Journal of Information Systems Engineering and Management
2025, 10(41s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Enhancing Cloud Resource Allocation with a Hybrid Deep Learning-Based Framework: A Comparative Study

Suhad Ibrahim Mohammad[1], Ziyad Tariq Mustafa Al-Ta'i [1]
[1]Department of Computer Science, College of Science, University of Diyala, Diyala, Iraq
Email: Scicomphd222307@uodiyala.edu.iq, Ziyad1964tariq@uodiyala.edu.iq

ARTICLE INFO

Received: 26 Dec 2024

Revised: 15 Feb 2025

Accepted: 21 Feb 2025

ABSTRACT

Optimal resource provisioning remains a core problem in cloud computing, particularly in dynamic and heterogeneous environments. Traditional provisioning techniques are not very agile in adapting to varied workloads and user demands in real-time, resulting in inefficient underutilization or overprovisioning of resources. In this paper, a hybrid deep learning-enabled method to enhance cloud resource provisioning using predictive modeling and discerning decision-making is presented. The architecture blends Convolutional Neural Networks (CNNs), Gated Recurrent Unit (GRU), and Long Short-Term Memory (LSTM) networks to learn effectively spatial and temporal connections in historical workload and resource usage data. High-level spatial features are learned from multi-dimensional input matrices using CNNs. In contrast, temporal dynamics and long-term dependencies are learned using GRU and LSTM units, enabling better prediction of future resource demands. Experimental comparisons on standard cloud simulation data sets show that the proposed hybrid model significantly outperforms traditional deep learning and rule-based policy allocation policies in terms of allocation accuracy, task completion time, and system throughout. The findings highlight the potential of advanced deep learning models to enhance resource allocation optimization, reduce running costs, and ensure enhanced service quality in real-time cloud computing.

Keywords: Cloud computing, Deep learning, Resource allocation, virtualization

Introduction

Cloud computing has emerged as the dominating model for computing services delivery over the internet. It provides end-users with on-demand access to scalable, elastic computing resources, enabling companies to execute applications without the need for significant investments in hardware. The growing use of cloud services across industries—spanning from healthcare and education to finance and e-commerce—has placed overwhelming pressure on cloud service providers to utilize resources to the best possible extent while providing high performance, reliability, and affordability [1].

One of the biggest challenges of cloud computing is the allocation of resources, i.e., allocating virtualized computing resources (CPU, memory, storage, bandwidth) to contending tasks and users in an optimum manner. In real-world setups, cloud setups are dominated by extremely dynamic and unpredictable workloads, where demands from users may fluctuate rapidly over small intervals of time. In such settings, traditional resource allocation techniques, which are usually rule-based or threshold-based policies, are not adequate. These are not adaptive and are slow to respond to sudden fluctuations in demand, resulting in resource utilization, overprovisioning, latency in services, and increased operational expenses [1,2 and 6].

Journal of Information Systems Engineering and Management
2025, 10(41s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/
Research Article

In recent years, DL approaches have emerged as promising intelligent cloud resource management solutions. Though ML models can learn from historical data and make informed decisions, they perform poorly with high-dimensional, non-linear, and temporally correlated data. However, DL is more favorable for learning complex representations and relationships from large datasets and is thus favorable for managing dynamic resources in cloud infrastructure [3, 4, and 7].

This work proposes a novel hybrid deep learning-based method that enhances cloud resource allocation through predicting workload demands and real-time allocation based on historical usage patterns. The proposed hybrid model leverages Convolutional Neural Networks (CNNs), Gated Recurrent Unit (GRU), and Long Short-Term Memory (LSTM) networks to benefit from their complementary strengths. They form a hybrid neural architecture that can carry out accurate time-series predictions and provide real-time decision-making support for resource provisioning. The research presents three primary contributions:

- ❖ Creation of more resilient, adaptive, and cost-effective cloud infrastructures that can handle growing and fluctuating workloads of modern applications using deep learning-based framework
- ❖ Building a hybrid deep learning model that integrates CNN, GRU, and LSTM models for on-the-fly cloud resource allocation.
- ❖ The proposed model's performance is analyzed on actual benchmark cloud workload traces to determine its effectiveness compared to traditional and machine learning-based methods.
- ❖ Measurement of performance parameters such as resource utilization, task response time, service level agreement (SLA) compliance, and system throughput to determine the method's real-world implications.

The remainder of the paper is structured as follows. Preliminaries are elaborated upon in Section 2. The Deep Learning Based Hybrid Framework is illustrated in Section 3. The experimental results are presented in Section 4. Lastly, Section 5 contains concluding notes.

Related work

The problem of efficient resource allocation in cloud computing has attracted considerable attention from both the research and industry communities. Various methods have been proposed to enhance resource allocation with the aim of minimizing costs, ensuring quality of service (quality of service), and optimizing resource utilization. Since Cloud Computing still hosts a variety of applications, such as real-time processing of data, artificial intelligence (AI), and the Internet of Things (IoT), it has become more difficult to guarantee optimal resource allocation [8-10]. Cloud infrastructure boasts virtuous resources like CPU, memory, storage, and network bandwidth, which are to be allocated dynamically according to fluctuations in workload [11]. However, the realization of efficient utilization of resources by reducing delays, power consumption, and running costs poses a significant challenge to cloud service providers (CSP). Classical resource allocation procedures, i.e., round-robin planning, first-first-first service (FCFS), and ranked planning, have been utilized extensively in the blame context. Classical approaches depend on pre-planned success in allocating resources between customers and programs. Though such methods provide simplicity and equity, they are not suitable for the dynamic and unpredictable nature of cloud labor. For example, round-robin planning distributes resources among functions, causing disabled and prospective bottlenecks regardless of their actual needs. Similarly, priority-based planning schedules tasks based on pre-defined rules, which are not capable of expressing the need for real-time fees. These static methods are struggling to adapt to fluctuations, resulting in over-control or over-provision of resources, increasing operational costs, and increasing service quality levels. To address such issues, several studies [8, 12-15] have identified the application of DL methods to make resources intelligent and dynamic. Learning patterns such as unsecured and observed learning, past charging habits, and resource usage patterns can facilitate future predictive and data-driven decisions. Cloud providers can predict resource demand, dynamically allocate resources, and optimize performance based on actual conditions with the help of the ML model. Deep learning, among all the leading MLs, has become extremely popular since it facilitates the retrieval of complex patterns and conditions from big data. The integration of machine learning (ML) and deep learning (DL) into cloud computing has uncovered significant improvements in efficiency, scalability, and cost savings, and research on AI-based ski solutions has been further advanced. For example, Wang et al. [16] provided a machine learning framework for resource allocation and used supervised learning to find commonalities in vast historical scenario data. The optimal or near-optimal solution of the most similar historical case is used to distribute radio resources for the current scenario based on extracted similarities. A simple beam allocation example in multi-user large MIMO systems proved that machine-learning-based resource allocation is better than conventional methods. Jayaprakash et al. [17] compared, analyzed, and classified cloud resource allocation clustering, optimization, and machine learning approaches for energy and performance efficiency. Optimization and clustering algorithms are used extensively in energy management since they can reduce energy consumption. They explain how multi-objective optimization approaches reduce energy consumption, avoid SLA violations, and improve service quality. We also cover how the firefly algorithm, whale optimization algorithm (WOA), particle swarm optimization (PSO), and genetic algorithm (GA) are used optimally in the field. They also indicate how deep neural networks (DNN), random forests, and support vector machines (SVM) are used to forecast cloud energy consumption and operate optimally. Anbarkhan [18] utilized regression models and neural networks to analyze historical data and quantify real-time measurements to deploy and configure cloud resources dynamically in a way that real-time demand is met efficiently. The algorithms can successfully forecast demand variations and shift resources dynamically. Simulations show the framework consumes 30% less resources than static approaches. The assignment method has also been adjusted to lower operational expenses by 25%. Sahar Badri et al. [19] utilized CNN to make pan minimization and throughput maximization. The data was encrypted using RSA. Ultimately, it is to be simulated and evaluated using a cloudlet simulator. The minimum reaction time of the 20s, the execution time of 0.43s, the energy consumption of 180 kWh, and the maximum utilization of 98% for job size 100 were attained. Song et al. [20] explored a cooperative computing paradigm with MEC servers, a remote MCC server, and mobile devices. One CNN-model-based MEC server handles computing tasks for all MDs. CNN-DQN optimizes downlink CRAN resource allocation and balances user quality of service and energy consumption. Qirui Li et al. [21] developed a multi-objective optimization system for cloud task scheduling and resource allocation using a united deep learning method that outperformed heterogeneous distributed deep learning, solving the cloud job MOO problem. Harshala Shingne et al. [22] introduced Firefly DRL, a heuristic deep learning-based scheduling method based on cloud-based Firefly and particle swarm optimization, which outperforms conventional scheduling and resource allocation algorithms. Junjie Cen et al. [23] proposed a resource allocation technique in a cloud-based collaborative computing environment that uses DRL, a Markov decision process (MDP), and hindsight experience repay to improve network interaction and reduce system delay in complex environments.

