2023 International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)

Load Balancing of Cloud Resources for Real-Time Tasks Management using Deep Learning

1st Savita Yadav,
Department of CSE(IOT)
Noida Institute of Engineering and Technology
Greater Noida, India
savita.yadav@niet.co.in

2nd Bhawana Goel
Department of Electrical Electronics and Communication Engineering,
Galgotias University,
Greater Noida, India,
bhawana.goel@Galgotiasuniversity.edu.in

2023 International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE) | 979-8-3503-4745-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICDCECE57866.2023.10150624

***Abstract*—Cloud computing is an emerging technology that improves the efficiency of computers by sharing their resources. New cloud-based apps mean more work for existing servers. Although much research has been done in the field of Cloud computing, there are still challenges that must be overcome in order to improve performance and provide the greatest possible customer happiness. In this work, we cover the Machine Learning methods used for assembling VM squads with certain CPUs in mind. Quality of Service (QoS) is achieved through the application of DL methods, which entails the increased efficiency with which resources are used and deployed, the decrease in waiting time, response time, and costs that results from this, and the resulting improvement in system reliability due to the more even distribution of load across all equipment. For those interested in load balancing algorithms, this article provides a thorough discussion of many machine learning and deep learning-based methods.**

Keywords—Real-Time Tasks, Deep Learning, Load Balancing and Cloud Resources

I. INTRODUCTION

It is impossible to overestimate the significance of health education at this time. It is generally agreed that aspects not directly related to medical treatment are more important in determining one's overall social and physical health. Population changes, classroom dynamics, family composition, cultural mores and practises, market forces, international commerce, and environmental upheaval all fall under this category of external factors. By using an all-encompassing approach, we can empower individuals and communities to improve their health, foster more public health leadership, promote healthy policies across areas, and set up long-term, sustainable health care systems. Many people now suffer from allergic responses and wheezing as a consequence of their lifestyle choices; here, we analyse the elements at work in this shift by analysing many case studies. Given the findings of the study, we may probably expect. The process of data mining is utilised for this purpose. Data mining techniques, that are used to unearth concepts or patterns in data, are of considerable assistance to the decision-making process. This study presents a data mining analysis of allergy-induced asthma utilising the agents paradigm and clustering techniques. Thanks to the agents made accessible by agent-based systems that use techniques for data mining [1] [2], we can further the proposed study. In order to spot infiltration attempts, the authors of this study use data mining techniques. Thus, we use trust to select and deploy non-threatening intrusion detection system (IDS) nodes, which monitor each of the other nodes in the network in real time in search of any signs of hostile intent. We determined that the proposed IDS greatly reduced the rate of false rate compared to other existing IDS by modelling it in NS-2, indicating its superiority. RELATIONAL EFFORT Enhanced Adaptive Acknowledgment (EAACK) was proposed as an intrusion detection system (IDS) for mobile ad hoc networks (MANETs) by Shakshuki et al., 2013. Their strategy calls for every confirmation of receipt to be recorded either by sender and verified by the receiver. They proved that their method, which employs digital signatures made utilizing DSA and RSA, is secure against a variety of attacks. Digitally signing all acknowledgements adds computational complexity to their method, however. July 2018 ISSN 1819-6608 Issue 14 Volume 13 Number 1 Journal of Applied Sciences and Engineering | Asian Research Publishing Network | 2006-2018 (ARPN). Everyone is fighting for their independence and won't give it up easy. www.arpnjournals.com 4388 Marti et al., 2000 provided an IDS approach for MANET that makes use of two modules dubbed the Monitoring and the Pathrater [3] [4]. In this configuration, the Watchdog acts as an IDS for the MANET, scanning the traffic on the MANET's next hop for any signs of intrusion. Once the Watchdog determines that the following hop in the channel did not transmit the packets within the allotted delay, it will increment the node's failure number. For a node to be marked as malbehaving by the Watchdog, its failure counter must increase over a certain threshold. The Pathrater is then used to tell the routing system to avoid delivering data via the pinpointed nodes. The problem with this strategy is that it relies on the Watchdog's ever-vigilant monitoring to detect intrusions. IDS for TWOACK MANETs was proposed by Liu et al., 2007. including the requirement that all data packets be acknowledged while travelling along a path with much more than 3 consecutive nodes. Once a packet is received from the node in the route immediately before the final node, that node must respond using its own hop count to confirm that it has received the packet [5] [6]. Node X knows the message was successfully transmitted from node X reach node Z through the entry point Y once it receives the TWOACK packet (the first of the three consecutive nodes along the route). If node Y receives the TWOACK packet before node Z, both are marked as malicious. However, there is a drawback to this method in that it forces the network to work more to route TWOACK packets. In order for wireless sensor networks to detect intrusions, a decentralized self-learning, energy-aware, and low-complexity protocol was proposed by Misra et al., 2010. In order to create a low-power IDS, they use a Learning Automata (LA) on bandwidth technique with a probabilistic model. Through their experiment, they proved that their technique could detect malicious data packets and remove them from a WSN. The key drawback of this method is that the LA needs a lot of practise to get good results. Haddadi and Sarram (2010) introduced a WLAN IDS paradigm that integrates abuse and anomaly - based intrusion IDS modules. Potentially problematic with this approach is that abuse based and anomaly - based intrusion IDSs may not have equal response times. It's also a significant computational load to

