Scalable Computing: Practice and Experience

Scientific International Journal
for Parallel and Distributed Computing

ISSN: 1895-1767

Universitatea de Vest din Timişoara

Volume 24(1) March 2023

SUBSCRIPTION INFORMATION: please visit `http://www.scpe.org`

Scalable Computing: Practice and Experience

Volume 24, Number 1, March 2023

TABLE OF CONTENTS

Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2023 SCPE. Volume 24, Issues 1, pp. 1–16, DOI 10.12694/scpe.v24i1.2008

POST-COVID REMOTE PATIENT MONITORING USING MEDICAL INTERNET OF THINGS AND MACHINE LEARNING ANALYTICS

SALKA RAHMAN[*], SURAIYA PARVEEN[†], SHABIR AHMAD SOFI[‡] AND SANIYA ZAHOOR[§]

Abstract. The Covid-19 pandemic disturbed the smooth functioning of healthcare services throughout the world. New practices such as masking, social distancing and so on were followed to prevent the spread. Further, the severity of the problem increases for the elderly people and people having co-morbidities as proper medical care was not possible and as a result many deaths were recorded. Even for those patients who recovered from Covid could not get proper health monitoring in the Post-Covid phase as a result many deaths and severity in health conditions were reported after the Covid recovery i.e., the Post-Covid era. Technical interventions like the Internet of Things (IoT) based remote patient monitoring using Medical Internet of Things (M-IoT) wearables is one of the solutions that could help in the Post-Covid scenarios. The paper discusses a proposed framework where in a variety of IoT sensing devices along with ML algorithms are used for patient monitoring by utilizing aggregated data acquired from the registered Post-Covid patients. Thus, by using M-IoT along with Machine Learning (ML) approaches could help us in monitoring Post-Covid patients with co-morbidities for and immediate medical help.

Key words: Internet of Things, Machine Learning, Data analytics, Data Sensing, Data Aggregation, Covid, Post-Covid

AMS subject classifications. 68T05

1. Introduction. The Coronavirus pandemic has brought researchers across the globe from different domains to work together in-order to have better understanding and evaluation of the virus. According to W.H.O coronavirus is defined as an infec- tious virus which is caused by SARC-CoV-2 virus. People who are infected by the virus will experience mild to moderate respiratory illness and will recover without and sort of medical help but some however, will need proper medical care and any age group can get affected by it and may get severely ill or even they can die [1]. The pandemic initiated from Wuhan city, located in China in August 2019 and within few months, the whole world also got infected by the spread of Covid-19 virus. The abnormal situation created panic and chaos all over the world. Further, governments across the globe followed several protocols and imposed safety measures such as wearing masks, applying sanitizers, restricting unnecessary movements, lockdowns and so on. However, with time it is very clear that none of it has actually helped in the long term instead, the common people suffered a lot. The pandemic came with many negative health impacts resulting in serious mental and physiological concerns such as anxiety, depression, loneliness, cardiac attacks, arrhythmia and other health issues. The whole concept of normalcy that we used to live back in the days got totally changed and as a result the patient care and healthcare systems also got affected. The elderly people and others with existing morbidity are a major concern in situations like these.The general symptoms of Covid-19 are pneumonia, body ache, fever, rashes, diarrhoea, cough, fatigue, and loss of taste or smell, breathlessness and so on. Further, from the on-going research it has been found that 10-20 percent of Covid recovered patients experience post-Covid symptoms and these conditions are known as long Covid or post-Covid symptoms. The symptoms seen after recovery from Covid-19 includes fever, memory problem, difficulty in breathing, chest pain, anxiety, body ache, headache, tiredness, vomiting, nausea, organ damage, brain malfunctioning, brain fog, and so on. The duration in which the long Covid symptoms occur is still not clearly specified however, after 3-4

[*]Department of Computer Science and Engineering, Jamia Hamdard University New Delhi, Delhi, India (salkarahman25@gmail.com)
[†]Department of Computer Science, Jamia Hamdard University New Delhi, Delhi, India (hussainsuraiya@gmail.com)
[‡]Department of Information Technology, National Institute of Technology Srinagar, Jammu and Kashmir, India (Shabir@nitsri.net)
[§]Department of Computer Science, University of Kashmir Srinagar, Jammu and Kashmir, India saniyazahoor.ku@gmail.com)

Table 1.1: Similarity of Symptoms [7]

S.NO.	SYMPTOMS	COVID	POST-COVID
1	Fever	Yes	No
2	Cough	Yes	Yes
3	Sour throat	Yes	Yes
4	Runny nose	No	Yes
5	Head ache	Yes	No
6	Body ache	Yes	Yes
7	Chest pain	Yes	Yes
8	Loss of movement	No	Yes
9	Red eyes	Yes	No
10	Rashes	Yes	No
11	Diarrhea	Yes	No
12	Loss of taste	Yes	Yes
13	Loss of smell	Yes	Yes
14	Fatigue	Yes	Yes
15	Anxiety	Yes	Yes
16	Depression	Yes	Yes
17	B.P irregularity	Yes	Yes
18	Sugar level fluctuation	Yes	Yes
19	Pulse rate fluctuation	Yes	Yes
20	Breathlessness	Yes	Yes
21	Brain fog	No	Yes
22	Kidney infection	No	Yes
23	Blood thickening	No	Yes
24	Discoloration on fingers/ toes	Yes	No
25	Joints pain	No	Yes
26	Cardio vascular diseases	No	Yes

months of recovery from Covid, people may experience post-Covid symptoms [2] [3][4][5]. The availability of data for post-Covid scenarios are quite limited however, by looking at the symptoms in both the cases, it is evident that Covid and post-Covid symptoms possesses similarity. Thus, the datasets available for Covid can also work for post-Covid scenarios. The similarity of symptoms are illustrated in Table 1.1.

Hence, the best way to avoid post-Covid conditions is to take precautions, get vaccinated and follow SOP there by reducing the chance of getting Covid-19 in the first place itself. Thus, post Covid patient needs proper medical care and for that advanced technologies needs to be introduced such as medical IoT-based devices, ML and deep learning approaches along with healthcare data analytics.

IoT-based systems are a connected network of devices and sensors such as smart-phones, tablets, smart watches, EEC-ECG sensors and so on, to exchange the data over the internet. These devices are embedded with sensors that allows them to transmit the data. This method can be used for the patient monitoring scenarios for Covid and post Covid conditions. In medical healthcare, IoT devices plays an vital role in processing the patient's condition, consider an example where a patient's blood pressure level needs to be monitored so, for that wearable IoT-based device such as smart Blood pressure monitoring watch can be used for analysing the condition of the patient on a regular basis in real time. The collected data can be stored on the cloud and pre-processing of the same can be done over there therefore, the doctors can then easily monitor and track the patient's progress by simply using his smartphone. In critical situations like the pandemic people avoid going for regular check-up's as it is risky thus, IoT-based remote healthcare monitoring system can be used for keeping the health track of Covid and post-Covid patients. So, different IoT sensors and devices like the fitness bands, oximeter, ECG/EEG sensors, qardiocore and so on can be utilized for acquiring the data on a regular

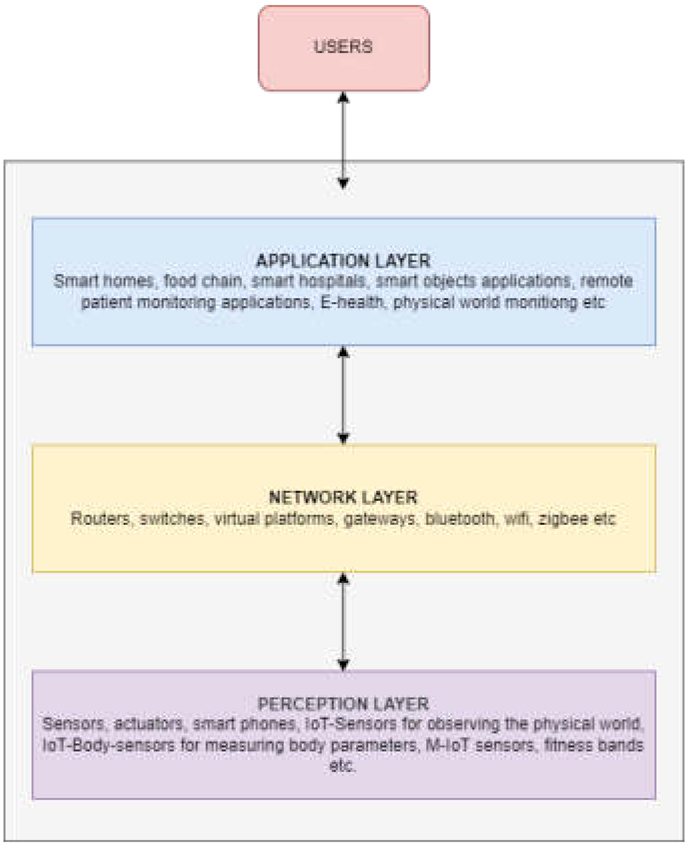
USERS
APPLICATION LAYER
Smart homes, food chain, smart hospitals, smart objects applications, remote patient monitoring applications, E-health, physical world monitiong etc
NETWORK LAYER
Routers, switches, virtual platforms, gateways, bluetooth, wifi, zigbee etc
PERCEPTION LAYER
Sensors, actuators, smart phones, IoT-Sensors for observing the physical world, IoT-Body-sensors for measuring body parameters, M-IoT sensors, fitness bands etc.

Fig. 1.1: General Architecture of IoT Technology

bases so that the monitoring of patients can be done while patient's stay at their respective homes. The general architecture of IoT-based technology consists of 3 layers i.e., the perception layer, the networks layer and the application layer as shown in Figure 1.1 [6].

The Perception layer acts as a physical layer where different sensors, devices, edge nodes etc. are connected in-order to collect the required data. Further, it helps in communicating with the surrounding and its various smart objects/ devices by detecting them. Once the data is gathered it is pushed to the next layer. The devices connected in the perception layer are sensors, actuators, Qr code and barcode readers, smart phones, portable pc's, surveillance camera etc. The Network layer analyses the sensory data acquired from the perception layer. Also, it helps in linking the smart object, network devices and servers. Further, it transmit the collected data by distributing and storing. The devices associated in the network layer are routers, gateways, hubs, switches, bridge, modules and so on. The Application layer acts as an interface between the user and the IoT-based system. It helps its users to access the data. Also, provides applications to its users by enabling them to use the functionalities of the sensors and devices. The different devices used at this layer are smart phone, pc's, smart watch, tablets and so on. Its applications can be seen in smart cities, hospitals, food chain supplies, etc.

ML approach is a method by which we can obtain hidden patterns and insights by using algorithms. The main objective of ML is to allow computer system to learn on its own by analyzing its environment without being explicitly programmed or hindrance by humans. Computers on its own are a very powerful machine which are good at calculations however, by applying ML we can make them intelligent predictors. To make the computers intelligent they need to go for training and for that datasets are used which helps the computer to gain experience and observe its environment, once the training of the model/ computer is done ML algorithms such as KNN, SVM, DC and so can be applied for accurate prediction. The basic work flow of the ML modelling

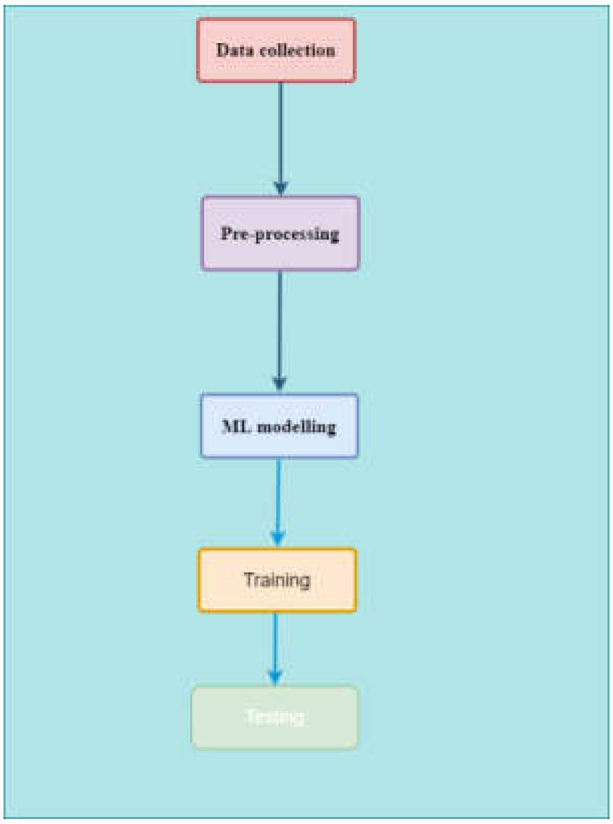

Fig. 1.2: Flow Diagram

is shown Figure 1.2.

The first step in the ML flow diagram is the data collection layer which is done in-order to gather all the related data of the proposed problem. There are a variety of methods used for data collection such as interviews, surveys, polls, observational, experimental and so on. Once the data is collected it is pre-processed in-order to have only the relevant data which is used for the modelling purposes. Once the model is built it is ready to go for training and testing purposes for providing accurate predictions. Further, ML algorithms are used for a no. of applications especially for diagnosis purposes in the healthcare sector. The conventional or traditional ML approaches like KNN, DC, SVM, Logistic Regression, Clustering approach, PCA etc. were used on its own for prediction however, the results obtained from these algorithm were not optimized therefore, the need for advanced technology came into existence. Ensemble learning is one of the powerful ML approach in which multiple ML algorithms are Integrated together such as regression, logistic regression, SVM, KNN, DC and so on in-order to form a stronger classifier for precise and accurate predictor machine/ model. The classifier has more predictive power, has lesser error rates and maximized accuracy on comparison with the traditional ML methods as discussed in [20]. The general layout of the ensemble learning classifier is shown in Figure 1.3.

At the bottom is the training dataset which consists of the dataset D that is divided into smaller partitions d1, d2, . . . , dn. These partitions of dataset are then used to train the classifiers or learners L1, L2, . . . , Ln at the second level. The learners makes classification and generates different outputs O1, O2, . . ., On respectively. The 3rd level is the combining decision which basically aggregates all the outputs and uses voting method to choose the best prediction. Finally the prediction P is generated. This advanced integrated approach can be incorporated for the analysis of Covid and post Covid cases along with IoT-based technology there by minimizing the spread of virus among the people. Thus, various IoT-based devices and sensors can be put on the human body for regular monitoring of the health status. The data provided by the sensing devices can be

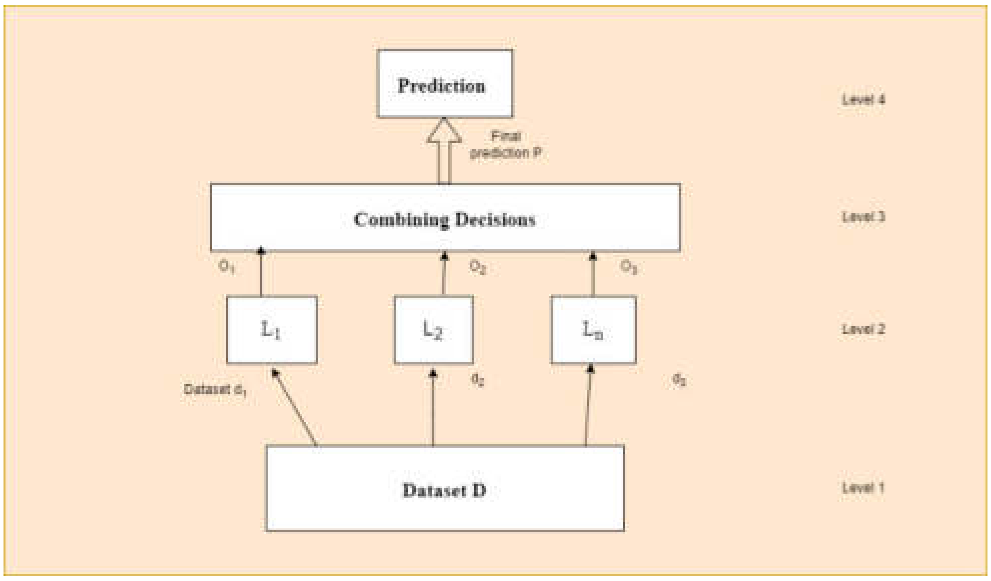

Fig. 1.3: General Layout of Ensemble Modelling

sent to the edge nodes where the aggregation of the gathered data can be done. Further, the aggregated data can be sent on to the cloud for pre- processing and modelling of the data in-order to generate reports for the doctors to have their expert recommendations. In this way the load on the medical staffs at hospitals can be seen reduced also leading to limited spread of the virus.

The organization of paper is: Section 2 discusses the literature survey, section 3 presents proposed IoT data models, section 4 implements the approaches for generation of data models, section 5 presents the real time setup and results, and section 6 gives conclusions.

2. Literature survey. There has been work done in IoT, ML, deep learning, cloud computing, cognitive systems, fog computing and Wearables for remote patient monitoring in Covid and post-Covid scenarios. These are discussed below.

2.1. IoT-Based technologies. The authors in [40] proposes an IoT-based sys- tem along with wireless sensor network architecture to monitor the quarantined pa- tients. Further, an urgent alarm system is also incorporated so that the healthcare professionals can get an urgent message via sms when there is any inconsistency in the patient's condition especially the elderly people needs more care. In the proposed architecture; Analog signals are received by the Arduino Uno which are sent by the sensors and transformed into digital format which is then processed by the micro- controller and the pre-processed data is then sent to the cloud so that healthcare professionals can access the data from anywhere. The authors in [43] proposes a re- mote smart home healthcare support system (ShHeS) in which patients are remotely monitored by the healthcare professionals and even patient's gets prescribed from the ease of their respective homes. The proposed mobile system interacts with the web application for better patient- doctor communication. Here, Sensors automatically captures the physiological parameters of the patents also, for service discover and context change of the environmental at home a hyperspace analogue to context (HAC) is added for accurate readings of the parameters. In this way, the new system reduces the hospital visits, queues and the cost of taking care of the patients at the hospitals.

An IoT-based Healthcare Assistance system is proposed in [32] for Covid-19 pa- tients. It's a survey based research paper in which they have integrated multiple IoT associated technologies like remote monitoring, robot based assistance, digital diag- nostic etc. which can be used for solving the problems faced during Covid-19 by the patients, health workers and doctors. They have also provided a wide range of do- mains in which existing health monitoring system have been developed which can be looked upon for solving the problems associated with the pandemic.

The authors in [34][13] proposes an IoT-based conceptual architecture that focuses on scalability, inter- operability, network dynamics, context discovery, reliability, and privacy issues faced in remote patient monitoring for the COVID-19 patients in hospitals and at home. Further, consent management module is also incorporated into the architecture that ensures data privacy and brings transparency to the patients. Also, an early-warning score method is used for checking and monitoring the hospitalized patients. An IoT- based System is proposed for safer movements in the pandemic using machine learning algorithms called SafeMobilit that integrates IoT, fog, and cloud solutions to monitor the social distancing between people and control the capacity in the spaces. Also, if any violation happens an alert message is generated and sent using IoT applications. Further, the fog node sends the information to the cloud server and displays the con- gestion sites in the environment. The results shows that SafeMobility has an accuracy of about 91 percent, hence the movement is detected in the indoor space as mentioned.[46]. In situations like the pandemic there is high risk of uncertainities therefore there is a need for awareness among the common people. Thus, a very impressive work done by the authors in [18] have proposed potential IoT and web technologies in the health sector for these scenarios.

2.2. ML approaches. The authors in [45]have proposed a method that classi- fies cough and is known as multi-criteria decision making (MCDM) method that uses ensemble learning techniques for COVID-19 cough classification. The Cambridge, Coswara, Virufy, and NoCoCo cough databases have been used for the training of the proposed method. Further, the proposed method incorporated TOPSIS. Soft ensemble and hard ensemble learning methods are used to identify the COVID-19 diagnostic model and machine learning techniques that give best results which are further used to classify the cough as COVID-19 or non-COVID-19 .Also, for the evaluation of the method entropy is used. A detailed study on machine learning, systems biology and bioinformatics is done thereby creating a disease model and host-microbe interaction disease network. The study can be utilized by clinicians and researchers to improve the prognosis, detection and treatment of post-COVID-19 mucormycosis that requires further evidence and testimonials in post-COVID-19 mucormycosis cases [16].The authors in [8] proposes an integrated method for the monitoring of Covid-19 by incorporating machine learning algorithms and IoT device. The model predicts the risk associated with COVID-19 pandemic by using ML algorithms such as Random forest (RF) and Naive Bayes (NB) classifier. For the data collection, IoT sensors are used and the accuracy associated with the RF and NB classifies are 97.The authors in [13] proposes a ML-based model for the prediction of anxiety during the Covid-19 pandemic in Saudi Arabia. The prediction models is developed by using Support Vector Machine (SVM) classifier and J48 Decision Tree algorithm. The results obtained for the early classification of two-class and three-class anxiety problems were exceptionally promising. The SVM gave 100 percent accuracy and the J48 Decision Tree gave 95.96 percent for the three-class problem and 93.50 percent for the two-class problem when predicting anxiety for earlier diagnosis and timely intervention using ten features.An Algorithm that helps in accessing the Covid-19 patients severity, needs and predict the length of their stay by using breathing frequency(BF) and oxygen sat- uration (SpO2) signals. BF and SpO2 signals of the confirmed Covid-19 patients who have been admitted in the ICU were used for the data collection that is then clustered using unsupervised ML method. The severity score S24 is utilised for representing patients severity; for the validity MS24 and maximum S24 score are used for evalu- ating the rates of intubation risk and long ICU stay. The proposed algorithm uses simple signals that efficiently visualizes the respiratory concerns of the Covid infected patients. The system includes 279 patients. The result obtained from unsupervised clustering for intubation recognition is about 87.8 percent [12]. In a similar manner the authors in [21] have proposed AI- based application for the diagnostic of diabetes retinopathy.

2.3. Deep Learning. The authors in [25] conducted a detailed review focussing the utility of AI in the domain of imaging for COVID-19 patient care. A systematic meta-analysis is done for technical merit and clinical relevance of the collected data. The main challenges and the gaps are discussed; thereby providing recommended solutions and remedies as well. The authors in [33] propose a deep learning-based framework by incorporating fuzzy logic that distinguishes between CXR images of pa- tients with Covid-19 infection and as well as patients who are not infected by covid-19 virus. The proposed model is known as CovNNet that is used for extracting relevant features from CXR images, which is then combined with fuzzy images that are gener- ated by a fuzzy edge detection algorithm. The proposed approach is validated using additional datasets.

Further, the experimental results shows promising results with an accuracy of about 81 percent. In [35] an in-depth study for imaging and medical image computing in COVID-19 diagnosis. The detailed study helps in providing tutorial for both clinicians and technologists, comprehensively review of the characteristics of COVID-19 as presented in medical images, inspect automated artificial intelligence- based approaches for COVID-19 diagnosis, and provide research limitations and the methods used to overcome them. By incorporating machine learning-based methods for the diagnosis of the disease will be helpful. Further, AI-based automated methods for COVID-19 diagnosis can provide promising results.

The authors in [28] presented aclassifier for COVID-19 detection which utilizes transfer learning for improving the performance and robustness of the DNN in vocal audio such as speed, cough and breathe. 10.29 h audio data is used that consists of four classes: noise, cough and sneeze. Further, the pre-trained COVID-19 audio classifiers use nested k-fold cross-validation. Thus, deep transfer learning method provides better vocal detection of Covid-19. The results obtained for cough, breath and speech are 0.982, 0.942 and 0.923 respectively. The authors in [26] described a model that uses artificial neural network (ANN) for the prediction of deaths and future daily cases due to COVID-19 in a generalized way so that it can fit the spread of different countries'. The proposed methodology predicts next ten days of the Covid- 19 spread that can be used for taking precautions and decreasing the spread. Also,87 percent accuracy is seen for predicting the number of cases and 86 percent accuracy is seen in predicting the mortality rate. In [9] an in-depth study of deep convolutional neural networks with the latest evelopments, protocols and applications is done to analyse the existing work. The rchitecture for COVID-19 prognosis is proposed with, limitations, outlooks and challenges for better modelling, design and training of the modules. Further, the study highlights the recent studies of the applications of deep CNN for, detection, screening, prognosis and classification, of COVID-19.

2.4. Cloud Computing. The study in [31] represents the characteristics, ap- plications and benefits of cloud computing and elaborates the improvement essential for cloud during COVID-19. The cloud computing helps the countries in handling the COVID 19, in commercial, educational, economic and health sectors. The results showed that cloud computing brings effectiveness during the epidemic. A framework that consists of wearable devices, clinical tests and hospital records for collecting Pa- tient data at the cloud infrastructure is proposed in [29]. For the Security at the cloud Audio-Steganography is used whereas for the authentication and faster transfer of data of a patient Near Field Communication (NFS is) used. Healthcare workers are performing their duty in critical situations like the pandemic the proposed automated framework will worker significantly by reducing the over load at hospitals.

The authors in [44] provided a model that transforms a modern hotel into a cloud- based virtual ward care centre that is integrated with cloud information system and social media which provides medical services just like hospitals. Also it provides an alternative option for patients to get quarantine. Routine ward rounds, measurements, recording of vital signs and medical consultation were done on social media platform to minimize the between staffs and patient. Therefore, Cloud-base information system are used for the Covid infected patients. An improved ML and cloud-based model is proposed to predict the threat of COVID-19 worldwide. Further, a case study in the paper illustrates the severity of CoV-2 spread worldwide. Using the proposed model statistically better predictions is obtained [14].An overview of cloud-based health- care system used for faster deployment in the organisations during the COVID-19 is proposed here. Detailed discussion on the consequences of cloud technologies and the issues related to data governance and privacy, data silos, unintended implications and data structures are done[42].

2.5. Cognitive Systems. A survey on Cognitive Computing (CC) is done in [17]. The paper consists of the challenges, solutions and future research directions as- sociated with it. Healthcare, cyber security, big data and IoT are the four fields where the application of the proposed work can be seen. A detailed study is carried out in which different applications areas for CC is discussed. The functionalities of cogni- tive dysfunction in patients after COVID-19 and the correlation between cognitive function and anxiety, depression, sleep, and olfactory function is done in [15]. A survey based research is carried for in which the applicability of Block chain technology along with in-depth discussion on the opportunities and challenges of the Cognitive Computing and Healthcare is done in [41].

2.6. Fog Computing. Fog assisted IoT-based framework is described for the protection and prevention from the coronavirus. The proposed framework predict COVID-19 infection by processing patient's health data. It is done by observing their symptoms thereby generating an emergency alert, medical reports, and precautions to the user, their guardians and as well as doctors and experts. It collects information from the hospitals and quarantine shelters through the patient IoT devices. Further, an alert message gets generated so that the government health agencies can take control to the outbreak of chronic illness and take quick actions accordingly [22]. The authors in [27]proposes an IoT, Fog computing, and Cloud computing based technologies for home hospitalization system. It allows the patients to recover and get treatment at their home, where patient health is monitored by the doctors and health workers to follow the hospitalization process and make recommendations to patients through monitoring units and mobile applications developed for this purpose.

A Health-care architecture is designed for processing, data collection, and trans- mission. It gives provides applicability of fog devices and gateways in Healthcare environment for current and future applications. The analysis for the cloud computing, IoT and fog computing is done to provide context-aware services to the users when required [37]. In [36] a cloud-fog-dew based monitoring Framework is proposed for Covid-19 management known as CONFRONT. The proposed model helps in di- agnosis and monitoring of the patients while they are in quarantine or home based treatments. To analyse large scale COVID-19 statistics data cloud servers are utilized because of its better scalability, computation and storage capabilities. It consists of heterogeneous sensors to realize the proposed model. Also, the Dew architecture has helped in improving the uptime of the proposed health care framework.

2.7. Wearables. The research in [24] is based on an in-depth survey, the pa- per explores virtualized care systems, wearable sensor devices, and real-time medical data analytics in COVID-19 patient health prediction. Analyses is performed and estimations are made for the COVID-19 patients. Descriptive statistics of compiled data from the completed surveys were calculated when appropriate. A predictive analysis is done in [23] on the downloaded dataset which are designed especially for COVID-19 quarantined patients. The wearable devices used on quarantined patients that provides the healthy and unhealthy patient data. The wearable device provides data of heart rate, SPO2, temperature, blood saturation, and blood pressure timely. The results of symptoms can identify the severity of the patient. In-depth study has analysed the risk of COVID-19 disease progression using random forest classifier algorithm.. The result has predicted the accuracy score of 99.26.

In [38] a proposed model is designed for detecting hidden signatures of Covid-19 by integrating biomedical sensors, and AI. The three types of biosensors: blood pres- sure biosensor, electrochemical biosensor, G-FET-based biosensor, and potentiometric biosensor are used for collecting the sensory data. The proposed model distinguishes between the severe and non-severe cases under the demographic features associated with it. The authors in [11] illustrated the utilization of different sensors used for improvement of the covid-19 disease control. The essential signs are measured and a details of the wearable systems used in emergencies such as epidemics is provided here. Implementation could help in better management and control and monitoring of the signs.

2.8. Wireless sensor network (WSN). A home health monitoring system is proposed in this research chapter to identify heart related risks and monitor oxygen level in a person by utilizing BSN simulator and energy performance of the network. This model helps to monitor and identify two types of condition in a person i.e., people with heart diseases and people who needs oxygen levels monitoring after recovery from the coronavirus disease [30].An IoT-based remote healthcare monitoring system is proposed in [39] that basically provides patient health conditions through Web browser. For simulation of the proposed framework Contiki OS with 6LoWPAN protocol stack and Cooja OS are used. The main focus of the work is to provide patient's vital parameters such as the ECG rate, glucose lever, oxygen level and so on through the Web browser; which is done by implementing CoAP protocol in Mozilla Firefox Web browserFor real-time regular health care monitoring a general three-tier omnipresent telemedicine scheme that is based on Wireless Sensor Network (WSN) is used to address healthcare related issues. Further, for the constant communication 3G/4G is incorporated with the wireless sensor network to increase the bandwidth that is calculated at TOA for proper communication [43].

3. Summary. A tabular summary of some of the papers in the literature review is presented in Table 3.1.

Table 3.1: Summary

References	Technology	Target population	Covid/Post-Covid
Sharma et.al[10]	IoT, Mobile Computing, Cloud Computing	Elderly people	Covid
Taiwo et.al [11]	IoT	General	Both
Verma et.al [16]	ML, Bioinformatics	General	Post-covid
Deepa et.al [17]	ML,Random Forest,Naïve Bayes,IoT	General	Covid
Boussen, et.al [20]	Unsupervised ML, clustering	Generic	covid
Born et.al [21]	AI, Data Analysis	General	Covid
Ieracitano et.al [22]	AI	General	Covid
Nabavi, et.al [23]	ML, AI	General	Covid
Pahar et.al [24]	Transfer learning,ML, Deep learning	General	Covid-19
Khanday et.al [26]	ML, Deep Learning.	General	Covid
Alhomdy et.al [27]	Cloud Computing	Generic	Covid
Moorthy.et.al [28]	Cloud computing, wearables.	Generic	Covid
Lim et.al [29]	Cloud Computing, Information system	General	Covid
Sreedevi et.al [32]	Cognitive computing, Big Data, IoT, Cyber Security.	Generic	Both
Delgado et.al [33]	Cognitive computing	Generic	Post-Covid
Singh et.al [35]	Fog, IoT.	Gardians, doctors	Both
Parker et.al [39]	Wearables	Generic	Covid
Hussain et.al [40]	Wearables, Data analytics, Random Forest.	Generic	Covid
Hemamalini et.al [41]	Bio Medical Sensors.wearables, AI	Generic	Covid
Rida et.al [45]	Wireless network.	Generic	Both

4. Proposed framework. The healthcare services and systems are still in the developing phase in India especially the rural areas have witness many challenges as proper medical facilities and infrastructure are not available to them. The coronavirus situation has further increased the level of difficulty in India, as a result rural areas have badly failed in tackling covid-19. The situation in urban India back in May 2021 was also very terrifying as they have failed in providing medical supplies, ventilators, treatments, oxygen cylinders and so on to its people thereby leading to countless deaths thus, we can imagine how worse the situation was in the rural areas. According to times of India around 52 percent of deaths were recorded in rural area itself in the month of May 2021 [47]. To overcome this issue we have proposed an IoT-Based remote patient monitoring system that helps in keeping the tract of the Covid and post-Covid patients while they stay at the ease of their homes. A variety of sensing devices are attached to the patients for regular monitoring along with lab testing's which provides the required data for the analysis purposes. Further, the gathered data is sent to the aggregate module where all the patient's medical data is combined in-order to form clusters which is then pushed to the cloud infrastructure for further processing and analysis. Also, medical expert module is also integrated in the cloud infrastructure layer that basically analysis and provide recommendations on the report generated by the modelling layer. The report along with the expert recommendation is sent to the patient as a feedback. The layout of the proposed framework consists of 4 different layers. At the bottom is the patient that uses IoT sensing devices for providing the data on regular basis also Lab testing results contribute in providing the data. Together the IoT sensing devices and lab testing results; form data monitoring layer that helps in gathering the required data of the respective patients. The data gathered is pushed to the aggregate module layer which helps in forming clusters of the collected data and further pushes it to the cloud layer. At the cloud the very first step is to pre- process the clusters by applying EDA method [10] [19]. Once the pre-processing is done, the data is sent to the ML modelling layer for applying classification algorithms in-order to classify the patients'

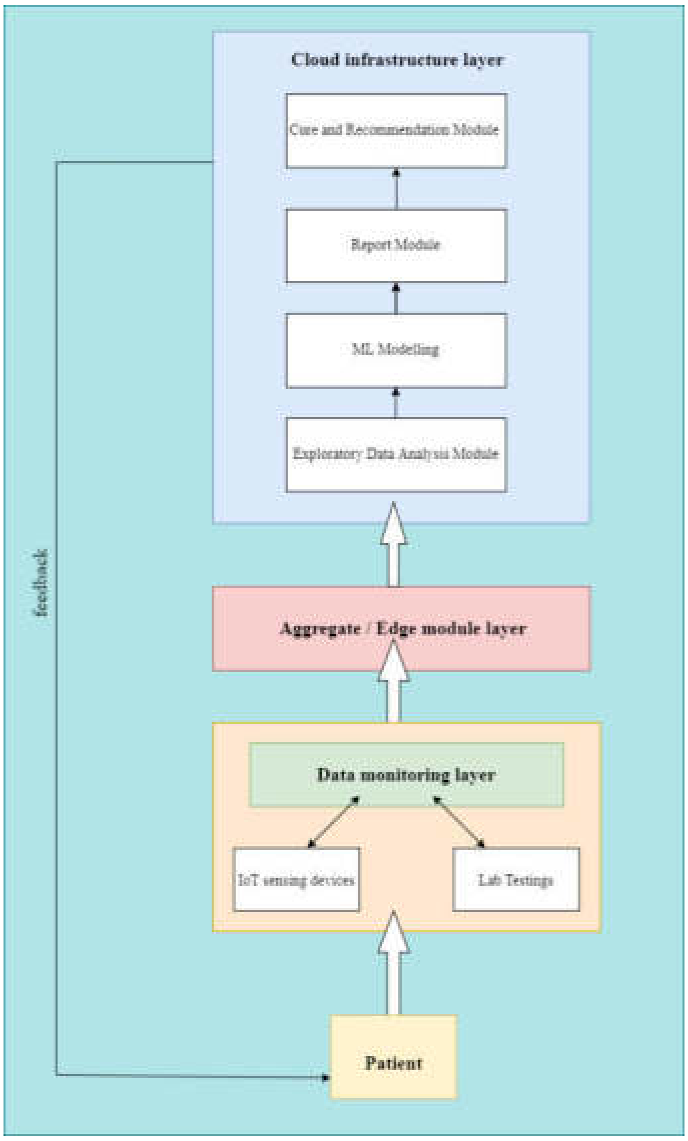
Cloud infrastructure layer
Cure and Recommendation Module
Report Module
ML Modelling
Exploratory Data Analysis Module
feedback
Aggregate / Edge module layer
Data monitoring layer
IoT sensing devices
Lab Testings
Patient

Fig. 4.1: Proposed Architecture Design

health status. The output generated is then transformed into documentation at the report module layer and the same is sent to the cure and recommendation module for expert opinions and recommendations. Once the processing is done the cloud sends the report along with the recommendations by the medical expert to the respective patient as a feedback.

