International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 6, November-December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806025

An Intelligent Predictive GPU Scheduling Framework for Deep Learning Workloads in Large-Scale Cloud Environments

Dr.R.Sugumar

Professor, Department of Computer Science & Engineering, SIMATS Engineering, Saveetha Institute of Medical and Technical Sciences (SIMATS), Chennai, India

ABSTRACT: The paper is about the problems of optimally scheduling the resources of GPUs when it comes to large-scale deep learning workloads within the cloud infrastructure. Due to the fact that the field of deep learning requires a noticeably large amount of computational power, GPUs play a crucial role in hastening these operations. Nevertheless, the management of the GPU resources in the cloud is still complicated because of the dynamics of the workload and resources allocation that requires efficient implementation. This paper introduces a predictive graphic card scheduling system, which takes advantage of machine learning to predict resource demands through the nature of workload. The architecture combines workload prediction, the control of the GPU resources, and optimization algorithms to allocate resources in advance that the deep learning tasks get the required amount of GIS in time and at the same time, the active time is minimized as well as the contention of the resources.

The framework relies on the historical performance data as well as analysis of workload behavior to forecast future demands which are then adjusted in terms of scheduling strategies. The strategy not only maximizes the use of the GPU but also leads to the overall performance of cloud-based systems since it minimizes the waste of resources and shortens the duration of tasks. To gauge the effectiveness of the suggested framework, the article logs the experiments carried on an extensive scale which shows the superiority of the suggested framework in comparison to the traditional ones in a large-scale cloud setting.

KEYWORDS: GPU Scheduling, Deep Learning, Cloud Computing, Resource Management, Predictive Framework, Machine Learning, Optimization Algorithms.

I. INTRODUCTION

Deep learning has emerged as a paradigm-shifting technology in the past few years in various areas, such as artificial intelligence, natural language processing, computer vision, and autonomous systems. Their complex architecture and huge computing demands have exceeded the capabilities of traditional computing systems, requiring Graphics Processing Units (GPUs) to run deep learning models with ease during training and inference, among various tasks. GPUs are also parallel processors designed with high-dimensional, matrix-based computations, which are prominent to deep learning. Consequently, the demand of GPU resources has greatly increased, especially in large-scale cloud computing systems where deep learning models need a lot of computational resources, memory, and storage.

Cloud infrastructures of large scale provide the most convenient environment to execute the workloads of the deep learning with its flexibility, scalability, and the possibility to deploy the resources on-demand. Cloud computing services such as Amazon web services (AWS), Microsoft Azure, and Google cloud are offering a variety of instances with GPUs that enable users to lease the resources of the CPU to handle their particular workloads. Nevertheless, it is difficult to handle these resources effectively in a cloud network. Cloud infrastructures are by nature dynamic in nature as they share resources amongst several users and applications. Furthermore, deep learning tasks can be highly unpredictable with regards to resource usage where some models could need enormous amounts of GPU resources and others may be lower. This unpredictability along with the complexity of cloud environments makes the process of scheduling the experience of the available GPU resources an important activity both to cloud service providers and customers.

Regardless of the developments in cloud computing, scheduling of the GPU resources is still a big challenge. Common GPU scheduling approaches, including First-Come-First-Serve (FCFS), Shortest Job Next (SJN), and Round Robin, are aimed at assigning resources and do not imply taking into consideration the needs of deep learning workloads. Such techniques usually result in poor usage of the GPU, contention and waiting time in users in some instances. In addition, non-predictive feature of these traditional methods cause inefficient scheduling where resources might be over-allocated or under-allocated and this creates inefficiency in running the system at high costs.

In order to help overcome these problems, it is necessary to design a smart and predictive GPU scheduling system. Predictive scheduling is the method of predicting the future resource needs of deep learning workloads and tasks utilizing their past behaviour and their workload properties. With the help of machine learning models and other predictive algorithms, such a framework can also predict what deep learning models require and can allocate its resources on a schedule, enhancing the utilization of its resources and system performance.

