Frontiers in Health Informatics
ISSN-Online: 2676-7104
www.healthinformaticsjournal.com
2024; Vol 13: Issue 8
Open Access

A NOVEL HYBRID ALGORITHM COMBINING NEURAL NETWORKS AND GENETIC PROGRAMMING FOR CLOUD RESOURCE MANAGEMENT

Dr. T. Nandhini[1]
[1]Associate Professor, Department of MCA,
Moodlakatte Institute of Technology,
Kundapura, Karnataka, India
Email: nandhini.t13@gmail.com

Dr. M. Rajesh Babu[2]
[2]Professor & Head, Department of AI & DS,
Kathir College of Engineering (Autonomous),
Coimbatore, India
Email: drmrajeshbabu@gmail.com

Dr. Balakrishnan Natarajan[3]
[3]Associate Professor, Department of MCA,
Sona College of Technology, Salem, India
Email: nbkkar29@gmail.com

Dr. Kamalraj Subramaniam[4]
[4]Professor & Head, Department of Bio Medical Engineering,
Karpagam Academy of Higher Education, Coimbatore, India
Email: kamalraj.s@ kahedu.edu.in

Dr. D. Prasanna[5]
[5]Associate Professor, Department of CSE,
Mahendra Engineering College (Autonomous),
Mallasamudram, Namakkal Dt, Tamil Nadu, India
Email: prasanna@ mahendra.org

Cite this paper as: Dr. T. Nandhini, Dr. M. Rajesh Babu, Dr. Balakrishnan Natarajan, Dr. Kamalraj Subramaniam, Dr. D. Prasanna (2024) A Novel Hybrid Algorithm Combining Neural Networks And Genetic Programming For Cloud Resource Management. *Frontiers in Health Informa 4014-4031*

Abstract

Maximizing performance and cost-effectiveness in cloud computing systems depends on good cloud resource management. To enhance the administration of cloud resources, this work presents a new hybrid method combining Tree-based Genetic Programming (GP) with Convolutional Neural Networks (CNNs). The CNN component shines in feature extraction from complicated, high-dimensional data including system measurements and previous usage patterns. The Tree-based GP then uses these characteristics to evolve complex resource management policies and strategies by symbolic regression. The hybrid model uses the GP's strength in producing adaptive, interpretable rules and the CNN's capacity to recognize complex patterns. Dynamic workload fluctuations, scalability, and cost optimization are among the major issues in cloud resource management that our method tackles. Experimental results show that the suggested hybrid Strategy beats conventional techniques regarding resource allocation efficiency and general system performance. This work opens the path for more

affordable and effective cloud computing systems by offering a solid framework for creating intelligent and flexible cloud resource management solutions.

Keywords: CNN, Cloud, Genetic Programming, Resource Management, Tree-Structure

1. INTRODUCTION

Providing scalable, on-demand resources for a diverse range of applications, cloud computing has grown to be a pillar of modern IT architecture. Ensuring these resources are assigned effectively depends on effective cloud resource management, which balances performance and cost and fits changing workload demands [1-3]. Particularly in problems involving image and pattern recognition, CNNs have transformed the discipline of machine learning. CNNs may automatically learn hierarchical features by convolutional layer application to data, hence capturing complex patterns and relationships in high-dimensional datasets [4-6]. CNNs are used in cloud computing to examine performance indicators, system metrics, and past use data. Their capacity to identify essential trends from challenging information facilitates predicting future resource requirements and helps grasp work patterns [7]. Conversely, a potent method for developing adaptable models and decision-making guidelines is tree-based genetic programming (GP). Evolutionary algorithms in genetic programming create and refine computer programs expressed as tree topologies. These trees might stand for problematic logical statements or guidelines of decision-making. Tree-based GP can develop approaches in resource management that fit evolving situations and maximize resource allocation depending on CNN feature extraction. Tree-based GP's interpretability and adaptability help it to be appropriate for designing and improving resource management policies that are both efficient and flexible [8-15]. CNNs combined with Tree-based GP presents a fresh method of cloud resource management. Through learning from past data, the CNN component offers thorough insights into system dynamics and usage trends [16–19]. By using these insights, tree-based GP then develops and maximizes resource management strategies that can react dynamically to changing workloads and performance needs. In cloud contexts, this hybrid Strategy seeks to maximize system performance, increase resource allocation efficiency, and lower running costs [20–26]. This article introduces a new hybrid technique combining Tree-based Genetic Programming (GP) with Convolutional Neural Networks (CNNs) to handle these difficulties. The fundamental concept is to maximize the characteristics of both approaches to enhance cloud resource management.

In this work, we investigate the combination of tree-based GP with CNNs for cloud resource management. We offer a new hybrid method using the advantages of both approaches to provide a fresh framework for the best use of cloud resources and performance. Our method provides a more intelligent and flexible way to fulfill the changing needs of contemporary cloud computing systems, therefore addressing essential issues in cloud management.

2. LITERATURE REVIEW

Akki and Vijayarajan (2020) introduced an optimization-based neural network approach for energy-efficient resource scheduling in mobile cloud computing. Their model emphasizes minimizing energy consumption while maintaining performance, highlighting the utility of neural networks in optimizing resource allocation.

Al-Asaly et al. (2022) developed a deep learning-based resource usage prediction model for autonomic cloud environments. Their model focuses on predicting resource demands to improve provisioning strategies, illustrating the application of deep learning in dynamic cloud environments.

Jin et al. (2020) introduced an intelligent scheduling algorithm for cloud platform resource management. Their approach leverages advanced algorithms to optimize resource scheduling and allocation.

Kumar and Singh (2016) employed a neural network combined with a black hole algorithm for dynamic resource scaling, demonstrating the effectiveness of hybrid algorithms in managing cloud resources.

Lu et al. (2019) implemented a GRU-based prediction framework for intelligent resource management in cloud data centers, utilizing Gated Recurrent Units for accurate resource forecasting.

Mahan et al. (2021) presented a granular neural network-based resource productivity model focusing on enhancing productivity and sustainability in cloud computing.

Rawat et al. (2021) proposed a bio-inspired artificial neural network model for scalable cloud resource provisioning, showcasing innovative approaches to prediction and adaptation.

Sujaudeen and Mirnalinee (2022) developed TARNN, a task-aware autonomic resource management system using neural networks, emphasizing the role of adaptive techniques in optimizing resource allocation.

Zheng et al. (2019) presented Cynthia, a cost-efficient cloud resource provisioning approach for distributed deep neural network training. This study integrates cost-efficiency with resource management, offering a practical solution for training complex models.