5. System Design and Mathematical Modelling. The proposed Architecture has three functionality layer i.e, the data monitoring layer, the aggregate module layer and the cloud which are associated with each other i.e., the output of one layer is taken as an input on next layer and the process continues till the end as shown in Figure 4.1. At the bottom is the Data monitoring layer where the data is generated with the help of sensors and devices such as EEG sensor, ECG sensor, fitness bands etc. , and as well as with the results of lab tests such as the blood test, x-rays, urine test and so on. The data acquired helps in the monitoring purposes for the Covid and post-Covid patients. The data monitoring layer has 2 modules i.e.,the IoT sensing module and the lab testing module. The IoT sensing module consists of different sensing devices which helps in generating sensing data, hat is sent to the edge layer. Let I1 be the set of sensors described by 5 attributes or features of a patient P1.

Let I1 = P1, B1, O1, D1, T1 such that P1, B1, O1, D1, T1 representing pulse rate, blood pressure, oxygen level, diabetes and temperature respectively subject to constraints such that $60 \leq \mathrm{P1} \leq 100$ per minute, $90/60 \leq B1 \leq 120/80$ mmHg, $92 \leq \mathrm{O1} \leq 100$. Thus, at the Aggregate module layer the data generated at data monitoring layer gets pushed to the Edge layer where all the acquired medical data gets aggregated in-order to form clusters. There are two feasible scenarios of the system i.e., the Stationary IoT sensors-devices & Edge and the Dynamic IoT sensors-devices & Edges. At the Stationary IoT sensors-devices & Edge (at home/office) the location of all the sensors, devices and edge node is fixed and known. All the IoT sensors are directly associated with the particular edge device, providing sensory data DI1 to the edge. Let I1, I2, I3, ..., Iu be the set of stationary IoT sensor-devices attached to the patients at home or at office directing data DI1 to the edge node E. Therefore the regular monitoring of patients health can easily transmitted as the patient's location is fixed. At Dynamic IoT sensors-devices & Edge (from home to office and vice versa), both the IoT devices-sensors along with the edge nodes are moving. Therefore, any IoT sensor can get associated to any edge depending upon the nearness between them. When the patients are moving the monitoring of their health can be done via this scenario. To calculate the distance, let {(a11, b11), (a12, b12), . . ., (anm, bnm)} represent the positions of sensors {I11, I12, I13, . . ., Inm} respectively and let (ae1, be1), (ae2, be2), . . ., (aey, bey) denotes the positions of edge nodes/ devices {E1, E2, E3, . . ., Ey}. Thus, distance between sensors I11 and I12 :

$$d11 = \sqrt{(a11 - a12)^2 + (b11 - b12)^2}$$

The distance between the sensor I11 and edge node E1 :

$$D11 = \sqrt{(a11 - ae1)^2 + (b11 - be1)^2}$$

At the Cloud infrastructure layer, all the clusters formed at the aggregate module leyer is pushed to the cloud for further pre-processing and analysis of the same in order to have a better understanding and evaluation of the results. Here, the main objective is to evaluate the constrained which have been defined i.e. if I1 greater than T1 or L1 greater than T2 where, I1, L1 are the set of sensors and lab tests, and T1, T2 are their threshold values respectively.

After this, it connects to the doctors Doi and sends recommendations along with the report to the patient as a feedback. The cloud infrastructure consists of four different modules performing their respective functionalities. At the bottom is the Exploratory data analysis module (EDA) that is used for analysing the aggregated data values pushed by the edge layer. It is basically a collection of multiple methods which are used for understanding the variables, clean the data and visualize the data before developing the model for better understanding and evaluation. It aims to fetch hidden patterns, trends, associations, relationships present within the data. Thus EDA process helps to process raw data in-order to get meaningful insights allowing the developer to comprehend the data before making any assumptions. The EDA module has further three components:

- *Understand the variable:* For understanding the variables the first step involved is to import all the necessary libraries and load the datasets for shaping and applying mathematical functions to comprehend the variables and other constraints.
- *Clean the data:* It is one of the most important component used for removing redundant values, null values, removing outliers, fixing over or under fitting, fixing missing values, applying feature selection and feature engineering methods to obtain the relevant data on which ML approaches can be applied.
- *Analyse the relationship:* Visualization helps in analyzing the data by using visualisation techniques such as correlation matrix and graphical representations of the data points helps in better identifying the relationships between the values.

Once the data is pre-processed by eliminating the null values, redundant values and so on by applying EDA method, it is ready to go for ML modelling. So, various ML and deep learning classification algorithms are applied on the data at the ML modelling module for accurate prediction. Classification algorithm is a type of supervised learning method in which the data is classified into distinct categories of output. In the case for pandemic it classifies Covid and post Covid patient's condi- tion weather the person falls under the normal, severe or acute category. When the classification is complete and the model generates an output the same is pushed to the report module for documentation. When the prediction is made by the modelling module, the output data is documented into the report so that the same can be sent to the doctors for their expert examination and recommendation of the result. The cure and recommendation module basically helps in the evaluating the result report generated by the report module at the cloud. The doctor set Doi examine the report and provide recommendations if needed to the patient and the same is sent back to the patient along with the report as a feedback. Thus, we have C $\leftarrow$ Doi, where C is the cloud and Doi = Do1, Do2, ..., Dox. Also, Do1 $\leftarrow$ P1 and recommendations, R1 $\leftarrow$ P1.

6. Post Covid Data Analysis. It is important to choose the correct data collection method as different tools are used for different research scenario. A dataset is a collection of related information that is gathered for a given problem. The dataset contains historical records or cases or observation and each case include numeric fea- tures that quantify a characteristic of an item. There are different methods available for collecting the data such as surveys, polls, experiments, reports and so on. In the case of Post-Covid patients the availability of the data is very limited and the research is still ongoing. We have seen similarity of symptoms in covid and post-covid cases however, exclusive datasets for Post-covid is a must therefore, in our research we have reached out directly to the PubMed and they have provided the required dataset. For the analysis purposes we have utilised RapidMiner studio tool which is a data science platform developed by Ralf Klinkenberg, Ingo Mierswa, and Simon Fischer at Technical University of Dortmund in 2001. It is a fully automated data science tool that provides an integrated environment for Machine Learning, Deep Learning, data analytics, text mining and predictive analytics [48]. The platform is used for commercial, business, educational, application development and research purposes as it provides features such as data processing, optimization, model validation and re- sults visualization. The complex analysis of the relevant features extracted from the dataset is discussed below.

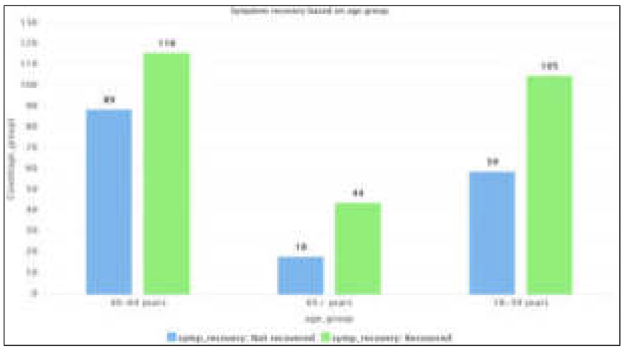

Fig. 6.1: Symptom recovery based on age group

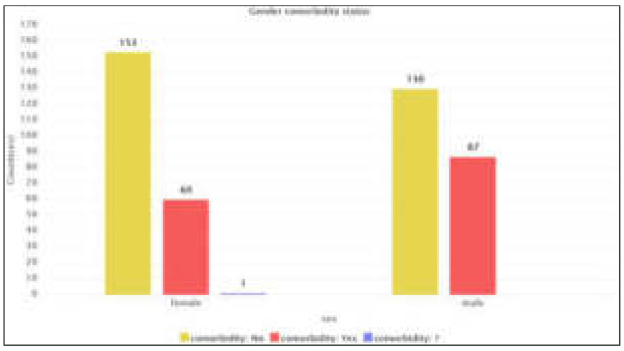

Fig. 6.2: Gender Comorbidity

Figure 6.1 represents the status of symptom recovery in different age groups. The x-axis shows the 3 age groups of the people as 18-39 years, 4-64 years and 65 and above. In the 18-39 age group 105 patients have recovered whereas 59 patients are not recovered, in 40-64 years age group 116 people have recovered whereas 89 people are still not recovered similarly in 65 and above age group 44 people have recovered and 18 people have not yet recovered.

Figure 6.2 represents the existing aliments in the gender. In the females 151 patients did not have any sign of comorbidity whereas 60 patients possesses signs of comorbidity. In a similar manner in the males 130 patients have no sign of comorbidity in them whereas 87 patients possesses comorbidity.

Figure 6.3 represents the status of respiratory issues on the three age groups. In the age group of 18-39 years 38.0 percent of respiratory issues are seen, in 40-64 years age group 48.28 percent of respiratory issue was recorded and lastly in 65 and above age group people possesses 13.8 percent of respiratory issues.

Figure 6.4 demonstrated the complication status in different age groups due to smoking. In the age group of 18-39 years 150 patients have no complications whereas 12 patients possesses complication. IN the age group of 40-64 years 158 patients have no sign of complications however 47 patients possesses complications. In a similar

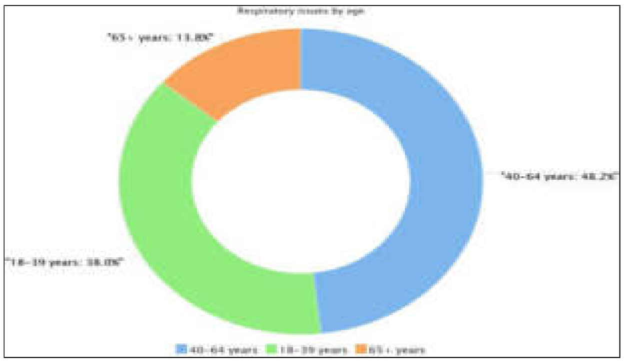

Fig. 6.3: Respiratory issues on age group

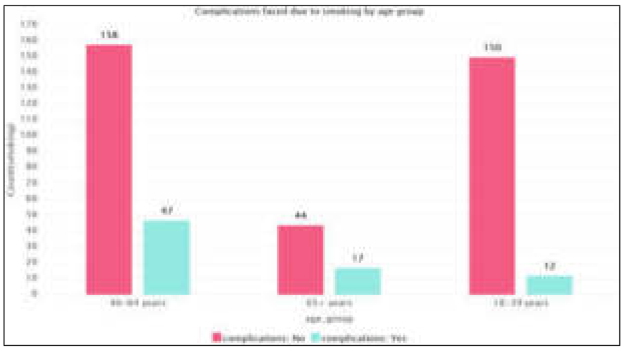

Fig. 6.4: Smoking complications on age

manner in the age group of 65 and above 44 patients shows no complications whereas 17 patients possesses complications.

Figure 6.5 shows Symptom recovery and complications in the infected patients. 126 patients have no recovery and has no complications either, 228 patients have re- covered and they also have no sign of complications in them. Whereas 40 patients have no recovery and also have complications. Even though, 37 patients have recovered however, possesses health complications.

Figure 6.6 illustrates the severity of symptoms in the infected patients and the complications related to it. The x-axis demonstrated the status complications in each of the severity group described as asymptomatic, mild to moderate and severe to very severe symptoms whereas the y axis shows the total count. In the asymptomatic group 36 patients have no sign of complications whereas 10 patients shows complications. In the mild to moderate symptom group 196 patients have no complications whereas 25 patients possesses complications. And lastly in the severe to very severe symptom group 122 patients have no sign of complications however, 42 patients possesses complications.

Figure 6.7 represents the presence of fatigue in patients according to their age group as 18-39 years, 40-64 years and 65 and above years of age groups. The x-axis shows the total number of patients facing fatigue and the y-axis shows the status of fatigue on age groups. In the 18-39 years of age group 59 patients have no sign of fatigue whereas 105 patients in the same group possesses fatigue. Similarly in the age group of 40 -64 years 100 patients have no sign of fatigue whereas 104 patients have fatigue. Lastly in 65 and above age group 34 patients show no sign of fatigue whereas 24 patients do have fatigue.

Figure 6.8 shows the presence of depression among males and females. The x- axis represents the categories of gender as male and female that shows the status of depression among them as no depression, mild to moderate depression and severe to very severe depression. And the y-axis shows the total number of patients in each of the categories. In the female group 149 patients have no depression, 50 possesses mild to moderate whereas 12 shows severe to very severe depression. In a similar manner in the male group 168 patients have no depression, 35 patients shows mild to moderate sign of depression and 14 patients falls under severe to very severe depression.

Figure 6.9 represents the presence of stress on the infected patients and their recovery. The x-axis shows the status of stress as mild to moderate, no stress, severe to very severe stress and the recovery associated with it. The y-axis shows the total count in each category. In the no stress group 80 patients are not recovered however 277 patients have fully recovered and are at the normal health status. Similarly in the mild to moderate stress group 20 patients have no sign of recovery whereas 31 patients have recovered back to normal. In the severe to very severe group 10 patient's show no sign of recovery and 7 patients have normal health status.

7. Discussion. The Post-Covid patient care has become more important be- cause of the fact that many patients develop complications after their recovery from the Covid-19. There have been many deaths reported in the Post-Covid era due to one or multiple organ failure which can be attributed to the unnoticed and unmonitored patients after covid. Although, we have analysed only few parameters for the Post-Covid scenario in our

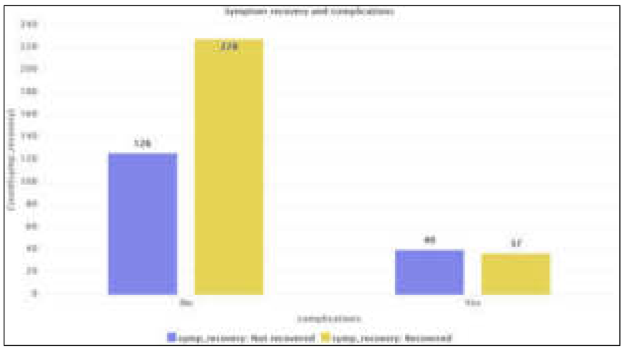

Fig. 6.5: Symptom recovery and complications

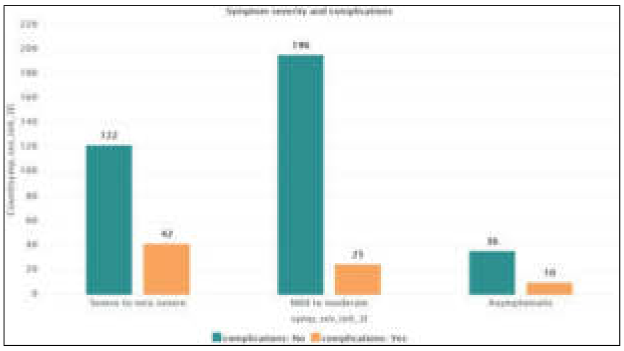

Fig. 6.6: Symptom severity and complications

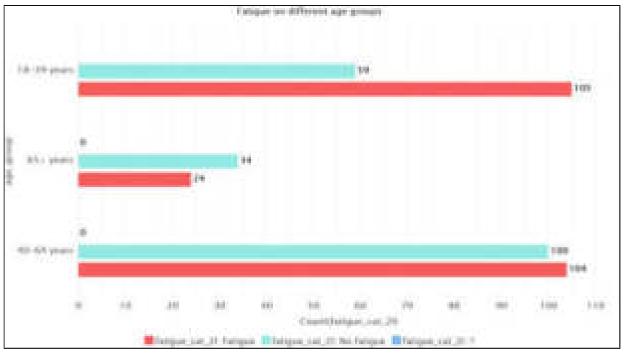

Fig. 6.7: Age group based fatigue

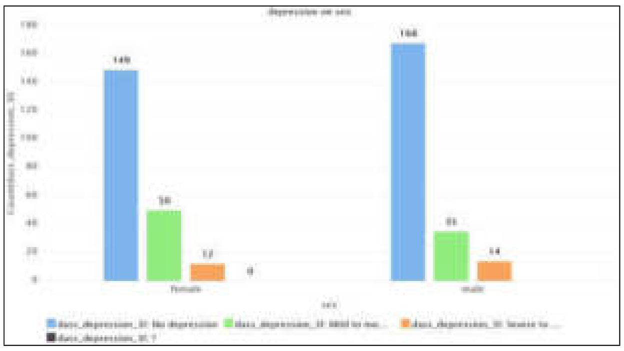

Fig. 6.8: Depression versus sex

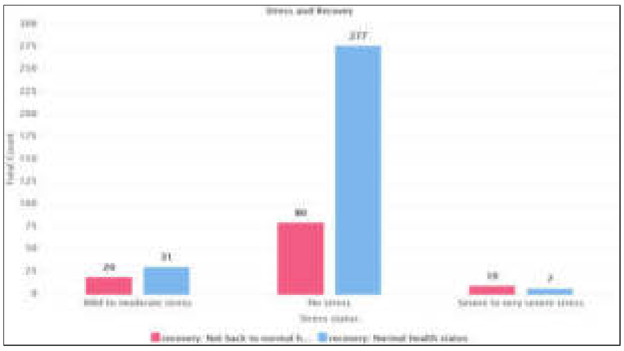

Fig. 6.9: Stress and recovery

data analysis but the trend in that the patients develop complications in the Post-Covid which is a matter of concern for all of us. It is therefore, a need of the hour to identify the parameters which should be monitored continuously in the Post-Covid era. The data has to be updated continuously and most importantly on regular basis. There should be data analysis to check the patients regularly and continuous if needed.

8. Conclusion. The medical IoT became the saviour in the Post-Covid and life situation. The remote health monitoring is possible through the M-IoT wearables but it also requires healthcare attention in real time to avoid any eventuality. The data aggregated from a patient remotely need to be analysed continuously and that is possible only through expert analysis and recommendations using the ML analytics. The proposed framework in the paper will help to offload a major portion of health monitoring for such patients from the overloaded medical staffs. Although the monitoring will be continuous However, the number of cases which require physical medical attention from the doctors and other medical staff will be lessen. The proposed work will be more required in the rural India where we have limited medical facilities as compared to the urban India.

REFERENCES

[1] *Coronavirus — who.int.* `https://www.who.int/health-topics/coronavirus#tab=tab_1}`. [Accessed 01-Dec-2022].

[2] *Coronavirus disease (COVID-19): Post COVID-19 condition — who.int.* `https://www.who.int/news-room/questions-and-}answers/item/coronavirus-disease-(covid-19)-post-covid-19-condition#:~:text=The`most common symptoms of,of breath or difficulty breathing. [Accessed 01-Dec-2022].

[3] *COVID-19: Long-term effects — mayoclinic.org.* `https://www.mayoclinic.org/diseases-conditions/coronavirus/in-}depth/coronavirus-long-term-effects/art-20490351}`. [Accessed 01-Dec-2022].

[4] *Episode #15 - Vaccine distribution — who.int.* `https://www.who.int/emergencies/diseases/novel-coronavirus-2019/media}-resources/science-in-5/episode-15---vaccine-distribution}`. [Accessed 01-Dec-2022].

[5] *Episode #47 - Post COVID-19 condition — who.int.* `https://www.who.int/emergencies/diseases/novel-coronavirus-2019/}media-resources/science-in-5/episode-47---post-covid-19-condition#:~:text=So C some of the most,have been}reported in patients.}` [Accessed 01-Dec-2022].

[6] *IOT Architecture: 3 Layers, 4 Stages Explained — zibtek.com.* `https://www.zibtek.com/blog/iot-architecture/#:~:text=}The three layers of IoT architecture&text=It proposes three layers Perception Network and Application.&}text=This is the physical layer,the need of the project.}` [Accessed 02-Dec-2022].

[7] *Post-COVID-19 syndrome among symptomatic COVID-19 patients: A prospective cohort study in a tertiary care center of Bangladesh — journals.plos.org.* `https://journals.plos.org/}` \verbplosone/article/figure?id=10.1371/journal.pone.0249644.g002. [Accessed 01-Dec-2022].

[8] F. M. Albagmi, A. Alansari, D. S. Al Shawan, H. Y. AlNujaidi, and S. O. Olatunji, *Prediction of generalized anxiety levels during the covid-19 pandemic: A machine learning-based modeling approach*, Informatics in Medicine Unlocked, 28 (2022), p. 100854.

[9] S. Alhomdy, F. Thabit, F. H. Abdulrazzak, A. Haldorai, and S. Jagtap, *The role of cloud computing technology: A savior to fight the lockdown in covid 19 crisis, the benefits, characteristics and applications*, International Journal of Intelligent Networks, 2 (2021), pp. 166–174.

[10] J. T. Behrens, *Principles and procedures of exploratory data analysis.*, Psychological Methods, 2 (1997), p. 131.

[11] F. Beltrán-Chávez, F. Mata-Rivera, M. Rivero, M. Torres-Ruiz, R. Zagal-Flores, G. Guzmán, and R. Quintero, *M-health system for cardiac and covid patient monitoring using body sensor networks and machine learning*, in Digital Innovation for Healthcare in COVID-19 Pandemic, Elsevier, 2022, pp. 217–244.

[12] J. Born, D. Beymer, D. Rajan, A. Coy, V. V. Mukherjee, M. Manica, P. Prasanna, D. Ballah, M. Guindy, D. Shaham, et al., *On the role of artificial intelligence in medical imaging of covid-19*, Patterns, 2 (2021), p. 100269.

[13] S. Boussen, P.-Y. Cordier, A. Malet, P. Simeone, S. Cataldi, C. Vaisse, X. Roche, A. Castelli, M. Assal, G. Pepin, et al., *Triage and monitoring of covid-19 patients in intensive care using unsupervised machine learning*, Computers in Biology and Medicine, 142 (2022), p. 105192.

[14] K. Cresswell, R. Williams, and A. Sheikh, *Using cloud technology in health care during the covid-19 pandemic*, The Lancet Digital Health, 3 (2021), pp. e4–e5.

[15] J. Daniel, A. Sargolzaei, M. Abdelghani, S. Sargolzaei, and B. Amaba, *Blockchain technology, cognitive computing, and healthcare innovations*, J. Adv. Inf. Technol, 8 (2017).

[16] N. Deepa, J. S. Priya, and T. Devi, *Towards applying internet of things and machine learning for the risk prediction of covid-19 in pandemic situation using naive bayes classifier for improving accuracy*, Materials Today: Proceedings, (2022).

[17] C. Delgado-Alonso, M. Valles-Salgado, A. Delgado-Álvarez, M. Yus, N. Gómez-Ruiz, M. Jorquera, C. Polidura, M. J. Gil, A. Marcos, J. Matías-Guiu, et al., *Cognitive dysfunction associated with covid-19: A comprehensive neuropsychological study*, Journal of Psychiatric Research, 150 (2022), pp. 40–46.

[18] M. Edeh, E. Otto, N. Richard-Nnabu, S. Ugboaja, C. Umoke, and D. Omachi, *Potential of internet of things and semantic web technologies in the health sector*, Nigerian Journal of Biotechnology, 38 (2021), pp. 73–83.

[19] C. Felix, A. V. Pandey, and E. Bertini, *Texttile: An interactive visualization tool for seamless exploratory analysis of structured data and unstructured text*, IEEE transactions on visualization and computer graphics, 23 (2016), pp. 161–170.

[20] M. A. Ganaie, M. Hu, et al., *Ensemble deep learning: A review*, arXiv preprint arXiv:2104.02395, (2021).

[21] S. Ghouali, E. Onyema, M. Guellil, M. A. Wajid, O. Clare, W. Cherifi, and M. Feham, *Artificial intelligence-based teleopthalmology application for diagnosis of diabetics retinopathy*, IEEE Open Journal of Engineering in Medicine and Biology, (2022).

[22] H. B. Hassen, N. Ayari, and B. Hamdi, *A home hospitalization system based on the internet of things, fog computing and cloud computing*, Informatics in Medicine Unlocked, 20 (2020), p. 100368.

[23] V. Hemamalini, L. Anand, S. Nachiyappan, S. Geeitha, V. R. Motupalli, R. Kumar, A. Ahilan, and M. Rajesh, *Integrating bio medical sensors in detecting hidden signatures of covid-19 with artificial intelligence*, Measurement, 194 (2022), p. 111054.

[24] S. A. Hussain, N. Al Bassam, A. Zayegh, and S. Al Ghawi, *Prediction and evaluation of healthy and unhealthy status of*

covid-19 patients using wearable device prototype data, MethodsX, 9 (2022), p. 101618.

[25] C. Ieracitano, N. Mammone, M. Versaci, G. Varone, A.-R. Ali, A. Armentano, G. Calabrese, A. Ferrarelli, L. Turano, C. Tebala, et al., *A fuzzy-enhanced deep learning approach for early detection of covid-19 pneumonia from portable chest x-ray images*, Neurocomputing, 481 (2022), pp. 202–215.

[26] N. Y. Khanday and S. A. Sofi, *Deep insight: Convolutional neural network and its applications for covid-19 prognosis*, Biomedical Signal Processing and Control, 69 (2021), p. 102814.

[27] A. Kumari, S. Tanwar, S. Tyagi, and N. Kumar, *Fog computing for healthcare 4.0 environment: Opportunities and challenges*, Computers & Electrical Engineering, 72 (2018), pp. 1–13.

[28] Y. Kuvvetli, M. Deveci, T. Paksoy, and H. Garg, *A predictive analytics model for covid-19 pandemic using artificial neural networks*, Decision Analytics Journal, 1 (2021), p. 100007.

[29] K. X. Lim, Y.-T. Chen, K.-M. Chiu, and F. M. Hung, *Rush hour: Transform a modern hotel into cloud-based virtual ward care center within 80 hours under covid-19 pandemic. far eastern memorialhospital's experience*, Journal of the Formosan Medical Association, 121 (2022), p. 868.

[30] G. B. Mohammad and S. Shitharth, *Wireless sensor network and iot based systems for healthcare application*, Materials Today: Proceedings, (2021).

[31] R. Moorthy, S. Udupa, S. A. Bhagavath, V. Rao, et al., *Centralized and automated healthcare systems: A essential smart application post covid-19*, in 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS), IEEE, 2020, pp. 131–138.

[32] N. Mukati, N. Namdev, R. Dilip, N. Hemalatha, V. Dhiman, and B. Sahu, *Healthcare assistance to covid-19 patient using internet of things (iot) enabled technologies*, Materials today: proceedings, (2021).

[33] S. Nabavi, A. Ejmalian, M. E. Moghaddam, A. A. Abin, A. F. Frangi, M. Mohammadi, and H. S. Rad, *Medical imaging and computational image analysis in covid-19 diagnosis: A review*, Computers in Biology and Medicine, 135 (2021), p. 104605.

[34] A. I. Paganelli, P. E. Velmovitsky, P. Miranda, A. Branco, P. Alencar, D. Cowan, M. Endler, and P. P. Morita, *A conceptual iot-based early-warning architecture for remote monitoring of covid-19 patients in wards and at home*, Internet of Things, 18 (2022), p. 100399.

[35] M. Pahar, M. Klopper, R. Warren, and T. Niesler, *Covid-19 detection in cough, breath and speech using deep transfer learning and bottleneck features*, Computers in Biology and Medicine, 141 (2022), p. 105153.

[36] R. Parker, E. Strakova, and J. Janikova, *Virtualized care systems, wearable sensor-based devices, and real-time medical data analytics in covid-19 patient health prediction*, American Journal of Medical Research, 8 (2021), pp. 50–59.

[37] A. Poonia, S. Ghosh, A. Ghosh, S. B. Nath, S. K. Ghosh, and R. Buyya, *Confront: Cloud-fog-dew based monitoring framework for covid-19 management*, Internet of Things, 16 (2021), p. 100459.

[38] S. Rezayi, *Controlling vital signs of patients in emergencies by wearable smart sensors*, in Wearable Telemedicine Technology for the Healthcare Industry, Elsevier, 2022, pp. 71–86.

[39] J. F. A. Rida, *Development of a remote health care wireless sensor network based on wireless spread spectrum communication networks*, Materials Today: Proceedings, (2021).

[40] N. K. Sharma, D. K. Gautam, L. K. Sahu, and M. Khan, *First wave of covid-19 in india using iot for identification of virus*, Materials today. Proceedings, (2021).

[41] P. Singh and R. Kaur, *An integrated fog and artificial intelligence smart health framework to predict and prevent covid-19*, Global transitions, 2 (2020), pp. 283–292.

[42] A. Sreedevi, T. N. Harshitha, V. Sugumaran, and P. Shankar, *Application of cognitive computing in healthcare, cybersecurity, big data and iot: A literature review*, Information Processing & Management, 59 (2022), p. 102888.

[43] O. Taiwo and A. E. Ezugwu, *Smart healthcare support for remote patient monitoring during covid-19 quarantine*, Informatics in medicine unlocked, 20 (2020), p. 100428.

[44] S. Tuli, S. Tuli, R. Tuli, and S. S. Gill, *Predicting the growth and trend of covid-19 pandemic using machine learning and cloud computing*, Internet of Things, 11 (2020), p. 100222.

[45] A. Verma and B. Rathi, *Machine learning based predictive model and systems-level network of host-microbe interactions in post-covid-19 mucormycosis*, Microbial Pathogenesis, 162 (2022), p. 105324.

[46] D. Yacchirema and A. Chura, *Safemobility: An iot-based system for safer mobility using machine learning in the age of covid-19*, Procedia Computer Science, 184 (2021), pp. 524–531.

Edited by: Giancarlo Fortino
Received: May 22, 2022
Accepted: Apr 9, 2023

Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2023 SCPE. Volume 24, Issues 1, pp. 17–33, DOI 10.12694/scpe.v24i1.2020

MVMS: RNN BASED PRO-ACTIVE RESOURCE SCALING IN CLOUD ENVIRONMENT

RIDDHI THAKKAR[*], DHYAN THAKKAR[†], AND MADHURI BHAVSAR[‡]

Abstract. Cloud computing offers various services to its users, ranging from infrastructure, and system development environment, to software as a service over the internet. Having such promising services available over the internet consistently, it has become an ever-demanding facility. As a reliable services provider, a cloud service provider (CSP) needs to deliver its services seamlessly to users and is also required to optimally utilize the resources. Optimal resource utilization eliminates over and under-provisioning and improves the availability of cloud services. Therefore, it is a great need to have a model allowing CSP to systematize its resources to cater to customers' demands. Such a model should be computationally light and quick enough to produce effective results. In this work, a simple yet effective neural network-based resource prediction model named MVMS is proposed, which enables a CSP to predict the customerś resource demand in advance. The results show that compared to GRU, the proposed Multi-Variate Multi-Step (MVMS) model predicts the resources accurately. Thus, CSP can schedule the resources precisely and process real-time requests of users. Experiments on the bitbrains dataset indicate that the proposed MVMS resource prediction model is quick and accurate, with lower RMSE and MAE values.

Key words: Cloud Computing, RNN, Multi-step Resource Prediction, Elasticity, Multivariate Resource Prediction, Pro-Active scaling

AMS subject classifications. 68M14, 68T07

1. Introduction.

A new dawn of data eruption has taken place with the advancement in hardware and software technologies and the way such services are available to individuals or organizations. As a consequence, high demand for data computing and storage emerged. With seamless computing power, network connectivity, and storage capabilities, cloud computing has taken the lead in solving such issues by providing an on-demand, infinite pool of resources through the internet and charging them on a pay-per-usage basis.

Cloud computing has become the first choice of numerous industries, organizations, education sectors, social media, and many others. All such applications generate a big data deluge that must be processed in real-time, necessitating the elastic scaling of cloud resources. A CSP must allocate resources optimally in order to meet all demands seamlessly. Resource allocation requires to perform dynamically based on the velocity of the input data stream, which requires the elastic nature of resource scaling. Optimized resource allocation necessitates precise resource consumption prediction, which aids in proactive and real-time decision-making. The importance of resource prediction in the cloud is depicted in figure 1.1.

The volume of streaming data is highly fluctuating in a cloud environment. Thus, it has always been challenging to accurately predict resource utilization in such a dynamic environment. The existing approaches for resource prediction include statical methods and conventional neural networks. Examples of statistics-based classical time series forecasting methods include: Autoregression (AR), Moving Average (MA), Autoregressive Moving Average (ARMA), Autoregressive Integrated Moving Average (ARIMA) and its other variations, Simple Exponential Smoothing (SES), Holt's linear trend method, Holt-Winters Exponential Smoothing (HWES), and others [28]. Shyam et al. [1] utilized a bayesian model to predict long and short-term resource necessities for resource-intensive applications. Singh et al. [2] operated linear regressor (LR), ARIMA, and SVR models for predicting non-stationary workloads for web applications to reduce resource provisioning inconstancy. In their extended work [33], the authors proposed triangulation resource provisioning (TRP) approach to provide optimal resource utilization for an hourly billing cycle. Such classical approaches were unable to adequately

[*]Institute of Technology, Nirma University, Ahmedabad, India (18ftphde29@nirmauni.ac.in)
[†]Institute of Technology, Nirma University, Ahmedabad, India (20bec027@nirmauni.ac.in)
[‡]Institute of Technology, Nirma University, Ahmedabad, India (madhuri.bhavsar@nirmauni.ac.in)

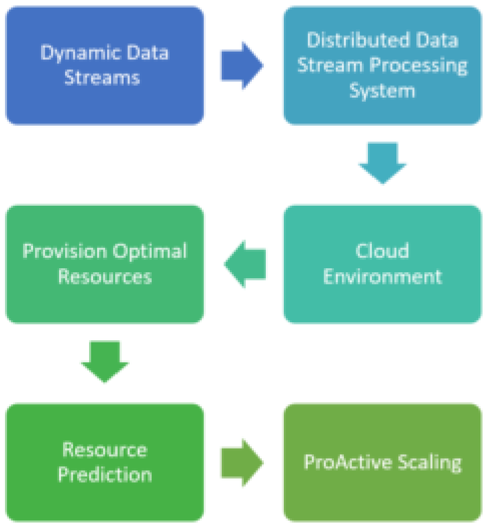

Fig. 1.1: Necessity for resource prediction

capture the non-linear patterns in time series data and were heavily reliant on the stationary nature of collected data. On the other hand, the inherent computational capabilities of neural networks allow the model to easily identify the intricate nonlinear relationship between the response variable and its predictor variables.