Cloud environments are characterized by very dynamic workloads of deep learning where the resource requirements cannot be predicted. Such workloads are very heavy to be computed on a standard processor and usually they make use of GPUs in order to carry out parallel processing operations effectively. Nonetheless, the differences in the usage of a GPU combined with the communal character of cloud-based resources cause severe problems in the management and scheduling of the GPU resources. The conventional time scheduling strategies fail to consider the high-level requirements of deep-learning applications, which frequently lead to the waste of GPUs and poor system resource utilization. Consequently, an intelligent time management system that is capable of forecasting the resource demand of deep learning jobs in real-time is urgently needed, as it would guarantee optimal and efficient resource utilization of the GPUs in large-scale cloud-based systems.

The increasing trend in the number of requests to the GPUs in the cloud, along with the complexity of scheduling the deep learning workloads, has led to the study of more intelligent and efficient scheduling mechanisms. An effective GPU scheduling model can improve the performance of deep learning tasks, minimize the resource wastage and optimize the system throughput. As machine learning approaches grow, there exist a high likelihood of creating predictive models that are able to predict the workload requirements and make efficient resource allocation. This may result in better use of resources, shorter waiting periods and less expenses to both the cloud service providers and consumers.

Additionally, the necessity of the more effective management of the resources is even more important as deep learning models are still being developed in the complex way. An intelligent scheduling framework can reduce the effects of resource contention and bottlenecks by preemptively making predictions regarding the needs of the resources, and scheduling workload demands to the GPUs to ensure that cloud-based deep learning applications operate optimally.
The key goal of this study is to create a smart predictive framework of scheduling a GPU that will be adapted to the demands of workloads that are specific to deep learning in large-scale clouds. The given framework will attempt to forecast the future needs of the deep learning tasks in terms of the resource usage of the GPUs, depending on historical activity and workload trends. Using the methods of machine learning and workload prediction models, the framework will be able to project resource requirements and assign GPU resources in advance, which guarantees effective consumption of cloud resources.

Besides working on the framework, this study will also analyse the effectiveness of the recommended scheduling mechanism using large-scale experiments. The evaluation will entail the comparison of the proposed framework to the conventional scheduling techniques concerning the use of GU and the time taken to complete the tasks and the overall performance of the system. These experiments will give us an idea about the effectiveness of predictive scheduling on large-scale cloud environments, as well as how it can be used to enhance the efficiency of deep learning workloads.
The smart framework of scheduling GPUs presented includes the use of machine learning to estimate the demands of deep learning tasks in their resources. The framework includes various hardware elements: workload prediction, management of the work with the help of a GPU and optimization algorithms. All these elements are very crucial in the provision of efficient and effective allocation of GPU resources.

1. **Workload Prediction**: The problem with deep learning workload is that its resource demands are unpredictable, which is one of the primary challenges of the process of scheduling these workloads. To deal with this, the suggested framework relies on machine learning models to forecast the future resource requirements of a deep learning task. The training of such models takes place with the help of historical performance data, such as the data on the computational needs of the workload, the memory consumption, and the time the workload takes to be

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 6, November-December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806025

processed. The prediction models will be able to anticipate future requirement of similar tasks by examining the past workload history.

2. **GPU Resource Management**: The framework allocates the use of the GPU resources in the most efficient manner based on the estimated workload demands. It takes into account different aspects including the quantity of accessible GPUs, their memory capacity and the processing power of each workload. The resource management component makes sure that the correct quantity of GPU resource is given to each task and as a result, there is a reduction in idle time, and there is no resource contention.
3. **Optimization Algorithms**: The framework includes optimization algorithms which only further optimize the allocation of resources. These algorithms compare various scheduling strategies and choose the most effective one depending on the workload requirements that are predicted. The optimization algorithms distribute the resources of the GPUs in the most efficient manner available by considering the priority of the tasks, the availability of the resources, and the constraint of the system.

This research makes several important contributions to the field of GPU scheduling in cloud environments:

1. **Development of a Predictive Scheduling Framework**: The main input of this work is the creation of a smart predictive customsized GPU scheduling system of deep learning tasks. The framework uses machine learning and forecasts the future resource requirement of deep learning tasks and proactively allocates the resources in GPUs to enhance the overall performance of the system.
2. **Evaluation of Scheduling Strategies**: The effectiveness of the proposed framework is also evaluated in the research by carrying out large experiments. The findings indicate the benefits of predictive scheduling in comparison with traditional scheduling, and evidences the optimization of the use of the framework in terms of the resources of the graphics card usage and decreasing the waiting period of users.
3. **Enhanced Cloud Resource Management**: The suggested system provides a more effective way of controlling the GU resources on mass clouds. The framework optimizes the allocation of GPUs, minimizing wastage of resources, enhancing throughput of the system and reducing the cost of operation by cloud service providers.