To the best of our knowledge, a significant quantity of research has been conducted in the past few years to comprehend, manage, and mitigate resource allocation in the cloud. The characterized by these difficulties is the subject of limited work. Consequently, the objective of this research is to propose a hybrid deep learning framework that incorporates CNNs for spatial feature extraction, GRNs for efficient recurrent modeling, and LSTM units for long-term sequence learning. This integrated model is designed to provide more accurate, adaptive, and context-aware resource allocation in cloud environments.

Deep Learning (DL) Methods

The development of deep learning has revolutionized artificial intelligence by allowing robots to learn difficult patterns and associations from data. Gated Recurrent Units (GRU), Convolutional Neural Networks (CNN), and Long Short-Term Memory (LSTM) are the three types of deep learning architecture used most frequently. Computer vision, natural language processing (NLP), and time-series forecasting are all areas that heavily rely on the application of these models because they can process all forms of data and tasks [24].

1. Convolutional Neural Networks (CNNs)

CNNs are specially crafted networks to process spatial and image data. They possess convolutional layers, which apply filters to select characteristic features such as shapes, textures, and edges. CNNs compare well with ordinary fully connected networks in that they reduce parameters using shared weights, which is appropriate for large-sized image recognition problems. CNNs find extensive applications in image classification, face detection, object detection, and medical imaging. They also enable applications like autonomous vehicles and video processing [25].

2. Long Short-Term Memory (LSTM) Networks

LSTM is a variation of Recurrent Neural Network (RNN) which has specifically been developed for application in sequence data such as text, speech, and time-series data. RNNs suffer from the vanishing gradient problem which detains RNNs from memorizing long-term relationships. LSTMs have universal applications in speech-to-text recognition, machine translation, chatbots, prediction of financial output, and sentiment analysis. As they have to maintain the relationships due to the long-term memory, they are highly capable to handle complex sequence tasks [26].

3. Gated Recurrent Unit (GRU)

GRUs are analogous to LSTMs but with fewer computations and simpler to understand. GRUs can be applied in natural language processing (NLP), time series prediction, and speech recognition. GRUs give the same kind of output as LSTMs but with fewer training periods and parameters. Thus, they are suitable for use in real-time when there is a problem of lightness of computation [26].

There are many applications of DL techniques. For image-based solutions, CNNs are suitable, whereas LSTMs and GRUs are suitable for sequence data. Whereas LSTMs are better suited for longer sequences, GRUs need less work in terms of processing and are faster compared to LSTMs. Deep learning, being based largely on such architectures, allows for artificial intelligence application spanning a larger timeframe.

Methodology

Figure 1 illustrates an effective two-phase training and validation of deep learning models like Convolutional Neural Networks (CNNs), Long Short-Term Memory networks (LSTMs), Gated Recurrent Units (GRUs), and ensemble methods. The models are applied on a unified dataset merging Google Cloud Jobs (GoCJ) data and Monte Carlo simulation outputs. The model addresses the actual problem of resource allocation and workload forecasting in virtualized cloud computing. GoCJ dataset provides the real-world demonstration of cloud infrastructure usage, where Monte Carlo simulations are employed to simulate uncertain system behavior and stochastic scenarios. Deep learning models learn pertinent patterns in the composite dataset during training so that they can accurately predict long-term system behavior and resource requirements. In the testing process, the performance of the models is rigorously tested to analyze their ability to predict workload patterns and optimally allocate resources in the cloud infrastructure. The combined assessment aids in identifying the most predictive accurate prediction models, operational performance, and usage of resources. The proposed framework is organized in the form of data generation, preprocessing, model training, and performance analysis, thus offering an effective solution to intelligent workload forecasting and resource assignment in cloud computing. The following is the step-by-step description of the proposed framework:

Phase 1: Data Generation

1. **Google Cloud Jobs Dataset & Monte Carlo Simulation**

A Monte Carlo simulation is embedded in the dataset, which is sourced from the Google Cloud Jobs (GoCJ) dataset, to make it more realistic and applicable for predictive modeling. The simulation is used to generate a synthetic but representative sales dataset by preserving the inherent uncertainty and variability usually found in actual cloud environments. This work-enhanced dataset is kept available for future experimentation and analysis, thus rigorous testing of resource allocation and forecasting models can be done. It has numerous individual computational tasks in it. Each of them is individually identified by specific job identifiers (83000, 83009, etc.). This facilitates precise tracking, referencing, and manipulation of individual tasks throughout the modeling process. The structured nature of the dataset facilitates the in-depth evaluation and construction of more accurate and uniform deep learning models for cloud workload prediction and optimization.

