Hybrid Deep Learning Model for Task Scheduling in Cloud Computing

Kishan Chandra Chaurasia, Sandeep Kumar Singh
Department of CSE, SHEAT Group of Institutions, Babatpur, Varanasi (U.P), India

Abstract - The cloud computing landscape offers an abundance of resources, spanning network bandwidth, storage capacity, and processing power. However, achieving fair distribution among users and tasks poses a significant challenge in meeting diverse demands and priorities. Task scheduling in the cloud differs significantly from traditional techniques and warrants heightened attention due to its critical role in cloud services. The exceptional cost associated with cloud operations and resource utilization underscores the importance of effective task scheduling. In this research work hybrid deep learning model is proposed for the task scheduling in cloud computing. The proposed model is the combination of CNN and LSTM. The proposed model is compared with other models in terms of accuracy, precision and recall.

Keywords - Cloud Computing, Machine Learning, Task Scheduling, CNN, LSTM

I. INTRODUCTION

The rapid advancement of the Internet of Things (IoT) has ushered in a new phase of digitalization, facilitating seamless communication and data exchange among diverse smart devices and sensors. However, this swift expansion presents challenges, notably in managing the vast influx of task requests generated by these interconnected devices. The sheer volume of data processing tasks originating from IoT devices overwhelms traditional processing methods, necessitating a more robust solution [1]. Cloud computing technology offers a powerful solution to this challenge. The cloud computing landscape offers an abundance of resources, spanning network bandwidth, storage capacity, and processing power. However, achieving fair distribution among users and tasks poses a significant challenge in meeting diverse demands and priorities. In the dynamic and heterogeneous cloud environment, resource allocation becomes increasingly intricate. Task and user requirements exhibit substantial diversity and variability, with varying needs for different types and quantities of resources. Furthermore, user demands may fluctuate over time, complicating resource allocation efforts [2]. Moreover, the cloud operates within finite resource constraints, necessitating prudent resource utilization to optimize consumer satisfaction and service quality. Resource conflicts and competition are inevitable, requiring load balancing to avert performance degradation from resource overload [3]. Concurrent contention among multiple users or tasks for the same resources can lead to delays and inefficiencies. Hence, an efficient resource allocation scheduling technique is essential to resolve disputes and ensure equitable and effective resource utilization in the cloud environment. Task scheduling in the cloud differs significantly from traditional techniques and warrants heightened attention due to its critical role in cloud services [4]. The exceptional cost associated with cloud operations and resource utilization underscores the importance of effective task scheduling. Cloud resources, diverse and distinct from one another, necessitate careful consideration in task allocation. Task scheduling plays a pivotal role in enhancing the flexibility and reliability of cloud-based frameworks. The primary objective is to determine the optimal sequence for completing activities, ensuring the best outcome for clients within allocated timeframes [5]. In cloud computing, the sequencing and needs of tasks and their subtasks influence the gradual allocation of resources across various components such as networks, firewalls, and containers. Consequently, task scheduling in the cloud emerges as a considerable challenge, as there's no assurance that a predefined order will remain effective during project preparation. The dynamic nature of task scheduling is essential due to the simultaneous availability of multiple tasks and the uncertainty surrounding resource availability, task flow, and execution techniques [6]. Today, the predominant method for parallel processing on vast data sets within cloud computing platforms is Google's Map/Reduce programming approach. This method involves two key stages: Map and Reduce. Initially, the user's extensive data is divided into smaller sub-tasks during the Map stage. These tasks are then allocated to the cloud server's resource pool using suitable scheduling techniques [7]. Once the sub-tasks are completed by the computing resources, the results are aggregated through the Reduce stage and returned to the user. By employing this model, task execution efficiency is enhanced through task division and synchronous parallel computing, simplifying complex problems. The cloud computing platform typically consists of two layers, as illustrated in Figure 1 of the basic cloud computing platform model.