II. RELATED WORK

The optimization of the scheduling of the tasks of deep learning on the GPUs has been investigated in a few recent works that aim to solve such issues as the problem of resource contention, the heterogeneity, and the scalability of the experiments in the cloud environment. These attempts are to enhance the allocation of the resource of the GPUs and job scheduling aimed at the growing complexity of the AI loads.

Weng et al. introduced an elaborate workload analysis and scheduling methodology in the environment of massively scalable heterogeneous GPU cluster in MLaaS in the Wild: Workload Analysis and Scheduling in Large-Scale Heterogeneous GPU Clusters [1]. They found difficulties in handling diverse AI loads, including different resource needs and time needed to complete a task, and offered a scalable scheduling model to improve the use of the GPU resources in the most efficient manner. The paper has revealed the significance of workload behavior cognition in order to make improved scheduling choices in service environments involving machine learning (MLaaS) applications.

In their article about Scheduling Ring-All-Reduce Learning Jobs in Multi-Tenant GPU Clusters with Communication Contention [2], the authors Yu et al. discussed the issue of distributing deep learning jobs in multi-tenant GPU clusters through the use of the ring-all-reduce method. They also designed a scheduling algorithm where communication contention is considered and as a result, they enhanced the performance of multi-tenant GPU cluster through reducing the communication delays incurred during model training. This paper has demonstrated that patterns of communication can have serious implications to the scheduling of resources particularly in multi-tenant environments.

In his recent study Kathiresan, Cost-Efficient and Scalable GPU Scheduling Strategies in Multi-Tenant Cloud Environments to AI Workloads [3], he suggested a number of scheduling strategies to balance the costs and scalability of the allocation of the use of the GPU resource in multi-tenant cloud computing. The study highlights the necessity of cost-effective scheduling algorithms that can effectively handle AI loads and pay minimal on the operation of cloud data centers.
In Scheduling with Untrusted Predictions [4], Bampis et al. focused on the problem of scheduling in an environment where there is no reliable prediction of the workload. They introduced a new strategy that addresses the vulnerabilities of unreliable predictions through the use of trust measures and adaptive scheduling. It is a robust and reliable way of scheduling because it is applicable when the workload demands are uncertain.

Yu, Wu, Ji, and Liu proposed a mathematical model that defines the scheduling of deep learning jobs in A Sum-Of-Ratios Multi-Dimensional Knapsack Decomposition to DNN Resource Scheduling [5]. Their strategy focuses on maximizing resource allocation taking into account various dimensions like memory, computation and communication and dramatically increasing resource efficiency and reducing job completion time.

In their article Megatron-LM: Training Multi-Billion Parameter Language Models with GPU Model Parallelism [6], Shoeybi et al. discussed the problem of the scalability of large language models with the help of the memory and computational resources, namely, the issue of GPU model parallelism. They have helped design efficient parallel training methods, a very important component of contemporary deep learning structures.

In their work DAPPLE: A Pipelined Data Parallel Approach to Training Large Models [7], the authors presented a pipelined data parallelism model to achieve more efficient training of a large deep learning model. This is a better technique in that the throughput and training time is improved because it overlaps computation and communication, which is essential when handling large-scale models on clouds.

In Memory-Efficient Pipeline-Parallel DNN Training [8], Narayanan et al. suggest a novel pipeline-parallel training of DNN that is highly efficient with regards to memory usage, and at the same time, it is highly efficient with regards to throughput during distributed training. Specifically, they are relevant to work with resource-constrained settings, in which large models should be trained using multiple GPUs.

The article by Karmaker et al. gave an in-depth description of AutoML and its challenges in AutoML to Date and Beyond: Challenges and Opportunities [9], which illuminates the increasing demand of automated machine learning systems that are capable of distributing resources and optimizing models without demanding manual tuning. Their results are consistent with the requirement of smart scheduling systems that would be able to respond to the changing workload requirements.