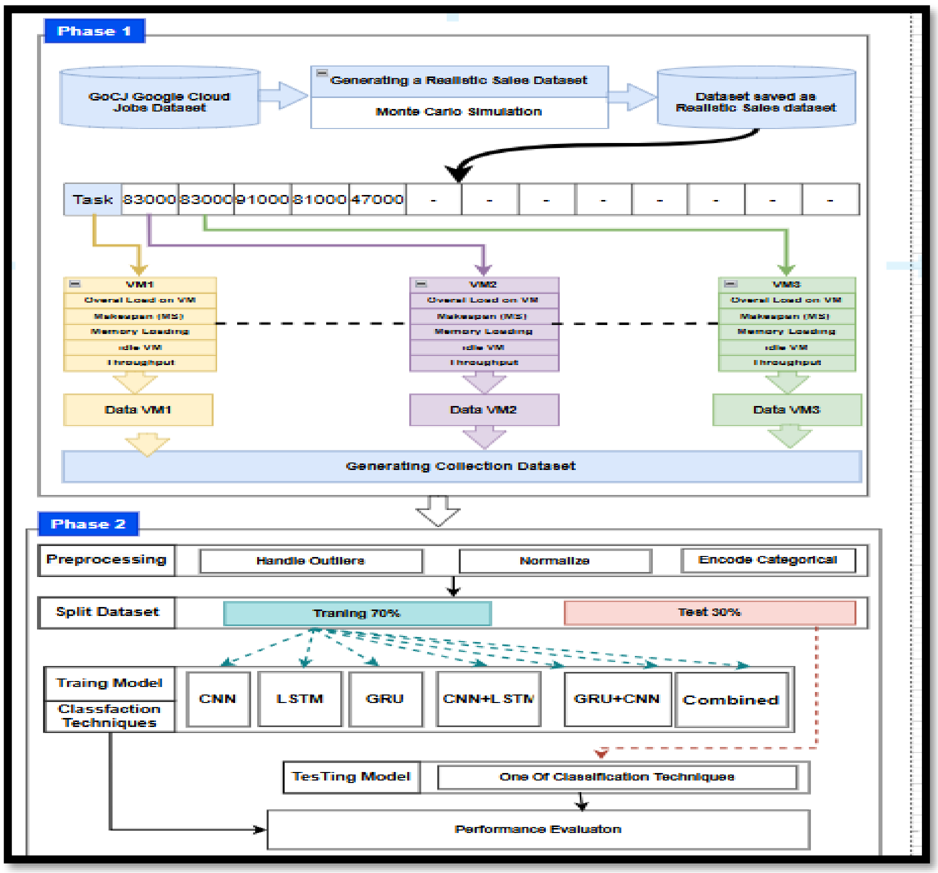


Figure 1. Proposed Framework by applying hybrid deep learning with cloud

2. Virtual Machine (VM) Workload Data Collection

The computational workloads are distributed systematically over several Virtual Machines (VMs), i.e., VM1, VM2, and VM3, each of which plays a pivotal role in executing and processing the corresponding workloads. To effectively monitor and evaluate the performance of each VM, a full set of performance metrics is continuously monitored while tasks are being executed. These quantities comprise the overall VM load, the overall computational capacity in use at any given time; makespan (MS), the aggregate time to complete independent or batched jobs; memory loading, which quantifies the memory consumed in processing; idle VM time, which captures instances of under-use or idling; and throughput, which captures the number of completed tasks within a specified time. The systematic collection of these measurements provides a detailed picture of the operational efficiency and resource needs associated with each VM. To facilitate intensive analysis, data for every virtual machine are kept isolated—VM1 Data, VM2 Data, and VM3 Data—so that individual performance can be analyzed. The data for all three VMs are combined into a single and complete Colocation Dataset. The merged dataset is a valuable resource for top-level analysis, i.e., task distribution optimization, performance bottleneck identification, and strategic resource planning. Such knowledge is critical to enabling overall efficiency and cost-effectiveness maximization of virtualized cloud infrastructure, whose performance and scalability at peak rely on the resourceful management.

Phase 2: Data Processing, Model Training and Hybrid models

1- Data Processing

Data preparation begins with Outlier identification and removal so that all variations at the end of the data are maintained to maintain the data set. This is to avoid training the model on unclean and unreliable data, which will not enhance performance and accuracy. Next, data is normalized, and it involves scaling the data set values to a standard degree. This is necessary to prevent functions with enormous numerical limits, which distort the model's learning process unevenly, from being initiated and all functions equal. Then the classified data is converted into numerical form, which can include non-conversion features during training.

2- Model Training

Processed datasets are then divided into three acres: a training kit (70%) used for model training, a verification kit (15%) for model optimization and hyperparameter setting, and a test set (15%) reserved to assess the final performance of the model. This partition ensures that the model is trained and valid on separate data, which allows for a fair evaluation of its generalization capacity. After preprocessing, many deep learning models are trained, including summoned neural networks (CNN), which are mainly used to remove spatial functions from data; Long short -term memory (LSTM) network, a type of recurrent nervous network (RNN) designed to handle sequence-based predicted features; And GAUDED recurrent devices (Gru), a custom version of LSTM, are specially designed to process lichened data by reducing certain limits for LSTM.

3- Hybrid Deep Learning Models

A- Hybrid LSTM + CNN Model for Sequential Data

The architecture described here mixes CNN and LSTM layers to take advantage of the power of both models in processing sequential data. CNNs are best at local spatial feature extraction, whereas LSTMs can model long-term temporal dependencies and thus the mix here is very much suitable for applications involving both spatial and sequential data, such as time-series prediction and sequence classification. The sequential data is input to the input layer, which is then input to the LSTM layer. The LSTM layer is configured with a kernel size of 1x256 and recurrent kernel size of 64x256 to learn temporal dependencies in the data. It has the ReLU activation function for the kernel and the sigmoid activation function for the recurrent kernel with return_sequences=True so that the LSTM outputs a sequence of values to be processed further. Following the LSTM, a dropout layer of 0.3 is employed to reduce overfitting by randomly dropping 30% of the neurons while training. Next, the model applies a 1D convolutional layer with kernel size 3x64x64 and utilizes 64 filters. Here, ReLU activation is employed as well to enable the model to learn sequence pattern features. The following MaxPooling1D layer helps down-sample by a factor of 2, lowering dimensions without losing the most important features. After the convolutional process, two fully connected (dense) layers are employed in the model. The first dense layer has 128 units using the ReLU activation, and the second is a dropout layer with a rate of 0.5 to prevent overfitting even more. Finally, the dense output layer with 5 units and softmax activation is employed to produce a probability distribution over the five target classes as the final prediction. Figure 2 Incorporating both CNNs and LSTMs, this hybrid model takes the best of both worlds by effectively modeling both spatial and temporal patterns in the data, which makes it suitable for use in speech recognition, time-series prediction, and other sequence-based applications.

