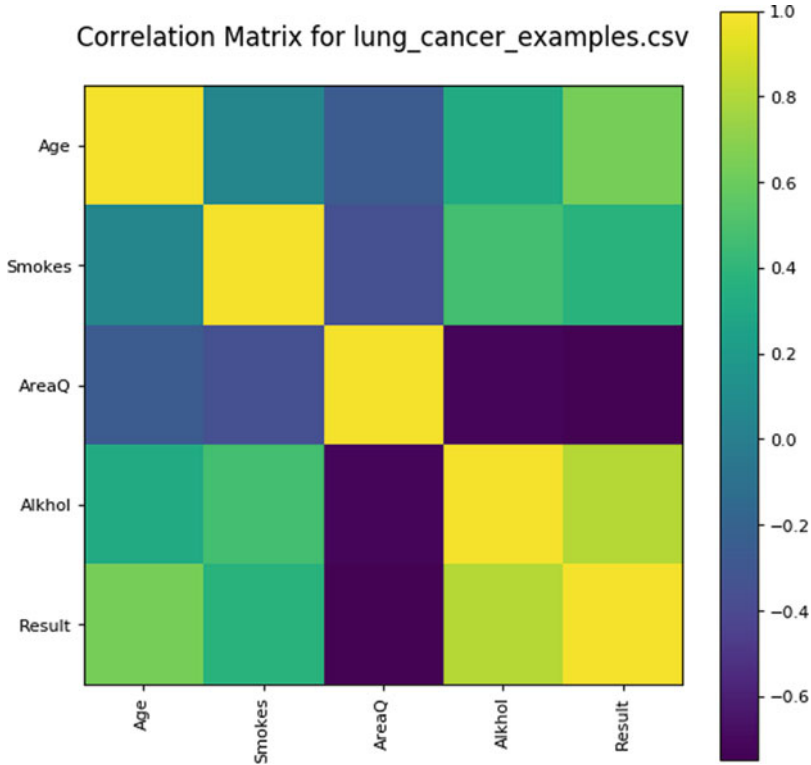


Fig. 1 Correlation matrix displaying relation between various attributes and lung cancer

Figure 1 is a correlation matrix drawn using the lung cancer dataset obtained from Kaggle [11]. Figure 1 represents the relation between lung cancer and different attributes. The paper’s structure is as follows: The Sect. 2 summarises the literature review on detecting and predicting lung cancer. The Sect. 3 discusses the system architecture and proposed method. Section 4 displays the performance analysis and results, and Sect. 5 concludes the paper.

2 Literature Survey

Srinivasan (2020) proposed algorithms for detecting cancerous nodules in the inserted lung image and differentiating lung cancer by size [1]. The proposed model makes use of the most effective feature extraction, such as Local Binary Pattern

(LBP), Scale-Invariant Feature Transform (SIFT), and Histogram of Oriented Gradients (HoG). The fuzzy particle swarm optimization (FPSO) technique selects the optimal feature after extracting texture, geometric, volumetric, and intensity information. The computational complexity of CNN is reduced because of a novel FPSO CNN. Bhatia and Sinha [2] describe a pipeline analysis strategy to highlight cancer-risked lung regions and extract features using UNet and ResNet models [2].

The feature set is fed through various classifiers, such as XGBoost and random forest, and the individual predictions are ensembled to estimate the likelihood of a CT scan being malignant. The ensembled model UNet+RandomForest performs well on the LIDC-IRDI dataset with 84% accuracy. Chaubey and Jayanthi (2020) proposed that rule-based framework predicts heart disease using the SVM decision tree and logistic regression [3].

This approach is based on five modules: prescreening, training, evaluation of individual models, rules of use, and comparison of cardiovascular prediction outcomes. Lynch et al. (2017) described a supervised learning model like the gradient boosting method, support-vector machine, and custom ensemble on the SEER database in this research. The custom ensemble performs well compared to the other models with a root mean square error (RMSE) value of 15.05 [4]. Shaikh and Rao (2021) discussed ML and DL approaches to cancer progression modelling.

Most of the predictions addressed are linked to specific machine learning, input, and data sample supervision [5]. The ML approaches used are support-vector machines and decision trees. The artificial neural network (ANN) is used in the DL approach. Hachesu et al. (2017) proposed that a systematic study aim was to evaluate existing patterns in the data on the risk factor for one year of death after lung cancer surgery. This research focused on looking for patterns in risk factor data for mortality one year after lung cancer thoracic surgery. The proposed dataset consists of 17 features and 470 records [6]. Using data mining algorithms, such as naive Bayes and maximum expectation, and then a regression analysis algorithm, a questionnaire was developed to predict the risk of death one year after lung surgery. In their article, Kadir and Gleeson [7] provided a brief overview of the main methods of predicting the proposed lung cancer to date and highlighted some of its strengths and weaknesses. They also discussed the challenges in developing and validating such strategies and explained the route to clinical adoption [7]. The average AUC score on dataset LIDC-IDRI is 0.70 and 0.91 for the dataset lung cancer data science bowel (DSB), respectively. Orozco et al. (2012) proposed a calculation method for separating lung lumps within CT thorax images in the frequency range [9].

Finally, a support-vector machine with a radial basis function as a kernel was used as the classifier. After evaluating and analysing the results, ten false negatives (FN) and two false positives (FP) were discovered, with sensitivity and specificity of 96.15 percent and 52.17 percent, respectively. The proposed methodology yielded an overall precision of 82.66 percent. Paing et al. (2019) presented a computer-aided diagnosis (CAD) method for detecting and staging lung cancer from computed tomography (CT) images. The average accuracy levels of 92.8% for detection and 90.6% for staging are achieved using BPNN [10]. Experimental findings reveal that the proposed CAD method provides preferable results to previous methods. Raouf

et al. (2020) applied several machine learning models such as logistic regression, support-vector machine, naïve Bayes, and artificial neural networks on various datasets [8, 11].

This study compares existing machine learning models and deep learning models proposed by different authors at a single sight. Shanthi and Kumar (2012) proposed a novel algorithm for selecting elements based on wrapping and using a stochastic diffusion search (SDS) algorithm [13]. The SDS will benefit from direct agent communication to identify the best sub-features. The results showed that the SDS neural network (SDS-NN) had classification accuracy by about 2.51% for the SDS decision tree and about 1.25% for SDS naïve Bayes. Singh and Gupta (2018) have shown an effective way to locate and classify CT scan-related images of lung cancer into benign and malignant tumours. This method first processes images using image processing techniques and then uses algorithms monitored in their classification [14]. The authors used a dataset consisting of 15,750 clinical images containing 6910 benign lung cancer and 8840 malignant lung cancer-related images, respectively. The multilayer perceptron (MLP) classifier performs well on the dataset with an accuracy of 88.55%.

Shao, Cao, and Liu (2012) proposed an algorithm for locating lung nodules by default experimental results and showed that this algorithm can achieve high accuracy and precision and reduce the risk of misdiagnosis, which can provide reference information to a radiologist who detects lung tumours [15]. The SVM model performs well on the problem with an accuracy of 93.407%, sensitivity of 80%, and specificity of 98.485%. Yu-Zin and Won-Ji (2019) proposed and utilised the Azure ML services provided by Microsoft to compare the existing models for lung cancer detection. The survey presents three models: two-class support decision jungle, multiclass decision jungle, and support-vector machine (SVM) [16]. The main limitation of the models is that they cannot handle Big Data efficiently. Zhou et al. (2002) discuss an automatic pathological diagnosis procedure named neural ensemble-based detection (NED), using an artificial neural network ensemble to identify lung cancer cells in images of needle biopsies found in the bodies of prospective subjects [17]. To improve the accuracy of false negatives, the full voting – a novel prediction combining method – is utilised in the first-level ensemble.

3 Proposed System

System architecture describes the high-level overview of major system components and critical working relationships as shown in Fig. 2. It represents the flow of execution.

3.1 Data Acquisition

3.2 Preprocessing Data

3.3 Training Data

3.4 Apply Machine Learning Algorithms

3.4.1 SVM

3.4.2 Logistic Regression

3.5 Test Data

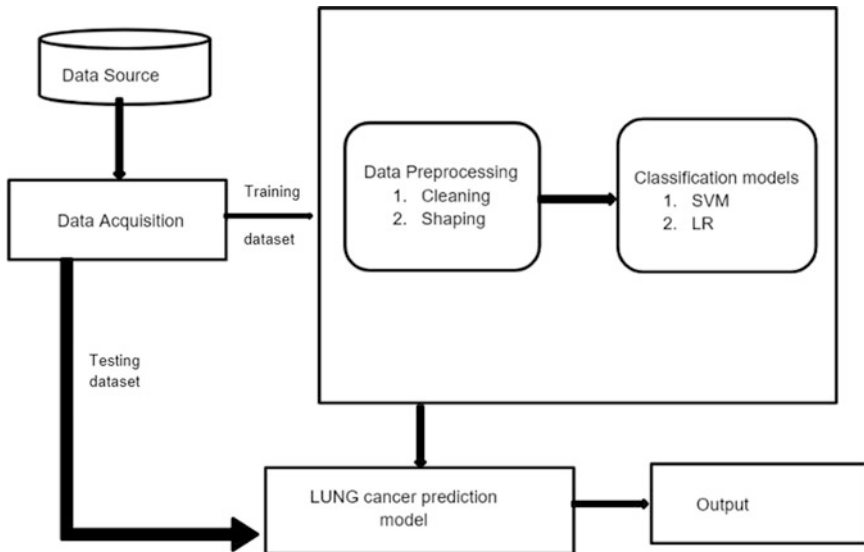


Fig. 2 System architecture

3.1 Data Acquisition

The objective of this step is to obtain a database that provides effective results. In this step, the author needs to collect different datasets that are available in data repositories, and the data might be raw data which is stored in the form of files and databases. Important features to identify the data are quality and quantity. When more data points are added to the database, the result becomes more accurate.

The dataset the author collected is stored and prepared to be used in the training phase. The dataset consists of 16 features (columns) and 309 rows. The features are, namely, GENDER, AGE, SMOKING, YELLOW_FINGERS, ANXIETY, PEER_PRESSURE, CHRONIC DISEASE, FATIGUE, ALLERGY, WHEEZING, COUGHING, etc., which are useful to predict lung cancer.

3.2 Preprocessing Data

The primary goal of this step is for the author to understand the characteristics of the data with which he or she will be working. The characteristics required by the author in order to predict the output must be presented in an understandable manner.

The author's data may contain noise, missing values, and duplicate values that cannot be used directly by machine learning algorithms. Data preprocessing is the process of identifying and removing duplicate values, as well as filling null values with the average of that attribute's values. As a result of completing this process, the final dataset is produced, which can be used for a machine learning model to produce more efficient and accurate results. The author divides the data 70:30.

3.3 Training Data

Machine learning algorithms, in fact, only learn from data. The dataset chosen by the author contains 16 features, of which 10 are chosen to produce more efficient results. The training data is the dataset obtained by the author after dividing the original dataset. The dataset is divided in a 70:30 ratio, with the training data accounting for 70% and the test data accounting for 30%. The training dataset is now fed into the chosen machine learning algorithms. The model can now make decisions based on the relationships between the features in the dataset. The model's performance is evaluated using the training data. The model's outputs improve as the dataset improves. The remaining 30% is used to test the model after it has been trained. The dataset that was used to train the model should be used during the test.

3.4 Apply Machine Learning Algorithms

3.4.1 SVM

Regression and classification can both benefit from the usage of support-vector machines (SVMs). Classification problems, on the other hand, are where they are most frequently employed. For example, in the SVM model, distinct classes are represented by a huge hyperplane. In order to minimise error, SVM will repeatedly generate the hyperplane. In order to create a more complex hyperplane, the SVM aims to partition data sets into classes. The following are the most important SVM support vectors concepts: The hyperplane's nearest data points are known as support vectors. The dividing line will be drawn using these data points. An object can be divided into multiple categories by a hyperplane, which is a plane of decision or space.

A cabinet's margin is defined as the distance between two rows of data items. The distance between the line and the supporting vectors is calculated as the perpendicular distance. While a tiny margin is considered harmful, a huge margin is considered beneficial.

The following are the key concepts in SVM:

Support Vectors – The data points near the hyperplane are called support vectors.

The dividing line will be defined with the help of these data points.

Hyperplane – It is a plane of decision or space divided between a set of objects with different categories.

Margin – It can be described as a gap between two rows in the data points of a cabinet of different classes. It can be calculated as the perpendicular distance from the line to the supporting vectors. A large margin is considered a good margin, and a small margin is considered a bad margin.

The SVM model is classified into linear SVM and nonlinear SVM [8].

Linear SVM: If a dataset can be classified into two classes with a single straight line, it is linearly separable data. The classifier used is the linear SVM classifier.

Nonlinear SVM: If a dataset cannot be classified into two classes with a single straight line, it is called nonlinearly separable data. The classifier used is the linear SVM classifier.

In the case of nonlinear SVM, there are several types of hyperplane (kernel) for segregating the data. Some popular kernels are polynomial kernel, Gaussian kernel, sigmoid kernel, Bassel kernel, and ANOVA kernel.

The hyperplane (kernel) used in this dataset is linear kernel. The mathematical representation of the SVM algorithm is as follows:

$$K(X_1, X_2) = e^{-\left(\frac{\|X_1 - X_2\|^2}{2\sigma^2}\right)} \quad (1)$$

In the above formula,

$K(X_1, X_2)$: Similarity between points X_1, X_2 .

' σ ': Variance

$\|X_1 - X_2\|$: Euclidean distance between two points X_1 and X_2

3.4.2 Logistic Regression

Logistic regression is also a type of supervised learning classification algorithm that is used to solve both regression and classification problems. In the classification problems, the target value may be either 0 or 1. Logistic regression algorithm works on the function called sigmoid which is an s-shaped curve, so the class variable results as 0 or 1, yes or no, true or false, correct or wrong, etc. It is a classification algorithm that works on mathematical functions. This algorithm uses a function

known as logistic or sigmoid which is a complex cost function. This function returns the value between 0 and 1. If the value is less than 0.5, then it is considered as 0, and if it is greater than 0.5, it is considered as 1. Thus, to build a model using logistic regression, sigmoid function is required.

There are three main types of logistic regression:

Binomial

The resultant variable can have only 2 classes either “0” or “1” which represent “true” or “false”, “yes” or “no”, “correct” or “wrong”, etc.

Multinomial

Here, the resultant variable can have more than three possibilities that are not in the order which means it has no measure in quantity like “class A” or “class B” or “class C”.

Ordinal

In this case, the resultant variable deals with organised categories. For example, rating of teaching for the faculty by students can be given as: “very bad”, “bad”, “average”, “good”, “very good”, and “excellent”. Here, each category can be given a score like 0, 1, 2, 3, 4, and 5.

$$f(x) = \frac{1}{1 + e^{-z}} \quad (2)$$

$f(x)$ = Output value which ranges from 0 to 1.

z = Input to the function

e = Base of natural logarithm

The value of the logistic regression must be in the range from 0 to 1. It never goes beyond this limit; only then it is possible to form an S-shaped curve, and it is called the sigmoid function or the logistic function. The resultant value which is greater than the threshold value reaches 1, and a value lower than the threshold value reaches 0.

Logistic regression is a supervised learning method for predicting the likelihood of a target variable. The target or dependent variable is dichotomous, which means there are only two possible phases. Simply put, the dependent variable is binary, with data represented as 1 (indicating success/yes) or 0 (indicating failure/no). The logistic-regression model predicts $P(Y = 1)$ as the X function statistically. It is one of the most basic machine learning algorithms available, and it can be used to solve a variety of classification problems, such as spam detection, diabetes prediction, and

cancer diagnosis. While logistic regression is a linear regression model, it uses a complex cost function that can be described as a “sigmoid function” or a “planning function” rather than a “line function”. Typically, the logistic regression hypothesis constrains the cost function to a value between 0 and 1. Thus, linear functions cannot be used since they can have values larger than or equal to 1, which is not possible in logistic regression.

To predict the cancer, as per the system architecture, the algorithm followed is shown below:

- Step 1: Load the cancer dataset as input.
- Step 2: Clean the missing values from the dataset.
- Step 3: Apply the normalisation technique on the dataset.
- Step 4: Apply feature selection.
- Step 5: Split the dataset into two subsets.
- Step 6: Perform the SVM using Eq. (1) and LR classification using Eq. (2) on the training dataset.
- Step 7: Evaluate the model using Eqs. (3), (4), (5), and (6).
- Step 8: Find and compare the accuracies of the SVM and LR classifiers.
- Step 9: Design the user interface for the model.
- Step 10: Enter the values in user interface, and then the result is obtained.

Train and Test:

- One method for rapidly evaluating the performance of your algorithm is to create a train and split test for your database.
- The training model is constructed from the assessment and training datasets.
- When the output values of the algorithm are reserved, we imagine the test set to be new data.
- We collect predictions from the training model on the inputs to the test dataset and compare them to the concealed output values in the test set.
- By comparing the predictions to the test dataset’s results, we can deduce the performance model for the test dataset.
- This is a metric that indicates how well a trained algorithm can make predictions about hidden data in an issue.