Tarnawski et al. proposed Piper: Multidimensional Planner to DNN Parallelization [10], which is a multidimensional version of the parallelization of DNN models. The system optimally divides work to utilize the maximum of the GPUs and the least overhead which is a new contribution to the effective scheduling of parallel tasks in a large-scale cloud infrastructure.

In Communications Contention Aware Multiple Deep Learning Job Scheduling [11], Wang et al. solved the problem of communication contention in multi-GPUs. Their strategy focuses on jobs according to the needs of communication aspect such that the jobs with the high level of communication do not adversely affect the performance of other jobs in a multi-tenant setting.

Peng et al. in Optimus: An Efficient Dynamic Resource Scheduler to Deep Learning Clusters [12] developed a dynamic framework of scheduling that is adaptable to changes in work load. Optimus enhances resource optimization based on the computational and communication requirements of the deep learning jobs, which is important to large-scale cloud-based deep learning systems.

Finally, Faisal et al in When Will My ML Job Finish? Toward Providing Completion Time Estimates by Predictability-Centric Scheduling [13] solved the problem of the estimation of job completion times in GPU clusters. They work on the basis of giving accurate estimates in job scheduling to optimize the use of resources and make the scheduling of AI workloads in dynamic environments.

All of these papers contribute to the knowledge of scheduling deep learning workloads on GPUs in the cloud setting, suggesting advanced scheduling methods and strategies to address the requirements of AI tools in increasing numbers. They emphasize the importance of having smarter, more flexible, and less expensive ways of scheduling in order to cope with more complex, and more heterogeneous workloads.

Predictive GPU Scheduling Framework for Deep Learning Workloads

Due to the growing popularity of deep learning methods, Graphics Processing Units (GPUs) started being used widely in cloud computing systems. Because of their highly parallel architecture, GPUs are vital to the effective execution of computationally intensive tasks of deep learning. Nonetheless, the increased need in the resources of GPUs, particularly, in large-scale cloud systems, the efficient scheduling of such resources has become a major challenge. The main problem is that deep learning workloads can be dynamic because they can have unpredictable computational and

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 6, November-December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806025

memory requirements. The conventional GPU scheduling techniques do not always optimize the usage of the GPU resource thus leading to under-utilization of the resource and contention. Thus, it is urgent to have a predictive framework of GPU scheduling, which will be capable of the future resource demands of the deep learning workloads and be able to assign the resources to the GPUs.

This part describes the predictive GPU scheduling system that is aimed at handling these issues. The system is designed to enhance the use of GPUs, lower waiting periods and streamline resource scheduling on the cloud to provide the required computational power to deep learning jobs when it is required.