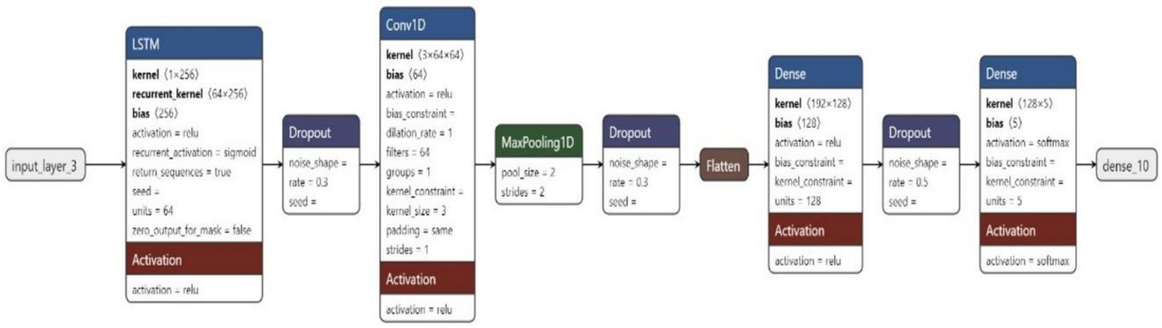


Figure 2. Represents a combination of an LSTM with CNN

B- Hybrid GRU + CNN Model for Sequential Data

The model presented combines GRU and CNN to process sequential data efficiently. GRU is a variant of LSTM and is highly efficient in processing temporal dependencies in sequential data, while CNNs are optimized to extract local features. Combining both components, the hybrid model becomes highly effective in operations such as time-series prediction, speech-to-text, and sequence classification. The architecture begins with an input layer accepting sequential data. Then comes the GRU layer where the kernel size is 1x192 and recurrent kernel size is 64x192, so that the GRU can capture temporal relationships within the sequence. The GRU uses ReLU as the activation function on the kernel and recurrent activation as sigmoid. The model is initialized with return_sequences=True so the GRU will return the entire sequence of data rather than the final state, that way all the information from each time step will pass through to the subsequent layers. A Dropout layer with dropout=0.3 is then employed to combat overfitting by randomly disabling 30% of the neurons when training. Next, a 1D Convolutional layer is introduced with kernel size 3x64x64 and 64 filters. In this case, the ReLU activation is employed, and the convolution layer is followed by MaxPooling1D. The max-pooling operation reduces the sequence dimensionality by half, with pool size 2and strides 2, so that the most important features are retained. The second Dropout layer at a rate of 0.3 is employed to again prevent overfitting. The Flatten layer is then applied to flatten the data into a 1D vector. The model then flows through two Dense layers. The first dense layer contains 128 units with the activation function being ReLU. The second dense layer has 5 units with softmax activation, giving a probability distribution over five target classes. Figure 4 present combining the GRU's temporal dependency learning ability with the CNN's local feature learning capability, this hybrid model is able to effectively learn both the sequence-level and local feature patterns of the data and is thus suited to a large variety of sequence-based prediction problems.

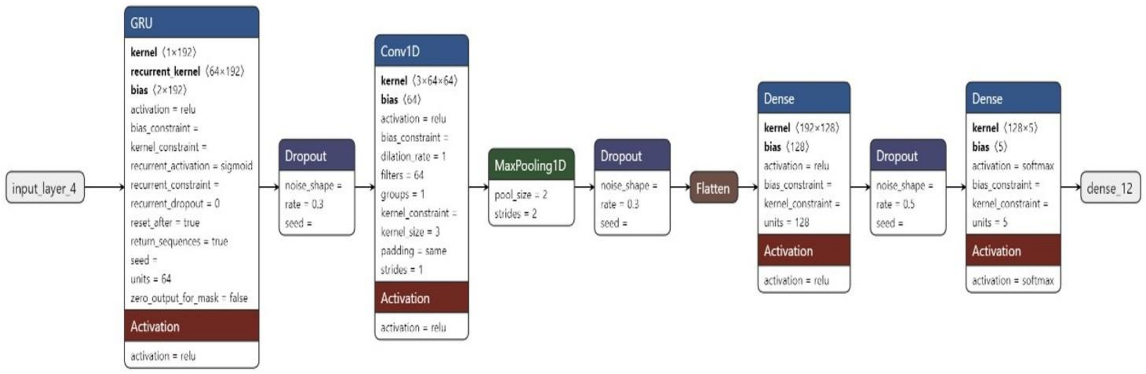


Figure 3. Represents combining a GRU with CNN

C- Hybrid LSTM + GRU + CNN Model for Sequential Data with Complex Dependencies

The architecture combines LSTM, GRU, and CNN to efficiently capture both temporal and spatial dependencies of sequential data. Each branch of the model reads the data in parallel, thus capturing a holistic view of the sequence by learning from different perspectives: temporal, recurrent, and spatial. The model begins with an input layer that accepts sequential data. That data is then fed through three parallel branches:

1. The LSTM branch learns long-term temporal relations through an LSTM layer with kernel size 1x256 and recurrent kernel size 128x256. The layer uses 64 units and a return_sequences=True argument to maintain the sequence output at each time step. A Dropout with rate 0.3 is applied to prevent overfitting. It is then passed into a Dense layer with 128 units and a ReLU activation.
2. The GRU branch uses a GRU layer with a kernel size of 1x192 and the recurrent kernel size is 64x192. With 64 units, the GRU is more effective in learning temporal dependencies with fewer parameters than LSTM. A Dropout layer with a rate of 0.3 is then followed by a Convolutional layer in order to extract more significant features.
3. The CNN branch employs a 1D Convolutional layer with kernel size 3x64x64 and 64 filters to capture local features of the sequence. A MaxPooling1D layer with pool size 2 and stride 2 is employed to downsample. A Dropout layer is also employed to avoid overfitting.

After being processed by the three branches, the outputs are concatenated to allow the model to fuse the temporal and spatial features that are learned by the LSTM, GRU, and CNN layers. The concatenated output is flattened using a Flatten layer to transform it into a 1D vector, and finally several Dense layers for classification. The final Dense layer gives 5 units with softmax activation to provide the classification probabilities. Figure 5 present the blend of LSTM + GRU + CNN is successful in tapping each component's own strengths, so it becomes adept at processing very demanding tasks dealing with both sequence-based and feature-based learning out of multi-dimensional data.

Journal of Information Systems Engineering and Management
2025, 10(41s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/
Research Article