Selecting the Final Algorithm:

- Test the accuracy of each model.
- Select the model with highest accuracy.
- Give different input values and test the output.

3.5 Test Data

After training the lung cancer disease prediction model on the dataset, the author tests the model on the remaining dataset. The ratio is 70:30; 70% of the data

is trained, and 30% of the data is tested in this step, which verifies the model's correctness and accuracy by providing it with a test dataset. The author applies the test data to both algorithms and then determines which model provides the highest level of accuracy.

4 Performance Analysis and Results

This is the last step in the prediction process. Various machine learning-based methods, such as SVM and logistic regression, are used to predict lung cancer in this paper. The author has chosen only ten important features in the dataset. It is important to measure the performance using these mentioned algorithms. The author used various metrics like confusion matrix, accuracy, precision, recall, and F1 score for evaluation.

Accuracy Ratio of number of correct predictions to total number of inputs in the dataset

It is expressed as:

$$Accuracy = \frac{\text{No of patients identified correctly having lung cancer}}{\text{Total No of Patient Records}} \quad (3)$$

Confusion Matrix The output is expressed in the form of a matrix and provides a complete measure of the system.

Table 1 Confusion matrix

	Positive	Negative
Positive	TP	FN
Negative	FP	TN

TP true positive; *FP* false positive; *FN* false negative; *TN* true negative

Precision Ratio of correct positive results to the total number of positive results predicted by the system.

For the lung cancer prediction, it is expressed as:

$$Precision(p) = \frac{\text{No of patients identified correctly having lung cancer}}{\text{No.of patients having lung cancer}} \quad (4)$$

Recall Ratio of correct positive results to the total number of all relevant samples.

For the lung cancer prediction, it is expressed as:

$$Recall = \frac{No\ of\ patients\ identified\ correctly\ having\ lung\ cancer}{No.\ of\ patients\ identified\ with\ lung\ cancer} \tag{5}$$

F1 Score Harmonic mean of both precision and recall. It evaluates test accuracy. The range is from 0 to 1.

It is expressed as:

$$F - measure = \frac{2PR}{P + R} \tag{6}$$

After applying various machine learning algorithms on the speech dataset, the results obtained for each individual algorithm is:

Evaluation for SVM:

Table 2 Confusion matrix for SVM

	Lung cancer	Not having lung cancer
Lung cancer	174	11
Not having lung cancer	19	23

TP = 174; FN = 11; FP = 19; TN = 23

Evaluation for logistic regression:

Table 3 Confusion matrix for logistic regression

	Lung cancer	Not having lung cancer
Lung cancer	153	32
Not having lung cancer	31	11

TP = 153; FN = 32; FP = 31; TN = 11

Table 4 Accuracy table

Algorithms	Accuracy
SVM	86%
Logistic regression	72%

Table 5 Evaluation metrics table

Algorithm	Precision	Recall	F1 score
SVM	0.90	0.94	0.91
Logistic regression	0.81	0.82	0.81

By applying various machine learning algorithms on the dataset, the evaluation metrics that are being used to compare are accuracy, confusion matrix, precision, recall and F1 score. The Tables 1, 2, 3, 4, and 5 results are represented with the help of evaluation metrics [12].



Fig. 3 Performance analysis

The graphical representation of the performance analysis of two machine learning algorithms is shown in Fig. 3. The x- and y-axes represent the algorithms used and the interval scale, respectively. Finally, the author concluded that SVM has the highest accuracy of 86%.

5 Conclusion

Globally, lung cancer is one of the leading and most prevalent causes of mortality from cancer, both in cases of persistent lung cancer and noncancerous lung cancer. The increase in mortality is primarily explained by recent diagnostic and therapeutic mistakes. As a result, early detection is important for avoiding death as a result of this illness.

The survival rate of lung cancer can be predicted using modern machine learning algorithms. As a result, determining the patient survival rate would be prudent. The purpose of this study was to apply data purification, trait selection, categorization, and classification approaches to accurately predict lung cancer survival. This article indicates that the support vector machine model has a maximum accuracy of 86%, but the logistic regression phase has an accuracy of 72%. Additionally, the support-vector machine separator maintains a high degree of phase precision across all phases. This function can be enhanced further by selecting the vector machine phase that achieves the maximum level of accuracy. Additionally, this can be performed by selecting certain features.

References

1. Asuntha, A., Srinivasan, A. “Deep learning for lung cancer detection and classification”. *Multimedia Tools and Applications*,79,1-32(2020).
2. Bhatia, S., Sinha, Y., Goel, L. “Lung cancer detection: a deep learning approach”. In: *Soft Computing for Problem Solving, Advances in Intelligent Systems and Computing* 817, https://doi.org/10.1007/978-981-13-1595-4_55, (2019).
3. Chaubey N.K., Jayanthi, P. “Disease diagnosis and treatment using deep learning algorithms for the healthcare system”. In: *Applications of Deep learning and Big IoT on Personalized Health care services*.PP:99-114, IGI Global (2020).
4. Chip M. Lynch, Behnaz Abdollahi, Joshua D. Fuqua and Alexandra R. deCarlo, “Prediction of lung cancer patient survival via supervised machine learning classification techniques”, *PMC*, DOI: <https://doi.org/10.1016/j.ijmedinf.2017.09.013>.
5. F.J.Shaikh, D.S.Rao, “Prediction of Cancer Disease using Machine learning Approach” 2021 6th International Conference on Recent Trends on Electronics, Information, Communication and Technology, RTEICT 2021, 2021.
6. Hachesu, P.R., Moftian,N., Dehghani,MSoltani, T.S “Analyzing a lung cancer patient dataset with the focus on predicting survival rate one year after thoracic surgery” .*Asian Pacific J.Cancer Prevention:APJCP* 18(6),1531 (2017).
7. Kadir,T., Gleeson, F.Lung “Lung cancer prediction using machine learning and advanced imaging techniques” *Res.2018 Jun*;7(3):304-312.doi: <https://doi.org/10.21037/tlcr.2018.05.15>.
8. Nikita Banerjee, Subhalaxmi Das “Machine Learning Techniques for Prediction of Lung Cancer” *International Journal of Recent Technology and Engineering (IJRTE)* ISSN: 2277-3878, Volume-8 Issue-6, March 2020.
9. Orozco HM, Villegas OOV, Maynez LO, Sanchez VGC, de Jesús Ochoa Dominguez H “Lung Nodule classification in Frequency Domain Using Support Vector Machine” *IEEE*, In international conference on information science, signal processing and their application (2012).
10. Paing,M.P.,Haamoto,K.,Tungjikusolmun,S.,Pintaviroj,C: “Automatic detection and staging of lung tumors using locational features and double-staged classifications”,*Applied Science*,9(11),2329 (2019).
11. Raof, Syed Saba; Jabbar, M A.; Fathima, Syed Aley (2020). [IEEE 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA) - Bangalore, India (2020.3.5-2020.3.7)] 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA) – “Lung Cancer Prediction using Machine Learning: A Comprehensive Approach.”, (), 108–115. doi:<https://doi.org/10.1109/ICIMIA48430.2020.9074947>.
12. Selvani Deepthi Kavila, Rajesh Bandaru, Tanishk Gali (2022). [Principles of Explainable AI in healthcare – Book Chapter] – “Analysis of Cardiovascular disease Using Model-Agnostic Explainable Artificial Intelligence Techniques” 27-54. Doi: 10.4018/978-1-6684-3791-9.ch002.
13. Shanthi S., Raj kumar.N, “Lung cancer prediction using stochastic diffusion search based feature selection and machine learning methods,” *Neural Processing Letters*,pg.no:2617-2630, DOI: <https://doi.org/10.1007/s11063-020-10192-0>, August 2021.
14. Singh,G.A.P., Gupta,P.K. “Performance analysis of various machine learning-based approaches for detection and classification of lung cancer in humans”. *Neural Computing Application* 31(10), 6863-6877(2018).
15. Shao H, Cao L, Liu Y (2012) “A detection approach for solitary pulmonary nodules based on CT images” *IEEE*, 2nd international conference on computer science and network technology.
16. Yu-Zin NAM, Won-Ji SHIN (2019) “A Study on Comparison of Lung Cancer Prediction Using Ensemble Machine Learning” *Shin/ Korean Journal of Artificial Intelligence*, 7(2), pp.19-24.
17. Zhou Z-H, Jiang Y, Yang Y-B, Chen S-F (2002) “Lung cancer cell identification based on artificial neural network ensembles” *Elsevier. Artificial Intelligence Med* 24:25–36.

Video to Text Generation Using Sentence Vector and Skip Connections



Hanumant Mule and Dinesh Naik

Keywords BiLSTM · CNN · Sentence vector · Skip-connection · Video captioning

1 Introduction

Individuals have become intrigued by the rapidly increasing digital culture, which has enticed them to interact with interesting multimedia content such as images, videos, voice notes, and texts. Video, the world's most common document type, is utilized for a variety of objectives, including commemorative events, instructional purposes, proof of concern, information transmission, and corporate promotion. It has been observed that, more cameras such as digital recorder, mobile cameras or CCTV are present than the individuals on the earth. According to the annual Internet study by CISCO, "by the end of 2021, 78% of mobile traffic will be due to the videos." Visual data has increased in recent years as most of the Internet consists of videos. To process this data effectively, there is need of robust algorithms. Nonetheless, it is difficult for computers to comprehend visual input and automatically produce its exact interpretation. Every minute, 500+ hours of videos are posted to YouTube; seeing them all is nearly impossible. Massive volumes of content are successfully indexed and retrieved through video-streaming and sharing services available 24/7. Although these websites classify videos according to their category and time, a precise segment is shown instead of the whole lengthy film. A textual alternative will be more effective in accomplishing the goal and saving time. Without assistance, humans can often provide an accurate video description without numerous grammatical mistakes or misunderstandings.

H. Mule (✉) · D. Naik

Department of Information Technology, National Institute of Technology Surathkal, Surathkal, Karnataka, India



Fig. 1 A person is folding a piece of paper

Figure 1 depicts the very accurate human-generated video description. However, it is not simple for computers to recognize the items involved in their interactions and produce a suitable description. With the advent of CNNs, computers can interpret a picture. The advancement of natural language processing has enabled computers to comprehend human language through word embeddings. Video to text generation is the process of providing a text for a video clip. Video processing is a complex task because videos include both spatial and temporal information. CNNs can collect the spatial information included in pictures, whereas one or more LSTM layers can capture the temporal information provided in the video. For all seq2seq activities, attention is useful in the E-D architecture. When the decoder's context vector is formed as a weighted sum of the encoder's output states, soft attention is beneficial. In this work, multi-headed attention is utilized. As a result, the following concisely highlights our contribution:

1. Word embedding vectors are generated for each token using BERT, GloVe, and ELMo. We compare different embedding techniques based on the quantitative measure.
2. Experiments are conducted using feature vectors extracted from video frames with the help of various CNNs, including Inception-v4, VGG-16, ResNet152, and NASNet-Large.
3. Encoder with skip connections.
4. Use of sentence vectors to help the decoder in generating the suitable description.

The following is how the paper is structured. Section 2 introduces works that are related. The suggested architecture is described in Sect. 3. The experimental results and discussion are covered in Sect. 4. Finally, in Sect. 5, we bring this study to a conclusion.

2 Related Work

This section will see the current research work done in this area. The initial emphasis was primarily on image captioning but was later extended to incorporate video captioning. For this task, most famous architecture is sequence-to-sequence (s2s) architecture. A video consists of a frame sequence, and a video to text generation model produces a series of words. As a result, video captioning is a s2s process [1]. describe the s2s architecture, where input sequence is encoded by encoder and a decoder outputs a translated sentence. After the outstanding performance of the s2s architecture in different sequence to sequence tasks [2], it is only apparent to apply this architecture to video captioning tasks [3]. In recent years, several variants of the s2s architecture have been extensively employed, including hierarchical techniques [4], GAN variants [5], and boundary-aware encoder approaches [6]. Earlier s2s research [1, 3] indicated that the decoder cells construct the next word depending on the context of the previous word and the fixed output of the encoder. As a result, the entire context of encoded information is frequently lost, and the final output is heavily dependent on the preceding hidden cell state. The advent of the attention mechanism [7] paved the way for this problem to be resolved. The attention strategy enables the model to keep context from start to end [8]. used recent improvements in computer vision and machine translation to show a deep-recurrent technique for automatically writing captions for images. He developed a model to optimize the probability of a training image containing the target description phrase [9]. extract more information from a photo by combining top-down and bottom-up approaches with an RNN that can selectively attend to semantic aspects in the image [10]. showed the generation and scoring of captions using a dual-stream RNN model. The algorithm gradually improves at extracting words, verbs, and adjectives from picture areas. Using the phrases above to describe the visual, the model outputs a coherent statement [11]. developed an innovative deep learning architecture based on transferred semantic attributes and long short-term memory (LSTM- TSA). The LSTM-TSA architecture conveys semantic information from pictures and videos to CNNs and RNNs from start to finish. As a result, picture and video semantics complement one another, enhancing video captioning quality [12]. proposed an encoder-decoder architecture where decoder consists of an attention LSTM which improved the caption context [13]. described a novel design for hLSTMat encoder-decoder that incorporates adaptive temporal and temporal attention. This method uses visual and semantic information separately [14]. created a sequential layer dubbed SeqVLAD by combining RCNs with a trainable locally aggregated descriptor (VLAD) layer vector. They validated the framework's effectiveness by performing video captioning and video action recognition tasks [15]. constructed attention in attention networks to study end-to-end attention fusion. It is one-of-a-kind in that it has several encoder and fusion attention units [16]. present a bidirectional LSTM that examines videos in both directions to get information for decoding future time steps. Soft attention is also utilized to focus on targets with predefined probabilities at each time step.

3 Methodology

Initially, we have extracted frames from the videos during the preprocessing step. The extracted frames are then scaled to comply to the input dimensions of pretrained CNN models, namely, VGG16, NASNet-Large, ResNet152, and InceptionV4. Our objective is to provide a caption for a video. For this purpose, we present an E-D architecture based on a sentence vector and encoder with skip connection. To capture the video’s spatial and temporal information, our model encodes the feature vectors extracted at frame level. Figure 2 depicts the detailed architecture of the planned work.

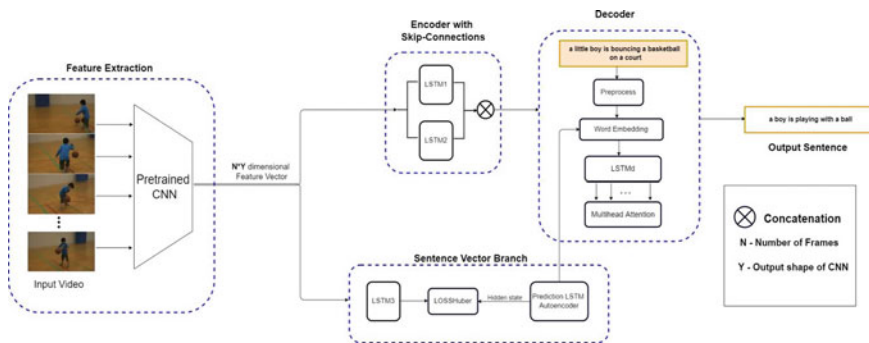


Fig. 2 Architecture of proposed model

3.1 Feature Extraction

Frames are extracted from the video and then resized and scaled as per the input requirements of the pretrained CNN models. Table 1 shows the input image size for the pretrained CNN and its respective output feature vector. FFMPEG library is used to extract the frames from the videos. After preprocessing, feature vectors are extracted from frames using CNN models. Feature vectors are the high-level representation of video. This work uses four 2D CNN models, NASNet-Large, Inception-v4, ResNet152, and VGG-16.

Table 1 Input image dimensions and output vector size for pretrained CNNs

Model	Input Image dimension	Output vector size
NASNet-large	331 × 331	4032
Inception-v4	299 × 299	1536
VGG-16	224 × 224	4096
ResNet152	224 × 224	2048

3.1.1 Encoder-Decoder

The encoder’s job is to extract temporal features from the two-dimensional CNN data retrieved at the frame level, which is accomplished by using long short-term memory (LSTM) networks. Our encoder is built with a two-layer LSTM and skip connections. Based on the video data, the decoder should be able to anticipate the caption. Each word in the caption is generated separately. Thus, the decoder generates the following word given a current and the encoder’s output. We use multi-head attention with eight heads in the model, which allows the decoder to attend to input from the encoder at numerous places from several representational spaces at the same time. The loss associated with the translation of video to text is calculated as given in Eq. (1):

$$\text{loss}_1 = - \sum_{t=1}^{N_w} \log P(w_t | E, w_1, w_2, \dots, w_{t-1}) \quad (1)$$

where N_w = number of words in the caption.