Cloud Service Providers need to continuously observe resource utilization to predict resource utilization, which is time-series data. Each point in time-series data is expressed by a time paired with one or more values, chronologically ordered. Due to its intrinsic nature as a time series, data may contain seasonal and nonseasonal cycles, different trends, missing values, outliers, and complex affinities among the variables. With the complex neuron architecture, deep neural networks can easily learn from time-series data and infer valuable information.

Karim [3] et al. proposed a hybrid RNN model to deal with the non-linearity of input data efficiently while foreseeing future CPU usage. The authors took into account a CPU usage parameter of historical workload. However, for multiple time-stamp predictions in advance, the proposed approach MVMS considers two parameters: CPU utilization and the CPU core. CPU cores are monitored along with the CPU utilization parameter as CPU allocation depends on the number of physical CPU cores available. Resource allocation for upcoming data streams can be accurately planned using resource prediction for multi time-stamps.

1.1. Research Contributions. Following are the major contributions of the paper:

- In this work, an RNN-based Pro-Active resource scaling model, namely MVMS, is proposed which utilizes the LSTM approach with the most promising hyperparameter settings to accurately predict multi-step resource utilization in advance.
- For precise multistep prediction, MVMS is trained over multiple variables to capture a nonlinear correlation between historical values of dependent and independent variables. Thus, the model is able to extract the complex hidden patterns from historical data and accurately predict the future resource demand to handle incoming requests with negligible errors.
- The model has been tuned by executing various combinations of the batch size and train:test dataset splitting ratio and identified the best set of values for both of them. This set of hyperparameters generates optimal CPU usage for individual VMs and helps in improving the long-term accuracy of the model.
- The proposed model is evaluated against three different evaluation metrics. The results show that compared to the GRU model, the proposed Multi-Variate Multi-Step (MVMS) resource prediction model gives less error and predicts the resources accurately. The analysis of results establishes that the proposed approach is fast at generating highly accurate resource utilization prediction for multiple

timestamps, considering the multiple variables during the prediction phase.

1.2. Organization. The structure of this paper is systematized as follows. Section 2 disuses the related work. Section 3 discusses the motivation behind this work. Section 4 elaborates on technical and conceptual details behind the proposed work. Section 5 introduces the architecture of the proposed model and discusses the dataset along with its pre-processing. Section 6 presents the experimental setup and model evaluation results. The last section 7 concludes and summarizes the complete paper along with the scope of future work.

2. Related Work. The remarkable change in the trend of resource utilization in the recent technological era has made a drastic difference in resource prediction approaches. Borkowski et al. [4] proposed ANN-based model for accurate resource provisioning prediction. As the author applied the offline machine learning approach, the model couldn't accurately respond to unseen data patterns that may appear during real-time prediction. Singh et al. [2] utilized Linear Regressor (LR), ARIMA, and SVR models for predicting non-stationary workloads for web applications to reduce resource provisioning inconstancy. LR is utilized to linearly classify the workload and in cases where the non-linear model is unable to predict better results. LR and SVR predict the slow-scale workload, wherein ARIMA is used to predict the fast-scale workload.

Chen et al. [5] proposed a graph-based deep hybrid probabilistic forecasting framework named Graph Deep Factors (GraphDF), which consists of a relational local and global model. GraphDF aims to improve prediction accuracy and computational efficiency.

Rahmanian et al. [6] proposed learning automata (LA) based, ensemble resource utilization forecasting algorithm for precise resource prediction in a virtualized environment.

Malik et al. [7] predicted multi-resource utilization with a hybrid model named FLGAPSONN consisting of GA-PSO (genetic algorithm - particle swarm optimization) and FLNN (functional link neural network). The authors utilized a combination of GA and PSO algorithms for training the model with high accuracy and FLNN for resource prediction. For multivariate resource prediction, Xu et al. [8] utilized the sliding window approach named S-MTF for converting multivariate time series data into supervised time-series data for predicting future resource usage with a modified GRU model.

Prasad and Madhuri [30] proposed a resource monitoring approach with reinforcement learning and machine learning concepts. Thonglek et al. [9] used an LSTM model to accurately predict resource allocation for a given job. Two-layered LSTM discovers the trade-off between resource allocation and usage, and CPU and memory usage. Kumar et al. [10] predicted cloud workload to reduce operational cost and SLA violations with the evolutionary algorithm and artificial neural network (ANN). The evolutionary algorithm reduces the effect of initial parameter selection. It shows a significant improvement in value selection for mutation and crossover rates.

Mason *et al.* [11] implemented an evolutionary neural networks (NN) approach to predict CPU utilization to reduce energy efficiency while dynamically scaling the resources. However, It failed to deliver better accuracy while predicting multiple future steps. To predict resource utilization, Zhu et al. [12] proposed long-term short-term memory (LSTM) encoder-decoder network with an attention mechanism. The attention mechanism gives importance to the parameters having a high impact on prediction results by assigning them more weight. However, based on the attention-based LSTM model results, the authors concluded that the attention mechanism is not having any positive or negative impact on the performance of the model. Chudasama and Bhavsar [29] proposed deep learning and queuing theory-based short-term resource prediction approach. The proposed hybrid approach utilizes the Bi-directional LSTM model to predict resources for one hour based on historical resource utilization.

Zhang et al. [13] proposed CPW-EAMC, wherein CEEMDAN-PE-Wavelet (CPW) reduces noise in input data and ENN-Attention-MLP-Context (EAMC) predicts multidimensional output for physical machines. The approach proposed in this work is not suitable for real-time resource prediction as it requires complete knowledge of the effects of each dimension on hardware. Tran et al. [14] identified highly correlated features through a fuzzy selection approach to improve prediction accuracy. In this approach, at every prediction window, the relationship between parameters is identified, which is not possible when dealing with a real-time workload.

Song et al. [15] applied the LSTM model to predict single-step host load to efficiently schedule resource allocation and optimally utilize them. Karim et al. [3] used CPU utilization for resource prediction whereas the MVMS approach operates CPU utilization, available CPU core, and nonstationary timestamp as an input to the model and predicts multi-step CPU utilization in advance. In [3] authors used the Bitbrains [16] dataset for their model training and evaluation. In this work, the same dataset is evaluated for the training and validation of the model.

Yang et al. [31] proposed a dynamic and automatic resource admission control approach that precludes resource-ceasing situations due to the unavailability of resources. The proposed approach is implemented in Hadoop YARN framework. The authors stated that this is a novel work in the said framework.

Table 2.1 lists the methodology, dataset, model evaluation metrics, and error metrics used in the literature for resource prediction.

Table 2.1: Comparative analysis of state-of-the-art techniques for resource prediction and proposed approach

Paper	**Model Used**	**Dataset**	**Model Evaluation Metrics**	**Error metrics**	**Multi variable Prediction**	**Multi Step Prediction**
[4]	ANN	Travis CI and GitHub	per-task duration of different tasks,	RMSD	×	×
[1]	Bayesian Model	AmazonEC2, Google CEData-Centers [17]	vCPU instance	MSE	×	×
[30]	Re-inforcement Learning, LSTM	Bitbrains [16]	CPU utilization, Disk read/write Throughput and Memory utilization	MAE, MSE, RMSE		×
[29]	Queuing theory, Bidirectional LSTM	Private Cloud dataset	workload of a web server	MAE, MSE, RMSE	×	
[2]	LR, ARIMA, SVR	ClarkNet, NASA	HTTP requests	MAE, MSE, RMSE, MAPE	×	×
[9]	LSTM	Googles cluster-usage trace [18]	Requested CPU and memory resource, Used CPU and memory resource	-		×
[10]	Evolutionary Algorithm	NASA, Saskatchewan	Number of HTTP requests	RMSE	×	
[31]	Novel admission control mechanism	Classic MapReduce	CPU Utilization	Makespan	×	×
[11]	Evolutionary Neural Networks (NN)	PlanetLab workload trace [19]	CPU utilization	MAE, MSE	×	
[14]	MF-LSTM	Googles cluster-usage trace [18]	CPU and Memory usage	MAE		×

[15]	LSTM	Googles cluster-usage trace [18], Traditional Distributed System [20]	CPU Usage	MSE, MSSE	×	
[3]	BHyPreC: Bi-LSTM Based Hybrid RNN	Bitbrains [16]	CPU Usage	MAE, MSE, RMSE, MAPE	×	
[5]	GraphDF: Graph Deep Factors	Google Trace, Adobe Workload Trace, Graph Construction	CPU Usage	MAPE	×	
[6]	Learning automata	CoMon project [21]	CPU Usage	RMSD, error ratio, absolute Error	×	
[7]	FLGA PSONN	Google cluster workload traces [17]	CPU and memory Usage	MAE		×
[8]	GRU	Alibaba, Google cluster workload traces [22]	CPU and memory Usage	MSE, RMSE, MAPE		×
MVMS	**Multi-variate, Multi-step LSTM**	**Bitbrains [16]**	**CPU utilization, CPU Cores**	**MAE, RMSE, MSE**		

3. Research Motivation. Real-time stream processing applications running on the cloud receive an enormous amount of data, which needs to process in no-time. Such a scenario requires an infinite pool of processing capabilities available on the easy go, which requires the CSP to predict resource demand well in advance. The following subsection describes the motivation for our work category-wise.

3.1. Multivariate analysis of time series data. The cloud performance data contains multiple metrics like timestamp, CPU utilization, memory utilization, disk read-write throughput, and network received-transmitted throughput. While performing resource prediction and analyzing its behavior, it is necessary to consider the parameters that significantly contribute to the system performance. Thus, in this work, along with CPU utilization, CPU cores are considered, as CPU allocation from a physical machine completely depends on the availability of physical CPU cores.

3.2. Multi-step resource prediction. In the cloud, resource demand changes with the number of active users and the data streams spawned by the various processes. The resource utilization history of these events contains hidden trends and seasons. Unfolding and identifying the patterns in such data will enable the prediction of the multiple time steps in advance. MVMS forecasts multistep CPU utilization by locating hidden patterns in CPU utilization and CPU core usage. Knowledge of future resource demand enables CSP to avail resources well in advance. MVMS projects the resource utilization for 12 timesteps in the future with a 5-minute interval. CSP can effectively schedule its resources for the future data stream within this time frame.

3.3. Proactive scaling. As in the cloud environment, resource demand fluctuates very rapidly. The elastic nature of the cloud allows the scaling of resources in and out as per demand. Herein, multi-step-ahead resource prediction enables CSP to schedule resources in advance. Such practice delivers a swift service experience to cloud users and facilitates a CSP to efficiently utilize the resources, which also improves the energy efficiency of the overall data center.

3.4. Non-Stationary Time-series data. Time-series data may have nonstationarity among the parameters. As evident from figure 3.1, CPU utilization data is highly fluctuating in a given time frame. Such behavior

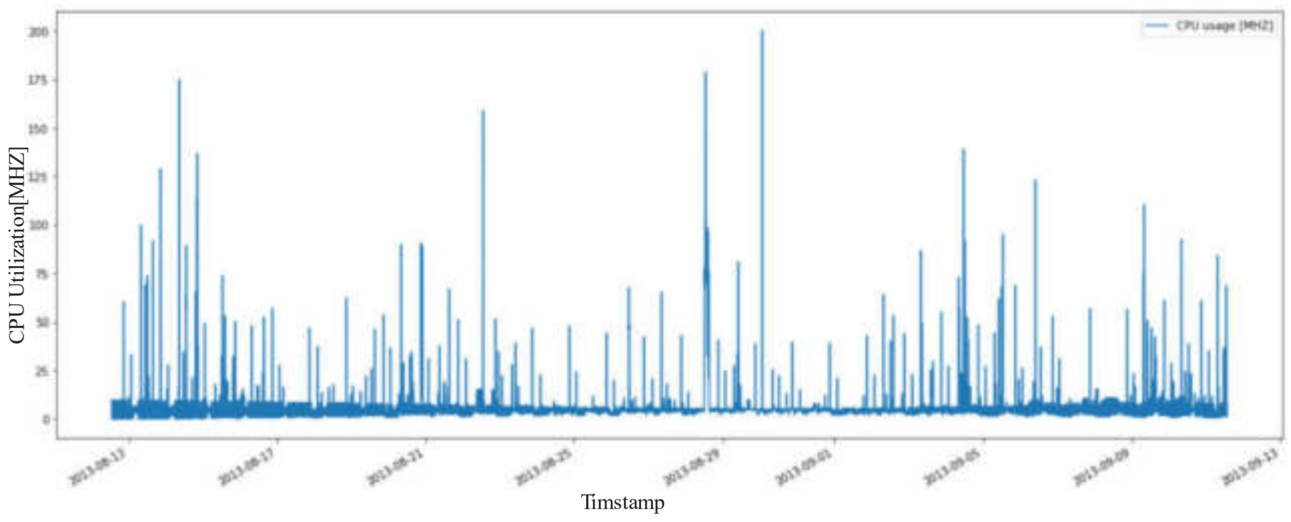

Fig. 3.1: CPU Utilization

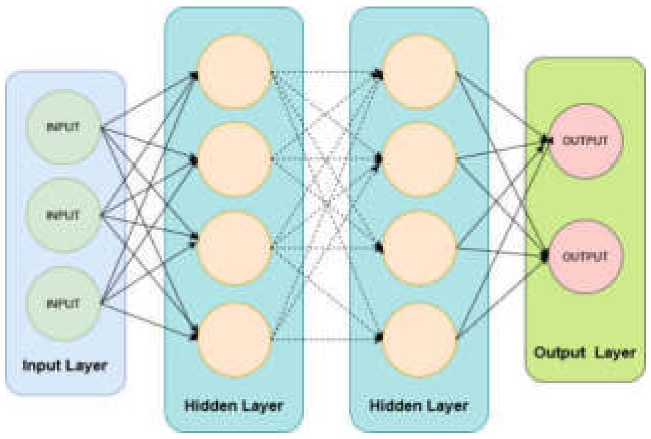

Fig. 4.1: A simplified block diagram of neural network [3]

of parameters significantly impacts the prediction efficiency and accuracy of any model. Thus, non-stationary data requires preprocessing and conversion to stationary data. In this work, the dataset is first transformed to stationary form, and then the proposed model is evaluated for resource usage prediction.

4. Background.

This section elaborates on the core concepts of the proposed model. The subsections 4.1 introduce the neural network. Subsections 4.2 and 4.3 discuss LSTM and GRU, respectively.

4.1. Neural Networks.

A model mimicking a human mind and forming a network of artificial neurons is known as a neural network (NN) or an artificial neuronal network (ANN). A neural network is a series of algorithms designed to unfold the hidden pattern in a collection of data. ANN is composed of an input layer, number of hidden layers, and an output layer. Each layer contains multiple neurons that are connected to each other, having weights and biases. Figure 4.1 depicts a simplified block diagram of a neural network with three input nodes in an input layer, two hidden layers containing four and three neurons, respectively, and the output layer containing one neuron. The neural network learns the hidden patterns from training data and updates the associated weights to accurately predict the output for newly arrived data.

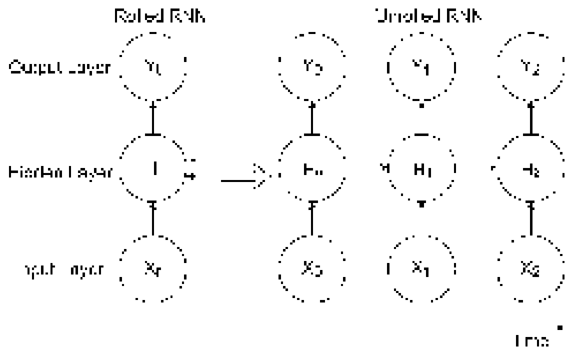

Fig. 4.2: Unrolled RNN [26]

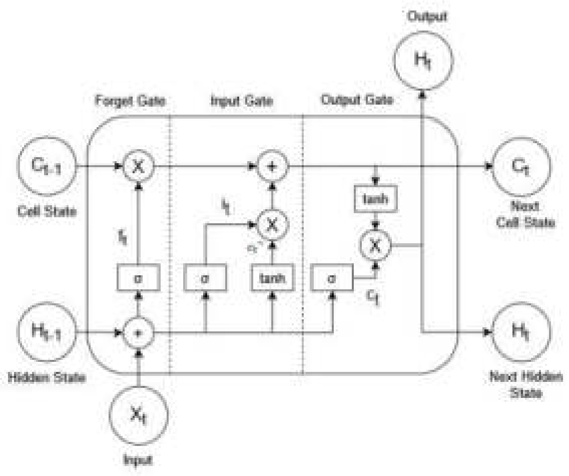

Fig. 4.3: LSTM Cell Architecture

A recurrent neural network (RNN) is a special kind of neural network that features sequential or time-series data. RNN differs by having a memory element, helping it to retain the states or information from previous sequences for predicting the output sequence. RNN is equipped with a feedback loop, where the output of the previous step is feedback to the network, which will be stored to leverage the output of the next step. Figure 4.2 depicts the unrolled RNN sequences, where X_t is the input values at time-stamp t , H_t is the hidden state values at time-stamp t, and Y_t is the output value.

There are two most common versions of RNN: Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM). This research work is formed on the foundation of LSTM architecture. We compared the results of the proposed approach with the GRU model.

4.2. Long Short-Term Memory. LSTM is a specialized neural network that accounts for memory effects in sequential data, like time series. LSTM efficiently addresses the back-propagation instability (vanishing gradient problem) found in traditional RNNs. Over longer sequences, the gradient loses parameter updates and becomes negligible in the vanishing gradient problem. LSTM addresses this issue by storing parameter updates for longer time sequence data. The LSTM cell architecture is depicted in figure 4.3.

A single LSTM cell contains three gates: input gate, output gate, and forget gate. Each of these gates contains the activation function, determining which output to preserve and which one to forget.

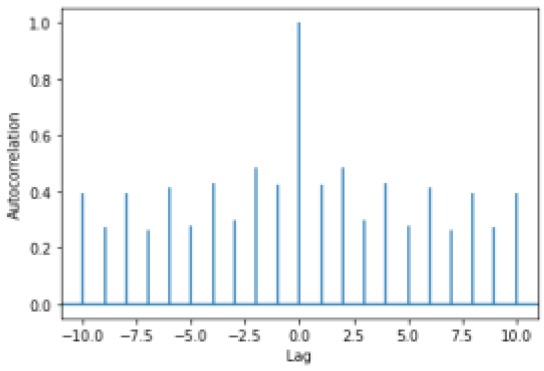

Fig. 5.1: Autocorrelation of CPU Utilization

With a new current input timestamp X_t, the LSTM cell updates its internal gates as follow [23]:

$$f_t = \sigma(W_f[H_{t-1}, X_t] + b_f) \tag{4.1}$$

$$i_t = \sigma(W_i[H_{t-1}, X_t] + b_i) \tag{4.2}$$

$$o_t = \sigma(W_o[H_{t-1}, X_t] + b_o) \tag{4.3}$$

$$c_t{}^{\sim} = \tanh(W_c[H_{t-1}, X_t] + b_c) \tag{4.4}$$

$$C_t = f_t * C_{t-1} + i_t * c_t{}^{\sim} \tag{4.5}$$

$$H_t = o_t * \tanh(C_t) \tag{4.6}$$

In MVMS, three LSTM units are stacked on top of each other. Through experimental tuning, each layer is operated with 100 memory cells.

4.3. Gated Recurrent Unit. GRU uses reset and update gate mechanism for updating the states of hidden neurons. The reset gate decides the amount of information omitted from the previous hidden state, and the update gate determines the amount of information from new input that should be used to update the hidden state. From the comparison between the performance matrices of the proposed model and GRU, it is derived that GRU is not suitable for time-series data with longer dependencies.

5. Multi-variate Multi-step Resource Prediction Model. The autocorrelation plots unfold the hidden patterns in the data and design an accurate prediction model [28]. Figure 5.1 shows the autocorrelation plot of CPU Utilization of a VM with 10 lags, visualizing the long-range dependency. The plot shows a high autocorrelation at lag 0, and then there is the alternate sequence of negative and positive spikes with a negative and positive lag. LSTM appears to be the best solution due to its inherent capacity to handle long-term dependency on volatile data. This section describes the proposed MVMS model and data set.

Figure 5.2 depicts the sequence of steps followed in the present work while developing the prediction model and producing the results.

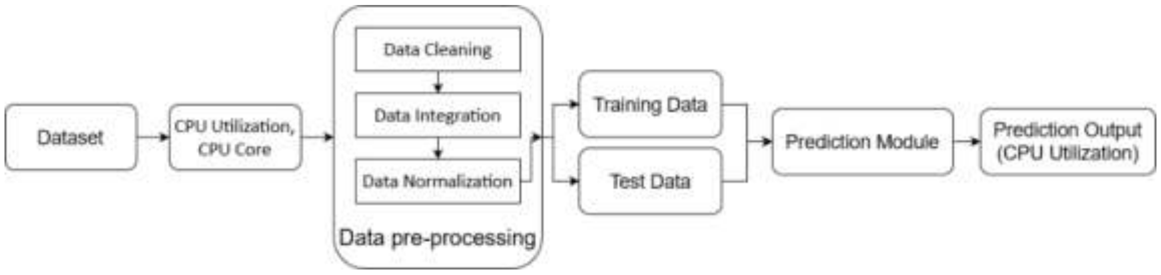

Fig. 5.2: Resource usage prediction sequence

5.1. Dataset. The proposed model is evaluated on a dataset on Bitbrains: a distributed data center, which hosts and manages the business computations [16]. The proposed model is evaluated on the fastStorage trace of the Bitbrains dataset. The fastStorage track comprises 1250 VMs interconnected via fast storage area network (SAN) storage devices. The file for each VM includes its performance metrics. The fastStorage dataset in total includes 5,446,811 CPU hours, holding 23,214 GB of memory with 5,501 cores.

After analyzing each VM file, it was observed that timestamps are not evenly distributed. Therefore, each VM file needs to be pre-proposed. Figure 5.3 shows the plot of unique timestamps in the dataset vs. the different timestamps, representing the uneven distribution of timestamps in a dataset. Section 5.2 describes the procedure followed for the dataset preprocessing. Once the dataset is evenly distributed, it is split into different train:test ratios and evaluated on the proposed model with various combinations of other hyperparameters.

5.2. Data pre-processing. The dataset contains the system information, e.g., CPU, memory, network, and disc read-write of 1250 VMs in the cloud. According to the author [16], the data is collected at a regular interval of 300 seconds. With the said time duration of data collection, there should be a total of 8640 entries for each VM. However, several timestamps are not collected at the interval of 300 seconds. These lagging or leading data entries will induce inefficiency in the training of the model with inaccurate prediction. Also, some VMs contain less than 5000 entries, while other VMs have more than 20,000 entries. Subsequently, such data entries lead to a different count of timestamps for each VM with an assorted combination of timestamps. With reference to the original work [16], it is required to have 63130 unique timestamps, starting at 1376314846, and ending at 1378906798 UNIX timestamps. With a total of 2591952 seconds and estimating the readings at 300 seconds apart, there should be a total of 2591952/300 = 8639.84 timestamps. According to [16], there are 8640 readings for each VM, and the timestamps are evenly distributed. However, the overall timestamp distribution is uneven, as depicted in figure 5.3. As there are missing and redundant entries recorded at irregular intervals, it requires processing the dataset before further usage.

The data pre-processing is carried out with the following steps:

- Data Cleaning
- Data Integration
- Data Normalization

In the first step of data pre-processing, the following iterative equation is used to even the data at time t:

$$X_t = \begin{cases} \sum_{i=0}^{i=T} \sum_{u=max(0,i-s)}^{u=min(T,i+s)} \frac{x_i - x_u}{i-u}, & \text{if } x_i \text{ is present} \\ \frac{\int_{i-u}^{i+u} x(v)\,dv\}}{2u}, & \text{if } x_i \text{ is not present} \end{cases} \tag{5.1}$$

Here, x is the uneven timestamp, X is the even timestamp

The above equation (5.1) removes duplicate readings and takes samples at an interval of 300 seconds. It returns a smoother and more consistent curve for parameters, across all VMs. With a weighted average, it synchronizes the values of all the columns for the missing timestamps found in multiple VMs. As all the columns have been converted to an even timestamp, they can now be further processed by the model.

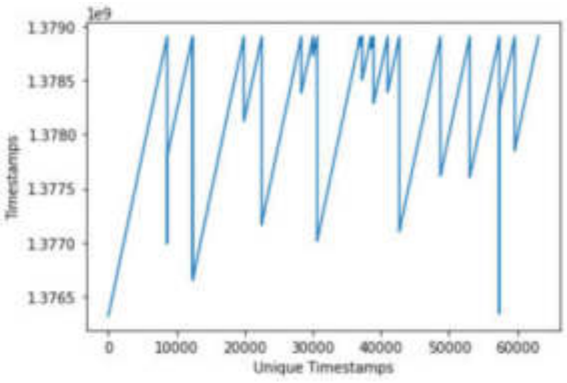

Fig. 5.3: Timestamp Distribution

Table 5.1: Hyper-parameters of the proposed model

Hyper-parameter	**Value**
LSTM Layers	3
Neurons in each LSTM Layer	100
Dense Layer	1
Neurons	12
History window	60
Prediction window	12

After the data is sanitized and integrated for 1250 VMs with 8640 entries each, in the second step, a total of 10800000, all at the synchronized timestamps with even 300-second intervals, are aggregated in a single file.

In the third step, the dataset is normalized for faster convergence and efficient performance in a range of 0-1. The MinMax scalar [24] is used for normalization:

$$X\_std = (X - X\_min)/(X\_max - X\_min) \tag{5.2}$$

$$X\_scaled = X\_std * (max - min) + min \tag{5.3}$$

Here, min, max = Feature Range

In the next step, the model is evaluated on the processed data.

5.3. Proposed Architecture and parameter selection. The LSTM layer consists of hidden layers with one or more neurons in each layer. A neuron is a signal processing unit that takes an input signal and uses an activation function to output a signal [11]. In MVMS, the inputs are the CPU cores and CPU utilization from historical data, and the output is the CPU utilization prediction for the multiple timestamps. By performing parameter sweeps manually and taking an educated guess, it was observed that a network with three stacked LSTMs with 100 neurons in each layer, followed by a single dense layer with 12 neurons, delivered the optimum performance. The observation revealed that more than three LSTM layers did not contribute to any improvement in performance and led to prolonged training time as a consequence of the additional weights to be trained. It was also learned that less than three LSTM layers are not delivering promising performance. The proposed RNN-based model, namely MVMS, is depicted in Figure 5.4, consisting of an input layer followed by three LSTM layers and a dense layer. Table 5.1 lists the hyper-parameters for the proposed model.

History and Prediction window. For the sake of brevity, rather than computing all the combinations of the history window with other parameters in the table, the history window value is borrowed from the work

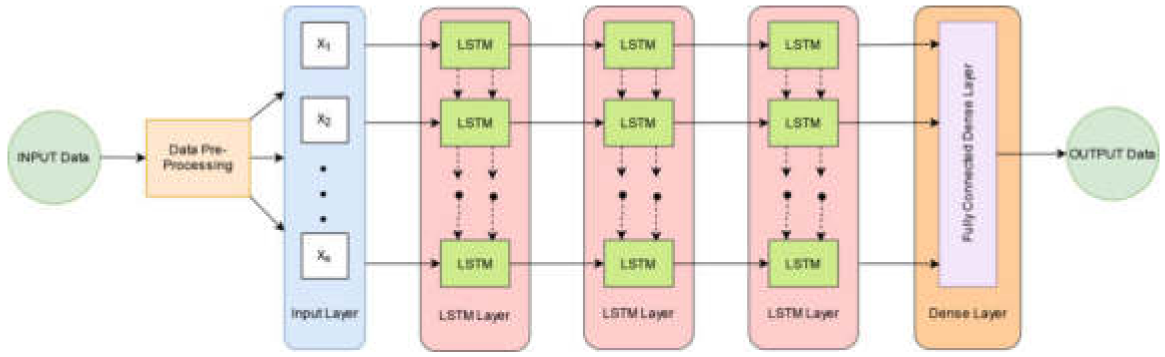

Fig. 5.4: The Proposed RNN based model

in [3] which significantly contributes to their optimal output. The proposed work predicts the CPU utilization for 12 timesteps in advance, providing significant information to CSP for preparing resources in advance to cater to data streams. Figure 5.5 depict the history window and prediction window for the proposed model. Multiple timestep forecasting directs optimal utilization of the cloud data center resources, increase the return of investment (RoI) for CSP, and also helps to reduce the carbon footprint [25].

5.4. Dataset Split Analysis. To perform a fair model evaluation and to determine whether there is overfitting or underfitting, the dataset is divided into three different train:test combinations.

- 60:40
- 70:30
- 80:20

In the above m:n split ratios, m% of the complete dataset is used for training, and the remaining n% is used for testing.

5.5. Optimal Batch Size. For the selection of batch size, it is essential to subtract the history and prediction window sizes from the total count of timestamps, for accurate multistep prediction. As per the discussion in section 5.1 and 5.2, each VM contains exactly 8640 unique timestamps with a history window of size 60 and a prediction window of 12 steps. Figure 5.5 depicts the history window and prediction window distribution. The history window and prediction windows overlap at the current value of the timestamp. Thus, adding one extra value makes the final count of timestamps for each VM 8569, which will be useful for prediction. If the batch size is not a perfect divisor of 8569, then in the last batch of the VM, there will be an overlapping of data from the next VM, which would cause the model to not comprehend the spike from the VM change, and hence reduce the performance. The factors of 8569 are 1, 11, 19, 41, 209, 451, 779, and 8569. As the smaller batch size increases the computation time as well as leads to a lower amount of information extracted per batch, and a batch size value lower than the number of neurons does not contribute more to model convergence. Therefore, factors 1, 11, 19, and 41 are not considered for the batch size, and 209, 451, 779, and 8569 were selected as different batch sizes. Now, with different combinations of train:test ratio and batch size, the proposed model is iterated 10, 25, and 50 times.

6. Results & Discussion. This section covers the evaluation of the proposed model. It first describes the metrics used to validate the performance of MVMS in section 6.1. Section 6.2 elaborates the results of the model, and section 6.3 discusses the time complexity of the model.

6.1. Evaluation Metrics. Performance metrics represent how well the model learned from trained data and how accurately it performed during testing. Therefore, to quantify the model performance and how much to improve it, mean absolute error (MAE), mean squared error (MSE), and root mean squared Error (RMSE) [32] was calculated. In this work, performance metrics illustrate the accuracy of CPU utilization predicted by MVMS and GRU. Equations 6.1, 6.2, and 6.3 are the formula used to derive the MAE, MSE, and RMSE,

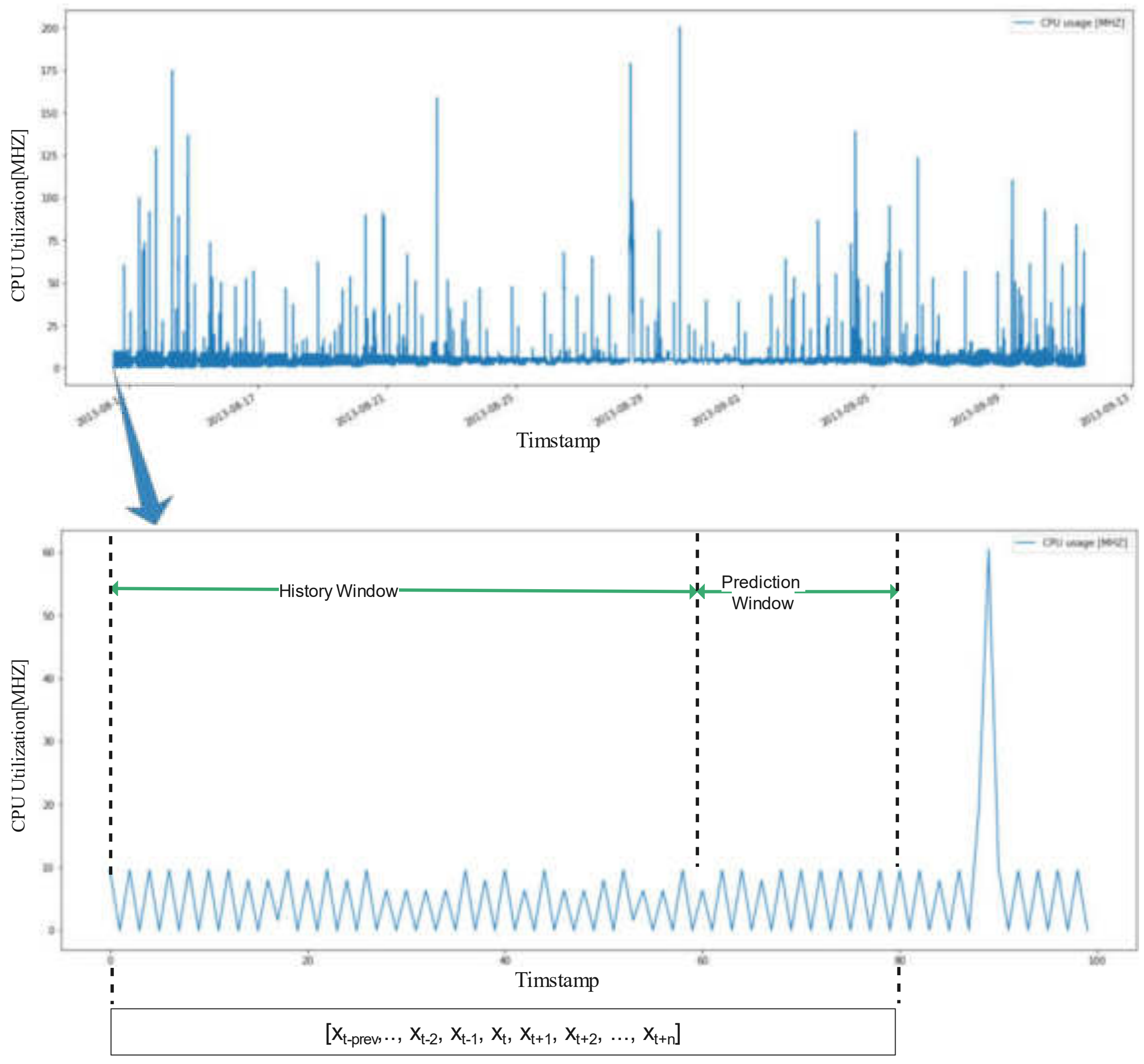

Fig. 5.5: History and Prediction Window

respectively. The lower the value of the error, the better the model performance.

$$MAE = (\frac{1}{n})\sum_{i=1}^{n}|y_i - x_i| \tag{6.1}$$

$$MSE = (\frac{1}{n})\sum_{i=1}^{n}(y_i - x_i)^2 \tag{6.2}$$

$$RMSE = \sqrt{(\frac{1}{n})\sum_{i=1}^{n}(y_i - x_i)^2} \tag{6.3}$$

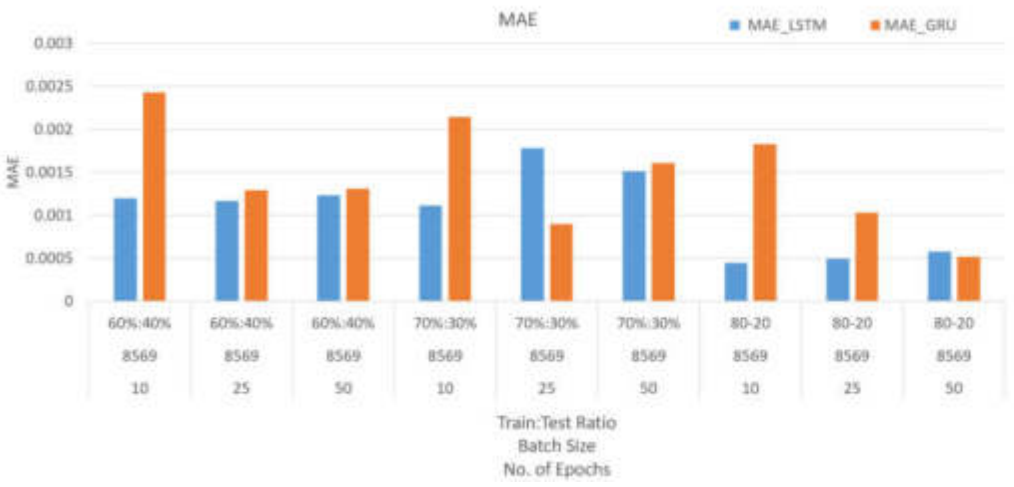

Fig. 6.1: Comparison between MAE of MVMS and GRU with 3 dataset splitting ratio

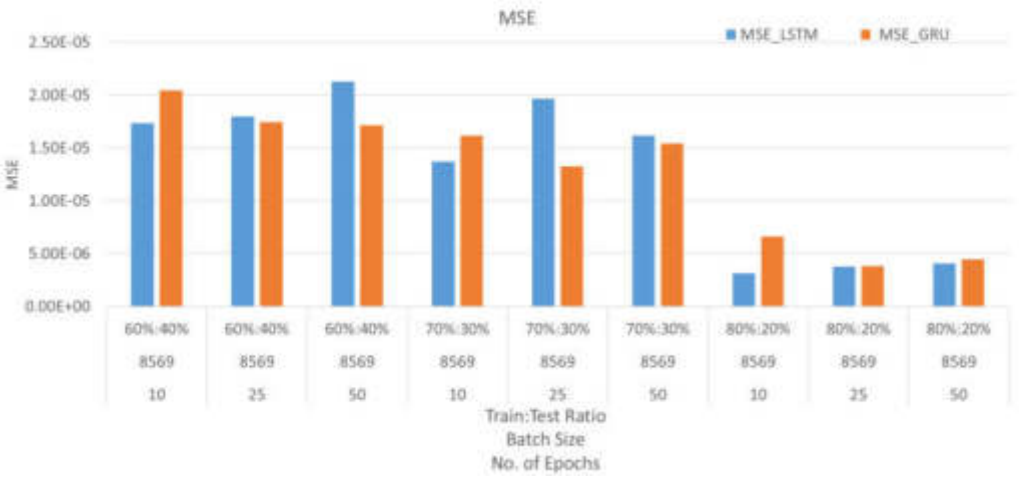

Fig. 6.2: Comparison between MSE of MVMS and GRU with 3 dataset splitting ratio

6.2. Performance Evaluation. MVMS is trained and validated using the Bitbrains dataset. The model is trained with various batch sizes and splitting ratios of a dataset to obtain the optimal prediction output.