have two IDSs evaluate the same data stream. Mohanapriya and Krishnamurthi, 2014 propose a lightweight, energy-efficient, and normal peers intrusion detection system to safeguard MANETs from the grey hole attack. However, in order for the IDS to function properly for their method, it must operate in full duplex mode, which dramatically increases the power requirements of the IDS. Liu et al. (2006) offered a game-theoretic framework for investigating the dynamics of potential attack pairs in wireless Ad-hoc Networks using a Bayes formulation [7] [8]. According to their suggested Bayesian hybrid detection technique for the defender, a less powerful lightweight module is used to analyse the enemy's type, while a more powerful heavyweight module acts as the last line of defence. When the Nash Equilibrium (NE) for the assaulter Bayesian game was compared between a static and dynamic setup, the researchers concluded that the latter model was the more realistic choice since it enabled the defence to dynamically update its belief in the opponent player's malice. The negative of their strategy is that it complicates the construction of a valid posterior distribution on the attacker player's nefarious intent. Liu (2003) proposed an incentive-based framework to model the motivations, objectives, and strategies of an attacker, based on a formalisation of game theory (AIOS). The author built a theoretical groundwork for AIOS modelling that takes into account the interdependency of AIOS, defender objectives, and defender tactics, hence enabling automated inference of AIOS. Thanks to AIOS modelling, defences may be able to guess what methods of attack the attacker would likely use. Insights from AIOS may help us evaluate risks and anticipate damages more precisely. The biggest issue with the plan, though, is that it assumes you know all there is to know about the game right from the get. Chen et al., 2001 propose a paradigm that makes use of two game-theoretic strategies for efficiently deploying an IDS. In the first tactic, the attacker and the intrusion protection agent's interactions are represented and evaluated with the use of noncooperative game theory. The resultant security value is calculated using Nash equilibrium for a hybrid model [9] [10].