3.2 Sentence Vector Branch

The feature vector from the video is given to an LSTM layer to produce an N-dimensional vector where N depends on the embedding technique. Using BERT, ELMo, and GloVe, we create 768-, 1024-, and 300-dimensional output vector, respectively. The feature vector from the video is given to an LSTM layer to produce an N-Dimensional vector where N depends on the embedding technique. We create 768-, 1024-, and 300-dimensional output vectors using BERT, ELMo, and GloVe. This feature vector is compared with hidden state of the LSTM auto-encoder trained on the caption set during the training process. Because this concealed state stores the caption’s content and is utilized to regenerate the caption. In this way, the decoder gets more information to identify the next word in the sentence. We refer to this vector as a sentence vector (SV) since it summarizes a caption. Huber loss is computed as in Eq. (3):

where $sv = SV$ from the language model
 $sv' = SV$ generated by the sentence vector branch:

$$e_k = \begin{cases} \frac{1}{2}(sv_k - sv'_k)^2 & \text{for } |sv_k - sv'_k| \leq \delta \\ \delta |sv_k - sv'_k| - \frac{1}{2}\delta^2 & \text{Otherwise} \end{cases} \quad (2)$$

$$\text{loss}_2 = \sum_{k=0}^{d_w} e_k \quad (3)$$

3.3 Multi-Head Attention

The entire video is encoded by the encoder, but the decoder selects only a subset of features using scalar dot product attention to generate the next word by giving a current word. The usage of multiple heads enables the decoder to simultaneously attend to input from the encoder at various points in distinct representational spaces.

3.4 Training

We train the proposed architecture using a combined loss function by merging two loss functions. The combined *loss* function is defined as in Eq. (4) where loss_1 and loss_2 is defined in Eqs. (1) and (3), respectively. Here, λ is a hyperparameter with value ranges between 0 and 1.

$$\text{loss} = \lambda \text{loss}_1 + (1 - \lambda) \text{loss}_2 \quad (4)$$

4 Experiments and Results

The experimental results and analysis are presented in this section. Microsoft Video Description (MSVD) and CHARADES datasets were used.

4.1 Datasets Used

1. *Microsoft Video Description (MSVD) dataset*: MSVD [17] dataset is one of the old datasets widely used globally. This dataset consists of 1970 YouTube video clips and its associated human annotations. This dataset was collected

by requesting AMT workers. AMT workers are guided to select short duration (10–25 seconds on average) clips with a single activity and mute the audio. The descriptions are available in Chinese, English, German, etc. languages. On average, 41 single-line sentences are available per video. For training and validation, 1200 and 100 videos are used, and for testing purpose 670 video clips are considered.

2. *CHARADES dataset*: The CHARADES [18] dataset includes 9848 daily indoor household activities videos. A total of 267 AMT employees from three continents shot these videos. AMT workers assigned with scripts which describe the activities to perform on the respective object. Objects and actions to be performed are mentioned in the predefined vocabulary. Videos are shot in 15 distinct indoor scenarios and are limited to 46 items and 157 action classes. The videos in the collection have an average duration of 30 seconds and reflect ordinary living activities. From the dataset, 7985 videos are used for training purpose and the rest for testing purpose.

4.2 Evaluations Metrics

Video to text generation task deals with computer vision and natural language processing. Metrics like BLEU [19], METEOR [20], ROUGE [21], and CIDER [22] used to evaluate the captions are taken from the natural language processing domain. We did a quantitative assessment utilizing the Microsoft COCO caption evaluation tool to compare our results with previous studies. To assess a projected sentence against all ground truth sentences, we employ established metrics like BLEU, METEOR, ROUGE, and CIDER. When the metrics are high, the produced sentence often correlates well with a human evaluation since they measure the entire sentence meaning and fluency. The abovementioned metrics are of choice in recent work because they better correspond with human judgment than other approaches. Consequently, we used them in our tests to compare the results of different architectures. All scores are reported as percentages. Table 2 shows the summary of metrics and meaning of higher values associated with each.

Table 2 Meaning of higher value of the metrics

Sr. No	Metric	Higher value of metric means
1	BLEU	Better resulting sentence
2	METEOR	The sequence of the candidate and reference sentences is more uniform
3	ROUGE	Higher machine translation accuracy rate
4	CIDER	Higher similarity between the candidate and reference sentences

4.3 Data Preprocessing

For each dataset, N frames per video are considered in this work. For MSVD and CHARADES, the dataset value of N is 28 and 80, respectively. Each frame is given as input to the pretrained CNN models and outputs Y -dimensional feature vector. So, for N frames, we get $N*Y$ dimension feature vector for the entire video. Table 1 shows the output vector size Y for each pretrained CNN model. For example, for each video from NASNet-Large, we have an $N*4032$ -D vector. Captions for the videos are compiled from various sources and vary in length. We removed the punctuation from the captions, converted to lowercase, and tokenize them. The vocabulary is refined to exclude misspelled and uncommon terms. Length of captions is limited to 20 tokens. Any caption that exceeds the maximum length is clipped, whereas captions with less than 20 tokens are padded. We have used three-word embedding techniques such as BERT, ELMo and GloVe. Table 3 shows the word embedding and output vector size for each token. In the results section, we have compared the performance of these word embeddings.

Table 3 Embedding techniques and its output vector size

Embedding	Output vector size
BERT	768
ELMo	1024
GloVe	300

4.4 Experimental Setup

For simplicity, let us assume NasNet-Large as feature extraction model and BERT as word embedding used for the experiments. NasNet-Large outputs 4032 dimensional feature vector for a single frame. So, the video to be processed is represented using $N*4032$ dimensional feature vector where N is the number of frames considered for the experiment. Then, this feature vector is passed to an encoder which employed with skip connections. The LSTM is assumed to have 512 units. Thus, we get an $N*512$ -D vector from layer one and an $N*512$ -D vector from layer 2, resulting in an $N*1024$ -dimensional feature vector. The sentence vector includes a single 768-unit LSTM layer and auto-encoder. The attention layers' unit count is set at 512. The decoder LSTM is assumed to have a total of 1024 units. For training, Adam Optimizer is used. During the training process, the learning rate and batch size are set to $6e-4$ and 64, respectively. During testing, beam width is considered as 3 during beam search.

4.5 Results and Analysis

1. *MSVD dataset*: On the MSVD dataset, we have compared our model performance with state-of-the-art methods using the abovementioned four metrics. We have analyzed the results using two ways: qualitative comparison and quantitative comparison. Figure 3 represents the qualitative result, which includes ground truth captions and the captions generated by the proposed model.



Ground Truth: { a kid plays basketball, a boy is bouncing a ball, a boy is playing with ball, a little boy bounces a basketball, an asian kid learning basketball }

Ours : a boy is playing with a ball

Fig. 3 The qualitative comparison of generated caption and ground truth captions

We have compared the BLEU-4, METEOR, ROUGE-L, and CIDER scores with existing work. Upon experimental results, NASNet-Large outperforms Inception-v4 VGG-16 and ResNet-152. Table 4 shows the comparison of the results with various methods. Among all the listed methods, our model outperformed all the metrics mentioned above.

Table 4 Performance comparison with state-of-the-art models on MSVD dataset

Sr. No	Models	BLEU@4	METEOR	ROUGE-L	CIDER
1	FGM [23]	13.68	23.9		–
2	LSTM-YT [24]	33.3	29.1		
3	TA [25]	41.9	29.6		51.67
4	S2VT [26]	–	29.8		
5	h-RNN [27]	49.9	32.6		65.8
6	MM-VDN [28]	37.6	29		
7	GloVe + Deep Fusion Ensemble [29]	42.1	31.4		
8	S2FT [30]	–	29.9		
9	HRNE [31]	43.8	33.1		
10	GRU-RCN [32]	43.3	31.6		68
11	VGG + C3D [33]	45.3	31		
12	SCN-LSTM [34]	51.1	33.5		77.7
13	TDFF [35]	45.8	33.3	69.7	73
14	BAE [36]	42.5	32.4		63.5
15	PickNet [37]	46.1	33.1	69.2	76
16	AMS2S [38]	40.3	31.5		
17	GoogLeNet + C3D [39]	41.9	29.6		51.4
18	Multimodal architecture [40]	48	31.6		68.8
19	VGG16 + Deep-LSTM + Attention [41]	41.2	33.4		
	<i>Ours</i>	<i>50.3</i>	<i>33.9</i>	<i>70.5</i>	<i>78.6</i>

We have compared the results with BERT, ELMo, and GloVe embeddings for every feature extraction technique. By experimental results, BERT embedding outperforms ELMo and GloVe. The same pattern is followed for NASNet-Large, Inception-v4, Resnet152, and VGG16 feature extraction techniques.

3. *CHARADES dataset*: On the CHARADES dataset, we have compared our model performance with state-of-the-art methods using the abovementioned four metrics. Table 5 shows the comparison with existing methods. On the experimental results, our model with NASNet-Large outperforms all other models in terms of METEOR, ROUGE, and CIDER metrics. For the BLEU-4 score, our model stands second.

Table 5 Performance comparison with state-of-the-art models on CHARADES dataset

Sr. No	Model	BLEU@4	METEOR	ROUGE-L	CIDER
1	HRL [42]	18.8	19.5	23.2	41.4
2	TSA-ED [43]	13.5	17.8	20.8	
	Ours	17.2	20.1	23.9	42.3

5 Conclusion

This paper proposed a video to text generation using sentence vector and encoder with skip connection. The sentence vector assists the decoder in improving the performance of the model. Experimental result comparisons of various feature extraction techniques like NASNet-Large, VGG16, Inceptionv4, and ResNet152 have been performed on the MSVD and CHARADES datasets. The model is evaluated with multiple word embedding techniques like BERT, ELMo, and GloVe. On the experimental analysis, NASNet-Large performed well on both datasets. For word embedding techniques, BERT outperforms ELMo and GloVe with all the four pretrained models on the datasets. Our future work would compare the results of different feature extraction techniques such as NASNet-Large, VGG16, VGG19, Inception-v4, I3D (two-stream inflated 3D ConvNets), C3D, etc. The model will be evaluated on different video datasets like MSR-VTT, MSVD, M-VAD, and YouCook II. Along with 2D CNNs, 3D CNNs boost the process of video to text generation.

References

1. Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." *Advances in neural information processing systems*. 2014.
2. Shao L, Gouws S, Britz D, Goldie A, Strophe B, Kurzweil R. 2017. Generating highquality and informative conversation responses with sequence-to- sequence models. ArXiv preprint. arXiv:1701.03185.
3. Venugopalan S, Rohrbach M, Donahue J, Mooney R, Darrell T, Saenko K. 2015. Sequence to sequence-video to text. In: *Proceedings of the IEEE international conference on computer vision*. 4534–4542.
4. Baraldi L, Grana C, Cucchiara R. 2017. Hierarchical boundary-aware neural encoder for video captioning. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. Piscataway: IEEE, 1657–1666.
5. Yang Y, Zhou J, Ai J, et.al. Video captioning by adversarial LSTM. *IEEE Transactions on Image Processing* 2018.
6. Shih H-C. 2017. A survey of content-aware video analysis for sports. *IEEE Transactions on Circuits and Systems for Video Technology* 28(5):1212–1231.
7. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. 2017. Attention is all you need. In: *30th Conference on Neural Information Processing Systems (NIPS 2017)*. Long Beach, California, USA: *Advances in neural information processing systems*, 5998–6008.
8. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: A neural image caption generator. *CoRR abs/1411.4555* (2014)
9. You, Q., Jin, H., Wang, Z., Fang, C., Luo, J.: Image captioning with semantic attention. *CoRR abs/1603.03925* (2016)
10. Fang, H., Gupta, S., Iandola, F., Srivastava, R.K., Deng, L., Dollar, P., Gao, J., He, X., Mitchell, M., Platt, J.C., Zitnick, C.L., Zweig, G.: From captions to visual concepts and back. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 1473–1482 (2015)

11. Pan, Y., Yao, T., Li, H., Mei, T.: Video captioning with transferred semantic attributes. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 984-992 (2017)
12. Gao, L., Guo, Z., Zhang, H., Xu, X., Shen, H.T.: Video captioning with attention-based lstm and semantic consistency. *IEEE Transactions on Multimedia* 19(9), 2045-2055 (2017)
13. Song, J., Li, X., Gao, L., Shen, H.T.: Hierarchical lstms with adaptive attention for visual captioning. *CoRR abs/1812.11004* (2018)
14. Xu, Y., Han, Y., Hong, R., Tian, Q.: Sequential video vlad: Training the aggregation locally and temporally. *IEEE Transactions on Image Processing* 27(10), 4933-4944 (2018)
15. Xu, N., Liu, A., Nie, W., Su, Y.: Attention-in-attention networks for surveillance video understanding in internet of things. *IEEE Internet of Things Journal* 5(5), 3419-3429 (2018)
16. Bin, Y., Yang, Y., Shen, F., Xie, N., Shen, H.T., Li, X.: Describing video with attention-based bidirectional lstm. *IEEE Transactions on Cybernetics* 49(7), 2631-2641 (2019)
17. D. L. Chen and W. B. Dolan. Collecting highly parallel data for paraphrase evaluation. In *ACL*, pages 190-200, 2011.
18. Sigurdsson, Gunnar A., et al. "Hollywood in homes: Crowdsourcing data collection for activity understanding." *European Conference on Computer Vision*. Springer, Cham, 2016.
19. Papineni, Kishore, et al." Bleu: a method for automatic evaluation of machine translation." *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002.
20. A. Lavie and A. Agarwal, "METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. 2005
21. C. Lin. 2004. Rouge: A package for automatic evaluation of summaries. in: *Text Summarization Branches Out*.
22. R. Vedantam, C. L. Zitnick, and D. Parikh. 2015. Cider: Consensus based image description evaluation. In *IEEE CVPR*.
23. J. Thomason, S. Venugopalan, S. Guadarrama, K. Saenko, and R. J. Mooney. 2014. Integrating Language and Vision to Generate Natural Language Descriptions of Videos in the Wild. In *Coling*, Vol. 2, 5, 9.
24. S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko. 2014. Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729*, (2014).
25. L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville. 2015. Describing videos by exploiting temporal structure. In *IEEE ICCV*.
26. S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko. 2015. Sequence to sequence-video to text. In *IEEE ICCV*.
27. H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu. 2016. Video paragraph captioning using hierarchical recurrent neural networks. In *IEEE CVPR*.
28. H. Xu, S. Venugopalan, V. Ramanishka, M. Rohrbach, and K.Saenko. 2015. A multi-scale multiple instance video description network. *arXiv preprint arXiv:1505.05914*, (2015).
29. S. Venugopalan, L. A. Hendricks, R. Mooney, and K. Saenko. 2016. Improving LSTM-based video description with linguistic knowledge mined from text. *arXiv preprint arXiv:1604.01729*, (2016).
30. Y. Liu and Z. Shi. 2016. Boosting video description generation by explicitly translating from frame-level captions. In *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 631-634.
31. P. Pan, Z. Xu, Y. Yang, F. Wu, and Y. Zhuang. 2016. Hierarchical recurrent neural encoder for video representation with application to captioning. In *IEEE CVPR*.
32. N. Ballas, L. Yao, C. Pal, and A. Courville. 2015. Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*, (2015).
33. Y. Pan, T. Mei, T. Yao, H. Li, and Y. Rui. 2016. Jointly modeling embedding and translation to bridge video and language. In *IEEE CVPR*.
34. Z. Gan, C. Gan, X. He, Y. Pu, K. Tran, J. Gao, L. Carin, and L. Deng. 2017. Semantic Compositional Networks for visual captioning. In *IEEE CVPR*.

35. X. Zhang, K. Gao, Y. Zhang, D. Zhang, J. Li, and Q. Tian. 2017. Task-Driven Dynamic Fusion: Reducing Ambiguity in Video Description. In IEEE CVPR.
36. L. Baraldi, C. Grana, and R. Cucchiara. 2017. Hierarchical Boundary-Aware Neural Encoder for Video Captioning. In IEEE CVPR
37. Y. Chen, S. Wang, W. Zhang, and Q. Huang. 2018. Less Is More: Picking Informative Frames for Video Captioning. arXiv preprint arXiv:1803.01457, (2018).
38. J. -C. Lin and C. -Y. Zhang, "A New Memory Based on Sequence to Sequence Model for Video Captioning," 2021 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC), 2021, pp. 470–476, <https://doi.org/10.1109/SPAC53836.2021.9539903>.
39. Yao, L., Torabi, A., Cho, K., Ballas, N., Pal, C., Larochelle, H., Courville, A. (2015). Describing videos by exploiting temporal structure. Proceedings of the IEEE International Conference on Computer Vision, 4507–4515.
40. Li, W., Guo, D., Fang, X. (2017). Multimodal architecture for video captioning with memory networks and an attention mechanism. Pattern Recognition Letters, 105, 23–29.
41. N. Yadav and D. Naik, "Generating Short Video Description using Deep-LSTM and Attention Mechanism," 2021 6th International Conference for Convergence in Technology (I2CT), 2021, pp. 1–6, <https://doi.org/10.1109/I2CT51068.2021.9417907>.
42. X. Wang, W. Chen, J. Wu, Y. Wang, and W. Y. Wang. 2017. Video Captioning via Hierarchical Reinforcement Learning. arXiv preprint arXiv:1711.11135, (2017).
43. X. Wu, G. Li, Q. Cao, Q. Ji, and L. Lin. 2018. Interpretable Video Captioning via Trajectory Structured Localization. In IEEE CVPR.