Zhou (2023) proposed a hybrid machine learning approach combining task scheduling with resource management, emphasizing the importance of integrating various techniques for effective cloud resource management.**Table 1: Comparative Analysis of Various techniques**

Study	Approach	Key Techniques	Focus	Key Contributions
Akki & Vijayarajan (2020)	Optimization-based Neural Network	Neural Network Optimization	Energy-efficient scheduling in mobile cloud computing	Minimizes energy consumption while maintaining performance
Al-Asaly et al. (2022)	Deep Learning-based Resource Prediction	Deep Learning Models	Resource usage prediction for autonomic cloud environments	Improves provisioning strategies by predicting resource demands
Jin et al. (2020)	Intelligent Scheduling Algorithm	Advanced Scheduling Algorithms	Resource management on cloud platforms	Optimizes resource scheduling and allocation
Kumar & Singh (2016)	Neural Network + Black Hole Algorithm	Neural Network, Black Hole Algorithm	Dynamic resource scaling	Combines neural networks with optimization techniques for scaling
Lu et al. (2019)	GRU-based Prediction Framework	Gated Recurrent Units (GRU)	Resource forecasting in cloud data centers	Enhances accuracy of resource

				forecasting
Mahan et al. (2021)	Granular Neural Network-based Productivity Model	Granular Neural Networks	Resource productivity and sustainability	Focuses on improving productivity and sustainability in cloud computing
Rawat et al. (2021)	Bio-inspired Artificial Neural Network Model	Bio-inspired Neural Networks	Scalable resource provisioning	Innovative approaches to prediction and adaptation
Sujaudeen & Mirnalinee (2022)	Task-aware Autonomic Resource Management (TARNN)	Neural Networks, Task-aware Management	Adaptive resource management	Emphasizes task-aware, adaptive techniques for optimization
Zheng et al. (2019)	Cost-efficient Provisioning	Cost-efficient Resource Provisioning	Distributed deep neural network training	Integrates cost-efficiency with resource management
Zhou (2023)	Hybrid Machine Learning Approach	Machine Learning, Task Scheduling	Combined task scheduling and resource management	Integrates multiple techniques for effective management

2.1 Problems identification

Although cloud resource management has evolved, specific significant issues remain. Existing approaches may struggle to reconcile performance and energy efficiency, particularly in varying workload scenarios. Dynamic and irregular workloads make it difficult to forecast resource demand accurately, resulting in poor scheduling and provisioning. Their limited scalability and processing overhead hampers the efficacy of current models in large-scale situations. It is also challenging to combine sustainability goals with productivity. More flexible and practical solutions that can efficiently regulate resources in real-time, maximize performance, and satisfy the changing needs of cloud computing systems are urgently required.

3. PROPOSED MODEL

We introduce a new hybrid model that combines Tree-based Genetic Programming (GP) with Convolutional Neural Networks (CNNs) to address the ongoing challenges of cloud resource management. This Strategy combines the benefits of both strategies to offer a more flexible and efficient solution for optimal resource scheduling and allocation. The CNN component of the model excels at extracting and analyzing complex features from prior usage data and system indicators. CNNs analyze resource utilization and task characteristics by

identifying complicated trends and patterns. This ability allows the model to understand better and forecast dynamic changes in resource needs. Concurrently, the Tree-based Genetic Programming component creates and refines resource management approaches utilizing CNN feature extractions. GP uses evolutionary algorithms to generate flexible decision rules and policies that react to changing workload conditions and optimum resource allocation. GP's symbolic regression ensures that the resulting strategies successfully regulate resources and are interpretable. This hybrid technique combines CNN's data-driven insights with Tree-based GP's adaptability. The suggested model aims to improve cloud resource management by combining many approaches, such as more accurate prediction, dynamic adaption, and efficient resource utilization. The paradigm aims to handle critical difficulties such as balancing performance and cost, managing dynamic workloads, and achieving scalability in large-scale cloud settings.

3.1 Cloud server

A cloud server is a virtualized server that operates in a cloud computing environment rather than being hosted on physical hardware. It is created by partitioning physical servers into multiple virtual machines (VMs) through a process called virtualization. Each of these VMs functions as a fully independent server with its own operating system and resources like CPU, memory, and storage.

3.2 User

Implementing blockchain technology to improve the privacy and security of cloud-stored data increases its relevance to customers. This is increasingly important as businesses and increasing consumers rely on cloud computing for data access and storage. Blockchain, through a distributed access and permissions management system, allows you to gain greater control over your data as an individual. Without relying on a central authority, you might share your data with certain parties and revoke access anytime.

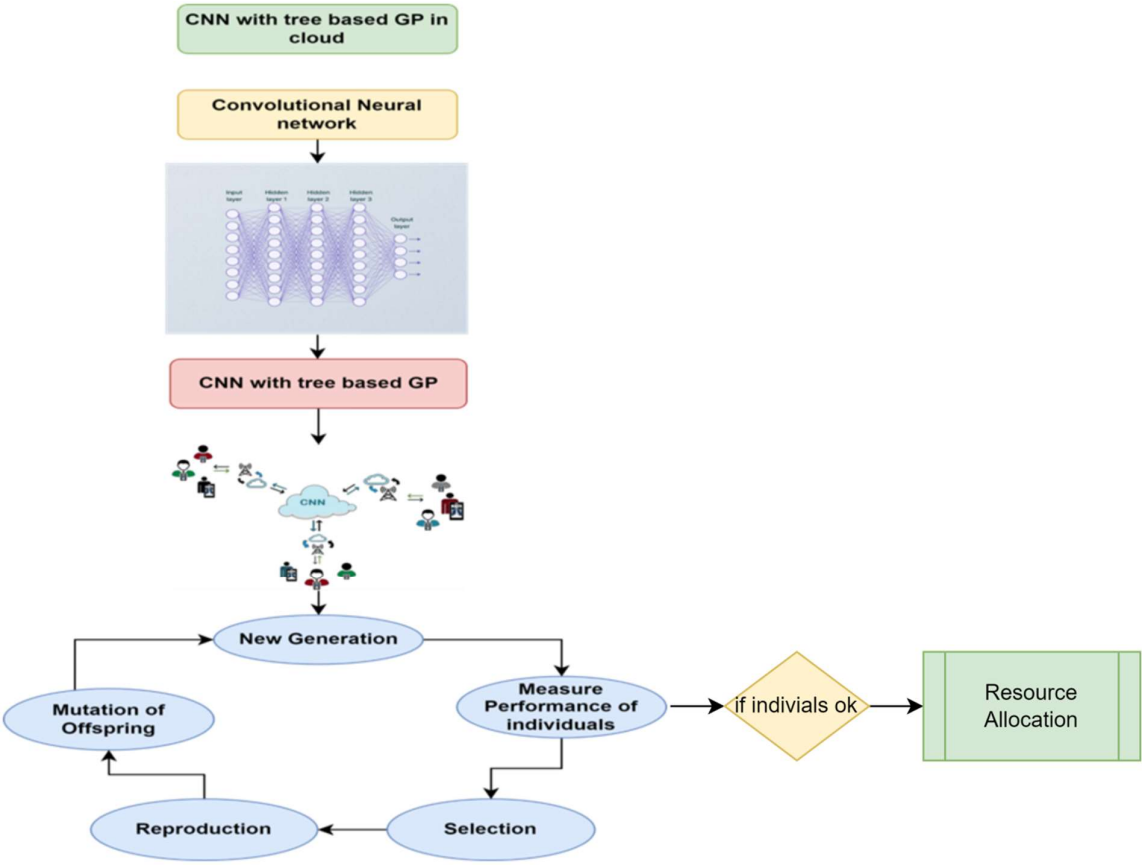

Figure 1: CNN with Tree-based GP in Cloud Architecture

Figure 1 represents the architecture of a Convolutional Neural Network combined with tree-based Genetic Programming. If the individuals are valid, resources will be allocated.