Initially, the model is evaluated on a dataset split ratio of 60%:40%, and batch size of 8569 over 10, 25, and 50 epochs. Then, the same batch size and number of epochs are implemented over 70%:30% split ratio. A subtle improvement with a bit negative fluctuations in performance metrics is observed. As shown in figure 6.1, the average value of MAE for 70%:30% is high as compared to 60%:40%, whereas in figures 6.2, 6.3 for MSE, and RMSE it is lower than 60%:40% for MVMS and GRU.

Thus, we further updated the dataset split ratio and evaluated the model with the same parameters over 80%:20% ratio, and observed the refinement in performance parameters. By comparing the results for all three dataset splitting combinations, the model performs optimally with 80%:20% ratio as shown in figures 6.1, 6.2, and 6.3.

After identifying the optimal dataset split ratio, the model is iterated over other batch sizes 779, 451, and 209. Figures 6.4, 6.5, and 6.6 show the MAE, MSE, and RMSE for 8569, 779, 451, and 209 batch sizes

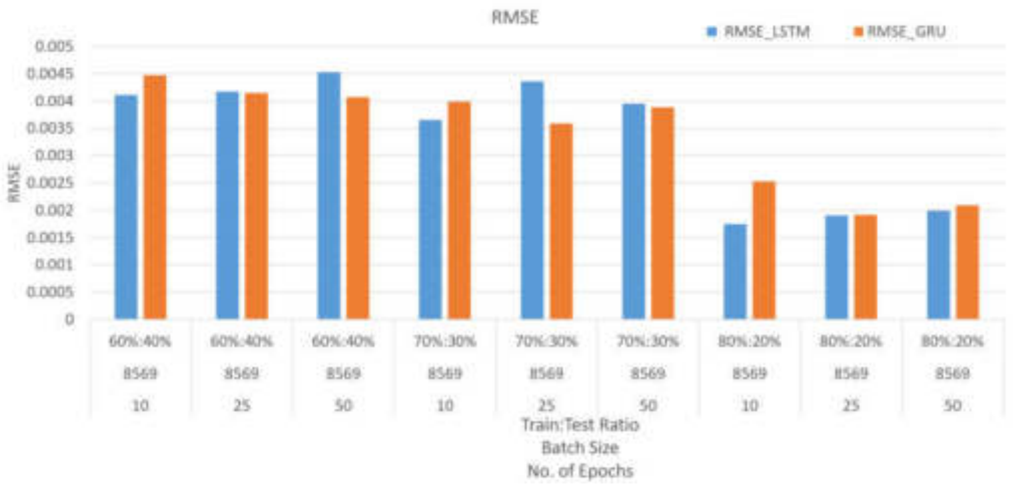

Fig. 6.3: Comparison between RMSE of MVMS and GRU with 3 dataset splitting ratio

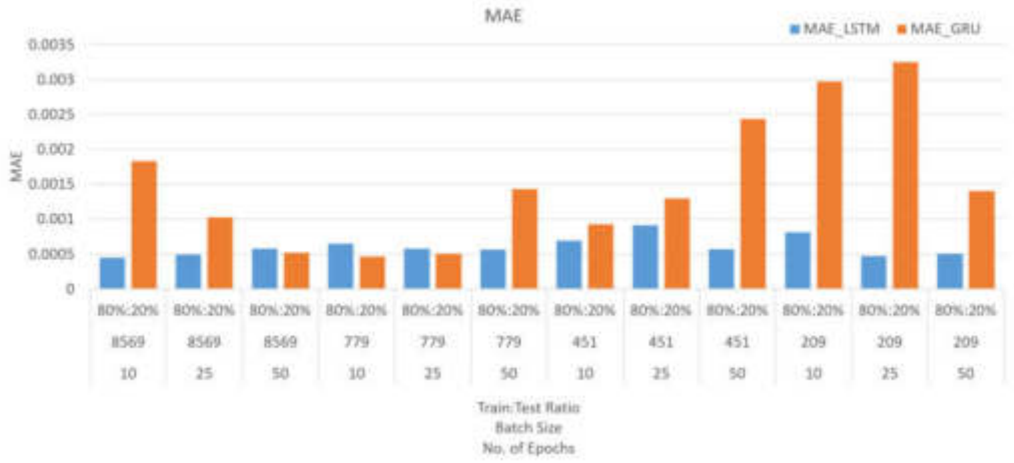

Fig. 6.4: Comparison between MAE of MVMS and GRU for all batch sizes

with 80%:20% dataset splitting ratio. It is deduced that by reducing the batch size, the model under-fits, and performance is reduced. From the performance parameters, we can notice that with the batch size of 8569 and 80%:20% ratio, the model performed optimally, and with the batch size 8569 and 70%:30% ratio, the performance of the model deteriorated. It is observed that the larger the batch size and the data split ratio, the model delivers the optimal performance, whereas with decreasing the dataset splitting ratio, the result decays.

The results show that the proposed approach performs better than the GRU model for all batch sizes and dataset splitting ratios. This work predicts the CPU utilization for multiple timestamps in the future, but GRU failed to maintain the values of hidden neurons for a longer time period and hence failed to predict the CPU utilization with high accuracy.

The proposed neural network architecture for time series prediction of multi-step, multi-variate, and multi-agent is inexpensive to train with exceptional results. Here, as the proposed model can predict the resources for individual VMs too, we referred to it as a multi-agent prediction model. The fewer layers in the proposed model reduce the computation complexity, which leads to a faster computation time while accurately converging to

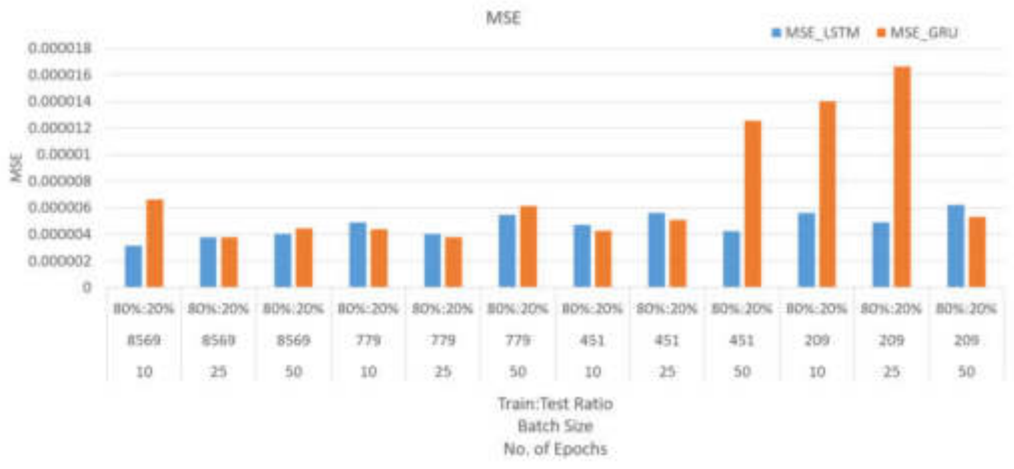

Fig. 6.5: Comparison between MSE of MVMS and GRU for all batch sizes

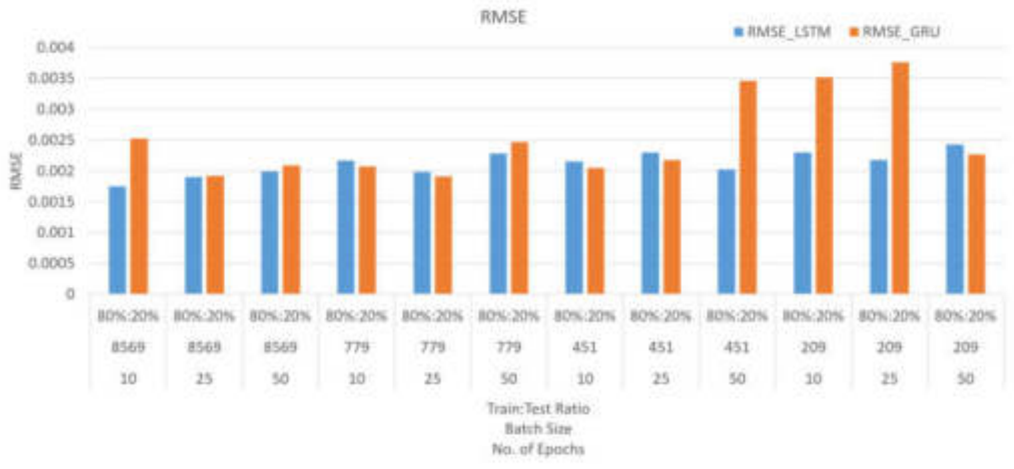

Fig. 6.6: Comparison between RMSE of MVMS and GRU for all batch sizes

the optimal results.

6.3. Training Time Complexity. The total time required to train a model depends on the number of training parameters. However, it also depends on the computing environment in which it is executed. As the experiment was conducted on high-end configuration systems, the proposed model took approximately 3 minutes for each epoch with the best configuration. Therefore, it converges in about 30 minutes to the optimal result. However, with the smaller batch size, it took an average of 12 minutes per epoch. Figure 6.7 shows the training time for various batch sizes and epochs. It is observed from the graph that the smaller the batch size, the larger the training time for any combination of dataset splitting ratios.

7. Conclusion and Future Work. The elastic nature of the cloud facilitates users with seamless access to resources on demand. A CSP needs to have enough resources available to provide the required services to the clients. Such action requires having knowledge of resource requirements in advance, allowing CSP to optimally operate the available resources. The proposed RNN-based architecture for time series prediction of multiple-variate and multi-step is inexpensive to train, with exceptional results. For validating the proposed model,

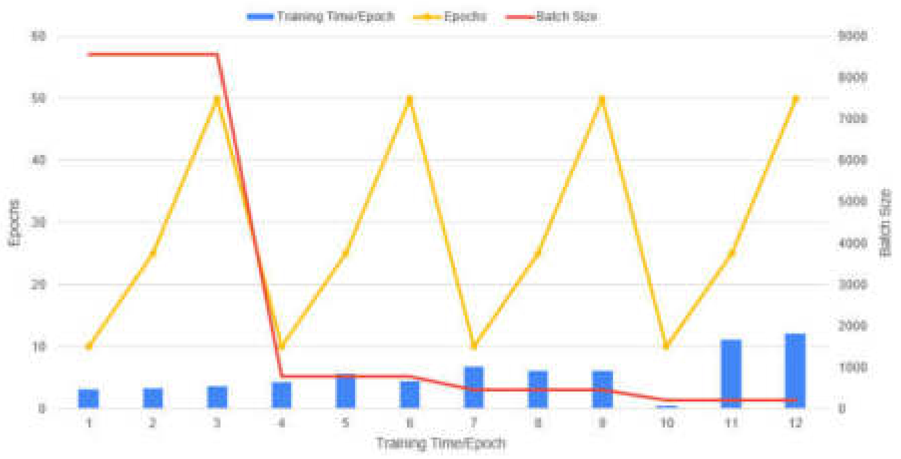

Fig. 6.7: Training Time

MAE, MSE, and RMSE are assessed. The proposed approach outperformed the GRU model by accurately predicting the resources. From the performance parameter values, it is observed that the model with more training data and a larger batch size converges very quickly to the optimal result.

In this work, the correlation between CPU cores and CPU utilization parameters is considered for resource prediction. However, based on the nature of a problem, the correlation among various parameters can be identified using machine learning approaches dynamically.

REFERENCES

[1] G. K. Shyam and S. S. Manvi, "Virtual resource prediction in cloud environment: a bayesian approach," *Journal of Network and Computer Applications*, vol. 65, pp. 144–154, 2016.

[2] P. Singh, P. Gupta, and K. Jyoti, "Tasm: technocrat arima and svr model for workload prediction of web applications in cloud," *Cluster Computing*, vol. 22, no. 2, pp. 619–633, 2019.

[3] M. E. Karim, M. M. S. Maswood, S. Das, and A. G. Alharbi, "Bhyprec: a novel bi-lstm based hybrid recurrent neural network model to predict the cpu workload of cloud virtual machine," *IEEE Access*, vol. 9, pp. 131476–131495, 2021.

[4] M. Borkowski, S. Schulte, and C. Hochreiner, "Predicting cloud resource utilization," in *Proceedings of the 9th International Conference on Utility and Cloud Computing*, pp. 37–42, 2016.

[5] H. Chen, R. A. Rossi, K. Mahadik, S. Kim, and H. Eldardiry, "Graph deep factors for forecasting with applications to cloud resource allocation," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 106–116, 2021.

[6] A. A. Rahmanian, M. Ghobaei-Arani, and S. Tofighy, "A learning automata-based ensemble resource usage prediction algorithm for cloud computing environment," *Future Generation Computer Systems*, vol. 79, pp. 54–71, 2018.

[7] S. Malik, M. Tahir, M. Sardaraz, and A. Alourani, "A resource utilization prediction model for cloud data centers using evolutionary algorithms and machine learning techniques," *Applied Sciences*, vol. 12, no. 4, p. 2160, 2022.

[8] M. Xu, C. Song, H. Wu, S. S. Gill, K. Ye, and C. Xu, "Esdnn: Deep neural network based multivariate workload prediction approach in cloud environment," *arXiv preprint arXiv:2203.02684*, 2022.

[9] K. Thonglek, K. Ichikawa, K. Takahashi, H. Iida, and C. Nakasan, "Improving resource utilization in data centers using an lstm-based prediction model," in *2019 IEEE International Conference on Cluster Computing (CLUSTER)*, pp. 1–8, IEEE, 2019.

[10] J. Kumar and A. K. Singh, "Workload prediction in cloud using artificial neural network and adaptive differential evolution," *Future Generation Computer Systems*, vol. 81, pp. 41–52, 2018.

[11] K. Mason, M. Duggan, E. Barrett, J. Duggan, and E. Howley, "Predicting host cpu utilization in the cloud using evolutionary neural networks," *Future Generation Computer Systems*, vol. 86, pp. 162–173, 2018.

[12] Y. Zhu, W. Zhang, Y. Chen, and H. Gao, "A novel approach to workload prediction using attention-based lstm encoder-decoder network in cloud environment," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, pp. 1–18, 2019.

[13] Y. Zhang, F. Liu, B. Wang, W. Lin, G. Zhong, M. Xu, and K. Li, "A multi-output prediction model for physical machine resource usage in cloud data centers," *Future Generation Computer Systems*, 2022.

[14] N. Tran, T. Nguyen, B. M. Nguyen, and G. Nguyen, "A multivariate fuzzy time series resource forecast model for clouds

using lstm and data correlation analysis," *Procedia Computer Science*, vol. 126, pp. 636–645, 2018.

[15] B. Song, Y. Yu, Y. Zhou, Z. Wang, and S. Du, "Host load prediction with long short-term memory in cloud computing," *The Journal of Supercomputing*, vol. 74, no. 12, pp. 6554–6568, 2018.

[16] S. Shen, V. Van Beek, and A. Iosup, "Statistical characterization of business-critical workloads hosted in cloud datacenters," pp. 465–474, IEEE, 2015.

[17] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in *Proceedings of the third ACM symposium on cloud computing*, pp. 1–13, 2012.

[18] J. Wilkes, "More google cluster data," *Google research blog, Nov*, 2011.

[19] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and experience*, vol. 41, no. 1, pp. 23–50, 2011.

[20] P. A. Dinda, "Load traces on unix systems.."

[21] K. Park and V. S. Pai, "Comon: a mostly-scalable monitoring system for planetlab," *ACM SIGOPS Operating Systems Review*, vol. 40, no. 1, pp. 65–74, 2006.

[22] "More google cluster data. google research blog."

[23] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[25] R. Thakkar, R. Trivedi, and M. Bhavsar, "Experimenting with energy efficient vm migration in iaas cloud: Moving towards green cloud," in *International Conference on Future Internet Technologies and Trends*, pp. 56–65, Springer, 2017.

[26] Wikipedia contributors, "Recurrent neural network — Wikipedia, the free encyclopedia.". [Online; accessed 15-May-2022].

[27] Wikipedia contributors, "Feedforward neural network — Wikipedia, the free encyclopedia.". [Online; accessed 14-May-2022].

[28] R. J. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. OTexts, 2018.

[29] Chudasama, V. & Bhavsar, M. A dynamic prediction for elastic resource allocation in hybrid cloud environment. *Scalable Computing: Practice And Experience.* **21**, 661-672 (2020)

[30] Prasad, V. & Bhavsar, M. Monitoring and prediction of SLA for IoT based cloud. *Scalable Computing: Practice And Experience.* **21**, 349-358 (2020)

[31] Yang, Z., Bhimani, J., Yao, Y., Lin, C., Wang, J., Mi, N. & Sheng, B. AutoAdmin: automatic and dynamic resource reservation admission control in Hadoop YARN clusters. *Scalable Computing: Practice And Experience.* **19**, 53-68 (2018)

[32] Holden, K., Peel, D. & Thompson, J. Economic forecasting: an introduction. (cambridge university Press,1990)

[33] Singh, P., Gupta, P. & Jyoti, K. Triangulation resource provisioning for web applications in cloud computing: A profit-aware approach. *Scalable Computing: Practice And Experience.* **20**, 207-222 (2019)

Edited by: Katarzyna Wasielewska
Received: Jun 16, 2022
Accepted: Mar 3, 2023

Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2023 SCPE. Volume 24, Issues 1, pp. 35–44, DOI 10.12694/scpe.v24i1.2041

SCALABLE DATA PROCESSING PLATFORM FOR EARTH OBSERVATION DATA REPOSITORIES

HRACHYA ASTSATRYAN[*], ARTHUR LALAYAN[†] AND GREGORY GIULIANIINSTITUTE FOR ENVIRONMENTAL SCIENCES, UNIVERSITY OF GENEVA, GENEVA, 1205, CHÂTELAINE, SWITZERLAND (GREGORY.GIULIANI@UNIGE.CH)

Abstract. Earth observation (EO) satellite data is essential to environmental monitoring. At a national and regional level, the open data cubes harness the power of satellite data by providing application programming interfaces and services to the end-users. The volume and the complexity of satellite observations are increasing, demanding novel approaches for data storing, managing, and processing. High-performance computing (HPC) and cloud platforms may improve Big EO data processing performance. However, it is necessary to consider several vital aspects for efficient and flexible EO data processing, such as the interoperability from cloud-HPC and EO data repositories, automatic provisioning and scaling of cloud-HPC resources, cost-effectiveness, support of new EO data formats and open-source packages, or linkage with data cube platforms. The article proposes a scalable EO data processing platform interoperable from cloud-HPC and EO data repositories. The platform enables linking any data repository supporting web coverage service or SpatioTemporal Asset Catalog Application Programming Interfaces (STAC-API), and any cloud or HPC resource supporting scheduling system API for providing access to the cluster backends.

Key words: Earth observation satellite data, cloud and HPC, DataCube, STAC, COG, Dask

AMS subject classifications. 68T09

1. Introduction.

Earth Observation (EO) satellite data is a crucial enabler for facilitating environmental monitoring [1]. The growing volume and complexity of EO data are making it harder to store, manage, and process [2]. The Earth Observation Data Cube (EODC) paradigm has been proposed to overcome the challenges associated with EO Big Data [3]. The Open Data Cube (ODC) is a well-known implementation of EODC helping scientists work with Analysis Ready Data (ARD) [4]. The ODC focuses on real environmental issues [23] instead of managing and solving EO Big Data challenges. National ODC systems have already been successfully implemented and provided by several countries [5] to monitor, detect and potentially resolve various types of environmental problems, like Australia [6], Switzerland [23], Brazil [22], or Armenia [7].

As a rich open-source environment harnessing satellite technology and EO data, the ODC provides communities with a diverse portfolio of advanced services and tools to make it easier to work with EO data. The essential components of ODC [9] are the following:

- datacube-core - a data analysis framework for data indexing, searching and ingesting.
- datacube-ows - interoperable access to indexed data via several Open Geospatial Consortium (OGC) web services (Web Map Service, Web Map Tile Service, Web Coverage Service).
- datacube-stats - application to calculate large-scale temporal statistics for processing the data indexed in the ODC catalog.
- datacube-explorer - software application responsible for the visualization and analysis.

Besides the enumerated benefits, ODC has some limitations and difficulties in use. Though it provides scalable computational opportunities and optimal data processing and fetching, it is hard to provision third-party computational infrastructures with ODC automatically. Newly implemented tools and platforms address the issues of optimal EO data processing and extracting in cloud-based virtualized environments. In recent years, researchers have used widely new methods to store, fast search, fetch and process the data optimally, such

[*]Institute for Informatics and Automation Problems National Academy of Sciences of Armenia 1, Paruyr Sevak str. 0014 Yerevan, Armenia (hrach@sci.am)

[†]Institute for Informatics and Automation Problems National Academy of Sciences of Armenia 1, Paruyr Sevak str. 0014 Yerevan, Armenia (arthurlalayan97@gmail.com)

as Cloud Optimized GeoTIFF (COG) [3] and the SpatioTemporal Asset Catalog (STAC) [4]. COG is a tile-based image format to support compression for reducing the transmitted data and the data transfer time. It allows users to request part of the file using an HTTP range request [3]. The STAC [8] provides particular metadata (in JSON or GeoJSON formats) about the remote sensing image to facilitate indexation and querying of EO data. Working with COGs and STAC metadata, it is evaded loading the corresponding image in memory to request, search or find an exciting part of the data. The search and query are organized in a flexible way on the level of metadata rather than the image having a bigger file size. The query finds data with the corresponding request, loads the part of the exciting area (or a particular band) from the EO data into memory, and proceeds with data processing. There are already public EO datasets with the STAC API, such as the Sentinel-2 Cloud Optimized GeoTIFFs on Amazon Web Services [6].

Since the storage, processing, and management of EO data is a complex task, several mentioned methods and tools need to be considered to overcome the hurdles associated with EO data by making data storage and processing more sensitive. Still, remote sensing data is growing daily, and scalable data processing is a real challenge [2]. Thus, there is a need to overcome the data processing obstacle, and distributed frameworks, such as Apache Spark or Dask (both with master-slave architecture), can cope with the Big EO data processing by providing scalability and efficiency [25].

The article proposes a scalable EO data processing platform interoperable from cloud-HPC and EO data repositories. It extends the functionality of ODC by bringing it to third-party computational infrastructures and providing optimized solutions. The platform enables linking any data repository supporting web coverage service or any cloud or HPC resource supporting Dask gateways.

The structure of the paper is the following. Section 2 (Motivation) describes the motivation of the work, and Section 3 (Related work) reviews several EO data processing techniques. Section 4 presents the scalable platform, Section 5 evaluates the platform, and Section 6 concludes the article.

2. Motivation. Increasing EO data requires enormous computing resources to process satellite remote sensing data. The research communities and end-users set up private HPC and cloud infrastructures or use the resources of global cloud providers to address the increasing needs of EO data processing, which becomes more complex and requires more hardware resources and flexible software. The following key performance indicators (KPI) are critical for the efficient processing of EO data:

1. **Performance and scalability.** Shared memory and distributed memory techniques and tools increase the application's performance and scalability. These techniques process EO data in parallel to reduce the execution time of the processing, as the area of interest or time-series study may enlarge the size of the processing data making the processing time longer.
2. **Interoperability from cloud-HPC and EO data repositories.** Interoperability of the EO data processing platform from the data and computing infrastructures gives several benefits. Firstly, separation from the computing infrastructure allows deploying the service using private or public HPC and cloud infrastructures. Secondly, by separating the processing platform from the data repository, the platform will become operable to extract the data from any remote sensing data repository.
3. **Automatic and fast cloud-HPC provisioning and scaling.** The predefined automated tools allow the processing of the request according to the data size without manual intervention. If the number of scaled processes is kept constant, many resources, such as CPU or energy, will be wasted even in the idle state. Therefore, it is essential to scale the processing on demand.
4. **Cost efficiency.** Using open-source solutions and deployments increases the cost-effectiveness of the processing platform. It is also essential to consider the flexibility in later updates.
5. **Support of new formats and novel solutions.** It is always essential to follow novel solutions and remote sensing image formats to make storing, fetching, or managing the data more efficient.
6. **Linkage with DC.** The EO data processing platform linkage with the Data Cubes (DC) is critical, as DC provides many vital tools and services to index and ingest data. Moreover, it provides the implementations of the OGC services, such as WMS, WCS, or WMTS.

[3]http://www.cogeo.org/
[4]http://stacspec.org/
[6]https://registry.opendata.aws/sentinel-2-l2a-cogs/

The ODC deployment on a single server is simple compared to the linkage with any HPC cloud infrastructure, considering KPIs. However, creating a scalable processing platform considering all the mentioned KPIs is challenging, as it needs to combine different architectures efficiently. Therefore, the national DCs (Australia, Armenia, Switzerland, or Brazil) rely on customized solutions using HPC infrastructures.

3. Related work. EO provides essential data for environmental monitoring, whereas the ODC offers various tools and services for processing remote sensing data and extracting useful information. Several research works discuss the use cases of ODC in monitoring different kinds of environmental issues on a local or global scale. The focus of this section is to examine the novel services and their limitations considering defined KPIs.

Many research projects utilize the EO satellite data for environmental monitoring and assessment, such as creating an analytical platform for weather data visualization [28] or crop yield estimation [29] using satellite image processing. Nevertheless, in the research works, the processing of EO data is carried out without considering the data processing performance considering novel data formats, solutions, and scalability. In addition, the absence of the DC is noticed, as this could facilitate the work of collecting data, processing, and visualizing them in both cases, therefore, none of the KPIs is considered.

The ODC services primarily rely on the computational facilities of the ODC server, where the actual computations are carried out. Swiss DC uses ODC [10] for efficient EO data analysis to monitor land degradation on a national scale, while [11] for forest monitoring of eastern Taiwan. The African ODC [12] provides a portfolio of services, such as crop phenology monitoring in Ghana, forest monitoring in Senegal, mangroves in Sierra Leone, and land degradation monitoring in Tanzania. The researchers [13] use ODC to monitor changes such as deforestation, urbanization, and coast evolution. Similar services are available in the Armenian ODC, which provides the air temperature forecast service validated for the Ararat Valley [16] and a shoreline delineation service with the case study for the Lake Sevan [24]. The works mentioned above use the ODC without considering KPIs 1, 2, and 3.

The authors [14] suggest a new DC on Demand approach to overcome the complexities in customizing the ODC per the user demand by generating an ODC instance virtually to address the user request (particular area, satellite type, and the observation time). However, the users may receive scalable computational resources per request by scaling the DC instance per user demand without considering KPIs 1 and 2.

The utilization of distributed computing frameworks is a natural choice to overcome the ODC scalability limitation [25, 15]. For instance, a pure python framework Dask enables running the Pandas or NumPy data frames locally or on a computational cluster relying on the master-slave architecture. The authors [25] use ODC and Dask to create a spatial data analysis tool for the scalable processing of EO data. The authors use a fixed number of worker nodes rather than on-demand worker node scaling without considering KPI 3.

The usage of Big Data frameworks, such as Apache Hadoop or Spark, is also quite promising and widely implemented, as the frameworks can provide efficient and distributed environments for big data processing [17]. The paper [18] shows the effectiveness of an adaptive Spark-based remote sensing data on-demand processing method on the cloud, with the data storage Hadoop Distributed File System (HDFS), which is more efficient in means of execution time than Hadoop. This method has limitations, as virtual machines are used instead of lightweight containerized images. Container-based virtualization saves resources and reduces the overhead of hypervisor-based virtualization; hence, the containers are faster than the virtual machines. Therefore, containers instead of virtual machines may reduce the necessary resources and provide faster scaling. Furthermore, writing code in Spark for remote sensing data processing is complex without using already ready-made and open-source libraries. Besides, Spark requires adjusting the programming code with the environment. The papers [19, 20] use Spark-on-Kubernetes deployed on a cloud to overcome the mentioned issues. In [19], researchers suggest a task scheduling mechanism to change the worker pods dynamically. GeoPySpark, based on Spark, provides a scalable remote sensing data processing system, where HDFS is deployed on nodes of Kubernetes to diminish data transfer between workers [20]. The limitation of the latest three works is the absence of linkage with DC, not interoperable from data repositories, as they either use HDFS or external storage. Hence, they can only process data from remote sources if they consider KPIs 2 and 6.

Serverless platforms provide only the code for EO data processing and transfer control of the server to the service provider. The Amazon Web Services (AWS) Lambda-based platform processes remote sensing images on a serverless platform [21] by splitting the image into small tiles and simulating using a Lambda function.

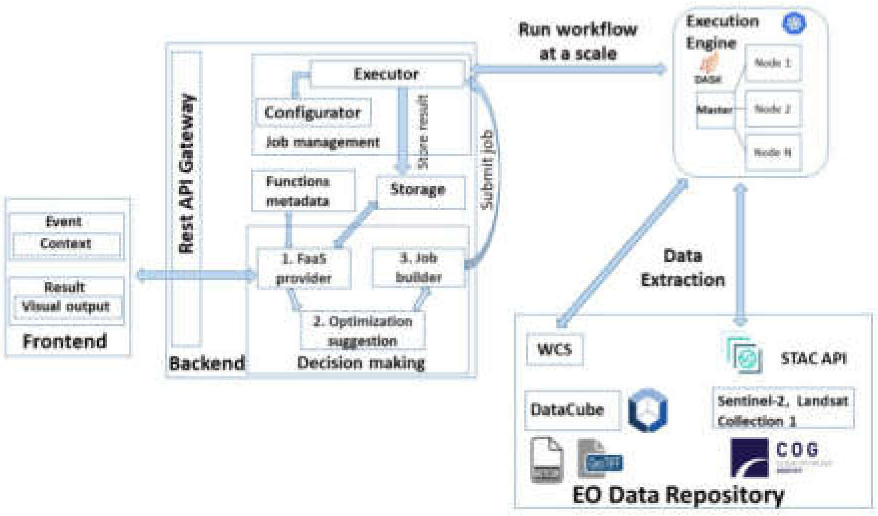

Fig. 4.1: The structure of the scalable data processing platform.

The authors also show that a novel Function as a service (FaaS) approach is faster than Spark and Ray. The limitation of this approach is that it depends on AWS lambda (KPI 2). Moreover, all implementations of FaaS will hinge on some product or cloud provider. Therefore, this work did not consider KPIs 2, 4, and 6. Besides the mentioned limitations, all mentioned works did not consider novel data formats and solutions (KPI 5). The paper aims to suggest a scalable platform to consider all mentioned limitations.

4. Architecture. As shown in Fig. 4.1, a novel scalable platform [9] is suggested to satisfy the KPI requirements, consisting of Frontend, Backend, Execution Engine, and Data Repository modules.

The suggested scalable EO data processing platform consisting of several layers hides the complexity of EO data storing, processing, and visualizing from a user. Performance and scalability (KPI 1) is obtained through the Execution Engine module, which can use any cloud or HPC resources supporting Dask gateways. Performance improvements are achieved by scaling the number of compute nodes for a Dask cluster and distributing data processing between the nodes. The data repositories and computational resources are separated to provide interoperability from cloud-HPC and EO data repositories (KPI 2). The platform is implemented using open-source solutions and deployments, considering new formats (COGs) and novel STAC API solution (KPI 5), and the possibility of fetching the data from the DC (KPI 6) using the WCS service to ensure the cost-efficiency (KPI 4). The optimization suggestion module ensures automatic and fast cloud-HPC provisioning and scaling (KPI 3), providing computing resources on demand based on the size of the input data.

The workflow is as follows: a user request to the Backend using Frontend, and the decision-making component first does its job, then it refers to the job management module, which in turn refers to the Execution Engine module, and the latter fetches data from Data Repositories, processes according to the request, stores the result and sends back as the response to the user.

4.1. Frontend. The Frontend module provides a user interface to interact with the Backend smoothly. The module supports asynchronous requests to submit event-driven workflows without waiting to complete existing ones. It offers quick and easy access to the resources by hiding the complexity of EO data processing and visualization from a user. The geoprocessing job is directly submitted to the scalable framework via a rich web-based interface. The Frontend module gets the result and visualizes it on a map. The Frontend query contains the area of interest, period, and function with its arguments in the request body. After the selection,

[9]https://github.com/arthurlalayan/eo-platform

Table 4.1: Parameters of a request body.