II. Related Work

Real-time inquiry of huge surveillance video data is crucial to smart urban applications like community security and intelligent transportation. Owing to limited resources, edge devices cannot execute complicated computer vision with latency and high accuracy, whereas traditional cloud-based techniques are not practical due to high transmission latency and exorbitant bandwidth cost. Given the infeasibility of cloud-only and edge-only systems, we introduce SurveilEdge, a cooperative sky platform for actual inquiries of huge surveillance video streams. In particular, we build a convolutional neural network (CNN) learning scheme to shorten training time with high accuracy and an intelligence task allocator to balance load among processing nodes and satisfy the latency-accuracy tradeoffs for real-time requests. Utilizing real-world surveillance video datasets, we develop SurveilEdge on a prototype 1 with numerous edge devices and a public Cloud. Evaluation findings show that SurveilEdge can achieve up to 7× less bandwidth cost and 5.4× quicker query fast response than the cloud-only solution, and up to 43.9% query accuracy and 15.8× speedup compared to edge-only alternatives [11]. In the proposed study, we use deep learning to allocate mobile platform resources dynamically. Users may install ad-hoc programmes on real-world mobile platforms. These apps may effect system execution speed, space complexity, and other runnable applications. To avoid such concerns, we allocated runtime resources for mobile platform data storage. When system connected to cloud data server it store whole file system on distant Virtual Machine (VM) and anytime a single programme needed which quickly install starting as remote computer to local device. For the suggested system, we applied deep learning-based Convolutional Neural Network (CNN) method with Tensorflow environment to decrease data storage and extraction time [12]. A modified double Q-learning technique is used to load balance virtual servers in this article. A load balancing controller implements the algorithm, which uses software - defined network technologies to leverage user demands. The findings show a significant decrease in disgruntled cloud users compared to popular algorithms. In conclusion, this study will lead cloud load balancing solutions that demand greater QoS [13]. This paper presents a workload characterization-based technique for scheduling bursty workloads on a scalability serverless architecture using a machine learning (ML) framework. To prevent SLA breaches, we load balance the inference workload using Amazon Web Services (AWS) ML platforms SageMaker and Lambda. We analyse our strategy using a deep learning-based recommender system [14]. In this research, we suggest a deep training and queuing theory approach to predict short-term resource demand for computer resources. Experiments demonstrate that the suggested model suggests SLA violation 5% better than the baseline method. With performance measures, the model increases resource elasticity [15].

III. Proposed Work

Load balancing is a crucial component of cloud computing that ensures a consistent workload across all of the various servers, network interfaces, storage devices, and virtual machines (VMs) running on those server farms (Mittal & Dubey, 2017). Task scheduling on VMs hosted on physical nodes might lead to situations in which certain VMs are overutilized while others are underused. The makespan of a virtual machine (the time it takes to do all the tasks assigned to it) grows as it is overworked. Underutilized VMs, on the other hand, shorten makespan but raise resource usage costs since the existing resources are not put to their full potential. As a result, controlling and balancing the makespan and price parameters is essential in order to distribute the workload evenly among the VMs. Makespan reductions should not raise resource consumption costs, and vice versa. In this study, a Deep Learning-based Deadlineconstrained, Dynamic Virtual Machine Provisioning and Load Balancing (DLD-PLB) Architecture for Workflows is presented and developed. Utilizing a Deep Learning-based method, an optimal timetable for virtual machines has been developed. The suggested framework has been developed using the Genome process tasks as input. A hybrid approach based Deadline-constrained, Variable VM Provision and Load Balancing (HDD-PLB) framework for Workflow execution has been provided, and its makespan and cost have been calculated and compared to those of our previous proposed framework for performing multiple optimization (Kaur A et al.,2018a; Kaur A. et al.,2018b). Composite Predict-EarliestFinish Time (PEFT) with ACO was used in previous load balancing proposals to maximise the efficiency of underused virtual machines (VMs), while a hybrid PEFT-Bat technique was offered to make the most of available surplus VMs. PROPOSED METHODOLOGY FOR THE DLD-PLB FRAMEWORK In this study, we use the power of TensorFlow with the Keras deep learning toolkit to analyse a collection of workflows.