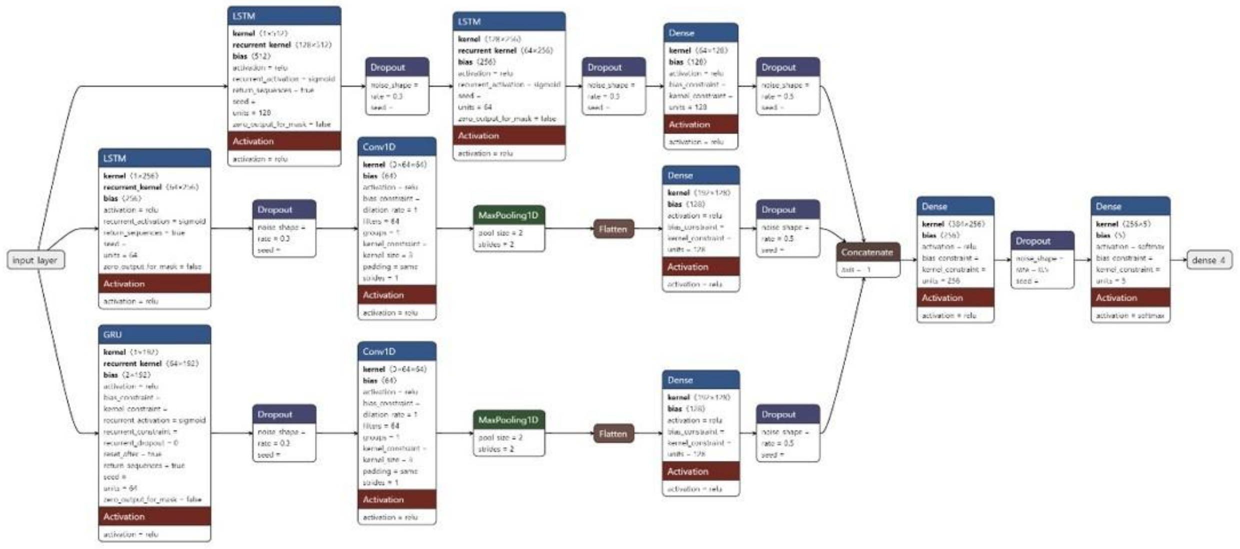


Figure 4. Combining three methods

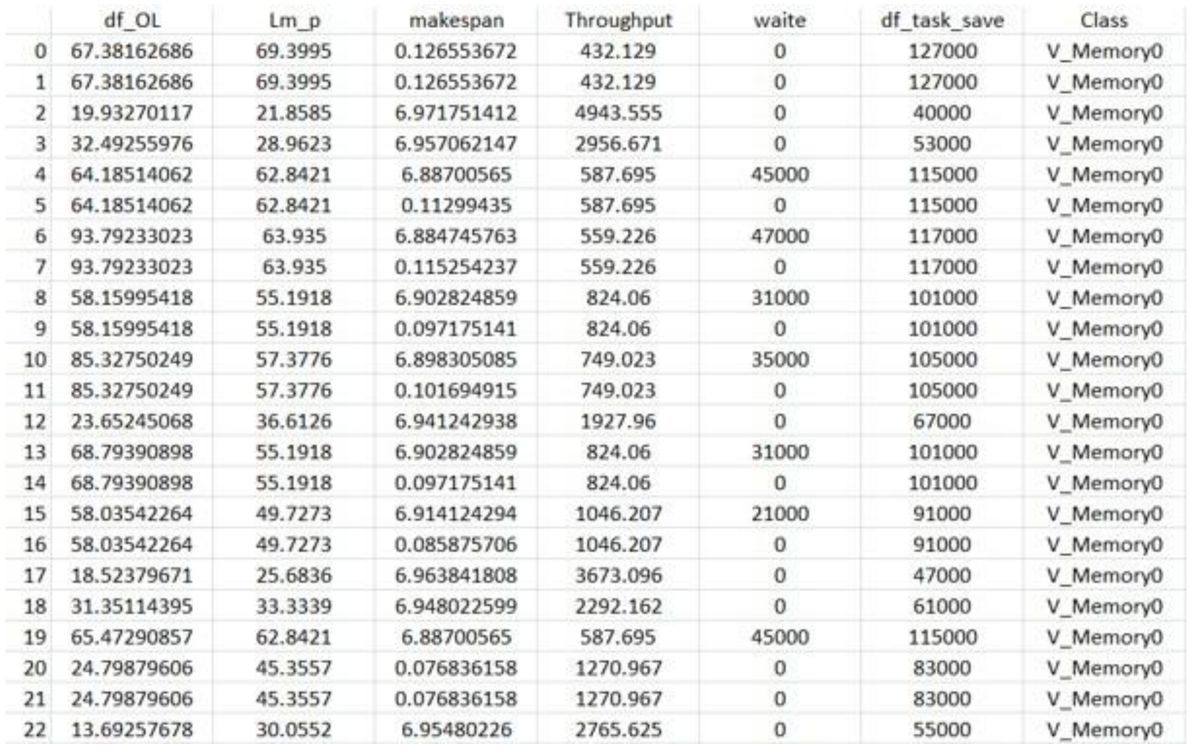
Testing & Performance Evaluation

To determine whether the models can produce accurate predictions, they go through a rigorous performance testing phase after training. To assess the model's accuracy and ascertain how well each model can categorize data and forecast outcomes based on the input features provided, classification techniques are used during this phase. A thorough analysis of each model's performance can be obtained by measuring a variety of performance metrics, including precision, recall, and F1 score. The best model with the highest accuracy and dependability, the largest capacity to generalize new, unknown data, and the best performance in real-world applications are selected based on the experiments.

1. Experimental Results

5.1 Datasets

GoCJ dataset is a treasure trove of cloud workload prediction and resource provisioning research. It has different sizes of jobs which may dynamically be generated based on pre-defined equations in an Excel sheet, as illustrated in Figure 1. The inclusion of Monte Carlo simulation introduces randomness variability, thereby rendering the dataset more realistic and the real-cloud workload scenario. This dataset is pivotal in facilitating efficient task allocation between virtual machines (VMs) to optimize resource utilization through performance measures such as VM choice, load balancing, makespan, throughput, waiting time, and system load. Through the assistance of intelligent task scheduling, the GoCJ dataset avoids VMs from becoming overloaded and aids significantly in the optimization of the entire cloud system's performance.



	df_OL	Lm_p	makespan	Throughput	waite	df_task_save	Class
0	67.38162686	69.3995	0.126553672	432.129	0	127000	V_Memory0
1	67.38162686	69.3995	0.126553672	432.129	0	127000	V_Memory0
2	19.93270117	21.8585	6.971751412	4943.555	0	40000	V_Memory0
3	32.49255976	28.9623	6.957062147	2956.671	0	53000	V_Memory0
4	64.18514062	62.8421	6.88700565	587.695	45000	115000	V_Memory0
5	64.18514062	62.8421	0.11299435	587.695	0	115000	V_Memory0
6	93.79233023	63.935	6.884745763	559.226	47000	117000	V_Memory0
7	93.79233023	63.935	0.115254237	559.226	0	117000	V_Memory0
8	58.15995418	55.1918	6.902824859	824.06	31000	101000	V_Memory0
9	58.15995418	55.1918	0.097175141	824.06	0	101000	V_Memory0
10	85.32750249	57.3776	6.898305085	749.023	35000	105000	V_Memory0
11	85.32750249	57.3776	0.101694915	749.023	0	105000	V_Memory0
12	23.65245068	36.6126	6.941242938	1927.96	0	67000	V_Memory0
13	68.79390898	55.1918	6.902824859	824.06	31000	101000	V_Memory0
14	68.79390898	55.1918	0.097175141	824.06	0	101000	V_Memory0
15	58.03542264	49.7273	6.914124294	1046.207	21000	91000	V_Memory0
16	58.03542264	49.7273	0.085875706	1046.207	0	91000	V_Memory0
17	18.52379671	25.6836	6.963841808	3673.096	0	47000	V_Memory0
18	31.35114395	33.3339	6.948022599	2292.162	0	61000	V_Memory0
19	65.47290857	62.8421	6.88700565	587.695	45000	115000	V_Memory0
20	24.79879606	45.3557	0.076836158	1270.967	0	83000	V_Memory0
21	24.79879606	45.3557	0.076836158	1270.967	0	83000	V_Memory0
22	13.69257678	30.0552	6.95480226	2765.625	0	55000	V_Memory0

Figure 5. GoCJ Excel worksheet generator.

Figure 2 is a large 3D scatter plot of the relationships among three key parameters: Job Size, Arrival Time, and Service Time, from a sample of 10,000 individual jobs. The X-axis of Figure 2 shows the job size in megabytes (MB), i.e., the volume of data each job is related to. The Y-axis represents the arrival time in seconds, or the actual time each job arrives in the system. The Z-axis is the processing time, or service time for each task on the cloud platform. The three-dimensional view gives a simple-to-picture model of task size, submission time, and processing time interaction within a cloud computing environment. The dispersal of data points along the 3D space reveals trends, clusters, and outliers, which help identify workload patterns, resource constraint, and inefficiency in scheduling. Moreover, the graph is a simple statistical modeling tool used by researchers to perform regression analysis, clustering, and correlation studies to improve task scheduling and resource allocation algorithms. By demonstrating the complex inter-relationships between such parameters, the 3D scatter plot is an essential tool to learning and optimizing cloud workload behavior and hence improving more efficient and intelligent ways of resource provisioning.