Request parameter	Description
Function name	A function that will be processed
Area	FeatureCollection in JSON format. It is a polygon containing the coordinates of the polygon points.
Optional arguments	Optional function arguments, such as the coordinate system (Geodetic Parameter Dataset:4326) for projecting or the data repository name from which the data will be extracted.

the Frontend calls the Backend using the POST method of the Hypertext Transfer Protocol (HTTP) and waits for the response. The information of a request body is shown in Table 4.1.

The users select an area of interest by drawing a polygon on the map. Then, the area is converted into a FeatureCollection object in the JSON format considering the OGC standard. The Backend module returns the uniform resource locator (URL) to download or visualize if the processing result is an image. In the visualization case, the WMS or WMTS services are accessible through the datacube-ows package. The dynamic tiling provided by titiler[7] optimizes large-scale visualization workflows by creating a tile server that can access raw data, scale, reproject, or encode the data into an image.

4.2. Backend. The Backend provides RESTful (Representational State Transfer) API to process the Frontend requests. It consists of decision-making and job management components.

The FaaS provider module of the decision-making component first checks whether the system supports the user's request and the function metadata component contains supported functions, processing predefined types, and the code. The module examines if the request has been processed in the past and the result will be retrieved without processing if it has already been processed. The checking functionality is based on the storage, which couples and stores request data and the processing results. The data will be processed by considering all tiles that match the user request area, and the last check will be considered if the tile was processed in the past.

Then, the complexity of the data processing is evaluated to determine the characteristics of the computational resources. The optimization module provides the number of computational nodes considering the input data size and the chunk size affecting the simulation speedup. The module relies on the adaptive scaling of Dask to balance the performance and resources by receiving a minimum and a maximum number of worker nodes as input. The minimum and the maximum numbers of worker nodes are determined considering the random-access memory (RAM) sizes of Dask nodes. Any heuristic and machine learning methods (regressions, neural networks) or multi-object optimization algorithms can be easily linked to the module to optimize performance, cost, or power consumption. The input data size is divided by the available RAM size of the worker node to calculate the minimum. At the same time, the maximum is equal to the minimum multiplied by 1.5. Rounding up provides an integer value for the maximum and minimum. For instance, in the case of the 64 GB input data size and 4 GB RAM per Dask worker node, the minimum is 64 / 4 = 16, while the maximum is 24 (16*1.5).

Finally, the job builder prepares and submits it to the job management module. The job contains information on the request body, chunk size, and the number of computing nodes provided by the optimization suggestion module. The chunk size parameter is set to auto, which means Dask will divide the data into optimal chunks.

The job management module consists of executor and configurator components. The executor receives the jobs, accesses the computing module for processing, executes the function, gets the result back, and stores it in the storage. If the computing resources are unavailable, the executor keeps jobs in the queue and executes them when resources become available. The configurator is based on the configuration files to configure the URLs and credentials of the remote Execution Engine and the Data Repositories. Therefore, it provides the

[7]https://developmentseed.org/titiler/

Table 4.2: Configuration parameters.

Request parameter	Description
URL of the Execution Engine module	The URL of a Dask Gateway
Credential	Encrypted credential for the Dask gateway
Data repositories	Contains the list of data repositories in which each repository has a name, URL and type (either WCS or STAC API)

flexibility to connect a remotely accessible Execution Engine module and later extract data from different data repositories. The configuration files are in JSON format and contain the information provided in Table 4.2.

4.3. Execution Engine. The Execution Engine module provides the scalability and the performance of the EO data processing. The module processes data in parallel using an HPC cluster and automatically scales the computing nodes on demand. A Dask framework [26] has been selected for distributed computing because it is an open-source Python library. The Dask is an easy-to-use library without configuration, setup, or complete code changes. It is possible to create a Dask cluster that can be deployed and scaled on the cloud or HPC. As the Execution Engine module is separated from the other modules, it is necessary to use, manage or scale it remotely. Dask Gateway [8] provides a secure and multi-user server for managing Dask clusters remotely.

Moreover, it can be installed on Hadoop and Kubernetes clusters or an HPC Job Queue. Hence, it is possible to connect to the Execution Engine module, scale on-demand, extract data, and execute a function on a Dask cluster parallel. Gateway provides a python API for connecting to the Dask remote cluster, and it can easily be deployed on a Kubernetes cluster, similar to other services. After the deployment, there is a need to use Gateway's python API to connect to the Gateway by providing a public host or IP, port, and authentication type and credentials. The Kerberos authentication method is chosen for platform security. After connecting to the Gateway, it is possible to create a Dask cluster by providing the number of nodes and their characteristics (such as CPU and memory). The cluster with the demanded resources and nodes will be created where the Gateway is deployed (e.g., in Kubernetes, each node will be a pod with the provided specifications). We consider Kubernetes as a Dask cluster Backend.

4.4. Data repositories. Data repositories are the storage that stores the remote sensing data. An essential goal of the repository is the availability of remote data to provide interoperability from the data repositories, which makes the platform flexible, as it can work with different data storage. A new solution for providing data using the STAC API and DC will be considered. STAC API provides a REST API to call and query EO data using only lightweight metadata in JSON format and then download it to process. In the case of the DC, there is a need to use WCS, the implementation of which has ODC, to make data open on the Internet. Therefore, the suggested platform is flexible and can be used with WCS and STAC API. The recommended platform considers Armenian DC [7] with the WCS on top of it and Sentinel-2 Cloud-Optimized GeoTIFFs [6] as an example of data repositories. The latest provides satellite images from Sentinel-2 in COG format with STAC API. The data from Armenian DC stored from several satellites, such as Landsat and Sentinel, is being used for monitoring various environmental problems [7, 16, 24]. The WCS is installed on the top of the Armenian DC using the datacube-ows package.

5. Platform evaluation. As a web application, the Frontend module provides a flexible user interface based on React JS framework and the open-source Leaflet library. The module makes it possible to process, analyze and visualize EO data by hiding the complexity of EO data processing. The workflow begins by selecting an exciting area as a rectangle on a map, selecting the period, and an already implemented function to process the EO data (see Fig. 5.1).

In the next stage, the Frontend interacts with the Backend using the Rest API, a middleware between the Frontend and the Backend. The platform's effectiveness is demonstrated by considering the Normalized

[8]https://gateway.dask.org/

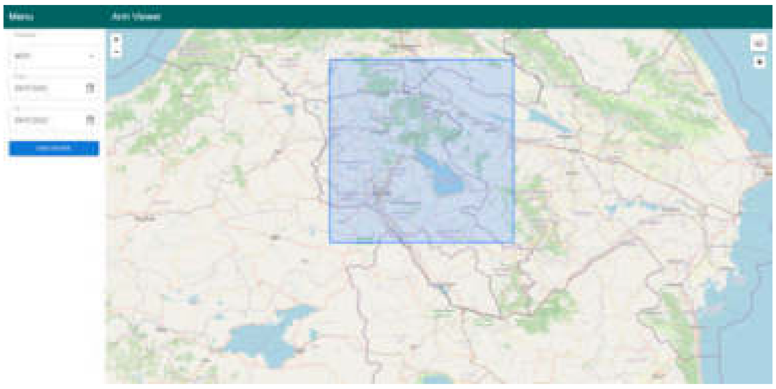

Fig. 5.1: Selecting an interesting area, period, and function in Frontend.

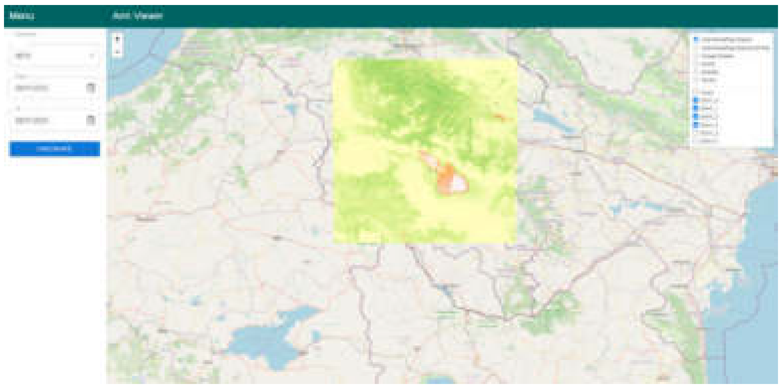

Fig. 5.2: NDVI result visualized on the map.

Difference Vegetation Index (NDVI) as an EO data processing function, which is a graphical indicator for determining the green density of land [30]. The analysis of the index consists of matrix operations, and the formula of the NDVI is the following:

$$NDVI = \frac{NIR - RED}{NIR + RED} \tag{5.1}$$

where RED and NIR are the red and near-infrared bands correspondingly extracted by pixels from satellite images received from the European optical imaging Sentinel-2 satellite. As the platform supports new formats and novel solutions, the data can be fetched in COG formats with STAC API or netCDF and GeoTIFF formats from DCs. After the processing, the Frontend module visualizes the output on an interactive map. Fig. 5.2 shows the mean NDVI calculation of some territory of Armenia visualized on a map. The NDVI function results range from -1 to 1, where green corresponds to good vegetation conditions (values around 1) and red to the opposite (-1).

Several experiments have been carried out to evaluate the processing speedup using the scaling and flexible functionality of the Backend module. For example, the average weekly NDVI for the territory of Armenia is calculated, considering 16 GB, 32 GB, and 64 GB of the input data. The resources of the CloudLab [27] have been used for the experiments using the various configurations of Dask clusters deployed on the Kubernetes system. In this particular experiment, each worker node of the Dask cluster corresponds to a pod with a fixed size of resources, 2 cores, and 4 GB RAM on Kubernetes. During experiments 1, 2, 4, 8, 16, and 32 number of Dask worker nodes is considered to evaluate the performance improvement in the distribution of computing.

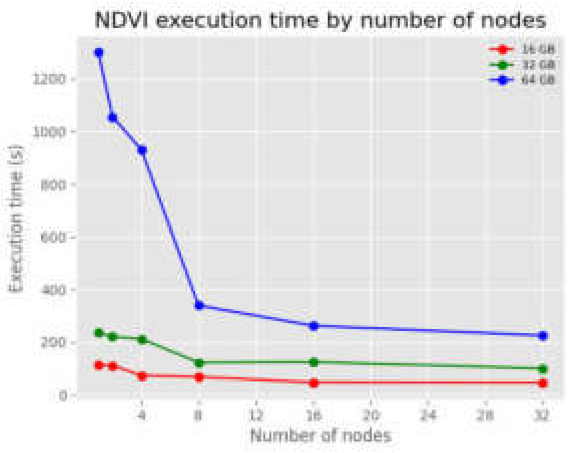

Fig. 5.3: NDVI execution times.

Fig. 5.3 shows the execution time of the NDVI with the specified number of compute nodes for the specified input sizes.

The figure shows that the EO data processing performance is boosted using the scaling of the computational resources in Dask. Scaling the number of workers from 1 to 32 improves performance by 2.44, 2.36, and 5.77 times for the 16, 32, and 64 GB of input data. The experiments show exciting behavior, as the execution time of the NDVI workflow using the optimization module show near execution times compared with the 32 nodes configuration. The optimal suggested configuration provided by the optimization service is 42%, 36%, and 22% behind compared to the run time for 16, 32, and 64 GBs of input data using 32 nodes. Increasing the input data size leads to an increase in the accuracy of the optimization module.

6. Conclusion. A scalable data processing platform has been presented for Earth observation data repositories satisfying several critical KPIs for flexible and optimal processing. The platform improves the EO data processing performance by scaling the processing on computational resources using either cloud or HPC. The platform's primary focus is interoperability from cloud-HPC and EO data repositories, enabling the use of any repository supporting WCS or STAC APIs or any cloud or HPC resource supporting Dask gateways. The platform relies on open-source solutions to provide cost-efficiency and is linked with the DC. In the future, it is planned to enrich the platform by considering various necessary functionality, evaluate the platform with different examples, and use the platform for past resolved issues with Armenian DC to boost performance. The optimal way of storing the satellite images will be studied by using compression techniques to find a trade-off between the data size, the transfer time, and the performance.

Acknowledgments. The research was supported by the University of Geneva Leading House and the State Committee of Science of the Republic of Armenia by the projects entitled "ADC4SD: Armenian Data Cube for Sustainable Development", "Self-organized Swarm of UAVs Smart cloud Platform Equipped with Multi-agent Algorithms and Systems" (Nr. 21AG-1B052) and "Remote sensing data processing methods using neural networks and deep learning to predict changes in weather phenomena" (Nr. 21SC-BRFFR-1B009).

REFERENCES

[1] Giuliani G., Egger E., Italiano J., *Essential Variables for Environmental Monitoring: What Are the Possible Contributions of Earth Observation Data Cubes?*, Data: Vol. 5, 2020, No. 4, doi: 10.3390/data5040100.
[2] Ma Y., Wu H., Wang L., *Remote sensing big data computing: Challenges and opportunities.*, Future Generation Computer Systems: Vol. 51, 2015, pp. 47-60, doi: 10.1016/j.future.2014.10.029.
[3] Giuliani G., Masó J., Mazzetti P., *Paving the Way to Increased Interoperability of Earth Observations Data Cubes.*, Data: Vol. 4, 2019, No. 113, doi: 10.3390/data4030113.

[4] Giuliani G., Chatenoux B., Bono A., *Building an Earth Observations Data Cube: lessons learned from the Swiss Data Cube (SDC) on generating Analysis Ready Data (ARD).*, Big Earth Data: Vol. 1, 2017, No. 1-2, pp. 100-117, doi: 10.1080/20964471.2017.1398903.

[5] Killough B., *Overview of the Open Data Cube Initiative.*, IGARSS 2018 - 2018 IEEE International Geoscience And Remote Sensing Symposium: 2018, pp. 8629-8632, doi: 10.1109/IGARSS.2018.8517694.

[6] Lewis A., Oliver S., Lymburner L., *The Australian Geoscience Data Cube — Foundations and lessons learned.*, Remote Sensing Of Environment: Vol. 202, 2017, pp. 276-292, doi: 10.1016/j.rse.2017.03.015.

[7] Asmaryan S., Muradyan V., Tepanosyan G., *Paving the Way towards an Armenian Data Cube.*, Data: Vol. 4, 2019, No. 3, doi: 10.3390/data4030117.

[8] Zhao Y., Yang X., Vatsavai R., *A Scalable System for Searching Large-scale Multi-sensor Remote Sensing Image Collections.*, In 2021 IEEE International Conference On Big Data (Big Data): pp. 3780-3783, doi: 10.1109/BigData52589.2021.9671679.

[9] Killough B., Siqueira A., Dyke G., *Advancements in the Open Data Cube and Analysis Ready Data — Past, Present and Future.*, In IGARSS 2020 - 2020 IEEE International Geoscience And Remote Sensing Symposium: pp. 3373-3375, doi: 10.1109/IGARSS39084.2020.9324712.

[10] Giuliani G., Chatenoux B., Benvenuti A., *Monitoring land degradation at national level using satellite Earth Observation time-series data to support SDG15 – exploring the potential of data cube.*, Big Earth Data: Vol. 4, 2020, No. 1, pp. 3-22, doi: 10.1080/20964471.2020.1711633.

[11] Cheng M., Chiou C., Chen B., *Open Data Cube (ODC) in Taiwan: The Initiative and Protocol Development.*, In IGARSS 2019 - 2019 IEEE International Geoscience And Remote Sensing Symposium: pp. 5654-5657, doi: 10.1109/IGARSS.2019.8898576.

[12] Mubea K., Killough B., Seidu O., *Africa Regional Data Cube (ARDC) is Helping Countries in Africa Report on the Sustainable Development Goals (SDGS).*, In IGARSS 2020 - 2020 IEEE International Geoscience And Remote Sensing Symposium: pp. 3379-3382, doi: 10.1109/IGARSS39084.2020.9324156.

[13] Vâjâială-Tomici C., Filip I., Pop F., *Landscape Change Monitoring using Satellite Data and Open Data Cube Platform.*, In 2020 IEEE 16th International Conference On Intelligent Computer Communication And Processing (ICCP): pp. 573-580, doi: 10.1109/ICCP51029.2020.9266254.

[14] Giuliani G., Chatenoux B., Piller T. , *Data Cube on Demand (DCoD): Generating an earth observation Data Cube anywhere in the world.*, International Journal Of Applied Earth Observation And Geoinformation: Vol. 87, 2020, pp. 102035, doi: 10.1016/j.jag.2019.102035.

[15] Gomes V., Carlos F., Queiroz G., *Accessing and Processing Brazilian Earth Observation Data Cubes with the Open Data Cube Platform.*, ISPRS Annals Of The Photogrammetry, Remote Sensing And Spatial Information Sciences: Vol. 4, 2021, pp. 153-159, doi: 10.5194/isprs-annals-V-4-2021-153-2021.

[16] Astsatryan H., Grigoryan H., Poghosyan A., *Air temperature forecasting using artificial neural network for Ararat valley.*, Earth Science Informatics: Vol. 14, 2021, doi: 10.1007/s12145-021-00583-9.

[17] Astsatryan H., Lalayan A., *Performance-efficient Recommendation and Prediction Service for Big Data frameworks focusing on Data Compression and In-memory Data Storage Indicators.*, Scalable Computing: Practice and Experience: Vol. 22, 2021, pp. 401-412. doi: 10.12694/scpe.v22i4.1945.

[18] Tan X., Di L., Zhong Y., *Spark-based adaptive Mapreduce data processing method for remote sensing imagery.*, International Journal Of Remote Sensing: Vol. 42, 2021, No. 1, pp. 191-207, doi: 10.1080/01431161.2020.1804087.

[19] Huang W., Zhou J., Zhang D., *On-the-Fly Fusion of Remotely-Sensed Big Data Using an Elastic Computing Paradigm with a Containerized Spark Engine on Kubernetes.*, Sensors: Vol. 21, 2021, No. 9, doi: 10.3390/s21092971.

[20] Guo J., Huang C., Hou J., *A Scalable Computing Resources System for Remote Sensing Big Data Processing Using GeoPySpark Based on Spark on K8s.*, Remote Sensing: Vol. 14, 2022, No. 3, doi: 10.3390/rs14030521.

[21] Yang G., Liu J., Qu M.,, *A Remote Sensing Image Processing System on Serverless Platform.* , In 2021 IEEE 45th Annual Computers, Software, And Applications Conference (COMPSAC): pp. 258-267, doi: 10.1109/COMPSAC51774.2021.00044.

[22] Ferreira K., Queiroz G., Vinhas L., *Earth Observation Data Cubes for Brazil: Requirements, Methodology and Products.*, Remote Sensing: Vol. 12, 2020, No. 24, doi: 10.3390/rs12244033.

[23] Chatenoux B., Richard J., Small D., *The Swiss data cube, analysis ready data archive using earth observations of Switzerland.*, Sci Data: Vol. 8, 2021, doi: 10.1038/s41597-021-01076-6.

[24] Astsatryan H., Grigoryan H., Abrahamyan R., *Shoreline delineation service: using an earth observation data cube and sentinel 2 images for coastal monitoring.*, Earth Science Informatics: 2022, doi: 10.1007/s12145-022-00806-7.

[25] Xu D., Ma Y., Yan J., *Spatial-feature data cube for spatiotemporal remote sensing data processing and analysis.*, Computing: Vol. 102, 2020, doi: doi.org/10.1007/s00607-018-0681-y.

[26] Rocklin M., *Parallel Computation with Blocked algorithms and Task Scheduling..*, 2015.

[27] Duplyakin D., Ricci R., Maricq A., *The Design and Operation of CloudLab.*, In Proceedings Of The USENIX Annual Technical Conference (ATC): 2019, pp. 1-14.

[28] Astsatryan H., Grigoryan H., *Weather Data Visualization and Analytical Platform.*, Scalable Computing: Practice and Experience: Vol. 19, 2018, pp. 79-86. doi: 10.12694/scpe.v19i2.1351.

[29] Anitha M., Priyanka S., *Review of Crop Yield Estimation using Machine Learning and Deep Learning Techniques.*, Scalable Computing: Practice and Experience: Vol. 23, 2022, pp. 59-80. doi: 10.12694/scpe.v23i2.2025.

[30] Huang S., Tang L., *A commentary review on the use of normalized difference vegetation index (NDVI) in the era of popular remote sensing.*, Journal of Forestry Research: Vol. 32, 2021, doi: 10.1007/s11676-020-01155-1.

Edited by: Katarzyna Wasielewska
Received: Aug 4, 2022
Accepted: Apr 4, 2023

Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2023 SCPE. Volume 24, Issues 1, pp. 45–55, DOI 10.12694/scpe.v24i1.2052

AN EFFICIENT BIO-INSPIRED ROUTING SCHEME FOR TACTICAL AD HOC NETWORKS

PIMAL KHANPARA AND SHARADA VALIVETI [*]

Abstract. Ad hoc networks are temporary networks, created mainly for applications that are infrastructure-less. Such networks and network nodes demand special characteristics like mobile nodes having dynamic topology, wireless medium, heterogeneous deployment environment, and reactive or proactive routing depending on the nature of the network which includes network parameters such as node placement, mobility model, number of participants in the network, patterns of mobility, etc. Due to these characteristics and the mobility of network nodes, the process of routing is quite challenging in the ad hoc environment, especially when the node mobility is high. Bio-inspired routing can be an effective solution to meet all the design requirements and deal with the issues of tactical ad hoc networks. Different types of nature-inspired routing mechanisms are possible to use for tactical networks. This paper proposes the design of a novel Ant Colony Optimization-based routing strategy for ad hoc networks. Ant-based algorithms are dynamic and have adaptive behavior. Hence, they are competent for routing in ad hoc networks. Our proposed routing scheme is evaluated based on the network's performance by varying different parameters. The performance of our proposed ACO-based routing approach is also compared to some existing ad hoc routing mechanisms. Different metrics in different deployment scenarios that can affect the efficiency of our proposed protocol are taken into consideration to evaluate the performance.

Key words: Bio-inspired Routing, Ant Colony Optimization, Ad hoc Networks, Hybrid Routing

AMS subject classifications. 68M14, 68T05

1. Introduction. In ad hoc networks, participating nodes are mobile and have dynamic connections with one another. Such nodes communicate information through wireless links and are organized in a decentralized manner. Participating nodes in ad hoc networks can become a part of the network or exit any time and thus form a topology, which keeps changing depending on the mobility of the participating nodes, i.e. the topology is dynamic. Hence, the position of nodes at any given time, cannot be known or assumed to be precisely available. The transmission range is fixed for every node in the network. There are no specific designated routers in ad hoc networks. To send messages to the destination nodes that are not in the transmission range, intermediate nodes act as routers and forward messages. Mobility of nodes, dynamic topology, and decentralized control are the fundamental properties of ad hoc networks. Due to these characteristics, one of the major challenging issues in ad hoc networks is routing. [18]. Participating nodes need to update their routing tables frequently, resulting in flooding control packets in the network which consume precious network resources. As a result, in ad hoc networks, it is hard to construct and maintain routes [26].

Enhancing routing process in ad hoc networks is an evergreen domain for study. As the network is decentralized, nodes themselves act as routers, as they move. So, routing is very important in ad hoc networks [15]. Routing strategies for ad hoc networks are mainly classified as Proactive (Table-driven), Reactive (On-demand), and Hybrid [1]. Table-driven routing protocols require each node to keep up-to-date route-related information. Updates are propagated through the network to modify nodes' routing tables in response to topological changes. In on-demand routing approaches, a node finds a route to its destination, only when there is a need to do so. Hybrid routing strategies integrate features of proactive and on-demand routing mechanisms [18].

The next section highlights the working of a Bio-inspired Routing strategy, Ant Colony Optimization, in a nutshell. Section III presents the proposed algorithm (MyANT) for ad hoc networks. Section IV discusses the experimental outcomes. Section V briefs the conclusions drawn, based on the experiments.

[*]Computer Science and Engineering Department, Institute of Technology, Nirma University. (pimal.khanpara@nirmauni.ac.in, sharada.valiveti@nirmauni.ac.in)

2. Related Literature. Ant Colony Optimization (ACO) [10] is a nature-inspired probabilistic approach for solving computational problems that can be mimimized to finding optimized routes.ACO's fundamental concept is based on observations of how ants choose the best path between their habitat and a food source. Artificial ants are used as agents to iteratively construct an optimized solution. ACO depends on heuristic values which are maintained in the form of pheromone trails [9]. In ACO, to find an optimal route, a group of ants moves on adjoining routes simultaneously and asynchronously. Each ant (agent) determines the next hop based on a probabilistic computation using the pheromone values available on the links and heuristic information [20]. The solution is formed progressively as the ants traverse from one hop to another hop. While traversing the route, an ant agent assesses this solution and leaves pheromone on the path. The use of the pheromone trail is to make a routing decision by the future ants [9].

Due to the escalation in the use of various mobile and electronic devices, Ad hoc wireless networks are becoming more and more common. These devices need to communicate with each other, without a designated infrastructure [17]. For providing such communication in an ad hoc environment, the concept of swarm intelligence can be considered to address issues with routing optimizations. Many routing mechanisms take benefit of this concept, i.e. AntNet [8], ARAMA [13] , and AntHocNet [7]. Dynamic topology, dependency on local information, integration of path quality to determine pheromone concentration, and support for multi-path are the fundamental features of ant-based algorithms which make them a good choice to implement routing logic in ad hoc networks [6] [23] [28].

AODV [21], DSR [14], and AntHocNet [11] are popular routing protocols, against which this proposed implementation is compared. AODV and DSR are reactive routing protocols. This work is compared against AODV, DSR, and AntHocNet. AODV and DSR are the most sought-after routing protocols, subject to modification by researchers to improve the efficiency of the routing strategy and also to embed other functionalities like intrusion detection, etc. AntHocNet is a hybrid multipath ant-based routing protocol that combines both reactive and proactive components. AntHocNet is based on a very popular ant-based routing protocol AntNet.

Ad hoc On-Demand Distance Vector routing (AODV) is a reactive routing approach for MANETs. In this protocol, three types of control packets are used for the process of routing: Route Request (RREQ), Route Reply (RREP), and Route Error (RERR). Each node maintains a routing table to store path-related information. For each data transmission request, the RREQ packets are broadcasted by the source node to establish a communication route. Upon discovery of the path, the destination issues an RREP packet which is to be traversed back to the source. RERR packets are used for route maintenance and to deal with link failures.

The Dynamic Source Routing protocol (DSR) is a reactive, and efficient routing approach designed especially for use in MANETs. DSR uses the concept of source routing technique in which the source node decides the whole sequence of nodes to be used in order to forward the packets to the destination node. The benefit of this technique is that intermediate nodes do not require to have latest routing data for forwarding the packets. There are two major components in DSR: Route Discovery and Route Maintenance. Each node maintains a route cache to store all the existing source-destination pairs [30]. This cache is used in the route discovery process to obtain the required route. If there is no path found in the cache, the algorithm initiates a route discovery phase to establish a new path.

AntHocNet is an ACO-based routing mechanism for MANETs. It is a hybrid algorithm that combines proactive and reactive elements: the protocol establishes, maintains, and enhances routes in a proactive manner following a reactive path setup phase. AntHocNet is based on the ACO optimization framework and stigmergy-driven shortest pathways following ant colony behavior.

In general, ant-based routing approaches use one or more types of ant agents for facilitating the communication in the network. Table 2.1 depicts the analysis of some classic ACO-based routing techniques. For this analysis, important factors affecting the routing performance, such as how many ants are used, how they participate in the routing process, the frequency of pheromone deposition, routing data storage structures, etc., [16] are taken into consideration.

3. Proposed Approach: MyANT. MyANT is a hybrid routing protocol for ad hoc networks that uses the concept of Ant Colony Optimization. As it is a hybrid algorithm, it contains the good features of both, the reactive and proactive routing strategies. The route setup process is done in a reactive manner while the route maintenance and improvement process are proactive. There are two kinds of mobile agents in each phase:

Table 2.1: Analysis of ACO-based Routing Approaches

Routing Protocol	**Type of Routing**	**Storage Structures**	**Next-hop Selection Parameters**	**Factors affecting Pheromone Deposition**	**Key Features**
AntHocNet [7]	Hybrid	Routing Table, Pheromone Table,	One-hop neighbors of forward ants	Hop-count and communication delay	Forward ants identify destination nodes and hop-distance
Ant Routing Algorithm for MAnet (ARAMA) [13]	Table-driven	Routing Table	Pheromone amount, lifetime of nodes, queuing delay	Route-quality	Ants compute queuing delay and route-cost
AntNet [5]	Table-driven	Routing Table	Node-queue length and pheromone amount	Amount of network traffic	Forward ants keep a track of total communication time for each destination node
Hopnet [27]	Hybrid	Zone-specific routing tables	hop-distance, amount of pheromone	Queuing delay and hop-count	Forward ants are responsible for finding the destination node along with its zone
Ant colony based Routing Algorithm (ARA) [12]	On-demand	Routing table	one-hop neighbors of forward ants	Queuing delay and hop-count	Ants consider hop-count for finding the best route
Ant Based Control (ABC) [2]	On-demand	Routing table	Network statistics	Lifetime of ants	Packet forwarding is based on shortest distance and pheromone deposited
Ad hoc Networking with Swarm Intelligence (ANSI) [22]	On-demand	Ant-decision table, routing table	Amount of pheromone and no.of hops	Communication delay and hop-count	Based on the amount of pheromone deposited on links, ants computes the link quality

Forward ants and Backward Ants. The forward ants in the reactive phase explore the network to search the paths from a source to a destination. The role of the backward ants is to establish the path information which is gathered by the forward ants. As the agents move along the path, they leave a pheromone amount starting from its source. The path for transmitting the data packets is selected based on the higher pheromone concentrations for the particular paths.

In this algorithm, routing-related data are kept and maintained in pheromone tables. The pheromone tables are maintained by each participant node i of the network. An entry Pdij in the pheromone table holds the routing information from node i to destination d over neighbor j. The routing data contains the pheromone value for every path from source node i to destination node d. In addition to pheromone tables, every network node is required to have an up-to-date neighbor table to retain information about neighboring nodes. Neighboring nodes are the nodes with which the node has a wireless link.

3.1. Route Discovery. In the initial stage of the communication process, the source node checks the entries available in its pheromone table, to find whether it has any data available, related to routing for the requested destination node. If there is no entry for the particular destination, the new route is formed through the reactive path setup process. The source node generates and transmits a packet called forward ant, which is to be forwarded to each intermediate node. The forwarding of this packet is accomplished using uni-casting if information for routing to the destination is present in the node's pheromone table; otherwise, forwarding takes place via broadcasting. All the visited nodes are stored in an array. When the forward ant is received by the destination node, a copy of it is generated which is called a backward ant. The backward ant retraces the exact route that was traversed by the forward ant but in the reverse direction. The backward ant modifies the pheromone tables based on the pheromone concentration. This helps in determining the best route among the available paths. The benefit of executing this update is that whenever another ant proceeds to find the route, it can find the optimal path easily. The pheromone value is obtained by the forward ant using the following equation:

$$PH_d^i = PH_d^i + \frac{\alpha}{T_i^d + T'^j_i}$$

where PH_d^i is the total pheromone value from source node i and destination node d. T_i^d is the total time the forward ant has travelled and T'^j_i is the time interval during which the link between node i and j is used for the connection. α is a user defined run-time parameter.

The backward ant revises the pheromone value at node k while traversing backward from node b, by using the following equation:

$$PH_b^k = PH_b^k + \frac{\alpha}{\tau}$$

Here, $\tau = T_d^i - T_i^k$.

3.2. Proactive Route Maintenance. After the construction of the first path using the on-demand path setup process, the proposed protocol initiates the proactive path maintenance process, in which it attempts to modify, expand and enhance the available routing-related data. This process executes as long as the communication session is active.

In the route maintenance process, the pheromone information, provided by the ants is broadcasted. The best available pheromone values are broadcasted periodically by all the nodes of the network. When the neighbor nodes derive the new pheromone values, they forward these new values as their periodic broadcast. The best available pheromone value is used by the forward ants in the proactive phase. The difference between reactive forward and the proactive forward ant is that the proactive forward ant is unicasted to the destination, unlike the reactive forward ant. The proactive backward ant is generated at the destination and traverses the same route covered by the forward ant from destination to source.

3.3. Managing Link Failures. Link failures can occur in ad hoc environments because of either physical changes like node mobility or unavailability of a node, or modifications that affect the topology and connections of the network, such as changing the radio range. As ad hoc networks are highly dynamic, link failures can occur frequently so the routing algorithm should be capable of dealing with such problems effectively. The proposed algorithm offers a solution for the link failures. In this algorithm, the reactive path setup process permits the source nodes to reconstruct the whole path if required. The on-demand path maintenance process lets the creation of new paths. This way, it provides backup path routing. Link failures are discovered through the disrupted transmission of either data or routing packets or using periodic short messages. These messages are transmitted by all the network nodes asynchronously after a fixed time interval. When node i gets such a message from node j, it assumes that j is its direct neighbor and marks the same in its neighbor table. It also adds a record in the pheromone table, specifying that there is a path from i to j with hop count 1 and the link between the nodes i and j is still alive. So, now node i expects a message from node j within a fixed time interval. If the message is not received, node i believes that the connection to j has expired or the link is broken.

When a node i finds that the link to its neighbor node j is broken, it deletes the entry for node j from its neighbor table. Then, it also modifies the pheromone table by generating the link failure notification message. For this, it checks the pheromone table to find destinations having a non-zero pheromone value. Through this, it can get the idea about the nodes which have the path to the next neighbor node currently. For the other nodes, node i sets the pheromone value to 0. Then, it generates a link failure notification message, which contains the source and the destination node addresses.

The link-failure message is constructed and broadcast to all the neighbors of node i. So, all the neighbors of node i will receive this message and modify their pheromone tables for the paths going through i to the specified destinations.

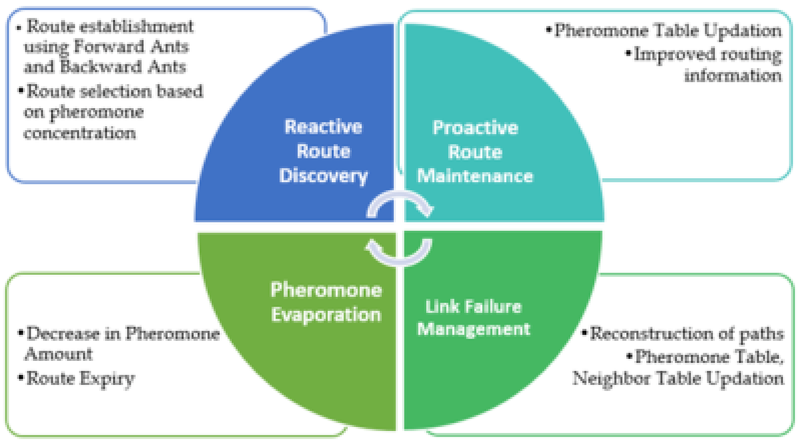
• Route establishment using Forward Ants and Backward Ants
• Route selection based on pheromone concentration
Reactive Route Discovery
Proactive Route Maintenance
• Pheromone Table Updation
• Improved routing information
Pheromone Evaporation
Link Failure Management
• Decrease in Pheromone Amount
• Route Expiry
• Reconstruction of paths
• Pheromone Table, Neighbor Table Updation

Fig. 3.1: Functional Blocks of MyAnt Framework

3.4. Pheromone Evaporation. Pheromone value along each path is decremented with time due to the pheromone evaporation process. When the amount of pheromone for a path becomes less than the pheromone threshold value, that path is declared as expired and a new path is required to be set up. If there is no evaporation in the algorithm then the routes selected by the first ant would tend to be overly enticing to the subsequent ants.