With TensorFlow, you can try out new training and optimization methods in a sandbox environment. Because of TensorFlow's versatility in handling both massive datasets (in the form of processes) and cloud computing's (in the form of heterogeneous environments), it has been included into the proposed DLDPLB framework's schedule and load balancing models. Fig. 1 depicts the process for the suggested framework. Tasks are gathered via parsing the workflow. Since deep learning techniques are often used to enormous datasets, the Genome sequence has been used as the input process because of the high volume of jobs it contains. There is a monetary and time investment associated with completing each assignment. Using deep learning, regression transforms a function (f) into a continuous output (y). On the basis of their known computing time and cost, a regression prediction model has been developed to estimate the continuous scheduling (unknown variable) of activities (known variables). In contrast, a classification model employs a mapping function to predict continuous output variables given input data. As the input jobs are not to be labelled, but rather a schedule is to be prepared based on their computing time and overall cost, classification predictive modelling has not been employed in the current study. As a result, a deep learning regress model has been implemented. Training a Convolutional Neural Network (CNN) is quite similar to training a traditional neural network, with the added complexity that comes from the use of convolution operations. A filter that produces a feature map performs convolution on the input. When using the suggested scheduling model of the DLD-PLB framework, the job that can be completed in the least amount of time is prioritised. For the DLD-PLB framework, the suggested deep learning method uses a CNN with three hidden layers:

- Layers of Convolution
- A layer that collects water
- The Role of the Relu

In order to eliminate dependencies between jobs, a convolution is employed to mix them together, and this results in a large dataset of tasks that can be utilised as input to the task allocation model of the suggested scheme. The characteristics relevant to the job at hand have been retrieved throughout the convolution process. The input undergoes many convolution procedures, each of which employs a unique filter to produce distinctive feature maps. The convolution layer combines these maps into a single output. After this last stage of processing, the output is sent into a kernel function (ReLU) to become non-linear. The ReLU function is sent into the hidden layers as activation for each one. As expressed in the Rectified Linear Unit (ReLU) notation, f(x) = max (0, x). ReLU function returns x if x is positive and 0 else.

In the current work, both the time and the money required to complete a job have been taken into account. For the purpose of further training and learning, weights have been assigned to these retrieved characteristics. The pooling layer, which follows the convolution layer, helps reduce the computational complexity of the learning process and shorten the time it takes to train input jobs. Following the allocation of tasks to VMs based on the projected schedule, the total cost and time required to execute the tasks are calculated. The outcomes are compared to a hybrid heuristic-metaheuristic approach for load balancing that was suggested before.