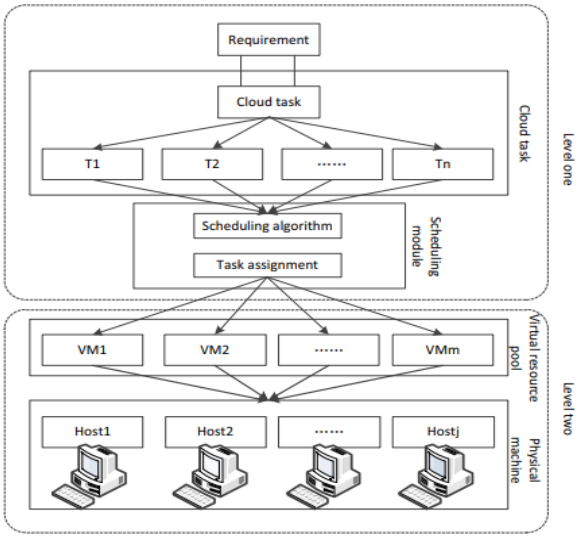

Figure 1: Cloud computing task scheduling model [8]

During the Map phase, tasks are categorized based on user-defined criteria. To effectively allocate n tasks to m computing resources within a resource pool, a suitable algorithm or strategy must be employed. Additionally, tasks must be executed on the resources in the order of their submission [9]. The mapping between computational resources and physical machines constitutes the second layer of scheduling. At this level, after subdividing tasks, the primary objective is to optimize the virtual resource pool provided to the cloud platform through task scheduling techniques. Cloud computing is tasked with providing multiple users with diverse services simultaneously. It's essential to consider each user's response time as well as the cost of delivering services to different users [10]. However, existing algorithms typically focus solely on the speed of task completion without considering that some computing resource processing may have a quick turnaround time but entail significant costs. Therefore, the task scheduling algorithm proposed in this study aims to reduce task completion time while also decreasing the service cost of core computer resources.

II. LITERATURE SURVEY

F. S. Alsubaei, et.al (2024) discussed that the major concern in cloud communication to schedule task that became complicated in the non-deterministic polynomial completeness (NP) of cloud systems [11]. A number of swarm intelligence (SI)-based approximation methods were developed. A dual machine learning (ML) method was suggested in which K-Means (KM) algorithm was deployed to optimize performance and to select cloud scheduling technology. The initial method was called Efficient Kmeans (Ekmeans) and latter one was Kmeans HEFT (KmeanH). Their major emphasis was on mitigating processing time and maximizing speed and efficacy for a given set of tasks. Diverse virtual machines (VMs) and task sizes were considered to compute this method. The results indicated the supremacy of suggested method over traditional methods.

B. Kruekaew, et.al (2022) introduced an independent method to schedule task in cloud computing known as Multi-objective task scheduling optimization-enabled Artificial Bee Colony Algorithm with a Q-learning algorithm (MOABCQ) [12]. This method was implemented for optimizing the way to schedule tasks and utilize resource, enhancing VM throughput, and balancing load among VMs on the basis of makespan, cost, and resource usage that were restrictions of parallel aspects. The CloudSim was applied to simulate the introduced method. According to experimentation, the introduced method was performed well to mitigate makespan, alleviate cost, diminish degree of imbalance, maximize throughput and enhance average resource usage as compared to standard techniques.

K. Li, et.al (2022) developed a method to schedule tasks on the basis of Particle Swarm Optimization (PSO) and Membrane Computing (MC) in cloud computing (CC) scenario [13]. First of all, a task scheduling (TS) model was deployed with time function (TF) and cost function (CF) as the target. After that, the PSO was considered to execute chaos operation in initializing population so that the diversity of rich understanding was improved. The sinusoidal function (SF)-based adaptive weight factor (AWF) method was implemented for avoiding the local optimum issue and MC was adopted in individual screening for enhancing the quality of individual solutions. In the end, the results exhibited that the developed method outperformed other techniques with respect to completion time and consumption cost while scheduling tasks.

P. Banerjee, et.al (2023) designed a Dynamic Heuristic Johnson Sequencing (DHJS) algorithm to schedule tasks in cloud computing (CC) based on a three-fold method [14]. At first, the associations were analyzed among jobs for creating a precedence graph (PG). At second, tasks were allocated to servers for converting PG into two-machine JS issue. At last, the designed algorithm was implemented for verifying the best order of jobs on every server so that the makespan was alleviated. The outcomes indicated that the designed algorithm was worked robustly to diminish makespan and utilize resources. Furthermore, this algorithm was proved scalable and effective to optimize the way of allocating resources and managing tasks in cloud systems.

F. Yao, et.al (2021) formulated a task duplication based scheduling algorithm (TDSA) for optimizing the makespan for budget-constrained workflows in cloud platforms [15]. Two methods, namely a dynamic sub-budget allocation (DSBA) and duplication-based task scheduling (DBTS) were deployed. The first one was aimed to recover idle budget of scheduled workflow tasks and redistribute remaining budget for enhancing completion time of unscheduled tasks. The latter one had deployed the idle slots on resources for selectively duplicating predecessors of tasks to enhance the completion time of tasks, and ensured their sub-budget restrictions. The experimental outcomes revealed that the formulated algorithm outperformed other methods, and enhanced makespan by 17.4% and resource usage by 31.6%.