The framework of the proposed routing algorithm, MyAnt along with its functional blocks is shown in Figure 3.1.

4. Implementation Results. The performance of MyAnt is analyzed by varying different parameters under different simulation scenarios. The results are also compared with three routing algorithms: AODV [21], DSR [14], and AntHocNet [11]. The Glomosim [19], [4], [24], [3] simulator is used for experimentation.

In the simulation setting, 50 nodes are taken which are placed in the rectangular area of 2000mX1200m in GlomoSim under different node placement scenarios. The Random Way Point model is chosen for node mobility. In this model, initially, when the simulation starts, a node stays at a specific location for a fixed time duration. This time is defined as the pause time. Then, the node travels towards the destination at the maximum speed specified. Upon arrival at the destination, the node keeps its movement on hold for the specified period and then starts proceeding again. This procedure is repeated during the specified simulation time. Here, the simulation time is taken to be 30 minutes. FTP/GENERIC protocol is used for data packet transmission. The radio range is set to 180 meters. 802.11 protocol is used at the MAC layer and free space signal propagation model at the physical layer. The important configuration parameters used for the simulation are available in Table 4.1.

Different scenarios can be considered by varying the node placement and node movement. The main metrics affected by changing these parameters are Packet delivery ratio (ratio of total packets that were successfully delivered to their destinations to all packets that were sent from their sources), End-to-End delay (the amount of time it takes a data packet to get from source to destination) and the Throughput (total number of packets transmitted) during one simulation [29] [25].

The performance of MyANT is compared with AODV, DSR, and AntHocNet. The node mobility parameter is changed by varying the Random Way Point Mobility model parameters like maximum node speed or pause time. The mobility of nodes increases as pause time decreases. If the pause time is higher, the movement of nodes decreases. It may be observed from Figure 4.1 that, the proposed approach works optimally and the packet delivery ratio is in line with the AntHocNet. The packet delivery ratio of MyAnt seems to be better than other routing protocols.

Different values of node pause times are considered to compute the packet delivery ratio as shown in Figure 4.2. Here also, MyAnt and AnthocNet show similar performances. The delivery ratio is higher for MyAnt compared to AODV and DSR when the pause time is longer.

Table 4.1: Simulation Configuration Parameters for MyAnt

Parameter	Value
PROMISCOUS-MODE	No
MAC-PROTOCOL	802.11
NETWORK-PROTOCOL	IP
ROUTING-PROTOCOL	MYANT
APP-CONFIG-FILE	app.conf
SIMULATION-TIME	30 Min
TERRAIN-DIMENSIONS	2000x1200
RADIO-RANGE	180 Meters
NUMBER-OF-NODES	50
NODE-PLACEMENT	Random
MOBILITY	RANDOM-WAY-POINT
TRAFFIC GENERATOR	FTP/GENERIC

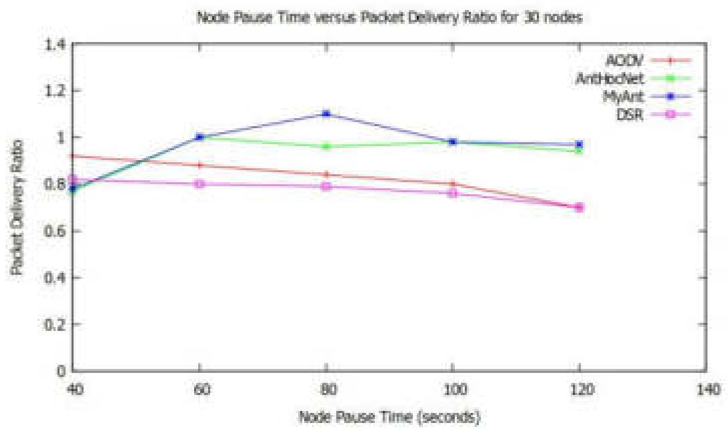

Fig. 4.1: Packet Delivery Ratio measured against node pause time (for 30 nodes)

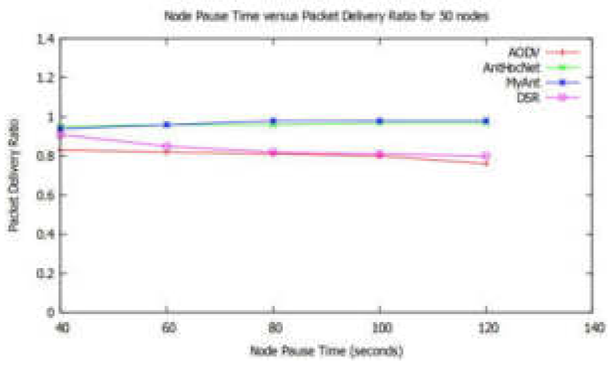

Fig. 4.2: Packet Delivery Ratio measured against node pause time (for 50 nodes)

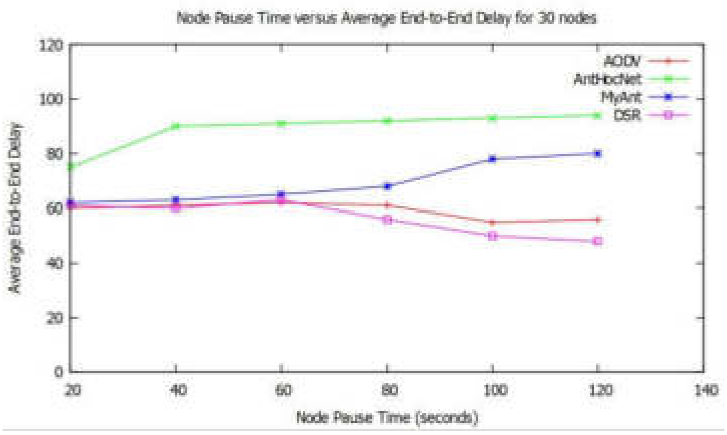

Fig. 4.3: Average End-to-End Delay measured against node pause time (for 30 nodes)

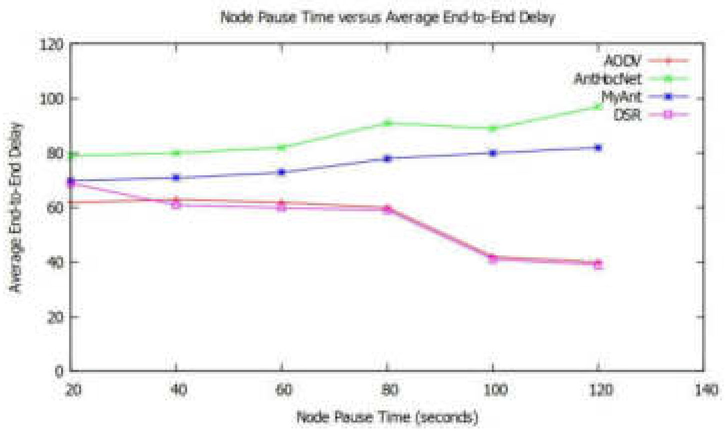

Fig. 4.4: Average End-to-End Delay measured against node pause time (for 50 nodes)

End-to-end delay is more in MyAnt than in AODV and DSR as shown in Figure 4.3 and Figure 4.4, while it is less than the delay in Anthocnet. The reason is, in AODV and DSR, intermediate nodes reply to route requests, so the time required for finding a path is less. While in MyAnt, the source node must wait until the destination node sends a backward ant.

Figure 4.5 and Figure 4.6 show the throughput of MyAnt compared to the other protocols. It is observed that the throughput of MyAnt is better than AntHocNet in both cases.

Figure 4.7 represents the statistics for the total number of packets transmitted and the total number of packets dropped due to the link failure for 49 nodes by varying node placement strategies. In grid placement, the whole physical terrain is divided into grids based on the number of nodes. Each node is placed within a grid unit. The number of nodes must be a square of an integer for grid placement. In the random placement scenario, nodes are scattered throughout the actual physical terrain at random. The physical terrain is divided into a number of cells in the uniform placement of the nodes depending upon the number of nodes in the simulation. A node is placed at random inside a cell.

Figure 4.8 presents the percentage of packets dropped in each scenario. The grid placement has a lower percentage value for dropped packets. While in the uniform node placement, more packets are dropped compared to the other strategies. The probable reason for these results is that as the node placement changes, the

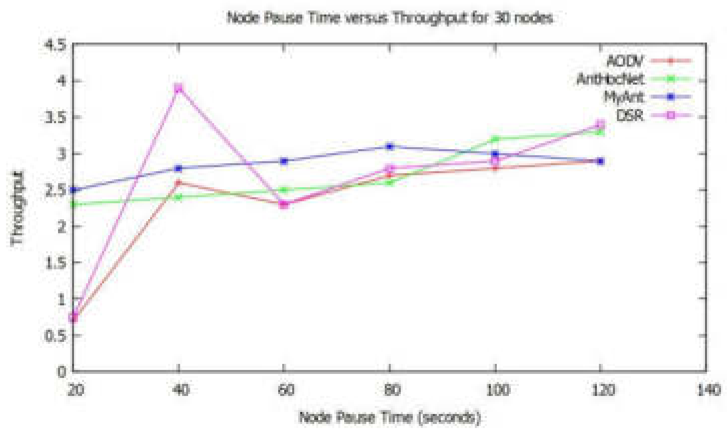

Fig. 4.5: Throughput measured against node pause time (for 30 nodes)

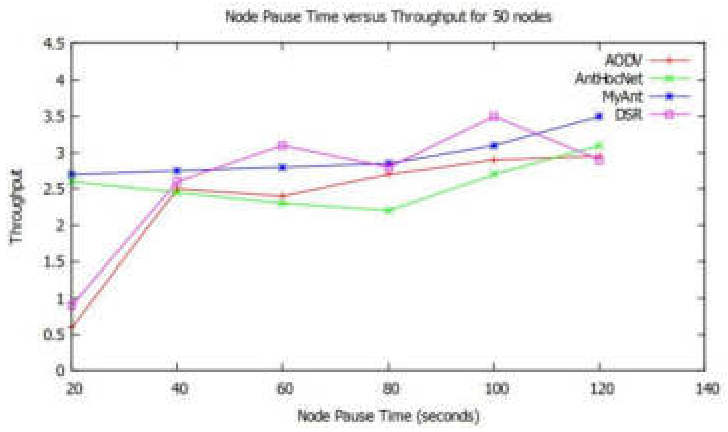

Fig. 4.6: Throughput measured against node pause time (for 50 nodes)

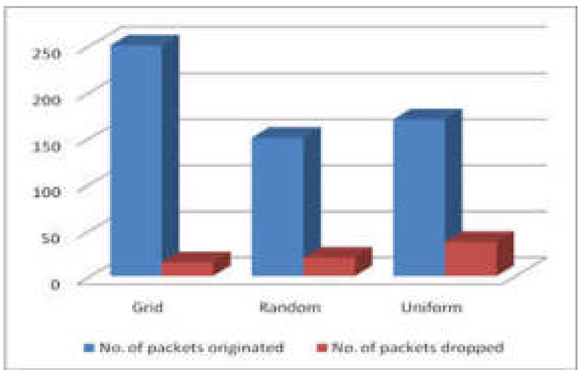

Fig. 4.7: No. of Packets Originated and No. of Packets dropped under different node placement

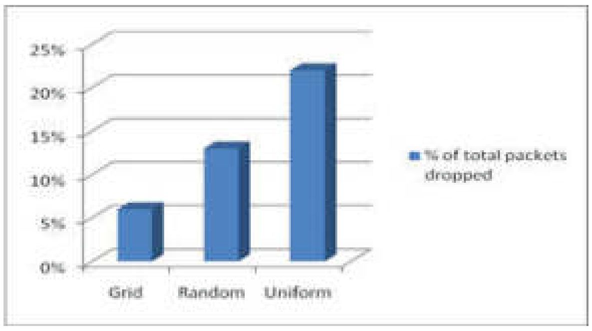

Fig. 4.8: Percentage of the total number of packets dropped under different node placement

distance between nodes keeps varying. Hence, it may happen that the nodes are out of transmission coverage and hence, the connection cannot be established between the nodes.

5. Conclusion and Future Scope. In this paper, a new nature-inspired improved routing protocol for Ad Hoc networks is proposed. This algorithm is motivated by the idea of Ant Colony Optimization and is suitable for the dynamically changing topology requirements. The high mobility of nodes is considered in this algorithm, so it is highly adaptive for ad hoc networks. It also contains the ability to deal with link failures in the network. The performance of the proposed protocol, MyAnt, is evaluated against popular routing protocols, such as AODV, DSR, and Anthocnet, under the GlomoSim simulator environment. Based on the results obtained, we can state that our proposed routing algorithm outperforms conventional reactive or proactive routing approaches for ad hoc networks. Our proposed routing algorithm can be easily integrated with any existing ad hoc network and can also be extended for multipath routing. There is a scope of improvement in the behavior of proactive ants to reduce the routing overhead.

REFERENCES

[1] S. Arjunan and P. Sujatha, *Lifetime maximization of wireless sensor network using fuzzy based unequal clustering and aco based routing hybrid protocol*, Applied Intelligence, 48 (2018), pp. 2229–2246.

[2] R. Asokan, A. Natarajan, and C. Venkatesh, *Ant based dynamic source routing protocol to support multiple quality of service (qos) metrics in mobile ad hoc networks*, International Journal of Computer Science and Security, 2 (2008), pp. 48–56.

[3] R. Bagrodia, R. Meyer, M. Takai, Y.-a. Chen, X. Zeng, J. Martin, and H. Y. Song, *Parsec: A parallel simulation environment for complex systems*, Computer, 31 (1998), pp. 77–85.

[4] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla, *Glomosim: A scalable network simulation environment*, UCLA computer science department technical report, 990027 (1999), p. 213.

[5] Y. Dash, *Nature inspired routing in mobile ad hoc network*, in 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), IEEE, 2021, pp. 1299–1303.

[6] R. Dhaya and R. Kanthavel, *Bus-based vanet using aco multipath routing algorithm*, Journal of trends in Computer Science and Smart technology (TCSST), 3 (2021), pp. 40–48.

[7] G. Di Caro, F. Ducatelle, and L. M. Gambardella, *Anthocnet: an adaptive nature-inspired algorithm for routing in mobile ad hoc networks*, European transactions on telecommunications, 16 (2005), pp. 443–455.

[8] M. Dorigo, M. Birattari, and T. Stutzle, *Ant colony optimization*, IEEE computational intelligence magazine, 1 (2006), pp. 28–39.

[9] M. Dorigo, G. Di Caro, and L. M. Gambardella, *Ant algorithms for discrete optimization*, Artificial life, 5 (1999), pp. 137–172.

[10] M. Dorigo, V. Maniezzo, and A. Colorni, *Ant system: optimization by a colony of cooperating agents*, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 26 (1996), pp. 29–41.

[11] F. Ducatelle, G. A. D. Caro, and L. M. Gambardella, *An analysis of the different components of the anthocnet routing algorithm*, in International Workshop on Ant Colony Optimization and Swarm Intelligence, Springer, 2006, pp. 37–48.

[12] M. Günes, M. Kähmer, and I. Bouazizi, *Ant-routing-algorithm (ara) for mobile multi-hop ad-hoc networks-new features and results*, in Proceedings of the 2nd Mediterranean Workshop on Ad-Hoc Networks (Med-Hoc-Net'03), 2003, pp. 9–20.

[13] O. Hussein and T. Saadawi, *Ant routing algorithm for mobile ad-hoc networks (arama)*, in Conference Proceedings of the 2003 IEEE International Performance, Computing, and Communications Conference, 2003., IEEE, 2003, pp. 281–290.

[14] D. B. Johnson, D. A. Maltz, J. Broch, et al., *Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks*, Ad hoc networking, 5 (2001), pp. 139–172.

[15] P. Khanpara, *A review on fuzzy logic based routing in ad hoc networks*, International journal of advanced research in engineering and technology, 5 (2014), pp. 75–81.

[16] P. Khanpara and S. Bhojak, *Routing protocols and security issues in vehicular ad hoc networks: A review*, Journal of Physics: Conference Series, 2325 (2022), p. 012042.

[17] P. Khanpara and B. Trivedi, *Security in mobile ad hoc networks*, in Proceedings of International Conference on Communication and Networks, Springer, 2017, pp. 501–511.

[18] P. Khanpara, S. Valiveti, and K. Kotecha, *Routing in ad hoc network using ant colony optimization*, in International Conference on Future Generation Communication and Networking, Springer, 2010, pp. 393–404.

[19] J. Martin, *Glomosim*, Global mobile information systems simulation library. UCLA Parallel Computing Laboratory, (2001).

[20] A. Nayyar and R. Singh, *Ant colony optimization (aco) based routing protocols for wireless sensor networks (wsn): A survey*, International Journal of Advanced Computer Science and Applications, 8 (2017).

[21] C. Perkins, E. Belding-Royer, and S. Das, *Ad hoc on-demand distance vector (aodv) routing*, tech. report, 2003.

[22] S. Rajagopalan and C.-C. Shen, *Ansi: A unicast routing protocol for mobile ad hoc networks using swarm intelligence.*, in IC-AI, Citeseer, 2005, pp. 104–110.

[23] R. Ramamoorthy and M. Thangavelu, *An enhanced bio-inspired routing algorithm for vehicular ad hoc networks*, Trends in Sciences, 19 (2022), pp. 4188–4188.

[24] M. Shah and P. Khanpara, *Survey of techniques used for tolerance of flooding attacks in dtn*, in Information and Communication Technology for Intelligent Systems, Springer, 2019, pp. 599–607.

[25] C. V. Subbaiah and K. Govinda, *A bio inspired optimization with reliable qos routing through efficient packet transmission in mobile ad-hoc network*, Renewable Energy Focus, 41 (2022), pp. 188–197.

[26] S. R. Valiveti, A. Manglani, and T. Desai, *Anomaly-based intrusion detection systems for mobile ad hoc networks: A practical comprehension*, International Journal of Systems and Software Security and Protection (IJSSSP), 12 (2021), pp. 11–32.

[27] J. Wang, E. Osagie, P. Thulasiraman, and R. K. Thulasiram, *Hopnet: A hybrid ant colony optimization routing algorithm for mobile ad hoc network*, Ad Hoc Networks, 7 (2009), pp. 690–705.

[28] M. Yashoda and V. Shivashetty, *Bi-crs: Bio-inspired cluster-based routing scheme for d2d communication in iot*, in Proceedings of International Conference on Recent Trends in Computing, Springer, 2022, pp. 187–199.

[29] R. Yesodha and T. Amudha, *A bio-inspired approach: Firefly algorithm for multi-depot vehicle routing problem with time windows*, Computer Communications, 190 (2022), pp. 48–56.

[30] A. Zier, A. Abouaissa, and P. Lorenz, *Firp: Firefly inspired routing protocol for future internet of things*, in ICC 2022-IEEE International Conference on Communications, IEEE, 2022, pp. 2948–2953.

Edited by: Katarzyna Wasielewska
Received: Sep 6, 2022
Accepted: Apr 11, 2023

Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2023 SCPE. Volume 24, Issues 1, pp. 55–67, DOI 10.12694/scpe.v24i1.2086

FRAMEWORK FOR PERFORMANCE ENHANCEMENT OF MPI BASED APPLICATION ON CLOUD

ASHWINI J P[*], SANJAY H A[†], NAYANA M C[‡], K ADITYA SHASTRY [§] AND MOHAN MURTHY M K[¶]

Abstract. Cloud technology is a major revolution that has happened in the computing era that has changed the way applications and resources are used. Elasticity is the key characteristic of the cloud, wherein the required number of resources are provided as a service with a pay-as-you-go principle. This reduces the huge cost involved in buying, installing, and maintaining the resources. Cloud computing, with its highly scalable resources, can be a good platform for High-Performance Computing (HPC). HPC application performance highly depends on the quantity of the resources, which makes the cloud a suitable candidate. But the HPC community is not very happy with cloud technology, and most of the users still think cloud technology is not suitable for HPC applications. But virtualization technology, which is the foundation of the cloud, degrades the performance of applications in the urge to improve utilization. The hypervisor layer and resource sharing by the virtual machines (VMs) hosted on the same node are the main reasons for performance degradation in the cloud. The majority of the HPC applications belong to the message passing (MPI) category, and for these applications, communication cost is the major stakeholder in deciding performance. If these applications are hosted on the cloud, it leads to further performance degradation as the process on a VM communicates through the virtual network interface, which in turn shares the network interface of the host machine with other VMs. MPI-based applications hosted on MPPs work in a bandwidth shared environment as multiple processes communicate over the same network. But in the cloud, as the number of VMs increases, per node bandwidth availability per communication reduces. To address the above issues, we have built a framework to enhance the performance of MPI-based HPC applications on VMs by considering proper VM placement strategy and resource reservation policies, with knowledge of resource availability and process communication patterns. A VM placement strategy for dynamic clustering of VMs with high priority for shared memory-based communication is proposed and tested. Results show that with a medium number of processes, there is an improvement of around 70% with our placement strategy for high data communicating processes. If there are fewer processes, and the single physical node can hold all the VMs, then the performance improvement is up to 500%.

Key words: Cloud Computing, HPC, Heterogeneous resources, k-means Template Clustering, Ranking of Resources

AMS subject classifications. 68M14

1. Introduction.

Cloud computing has given rise to a new business model wherein hardware and software resources are procured as a service from cloud vendors and paid as per usage. This model reduces the burden of a large initial investment and resource management and maintenance costs. The main feature of the cloud is elasticity, wherein the resources can be scaled according to requirements. This makes the cloud an ideal solution for many kinds of applications. Virtualization is the core technology involved in cloud computing, which enhances resource utilization. But the downside of virtualization is that it sacrifices the application's performance. The virtualization layer and resource sharing are important culprits in reducing the performance of an application.

High-Performance Computing applications are used by a wide range of users like researchers, industry, etc. These applications are executed on a cluster/grid to gain maximum performance through parallelization. The main criteria for HPC applications is the quantity of resources available. The cloud can be a suitable platform for HPC applications considering the number of resources available at lower cost and ease of access. But as the performance of a VM is less compared to an actual physical machine, we avoid the cloud for HPC applications. Some of the solutions already existing in the market provide cluster-as-a-service wherein users will pay for

[*]Department of AIML, JNN College of Engineering, Shivamogga,India, 577 204 (ashwinijp@jnnce.ac.in).
[†]Department of ISE, MS Ramaiah Institute of Technology,Bengaluru, India, 560054 (sanjay.ha@msrit.edu)
[‡]Department of PG, Nitte Meenakshi Institute of Technology,Bengaluru, India, 560064 (nayanamc91@gmail.com)
[§]Department of ISE, Nitte Meenakshi Institute of Technology,Bengaluru, India, 560064 (adityashastry.k@nmit.ac.in)
[¶]IBM, USA, (maakem@gmail.com)

static cluster instances. They believe that HPC applications perform better only with dedicated resources [1–2]. These solutions are costly from the perspective of cloud vendors, as the study [3] shows that the utilisation of resources is very low in these static cluster instances.

A good solution for these problems could be dynamically creating a cluster of VMs based on user requirements and promising adequate resources based on application requirements. Most HPC applications belong to the message passing (MPI) category, where multiple processes communicate with each other to solve a problem. Communication costs are the main bottleneck in deciding the performance of these applications. According to [4], the communication performance of VMs is very low because every communication goes through a virtual network interface to the network interface of the host machine, which is shared by all VMs on that machine. Along with this, the hypervisor layer adds some extra delay for VMs to access hardware. As the number of VMs hosted on a node increase, resource availability decreases. It has been observed that [4] VMs communicating through shared memory achieve high performance compared to classic TCP/IP based communication. In this work, we propose a framework for enhancing the performance of VMs hosting MPI-based HPC applications by combining communication aware scheduling with the reservation of adequate network bandwidth. The proposed framework provides a VM placement strategy to set up a dynamic cluster of VMs hosting individual processes of HPC applications with shared memory-based communication or bandwidth reservation. Results show that with intelligent placement and resource reservation, performance is improved. If the number of processes is less, then with pure shared memory-based communication, performance is enhanced by up to 500% when compared to a normal scheduling policy. With a medium number of processes wherein some of the processes are communicating with traditional TCP/IP based communication, performance is improved by up to 70%.

The rest of the paper is organized as follows. In section 2, we discuss important works done in this area. Section 3 explains the proposed framework. Section 4 gives details about the implementation and results, followed by the conclusion.

2. Related work. In [1], the authors Jisha S et al. focus on different frameworks for cloud computing. Cloud computing is a developing area that allows users to deploy applications with better scalability, availability, and fault tolerance. Cloud computing focuses on delivering virtual network services so that users can access services anywhere in the world with the necessary quality of service requirements. Cloud computing is a technique where resources are accessed, and services are needed to perform functions on-demand.

The work done by Peter Sempolinski et al. in the paper [2] is focused on the comparison between different cloud environments like OpenNebula, Platform ISF, VMware Vsphere, Eucalyptus, and Nimbus. Begin with a short summary comparing the current raw feature sets of these projects. After that, deepen the analysis by describing how these cloud management frameworks relate to the many other software components required to create a functioning cloud computing system. They also analyze the overall structure of each of these projects and address how the differing features and implementations reflect the different goals of each of these projects. Lastly, they discuss some of the common challenges that emerge in setting up any of these frameworks and suggest avenues of further research and development. These include the problems of fair scheduling in the absence of money, eviction or preemption, the difficulties of network configuration, and the frequent lack of clean abstraction. The work done by Nicholas Robison et al. in [3] is focused on describing experiences with using virtualization for virtual high performance computing clusters for education and compares the performance of the popular OpenNebula virtualization manager using both NFS and SSH for virtual machine image sharing. Their results show it is possible to develop an effective teaching environment using commodity desktop computers and network hardware along with open-source virtualization software. The work done by Nan Li et al. [4] is focused on a comprehensive survey, with sufficient analysis, of current inter-domain communication mechanisms on Xen-based hosting platforms. Techniques proposed by recent researchers to enhance communication performance are compared and discussed from three perspectives, which are locations, access, and management of shared memory regions. In addition, detailed overviews of various topics regarding virtualization.

In [5], Dhabaleswar K. Panda et al. focus on the following three steps to demonstrate the ability to achieve near-native performance in a VM-based environment for HPC. First, they have proposed Inter-VM Communication (IVC), a VM-aware communication library. It will support efficient shared memory communication among computing processes that are on the same physical host, even though they may be in different VMs.

This is critical for multi-core systems, especially when individual computing processes are hosted on different VMs to achieve fine-grained control. Second, they have designed MVAPICH2-ivc, a VM-aware MPI library based on MVAPICH2 (a popular MPI library), which allows HPC MPI applications to transparently benefit from IVC. Finally, they evaluate MVAPICH2-ivc on clusters featuring multi-core systems and high-performance InfiniBand interconnects.The work done in [6] is focused on integrating HPC interconnect semantics into the VMM split driver model. They aim to decouple data transfers from the virtualization layers and explore direct application-to-NIC data paths. Nonetheless, the implications of this mechanism on the overall throughput constitute a possible caveat of their approach: the way the control path interferes with data communication may result in significant overhead. To justify developing a framework to support standard HPC interconnect features (user-level networking, zero-copy, and so on) in VM environments, we must first investigate the behaviour of HPC applications in such environments. Hence, they deploy network benchmarks and a real scientific application in a cluster of Para virtualized Xen VMs and present some preliminary results.

With the work done by Richard L. Graham and Galen Shipman in [7], with local core counts on the rise, taking advantage of shared memory to optimize collective operations can improve performance. The authors studied several on-host shared memory optimised algorithms for MPI-Bcast, MPI-Reduce, and MPI-Allreduce, using tree-based, and reduce-scatter algorithms. In [8], the authors propose an improved K-means data clustering algorithm. We use the method based on this work to create a cluster of computing resources. This paper focuses on building a strategy to cluster the resources (VMs) for efficiently running MPI applications by building a degree of similarity function based on user requirements.

Most of the solutions provided by cloud vendors for HPC applications include provisioning of static clusters. They end up paying a high price in terms of resource wastage. Our work aims at the deployment of dynamic clusters for HPC applications without compromising performance. The main objectives of this work are:

- Grouping the VM templates based on CPU and memory values to provide a homogeneous environment.
- Ranking the host machines according to the amount of free CPU and memory available.
- Placement strategy for VMs to enhance the communication performance.
- Network resource reservation for VMs on different hosts to communicate.

3. Proposed Work. The proposed framework is shown in Fig 3.1. It aims to enhance the performance of HPC applications on cloud platforms by communication aware VM placement and reservation of resources.

Virtual machines are usually hosted in accordance with the topology of the data center or scheduling policies like round robin, matchmaking, etc. A user requests a virtual machine with his requirements for CPU cores and memory. If a virtual machine template with the same characteristics as the user requirement is available, then it will be chosen, or a new template will be selected. Amazon divides the VMs into different categories, like small, medium, large, and extra-large. This solution is easy to implement, but there is a high possibility of mismatch between the selected template and user requirements, which may lead to either resource wastage or application performance degradation.

Therefore, in the proposed framework, we provide the template exactly as per the user's requirement. The first step is to search for the template. Searching for the required one in the complete template database is quite time consuming. For this purpose, the first module of the framework is to group the VM templates according to CPU and memory values. Grouping will localise the search to a group rather than a complete database. Every time a new template is added, grouping modules need to be executed. After selecting the VM template, the next step is to host it. As the work considers only MPI-based HPC applications, we have considered communication resources as being of the highest priority, along with processing and memory resources. In this regard, the placement strategy gives the highest priority to shared memory-based communication when hosting the virtual machines that are part of the cluster. For this purpose, we need to rank the machines that host the virtual machines depending on various resource availability factors. The CPU and memory requirements are provided by the user, and consistent provisioning of the requested quantity needs to be maintained according to the SLA. A strategy to rank the machines according to the availability of multiple resources is used, and VMs are hosted in descending order of their rank.

As the rank of the machine changes with applications hosted on it, this module must be executed every time a new request comes from a user. It is impractical to think that we will be able to host all virtual machines on a single host so that they can take advantage of shared memory communication. The bottleneck in the overall

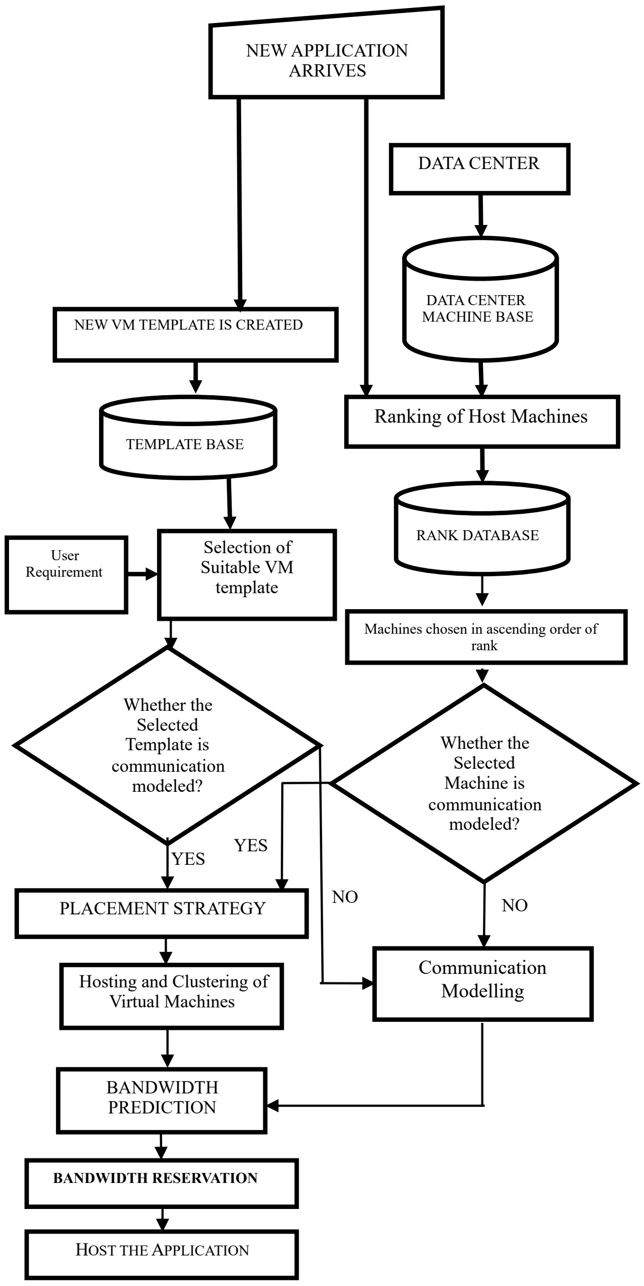
NEW APPLICATION ARRIVES
DATA CENTER
DATA CENTER MACHINE BASE
NEW VM TEMPLATE IS CREATED
Ranking of Host Machines
TEMPLATE BASE
RANK DATABASE
User Requirement
Selection of Suitable VM template
Machines chosen in ascending order of rank
Whether the Selected Template is communication modeled?
Whether the Selected Machine is communication modeled?
YES
YES
NO
NO
PLACEMENT STRATEGY
Hosting and Clustering of Virtual Machines
Communication Modelling
BANDWIDTH PREDICTION
BANDWIDTH RESERVATION
HOST THE APPLICATION

Fig. 3.1: Proposed Framework

performance of the application will be the VMs that are part of the cluster but hosted on different nodes. In [3], we have proposed a bandwidth modelling and prediction framework for MPI based HPC applications on the cloud. This framework is specific to the size of data exchanged amongst processes and the hosting of VMs and node machines.

So, every time a new VM template is created, it must be modelled for bandwidth prediction. Also, whenever a new host machine is added to the data center which is a very rare event to happen, it must be modelled. It has been observed that the reservation of the network bandwidth for the applications drastically improves the performance. Predicted bandwidth will be reserved between clustered VMs that are hosted on different machines. Reservations make sure that communication between VMs will not fluctuate in accordance with the change in available bandwidth.

3.1. Grouping the VM templates based on CPU and Memory values. Starting a VM from scratch takes a long time. For this reason, cloud vendors provide preconfigured virtual machine images called virtual machine templates. In a normal cloud, users choose VM templates and the required number of VMs for that template will be instantiated.

In [9], we worked on clustering the virtual machines on the cloud. The K-Means algorithm is used to cluster the virtual machines dynamically based on a similarity metric (execution time of a benchmark application). According to the results, the best clusters are made up of virtual machines with similar capabilities and are also more powerful than those with different characteristics. Once the best clusters are obtained, the most efficient machines based on the user's requirements in the best clusters can be chosen considering network parameters. However, when we attempted to implement the work in a real-world cloud environment, we realized that it was impractical because VMs would not be available in a running state. They must choose the VM templates that will be instantiated.