Machine Learning Techniques for Covid-19 Pandemic Updates for Analysis, Visualization, and Prediction System



D. Radha, P. Ratna Kumari, and M. Dhanalakshmi

Keywords Covid-19 · Analyses · Visualize · Forecast machine learning

1 Introduction

The SARS-CoV-2 Covid-19 sickness (coronavirus) started in Wuhan, China, at some point during December 2019. Inside a month, in excess of 10,000 individuals were tainted and hundreds passed on [1]. The underlying episode caused a few passings, as the clinical frameworks were not fit for dealing with numerous truly sick victims. As of Oct 30, 2021, there were 244,897,472 affirmed cases and 4,970,435 passings from coronavirus around the world [2] announced across the world because of this pandemic. In this review, we have fused the standards of information science [3] for the forecast of coronavirus movement (as shown in Fig. 1).

The flare-up of coronavirus is difficult for any administration, with respect to the limit and the executives of general well-being frameworks to confront the cataclysmic crisis [4, 5]. The forecast model can help clinics and medical care board to appropriately assign assets, consequently decreasing the strain and permitting the circumstance to be taken care of with no sweat.

D. Radha (✉) · P. Ratna Kumari · M. Dhanalakshmi
Raghu Engineering College, Department of Computer Science and Engineering, Visakhapatnam,
Andhra Pradesh, India
e-mail: Dhanalakshmi.m@raghuenggcollege.in

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
R. Misra et al. (eds.), *Machine Learning and Big Data Analytics*,
Springer Proceedings in Mathematics & Statistics 401,
https://doi.org/10.1007/978-3-031-15175-0_43

529

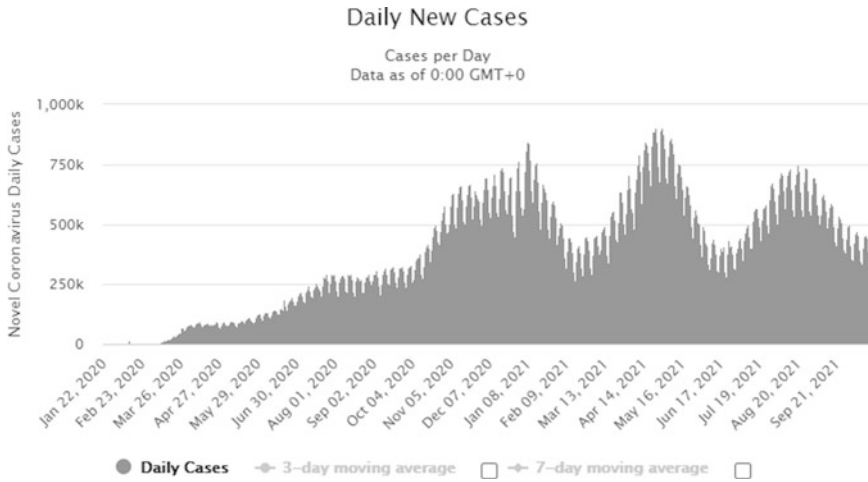


Fig. 1 The daily new cases [6].

2 Related Work

Numerous exploration works have been completed to foresee the episode of coronavirus [7]. Vomlel et al. chipped away from STEMI at the dataset of patients, and various classifiers utilized for expectations were, specifically, calculated relapse, LogitBoost, Choice Tree (CT), NBC, Neural Organizations (NO), and the two adaptations of Bayesian organization classifiers (BOC) [8]. Kumar et al. utilized the autoregressive integrated moving average (ARIMA) method for anticipating the episode in the main 15 European nations.

Tuli et al. [9] presented a machine learning (ML) method that can run constantly on CSF (cloud server farms) for exact forecast of lay out and proactive advancement of key reaction close by the public authority and residents. Great Weibull models fitted well on their dataset instead of plan Gaussian models [10]. Petropoulos et al. presented a genuine system with anticipate continuation of coronavirus by live estimating. They produce ten-days-ahead point theories and suspicion ranges. A weak uncovered convincing recuperated (SEIR) meta-people model was utilized to anticipate the spread across all tremendous metropolitan organizations in China, with 95% solid reaches [11]. Yang et al. utilized the changed SEIR model to prompt the plague wind [12]. They utilized a man-made mindfulness (PC-based knowledge) approach, prepared on the 2003 SARS information, to anticipate the pandemic. Bhatnagar et al. [13] made a numerical model for expecting the spread of coronavirus in nations utilizing different sorts of cutoff points and offered their model a chance genuine information of nations.

A confined Poisson model was consolidated by the force law and the pivotal law as proposed by Zhang et al. [14] to zero in on the coronavirus scenes in six basic western nations. Maier et al. [15] have presented a self-centered model that gets the contaminated people furthermore individuals wide seclusion rehearses by virtue of rule frameworks. Li et al. [16] zeroed in on the transmission association of coronavirus. It utilized forward figure and thus around induction of the plague circumstance and the basic evaluation assisted applicable nations with making more real. Wu et al. [11] expected the conjecture for simply the critical metropolitan networks of China, while Zhang et al. [14] expected for six huge western countries. On the other hand, the proposed strategies guess the wellspring of accuracy is MSE, MAE, and RMSE.

3 Proposed Method

The proposed methods are basically consisting of four three main steps: Dataset preparation, data preprocessing, application of ying classification algorithms as shown in Fig. 2.

3.1 *Strategies for Measuring Utilizing AI*

Not many of the significant determining strategies are as follow:

Linear Regression Straight relapse is maybe one of the most notable and surely known calculations in insights and AI.

Logistic Regression Calculated relapse is another strategy acquired by AI from the field of insights. It is the go-to strategy for twofold grouping (issues with two class esteems). Calculated relapse resembles direct relapse in that the objective is to find the qualities for the coefficients that weight each info variable.

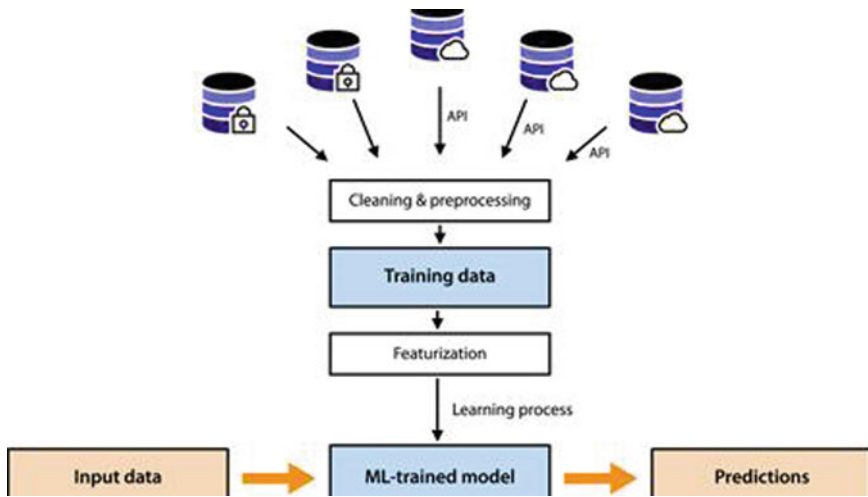


Fig. 2 Preprocessing model.

Linear Discriminant Examination Strategic relapse is a grouping calculation customarily restricted to just two-class characterization issues. On the off chance that you have multiple classes, the direct discriminant investigation calculation is the favored straight arrangement procedure.

SVM (Support-Vector Machines) SVMs (support-vector machines) [17] are perhaps one of the most notable and examined AI computations. A hyper plane is a line that parts the data variable space. In SVM, a hyper plane is picked to best separate the concentrations in the data variable space by their gathering, either class 0 or class 1. In two estimations, you can envision this as a line and what about we expect that all of our criticism centers can be completely disconnected by this line. The SVM learning estimation finds the coefficients' results in the best division of the classes by the hyper plane. The overall module depiction flowchart is shown in Fig. 3.

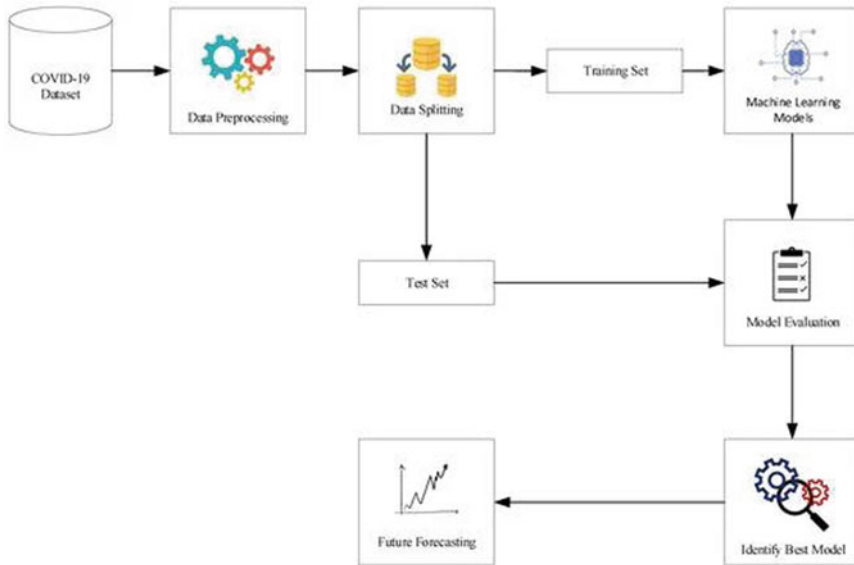


Fig. 3 Module description flowchart.

Problem Statement

Designing a Covid-19 analysis, visualization, and prediction system as it helps to take necessary precautions in the future.

3.2 Proposed Algorithms

SVR (Support-Vector Regressor)

SVR is a powerful algorithm that allows us to choose how tolerant we are of errors, both through an acceptable error margin (ϵ) and through tuning our tolerance of falling outside that acceptable error rate. Our original training dataset for every country was stated in a finite-dimensional state and so the sets to discriminate were not linearly separable in that space. To resolve this problem, our original finite-dimensional state was mapped into a higher-dimensional space. By doing this, we could find the prediction of different countries in a nonlinear approach. The model is defined as a comprehensive evaluation of the gram matrix along with the predictors $x(i)$ and $x(j)$. The gram matrix is a $n \times n$ dimensional matrix that contains the elements $g(i, j)$. The process comprises obtaining a nonlinear SVM regression model by replacing the dot product of the predictors with a nonlinear kernel function comprising $G(x_1, x_2)$ as $\alpha(x_1)$ and $\alpha(x_2)$, where $\alpha(x_1)$ comes out to be greater than $G(x_1, x_2)$, and $\alpha(x_2)$ comes out to be less than the function modeled. Some regression problems cannot be described using a linear model; we need nonlinear

models. A nonlinear SVM regression model can be obtained by replacing the dot product $x_1' x_2$ with a nonlinear kernel function $G(x_1, x_2) = \langle \alpha(x_1), \alpha(x_2) \rangle$, where $\alpha(x)$ is a transformation that maps x to a high-dimensional space. Statistics and Machine Learning Toolbox provides the following built-in semi-definite kernel functions. The kernel function of linear dot product is as shown:

$$G(x_j, x_k) = x_1' x_2 \quad (1)$$

The kernel function of Gaussian is the following:

$$G(x_j, x_k) = \exp\left(-\left|(x_i - x_k)^2\right|\right) \quad (2)$$

The kernel function of polynomial is the following:

$$G(x_j, x_k) = (1 + x_1' x_2)^q \quad (3)$$

where q is in $\{2, 3, \dots\}$.

Algorithm

Step-1: Bringing in the libraries

Step-2: Perusing the dataset

Step-3: Component scaling

Step-4: Fitting SVR to the dataset

Step-5: Foreseeing another outcome

Step-6: Imagining the SVR results (for higher goal and smoother bend)

Linear Regression (LR)

It is a statistical approach for modeling the relationship between a dependent variable and a given set of independent variables. All the values in the dataset were plotted. After plotting the points, we created the best-fit line. A best-fit line is the one that minimizes the error, i.e., it should have a minimum difference between the actual and predicted values. We found the slope of the line and also its y-intercept. After getting the equation of the line, we were able to predict the new values, which is the number of patients in an individual country. The expression for representing a line is given as $y = mx + c$, where “ m ” is the slope. The formula for calculating the slope is

$$m = \frac{\sum (x - \bar{x})(y - \bar{y})}{\sum (x - \bar{x})^2} \quad (4)$$

where \bar{x} and \bar{y} are the mean values.

Random Forest Regressor (RFR)

We used random forest regressor (RFR), as it fits several classifying decision trees. The subsample size was controlled with the `maxsamples` parameter. We loaded

the specific model into our training environment and initiated all the parameters to random values. We got the same result every time we ran the model on the given dataset. Then, we fitted this model on the dataset so that we could easily predict the number of Covid-19 cases on any day using our trained model. The random forest regressor model comprises parameters such as the number of trees and the number of features represented by B and M , respectively. Here, the values of B and M are less than or equal to the dimensional value d . $T(i)$ represents the tree at index i . The tree (i) is constructed in such a way that at each node, a random value from a subset of features is chosen considering splits on those features only:

$$D = ((x_1, y_1), \dots, (x_n, y_n)) \quad (5)$$

where D = observed data point. The parameters are B = number of trees, M = number of features, x_i = d -dimensional vector, $B, M \leq d$, and T_i = tree T .

Polynomial Regression Polynomial relapse is a type of straight relapse where the connection between the free factor A and ward variable B is displayed as the furthest limit polynomial. Polynomial regression fits a nonlinear connection between the worth of A and the relating contingent mean of B , denoted $E(B|A)$ [18].

$$B = c_0 + c_1A_1 + c_2A_1^2 + c_3A_1^3 + \dots + c_nA_1^n$$

Here, B is the reliant variable (yield variable), A_1 is the free factor (indicators) c_0 is the predisposition, and $c_1, c_2, c_3 \dots c_n$ are the loads in the relapse condition.

As the level of the polynomial condition (n) becomes higher, the polynomial condition turns out to be more muddled, and there is plausibleness of the model tending to overfit which will be examined in the later part.

- Comparison of Regression Conditions

Straightforward direct regression $\implies B = c_0 + c_1A$

Various straight regression $\implies B = c_0 + c_1A_1 + c_2A_2 + c_3A_3 + \dots + c_nA_n$

Polynomial regression $\implies B = c_0 + c_1A_1 + c_2A_1^2 + c_3A_1^3 + \dots + c_nA_1^n$

From the over three conditions, we see that there are a few unobtrusive contrasts in them.

Algorithm

Step-1: Import libraries and dataset.

Step-2: Isolating the dataset into two parts.

Step-3: Placing linear regression to the

Step-4: Placing polynomial regression to the dataset

Step-5: In this progression, we are envisioning the straight regression results utilizing dissipate plot.

Step-6: Envisioning the polynomial regression results utilizing a disperse plot.

Step-7: Here, foreseeing new outcome with both straight and polynomial regression.

Bayesian Ridge Regression: In the Bayesian viewpoint, we characterize direct backslide using probability scatterings instead of point measures [19]. The response, y , isn't evaluated as a single worth but is believed to be drawn from a probability dissemination. The model for Bayesian direct relapse with the response analyzed from a normal allocation is as follows:

$$y \sim N(\beta^T X, \sigma^2 I)$$

Here, yield, y , is made from a general (Gaussian) circulation depicted by a mean and change. The back probability of the model limits is dependent upon the readiness data sources and yields:

$$P(\beta|y, X) = \frac{P(y|\beta, X) * P(\beta|X)}{P(y|X)}$$

Here, $P(\beta|y, X)$ is the back probability course of the model limits given the information sources and yields. This is identical to the likelihood of the data, $P(y|\beta, X)$, improved by the prior probability of the limits and confined by a normalization predictable. This is an essential explanation of Bayes hypothesis, the chief supporting of Bayesian derivation:

$$Posterior = \frac{Likelihood * Prior}{Normalization}$$

Priors On the off chance that we have space knowledge, or a theory for what the model boundaries ought to be, we can remember them for our model, dissimilar to the frequent approach which expects all that there is to know about the boundaries comes from the information. On the off chance that we don't have any early appraisals, we can utilize noneducational priors for the boundaries like a normal dissemination.

Posterior The aftereffect of performing Bayesian direct relapse is an appropriation of conceivable model boundaries dependent on the information and the earlier. This permits us to evaluate our vulnerability about the model: in the event that we have less information focuses, the back conveyance will be more fanned out.

4 Experimental Results

The strategy was prepared on Windows 10 working framework with an Intel i3 processor and 8 GB Slam. The dataset is accumulated from numerous information sources—a few credible government sites [17] and for time-series gauging [20]. All AI models were prepared on Google Colab and Numpy, Matplotlib, and Pandas were utilized to redesign a unique guide to envision how the pandemic infection is spreading around the world, country-wise. The dynamic map will show how it varies from latest, last upgrade, conformed, deaths, recovered, and active cases with the help of legend as shown in Figs. 4 and 5.