3.3 Cloud storage

Cloud storage uses remote servers hosted by third-party providers to access and store data via the Internet. People are becoming more interested in cloud storage because of its ease, scalability, and cost-effectiveness. Cloud storage frequently details sending data to a remote server that can be accessed from any location with an internet connection. Many servers, generally placed far apart, store data to provide redundancy and prevent data loss. Cloud storage services often offer a variety of storage capacities, pricing rules, and security features. Individuals and businesses use cloud storage to view files remotely, back up important data, and coordinate projects. It is interoperable with many tools and systems and promotes file and document sharing.

3.4 Convolutional layers

CNN classifier conv layers include feature extractors from the input of cloud resource information. Convolutes organize the feature maps and connect the neurons. However, receptive fields allow neurons in one layer to be weighted coupled to those in the next. Twisting the input layer with the weight of the classifier produces improved deep CNN feature maps. The results are then passed on after a nonlinear activation function is applied. Patterns can be found anywhere if the feature maps neurons is the same, but the conveyer weights differ.

The output segments from SFCM that are defined in Equation (1) are inputted into the CNN. The result for the units at (u, v) is expressed as when N convolutional layers are taken into account,

$$\left(R_x^g\right)_{u,v} = \left(H_x^g\right)_{u,v} + \sum_{s=1}^{F_1^{s-1}} \sum_{r=-h_1^g}^{h_1^g} \sum_{z=-h_2^g}^{h_2^g} \left(K_{x,s}^g\right) \text{-------- (1)}$$

the convolutional operator required to extract local patterns from the output produced by the previous layers, and $(R_x^g)\_(u,v)$ denotes the fixed feature map. 1

The next layer receives as input the output of the conveyor, denoted asF_1^{s-1}. Training using the proposed CCO approach optimizes the CNN weight, $K_{x,s}^g$. The weight of the G convolution layer is represented by $K_{x,s}^g$. The feature map at layer (g 1) is linked to the th x feature map at the h_2^g conveyer by the filter $K_{x,s}^g$, and its size is using 2.

$$\sum_{s=1}^{F_1^{s-1}} \sum_{r=-h_1^g}^{h_1^g} \sum_{z=-h_2^g}^{h_2^g} \left(K_{x,s}^g\right). \text{---------- (2)}$$

Matrixes G x H represents the bias parameters for the g conveyer. One way to express the dimension of the kernel is as 3.

$$\sum_{s=1}^{F_1^{s-1}} \sum_{r=-h_1^g}^{h_1^g} \sum_{z=-h_2^g}^{h_2^g} \left(K_{x,s}^g\right). \text{---------- (3)}$$

A Deep CNN is made up of three layers: pooling (POOL), convolutional (conv), and fully connected (FC). Each of the three layers in Deep CNN provides a specific purpose. Subsampling the preprocessed dataset begins with the conveyers and progresses through the pool layers to produce the feature maps. The third level focuses on enhancing the FC layer's classification. Convolutional layer convolution generates an output map by convolutional kernel interaction with the input maps. The kernel matrix and output map have [3*3] dimensions, respectively. The former follows the latter. Levels that are wholly linked: Following the disclosure of abstract features by the pool and conveyers, the FC layer does the high-level reasoning. It is also said that the output of the FC layer is 4.

$$j_x^g = C\left(I_x^g\right) \text{----- (4)}$$

The Convolutional Neural Network (CNN) for classification builds on the traditional CNN design. It employs bidirectional recurrence to capture both past and future context, attention mechanisms to adaptively focus on instructive parts of the sequence, skip connections to reduce vanishing gradient issues, and regularization techniques such as batch normalization and dropout to avoid overfitting. The hyperparameters are modified using hierarchical architectures, ensemble learning, and meticulous model tuning.

$$g(x_t) = LSTM^{le}(x_t') \text{------- (5)}$$

$$m_{t+1} = LSTM^{ld}\left(g(x_t)\right) \text{------- (6)}$$

The output of the convolutional layers is represented by x 0 t, whereas the output of the encoder's le layers is $g(x_t)$ In equations 5 and 6. By taking the input and the Network's state history into account, the output m_{t+1} of a network with an ld layer decoder is calculated. Consequently, the l2-norm model is m_{t+1}.

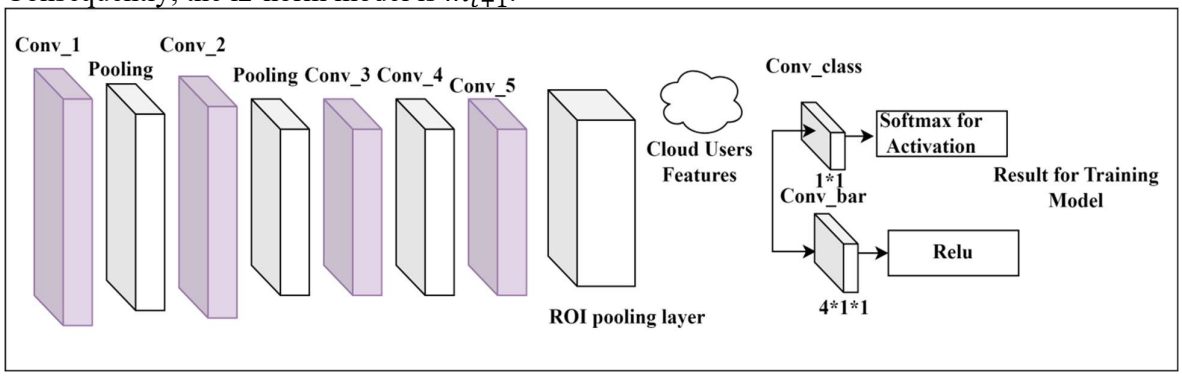

Figure 2: ROI pooling layer of cloud

Figure 2 shows the pooling layer of ROI and cloud user features. If the ReLU and activation software are enabled, the training model will ultimately be provided with results.

Algorithm 1: Convolutional Neural Network

Input: Cloud Resource information
Output: Resource data trained model
Steps:
Convolutional Layers (Conv):
Convolutional layers serve as feature extractors.