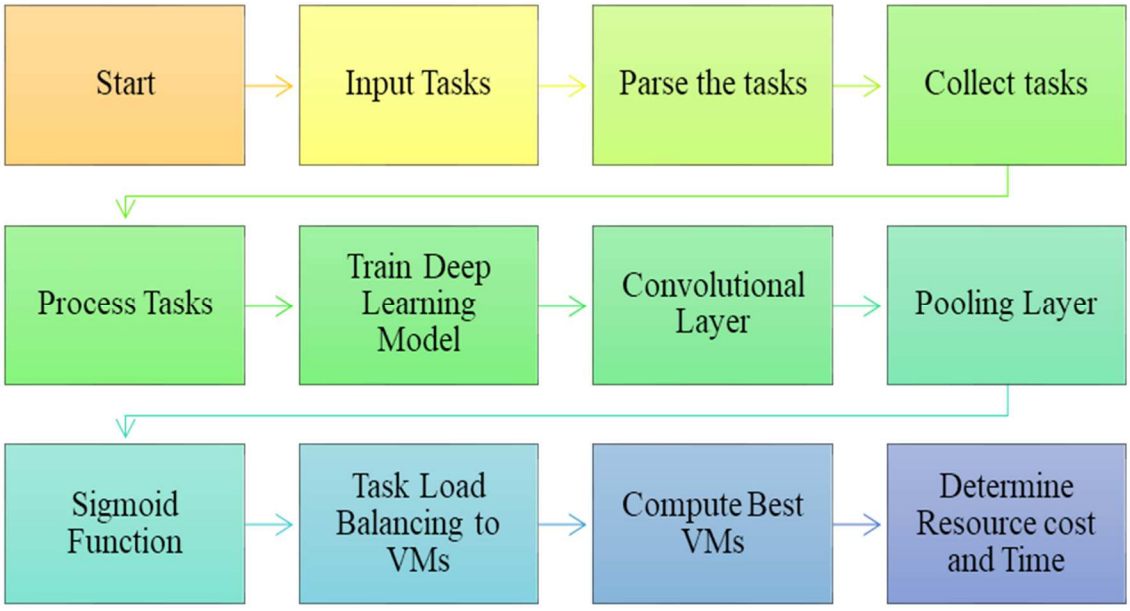


Fig. 1 System Flowchart

The suggested architecture (fig. 1) is based on a concept in which virtual machines share a single physical host's hardware (processor, memory, and disc). A variable number of virtual machines (VMs) have been launched on a single host computer. The IAAS cloud concept has two layers: hardware and virtualization. Included are the various computing, memory, and storage components that make up a cloud datacenter. A hypervisor is used to create several virtual machines (VMs) on a single piece of host hardware. Cloud workflow simulator (CWS) has been used to replicate a cloud environment in which to test the proposed framework. Workflow execution now requires an exponentially growing number of virtual machines. Virtual machines are generated on a host computer (host). The host's resource (CPU cycles and memory) is split across all the virtual machines running on its hardware. The hypervisor performs extensive resource sharing, which improves usage of actual machine (host) resources as the quantity of VMs increases.

The task scheduling model has been fed a very complex workflow including many jobs. The input pathway has been used as training data since it contains jobs with dependencies. The convolution layer is responsible for extracting the features of jobs, such as the amount of time and money spent computing. By calculating the linear combination of the input with the filter, the convolved characteristics (extracted features) are produced. That is to say, a filter operation has been applied to the feature map that was produced as a result of the convolution. We have applied this procedure to the jobs in such a way that only those that fail to meet our overall completion time and expense targets will be filtered out. Multiple convnets and filtering techniques were used to accomplish this. A massive list of activities, along with a rough timeline, have been compiled based on time and budget constraints. These values are the input task characteristics that have been retrieved. On the premise of the retrieved characteristics, weights have been allocated to the input jobs. For experimental reasons, scientists utilise an application called Genome, which is a workflow management system.

A directed acyclic graph (DAG) represents the workflow activities (DAG) G(T,E) such that 'T' is the set of n tasks {t1,t2,….tn} in order to run on a virtual machine, and 'E' is the collection of edges reflecting the interdependencies between the jobs that make up the workflow. In addition, in a directed acyclic graph (DAG), an entrance task (tentry) has no parent node and an exiting task (texit) has no child nodes. The transition period between these jobs is instantaneous. Fig. 2 demonstrates that the tentry-t1 and tentry-t2 edges are both unweighted. Equally unweighted are the edges connecting nodes t6 and t7 to texit.

The activities of the Genome process are what are meant by "cloud tasks" in the proposed algorithm. This workflow is ideal for implementing the suggested deep learning framework because of its huge database in terms of the overall number of jobs. The labels represent the criteria used in the analysis; in this case, time and money. The number of weights is determined by the number of features, which in this case is 2. Labels are given the values W 1 (time) and W 2 (frequency) (cost). An "epoch" is a single round in a longer training session. Each hidden uses a non-linear non - linear activation to extract features during convolution. For each k, Y[k] represents the result of one buried layer with ' k ' nodes $[h_1, h_2, \ldots h_k]$ during an epoch is computed as (1):

$$y[k] = v^* h[k] \quad (1)$$

where "v" represents the edge weight vector from the hidden node to the output node, and $v = [v_1, v_2, \ldots, v_k]$. The proposed deep learning technique utilises a total of two hidden layers. Weights for the future are used to set these layers up.

(W_1) and cost (W_2) metrics. A key epoch member has been initialised with the help of gradient descent optimization. In addition, the value for'n'inputs at each concealed node is calculated as (2) and (3):

$$h[k] = f(\sum_{i=1}^{n} w[k] * x[i]) \quad (2)$$

CNN with ' l ' hidden layers, $h^1 = f(w^l * h^{l-1}), l \geq 2$ (3)