FIPS	Admin2	Province_State	Country_Region	Last_Update	Lat	Long_	Confirmed	Deaths	Recovered	Active
0	NaN	NaN	Afghanistan	2021-06-10 04:23:17	33.93911	67.709953	84050	3305	59871.0	20874.0
1	NaN	NaN	Albania	2021-06-10 04:23:17	41.15330	20.168300	132415	2452	129761.0	202.0
2	NaN	NaN	Algeria	2021-06-10 04:23:17	28.03390	1.659600	132034	3544	91894.0	36596.0
3	NaN	NaN	Andorra	2021-06-10 04:23:17	42.50630	1.521800	13791	127	13569.0	95.0
4	NaN	NaN	Angola	2021-06-10 04:23:17	-11.20270	17.873900	36115	811	29553.0	5751.0

Fig. 4 Dynamic map of varies latest and longitude, last upgrade, conformed, deaths, recovered, and active cases.

10/15/21	10/16/21	10/17/21	10/18/21	10/19/21	10/20/21	10/21/21	10/22/21	10/23/21	10/24/21	10/25/21	10/26/21
2.790000e+02	2.790000e+02	2.790000e+02	2.790000e+02	2.790000e+02	2.790000e+02	2.790000e+02	2.790000e+02	2.790000e+02	2.790000e+02	2.790000e+02	2.790000e+02
8.604745e+05	8.616667e+05	8.627468e+05	8.642734e+05	8.658549e+05	8.675312e+05	8.691704e+05	8.708220e+05	8.722379e+05	8.733741e+05	8.748200e+05	8.763266e+05
3.768453e+06	3.771372e+06	3.773530e+06	3.773917e+06	3.785052e+06	3.790715e+06	3.795996e+06	3.802042e+06	3.804906e+06	3.807094e+06	3.812767e+06	3.816698e+06
0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
2.695500e+03	2.698000e+03	2.701000e+03	2.703500e+03	2.707000e+03	2.710000e+03	2.711000e+03	2.712500e+03	2.714500e+03	2.715000e+03	2.716000e+03	2.717000e+03
3.836500e+04	3.870500e+04	3.908500e+04	3.948800e+04	3.983900e+04	3.983900e+04	4.008400e+04	4.008400e+04	4.023800e+04	4.023800e+04	4.043300e+04	4.043300e+04
3.809530e+05	3.811120e+05	3.816115e+05	3.816730e+05	3.822820e+05	3.838855e+05	3.855690e+05	3.871720e+05	3.888345e+05	3.901655e+05	3.911750e+05	3.925050e+05
4.488460e+07	4.491674e+07	4.493436e+07	4.505091e+07	4.513215e+07	4.522090e+07	4.530104e+07	4.540047e+07	4.542746e+07	4.544426e+07	4.554716e+07	4.560902e+07

Fig. 5 First 5 rows of the latest recorded cases dataset.

	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20
count	277.000000	277.000000	279.000000	279.000000	279.000000	279.000000
mean	20.305266	22.500100	1.996416	2.347670	3.372760	5.139785
std	25.206138	75.408236	26.637656	26.735265	33.284971	46.326319
min	-51.796300	-178.116500	0.000000	0.000000	0.000000	0.000000
25%	4.860416	-23.041800	0.000000	0.000000	0.000000	0.000000
50%	21.521757	20.939400	0.000000	0.000000	0.000000	0.000000
75%	40.463667	85.240100	0.000000	0.000000	0.000000	0.000000
max	71.706900	178.065000	444.000000	444.000000	549.000000	761.000000

8 rows x 646 columns

Fig. 6 Mean and standard deviation of conformed cases.

Figure 6 shows the mean and standard deviation of conformed cases from the months of 1/22/2020 to 10/26/2021. The mean value of 1/22/2020 is 1.996416 and 8.763266e+05 for 10/26/2021, and the standard deviation value of the month 1/22/2020 is 26.637656 and 3.816698 + 06 for 10/26/2021.

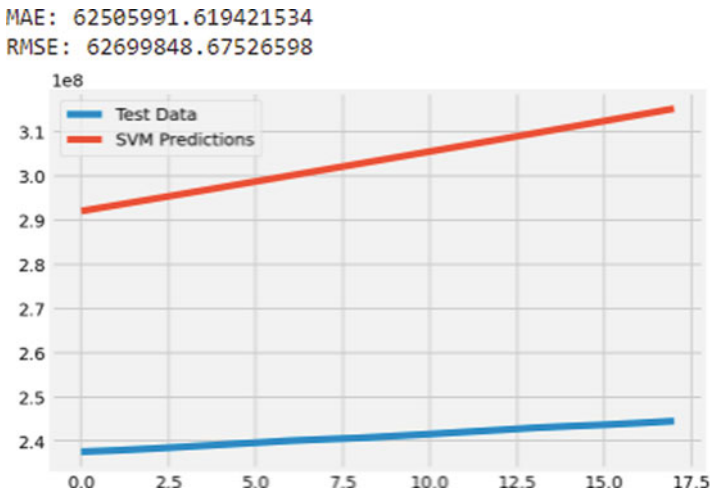


Fig. 7 SVM prediction checking against testing data.

Figure 7 shows that, in the SVM prediction checking against testing data, we can calculate the mean absolute error and root mean squared error (RMSE) or MSE so that the MAE is 62505991.619421534 and RMSE 62699848.67526598. We can see

that the blue color indicates the SVM prediction and the red color indicates the test data of conformed data.

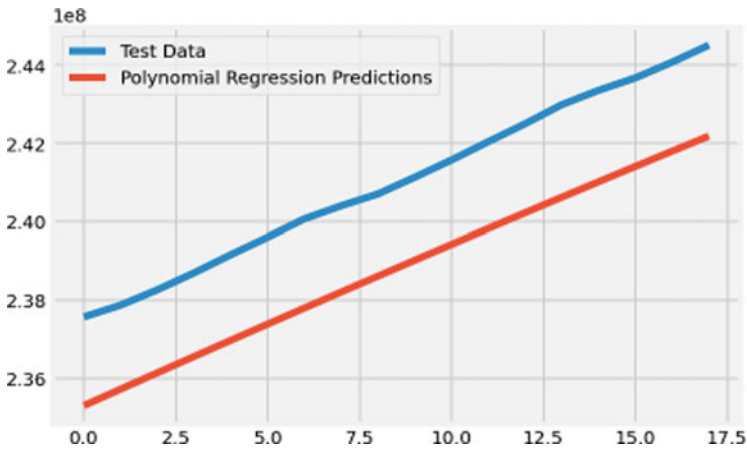


Fig. 8 Polynomial regression prediction.

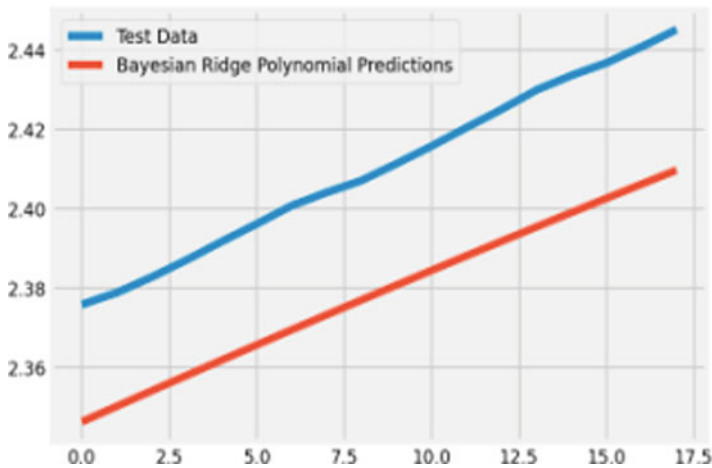


Fig. 9 Bayesian ridge polynomial Checking against testing data prediction checking against testing data.

Figures 8 and 9 show bar graph polynomial regression prediction and Bayesian ridge polynomial performance based on test data and PRP and BRP using the conformed cases. BRP gave the best results and then the PRP on test data.

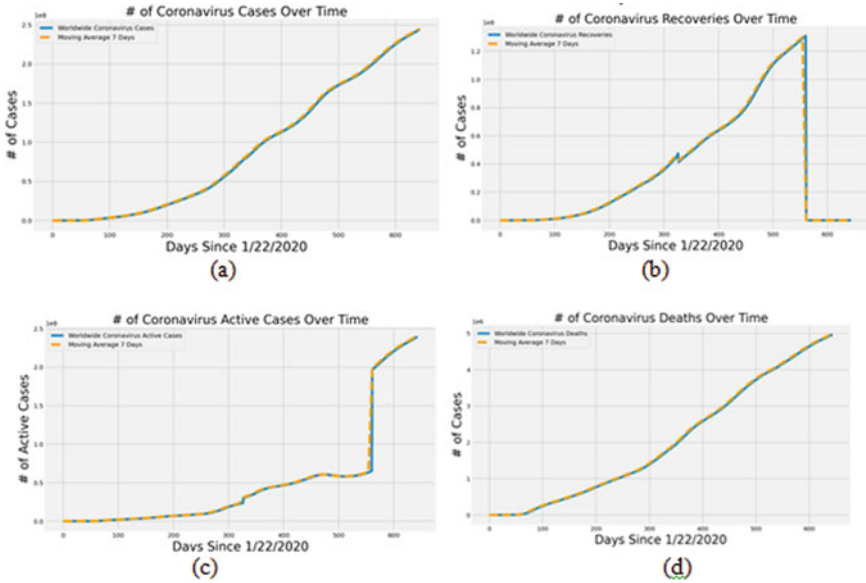


Fig. 10 (a–d) Recoveries, deaths, active cases of covid-19 cases over time.

Figure 10a–d shows that the Covid-19 cases provide a comprehensible graph of the pandemic virus-affected worldwide coronavirus cases till 10/28/2021. We can observe (Fig. 10a–d) the number of recoveries, deaths, and active Covid-19 cases over time, in which the blue color shows the worldwide coronavirus cases and the orange color shows the moving average 7 days, which shows the best results.

Figure 11a–c shows that the Covid-19 cases provide a comprehensible graph of the pandemic virus-affected worldwide coronavirus cases till 10/28/2021.

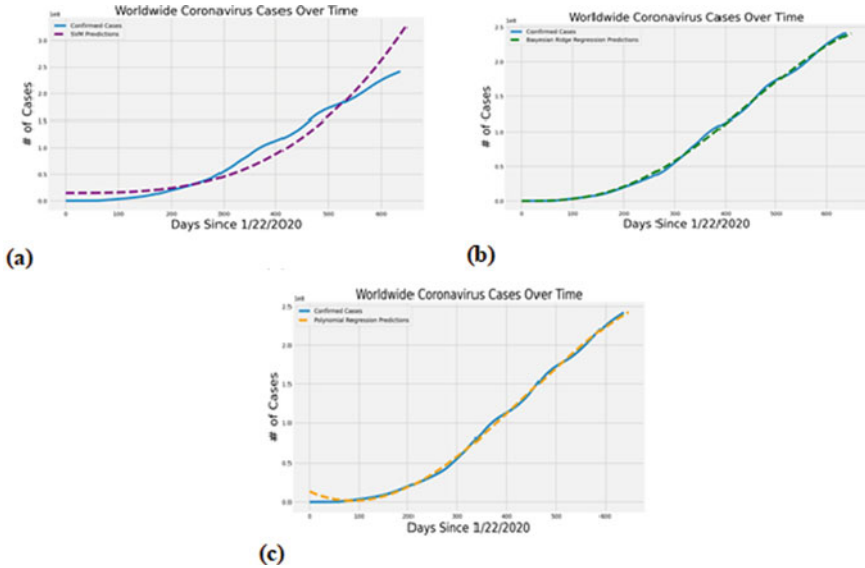


Fig. 11 (a–c) SVM, BRP, and PP worldwide coronavirus cases over time.

We can observe (Fig. 11a–c) the SVM, BRP, and PP prediction throughout the worldwide coronavirus cases over time, in which the blue color shows the conformed cases, the purple color shows SVM predictions, and the orange color shows the PP which shows the best results.

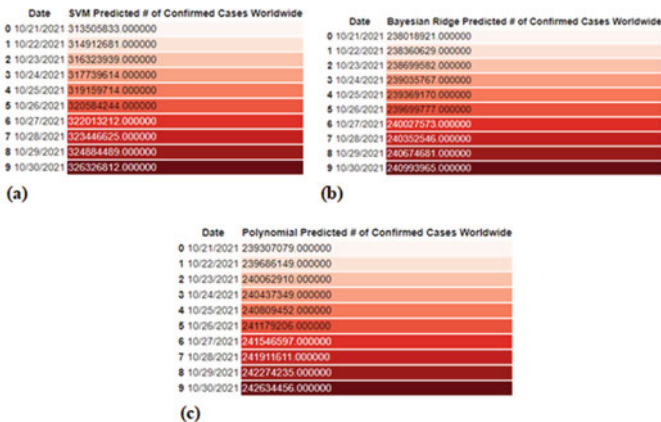


Fig. 12 (a–c) SVM, BRP, and PP of future predictions.

Figure 12a–c shows that the Covid-19 cases provide a comprehensible graph of the pandemic virus-affected worldwide coronavirus cases till 10/28/2021. We

can observe (Fig. 12a–c) the SVM, BRP, and PP number of coronavirus cases worldwide, which shows that the future predictions of ten-day results of SVM, PP and BRP. We can observe that in the three methods of results, BRP gave the best results followed by the SVM and PP.

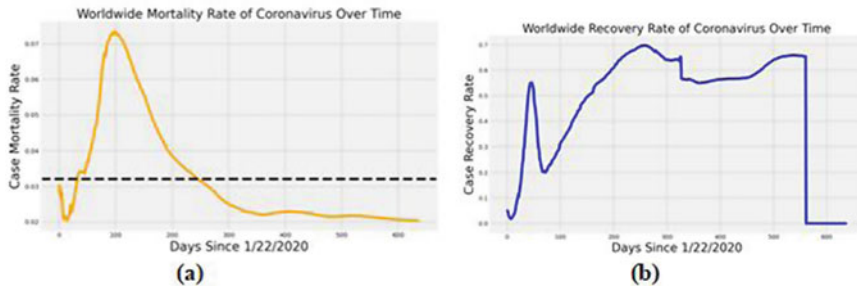


Fig. 13 (a) Worldwide mortality rate of coronavirus over time and (b) recovery rate of coronavirus over time.

Figure 13a and b shows that the Covid-19 cases provide a comprehensible graph of the pandemic virus affected toward an assumption regarding the overall mortality status across the globe. In the worldwide coronavirus cases till 10/28/2021, we can observe the worldwide mortality rate of coronavirus, in which the higher death rate is exhibited one stage, and then slowly the mortality rate will be decreased, and Fig. 13b shows an overview of the recovery rate all over the world; the recovery rate daily since 1/22/2020 to 10/28/2021 is moving slowly decreasing till 10/28/2021.

5 Conclusion and Future Enhancement

The proposed procedure predicts the absolute number of Covid-19 tainted cases, complete number of everyday new cases, all-out number of passings, and all-out number of day-by-day new passings. The SVR has been accounted for to beat the consistency regarding other straight, polynomial, and calculated relapse models. The fluctuation in the dataset is tended to by the proposed system. The infection spread is essentially high, and on the off chance that appropriate regulation measures with physical separating and hygienist are kept up with, we can lessen the spikes in the dataset and henceforth bring down the pace of movement.

These forecasts help the public authority in different ways. It helps play it safe during the pandemic. It assists with assessing the number of immunizations that must be delivered and the number of beds to be organized and in taking seclusion choices, for example, lock down and check-in time. In future, these predictions can be made more accurately by using deep learning techniques.

Further Enhancements Can Be Made

Based on these estimations, a system can be developed which predicts the number of extra beds required and number of vaccines to be produced designing social distancing trackers. Disease detection system can be done from x-ray scans. The improvements in machine learning, artificial intelligence, convolutional neural networks, and deep learning can also help in the field of medicine for the preparation of vaccines.