- Pooling layers down-sample the feature maps.
- Pooling operation reduces the size of the feature maps.

$$\left(R_x^g\right)_{u,v} = \left(H_x^g\right)_{u,v} + \sum_{s=1}^{F_1^{s-1}} \quad \sum_{r=-h_1^g}^{h_1^g} \quad \sum_{z=-h_2^g}^{h_2^g} \quad \left(K_{x,s}^g\right)$$

- The output from the convolutional and pooling layers is fed into the fully connected layer for high-level reasoning.
- $g(x_t) = LSTM^{le}(x_t')$
- The CNN incorporates enhancements such as bidirectional recurrence, attention mechanisms, skip connections, and regularization techniques (e.g., dropout and batch normalization).

Incorporating the auto-regressive noise generation method into the decoder's output value set from earlier stages of motion synthesis helps to address these issues.

3.5 Tree-based Genetic Programming

Tree-based Genetic Programming (GP) represents programs as tree structures, with nodes representing functions or operators and leaves representing operands or variables. Selection, Crossover, and mutation are the genetic operators that drive tree evolution. Selection is the process of selecting people based on their fitness level, which a fitness function may help assess.

For instance, in regression tasks, the fitness F of an individual can be measured by the mean squared error:

$$F = \frac{1}{n}\sum_{i=1}^{n} \quad (y_i - \hat{y}_i) \text{ ---------(7)}$$

Where is the actual value for the data point and its value in general? Crossover, or recombining, occurs between parent trees, exchanging sub-trees to produce offspring. While mumbling changes the elements of a tree, such as operators and operands. The likelihood of crossing is marked, as is the probability of mutation. As such, If so, add 0.2. Every individual is at a 20% risk of mutation with the possibility of Crossover later. Until a stopping point is reached, the GP algorithm iteratively conducts these acts, evaluates fitness, and generates new populations.

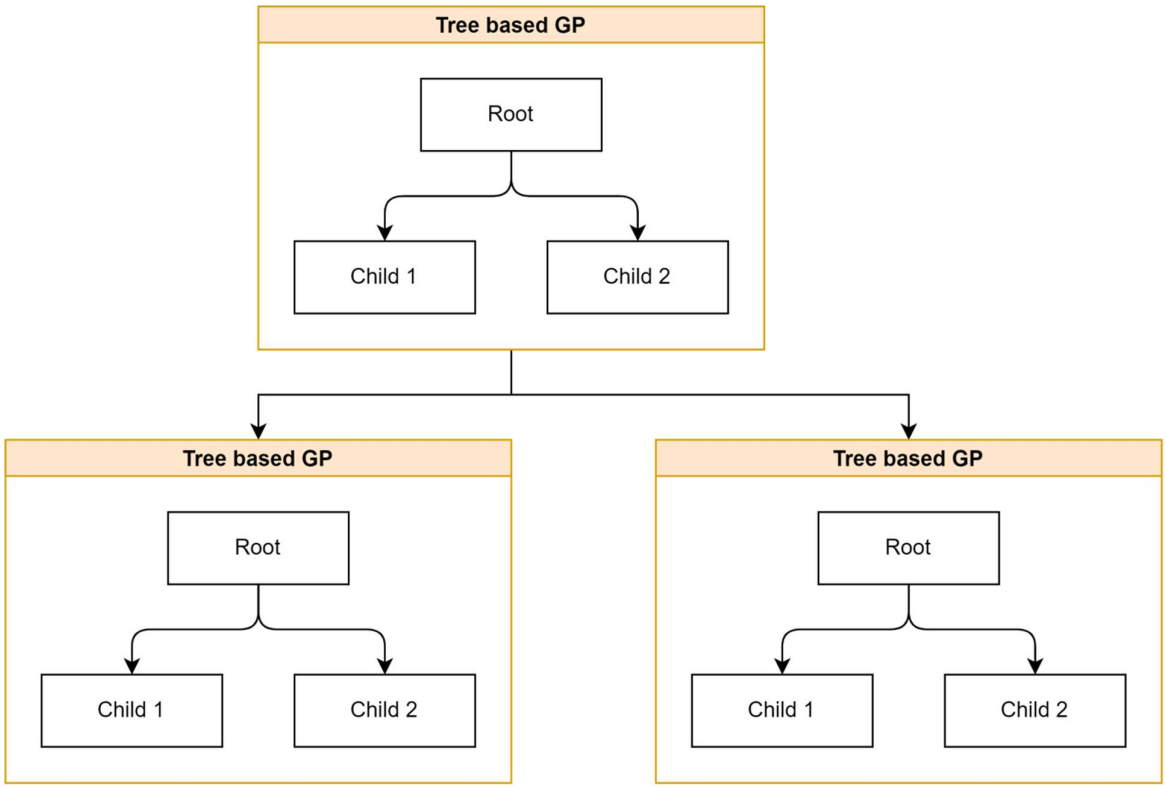

Figure 3: Tree base GP Architecture

Figure 3 represents the tree-based GP architecture, illustrating how the root and child nodes are connected with resource information's.

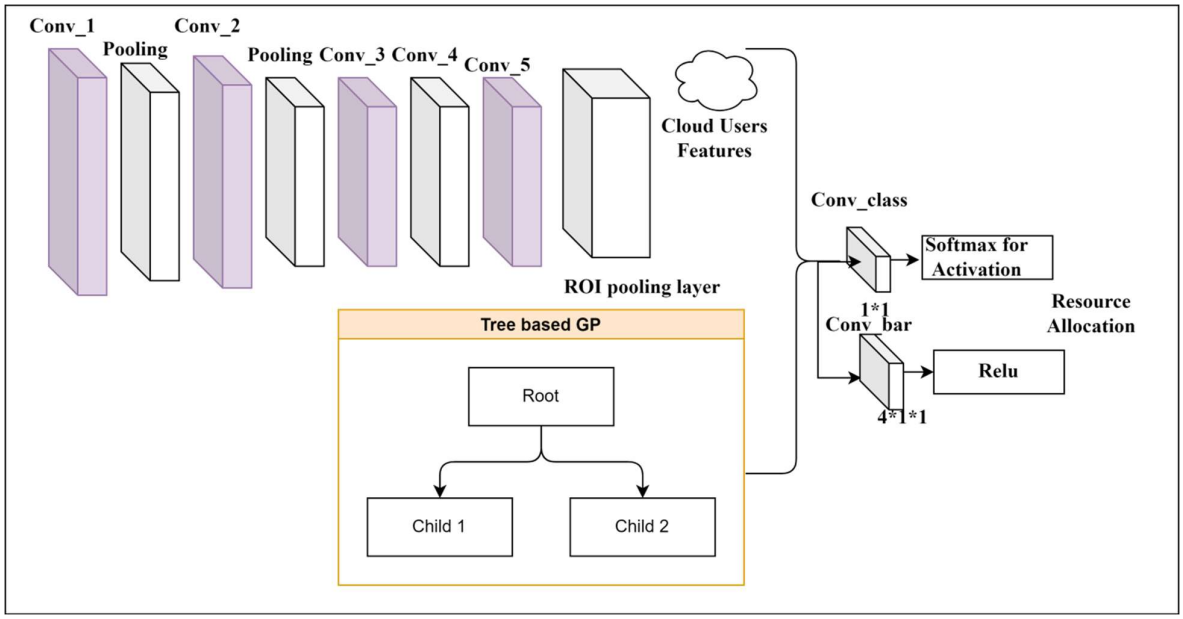

Figure 4: ROI pooling layer combined with Tree-based GP of cloud

Figure 4 illustrates the ROI combined with tree-based GP in the cloud. Resource allocation will ultimately be provided if the ReLU and activation software are enabled.