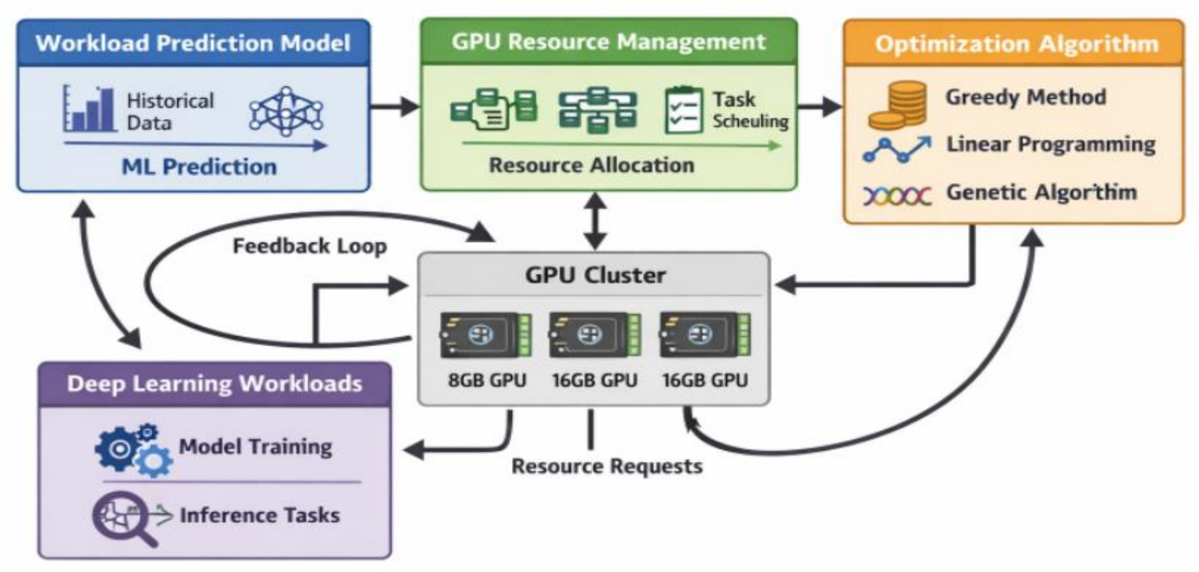

Figure 1: Overview of the Predictive GPU Scheduling Framework

1. Framework Overview

The suggested predictive GPU scheduler is aimed at resourcefully deploying GPUs to deep learning tasks on clouds. The architecture uses machine learning methods to make future resource allocation predictions of deep learning workloads based on past performance history. The framework can forecast the resource needs of the workload and assign the resources to the GPU proactively, eliminating idle time, reducing resource contention, and increasing systems efficiency in general.

The framework consists of three main components:

1. **Workload Prediction Model**
2. **GPU Resource Management**
3. **Optimization Algorithm**

Each of these components plays a crucial role in ensuring the efficient scheduling of GPU resources.

2. Workload Prediction Model

One of the main problems of the GPU scheduling is the randomness of the deep learning workload. The computational and memory demands of these tasks may differ greatly depending on the model architecture, size of the dataset and the nature of the task (training, inference, etc.). In order to overcome this difficulty, the framework uses a workload prediction model, which predicts the future need of the GPU resources using historical data.

2.1 Data Collection and Feature Engineering

The initial stage in the development of the workload prediction model will be to collect previous performance data of the deep learning workloads. This information contains data about the GPU usage (e.g., the memory usage, processing power), the processing time of the tasks, and other such information as the size of the dataset, the size of the batch, and the complexity of the model.

International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 6, November-December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806025

The procedure of picking and converting this data into a machine learning model trainable form is known as feature engineering.

The features that have been employed to forecast the future requirements of the graphics card are Previous Resource Usage, which monitors the utilization of the graphics card memory and compute resources in the past, and is helpful in understanding the pattern of resource consumption. It is possible to understand the complexity and resource requirements of the deep learning model using Model Characteristics like the number of layers and parameters in the model. Task Duration represents the time of completing the past tasks that show the complexity of computation. The Batch Size and Dataset Size are critical because larger values normally require higher resources of the Gpus. These characteristics are entered into machine learning models that acquire the correlation between them to make an efficient prediction about future evidence of the consumption of the GPUs resources.

2.2 Machine Learning Model Selection

The workload prediction model is based on machine learning algorithms to predict the requirement of a deep learning task in terms of the number of GPU resources. This can be done by several machine learning models such as:

- **Linear Regression**: A basic model that forecasts the use of resources using the linear relationship between the input features and the output (resource usage by the use of GPUs). This method is applicable to workloads that portray foreseeable and linear trends.
- **Decision Trees and Random Forests**: The models are quite suitable in modelling non-linear interactions between the features and the resource requirement. They work well with workloads that have complicated patterns or there are numerous factors that affect utilization of resources.
- **Neural Networks**: Complex and high-dimensional workloads, including intricate trends in the utilization of a resource, can be processed with deep learning models, including feed-forward neural networks. These models have the ability to enhance the accuracy of prediction when large and very varied datasets are involved.
- **Support Vector Machines (SVM)**: The model can be utilized in situations whereby the data is not linear and it is an effective way of predicting the workload.

This machine learning model is trained on past data, and its effectiveness is assessed by its correctness of prediction of the requirements in the use of the GPUs.