Cloud vendors like Amazon divide the templates into three to four different categories, and users must select one from amongst these. But provisioning of the VM as per user requirements will improve the utilization and reduce the cost. The template base is a large collection of all the VM templates available in the cloud and searching for the user-requested template in a huge database takes time. To save time in searching for the requested template, we group the VM templates according to the CPU and memory values as a similarity metric. Grouping will localize the search to a particular group rather than the complete database. Each application varies in its priority towards usage of processing elements and memory. Some require a lot of CPU power, while others require a lot of memory. Depending on the priority towards CPU and memory, each template is marked with a weightage using the equation 3.1.

$$\begin{aligned} TemplateWeightage = CPU_{PRIRT} \times \\ \frac{CPU - value - given - for - VM - Template}{Total - Number - Of - Physical - CPUs - In - Single - Workstation} \\ + MEM_{PRIRT} \times \frac{Memory - Value - Given - For - VM - Template}{Total - Memory - Of - Single - Workstation} \end{aligned} \tag{3.1}$$

K-means algorithms are applied to group the templates according to their weightage. When a new requirement arises, we locate the group with similar CPU and memory values as the required one. And our search will remain local to that group. The K-means algorithm is a prototype based partitioning technique that attempts to find a user-specified number of clusters (K), which are represented by their centroids. The K-Means algorithm takes the input parameter K provided by the administrator and partitions a set of 'N' virtual machine templates into 'K' clusters based on "template weightage" so that the resulting intra-cluster similarity is high, but the inter-cluster similarity is low.

Algorithm 1 VM Grouping based on CPU using KMeans

```
1: Calculate template weightage for all the N templates
2: Select K virtual machine template weightages as the initial centroids
3: repeat
4:     Form K clusters by assigning virtual machine template weightages to the closest centroid
5:     Re-compute the centroid of each cluster
6: until the centroid does not change
```

3.2. Ranking the host machines based on free CPU and Memory available. Once a VM is chosen, our next step is to find the best host to instantiate the VMs. As already mentioned, MPI based applications spend more time on communication, and the goal is to place VMs such that their communication time will be less. According to studies and experiments, VMs instantiated on the same host with shared memory-based communication will have better network performance than VMs communicating via the traditional TCP/IP

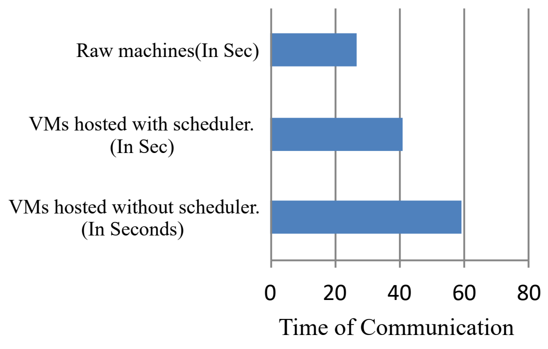

Fig. 3.2: Performance comparison of application hosted on a raw machine, VM with scheduled placement and without scheduling

protocol suite.To use this advantage, our placement strategy places the maximum possible number of VMs on the same host machine with more available resources. In order to find the host machine with the maximum resources, every host machine is ranked based on the available resources. The work considers CPU and memory resources, and hence the host machine that has the most CPU and memory is considered the highest-ranking host machine. Equation 3.2 is used to determine ranking.

$$\begin{aligned} Rank = (x \times percentage - of - CPU - usage - of - single \\ -core - of - host - with - minimum - usage) + \\ (y \times percentage - of - available - memory) \end{aligned} \tag{3.2}$$

where x and y values will range from 0 to 1 depending on the nature of the application the user is requesting. If the application is CPU intensive, then x value will be higher. If it is memory intensive, then y value will be high. A very simple method deployed to set x and y values was to find the time spent by each process in memory and on the CPU by 'ps' command.

3.3. Placement Strategy. Hosting the VM template on a machine is called VM placement. An appropriate scheduling strategy plays a very important role in enhancing the performance of VM. The graph in Fig. 2 shows the performance of the National Parallel Benchmark (NPB), which is an MPI based HPC benchmark on raw machines, VMs hosted without scheduling policy, and VMs hosted with scheduling policy. The experiment was conducted on a private OpenNebula based cloud which uses matchmaking as a scheduling policy. In this policy, the VM will be hosted on a machine with the highest set of resources. We can observe that VMs placed with scheduling show better performance than VMs placed without scheduling. But still, there is a huge gap between the performance of applications on raw machines and VMs.

A placement strategy which considers the characteristics of MPI based HPC applications is proposed in this paper. Communication between individual processes plays an important role in deciding the performance of MPI applications. Communication on VMs is prone to delays because of the virtualization layer. Fig 3.2shows the extra burden an application must bear while communicating through VMs. All the VMs hosted on a machine share the network interface of the host machine, through which they will be connected through a virtual interface. To improve the performance, we conducted experiments using XWAY, which is a shared memory based inter-domain communication mechanism plugin for Xen version 3.0. It provides an accelerated communication path between VMs on the same physical machine by forcing their shared memory based communication.

The graph in Fig 3.3 shows the improvement in shared memory-based communication over normal TCP/IP based communication.

The placement strategy proposed in the work enhances shared memory-based communication amongst VMs which are part of an MPI cluster by placing the maximum possible VMs on a single host. Host machines will

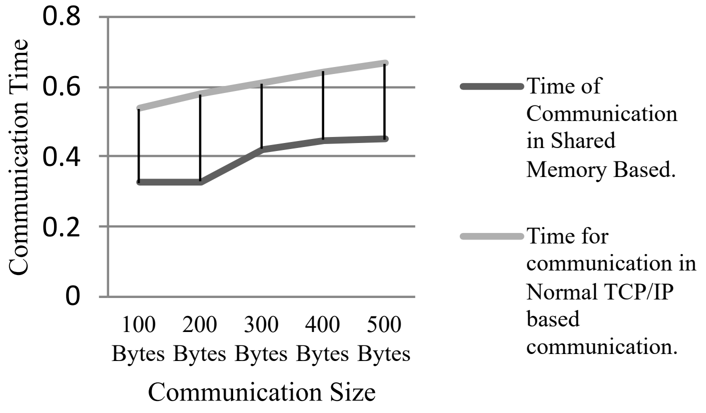

Fig. 3.3: Comparison of shared memory based communication with XWAy to normal communication

be chosen in decreasing order of their rank. If the user is requesting K processes (one VM for one process), then we fit the maximum possible VMs into the highest ranked node machine and continue this procedure in descending order of rank until all ”k” VMs are instantiated.

Algorithm 2 Algorithm for VM Placement

```
1: Let M ∈ Cluster of VM templates based on similarity metric
2: R ∈ rank of each host based on resource availability
3: P = number of VMs requested by user
4: repeat
5:     N= Next Highest Rank Node from R
6:     Fit maximum possible VMs from M in N
7: until P
```

3.4. Network Bandwidth Reservation. Even after applying the placement strategy, a few communicating VMs may be on different hosts due to a lack of resources on the same host. They use TCP/IP stack-based communication, thereby becoming a bottleneck in the overall performance of the application. A study shows that resource reservation will enhance the performance drastically. But reservation without prior knowledge of application requirements is costly and leads to wastage of resources. The next objective aims at improving the communication performance of VMs hosted on different host machines by adequate reservation of network bandwidth. Open Switch, which is an OpenFlow protocol-based tool, is used for the reservation of bandwidth. This work includes the following two parts:

- Modeling and prediction of network bandwidth resource of an application.
- Reservation of predicted bandwidth using proper tools

In this work, for the given MPI application, we predict the bandwidth requirement depending on the data size communicated between individual processes. Prediction of a variable depends on various known variables. Here, depending on the size of the data transmission and the ideal time of data transmission, we predict the bandwidth. Block diagram Fig 3.1 depicts the proposed model. User requirements in terms of the number and type of VMs and application details are collected. A given application is profiled on a single machine using the MPI profiling tool. Profile data will give us the amount of data communicated by each process. Communication costs are indirectly proportional to the bandwidth available. As the bandwidth increases, communication time reduces. But communication time does not entirely depend on bandwidth. A few other things, like processing delay, buffer availability, also influence the communication cost. For a given data size, if we increase the bandwidth linearly, after a certain limit, communication time remains constant without the

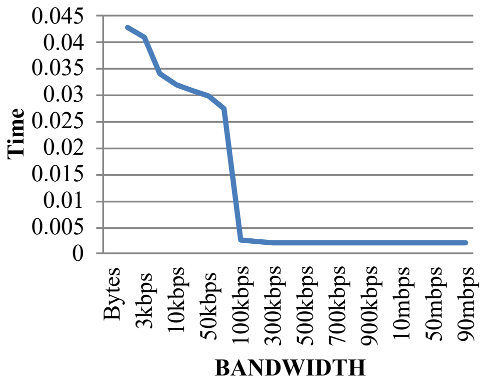

Fig. 3.4: Variation in Time of Data transmission for data size of 10kB for various bandwidths

effect of the increase in bandwidth. Consider the graph shown in Fig.4. For the data size of 10kB, we can see that up to 100kbps of communication time keeps on reducing. However, starting at 100Kbps, time remains constant for all bandwidths. We refer to this bandwidth as "Ideal Bandwidth". For a given data size, any bandwidth below ideal bandwidth decreases the performance, and above ideal bandwidth will result in wastage of resources.

To find Ideal Bandwidth, for the given environment, 'Z' number of bytes will be sent from one VM to another VM with various bandwidths Bi and time taken Ti which will be noted down. Bandwidth B from which time remains constant, is considered as ideal bandwidth. For various sizes of data, Ideal Bandwidth will be noted down and the best fit method will be applied to find the best suited polynomial defining function. Using this function, we predict the bandwidth requirement for the individual processes, which will be reserved for the virtual machines hosting the processes.

4. Experimental Setup and Results. A cloud environment is setup using the Open Nebula cloud platform and the Xen hypervisor is used for VM creation. The Open Nebula is an open-source enterprise-ready cloud management platform that can work with a variety of hypervisors, including those that support a variety of guest operating systems such as Windows and Linux, as well as network and storage support and management tools in a single, tested installable image.

The benchmark application considered for evaluation of our placement strategy is 'NAS Parallel Benchmark' (NPB). It is a small set of programs designed to help evaluate the performance of parallel supercomputers. The benchmarks are derived from computational fluid dynamics (CFD) applications and consist of five kernels and three pseudo-applications in the original "pencil-and-paper" specification. The benchmark suite has been extended to include new benchmarks for unstructured adaptive mesh, parallel I/O, multi-zone applications, and computational grids. Problem sizes in NPB are predefined and indicated as different classes. Reference implementations of NPB are available in commonly used programming models like MPI and OpenMPI.

The original eight benchmarks specified in NPB mimic the computation and data movement in CFD applications. The five kernels are:

- IS $\rightarrow$ Integer Sort, random memory access
- EP $\rightarrow$ Embarrassingly Parallel
- CG $\rightarrow$ Conjugate Gradient, irregular memory access and communication
- MG $\rightarrow$ Multi-Grid on a sequence of meshes, long- and short-distance communication, memory intensive
- FT $\rightarrow$ discrete 3D fast Fourier Transform, all-to-all communication

Three pseudo applications are:

- BT $\rightarrow$ Block Tri-diagonal solver
- SP $\rightarrow$ Scalar Penta-diagonal solver
- LU $\rightarrow$ Lower-Upper Gauss-Seidel solver

Table 4.1: Different configurations of Template instances observation

Template	CPU(in %)	Memory (in Mb)	Template weightage
0	25	1024	5.208333
1	10	1024	3.333333
2	20	1536	5.625000
3	10	512	2.291667
4	20	2048	6.666667
5	30	1536	6.875000
6	10	1024	3.333333
7	20	1024	4.583333
8	40	2048	9.166666
9	15	512	2.916667
10	25	2048	7.291667
11	12	512	2.541667
12	15	2560	7.083333
13	10	512	2.291667
14	20	2560	7.708333

Table 4.2: Grouping of templates based on CPU and Memory values

Group1	Group2
Template 0	Template 1
Template 2	Template 3
Template 4	Template 6
Template 5	Template 7
Template 8	Template 9
Template 10	Template 11
Template 12	Template 13
Template 14	

The Benchmark Classes are:

- Class S → small for quick test purposes
- Class W → workstation size (a 90's workstation; now likely too small)
- Classes A, B, C → standard test problems; 4x size increase going from one class to the next
- Classes D, E, F → large test problems; 16x size increase from each of the previous classes

4.1. Grouping the VM Templates. We created 15 different templates, and Table 4.1 shows the different configurations of VM Template instances. Using CPU and memory values, we have calculated the template weightage of each of the templates. Table 4.2 shows the grouping of VM templates based on CPU and memory values. The number of groups is specified by the administrator, and as the number of groups increases, the degree of homogeneity will also increase. When a user requests a VM, the framework will search for a group whose centroid is close to the requested VM criteria.

4.2. Ranking of Host Machines. Three workstations with different workloads are used for hosting the VMs. As a first step, we run the ranking script, and Table 4.3 shows the rank of these workstations.

Values show that workstation named 'Node1' carries high rank with more memory and CPU. Placement policy induced will host the VMs requested by user in 'Node1'. Once the resources are exhausted remaining VMs will be hosted in 'Node2' and finally 'Node3' will be selected if required.

Table 4.3: CPU-MEM-PER values of 3 host machines

Host Machine	CPU-MEM-PER
Node1	44.011
Node2	43.677
Node3	41.879

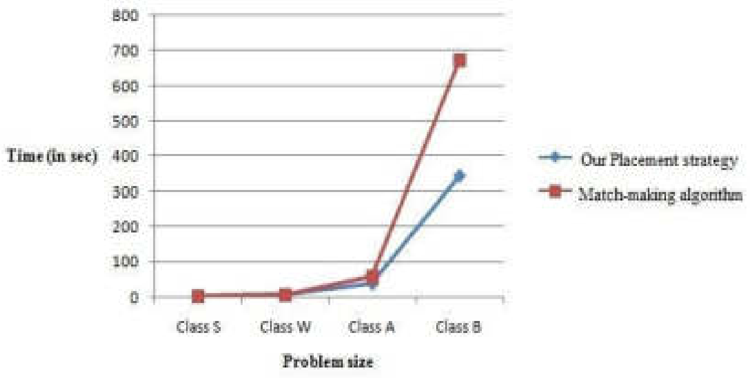

Fig. 4.1: Improved performance in VMs created with our placement strategy and match-making algorithm for four processes

4.3. Network bandwidth prediction and reservation. Individual MPI-based applications have individual processes communicating amongst themselves. This module starts with profiling the application to find the quantity of communication between individual processes. The benchmark application we have considered has all the processes communicating almost an equal quantity of data. If we must handle applications with different data sizes communicated amongst processes, then we can consider the maximum size communicated. Selected VM template and host machines will be modeled, and the bandwidth requirement for a process will be calculated as per the model function chosen by the framework. To test the correctness of this work, a benchmark application was run with various bandwidths amongst VMs carrying processes.

4.4. Comparative study of our Placement Strategy and Open Nebula default match making strategy. OpenNebula uses a default match-making scheduling algorithm. It implements the rank scheduling policy. The goal of this algorithm is to prioritise resources more suitably for the VMs. VMs are placed on the host machine with the most resources in this case.To analyze the performance of our placement strategy, we are using the NAS-NPB parallel benchmark, which is based on MPI. Here we are using the FT benchmark on classes S, W, A, and B. In Class S and Class W communication data sizes are small. But in classes A and B large data sets of around a gigabyte are communicated. The performance of applications over various classes in different placement strategies is compared by varying the number of processes.

4.4.1. Case 1: For 4 processes. Benchmark application execution times for NPB-FT applications using 4 processes are shown in Table IV using our placement strategy and match-making scheduling algorithm. Here, only 4 VMs are formed, and they reside on a single machine, increasing the performance drastically due to shared memory-based communication. We can observe that our placement strategy gives good performance as communication data size increases. On average, we have improved the performance of VMs with our strategy by 552%. In class ‘S’ and class ’W’ as communication size is very small, there is not much difference between the two scheduling strategies. But in classes A and B where maximum communication happens between processes, we can observe that our placement strategy gives very good performance (factor of 7) compared to matchmaking. The graph in Fig 4.1 shows that as communication size increases, our strategy gives good performance.

Table 4.4: Number of Processes: 4

Problem size	Execution time in our Placement strategy (in seconds)	Execution time in Match-making algorithm (in seconds)
Class S	0.47	2.17
Class W	1.17	5.03
Class A	7.91	63.75
Class B	85.08	774.12

Table 4.5: Comparing the results of NPB application obtained from VMs created using Our Placement strategy and Match-making algorithm for 8 processes

Problem size	Execution time in our Placement strategy (in seconds)	Execution time in Match-making algorithm (in seconds)
Class S	2.09	2.65
Class W	5.00	6.64
Class A	38.45	56.18
Class B	342.91	671.77

4.4.2. Case 2: For 8 processes. Table 4.4 shows the execution time of a benchmark application using our strategy and matchmaking strategy for 4 processes.

Table 4.5 shows the execution time of a benchmark application using our strategy and matchmaking strategy for 8 processes. Out of 8 machines, 5 VMs were placed at ”Node 1” and 3 at ”Node 2”. In Class A onwards, data communication will be more, and in such a scenario, our placement strategy gives good performance, and we are observing performance enhancement of 46.11% for Class A and 95.09% for Class B. On an average, we have improved the performance of VMs with our strategy by 70.06% for Class A and B. Fig 4.1 shows the execution time difference between the two methods for 8 process scenarios.

4.4.3. Case 3: For 16 processes. Table 4.6 shows the execution times of two strategies for 16 processes. Once again, we can see good performance from our placement strategy from class A onwards. We can observe a performance enhancement of 36.90% for Class A and 28.62% for Class B. On an average, we have improved the performance of VMs with our strategy by 34.26% for Class A and B. Fig 4.2 shows the execution time difference between the two methods for 16 process scenarios.

In all three cases, we see that for calls S and W, there is no real difference between matchmaking and our strategy. But from call A onwards, we can see the performance enhancement. Class S and W deal with minimum data and other overheads that are greater compared to communication time. But from class A onwards, data communication will be huge, and here our placement strategy works better than others. High Performance Computing application performance depends on communication resources as much as computing resources. Clusters and grids are considered the best platforms for their implementation. The scalability of the cloud will be very suitable for forming better clusters or grids. But at the same time, the heterogeneity and shared resources of the cloud environment pose a challenge.

5. Conclusion and Future work.

Even though resources are the main requirement for HPC based applications, the cloud is never considered as a solution for them. There are many cloud vendors providing HPC on the cloud. Still, MPI-based HPC application users are not satisfied with the performance of the application in the cloud. Most of the HPC applications are MPI based, and for them, communication performance plays a leading role as processes need to communicate with each other a lot. VMs hosted on the same machine use shared memory-based communication that leads to improved speed compared to TCP/IP based communication. This work proposes a virtual machine (VM) placement strategy for HPC applications with better communication performance. Our cloud is setup using OpenNebula Cloud management software and the Xen hypervisor for VM creation. We have proposed a placement strategy wherein the maximum VMs of the cluster are hosted on a single machine with maximum resources. Observations show that as communication size increases, our method is providing good results compared to the default placement strategy. In the future, we would like to consider data availability in our placement strategy.

Table 4.6: Comparing the results of NPB application obtained from VMs created using Our Placement strategy and Match-making algorithm for 16 processes

Problem size	Execution time in our Placement strategy (in seconds)	Execution time in Match-making algorithm (in seconds)
Class S	2.44	2.10
Class W	3.99	3.61
Class A	56.03	78.39
Class B	520.24	669.14

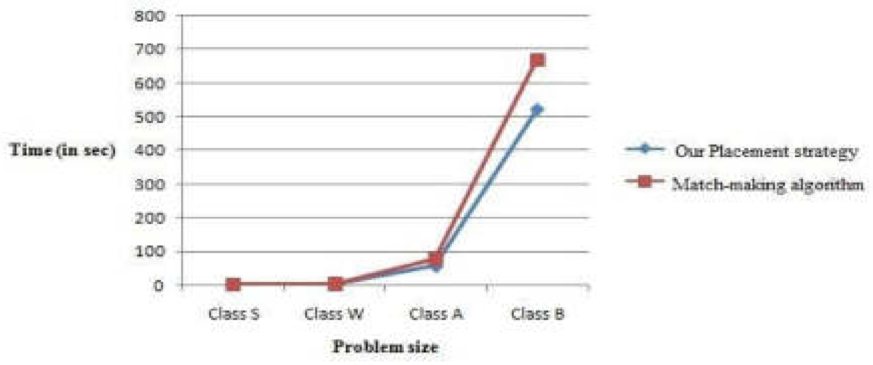

Fig. 4.2: Variation in Time of Data transmission for data size of 10kB for various bandwidths

REFERENCES

[1] Jaliya Ekanayake and Geoffrey Fox, ”High Performance Parallel Computing with Clouds and Cloud Technologies” in First International Conference on CloudComp on Cloud Computin Munich, Germany, October 19 - 21, 2009.

[2] Raihan Masud ”High Performance Computing with Clouds”, http://ix.cs.uoregon.edu/~raihan/HPC_with_Clouds_ Raihan_Masud.pdf, 2011

[3] Danien Warneke and Odej Kao. (2011, February). ”Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud”, Journal IEEE Transactions on Parallel and Distributed Systems (TPDS), Volume:22 , Issue: 6), pp 985 - 997.

[4] Hyunjeong Yoo, Cinyoung Hur, Seoyoung Kim, and Yoonhee Kim. (2009 December). ”An ontology-based resource allocation service on Science Cloud”, International journal of Grid and Distributed Computing, Volume 63 of the series Communications in Computer and Information Science. PP 221-228.

[5] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauery, Ian Pratt, Andrew Warfield, ”Xen and the Art of Virtualization”, ACM SIGOPS Operating Systems Review, Volume 37(5), Pages 164-177.

[6] Nan Li, Yiming Li. ”A study of Inter – Domain communication mechanisms on Xen – based hosting platforms”, Course Project Report in course Advanced perating systems, Computer Science Department, UCSB

[7] Rich Wolski, Neil Spring, and Jim Hayes, (1999 October). ”The Network Weather Service: A Distributed Resource Performance Forecasting service for meta computing”, Journal of Future Generation Computing Systems, 15(5-6):757.768

[8] Shi Na, Liu Xumin, ”Research on k-means Clustering Algorithm: An Improved kmeans Clustering Algorithm”, ”Third International Symposium on Intelligent, Information Technology and Security Informatics”, Jinggangshan University, Jian, China, April 2010

[9] Akioka, Sayaka & Muraoka, Yoichi. (2010). HPC Benchmarks on Amazon EC2. 1029-1034. 10.1109/WAINA.2010.166.

[10] Jisha S.Manjaly, Jisha S, (June, 2009). ”A Comparative Study on Open-Source cloud Computing frameworks”, International Journal of Engineering And Computer Science ISSN:2319-7242 Volume 2, Issue 6. Page No. 2026-2029.

[11] Sempolinski P, Thain D, ”Cloud Computing Technology and Science”, IEEE Second International Conference on Cloud Computing Technology and Science (CloudMo), Nov. Indiana University, IN, USA, December 2010.

[12] Nicholas Robinson, Thomas Hacker, ”Comparison of VM deployment Methods for HPC education”, Proceedings of the 1st Annual Conference on Research in Information Technology (RIIT'12), Calgary, AB, Canada, October 2012.

[13] Wei Huang, Matthew J. Koop, Qi Gao, Dhabaleswar K. Panda, ”Virtual machine aware Communication Libraries for High Performance Computing”, Proceedings of the ACM/IEEE Conference on SuperComputing, Reno, NV, USA, November 2007

[14] Anastassios Nanos, Georgios Goumas, Nectarios Koziris, ”Exploring I/O Virtualization data paths for MPI applications in a Cluster of VMs: A Networking perspective”, Proceedings of the Conference on Parallel processing (Euro-Par 2010).

Ischia, Italy, August 2010

[15] Richard L. Graham, Galen Shipman, “MPI Support for Multi-core Architectures: Optimized Shared Memory Collectives”, Recent advances in Parallel Virtual machine and Message Passing Interface. 15th European PVM/MPI Users’ Group Meeting, Dublin, Ireland, September 2008

[16] NAS Parallel Benchmarks, http://www.nas.nasa.gov/publications/npb.html

[17] Hai Zhong, Kun Tao, Xuejie Zhang, ”An approach to optimized resource scheduling algorithm for open-source cloud systems” in IEEE transactions 2010.

[18] Hai Zhong, Kun Tao, Xuejie Zhang, ”An approach to optimized resource scheduling algorithm for open-source cloud systems” in IEEE transactions 2010

[19] Zach Hill and Marty Humphrey, A quantitative analysis of High Performance Computing with Amazon’s EC2 Infrastructure: The Death of the local cluster ?,Proceedings of the 10th IEEE/ ACM International Conference on Grid Computing (Grid 2009). Oct 13-15, 2009. Banff, Alberta, Canada. Penguin on demand, http://www.penguincomputing.com/POD/Penguin-On-Demand

[20] Pretto, G.R., Dalmazo, B.L., Marques, J.A. et al. Janus: a framework to boost HPC applications in the cloud based on SDN path provisioning. Cluster Comput 25, 947–964 (2022). https://doi.org/10.1007/s10586-021-03470-6.

[21] Saeed Alshahrani, Waleed Al Shehri, Jameel Almalki, Ahmed M. Alghamdi, Abdullah M. Alammari, ”Accelerating Spark-Based Applications with MPI and OpenACC”, Complexity, vol. 2021, Article ID 9943289, 17 pages, 2021. https://doi.org/10.1155/2021/9943289.

[22] D. J. Milroy et al., ”One Step Closer to Converged Computing: Achieving Scalability with Cloud-Native HPC,” 2022 IEEE/ACM 4th International Workshop on Containers and New Orchestration Paradigms for Isolated Environments in HPC (CANOPIE-HPC), Dallas, TX, USA, 2022, pp. 57-70, doi: 10.1109/CANOPIE-HPC56864.2022.00011.

[23] A. Fernandez, ”Evaluation of the Performance of Tightly Coupled Parallel Solvers and MPI Communications in IaaS From the Public Cloud,” in IEEE Transactions on Cloud Computing, vol. 10, no. 4, pp. 2613-2622, 1 Oct.-Dec. 2022, doi: 10.1109/TCC.2021.3052844.

[24] İNCE, MUHAMMED NUMAN; GÜNAY, MELİH; and LEDET, JOSEPH (2022) ”Lightweight distributed computing framework for orchestrating high performance computing and big data,” Turkish Journal of Electrical Engineering and Computer Sciences: Vol. 30: No. 4, Article 26. https://doi.org/10.55730/1300-0632.3866.

Edited by: Katarzyna Wasielewska
Received: Aug 4, 2022
Accepted: Apr 4, 2023

Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2023 SCPE. Volume 24, Issues 1, pp. 69–80, DOI 10.12694/scpe.v24i1.2088

MULTI-CLASS BRAIN TUMOR CLASSIFICATION AND SEGMENTATION USING HYBRID DEEP LEARNING NETWORK (HDLN) MODEL

PARASA RISHI KUMAR, KAVYA BONTHU, BOYAPATI MEGHANA, KONERU SUVARNA VANI[*] AND PRASUN CHAKRABARTI[†]

Abstract. Brain tumor classification is a significant task for evaluating tumors and selecting the type of treatment as per their classes. Brain tumors are diagnosed using multiple imaging techniques. However, MRI is frequently utilized since it provides greater image quality and uses non-ionizing radiation. Deep learning (DL) is a subfield of machine learning and recently displayed impressive performance, particularly in segmentation and classifying problems. Based on convolutional neural network (CNN), a Hybrid Deep Learning Network (HDLN) model is proposed in this research for classifying multiple types of brain tumors including glioma, meningioma, and pituitary tumors. The Mask RCNN is used for brain tumor classification. We used a squeeze-and-excitation residual network (SE-ResNet) for brain tumor segmentation, which is a residual network (ResNet) with a squeeze-and-excitation block. A publicly available research dataset is used for testing the proposed model for experiment analysis and it obtained an overall accuracy of 98.53 %, 98.64 % sensitivity and 98.91 % specificity. In comparison to the most advanced classification models, the proposed model obtained the best accuracy. For multi-class brain tumor diseases, the proposed HDLN model demonstrated its superiority to the existing approaches.

Key words: Classification, segmentation, ionizing radiation, proposed methodology, and existing models.

AMS subject classifications. 68T07

1. Introduction.

Unnatural and uncontrolled cell development in the brain is known as a brain tumor [13]. It also has the possibility of spreading to other bodily organs and having an impact on human functions [31]. Many different categories exist for classifying brain tumors, including primary and secondary [21]. Primary tumors are those that first appear in the brain [26]. On the other hand, tumors that first develop before spreading to the brain are referred to as secondary tumors [11]. Brain tumor detection and classification can be done using a variety of imaging techniques, but MRI imaging technique is the popular method for brain tumor detection [5]. The frequent kind of brain tumor with glial cell origins is gliomas [25]. The majority of meningioma tumors are benign, a benign tumor called meningioma develops on the membrane [23]. However, the pituitary glands that manage hormones and control bodily processes are where a pituitary tumor originates [35].

From the above information, identifying brain tumors early and classifying them become crucial tasks in case evaluation [16] [7]. In some complex circumstances, the classification stage may also be a difficult assignment for doctors or radiologists [24]. Experts must work on these cases to identify the tumor location and evaluate its tissues in comparison to adjacent regions [15]. To find brain cancer early and in short time, a Computer Aided Diagnosis (CAD) system is required without requiring human participation, because this task takes some time [6].

A subset of machine learning (ML) called deep learning (DL) is focused on hierarchical feature learning and learning data representation [4]. A system of many layers of nonlinear processing identities is used by CNN to extract features [1]. As we go deeper into the network, data abstraction is aided by the fact that the input of one sequential layer becomes the output of the preceding one. CNN's are a class of DL that are frequently employed in the analysis of visual data [18]. They are created to require the least amount of preprocessing possible. The biological processes of the human brain served as its inspiration.

Feature learning and unlimited accuracies are the key benefits of CNNs over standard machine learning and conventional neural networks [33] [10]. Feature learning and unlimited accuracies are improved by increasing

[*]Velagapudi Ramakrishna Siddhartha Engineering College, Kanuru, Vijayawada, Andhra Pradesh, India
[†]ITM SLS Baroda University, Vadodara, India (drprasunchkrabarti@gmail.com)

training samples and result in a more accurate and reliable model [34]. For classifying and segmenting different types of brain tumors, we propose a new HDLC model in this research. Different configurations are used to evolve the network architecture to obtain the most suitable structure. The proposed method adopts the Mask RCNN for different types of brain tumor classification and a SE-ResNet for brain tumor segmentation. The CE-MRI dataset showed the best classification accuracy ever with the proposed model, which is publicly available on figshare.

The main contribution of the research is the following:

- First a local smoothing filter and non-local mean filter are used for prepossessing the MRI images for removing unwanted noise in the image. The image quality and contrast of the image is improved by this stage.
- After the completion of a preprocessing phase the deep learning-based Mask RCNN is used for effective classification. The proposed classification network model is designed for multiple brain tumor classification including pituitary, meningioma, and glioma.
- Then proposed a SE-ResNet for the accurate segmentation of brain tumors. It uses multi-stage architecture and convolution blocks in each stage. The results from this proposed network were more accurate in terms of disease diagnosis.
- The proposed method is evaluated on the publicly accessible database (CE-MRI). The experiments are performed on the Python platform. The proposed approach outperforms the state efficiency for all other approaches, according to the experimental data.

The remaining sections of the paper are divided and structured as follows Literature review is discussed in Section 2 the proposed methodology is presented in Section 3 experimental analysis results of the proposed model and comparison results with other recent state of the art methods are provided in Section 4. Section 5 also gives a conclusion and recommendations for further research.

2. Related Prior Research. In this section, the methods and earlier research for MRI-based segmentation and classification of brain tumors are presented. An innovative deep learning-based model is developed by Kulkarni et al. [14] in this research. This method is developed for the classification of brain tumor diseases. The morphological operations are first applied to the input image after thresholding. The effective features are extracted by CNN. The proposed deep learning model is performed by using a transfer learning approach. The AlexNet model is used for brain tumor classification. GoogLeNet model is used for classifying malignant brain tumors.

The effective hybrid-brain-tumor classification (HBTC) is developed by Nawaz et al. [19] for multiple classifications of brain tumors. The proposed model classifies metastatic (meta), meningioma (menin), glioma, and cystic (cyst) brain tumors in this research. The performance of the brain tumor detection process is enhanced by this proposed network model, and also it reduces the inherent complexity of the process of diagnosing brain tumors. Run-length matrix (RLM), co-occurrence matrix (COM), and gradient feature are recovered from the segmented dataset. To categorize meta, menin, glioma, and cyst tumors, hybrid multi-features were applied, and the nine most optimized features were chosen. These features were then fed to the framework's classifiers, random tree (RT), meta bagging (MB), J48, and multilayer perception (MLP).

The novel deep network model is proposed by Kumar et al. [12] in this research. The vanishing gradient and overfitting problems are resolved by using ResNet-50 and global average pooling. A simulation was run utilizing a dataset of 3064 brain MRI images with three tumors to assess the performance of the proposed model.

The deep learning-based Unet architecture with ResNet50 as a backbone is developed by Sadad et al. [6] for the segmentation of the brain tumor using the Figshare data set. The effectiveness of detection is improved by introducing the preprocessing and data augmentation techniques. Reinforcement learning through transfer learning and evolutionary algorithms are used to classify brain tumors. There is also the use of additional deep learning techniques including InceptionV3, MobileNet V2, DenseNet201, and ResNet50.

The Grab cut method is developed by Saba et al. [22] for effective actual brain tumor segmentation. A serial-based method is used for fine-tuning the transfer learning-based VGG-19 model for extracting the efficient features, then those features are effectively combined with hand-crafted (shape and texture) features. Entropy is utilized to optimize these characteristics for accurate and quick classification, and classifiers are given a

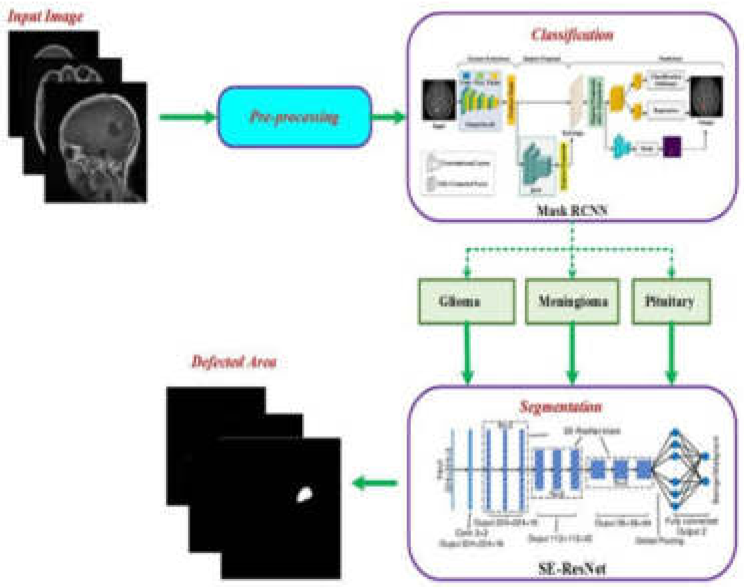

Fig. 3.1: Schematic diagram of the proposed methodology

fused vector as a result. A new hybrid paradigm comprised of a CNN and neural autoregressive distribution estimation (MODE) was proposed by Hashemi et al.[8] in this research. Data acquisition, feature extraction, and classification are the three learning stages this paper indicates for a hybrid architecture. This study's main focus is on automatically predicting the distribution of data and extracting its features in a quick and precise manner.