Rectified Linear Unit (ReLU) activation functions are often used in suggested algorithms since they are inexpensive to calculate and need fewer mathematical calculations. To that end, it is generally agreed that this function is the optimal activation function for deep Neural Networks. The non-linear ReLU functional is mapped to a linear one through the ReLU mapping. This has been done in present work as (4):

$$f'(x) = \frac{d\left(\ln\left(1+ x^{x[n]*h[n]}\right)\right)}{dx} \quad (4)$$

The second layer then does a weighted sum on the that input before firing depending on yet another ReLu activation. However, the suggested technique avoids using other convolution layers like sigmoid & tanh functions, which need processing heavy calculations owing to their dense activations. In a perfect world, just a small percentage of the network's neurons would need to be active. Effective resource usage schedule prediction algorithm for time-sensitive process tasks

Input: Outcome of time-sensitive cloud tasks: a schedule forecast that maximises available resources

Step 1: Initial weight of labels W_1 and W_2 and member of epoch with gradient descent optimisation.

Step 2: For (K ≤ epoch)

Do

$y[k] = X[k] * h[n]$ (feature extraction)

$y[n] = \sum_{k=1}^{n} x[K] * h[K]$ (feature extraction for all the input tasks)

$x[n] \leftarrow$ Numberoffeatures

$f'^{(x)} = \frac{d(\ln(1+x^{x[n]*h[n]}))}{dx}$ end

$f'(x) \leftarrow$ ReluLayerMapping

Step 3: Update weight by optimization

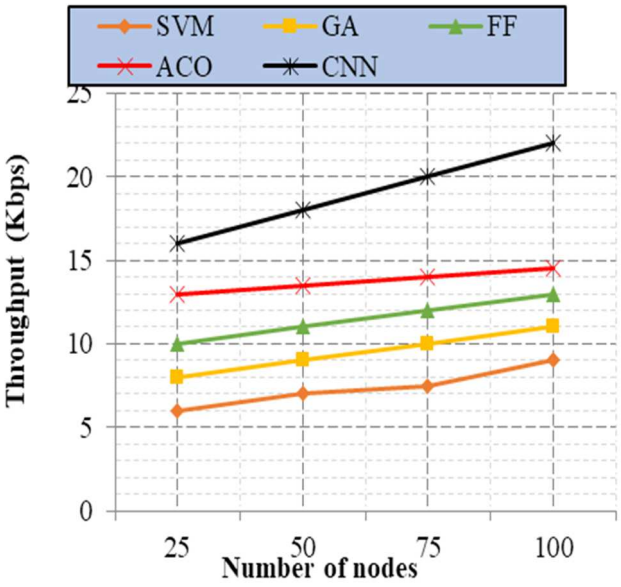
$$\Delta W_{x[n]}^{K} = d\left(I_K^{x[n]}\right) y_x^n$$