References

1. Liu Y Li J, Guo K, Viedma EH, Lee H, Liu J, Zhong N, Gomes LFAM, Filip FG, Fang SC, Özdemir, Gu Z, Xia S, Shi B, Zhou XN, Shi Y, Liu J (2020) What are the underlying transmission patterns of COVID-19 outbreak?—an age-specific social contact characterization. *EClinicalMedicine* 22:10035.
2. Temesgen A, Gurmessa A, Getchew Y (2018) Joint modeling of longitudinal cd4 count and timeto-death of hiv/tb co-infected patients: a case of Jimma university specialized hospital. *Ann Data Sci* 5(4):659
3. Olson DL, Shi Y, Shi Y (2007) Introduction to business data mining, vol 10. McGraw-Hill/Irwin, Englewood Clifs
4. MS et al (2020) Culture vs policy: more global collaboration to effectively combat COVID-19. *Innovation* 1(2):100023
5. <https://www.worldometers.info/coronavirus/worldwide-graphs/>
6. <https://www.worldometers.info/coronavirus/coronavirus-cases/>
7. Vomlel J, Kruzik H, Tuma P, Precek J, Hutyra M (2012) Machine learning methods for mortality prediction in patients with st elevation myocardial infarction. *Proc WUPES* 17(1):204
8. Kumar P, Kalita H, Patairiya S, Sharma YD, Nanda C, Rani M, Rahmani J, Bhagavathula AS (2020) Forecasting the dynamics of COVID-19 pandemic in top 15 countries in April 2020: Arima model with machine learning approach. medRxiv
9. Tuli S, Tuli S, Tuli R, Gill SS (2020) Predicting the growth and trend of COVID-19 pandemic using machine learning and cloud computing. *Internet Things* 11:100222
10. Petropoulos F, Makridakis S (2020) Forecasting the novel coronavirus COVID-19. *PLoS ONE* 15(3):e0231236
11. Wu JT, Leung K, Leung GM (2020) Nowcasting and forecasting the potential domestic and international spread of the 2019-nCoV outbreak originating in Wuhan, China: a modelling study. *Lancet* 395(10225):689
12. Yang Z, Zeng Z, Wang K, Wong SS, Liang W, Zanin M, Liu P, Cao X, Gao Z, Mai Z et al (2020) Modified SEIR and AI prediction of the epidemics trend of COVID-19 in China under public health interventions. *J Thorac Dis* 12(3):165
13. Bhatnagar MR (2020) Covid-19: mathematical modeling and predictions, submitted to ARXIV. <http://web.iitd.ac.in/~manav/COVID.pdf>
14. Zhang X, Ma R, Wang L (2020) Predicting turning point, duration and attack rate of COVID-19 outbreaks in major western countries. *Chaos Solitons Fractals* 135:109829
15. Maier BF, Brockmann D (2020) Effective containment explains subexponential growth in recent confirmed COVID-19 cases in China. *Science* 368(6492):742
16. Li L, Yang Z, Dang Z, Meng C, Huang J, Meng H, Wang D, Chen G, Zhang J, Peng H et al (2020) Propagation analysis and prediction of the COVID-19. *Infect Dis Model* 5:282
17. Carrieri, V., Lagravinese, R., & Resce, G. (2021). Predicting vaccine hesitancy from area-level indicators: A machine learning approach. medRxiv

18. Ritonga, M., Al Ihsan, M. A., Anjar, A., & Rambe, F. H. (2021, February). Sentiment analysis of COVID-19 vaccine in Indonesia using Naïve Bayes Algorithm. In IOP Conference Series: Materials Science and Engineering (Vol. 1088, No. 1, p. 012045). IOP Publishing.
19. Analysis, visualization and prediction of COVID-19 pandemic spread using machine learning By, Sen, S.; Thejas, B. K.; Pranitha, B. L.; Amrita, I.
20. https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_deaths_global.csv

Design and CFD Analysis of Drone Thrust with Duct



K. S. S. Gurudatta, V. Harikiran, M. V. D. K. Raju, B. Rama Krishna, and I. Jagadeesh

Keywords Duct · Thrust force and moment · potential improvements of UAV · Aerodynamics

1 Introduction

Unmanned aerial vehicles (UAVs) are being increasingly implemented in a variety of industries such as agriculture, military, transportation, entertainment, and many public services such as emergency response and surveillance [1]. The exponential advances that the electronics field has undergone in the past decade have allowed the technological world to downsize their instruments. This has played an important role in the increasing interest for small-scale UAVs and micro aerial vehicles (MAVs), since the instrumentation possibilities are continuously growing and, in turn, expanding the potential of UAV applications [2]. Unmanned vehicles propose several advantages over their manned counterparts. Eliminating the requirements of a human-operated device in order to carry out investigative, rescue, or military aerial missions which are considered to be of high risk is the most notable safety feature of UAVs [3]. The ability to navigate in environments with small available aerial space, such as highly developed metropolitan areas, is a challenge that can be overcome by the significantly small size and the enhanced maneuverability of UAVs.

The majority of small unmanned aerial vehicles (UAVs) implement a multi-rotor configuration to carry out maneuvers such as hover and vertical takeoff and

K. S. S. Gurudatta · V. Harikiran (✉) · M. V. D. K. Raju · B. Rama Krishna
Department of Mechanical Engineering, Avanthi Institute of Engineering and Technology,
Visakhapatnam, Andhra Pradesh, India

I. Jagadeesh
Department of Mechanical Engineering, Avanthi Institute of Engineering and Technology,
Visakhapatnam, Andhra Pradesh, India

Tata Consultancy Services, Mumbai, Maharashtra, India

landing. The multi-rotor propulsion simplifies the control system and increases the maneuverability of the vehicle. However, the multi-rotor propulsion is not as efficient as a fixed-wing vehicle for forward flight. This has limited the range and endurance of small multi-rotor UAVs. The forward flight of a multi-rotor vehicle is typically achieved by tilting the rotor disk (and the vehicle) to project a component of the propeller thrust in the direction of motion. The rotor will operate with its disk data nonzero angle of attack, α_P , with respect to the free-stream velocity. The projection of the free-stream velocity perpendicular and parallel to the rotor disk results in variation of the effective angle of attack, α_b , and the generated force of the blade elements. As a result, the aerodynamic performance of the rotor deviates from a conventional propeller operating with its rotor disk perpendicular to the free-stream velocity. In extreme conditions, the variation of α_b can cause flow separation over the propeller disk and reduce the net thrust. The load nonuniformity over the propeller disk can also result in yaw, pitch, or rolling moments on the vehicle, affecting its stability. An understanding of the aerodynamic performance of small propellers at a wide range of disk angles is required for the design of efficient and stable UAVs. A UAV performs a variety of maneuvers in a single flight mission, which requires operation of its rotor at wide range of angle-of-attack covering $90^\circ \leq \alpha_P \leq 0^\circ$.

2 Literature Survey

2.1 Classical Momentum Theory

Newton's second law of motion states that the force exerted on a body, or fluid, is equal to the product between its mass, m , and the acceleration of the body, \vec{a} , such that:

$$\vec{F} = m \cdot \vec{a} \quad (1)$$

The right-hand side of Eq. (1) may be expressed more generally as the time rate of change of momentum, written as:

$$\vec{F} = \frac{d}{dt} (m \cdot \vec{V}) \quad (2)$$

where \vec{V} is the velocity of the body or fluid. The left-hand side of Eq. (2) represents the forces exerted on the fluid. The surfaces forces in \vec{F} are pressure, p , and shear stress acting on the differential control surface dS of the fluid. The body forces, F_{body} , in \vec{F} are forces such as gravity and electromagnetic forces, which act

upon the mass enclosed by the differential volume $d\vartheta$ and viscous forces F_{vis} . Substituting these into Eq. (2), the following result is obtained:

$$-\iint p \cdot dS + \iiint \rho F_{body} \cdot d\vartheta + F_{vis} = \frac{\mu}{dt} (m \cdot v) \quad (3)$$

where ρ is the density of the fluid. The right-hand side of Eq. (3) is composed of two terms, one of which represents the net flux of momentum across the control surface dS expressed as:

$$\iint (\rho \vec{V} \cdot dS) \vec{V} \quad (4)$$

Also, a second one, which represents the time rate of change of momentum due to fluctuations of the properties of the flow within the volume $d\vartheta$, is expressed as:

$$\frac{u}{\partial t} \iiint \rho V d\vartheta \quad (5)$$

Therefore, substituting these two expressions into Eq. (3), the momentum equation is written in its integral form as:

$$-\iint p \cdot ds + \iiint \rho \vec{F}_{body} \cdot d\vartheta + \vec{F}_{vis} = \frac{u}{\partial t} \iiint \rho \vec{V} d\vartheta + \iint (\rho \vec{V} \cdot dS) \vec{V} \quad (6)$$

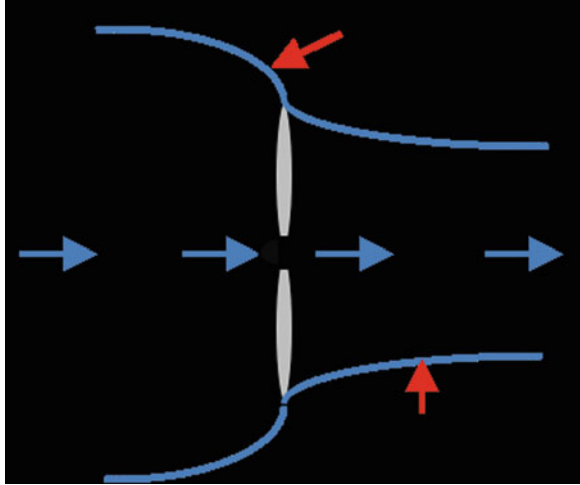
For a propulsion unit such as a propeller, the net contribution of the terms in the left-hand side of Eq. (6) can be conglomerated as a single resulting value of thrust, T , which may be positive or negative depending on the operating state of the propeller, thus simplifying Eq. (6) to:

$$T = \frac{\partial}{\partial t} \iiint \rho V d\vartheta + \iint (\rho V \cdot dS) V \quad (7)$$

The application of the momentum theorem to the performance of propellers assumes steady-state operation rather than the periodicity experienced by the individual blades. Therefore, the time derivative term in Eq. (7) vanishes, and Eq. (6) reduces to:

$$T = \iint (\rho \vec{V} \cdot dS) \vec{V} \quad (8)$$

Fig. 1 Generic flow across an open propeller



A generalized schematic of the flow across an open propeller is shown in Fig. 1. The stage denoted by the subscript ∞ corresponds to a location far upstream from the rotor disk plane, where the pressure p_∞ is equal to the atmospheric pressure, and the velocity V_∞ is the free stream velocity (velocity at which the rotor disk is travelling at). Stages 1 and 2 are immediately upstream and downstream of the propeller plane, respectively. Stage 3 is located far downstream of the rotor disk.

The velocity at the rotor disk, V_1 , is equal to the free stream velocity increased by the induced velocity v_i and is discussed with further detail in this thesis. Since Stage 2 is immediately downstream, $V_2 = V_1$. Similarly, the flow velocity experiences a change in magnitude once more in Stage 3, as the stream tube of the exit jet undergoes contraction.

In summary:

$$V_0 = V_\infty \quad (9)$$

$$V_1 = V_2 = V_\infty + v_i \quad (10)$$

$$V_3 = V_\infty + v_j \quad (11)$$

The principle of conservation of mass indicates that the mass flow rate must be constant at stages 0, 1, 2, and 3. Therefore, the following equality applies:

$$\dot{m} = \rho V_0 A_0 = \rho V_1 A_1 = \rho V_3 A_3 \quad (12)$$

Integrating Eq. (8) between stages 3 and 0, the following is obtained:

$$\frac{3}{0} = \iint \left(\rho \vec{V} \cdot dS \right) \vec{V} = \dot{m} (V_\infty + v_j) - \dot{m} V_\infty = \dot{m} v_j \quad (13)$$

where now the only variable required to compute the thrust generated is the fluid velocity increment at the exit jet. Evaluating the change of kinetic energy between stages 0 and 3, the ideal power P_i transmitted to the fluid is equal to the change in kinetic energy per unit time of the fluid, such that:

$$P_i = \Delta KE = \frac{1}{2} \dot{m} (V_\infty + v_j)^2 - \frac{1}{2} \dot{m} V_\infty^2 = \frac{1}{2} \dot{m} v_j (2V_\infty + v_j) \quad (14)$$

From a disk actuator model of the propeller, the same ideal power is expressed as the product of the inlet velocity and the thrust produced as:

$$P_i = T (V_\infty + v_i) \quad (15)$$

Then, by equating both expressions for the ideal power, namely, Eqs. (14) and (15),

$$\frac{1}{2} \dot{m} v_j (2V_\infty + v_j) = T (V_\infty + v_i) \quad (16)$$

thus, substituting Eq. (13) into Eq. (16), a relationship between the induced velocity at the rotor disk and the velocity increment at the exit jet is established as:

$$v_j = 2v_i \quad (17)$$

Using Eq. (17) along with the mass conservation, the mass flow rates at stages 3 and 1 are equated, such that:

$$(V_\infty + v_i) A_1 = \rho (V_\infty + v_j) A_3. \quad (18)$$

$$\sigma = \frac{A_3}{A_1} = \frac{(V_\infty + v_j)}{(V_\infty + 2v_j)} \quad (19)$$

The expression for the exit area-to-rotor disk ratio is defined as substituting Eq. (17) into equation Eq. (19):

$$\sigma = \frac{(V_\infty + v_j)}{(V_\infty + 2v_j)} \quad (20)$$

For an open propeller in axial (or climb) flight, the value of σ from Eq. (20) is between 0.5 and 1, according to Paragraph [4].

2.2 Theoretical Background on Ducted Propellers

Ducted propellers as a mean of propulsion present an alternative to the conventional open propellers and have been considered in many designs in the aeronautical and marine fields in the past. A ducted propeller can potentially increase the thrust output, T , when compared to an open propeller of the same rotor disk area and with the same power input, P . The performance benefits achieved by using a duct around the propeller are due to the manner in which the presence of the duct influences and changes the flow field generated by the propeller's rotational motion, or, as stated by McMahon [5]: "the function of the duct is to set the operating point of the rotor." Placing a duct around the propeller decreases the pressure at the leading edge of the duct and consequently allows the rotor disk to draw in more air [6] and increase the flowrate through the rotor disk plane. This in turn decreases the effective angle of attack of the propeller blades α_b [6–8] as shown in Fig. 1, thus off-loading the rotor disk, which means reducing the contribution from the propeller to the total thrust of the ducted propeller system. Decreasing the effective angle of attack of the blades also increases the advance ratio limit at which the blade would begin to stall. The increased mass flow rate over the leading edge of the duct generates a section of the duct where low pressure suction forces are present, providing the propulsion system with an increment in thrust [4, 6].

One of the primary mechanisms through which ducted propellers become advantageous is their potential to reduce the contraction of the exit jet flow observed in open propellers [4, 6, 7], caused by the increase in the kinetic energy of the flow when it crosses the rotor disk plane. Momentum theory along with the disk actuator model predicts that the increase in the wake velocity of an open propeller v_j is twice the induced velocity of the flow at the rotor disk plane v_i :

$$v_j = 2v_i \quad (21)$$

thus, due to the principle of mass conservation, the cross-sectional area of the exit jet A_j must be $\frac{1}{2}$ of that at the rotor disk. In the case of the ducted propeller, the exit jet area is dictated by the geometry of the duct [4, 8], and if the flow is able to remain attached to the internal surface of the duct, the exit jet area may even increase with respect to the rotor disk area, resulting in a decrease in the exit jet velocity and, thus, a decrease in the energy expenditure required to produce thrust [4, 9]. This concept

$$\sigma = \frac{(V_{\infty} + v_j)}{(V_{\infty} + 2v_j)}$$

Fig. 2 The schematic provides an explanation of the effects of using a duct around a propeller. The left case indicates the open propeller, and the right case is the ducted propeller. The figure illustrates that the implementation of the duct accelerates the inlet flow V_a at the rotor disk plane resulting in a reduction of α_b and reduces the exit jet velocity V_j by increasing A_j

is illustrated in Fig. 2. Momentum theory predicts that the ratio σ in axial (or vertical climb) is obtained through the following expression:

The placement of a duct around the propeller imposes a physical barrier that helps in the mitigation of the effects from the three-dimensional tip vortices emanated from the flow leaking at the blade tips as it was previously discussed [5, 10, 11]. Reducing the amount of energy that is wasted through the generation of these vortices, and permitting the propeller blades to be loaded even more toward the blade tips, improving the efficiency of the propeller. In addition to the performance benefits, using a ducted propeller can attenuate the noise emanated from the open rotor [4, 12] and serves as a safety feature by protecting personnel operating near the vicinity of the rotor, as well as the equipment itself [4].

The benefits of ducted propellers are well-known when the system is operating at the hover condition where, theoretically, momentum theory predicts that a ducted propeller with $\sigma = 1$ in hover can potentially increase the thrust output by 26% when compared to an open propeller of the same size and same power consumption. However, in axial and forward flight, the benefits of the duct may diminish as the free stream velocity increases. One reason for this loss in performance is due to the decreased ability of the duct to reduce the slipstream contraction. As the free stream velocity increases, the contraction of the slipstream from open propellers diminishes, and the ducted propellers lose their ability to accelerate the flow and decrease the contraction of the slipstream.

In addition to this loss in performance, the benefits achieved by using a duct come with the complication of the weight addition by the duct which reduces the payload of the aircraft, the drag created by the duct, and the increment in structural complexity of the aircraft. Therefore, the majority of the applications that use ducted propellers have been dedicated to improve the hovering flight conditions, but scarce efforts have been made to enhance the performance of ducted propellers in axial or climbing flight.

3 Mathematical Modelling

In this section, the simulation setup utilized to perform the investigation with the ducted propeller is described. All of the experiments were carried out in ANSYS Fluent 2021R1. The governing equations of the flow and the CFD solution method will be presented. The geometry creation and the mesh generation are also reported.

Computational fluid dynamic (CFD) is a robust tool for the analysis of systems comprising external or internal fluid flow and heat transfer. This analysis is made by computer-based simulation with higher accuracy. This method has been extended to a wide range of applications, such as aerodynamics, combustion, environment, medicine, agriculture, and many other applications.

The governing equations of fluid flow describe the mathematical formulations of the laws of conservation of physics.