K. Sharma, et.al (2023) projected a Periodic Min-Max (PMW) algorithm to schedule multi-robot task in cloud computing (CC) platform [16]. The Amazon web service (AWS) platform was employed to develop this algorithm which scheduled the multi-robot task. The task executed with the robots was taken as a single service concerning cloud platform and it became more effectual after the maximization of number of services with time. An analysis was performed on the projected algorithm with respect to time consumed to accomplish task and to balance load. The experimental outcomes depicted that the projected algorithm was more effective to enhance these factors up to 3-7% in comparison with the traditional methods.

S. Mangalampalli, et.al (2024) established a novel MOPDSWRL technique with the objective of scheduling complex workflows with more task dependencies [17]. First of all, this technique was aimed to compute priorities of all workflows according to their dependencies and evaluate priorities of VMs with regard to electricity cost at datacenters for mapping workflows onto precise VMs. The scheduler, in which Deep Q-Network (DQN) method comprised, was fed with these priorities for scheduling tasks relied on priorities of tasks and VMs in a dynamic way. The Workflowsim was applied to compute the established technique. The simulations confirmed the superiority of established technique over other methods and offered lower makespan and power consumption.

III. RESEARCH METHODOLOGY

Task scheduling in cloud computing is a critical aspect that aims to efficiently allocate resources to tasks or workloads in order to optimize performance, reduce costs, and enhance overall system reliability. Traditional approaches to task scheduling have been based on static heuristics or greedy algorithms, which may not be suitable for the dynamic and complex nature of cloud environments. The CNN model is proposed in this research work for the task scheduling in cloud computing. In Convolutional Neural Network, the input layer is consisted of a 3D (three dimensional) matrix of pixel intensities as feature map for diverse color channels. It can be defined as an induced multi-channel picture and the pixel of this image is called a particular feature. Each neuron is related to a small part of adjacent neurons from the accessible field. These maps are undergone from diverse conversions namely filtering and pooling. The initial operation focuses on convoluting a filter matrix with the values of an interested field of neurons and considering a nonlinear function for acquiring the final responses. The latter operation, such as average pooling, L2-pooling and LCM (local contrast normalization) aims to define the responses of a receptive field into one value with the objective of generating feature descriptions having more robustness. The architecture of CNN is illustrated in Figure 2, which allows efficient processing of image data.

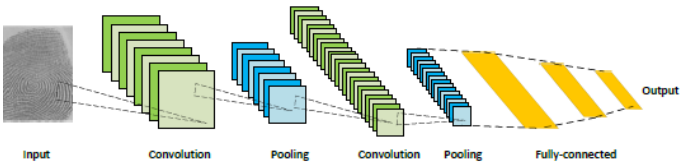

Figure 2: Architecture of convolutional neural network.

CNN is a kind of DL (Deep Learning) algorithm in the area of detecting the object and classifying an image. Similar to ML (Machine Learning) algorithms, this algorithm is capable of learning the metrics for which a training set is employed so that least empirical and structural risk is obtained such as the LF (loss function) is alleviated. Diverse operations support distinct LFs that implies the error occurred in predictive values and value having label of correct. DL is useful to stack the layers of metrics and establish an association amid them and AF (activation function). After that, the network becomes adaptable for the nonlinear functions which have complexity. Unlike the traditional NNs (neural networks), there are 3 layers for building the Convolutional Neural Network which are Conv (convolutional), Pooling and FC (fully connected) layers. The initial layer is employed for mapping an input image of multilayer into an output. Every image layer is considered as a channel. The mapping task is accomplished using a kernel for every channel. A convolution is exploited with the kernel with the purpose of separating every layer into small units (5×5) and generating a number from every unit. The convolution of every unit is expressed as:

$$E_{p,q} = \sum_{i,j} A_{ij} \cdot W_{ij}$$

In this, a square matrix based on (p, q) is represented with A, the correspondence components are defined through A_{ij} andW_{ij} and the output is obtained in the form of $E_{p,q}$. The SF (sigmoid function) or ReLU (rectified linear unit) function plays a role of an activation function to make the mapping non-linear. The given equation defines the ReLU function:

$$f(x) = \begin{cases} x, & x >= 0 \\ 0, & x < 0 \end{cases}$$

This layer aims to abstract the classic attributes from the origin picture. The lower layers are executed for retrieving the horizontal or vertical edges and the upper layers emphasize on integrating these attributes to corners, crosses, or other complicated attributes of the image. The second layer is placed after the Conv layers. It is undergone the image with a stride. This layer is utilized to execute a pooling operation on every unit window (2 × 2 or 3 × 3). The pooling operation often leads to select the value of a unit which may be maximum or average. An average pooling is defined as

$$P_{ave} = \frac{1}{r^2} \sum_{i,j=1}^{r} A_{i,j}$$

This layer is adopted for integrating the attributes whose extraction is done from the initial layers and selecting the considerable attributes of every window. The last layer helps in converting the 2-D or 3-D (two or three dimensional) input into 1-D (one-dimensional) array and combining every input linearly. This layer often deploys AF (Activation Function). It is useful to evaluate the way of working of every attribute on the final output.

```
1 model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (1, 10)                   40

 dense_1 (Dense)             (1, 5)                    55

=================================================================
Total params: 95 (380.00 Byte)
Trainable params: 95 (380.00 Byte)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 3: Proposed Model

IV. RESULT AND DISCUSSION

This work is related to task scheduling in cloud computing using machine learning techniques. The proposed method is the hybrid method of LSTM and CNN. The proposed method is compared with existing methods in terms of certain parameters. The results of proposed method are compared in terms of accuracy, precision and recall

4.1. Performance Analysis Parameters

Following are the various parameter used for the performance evaluation: -

1. **Accuracy:** Accuracy is perhaps the most intuitive metric and it is computed by dividing correctly predicted cases with overall cases in the dataset.

Formula: (True Positives + True Negatives) / (True Positives + True Negatives + False Positives + False Negatives).

2. Precision: Precision focuses on the correctness of positive predictions. It measures the ratio of correctly predicted positive instances to all instances predicted as positive.

Formula: True Positives / (True Positives + False Positives)

3. Recall (Sensitivity or True Positive Rate): Recall assesses the model's ability to correctly identify all positive cases in the dataset. It measures the ratio of correctly predicted positive instances to all actual positive instances.

4.2. Results

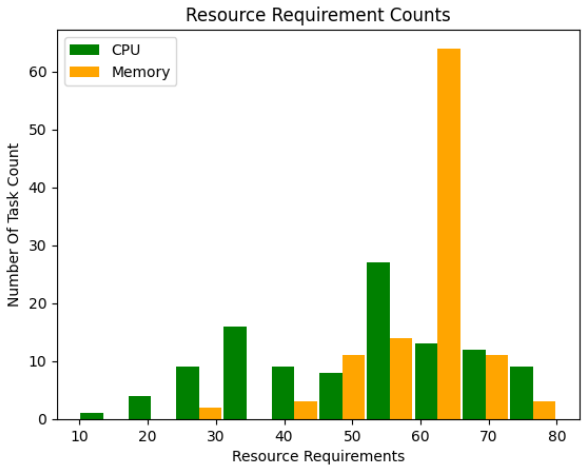

Figure 4: Resource requirement counts