ΔW ← Update Weights

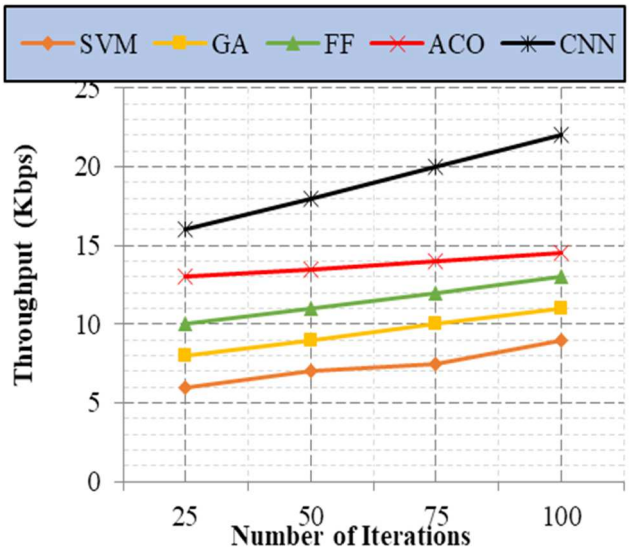
Step 4: Make Model

IV. Results & Discussion

The simulation results obtained after deploying the deep learning-based strategy for forecasting an optimum schedule of the input process (Genome) tasks have been used to evaluate the algorithms' performance. Python (PyCharm) was used to develop the DL model, and "Tensorflow" was employed as the DL model's backend for storing the DL training data of jobs. In this paper, I compare the timetable anticipated using a deep learning technique to one forecasted using a more traditional method. The cloud-based workflow simulator has been updated to include hybrid heuristic-metaheuristic methods. Makespan results of the suggested framework using deep learning and hybrid-heuristic-metaheuristic techniques are provided. The virtual machines (VMs) count is a dynamically determined range from 2 to 32. The findings of the suggested load balancing model using a deep learning algorithm have been analysed, and they have been shown to be superior than those of the prior model for task scheduling in the proposed framework, which relied on a mix of heuristic and metaheuristic techniques, which is given in Fig. 2 and Fig. 3.

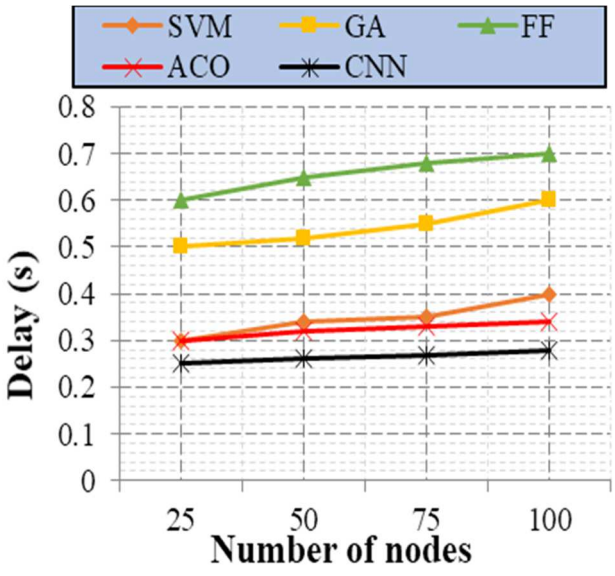


(a)

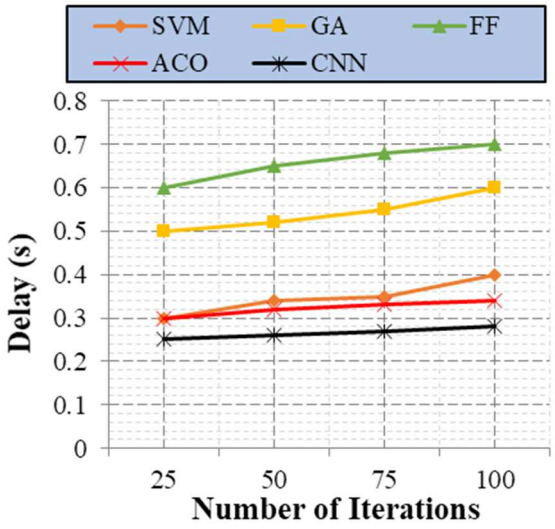


(b)

Fig. 2 (a) Throughput vs Number of nodes,(b) Throughput vs Number of Iterations



(a)



(b)

Fig. 3 (a) Delay vs Number of nodes,(b) Delay vs Number of Iterations

V. CONCLUSION

Another new suggestion is a Deep Learning Algorithm-based method for optimising load distribution in the cloud. Conclusions The suggested load balancing model based on deep training algorithm has been shown to provide better results than the prior model for load balancing in the proposed framework, which was focused on hybrid-heuristic-metaheuristic approaches. Many questions have been thrown up by this study's suggested DLD-PLB framework, which is built on a deep learning algorithm. Due to the fact that cloud computing takes use of recent developments in computer technology, especially internet-based technology, to provide services in the forms of IAAS, PAAS, and SAAS. Another new cloud computing paradigm is Machine Learning-as-a-Service (MLAAS). To train tasks and execute deep learning, large datasets may be stored and processed using cloud services. Key firms in the cloud computing industry (Google, Amazon, and Microsoft) are employing deep learning methods to efficiently manage massive amounts of data (also known as "Big Data"), making the cloud computing architecture a good fit for them. Amazon Deep Learning services, Microsoft Azure Deep Learning, and Cloud Based Artificial Intelligence are just a few examples of MLAAS offerings from these three cloud computing powerhouses that facilitate rapid training model and deployment with minimal need for specialised knowledge in data science. The cloud is an ideal environment for deep learning due to its capacity to store large amounts of data in both organised and unstructured formats, as well as its support for virtualization and scalability.