Journal of Information Systems Engineering and Management
2025, 10(41s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ **Research Article**

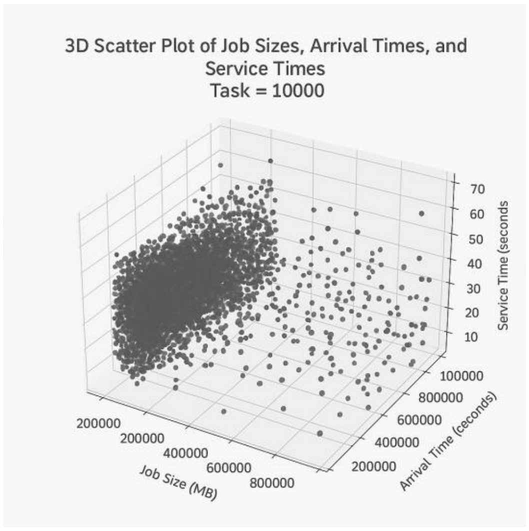


Figure 6. 3D scatter plot size, arrival times, and service times

Figure 3. shows a clear positive slope between memory loading and CPU load, which proves that CPU load is increased proportionally as memory usage is increased. The trend shows that there is a linear correlation between the level of memory utilized by the system and the load exerted on the CPU, which is typically indicative of the processing load of the system as a whole. Additionally, from the graph, one can observe that different models are irregularly distributed on the chart with some of them pulling more CPU at a specific memory usage while others show greater utilization of resources. This asymmetrical distribution of model performance is valuable information to developers since it illustrates the disparity in resources consumed by different models. With a correct analysis of such a trend, developers can more intuitively understand the trade-off between CPU usage and memory usage and make more suitable choices in selecting the most suitable model for the intended application. For computational performance or resource saving, this analysis allows developers to choose a model that gives the best trade-off between performance and efficiency according to what the system requires and can withstand.

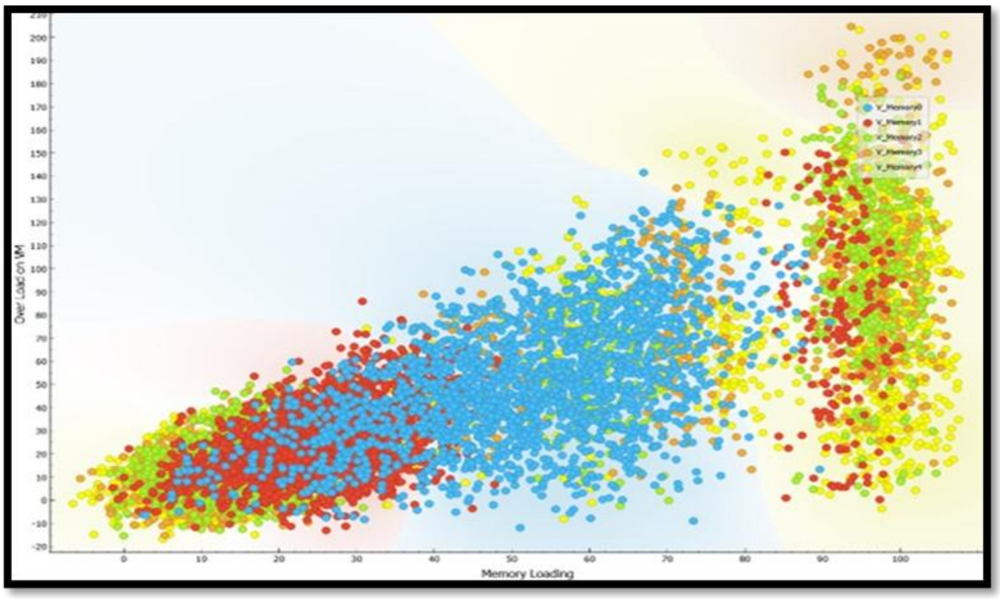


Figure 7. Distributed task relationship between over load on V.M and memory loading

Journal of Information Systems Engineering and Management
2025, 10(41s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/
Research Article

5.2 Results

The cloud resource provisioning based on deep learning was heavily benchmarked with a big and strong dataset that was generated from actual Google Cloud Jobs (GoCJ). The dataset was fine-tuned with utmost attention using Monte Carlo simulations that included probabilistic wobbles to simulate the natural randomness and fluidity of clouds. Before model training, the data was preprocessed heavily before it involved outlier handling, numerical attribute scaling, and one-hot encoding for categorical features. These operations were employed to ensure data consistency, improve model convergence, and provide more stable training.

Monte Carlo simulations were necessary to augment the dataset with realistic and varied patterns of workload. Probabilistic augmentation allowed the deep learning models to capture the uncertainty and variability like user requests and available capacity better than in the past by adding uncertainty and variability. These models learned on this dataset worked better in workload prediction, dynamic task scheduling, and intelligent provisioning of resources.

The data treated were divided into training, validation, and test sets in a common ratio of 70:15:15 to possess a good system of evaluation in order to test the model in order to generalize and predict. A set of current state-of-the-art deep architectures was implemented and tested, i.e., CNN for space feature extraction, LSTM and GRU for time sequence modeling, and ensemble approaches that correctly averaged the advantages of those models. A range of classification metrics has been used in measuring model performances: precision, recall, F1-score, and specificity. Measured exactly the proportion of well-classified positive instances, whereas recall (sensitivity) measured the model's ability to recover true positives from all existing actual positives. The F1-score, as the harmonic means of precision and recall, was especially relevant in class imbalance contexts. Specificity also supplemented these measures by measuring the precision in identifying negative examples, thus offering a full view of the classification capabilities of the model.

Among the models experimented with, the hybrid model that integrated CNN, LSTM, and GRU techniques achieved the highest overall performance. The hybrid model was able to effectively manage both spatial and temporal dependencies in the data, enabling it to detect fine workload patterns, capture short-term task bursts, and understand long-term demand trends. Such capabilities translated into highly accurate predictions and optimized resource allocation, preventing virtual machine overloading, reducing execution delays, and improving overall system efficiency (see Figures 8, 9, 10, 11).

In brief, the results indicate the enormous benefits of combining various deep learning architectures and probabilistic modeling approaches. This approach not only enhances the forecasting performance of cloud workloads but also greatly contributes to adaptive and intelligent cloud resource management systems.

Table 1. F1-value performance of three algorithms and hybrid methods

Datasets	CNN	GRU	LSTM	CNN+LSTM	GRU+ CNN	Combined
V_memory0	0.99	0.98	1	1	1	1
V_memory1	0.99	0.97	1	1	1	1
V_memory2	0.99	0.97	0.99	0.99	1	0.99
V_memory3	0.96	0.98	0.99	0.98	0.99	0.96
V_memory4	0.98	0.98	1	1	1	0.96

Journal of Information Systems Engineering and Management
2025, 10(41s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