The equations of laws of conservations can be summarized as:

- Conservation of mass
- Conservation of momentum

3.1 Continuity Equation

3.1.1 Continuity Equation

The continuity equation resulted from applying the conservation of mass law to the fluid flow. The analysis is made in an infinitesimal control volume; the form of the equation is expressed as:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho V) = 0 \quad (22)$$

where ρ is the density of the fluid; R is the velocity of the fluid.

The rate of the increase in the density is represented by the first term in Eq. (22), and the rate of mass flux per unit volume passing out the control surface is represented in these continuity terms. For simplicity and using Lagrangian approach and by applying substantial, Eq. (22) can be written as:

$$\frac{D\rho}{Dt} + \rho (\nabla \cdot V) = 0 \quad (23)$$

For incompressible flows, the first term in Eq. (23) becomes zero, and the first term can be mathematically written:

$$\frac{D\rho}{Dt} = 0 \quad (24)$$

The continuity equation can be written as:

$$(\nabla \cdot V) = 0 \quad (25)$$

Alternatively, the Cartesian coordinate system can be simplified to:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial x} + \frac{\partial w}{\partial x} = 0 \quad (26)$$

3.1.2 Momentum Conservation Equation

The momentum equation can be obtained by applying the Newton's second law on a fluid passing an infinitesimal control volume. The form of the equation can take the following:

$$\frac{\partial}{\partial t} (\rho V) + \nabla \cdot \rho V V = \rho f + \nabla \cdot P_{ij} \quad (27)$$

The first term in Eq. (27) denotes the rate of increase of momentum in the control volume. The rate of momentum per unit volume lost by convection is represented by the second term in the equation and can be expressed as:

$$\nabla \cdot \rho V V = \rho V \cdot \nabla V + V (\nabla \cdot \rho V) \quad (28)$$

Substituting this term in Eq. (27), the momentum equation can be simplified to the following form:

$$\rho \frac{DV}{Dt} = \rho f + \nabla \cdot P_{ij} \quad (29)$$

The body force per unit volume is expressed in the first term on the right-hand side. An example of common force is the gravitational force. The surface forces per unit volume are represented in the second term in Eq. (28). These forces are resulted from applying the external stresses on the fluid element.

For Newtonian fluids, the relation between the stress tensor and pressure and velocity can be written as:

$$P_{ij} = -p\delta_{ij} + \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) + \delta_{ij} \mu \frac{\partial u_k}{\partial x_k} \quad i, j, k = 1, 2, 3 \quad (30)$$

whereas

u_1, u_2, u_3 represent the three components of the velocity vector V .

x_1, x_2, x_3 represent the three components of the position vector.

δ_{ij} is the Kronecker delta function ($\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ if $i \neq j$).

μ represents the coefficient of dynamic viscosity.

μ' is the second coefficient of viscosity.

Now the Navier-Stokes equation can be written as:

$$\rho \frac{DV}{Dt} = \rho f - \nabla p + \frac{\partial}{\partial x_j} \left[u \left(\frac{\partial \mu_i}{\partial x_j} + \frac{\partial \mu_j}{\partial x_i} \right) - \frac{2}{3} \delta_{ij} \mu \frac{\partial u_k}{\partial x_k} \right] \quad (31)$$

In the Cartesian coordinate system, the Navier-Stokes equation can be written in three different scalar equations:

$$\begin{aligned} \rho \frac{Du}{Dt} = \rho f_x - \frac{\partial p}{\partial x} + \frac{\partial}{\partial x} \left[\frac{2}{3} \mu \left(2 \frac{\partial \mu}{\partial x} - \frac{\partial v}{\partial y} - \frac{\partial w}{\partial z} \right) \right] + \frac{\partial}{\partial y} \left[\mu \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) \right] \\ + \frac{\partial}{\partial x} \left[\mu \left(\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z} \right) \right] \end{aligned} \quad (32)$$

$$\begin{aligned} \rho \frac{Dv}{Dt} = \rho f_y - \frac{\partial p}{\partial y} + \frac{\partial}{\partial y} \left[\frac{2}{3} \mu \left(2 \frac{\partial v}{\partial y} - \frac{\partial u}{\partial x} - \frac{\partial w}{\partial z} \right) \right] + \frac{\partial}{\partial x} \left[\mu \left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \right) \right] \\ + \frac{\partial}{\partial x} \left[\mu \left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \right) \right] \end{aligned} \quad (33)$$

$$\begin{aligned} \rho \frac{Dw}{Dt} = \rho f_z - \frac{\partial p}{\partial z} + \frac{\partial}{\partial z} \left[\frac{2}{3} \mu \left(2 \frac{\partial w}{\partial z} - \frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} \right) \right] + \frac{\partial}{\partial x} \left[\mu \left(\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z} \right) \right] \\ + \frac{\partial}{\partial y} \left[\mu \left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \right) \right] \end{aligned} \quad (34)$$

For incompressible flow and constant viscosity, Eq. (31) can be simplified to:

$$\rho \frac{DV}{Dt} = \rho f + \nabla p + \mu \nabla^2 V \quad (35)$$

3.2 Calculation of Aerodynamic Characteristics of a Propeller

A propeller creates a thrust force out of the supplied power. The magnitude of this force is not constant for a given propeller but depends on the velocity of the incoming air and the rotational velocity of the propeller itself. Thus, tests of propellers usually cover a wide regime of operating conditions. Propellers having

the same shape but scaled by a size factor behave similar. In order to make a comparison of propellers of different size easier, aerodynamicists try to get rid of the units. Then, it is possible to use the results of a small-scale wind tunnel model to predict the performance of a full-scale propeller. Similar to airfoils and wings, the performance of propellers can be described by dimensionless (normalized) coefficients. While an air foil can be characterized by relations between the angle of attack, lift coefficient, and drag coefficient, a propeller can be described in terms of advance ratio, thrust coefficient, and power coefficient. The efficiency, which corresponds to the L/D ratio of a wing, can be calculated from these three coefficients. These coefficients are helpful for the comparison of propellers of differing diameters, tested under different operating conditions.

Thrust	C_T	$C_T = \frac{T}{\rho \cdot n^2 \cdot D^4}$
Power	C_P	$C_P = \frac{P}{\rho \cdot n^3 \cdot D^5}$
Advance ratio	v/nD	$v/nD = \frac{v}{n \cdot D}$
Efficiency	η	$\eta = \frac{v}{n \cdot D} \cdot \frac{C_T}{C_P}$

where:

v	Velocity	m/s
D	Diameter	M
n	Revolutions per second	1/s
ρ	Density of air	kg/m ³
P	Power	W
T	Thrust	N

4 Simulation Methodology

Computational fluid dynamics (CFD) is a branch of fluid mechanics that uses numerical analysis and data structures to analyze and solve problems that involve fluid flows. Computers are used to perform the calculations required to simulate the free-stream flow of the fluid and the interaction of the fluid (liquids and gases) with surfaces defined by boundary conditions.

The CFD code used in the present study is Ansys Workbench. Fluent CFD code (which works under the Ansys Workbench environment) is a common commercial software package. As discussed earlier in the literature survey, it has been used in many numerical applications and provided comprehensible results by solving 3D Navier-Stokes continuity, momentum, and energy equations. In this study, Fluent was selected as a computational tool to perform the numerical fluid and heat transfer

analysis. Throughout this research, each model was analyzed by three main solution steps, namely, Ansys preprocessing, Ansys solution, and Ansys post-processing.

4.1 Geometry Creation

All geometries for the problems investigated in the present research were created by Ansys Space Claim. Space Claim is a solid modeling CAD (computer-aided design) software that runs on Microsoft Windows and developed by Space Claim Corporation. The Space Claim is an application designed to be employed as a geometry editor for all kind aspects of shapes in 2-D and 3-D problems. The Space Claim is well-known for its flexibility in creating and editing and complex geometry without almost any penalty on the solution accuracy. Space Claim excels at the rapid creation of any 3D geometry.

4.2 Mesh Generation

Mesh generation is the practice of creating a mesh, a subdivision of a continuous geometric space into discrete geometric and topological cells. Often these cells form a simplicial complex. Usually, the cells partition the geometric input domain. Mesh cells are used as discrete local approximations of the larger domain. Meshes

Fig. 3 Isometric view of the propeller

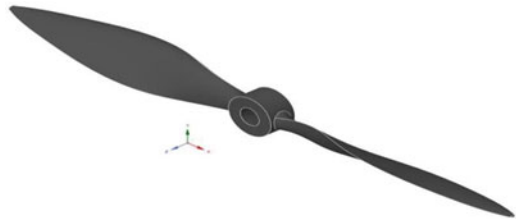


Fig. 4 Propeller with duct

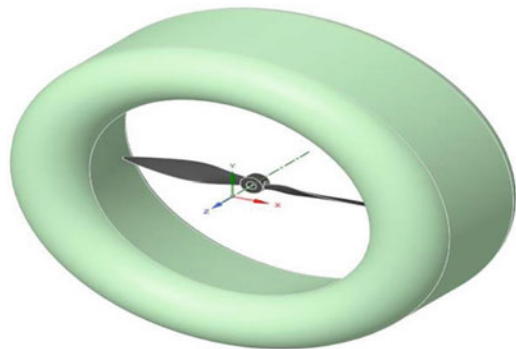


Fig. 5 Propeller with moving reference fluid domain around it

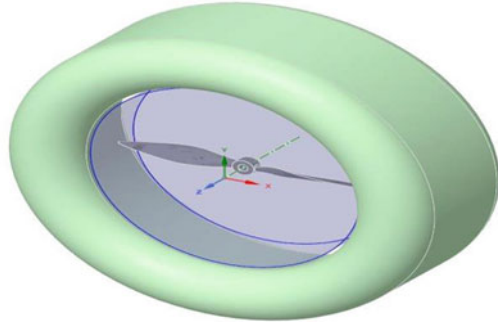


Fig. 6 Propeller with static fluid domain

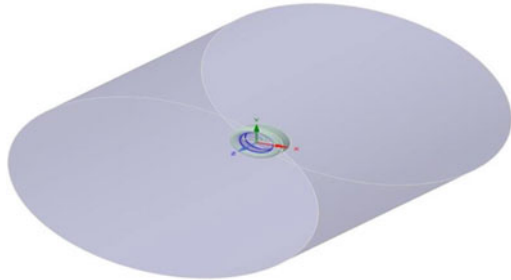


Fig. 7 Fluent meshing has a parallel processing capability which can generate high quality of mesh like mosaic mesh and poly hexcore mesh with a good orthogonal quality and aspect ratio. Surface mesh of MRF generated through Fluent mesher



are created by computer algorithms, often with human guidance through a GUI, depending on the complexity of the domain and the type of mesh desired. The goal is to create a mesh that accurately captures the input domain geometry, with high-quality (well-shaped) cells and without so many cells as to make subsequent calculations intractable. The mesh should also be fine (have small elements) in areas that are important for the subsequent calculations. The solution should not depend on the mesh; in other words, all solutions obtained in this research were tested by the so-called mesh dependency test. In this test, the number of cells is increased to the level where any extra refinement does not affect the solution accuracy (Figs. 7, 8, 9, 10, and 11).

Fig. 8 Surface mesh of propeller with duct generated through Fluent mesher



Fig. 9 Surface mesh of the entire fluid domain generated through Fluent mesher

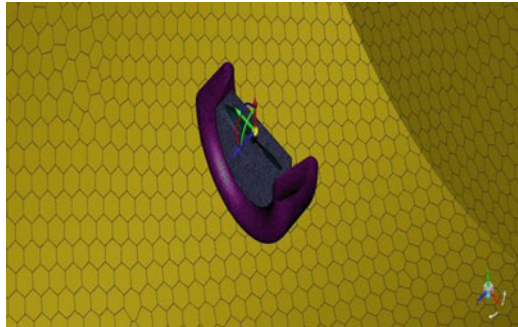


Fig. 10 Volume mesh of the entire fluid domain generated through Fluent mesher

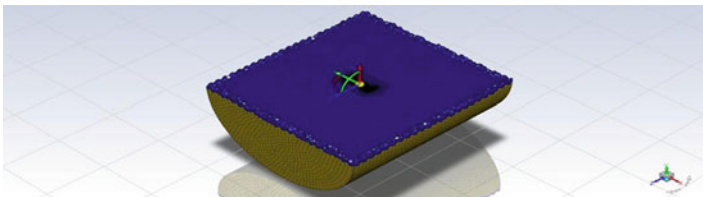
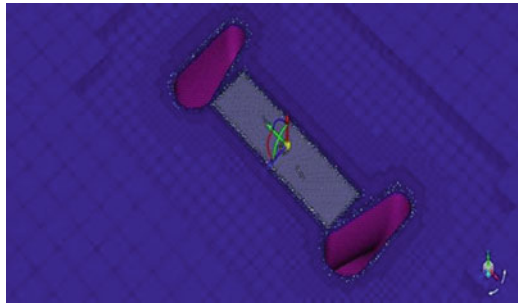


Fig. 11 Volume mesh of the entire fluid domain generated through Fluent mesher

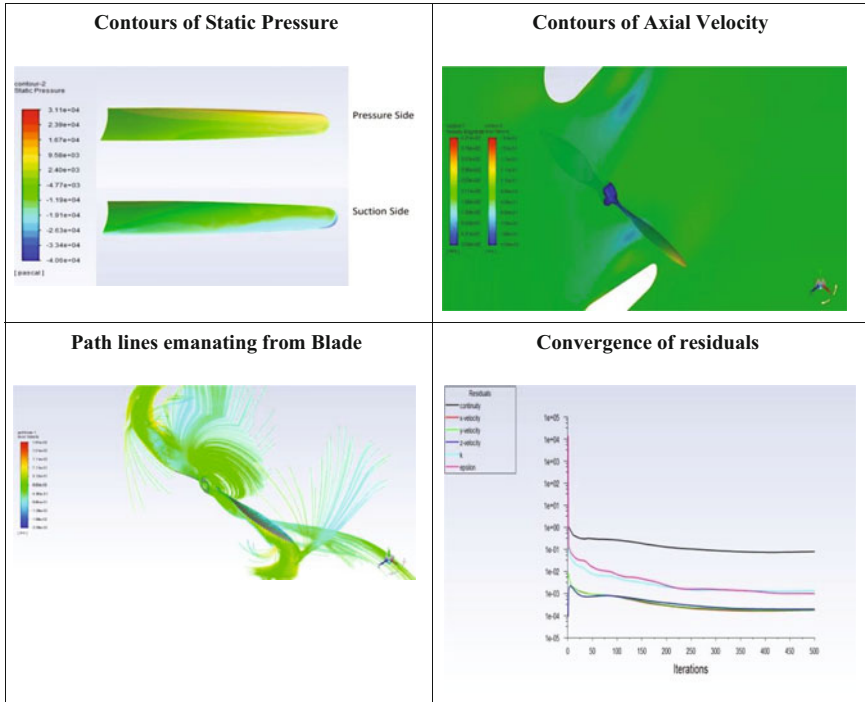


Fig. 12 Contours of static pressure with axial velocity and convergence of residuals

5 Results

The simulations have been performed for the propeller with and without duct as shown in Fig. 12, and contours of pressure, velocity, and path lines have been plotted. Report definitions for thrust force have also been created to observe the convergence of the simulation with respect to iteration progress.

It can be clearly seen that on the pressure side the pressure is high in comparison to the suction side creating a higher thrust force.

It can be seen that the percentage of error in residuals is very small concluding that the simulation is pretty accurate.

It can also be observed that the slope of lift and moment is tending to zero as shown in Fig. 13, i.e., the change in lift and moment with respect to the number of iterations as shown in Table 1 is negligible, and the solution is stable.

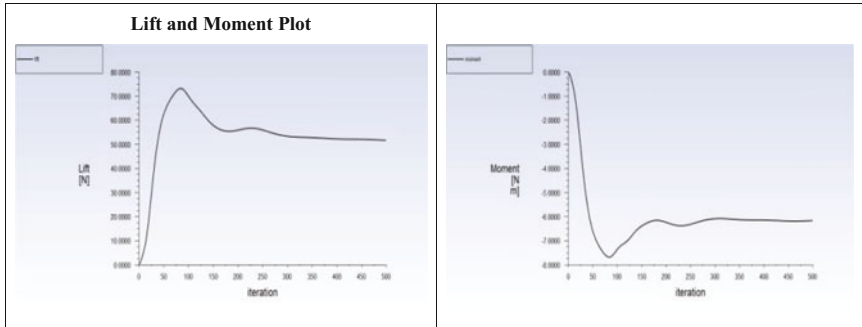


Fig. 13 Slope of lift and moment

Table 1 Comparison of RPM in propeller with and without duct

S. No.	RPM	Propeller		Propeller with duct	
		Lift	Moment	Lift	Moment
1	30,000	52.027	6.0465	54.125	6.535
2	35,000	58.15	6.125	60.25	6.775
3	40,000	62.735	6.4	63.95	6.93

5.1 Comparison of Results with and Without Duct

6 Conclusions

The ducted propeller has shown a higher increase in performance in terms of thrust force even at higher RPMs. It can also be seen that the moment required increases with the introduction of duct. The trade-off between the moment required and thrust increase is within the acceptable specifications of the motor.