REFERENCES

[1] Wang, S., Yang, S., & Zhao, C. (2020). SurveilEdge: Real-time Video Query based on Collaborative Cloud-Edge Deep Learning. IEEE INFOCOM 2020 - IEEE Conference on Computer Communications, 2519-2528.

[2] (2019). Dynamic Resource Allocation and Memory Management using Deep Convolutional Neural Network. International Journal of Engineering and Advanced Technology.

[3] Tennakoon, D., Chowdhury, M.U., & Luan, T.H. (2018). Cloud-based load balancing using double Q-learning for improved Quality of Service. Wireless Networks, 1-8.

[4] Chahal, D., Palepu, S.C., Mishra, M., & Singhal, R. (2020). SLA-aware Workload Scheduling Using Hybrid Cloud Services. Proceedings of the 1st Workshop on High Performance Serverless Computing.

[5] Chudasama, V., & Bhavsar, M.D. (2020). A Dynamic Prediction for Elastic Resource Allocation in Hybrid Cloud Environment. Scalable Comput. Pract. Exp., 21, 661-672.

[6] Pradhan, A., Bisoy, S.K., Kautish, S., Jasser, M.B., & Mohamed, A.W. (2022). Intelligent Decision-Making of Load Balancing Using Deep Reinforcement Learning and Parallel PSO in Cloud Environment. IEEE Access, 10, 76939-76952.

[7] Jameel, E.J., & Ibrahim, A.A. (2022). Fog and Cloud Load Balancing Using Regression Based Reccurent Deep Learning Algorithm. 2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), 1-5.

[8] Belkout, N.E., Zeraoulia, K., Shahzad, M.N., Liu, L., & Yuan, B. (2022). A Load Balancing and Routing Strategy in Fog Computing using Deep Reinforcement Learning. 2022 International Conference on Electrical, Computer and Energy Technologies (ICECET), 1-8.

[9] Ouamri, M.A., Barb, G., Singh, D., & Alexa, F. (2022). Load Balancing Optimization in Software-Defined Wide Area Networking (SD-WAN) using Deep Reinforcement Learning. 2022 International Symposium on Electronics and Telecommunications (ISETC), 1-6.

[10] (2020). DEEP LEARNING BASED LOAD BALANCING USING MULTIDIMENSIONAL QUEUING LOAD OPTIMIZATION ALGORITHM FOR CLOUD ENVIRONMENT. October-2020.

[11] Aditiya, & Rana, R. (2022). Simulation of load balancing algorithms using cloud analyst for distinct regions IT resources. International Journal of Research in Engineering and Innovation.

[12] Li, P., Xie, W., Yuan, Y., Chen, C., & Wan, S. (2022). Deep Reinforcement Learning for Load Balancing of Edge Servers in IoV. Mobile Networks and Applications, 27, 1461 - 1474.

[13] Kruekaew, B., & Kimpan, W. (2022). Multi-Objective Task Scheduling Optimization for Load Balancing in Cloud Computing Environment Using Hybrid Artificial Bee Colony Algorithm With Reinforcement Learning. IEEE Access, 10, 17803-17818.

[14] Kaur, A., Kaur, B., Singh, P., Devgan, M.S., & Toor, H.K. (2020). Load Balancing Optimization Based on Deep Learning Approach in Cloud Environment. International Journal of Information Technology and Computer Science, 12, 8-18.

[15] Sawwashere, S.S., Ambhaikar, A., & Malik, L.G. (2022). IL2ATL: Design of a high efficiency load balancing model using augmented deep incremental transfer learning. 2022 10th International Conference on Emerging Trends in Engineering and Technology - Signal and Information Processing (ICETET-SIP-22), 1-6.