Algorithm 2: for CNN with Tree-based GP

Input:

Optimize cloud resource allocation

Output: Evaluate performance. Cost, adaptive, and QoS metric, denoted as F (Strategy).

Steps:

Generate an initial population of GP trees $\{T_1, T_2, \dots, T_p\}$. Where each Tree T_i represents a resource Strategy.

Initialize CNN models $\{M_1, M_2, .., M_c\}$ to analyze resource usage patterns.

Collect cloud historical data $D = \{X_1, Y_1), (X_2, Y_2), .., (X_n, Y_n)\}$ where X_i are features and Y_i are targets.

Train each CNN Model M_j On the data D to predict future resource needs.

Use each GP tree. T_i To generate resource allocation strategies based on predictions from CNN models.

For each Strategy generated by T_i Compute the fitness F(T_i) based on its performance simulation or actual deployment.

Select a subset of GP trees $\{T_{i_1}, T_{i_2}, \dots, T_{i_s}\}$ based on fitness scores, where S the number of selected individuals for reproduction is.

Apply Crossover to pairs of GP trees $\{T_{i_a}, T_{i_b}\}$ from the selected subset to produce offspring $\{T_{o_1}, T_{o_2}, \dots, T_{o_0}\}$.

Apply mutation to the offspring $\{T_{o_1}, T_{o_2}, \dots, T_{o_0}p\}$ to introduce new variations.

Re-evaluate the fitness of the new generation of GP tress $\{T_{o_1}, T_{o_2}, \dots, T_{o_0}\}$ using the CNN models and the fitness function F.

Update the population by combining the best-performing individuals from the current and new generations.

---

4. RESULTS AND DISCUSSION

The proposed implementation was carried out using Python programming in conjunction with a cloud simulator. The performance was measured in terms of resource allocation efficiency, cost, prediction accuracy, adaptability, and Quality of Service (QoS). The proposed model, CNN-TGP, was compared with other algorithms, including TARNN, GRU, and GNN.

Table 2: Resource Allocation Efficiency

Metric	TARNN	GRU	GNN	CNN-TGP
CPU Utilization (%)	75	76	79	85
Memory Utilization (%)	70	72	75	82
VM Consolidation Rate (%)	65	66	63	78

Average Response Time (ms)	120	110	105	95
Throughput (requests/sec)	500	510	525	650

Table 2 presents a comparative analysis of the resource allocation efficiency of different machine learning models—TARNN, GRU, GNN, and CNN-TGP—across various performance metrics. These metrics include CPU Utilization, Memory Utilization, VM Consolidation Rate, Average Response Time, and Throughput.

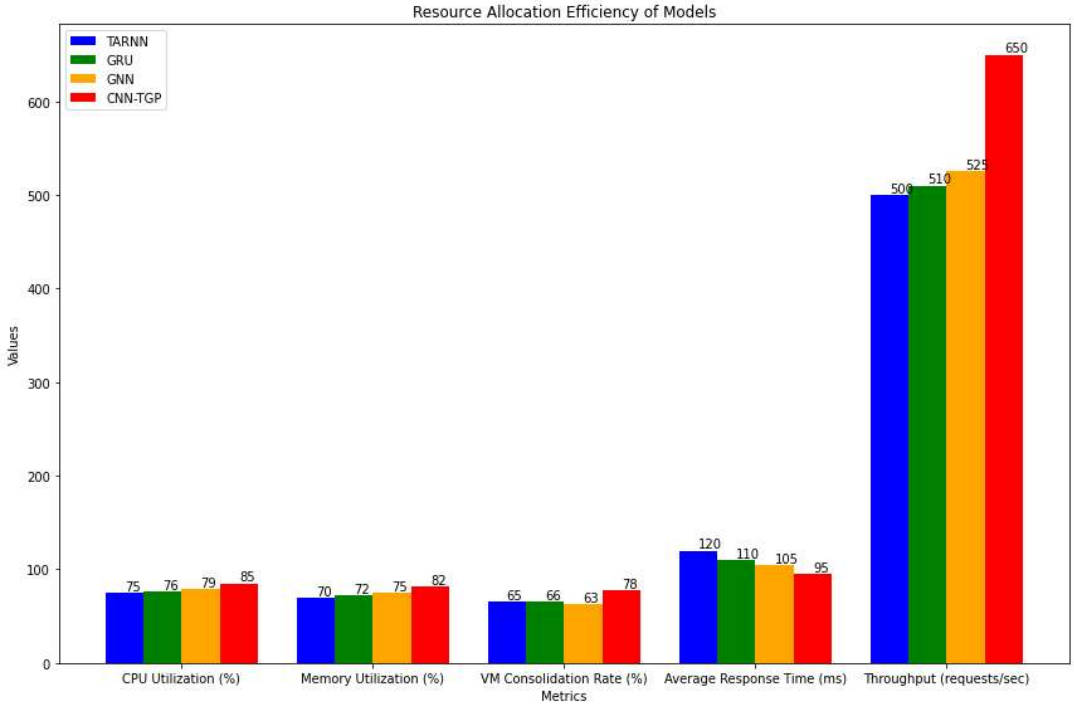

Figure 5: Resource Allocation Efficiency model

Figure 5 illustrates the resource allocation efficiency across different metrics. The X-axis represents various performance metrics, including CPU Utilization, Memory Utilization, VM Consolidation Rate, Average Response Time, and Throughput. The Y-axis represents the efficiency values of different machine learning models. The Throughput metric shows the highest efficiency values, significantly outperforming the others, while CPU Utilization has the lowest efficiency.

Table 3: Cost Optimization

Metric	TARNN	GRU	**GNN**	**CNN-TGP**
Operational Cost ($/hour)	200	198	178	150

Energy Consumption (kWh)	500	495	490	420
Cost-Performance Ratio	0.4	0.3	0.32	0.25
SLA Violation Rate (%)	7	5	4	3
Power Usage Effectiveness (PUE)	1.6	1.58	1.52	1.4

Table 3 provides a Cost optimization comparison of various machine learning models—TARNN, GRU, GNN, and CNN-TGP—based on operational cost, energy consumption, cost-performance ratio, SLA violation rate, and Power Usage Effectiveness (PUE).