References

- Gent,Edd, “The Future of Drones: Uncertain, Promising and PrettyAwesome,,” *LiveScience*, <https://www.livescience.com/52701-future-of-drones-uncertain-but-promising.html>
- Frank, J. McGrew, M. Valenti, D. Levine and J. P. How, “Hover, Transition, and Level Flight Control Design for a Single-Propeller Indoor Airplane,,” AIAA Guidance, Navigation and Control Conference and Exhibit, August2007. <https://doi.org/10.2514/6.2007-6318>.
- S. Siebert, J. Teizer, “Mobile 3D mapping for surveying earthwork projects using an Unmanned Aerial Vehicle (UAV) system,,” *Automation in Construction*, May 2014. <https://doi.org/10.1016/j.autcon.2014.01.004>.
- J. L. Pereira, “Hover and Wind-Tunnel Testing of Shrouded Rotors for Improved Micro Air Vehicle Design,,” Ph.D. Dissertation, Aerospace Engineering Dept., University of Maryland, CollegePark, 2008.
- J. F. McMahon, “Characteristics of Ducted Propellers,,” *Propellers/Shafting '94 Symposium*, 1994.

6. D. M. Black, H. S. Wainauski, and C. Rohrbach, "Shrouded Propellers – A Comprehensive Performance Study," AIAA 5th Annual Meeting and Technical Display, no. 68–994, 1968.
7. R. J. Platt Jr, "Static Tests of a Shrouded and Unshrouded Propeller," NACA RM-L7H25, February 1948.
8. J. D. Van Manen and M. W. C. Oosterveld, "Analysis of Ducted-Propeller Design," Annual Meeting of the Society of Naval Architects and marine Engineers, Vol. 288, New York, pp. 522–562, 1966.
9. S. Yilmaz, D. Erdem, and M. S. Kavsaoğlu, "Performance of a ducted propeller designed for UAV applications at zero angle of attack flight: An experimental study," Aerospace Science and Technology, Vol. 45, pp. 376–386, June 2015. <https://doi.org/10.1016/j.ast.2015.06.005>
10. A. Akturk and C. Camci, "Tip Clearance Investigation of a Ducted Fan Used in VTOL UAVs Part I: Baseline Experiments and Computational Validation," Proceedings of ASME 2011 Turbo Expo: Turbined Technical Conference and Exposition, Vol. 7, pp. 331–344, June 2011. <https://doi.org/10.1115/GT2011-46356>
11. A. Akturk and C. Camci, "Tip Clearance Investigation of a Ducted Fan Used in VTOL UAVs Part 2 : Novel Treatments Via Computational Design," Proceedings of ASME 2011 Turbo Expo: Turbined Technical Conference and Exposition, Vol. 7, pp. 3345–357, Vancouver, June 2011. <https://doi.org/10.1115/GT2011-46359>
12. M. Lazareff, "The Aerodynamics of V/STOL Aircraft," Advisory Group for Aerospace Research and Development, Rhode-Saint Genèse, pp. 237–290, May 1968.

Index

A

Adaptive Synthetic (ADASYN) sampling, 280, 281, 283, 287
Aerodynamics, 546, 552, 554–555
AI models, 366–368, 374, 537
Alex Net, 392
Algorithm, 3, 13, 23, 40, 59, 77, 106, 143, 160, 172, 192, 204, 225, 248, 257, 269, 280, 296, 309, 319, 358, 368, 402, 414, 425, 438, 448, 460, 480, 497, 502, 515, 531, 557
Alzheimer disease (AD), 222, 459–475
Analyses, 1–9, 15–18, 20, 23–34, 38, 42, 43, 46, 47, 50–52, 76, 80, 83–90, 97, 105, 112–113, 120, 124, 131, 134–138, 150, 160, 166–168, 171–188, 191–200, 215, 219–221, 233, 253, 256, 269, 272, 274, 276, 280–282, 287–289, 303, 320, 330, 331, 333, 337, 346, 348, 349, 355–363, 365–368, 370, 371, 380–385, 388, 396–398, 411, 412, 424, 430–433, 436, 439, 441, 447–457, 465, 479, 480, 486, 487, 494, 496, 498, 502–504, 511–513, 520, 523–525, 529–543, 545–560
Analysis of algorithms, 171–188
Arduino, 95, 97–101, 248
Artificial flora, 425
Artificial Intelligence (AI), v, 14, 16, 23, 37–47, 159, 161, 169, 217, 237, 238, 247, 356, 366–368, 374, 378, 379, 384, 401, 406, 447–449, 496, 531–533, 537
Artificial neural network (ANN), 14–17, 24, 38, 40, 45, 46, 163–165, 167, 306, 368, 372, 373, 415, 416, 496, 504, 505

B

BERTSUM, 213–222
Bidirectional Encoder Representations from Transformers (BERT), 214–218, 220, 221, 231, 516, 519, 522, 524, 525
Big data, v, 14, 17, 18, 20, 40, 127–138, 255, 257, 329–331, 401, 473, 505
BiLSTM, 463, 467
Binary classification, 145, 230, 269, 275, 276, 441, 465
Bio-BERT, 213–222
Botnet, 295–306
Botnet detection, 297, 298
Breast cancer, 491–499
Building energy management system, 309

C

CatBoost, 269, 273–276, 285, 286, 288–292
Challenges, 1, 3, 14, 37–47, 49, 60, 61, 76, 90, 96, 111, 162, 194, 203, 225, 248, 249, 264, 309, 341, 377–385, 413, 463, 480, 504, 545
Chatbot, 159–169
CityLearn, 309, 310
Classifier, 15, 16, 52, 149, 156, 160, 163, 172, 173, 175, 176, 178–188, 197, 217, 250, 275, 280, 282, 290, 298–300, 302–306, 321–326, 359–361, 371, 393, 394, 397, 403, 414, 419, 435–445, 452, 464–468, 481, 486, 487, 494, 495, 504, 505, 508, 510, 530
Cloud computing (CC), 2, 3, 8, 76, 107, 111, 112, 116, 255–257, 264, 423–425

- Clustering, 15, 40, 282, 297, 298, 304, 423–433, 438, 461, 462, 466, 468, 496, 502
- Colorization, 141–156
- Computer vision, 14, 43, 59, 147, 248, 388, 398, 406, 411, 447, 448, 479–481, 517, 521
- Confusion matrix, 18, 20, 168, 197–199, 273, 274, 323, 326, 372, 393–396, 441, 445, 474, 511–513
- Content based image retrieval, 203–211
- Convolutional neural networks (CNNs), 52, 53, 147, 149, 152, 156, 193, 204, 214, 217, 245–253, 319–327, 366, 367, 412, 413, 417, 419, 447–457, 460–468, 504, 516–519, 522, 525, 543
- Copra, 387–398
- CORD-19, 215, 218–221
- COVID-19, 213–222, 245, 246, 248, 249, 252, 435, 437, 529–543
- CTU-13, 297–299, 305
- Cyberbullying, 171–188

- D**
- Darknet, 63, 64
- Data analysis, 15, 17, 23–34, 193, 281, 330, 348, 381–384, 502
- Data mining, 14, 15, 24, 27, 182, 329, 331, 491–499, 502, 504
- DDoS, 296, 297
- Decision tree, 14–16, 24, 25, 27–29, 34, 160, 163, 165, 167, 173–176, 183, 185, 192, 194, 200, 281–283, 291, 298, 299, 302, 303, 305, 306, 436, 441, 443–445, 461, 463, 466, 468, 471, 472, 481, 494, 496, 504, 505, 534
- Decoder, 52, 53, 55, 147, 149, 150, 156, 214, 217, 218, 220, 516, 517, 519, 520, 522, 525
- Deep learning, 14, 106, 141, 147–150, 152, 153, 159–169, 173, 192, 193, 200, 204, 246–248, 320, 366, 371, 384, 387–398, 402, 409, 414, 447, 448, 460–464, 466–468, 475, 497, 505, 517, 542
- Deep neural models, 205
- Deep neural networks (DNNs), 54, 106, 107, 109–111, 115, 162, 226, 227, 229, 234, 248, 388–390, 392, 393, 397, 401, 403, 448–451, 456
- Deep-Q learning, 106–110, 116
- Defective, 128–130, 133–138
- Demand response, 310
- Demand side management, 307
- DenseNet, 320, 449, 452, 454–456, 461
- Drone images, 59–72
- Duct, 479, 545–560

- E**
- Edge device, 5, 7, 107, 111, 112, 287, 329, 331, 336
- Edge-fog cloud computing, 106, 107, 112
- EfficientNet, 448, 449, 456
- Electron Js, 235–243
- Employee attrition, 279–292
- Encoder, 52, 53, 55, 147, 149, 150, 156, 214–218, 516–520, 522, 525
- Energy efficiency, 330, 331, 423–433

- F**
- Face detection, 242, 247, 248, 250
- Face expression recognition, 238, 241
- Face recognition, 14, 15, 248, 482
- Factored machine translation, 119–125
- Failure, 13, 14, 17–20, 97, 205, 509
- Feature extraction, 60, 62, 143, 145, 173, 204–207, 209, 211, 319, 320, 413, 414, 438, 440, 448, 451, 452, 456, 465, 466, 480, 483–487, 503, 518–519, 522, 524, 525
- Federated learning, 280, 287–290, 403
- Feed forward neural network, 160, 162–169, 217
- Fog computing, 3, 111, 112, 116, 256, 257, 329–338, 423–426
- Fuzzy logic, 38, 45, 47, 427–428

- G**
- Generative adversarial networks (GANs), 49–57, 147, 149, 150, 156, 401–409, 517
- Genetic algorithm, 47, 309, 319–327
- Gradient boosting, 282, 283, 285, 291, 463, 504
- Gradient boosting machine (GBM), 472–475

- H**
- Hadoop, 128–131, 133, 138, 258
- Healthcare, v, 23, 47, 59, 128, 280, 356, 377–385, 402, 435, 495, 496, 502

I

Image processing, 141, 166, 248, 368, 369, 411, 412, 481, 505
 Images, 15, 23, 46, 50, 53, 57, 59, 127, 141–156, 166, 174, 203–211, 214, 236, 247, 261, 319, 344, 365–374, 384, 388, 389, 401, 411, 414, 448, 460, 479–487, 492, 503, 515, 519
 Image-to-image translation, 53, 54, 56, 148–150, 155, 156
 Indian Languages, 225–234
 Information society, 342
 Information technology, 14, 350
 Internet of things (IoT), v, 1–3, 5, 7, 9, 14, 24, 25, 93–102, 105–107, 113, 203, 255–265, 297, 329–338, 348, 423, 425
 IoTCloud, 258, 259
 Israeli conflict, 192, 200

K

K-nearest neighbors (KNNs), 15, 16, 24, 25, 160, 163, 165–167, 173, 249, 269, 272–276, 281, 289, 291, 292, 298, 302, 303, 306, 324, 417, 436, 441, 444–445, 461, 466–468

L

Lemmatization, 120, 161, 438–440, 442–445
 Load balancing, 75–90, 330
 Logistic regression, 25, 27, 28, 34, 160, 163, 165, 166, 172, 175, 176, 183, 185, 228–230, 269, 271, 273–276, 280–282, 287, 291, 298, 302, 304, 306, 324, 436, 441, 442, 445, 462, 494–495, 498, 502, 504–506, 508–513, 531
 Lossless encoding-decoding, 226
 Low resource NLP, 234

M

Machine learning (ML), 13, 23, 43, 50, 61, 105, 145, 159, 172, 192, 204, 229, 237, 249, 258, 269, 280, 296, 355, 368, 378, 401, 411, 424, 436, 459, 479, 497, 502, 530
 Manholes, 93–102
 Manufacturing, 128–131, 138
 Matplotlib library, 360, 537
 Message Queue Telemetry Transport (MQTT), 257, 258, 329, 330, 333, 336, 337
 Microservice chain, 76–82, 84–90
 Microservices Architecture (MSA), 76

Minor lung cancer, 502
 ML techniques, 14–20, 23–34, 145, 163, 165, 185, 282, 402, 462, 468, 494, 501–513, 529–543
 Mobile edge computing, 1–9
 Models, 4, 14, 24, 40, 50, 60, 76, 96, 106, 119, 128, 143, 159, 172, 192, 203, 214, 226, 236, 247, 269, 280, 296, 309, 322, 329, 341, 356, 366, 377, 390, 401, 412, 425, 436, 448, 459, 492, 502, 517, 529, 549
 Moisture, 134, 265, 309, 329, 331, 387–398
 Moods, 235–243, 355, 356, 358–360, 363
 Multi-agent reinforcement learning, 307–316
 Muzzle pattern, 479–487

N

Naïve Bayes (NB), 15, 24, 172, 173, 179, 180, 185, 192, 194, 200, 282, 291, 298, 300, 302, 304, 306, 324, 356, 359–361, 417, 466, 486, 495, 497, 498, 504, 505
 Nanotechnology, 38–41
 Natural language processing (NLP), 14, 38, 41, 120, 159–162, 164, 165, 175, 192, 213–219, 221, 225–234, 355, 358, 362, 436, 438, 439, 516, 521
 Network lifetime, 332, 333, 335, 430–433
 Network slicing, 116
 Neural network, 15, 24, 38, 40, 54, 62, 107, 109–111, 115, 116, 160, 162–169, 200, 226, 227, 229, 234, 248, 287, 288, 366, 379, 388–390, 392, 393, 397, 405, 407, 411–421, 466, 496–498
 Non small cell lung cancer (NSCLC), 502

O

Object detection, 50, 59–72, 150, 248, 251, 482
 Offloading, 1–9, 105–108, 110, 113, 116
 Open CV, 248, 483
 ORB, 480, 484, 486, 487

P

PatchGAN, 53, 55, 149, 150, 156
 Pathology images, 365–374
 Pattern recognition, 43, 165, 411, 412, 496
 Pest classification, 449, 454
 Pix2Pix, 50, 53–55
 68 points facial detection, 241
 Politics, 191–200, 342, 350
 Potential improvements of UAV, 545

Pre-trained CNNs, 319–327, 448, 518, 519, 522
 Project, 13–20, 24, 38, 96, 129, 131, 133, 134, 233, 236, 243, 247, 249, 250, 252, 270, 546
 Public opinion, 191
 Publish, 329, 337

R

Random forest, 13, 14, 17–20, 160, 163, 165, 167, 172, 176, 183, 185, 281, 282, 291, 292, 368, 372, 373, 436, 441–445, 486, 494, 496–498, 534, 535
 Random forest classifier, 163, 176, 185, 282, 324, 436, 445, 481
 Raspberry Pi, 329, 337, 390
 Reinforcement learning, 14, 105–116, 214, 308
 Relevance feedback, 203–211
 Resource allocation, 1–9, 105–116
 RevCode, 225–234
 Robots, 44, 295

S

Scale-invariant feature transform (SIFT), 480–482, 484–486, 504
 Scheduling algorithm, 90
 Sentence vector, 216, 217, 515–525
 Sentiment analysis, 173, 191–200, 355–363
 Shap analysis, 280
 Sigfox sensors, 256, 260–262
 Skip connection, 53, 54, 62, 149, 150, 156, 515–525
 Smart phone, 100
 Social networks, 171–188, 191, 343–345, 347
 Speeded up robust features (SURF), 51, 145, 156, 480–482, 484–487
 SqueezeNet, 389–398
 Statistical machine translation, 120
 Stemming, 161, 163, 165, 196, 359, 438–440, 442–445
 Stock markets, 355–363
 Subscribe, 247, 329, 331, 337
 Supervised classification, 436, 441, 442
 Support vector machine (SVM), 14–16, 24, 25, 27–29, 33, 34, 51, 145, 146, 156, 160, 163, 165, 167, 172, 173, 177, 183,

192, 228–233, 249, 280–282, 291, 292, 298, 302, 305, 306, 368, 371, 373, 413, 414, 417, 436, 441, 443, 445, 460–462, 464–466, 468, 479–487, 502, 504–508, 510–513, 532–534, 538, 539, 541, 542

T

TEM images, 319, 320, 324
 TensorFlow, 235–243, 251, 498
 Text classification, 194, 229
 Text summarization, 213–222
 Tiny face detector, 240
 Tokenization, 122, 160, 161, 163, 219, 229, 359, 436, 438
 Transfer learning, 52, 65, 192, 214, 215, 247, 248, 320, 368, 371, 374, 389, 392, 393, 398, 448, 449, 451, 452, 481
 Twitter, 171–173, 182, 191–193, 195, 200, 342, 345, 347, 355–363, 435–445

U

Underground drainage, 93, 95, 101
 U-Net, 52–54, 149, 150, 156
 User generated content (UGC), 343–345, 347, 348

V

Vehicle re-identification, 203–211
 Video captioning, 517
 Virus texture classification, 319–327
 VisDrone, 61, 65, 69, 71, 72
 Visualize, 23, 68, 129, 134, 349, 467

W

Water, 51, 93, 95–98, 100, 101, 257, 260, 262, 265, 308, 310, 312, 331, 387, 448
 Web scraping, 360
 Wireless sensor networks (WSNs), 96, 256, 423–433

Y

You Only Look Once (YOLO), 59–72