3. Proposed Methodology. A tumor in the brain is an unexpected lump of flesh in which cells have developed and multiplied uncontrollably. At present, it is a very destructive problem. Because of the complicated nature of the tumor's structure, a timely and accurate diagnosis of brain tumor is essential. The proposed model for identifying brain tumor disease on an MRI consists of the following four fundamental steps: (1) Preprocessing, (2) feature extraction, (3) classification (4) Segmentation, and (5) performance Evaluation.

The first stage is called preprocessing when the images are normalized to produce a uniform contrast. The non-local mean filter and a local smoothing filter are employed for eliminating the noise thus the image can be enhanced. After preprocessing, the effective structural and textual features are extracted by using ResNet-50 for reducing the complexity, and computational cost of the proposed model. The next phase is classification. The multiple brain tumor classification stages such as glioma, meningioma, and pituitary are classified by using a deep learning-based Mask RCNN classifier. Then, the SE-ResNet model is used to segment the defective area. The proposed model's effectiveness is then assessed using test images. The block diagram of the proposed HDLN brain tumor classification is presented in Figure 3.1.

3.1. Problem Statement. Previous deep learning models required numerous steps to analyze the data, and handcrafted features that were retrieved could not guarantee greater classification performance. This illustrates the inadequate nature of automatic learning. Additionally, it performs poorly when there is an unequal distribution of the sample data in a multiclass classification task, which is a problem that is quite common in the medical field. To solve this problem, we propose a new HDLN model for accurate classification and segmentation of multiple brain tumor diseases.

3.2. Image Acquisition. In medical applications, the expansion of 2D images becomes more useful because CE-MRI brain slices often have a small number and a wide slice gap. This study evaluates its methods using 3064 T1-weighted CE-MR images from 233 individuals, which include 930 pituitary, 1426 gliomas, and

Table 3.1: The number of slices and associated patient count for each form of brain tumor (meningioma, glioma, and pituitary) in the dataset.

Category	**Number of Images**	**Patients**
Meningioma	**708**	**82**
Pituitary	**930**	**60**
Glioma	**1426**	**91**
Total	**3064**	**233**

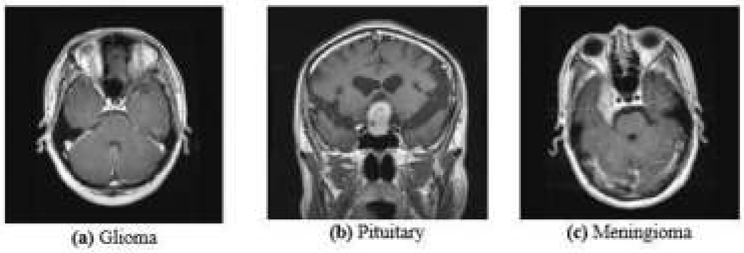

Fig. 3.2: Brain Tumor types

708 meningiomas brain tumors. The radiologist manually defined the tumor boundaries. The 512 × 512 resolution. Mat files used for the dataset images that are accessible on Figshare. A 224 ×224 input layer size was used for designing the proposed framework. Table 3.1 shows the database description, and Figure 3.2 shows the distribution of tumor types.

3.3. Pre-processing stage. The system gathers information about the brain using MRI images as its input. A local smoothing filter and a non-local mean filter are used for preprocessing the MRI images. When photos are compressed or image data is transferred, these filters were employed to reduce any unwanted noise. The image quality and contrast are enhanced during this stage.

3.4. Classification stage. Both pixel-level segmentation and object detection are done by the most recent DL model called Mask-RCNN. It is a faster RCNN extension that can also perform segmentation in addition to localization and classification. Mask-RCNN also separates class prediction from the mask. The segmentation is performed by using the FCN network, which provides a minimal amount of network overhead. Mask- RCNN network is proposed in this research, in this classification network the features are extracted by using DenseNet-41. DenseNet-41 is the feature computation layer of the classification network. DenseNet is the most significant network model of CNN. A collection of dense blocks are present in the DenseNet that are gradually linked to one another using additional pooling and convolutional between succeeding dense blocks. The problem of the top-level key points lacking target location information may be slightly improved by DenseNet by presenting the complicated transformations that accomplish this. DenseNet model reduces the parameters, which lowers the cost. The process of key point propagation is also assisted by DenseNet and it encourages their reuse based on this network is more appropriate for classifying brain tumors. Figure 3.3 illustrates the structure or flow of the proposed classification model. Different networks make up the standard Mask-RCNN network such as a convolutional backbone network, region proposal network, RoI classifier, and bounding box regressor. The following section goes over each stage in great depth.

3.4.1. Feature Extraction. From the input MR images, the effective related features are extracted by the backbone network. Any CNN model that was designed with image analysis might be this network. For a key point extraction, network acquire dependable and more discriminating features. Furthermore, the more computational overhead is required for increasing the depth size of the network and the network weights optimization is more challenging, which could lead to an increasing gradients problem. ResNet model is the backbone network for Mask-RCNN and has been used in previous works in medical image analysis. Although

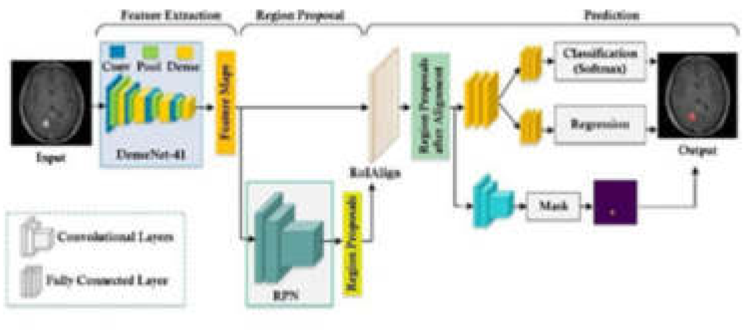

Fig. 3.3: Classification network configuration

the ResNet model has numerous parameters and uses skip connections, this eventually leads to the vanishing gradient problem. A Mask-RCNN with DenseNet framework is implemented in the multi-class classification of brain tumor diseases. The features are more specifically computed by using the DenseNet-41 network. Since the number of dense connections is connected in the DenseNet model. Based on these dense connections, the DenseNet model computes effective set of image features. The initial convolution layer of DenseNet-41 has 24 channels instead of 64, and the kernel is 3×3 instead of 7×7, resulting in fewer parameters from the actual model. To handle the computational complexity of the network, each dense block's layers are adjusted.

3.4.2. Region Proposal Network. The ROIs are generated by feeding the obtained features to the RPN network in this stage, then the RPN network produces the feature map. The final classification of brain tumors is done by localizing the ROIs to the tumor regions. The sliding window manner is used for scanning the entire image by using 3×3 CL in the RPN module for generating the relevant anchors. The image's whole surface is covered by these anchors, which are bounding boxes of various sizes. 20 k anchors of different sizes and scales are present in the RPN network, the whole images are covered by overlapping these anchors with each other. The more tumor-related objects are present in the top anchors, these anchors are chosen using RPN prediction, and bounding box regression is then used to adjust their location and size. Then the classification is performed by passing the several RoIs generation to the next stage.

3.4.3. Bounding Box Regression with RoI Classification. The generated feature map and RoI are sent to this network as input. This network is more sophisticated than the RPN, which only yields two classes, including background and foreground. One of three classes is classified by the proposed RoIs such as pituitary, meningioma, and glioma and the bounding box's size is also increased by this network. From the feature maps, all RoIs are pooled into one fixed size in this stage. For candidate regions with arbitrary sizes, the fixed length of key point vectors is obtained by utilizing the RoIAlign layer for resizing the feature maps. In the RoI pooling layer, misalignment problems are avoided by performing the bilinear interpolation process, which utilizes the quantization operation. Final classification results are obtained by feeding the key points to the classification and regression layer.

3.4.4. Brain Tumor Segmentation using SE-ResNet. The brain tumor is effectively segmented by using the SE-ResNet model in this research. The SE-ResNet ignores the spatial information in favor of examining the dependencies among its convolutional features' channels. Excitation and squeeze operations are present in the SE block that scales the relevance and overall information of each feature map, respectively. The significant information from each channel is extracted by the squeeze operation using global average pooling. A nonlinear function is used by the excitation operation for computing the inter-channel dependencies. Because of its superior ImageNet image classification performance, one of the most popular CNN models is SE-ResNet model. Additionally, SE network is simple to implement because, it only involves adding SE block without altering the structure of the preexisting model. During training, the deep gradient for the seamless transmission is enabled by the residual block, when increasing the network layers in the network. Which is used for effectively training the whole network. The 1D convolutions are used by the proposed network instead of 2D

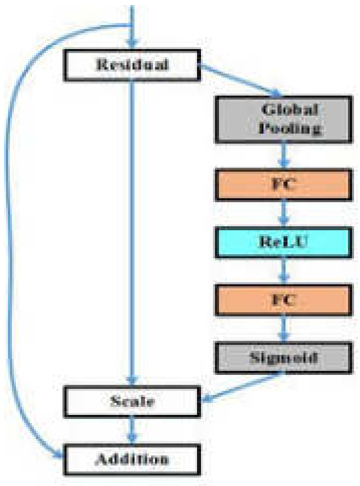

Fig. 3.4: SE-ResNet module structure

convolutions. This is the main difference between the original SE-ResNet and our proposed network model. Figure 3.4 shows the structure of the SE-ResNet model. In this model, the fully connected layer is represented by FC, ReLU and the activation functions are depicted by Sigmoid.

4. Results And Discussion. This research effectively classifies multi-class MRI images. The HDLM is used for the classification and segmentation of the CE-MRI dataset. Multiple classifications of brain tumor diseases are classified by using Mask RCNN. The hybrid SE-ResNet model is developed for the segmentation of multiple brain tumor diseases, where 70% of the dataset was used for training purposes and 30% for testing purposes. The Python platform is used for achieving the classification and segmentation results with computer resources of 8 GB of RAM and an i5 processor. The classification and segmentation performance is increased by adding several approaches to the proposed model.

4.1. Performance Evaluation for Classification. In the deep learning models, the classification performance is analyzed by using a confusion matrix widely. Accuracy, sensitivity, specificity, and precision are used to measure the classification performance of the model. The confusion matrix is used for computing these calculations. All data samples are comprised of True Positive (TP) indices and the model is associated with a specific class for effective classification of the model. Additional samples are available for the True Negative (TN) indices in the confusion matrix. Other successfully determined classes are related to these samples. In the uncertainty matrix, the False Negative (FN) and False Positive (FP) indices represent the classifier's estimated number of incorrect samples.

The confusion matrix is used to retrieve accuracy performance parameters for experimental analysis. By dividing the total value of the confusion matrix by the proportion of true positive and true negative value, Equation 4.1 define accuracy. The confusion matrix factor determines the classifier's performance in the proposed model. Similarly,

$$Sensitivity(Recall) = \frac{TP}{TP + FN} \tag{4.1}$$

$$Specificit = \frac{TN}{N + FP} \tag{4.2}$$

$$Precision = \frac{TP}{TP + FP} \tag{4.3}$$

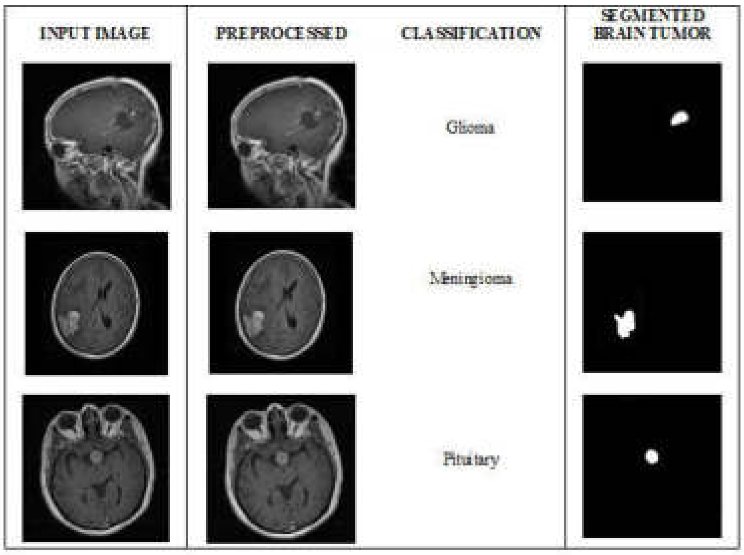

Fig. 4.1: Experimental results of the proposed model

Table 4.1: Performance results for classification of the proposed model

Tumor type	**Accuracy (%)**	**Specificity (%)**	**Sensitivity (%)**	**Precision (%)**
Glioma	**99.34**	**98.65**	**98.79**	**99.02**
Meningioma	**99.23**	**98.87**	**98.45**	**98.98**
Pituitary	**99.32**	**99.23**	**98.69**	**99.04**

$$Accuracy(\%) = \frac{TP + TN}{TP + FN + TN + FP} \tag{4.4}$$

For training, 80% of the data is used and for testing, 20% data is used in the experiment analysis. The ground truth labels and predicted labels are used for evaluation metric computation Similarly, equations 4.2, 4.3, 4.4

4.2. Experimental Results. The multiple classification tasks are performed by using a deep learning-based method. The MRI image classification such as glioma, meningioma, and pituitary are evaluated in terms of performance. All classification tasks are performed by using the HDLN model with the aforementioned network parameters. The experimental results of the proposed model are shown in Figure 4.1.

The extracted accuracy metrics from the confusion matrices are displayed in Table 4.1. The class-wise accuracies, specificities, precisions, and sensitivities of the proposed model is shown in the table. Classification accuracy for meningioma, glioma, and pituitary tumors is 99.23%, 99.34%, and 99.32%, respectively. The performance of the classification results is significantly improved from this observation. The proposed approach provided an overall accuracy, sensitivity, and specificity of 99.23%, 98.64%, and 98.91%. The class-wise performance is determined by employing a small sample of the validation set's images. The multi-class brain tumor classification is effectively performed by the proposed HDLN model.

With 26 epochs and a 0.01 learning rate, the training phase is conducted to achieve higher classification results. For each epoch, the experiment analysis used 31 iterations in total. The training and validation accuracy and loss graph for multiple brain tumor classification is shown in Figure4.2. An accuracy of 99.23% was used to validate the training progress. In the overall training process, 1 min 54 s timing is used for reaching the final iteration. For feature extraction, the two convolutional layers are then applied. Effective feature

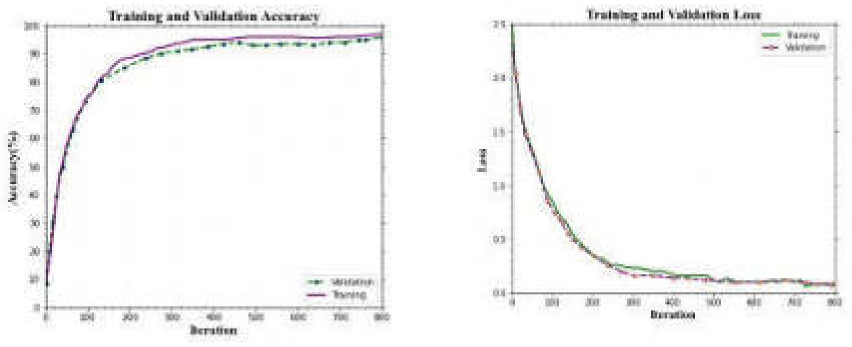

Fig. 4.2: The accuracy and loss of the multi-class classification of brain tumor diseases during training and validation.

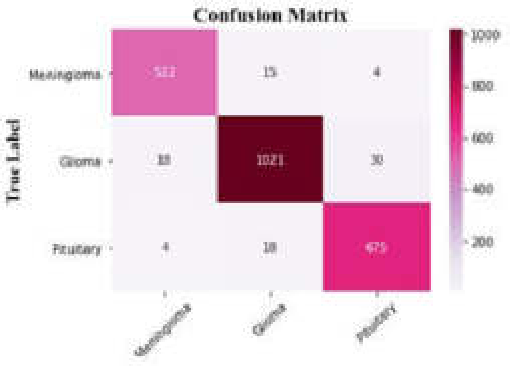

Fig. 4.3: The confusion matrix of the proposed model

extraction improves the validation accuracy during the training phase.

The proposed model's confusion matrix is displayed in Figure 4.3. The predicted values are represented by X-axis and the true labels (ground truth) are represented by Y-axis. The area under the receiver operating characteristic (AUC ROC) curves for different tumor classes in the brain CE-MRI dataset is shown in Figure 4.4.

4.3. Comparative Analysis of Other Classification Models. We compare our proposed model classification results with those from earlier research over the same dataset in this section. The performance result comparison for multi-class classification is shown in Table 3. The proposed Mask RCNN classification network achieves the highest classification results in Table 4.2. 94.82% accuracy is achieved by [30] for the classification of brain tumors using the VGG19 model. In [29], CNN achieves an accuracy of 96.14%. In [2], the authors used transfer learning for feature extraction using a pre-trained GoogleNet model. Three different classifiers are used for the classification of obtained features such as KNN, SVM, and Softmax, 97.1% is achieved by this research. Three alternative processing techniques, including residual blocks, attention modules, and hypercolumn approach were used by the DL model BrainMRNet [32] to classify brain tumors. 97.69% accuracy is achieved by this research for classification. Multilayer perception (MLP) is used to attain effective classification performance [28], and they were able to achieve an accuracy of 98.15%. The higher performance is achieved by the

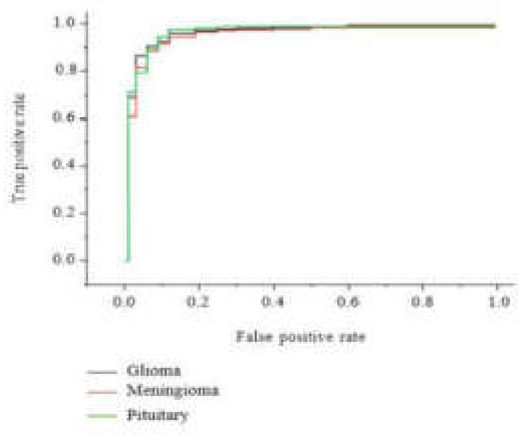

Fig. 4.4: AUC ROC curves of different tumors in the CE-MRI dataset

Table 4.2: Comparison analysis of classification performance

Technique	**Classification model**	**Accuracy (%)**
Swati et al.	**VGG19**	**94.82**
Sultan et al.	**CNN**	**96.14**
Deepak et al.	**GoogLeNet and SVM**	**97.10**
BrainMRNet	**Attention Module Hyper column technique and Residual block**	**97.69**
Proposed classification model	**Mask RCNN (DenseNet-41)**	**99.23**

proposed Mask RCNN classification model than the existing deep learning models and it successfully classifies between multiple classes of brain tumors. The main advantages of the proposed classification network model include the prevention of overfitting and the elimination of any adverse network performance impacts related to the classification process. The features of the tumors are successfully extracted by the proposed model including inter-scale variability, and also the final classification performance is enhanced by the proposed model. While compared to the previous model, the proposed model effectively classifies multiple brain tumor classifications. The proposed method achieves 99.23% accuracy, 99.08% specificity, and 99.12% sensitivity. The proposed model automatically diagnoses and pre-screens the brain tumors due to its high classification accuracy. Figure 4.5 shows the performance evaluation for classification.

4.4. Comparative Analysis of Other Segmentation Models. The proposed model is compared with other segmentation techniques described in this section. The comparative analysis for segmentation models with the average results given in previous research for evaluating the performance.

A quantitative comparison employing various performance measurement parameters, including accuracy, dice score, and mean IoU is displayed in Table 4.3. The RCE technique is proposed by Sheela et al. [28]. The tumor segmentation is done by using fuzzy c-means and an active contour model in this research, an average accuracy of 91% was attained. 0.950 is achieved by [17], and the computation cost is increased due to a large number of parameters in this model. The multi-scale CNN model is developed by [3], and three different spatial scales are used for processing the input MR images using multiple processing routes in this research. They obtained a 0.828 dice score and an average segmentation accuracy of 97.30% respectively. The approach in [18] has a 95.90% accuracy rate and a dice score of 0.955.

While compared to the existing segmentation techniques, the proposed model achieves better performance. The proposed SE-ResNet segmentation model for multi-class brain tumors achieved an overall accuracy of 97.83%, mean IoU of 0.976, and dice score of 0.974. Due to structural complexities, most of the existing model

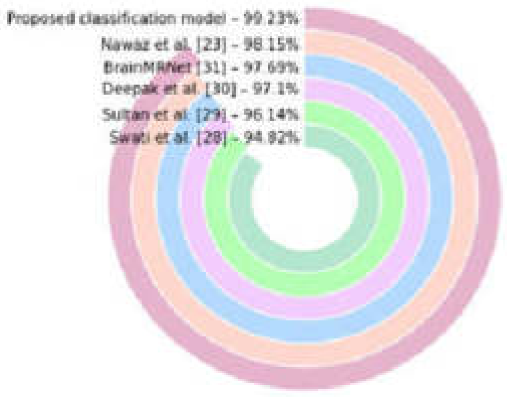

Fig. 4.5: Performance evaluation for classification

Table 4.3: Comparison analysis of classification performance

Technique	**Segmentation Model**	**Dice**	**Mean IoU**	**Accuracy (%)**
Sheela et al.	**Fuzzy-C-Means and Active Contour**	**0.665**	**-**	**91.00**
Masood et al.	**Traditional Mask-R-CNN**	**0.95**	**0.95**	**95.10**
Pernas et al.	**Multiscale CNN**	**0.828**	**-**	**97.30**
Masood et al.	**Mask-RCNN (ResNet-50)**	**0.955**	**0.951**	**95.90**
Proposed Segmentation Model	**SE-ResNet**	**0.974**	**0.976**	**97.83**

like produces only hand-crafted features and these features are not effective for detecting the tumor region[20]. But our proposed model produces effective deep features and these features accurately determine the tumor regions [9]. The entire image is directly processed for the segmentation in some previous approaches, which leads to misclassification as a result of the complex background (e.g., MRI artifacts, brain tissues overlapping with tumor boundaries, etc.), which decreases the segmentation accuracy of the image. A region-based approach is used by this approach in [16] to localize tumors, and tumor segmentation requires further processing. The proposed segmentation model leverages the RoIAlign layer and localized RoIs for segmentation in contrast to previous techniques, which increases the segmentation accuracy and ultimately reduces the segmentation space[27]. Additionally, our proposed model uses fewer computational resources to achieve high performance for multi-class brain tumor segmentation. Performance evaluation for segmentation is shown in Figure 4.6.

5. Conclusions. In this paper, we proposed a novel automatic method for the detection of brain tumor diseases from MRI images. For multi-class brain tumor detection, this research proposes a new HDLN model for automated multi-class classification. The multiple brain tumor disorders including pituitary, meningioma, and glioma are classified by the proposed model using MRI images. A publicly tested dataset (CE-MRI) is employed for testing experimental analysis. First, image pre-processing is achieved in the MRI images by applying to filter for noise reduction and image enhancement. Next, the brain tumor classification is done by using the Mask RCNN model. Then the SE-ResNet model was then used to segment the multiple MRI images. The proposed HDLN model effectively classifies and segments the MRI images with an overall accuracy of 98.53%. Experimental comparisons reveal that the tumor region is more accurately defined by the proposed HDLN model and can function in the proposed network model as a newly developed automatic diagnostic tool. The effective features with an accurate representation of brain tumors are also computed efficiently by the proposed HDLN model while comparing them to the existing models. In the future, we will focus on classifying and segmenting brain tumors using more difficult datasets. To increase the model's accuracy even further, we also want to optimize hyper-parameters and increase training samples.

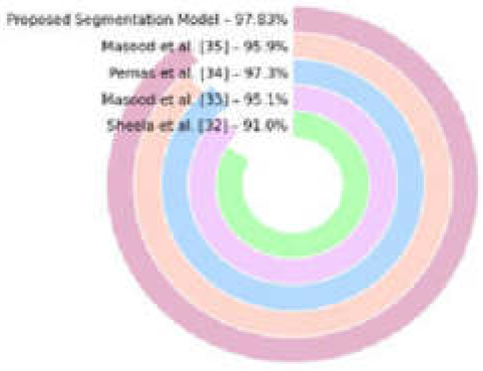

Fig. 4.6: Performance evaluation for segmentation

Acknowledgement. We declare that this manuscript is original, has not been published before, and is not currently being considered for publication elsewhere.

REFERENCES

[1] T. K. Barsiya, L. Bhargava, S. Agrawal, A. Tiwari, and A. Saxena, *Implementation of brain tumor segmentation using cnn deep learning algorithm*, in Soft Computing for Problem Solving: Proceedings of SocProS 2020, Volume 2, Springer, 2021, pp. 757–765.

[2] S. Deepak and P. Ameer, *Brain tumor classification using deep cnn features via transfer learning*, Computers in biology and medicine, 111 (2019), p. 103345.

[3] F. J. Díaz-Pernas, M. Martínez-Zarzuela, M. Antón-Rodríguez, and D. González-Ortega, *A deep learning approach for brain tumor classification and segmentation using a multiscale convolutional neural network*, in Healthcare, vol. 9, MDPI, 2021, p. 153.

[4] K. Ejaz, M. S. M. Rahim, A. Rehman, H. Chaudhry, T. Saba, A. Ejaz, and C. F. Ej, *Segmentation method for pathological brain tumor and accurate detection using mri*, International Journal of Advanced Computer Science and Applications, 9 (2018), pp. 394–401.

[5] S. Gull, S. Akbar, H. U. Khan, et al., *Automated detection of brain tumor through magnetic resonance images using convolutional neural network*, BioMed Research International, 2021 (2021).

[6] S. Gull, S. Akbar, and K. Safdar, *An interactive deep learning approach for brain tumor detection through 3d-magnetic resonance images*, in 2021 International Conference on Frontiers of Information Technology (FIT), IEEE, 2021, pp. 114–119.

[7] A. Gurunathan and B. Krishnan, *Detection and diagnosis of brain tumors using deep learning convolutional neural networks*, International Journal of Imaging Systems and Technology, 31 (2021), pp. 1174–1184.

[8] R. Hashemzehi, S. J. S. Mahdavi, M. Kheirabadi, and S. R. Kamel, *Detection of brain tumors from mri images base on deep learning using hybrid model cnn and nade*, biocybernetics and biomedical engineering, 40 (2020), pp. 1225–1232.

[9] S. Hossain, A. Chakrabarty, T. R. Gadekallu, M. Alazab, and M. J. Piran, *Vision transformers, ensemble model, and transfer learning leveraging explainable ai for brain tumor detection and classification*, IEEE Journal of Biomedical and Health Informatics, (2023).

[10] E. Irmak, *Multi-classification of brain tumor mri images using deep convolutional neural network with fully optimized framework*, Iranian Journal of Science and Technology, Transactions of Electrical Engineering, 45 (2021), pp. 1015–1036.

[11] A. Jijja and D. Rai, *Efficient mri segmentation and detection of brain tumor using convolutional neural network*, International Journal of Advanced Computer Science and Applications, 10 (2019).

[12] R. L. Kumar, J. Kakarla, B. V. Isunuri, and M. Singh, *Multi-class brain tumor classification using residual network and global average pooling*, Multimedia Tools and Applications, 80 (2021), pp. 13429–13438.

[13] M. J. Lakshmi and S. Nagaraja Rao, *Brain tumor magnetic resonance image classification: a deep learning approach*, Soft Computing, 26 (2022), pp. 6245–6253.

[14] D. Lamrani, B. Cherradi, O. El Gannour, M. A. Bouqentar, and L. Bahatti, *Brain tumor detection using mri images and convolutional neural network*, International Journal of Advanced Computer Science and Applications, 13 (2022).

[15] M. Madgi, S. Giraddi, G. Bharamagoudar, and M. Madhur, *Brain tumor classification and segmentation using deep learning*, in Smart Computing Techniques and Applications: Proceedings of the Fourth International Conference on Smart Computing and Informatics, Volume 2, Springer, 2021, pp. 201–208.

[16] S. Maqsood, R. Damaševičius, and R. Maskeliūnas, *Multi-modal brain tumor detection using deep neural network and*

multiclass svm, Medicina, 58 (2022), p. 1090.

[17] M. Masood, R. Maham, A. Javed, U. Tariq, M. A. Khan, and S. Kadry, *Brain mri analysis using deep neural network for medical of internet things applications*, Computers and Electrical Engineering, 103 (2022), p. 108386.

[18] M. Masood, T. Nazir, M. Nawaz, A. Mehmood, J. Rashid, H.-Y. Kwon, T. Mahmood, and A. Hussain, *A novel deep learning method for recognition and classification of brain tumors from mri images*, Diagnostics, 11 (2021), p. 744.

[19] S. A. Nawaz, D. M. Khan, and S. Qadri, *Brain tumor classification based on hybrid optimized multi-features analysis using magnetic resonance imaging dataset*, Applied Artificial Intelligence, 36 (2022), p. 2031824.

[20] M. M. Nayak and S. D. KA, *Effective mri based brain tumor detection using modified u-net model*, in 2022 IEEE 2nd Mysore Sub Section International Conference (MysuruCon), IEEE, 2022, pp. 1–7.

[21] A. Rehman, M. A. Khan, T. Saba, Z. Mehmood, U. Tariq, and N. Ayesha, *Microscopic brain tumor detection and classification using 3d cnn and feature selection architecture*, Microscopy Research and Technique, 84 (2021), pp. 133–149.

[22] T. Saba, A. S. Mohamed, M. El-Affendi, J. Amin, and M. Sharif, *Brain tumor detection using fusion of hand crafted and deep learning features*, Cognitive Systems Research, 59 (2020), pp. 221–230.

[23] D. K. Sahoo, S. Mishra, and M. N. Mohanty, *Brain tumor segmentation and classification from mri images using improved flicm segmentation and sca weight optimized wavelet-elm model*, International Journal of Advanced Computer Science and Applications, 13 (2022).

[24] S. Sajid, S. Hussain, and A. Sarwar, *Brain tumor detection and segmentation in mr images using deep learning*, Arabian Journal for Science and Engineering, 44 (2019), pp. 9249–9261.

[25] B. Saju, L. Thomas, F. Varghese, A. Prasad, and N. Tressa, *Deep learning-based brain tumor classification prototype using transfer learning*, in 2023 International Conference on Advances in Intelligent Computing and Applications (AICAPS), IEEE, 2023, pp. 1–7.

[26] H. A. Shah, F. Saeed, S. Yun, J.-H. Park, A. Paul, and J.-M. Kang, *A robust approach for brain tumor detection in magnetic resonance images using finetuned efficientnet*, IEEE Access, 10 (2022), pp. 65426–65438.

[27] M. Sharma, P. Sharma, R. Mittal, and K. Gupta, *Brain tumour detection using machine learning*, Journal of Electronics, 3 (2021), pp. 298–308.

[28] C. J. J. Sheela and G. Suganthi, *Brain tumor segmentation with radius contraction and expansion based initial contour detection for active contour model*, Multimedia Tools and Applications, 79 (2020), pp. 23793–23819.

[29] H. H. Sultan, N. M. Salem, and W. Al-Atabany, *Multi-classification of brain tumor images using deep neural network*, IEEE access, 7 (2019), pp. 69215–69225.

[30] Z. N. K. Swati, Q. Zhao, M. Kabir, F. Ali, Z. Ali, S. Ahmed, and J. Lu, *Brain tumor classification for mr images using transfer learning and fine-tuning*, Computerized Medical Imaging and Graphics, 75 (2019), pp. 34–46.

[31] T. Tazeen, M. Sarvagya, and M. Sarvagya, *Brain tumor segmentation and classification using multiple feature extraction and convolutional neural networks*, International Journal of Engineering and Advanced Technology, 10 (2021), pp. 23–27.

[32] M. Toğaçar, B. Ergen, and Z. Cömert, *Tumor type detection in brain mr images of the deep model developed using hypercolumn technique, attention modules, and residual blocks*, Medical & Biological Engineering & Computing, 59 (2021), pp. 57–70.

[33] M. Vimala and P. R. Kumar, *Real-time multi fractal ensemble analysis cnn model for optimizing brain tumor classification and survival prediction using svm*, Biomedical and Pharmacology Journal, 16 (2023).

[34] L. Wang, S. Wang, R. Chen, X. Qu, Y. Chen, S. Huang, and C. Liu, *Nested dilation networks for brain tumor segmentation based on magnetic resonance imaging*, Frontiers in Neuroscience, 13 (2019), p. 285.

[35] P. Windisch, P. Weber, C. Fürweger, F. Ehret, M. Kufeld, D. Zwahlen, and A. Muacevic, *Implementation of model explainability for a basic brain tumor detection using convolutional neural networks on mri slices*, Neuroradiology, 62 (2020), pp. 1515–1518.

Edited by: Katarzyna Wasielewska
Received: Dec 30, 2022
Accepted: Apr 8, 2023

AIMS AND SCOPE

The area of scalable computing has matured and reached a point where new issues and trends require a professional forum. SCPE will provide this avenue by publishing original refereed papers that address the present as well as the future of parallel and distributed computing. The journal will focus on algorithm development, implementation and execution on real-world parallel architectures, and application of parallel and distributed computing to the solution of real-life problems. Of particular interest are:

Expressiveness:
- high level languages,
- object oriented techniques,
- compiler technology for parallel computing,
- implementation techniques and their efficiency.

System engineering:
- programming environments,
- debugging tools,
- software libraries.

Performance:
- performance measurement: metrics, evaluation, visualization,
- performance improvement: resource allocation and scheduling, I/O, network throughput.

Applications:
- database,
- control systems,
- embedded systems,
- fault tolerance,
- industrial and business,
- real-time,
- scientific computing,
- visualization.

Future:
- limitations of current approaches,
- engineering trends and their consequences,
- novel parallel architectures.

Taking into account the extremely rapid pace of changes in the field SCPE is committed to fast turnaround of papers and a short publication time of accepted papers.

INSTRUCTIONS FOR CONTRIBUTORS

Proposals of Special Issues should be submitted to the editor-in-chief.

The language of the journal is English. SCPE publishes three categories of papers: overview papers, research papers and short communications. Electronic submissions are preferred. Overview papers and short communications should be submitted to the editor-in-chief. Research papers should be submitted to the editor whose research interests match the subject of the paper most closely. The list of editors' research interests can be found at the journal WWW site (`http://www.scpe.org`). Each paper appropriate to the journal will be refereed by a minimum of two referees.

There is no a priori limit on the length of overview papers. Research papers should be limited to approximately 20 pages, while short communications should not exceed 5 pages. A 50–100 word abstract should be included.

Upon acceptance the authors will be asked to transfer copyright of the article to the publisher. The authors will be required to prepare the text in $\LaTeX\,2_\varepsilon$ using the journal document class file (based on the SIAM's `siamltex.clo` document class, available at the journal WWW site). Figures must be prepared in encapsulated PostScript and appropriately incorporated into the text. The bibliography should be formatted using the SIAM convention. Detailed instructions for the Authors are available on the SCPE WWW site at `http://www.scpe.org`.

Contributions are accepted for review on the understanding that the same work has not been published and that it is not being considered for publication elsewhere. Technical reports can be submitted. Substantially revised versions of papers published in not easily accessible conference proceedings can also be submitted. The editor-in-chief should be notified at the time of submission and the author is responsible for obtaining the necessary copyright releases for all copyrighted material.