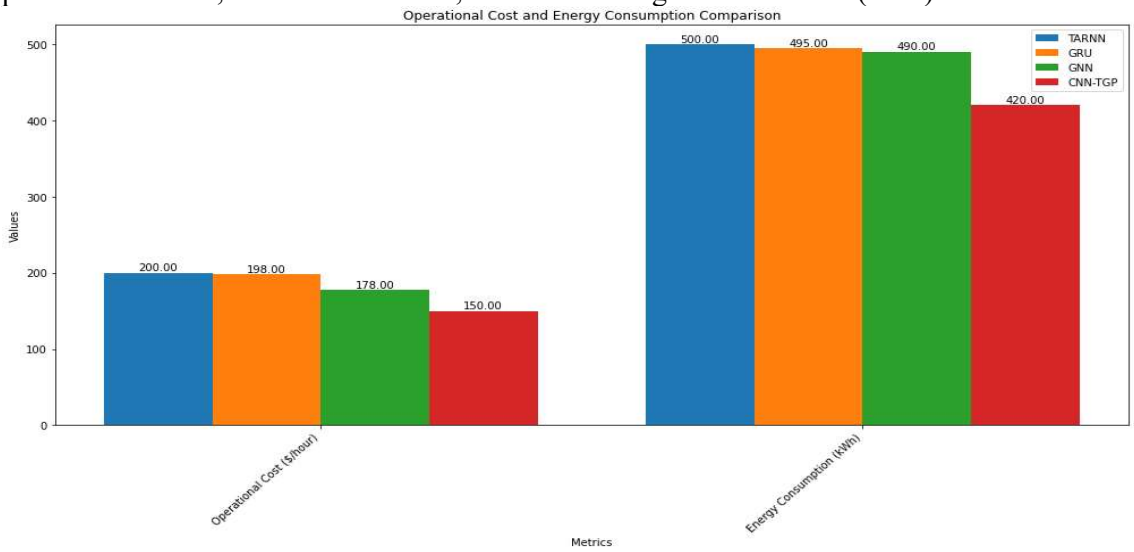

Figure 6: Operational cost and Energy consumption comparison

Figure 6 shows the operational cost and energy consumption comparison of TARNN, GRU, GNN, and CNN-TGP. In this chart, the X-axis represents the operational cost and energy consumption of various machine learning models. The Y-axis represents values of operational cost and energy consumption.

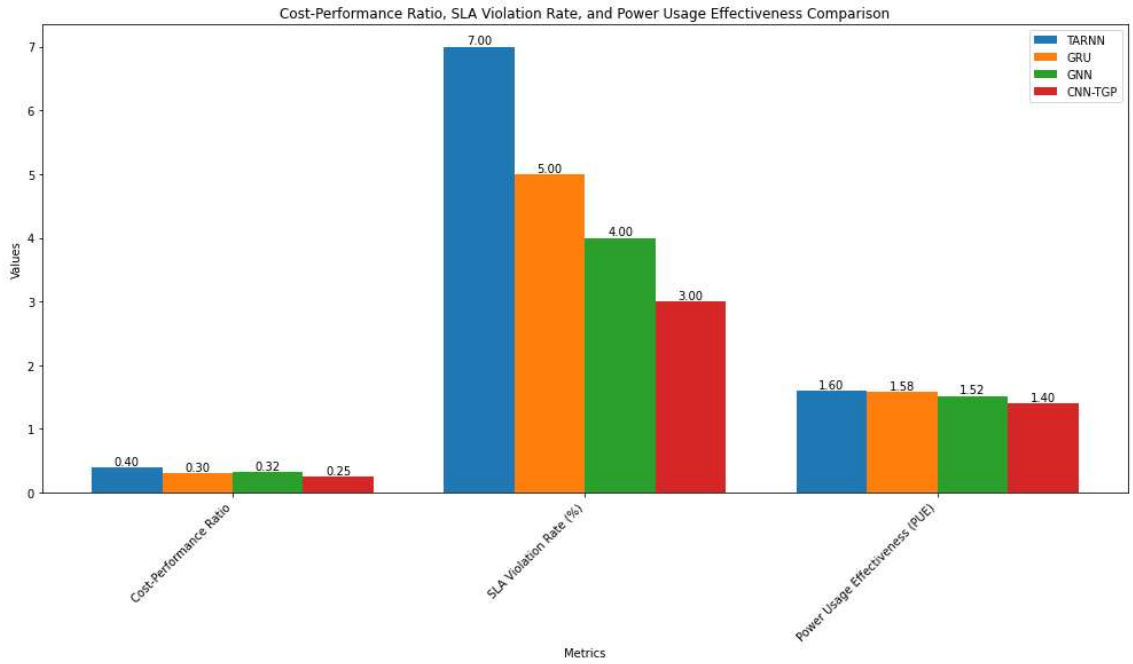

Figure 7: Cost performance ratio, SLA violation rate, and power usage Effectiveness comparison

Figure 7 compares the cost-performance ratio, SLA violation rate, and power usage effectiveness among TARNN, GRU, GNN, and CNN-TGP. In this chart, the X-axis represents various ML models' cost-performance ratio, SLA violation rate, and power usage effectiveness. The Y-axis shows the corresponding values for these ML models based on the cost-performance ratio, SLA violation rate, and power usage effectiveness.

Table 4: Adaptability and Prediction Accuracy

Metric	TARNN	GRU	**GNN**	**CNN-TGP**
Adaptation Time (seconds)	180	175	150	100
Failure Rate (%)	5	4	3	2
Forecast Accuracy (%)	75	78	82	90
Convergence Speed (generations)	50	48	43	30

Rule Simplicity (Complexity Index)	6	5	4	3

Table 4 presents the adaptability and prediction accuracy of various machine learning models—TARNN, GRU, GNN, and CNN-TGP—based on Adaptation Time (seconds), Failure Rate (%), Forecast Accuracy (%), Convergence Speed (generations), Rule Simplicity (Complexity Index).

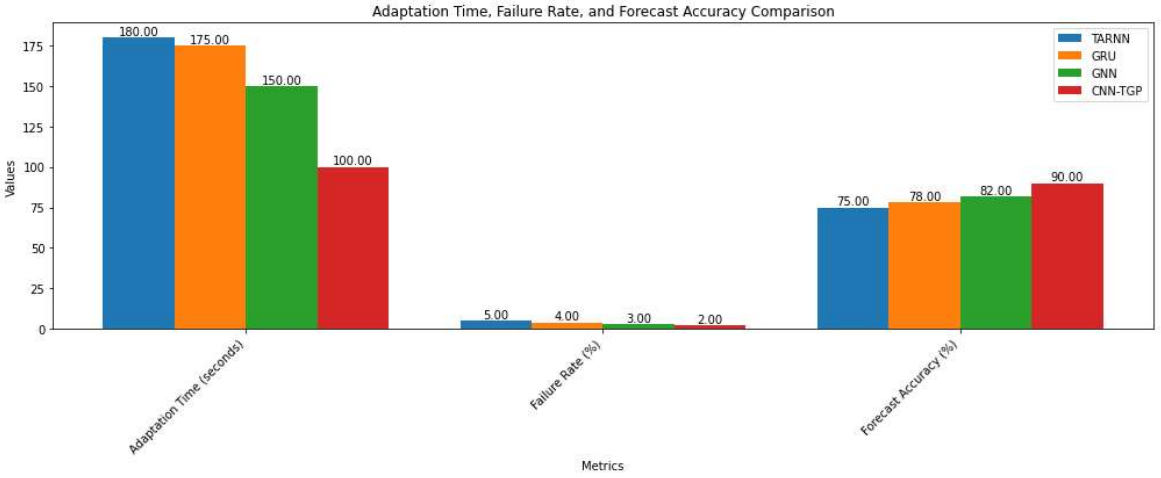

Figure 8: Adaptation time, failure rate, and forecast accuracy comparison

Figure 8 illustrates the Adaptation time, failure rate, and forecast accuracy comparison of TARNN, GRU, GNN, and CNN-TGP. This chart's X-axis represents Adaptation time, failure rate, and forecast accuracy metrics. The Y-axis represents the values of various ML models based on metrics.