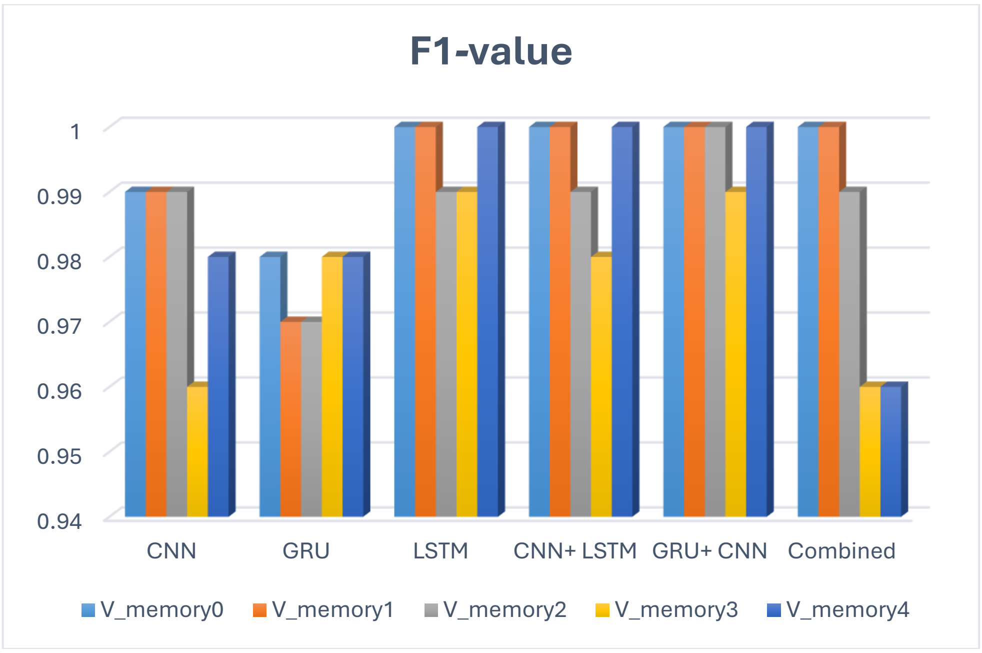


Figure 8. Three algorithms' F1-value performance as well as hybrid approaches

Table 2. Precision performance of three algorithms and hybrid methods

Datasets	CNN	GRU	LSTM	CNN+LSTM	GRU+ CNN	Combined
V_memory0	1	0.98	1	1	1	1
V_memory1	0.99	0.99	1	1	1	1
V_memory2	1	0.97	0.99	0.99	1	1
V_memory3	0.96	0.97	0.99	0.98	0.98	0.92
V_memory4	0.98	0.96	1	1	1	1

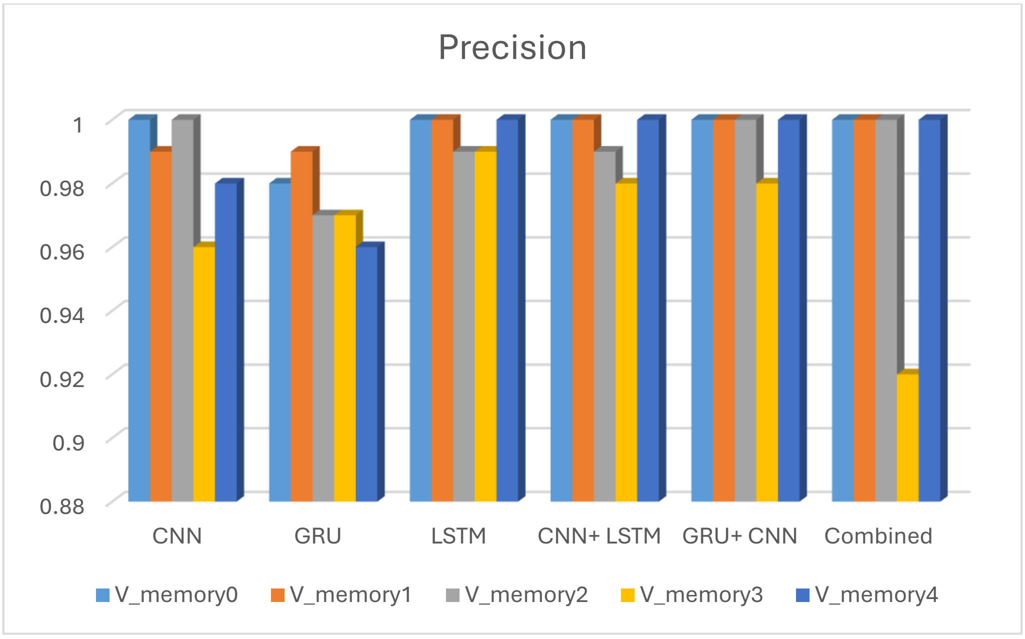


Figure 9. Three algorithms' precision performance as well as hybrid approaches

Table 3. Recall performance of three algorithms and hybrid methods

Datasets	CNN	GRU	LSTM	CNN+ LSTM	GRU+ CNN	Combined
V_memory0	0.99	0.98	1	1	1	1
V_memory1	1	0.99	1	1	1	1
V_memory2	0.97	0.97	1	0.99	0.99	0.99
V_memory3	0.97	0.96	0.99	0.99	1	1
V_memory4	0.99	0.95	0.99	0.99	1	0.92

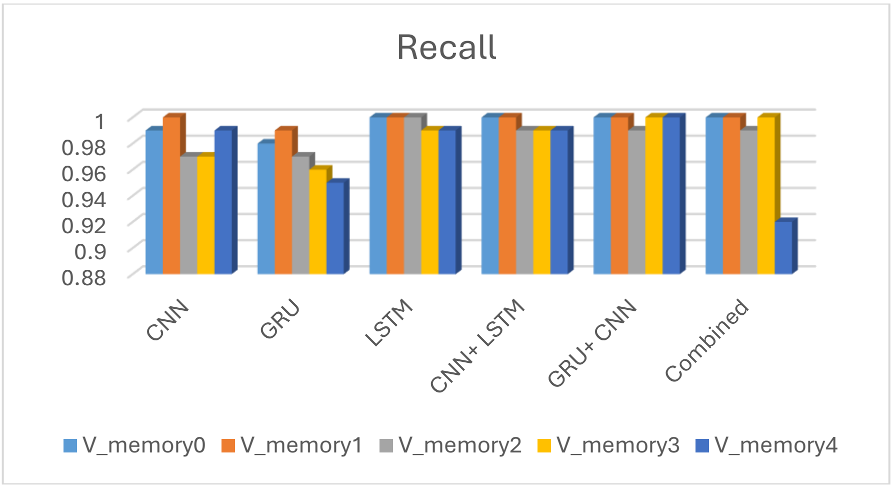


Figure 10. Three algorithms' recall performance as well as hybrid approaches

Table 4. Specificity performance of three algorithms and hybrid methods

Datasets	CNN	GRU	LSTM	CNN+ LSTM	GRU+ CNN	Combined
V_memory0	1	0.98	1	1	1	1
V_memory1	1	0.99	1	1	1	1
V_memory2	1	0.97	1	1	1	1
V_memory3	0.99	0.96	1	1	1	0.98
V_memory4	0.99	0.96	1	1	1	1

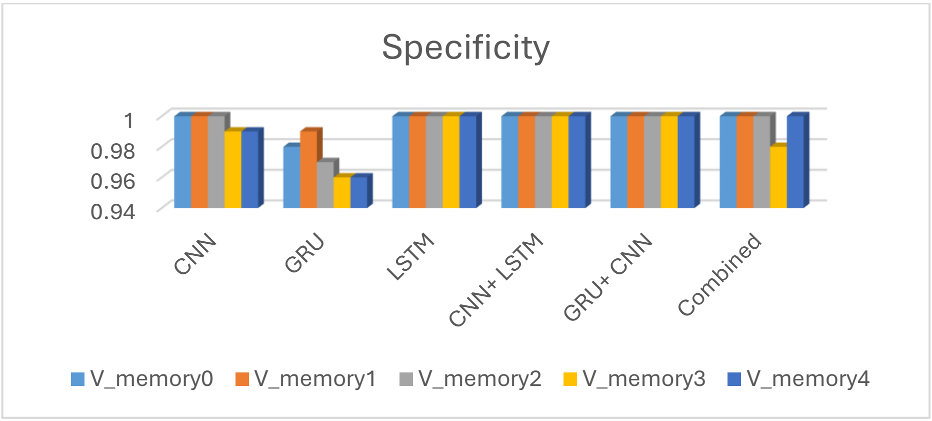


Figure 11. Three algorithms' specificity performance as well as hybrid approaches

Conclusion and Future works

The study evaluated the performance of the Cloud Resources allocation based on the modern DL models, namely CNN, LSTM, GRU, and Hybrid Architecture. Experimental results demonstrated that the hybrid model performed much better than individual architecture in predicted accuracy, stability, and efficient use of resources. According to deep learning strength, the targeted model effectively fought the resources allocated with UPS and rackets and fought for disability in the virtualized cloud system. Conclusions support the need for different deep teaching architectures to complement future performances, enable the charge balance, and avoid delay in the execution. In addition, the performance for calculations as accurate, recalls, F1 scores, and specificity provided significant insight into the skills of the hybrid model. Finally, this research emphasizes the transformation ability to further cloud resource management, improve planning efficiency, and energy saving, and strengthen the system. These results pave the way for further research that can further improve the intensive learning-controlled cloud distribution mechanisms to achieve greater efficiency, scalability, and cost-effectiveness in modern cloud computing systems.