As shown in figure 4, the resource requirements count is represented correspond to number of task count.

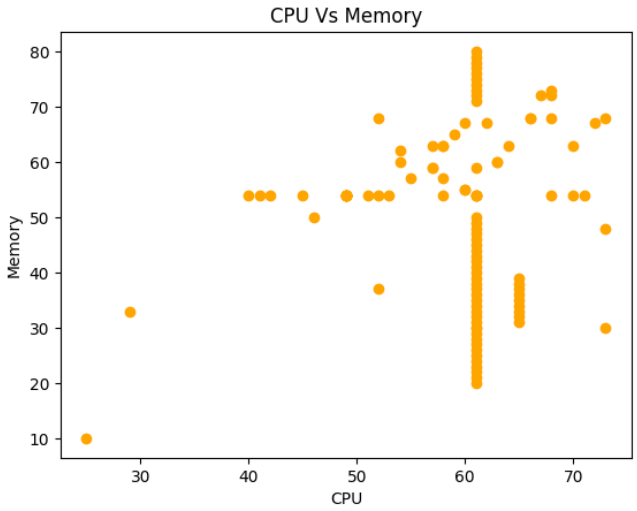

Figure 5: CPU and Memory Required

As shown in figure 5, the memory requirement for the CPU to execute task on the certain machines.

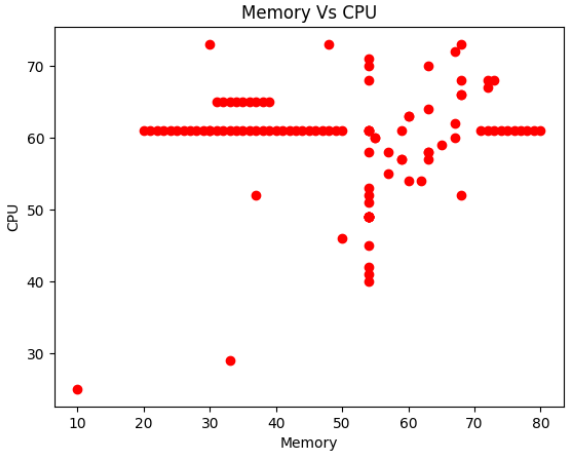

Figure 6: Memory Vs CPU

As shown in figure 6, the graph illustare the memory required for the CPU for the task execution.

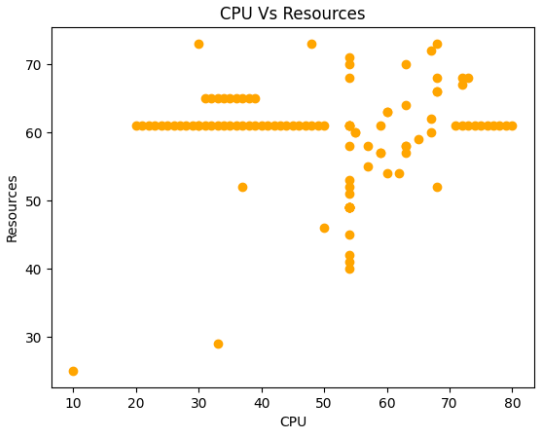

Figure 7: CPU vs Resources

As shown in figure 7, the number of resources required for the CPU for the task execution.

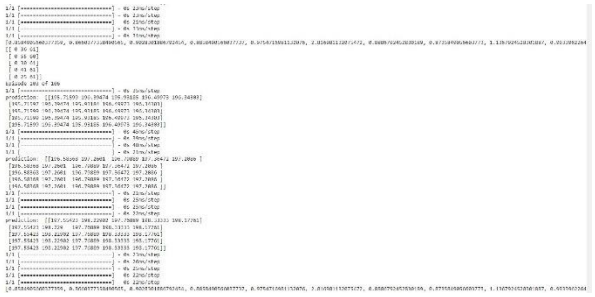

Figure 8: Execution of Proposed Model

As shown in figure 8, the proposed model is the deep learning model and execution of the model correspond to epoch values.

Table 1: Performance Analysis

Model	Accuracy	Precision	Recall
SVM	87.67 percent	85 percent	85 percent
KNN	82.34 percent	82 percent	82 percent
Proposed	92.34 percent	92 percent	92 percent

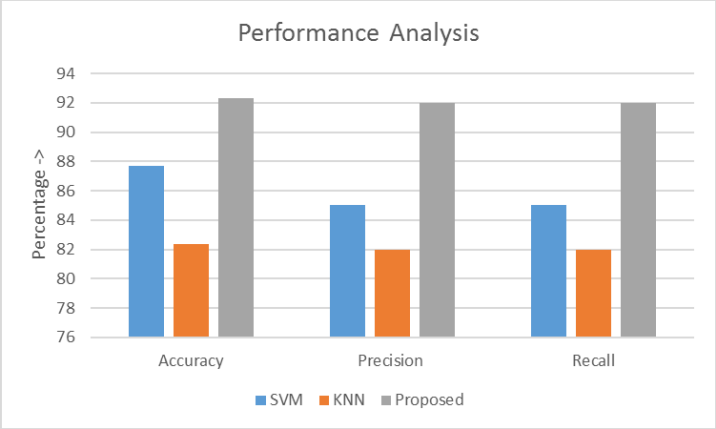

Figure 9: Performance Analysis

As shown in figure 9, the performance of proposed model is compared with SVM and KN for the task scheduling in cloud computing. The proposed model achieves accuracy, precision and recall above 90 percent which is quite high as compared to existing models.

V. CONCLUSION

The predominant method for parallel processing on vast data sets within cloud computing platforms is Google's Map/Reduce programming approach. This method involves two key stages: Map and Reduce. Initially, the user's extensive data is divided into smaller sub-tasks during the Map stage. These tasks are then allocated to the cloud server's resource pool using suitable scheduling techniques. Once the sub-tasks are completed by the computing resources, the results are aggregated through the Reduce stage and returned to the user. By employing this model, task execution efficiency is enhanced through task division and synchronous parallel computing, simplifying complex problems. The deep learning model is proposed in this research work for task scheduling in cloud computing. The proposed model is implemented in python and results are compared with SVM, KNN models. It is analysed that proposed model achieves accuracy, precision and recall above 90 percent which is approx. 8 percent higher than other models like SVM and KNN.

VI. REFERENCES

[1] M. Sardaraz and M. Tahir, "A Hybrid Algorithm for Scheduling Scientific Workflows in Cloud Computing," in IEEE Access, vol. 7, pp. 186137-186146, 2019

[2] Y. -H. Jia et al., "An Intelligent Cloud Workflow Scheduling System with Time Estimation and Adaptive Ant Colony Optimization," in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 51, no. 1, pp. 634-649, Jan. 2021