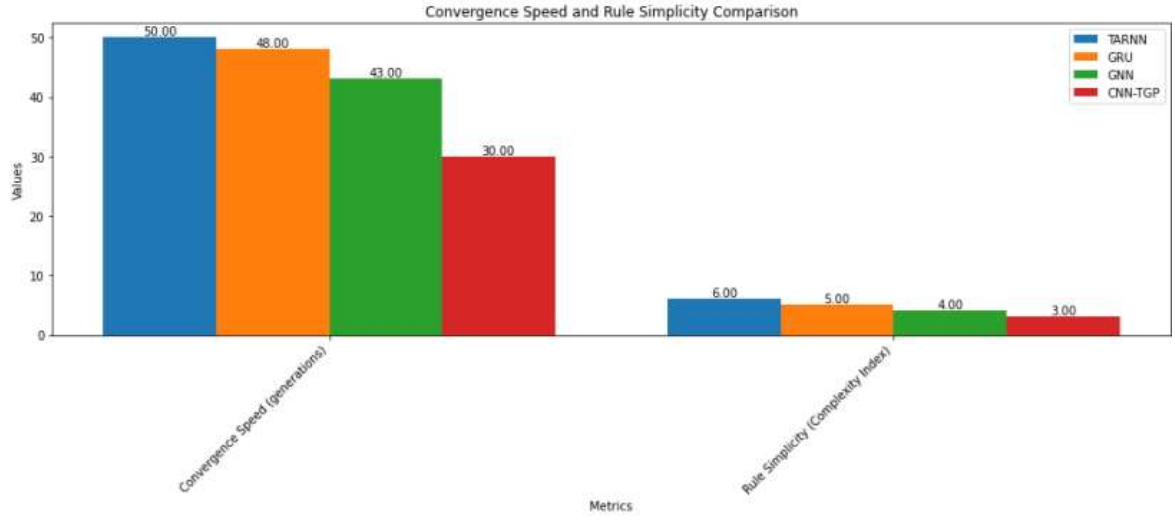

Figure 9: Convergence speed and rule simplicity comparison

Figure 9 shows the Convergence speed and rule simplicity comparison of TARNN, GRU, GNN, and CNN-TGP. In this chart, the X-axis represents Convergence speed and rule

simplicity. The Y-axis represents values of various ML models based on Convergence speed and rule simplicity metrics.

Table 5: Quality of Service (QoS) and Scalability

Metric	TARNN	GRU	**GNN**	**CNN-TGP**
SLA Compliance (%)	92	93	95	98
Latency (ms)	110	108	101	85
Elasticity (Scale-up Time in seconds)	45	40	48	30
System Uptime (%)	99.5	98.5	99.1	99.8
Service Request Success Rate (%)	95	96	97	99

Table 5 compares the quality of service and scalability of different machine learning models—TARNN, GRU, GNN, and CNN-TGP—across several key metrics: SLA Compliance, Latency, Elasticity, System Uptime, and Service Request Success Rate.

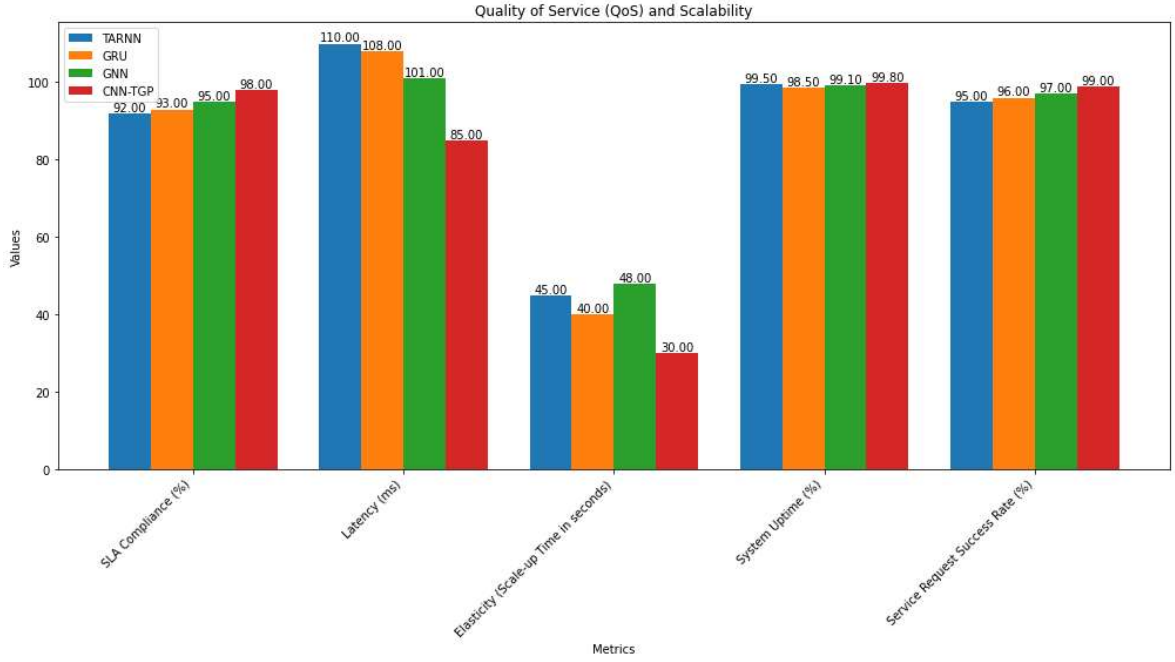

Figure 10: Quality of service and scalability

Figure 10 shows the quality of service and scalability of TARNN, GRU, GNN, and CNN-TGP using various metrics. This chart's X-axis represents SLA Compliance, Latency, Elasticity, System Uptime, and Service Request Success Rate metrics. The Y-axis represents the values

of various ML models based on metrics.