References

[1] S. Jayaprakash, M. D. Nagarajan, R. P. d. Prado, S. Subramanian, and P. B. Divakarachari, "A systematic review of energy management strategies for resource allocation in the cloud: Clustering, optimization and machine learning," Energies, vol. 14, p. 5322, 2021.

[2] H. Zheng, K. Xu, M. Zhang, H. Tan, and H. Li, "Efficient resource allocation in cloud computing environments using AI-driven predictive analytics," Applied and Computational Engineering, vol. 82, pp. 17-23, 2024.

[3] M. Afrin, J. Jin, A. Rahman, A. Rahman, J. Wan, and E. Hossain, "Resource allocation and service provisioning in multi-agent cloud robotics: A comprehensive survey," IEEE Communications Surveys & Tutorials, vol. 23, pp. 842-870, 2021.

[4] A. Yousafzai, A. Gani, R. M. Noor, M. Sookhak, H. Talebian, M. Shiraz, et al., "Cloud resource allocation schemes: review, taxonomy, and opportunities," Knowledge and information systems, vol. 50, pp. 347-381, 2017.

[5] D. H. Hussein, G. Maqdid, and S. Askar, "Dynamic Resource Allocation in Cloud Networks Using Deep Learning: A review," The Indonesian Journal of Computer Science, vol. 14, 2025.

[6] T. Khan, W. Tian, G. Zhou, S. Ilager, M. Gong, and R. Buyya, "Machine learning (ML)-centric resource management in cloud computing: A review and future directions," Journal of Network and Computer Applications, vol. 204, p. 103405, 2022.

[7] X. Chen, L. Yang, Z. Chen, G. Min, X. Zheng, and C. Rong, "Resource allocation with workload-time windows for cloud-based software services: a deep reinforcement learning approach," IEEE Transactions on Cloud Computing, vol. 11, pp. 1871-1885, 2022.

[8] B. S. Ahamed, D. Poornima, A. S. Sheela, and S. Nivetha, "Allocation of Resources in the Cloud Conducted Efficiently Through the Use of Machine Learning," in 2023 International Conference on Emerging Research in Computational Science (ICERCS), 2023, pp. 1-6.

[9] S. Bhagat and P. Gupta, "Neural network and deep learning-based resource allocation model for multilayered cloud," in Machine Learning and Optimization Models for Optimization in Cloud, ed: Chapman and Hall/CRC, 2022, pp. 73-94.

[10] J. L. Berral, C. Wang, and A. Youssef, "{AI4DL}: Mining behaviors of deep learning workloads for resource management," in 12th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 20), 2020.

[11] Á. L. García, J. M. De Lucas, M. Antonacci, W. Zu Castell, M. David, M. Hardt, et al., "A cloud-based framework for machine learning workloads and applications," IEEE access, vol. 8, pp. 18681-18692, 2020.

[12] H. V. Dang, M. Tatipamula, and H. X. Nguyen, "Cloud-based digital twinning for structural health monitoring using deep learning," IEEE transactions on industrial informatics, vol. 18, pp. 3820-3830, 2021.
[13] B. Raviprasad, C. R. Mohan, G. N. R. Devi, R. Pugalenthi, L. Manikandan, and S. Ponnusamy, "Accuracy determination using deep learning technique in cloud-based IoT sensor environment," Measurement: Sensors, vol. 24, p. 100459, 2022.
[14] Archana, C. HP, Khushi, P. Nandini, Sivaraman, and P. Honnavalli, "Cloud-based network intrusion detection system using deep learning," in The 7th Annual International Conference on Arab Women in Computing in Conjunction with the 2nd Forum of Women in Research, 2021, pp. 1-6.
[15] M. B. Hassan, E. S. A. Ahmed, and R. A. Saeed, "Green machine learning approaches for cloud-based communications," in Green Machine Learning Protocols for Future Communication Networks, ed: CRC Press, 2023, pp. 129-160.
[16] J.-B. Wang, J. Wang, Y. Wu, J.-Y. Wang, H. Zhu, M. Lin, et al., "A machine learning framework for resource allocation assisted by cloud computing," IEEE Network, vol. 32, pp. 144-151, 2018.
[17] S. Jayaprakash, M. D. Nagarajan, R. P. d. Prado, S. Subramanian, and P. B. Divakarachari, "A systematic review of energy management strategies for resource allocation in the cloud: Clustering, optimization and machine learning," *Energies,* vol. 14, p. 5322, 2021.
[18] S. H. Anbarkhan, "Optimizing Cloud Resource Allocation with Machine Learning: Strategies for Efficient Computing," *Ingénierie des Systèmes d'Information,* vol. 30, 2025.
[19] Badri, Sahar, et al. "An efficient and secure model using adaptive optimal deep learning for task scheduling in cloud computing." Electronics 12.6 (2023): 1441.
[20] Song, Xia, Rong Chai, and Qianbin Chen. "Joint task offloading, CNN layer scheduling and resource allocation in cooperative computing system." Communications and Networking: 14th EAI International Conference, ChinaCom 2019, Shanghai, China, November 29–December 1, 2019, Proceedings, Part I 14. Springer International Publishing, 2020.
[21] Li, Qirui, et al. "UDL: a cloud task scheduling framework based on multiple deep neural networks." Journal of Cloud Computing 12.1 (2023): 114.
[22] Shingne, Harshala, and R. Shriram. "Mutated Deep Reinforcement Learning Scheduling in Cloud for Resource-Intensive IoT Systems." Wireless Personal Communications 132.3 (2023): 2143-2155.
[23] Cen, Junjie, and Yongbo Li. "Resource Allocation Strategy Using Deep Reinforcement Learning in Cloud-Edge Collaborative Computing Environment." Mobile Information Systems 2022 (2022).
[24] F. M. Shiri, T. Perumal, N. Mustapha, and R. Mohamed, "A comprehensive overview and comparative analysis on deep learning models: CNN, RNN, LSTM, GRU," arXiv preprint arXiv:2305.17473, 2023. Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: analysis, applications, and prospects," *IEEE transactions on neural networks and learning systems,* vol. 33, pp. 6999-7019, 2021.
[25] I. Uluocak and M. Bilgili, "Daily air temperature forecasting using LSTM-CNN and GRU-CNN models," Acta Geophysica, vol. 72, pp. 2107-2126, 2024.
[26] S. Nosouhian, F. Nosouhian, and A. K. Khoshouei, "A review of recurrent neural network architecture for sequence learning: Comparison between LSTM and GRU," 2021.