[3] H. Ali, M. S. Qureshi, M. B. Qureshi, A. A. Khan, M. Zakarya and M. Fayaz, "An Energy and Performance Aware Scheduler for Real-Time Tasks in Cloud Datacentres," in IEEE Access, vol. 8, pp. 161288-161303, 2020,

[4] M. Soualhia, F. Khomh and S. Tahar, "A Dynamic and Failure-Aware Task Scheduling Framework for Hadoop," in IEEE Transactions on Cloud Computing, vol. 8, no. 2, pp. 553-569, 1 April-June 2020

[5] G. Fan, L. Chen, H. Yu and D. Liu, "Modeling and Analyzing Dynamic Fault-Tolerant Strategy for Deadline Constrained Task Scheduling in Cloud Computing," in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 50, no. 4, pp. 1260-1274, April 2020

[6] H. R. Faragardi, M. R. Saleh Sedghpour, S. Fazliahmadi, T. Fahringer and N. Rasouli, "GRP-HEFT: A Budget-Constrained Resource Provisioning Scheme for Workflow Scheduling in IaaS Clouds," in IEEE Transactions on Parallel and Distributed Systems, vol. 31, no. 6, pp. 1239-1254, 1 June 2020

[7] K. Pradeep and T. P. Jacob, "Comparative analysis of scheduling and load balancing algorithms in cloud environment," 2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2016, pp. 526-531

[8] S. Gao, Y. Li and H. Huang, "A Multi-Objective Task Scheduling Method based on ACO in Cloud Environment," 2020 IEEE 6th International Conference on Computer and Communications (ICCC), 2020, pp. 1554-1559

[9] N. Chaudhary and M. Kalra, "An improved Harmony Search algorithm with group technology model for scheduling workflows in cloud environment," 2017 4th IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics (UPCON), 2017, pp. 73-77

[10] L. Kapoor et al., "SLOPE: A Self Learning Optimization and Prediction Ensembler for Task Scheduling," 2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2018, pp. 1-7

[11] F. S. Alsubaei, A. Y. Hamed and M. K. Elnahary, “Machine learning approach to optimal task scheduling in

cloud communication", Alexandria Engineering Journal, vol. 89, pp. 1-30, 20 January 2024

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

[13] K. Li, L. Jia and X. Shi, "Research on Cloud Computing Task Scheduling Based on PSOMC," in Journal of Web Engineering, vol. 21, no. 6, pp. 1749-1766, September 2022

[14] P. Banerjee et al., "MTD-DHJS: Makespan-Optimized Task Scheduling Algorithm for Cloud Computing With Dynamic Computational Time Prediction," in IEEE Access, vol. 11, pp. 105578-105618, 2023

[15] F. Yao, C. Pu and Z. Zhang, "Task Duplication-Based Scheduling Algorithm for Budget-Constrained Workflows in Cloud Computing," in IEEE Access, vol. 9, pp. 37262-37272, 2021

[16] K. Sharma et al., "Cloud Based Multi-Robot Task Scheduling Using PMW Algorithm," in IEEE Access, vol. 11, pp. 146003-146013, 2023

[17] S. Mangalampalli et al., "Multi Objective Prioritized Workflow Scheduling Using Deep Reinforcement Based Learning in Cloud Computing," in IEEE Access, vol. 12, pp. 5373-5392, 2024