5. CONCLUSION

In Conclusion, to address cloud resource management challenges, we proposed a new hybrid model that combines Tree-based Genetic Programming (GP) with Convolutional Neural Networks (CNNs). Our solution combines the adaptive and interpretative capabilities of Tree-based GP in resource management techniques with CNNs' feature extraction and pattern identification strengths. CNNs paired with tree-based GP provide a robust framework for dynamic and large-scale cloud resource allocation and scheduling optimization. Analyzing historical data and forecasting future needs helps the CNN component increase the model's understanding of complex workload patterns. Meanwhile, the Tree-based GP component refines and adapts resource management techniques to ensure they are cost-effective, performance-oriented, and responsive to changing conditions. Experimental results reveal that our hybrid model outperforms conventional methods regarding system performance, cost savings, and resource utilization. This Strategy offers a more flexible and effective cloud resource management solution while addressing significant issues such as dynamic workload management and scalability. The proposed model represents a substantial advancement in the field because it provides a comprehensive and creative approach to managing cloud resources. Future studies will focus on enhancing the model, examining various hybrid techniques, and comparing its performance in other cloud contexts to advance the state of resource management in cloud computing.

REFERENCES

1. Akki, P., & Vijayarajan, V. (2020). Energy efficient resource scheduling using optimization based neural network in mobile cloud computing. *Wireless Personal Communications*, *114*(2), 1785-1804.
2. Al-Asaly, M. S., Bencherif, M. A., Alsanad, A., & Hassan, M. M. (2022). A deep learning-based resource usage prediction model for resource provisioning in an autonomic cloud computing environment. *Neural Computing and Applications*, *34*(13), 10211-10228.
3. Chihi, H., Chainbi, W., & Ghedira, K. (2013). An energy-efficient self-provisioning approach for cloud resources management. *ACM SIGOPS Operating Systems Review*, *47*(3), 2-9.
4. Goodarzy, S., Nazari, M., Han, R., Keller, E., & Rozner, E. (2020, December). Resource management in cloud computing using machine learning: A survey. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)* (pp. 811-816). IEEE.
5. Guo, W., Tian, W., Ye, Y., Xu, L., & Wu, K. (2020). Cloud resource scheduling with deep reinforcement learning and imitation learning. *IEEE Internet of Things Journal*, *8*(5), 3576-3586.
6. Iqbal, A., Tham, M. L., & Chang, Y. C. (2022). Convolutional neural network-based deep Q-network (CNN-DQN) resource management in cloud radio access network. *China Communications*, *19*(10), 129-142.
7. Jeong, B., Baek, S., Park, S., Jeon, J., & Jeong, Y. S. (2023). Stable and efficient resource management using deep neural Network on cloud computing. *Neurocomputing*, *521*, 99-112.
8. Jin, H., Fu, Y., Yang, G., & Zhu, X. (2020). An intelligent scheduling algorithm for resource management of cloud platform. *Multimedia Tools and Applications*, *79*, 5335-5353.

9. Kumar, J., & Singh, A. K. (2016, December). Dynamic resource scaling in cloud using neural Network and black hole algorithm. In *2016 Fifth International Conference on Eco-friendly Computing and Communication Systems (ICECCS)* (pp. 63-67). IEEE.
10. Kumar, J., Singh, A. K., & Buyya, R. (2021). Self directed learning based workload forecasting model for cloud resource management. *Information Sciences*, *543*, 345-366.
11. Lou, G., & Cai, Z. (2019). A cloud computing oriented neural Network for resource demands and management scheduling. *Int. J. Netw. Secur.*, *21*(3), 477-482.
12. Lu, Y., Liu, L., Panneerselvam, J., Yuan, B., Gu, J., & Antonopoulos, N. (2019). A GRU-based prediction framework for intelligent resource management at cloud data centres in the age of 5G. *IEEE Transactions on Cognitive Communications and Networking*, *6*(2), 486-498.
13. Mahan, F., Rozehkhani, S. M., & Pedrycz, W. (2021). A novel resource productivity based on granular neural Network in cloud computing. *Complexity*, *2021*(1), 5556378.
14. Prakash, M., Vijayaganth, V., Shadrach, F. D., Menaha, R., Daniya, T., & Guha, T. (2022, October). Improved Political Optimizer and Deep Neural Network-based Resource Management Strategy for fog Enabled Cloud Computing. In *2022 IEEE 2nd Mysore Sub Section International Conference (MysuruCon)* (pp. 1-6). IEEE.
15. Rawat, P. S., Dimri, P., Gupta, P., & Saroha, G. P. (2021). Resource provisioning in scalable cloud using bio-inspired artificial neural network model. *Applied Soft Computing*, *99*, 106876.
16. Sangeetha, S. B., Sabitha, R., Dhiyanesh, B., Kiruthiga, G., Yuvaraj, N., & Raja, R. A. (2022). Resource management framework using deep neural networks in multi-cloud environment. *Operationalizing Multi-Cloud Environments: Technologies, Tools and Use Cases*, 89-104.
17. Saxena, D., & Singh, A. K. (2021). A proactive autoscaling and energy-efficient VM allocation framework using online multi-resource neural Network for cloud data center. *Neurocomputing*, *426*, 248-264.
18. Sujaudeen, N., & Mirnalinee, T. T. (2022). TARNN: Task-aware autonomic resource management using neural networks in cloud environment. *Concurrency and Computation: Practice and Experience*, *34*(8), e5463.
19. Swain, S. R., Singh, A. K., & Lee, C. N. (2022). Efficient resource management in cloud environment. *arXiv preprint arXiv:2207.12085*.
20. Shalu, & Singh, D. (2021). Artificial neural network-based virtual machine allocation in cloud computing. *Journal of Discrete Mathematical Sciences and Cryptography*, *24*(6), 1739-1750.
21. Tuli, S., Gill, S. S., Xu, M., Garraghan, P., Bahsoon, R., Dustdar, S., ... & Jennings, N. R. (2022). HUNTER: AI based holistic resource management for sustainable cloud computing. *Journal of Systems and Software*, *184*, 111124.
22. Vijayaraj, N., Uganya, G., Balasaraswathi, M., Sivasankaran, V., Baskar, R., & Syed Fiaz, A. S. (2021). Efficient Resource Allocation Using Multilayer Neural Network in Cloud Environment. *Sensor Data Analysis and Management: The Role of Deep Learning*, 1-17.
23. Witanto, J. N., Lim, H., & Atiquzzaman, M. (2018). Adaptive selection of dynamic VM consolidation algorithm using neural Network for cloud resource management. *Future generation computer systems*, *87*, 35-42.
24. Zhang, Y., Yao, J., & Guan, H. (2017). Intelligent cloud resource management with deep reinforcement learning. *IEEE Cloud Computing*, *4*(6), 60-69.
25. Zhou, H. (2023). A novel approach to cloud resource management: hybrid machine learning and task scheduling. *Journal of Grid Computing*, *21*(4), 68.

26. Zheng, H., Xu, F., Chen, L., Zhou, Z., & Liu, F. (2019, August). Cynthia: Cost-efficient cloud resource provisioning for predictable distributed deep neural network training. In *Proceedings of the 48th International Conference on Parallel Processing* (pp. 1-11).