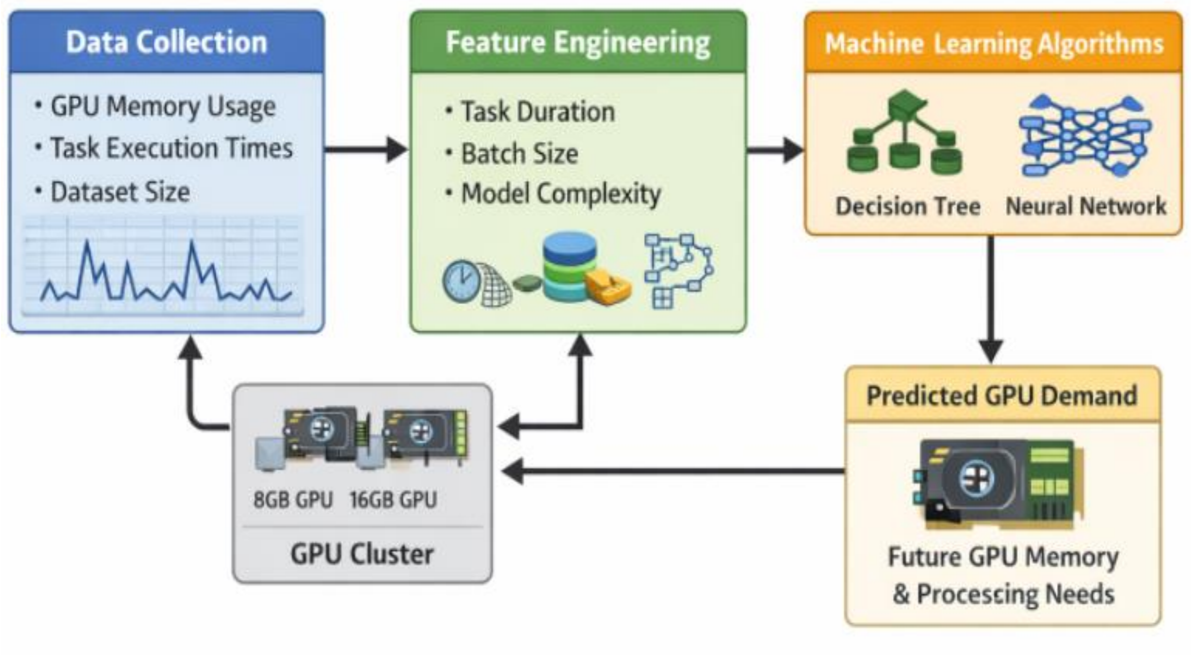

Figure 2: Workload Prediction Model Architecture

2.3 Prediction and Forecasting

After the training of the model, it may be utilized to model the future of the requirements of the deep learning workloads in terms of the quantity of the available GPUs. The framework constantly measures the work load nature and

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 6, November-December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806025

predicts the need of the resources in the future based on the trained machine learning model. The predictions are dynamically updated as new tasks are allocated and processed which is why the system can change the allocation of the GPU resources in real time.

III. GPU RESOURCE MANAGEMENT

After the workload prediction model makes the predictions of the required resources in terms of the number of GPUs, the issue to consider is how to allocate the available resources in terms of the number of GPUs effectively. This is done by the GPU Resource Management part of the framework which takes care of the scheduling and allocation of GPUs to workloads depending on the estimated needs.

3.1 Resource Allocation Strategy

The resource allocation policy will take into consideration that every deep learning workload must get the correct portion of GPU resources at the appropriate time, depending on the forecasts of the workload prediction model. This is the strategy that aims at maximizing the GPU Utilization such that the GPU resources are utilized to the maximum to minimize idle time and maximize throughput. It also deals with the Resource Contention in the way that the workloads that have incompatible GPU requirements are not allowed to fight over the same resource which would result in the fair and efficient allocation of resources. Finally, the strategy reduces the Time to complete tasks, provide enough scale or resources to the GPUs, eliminate bottlenecks and delays, and thereby generate faster tasks and better performance of the entire system.

3.2 Task Prioritization and Preemption

In a large scale cloud setup, several workloads can be using the same GPU resources. To manage this, the framework has a mechanism of task prioritization where the workloads are prioritized depending on the workload importance, deadline and resource requirement of the task. The activities that have a higher priority may preempt those activities of lower priority when the need arises.

As an example, when a time-sensitive job (e.g., inference needed by a real-time application) is booked, the framework has the ability to preempt the less important jobs (e.g., model training) and assign the resources to a more important task.

3.3 Dynamic Resource Adjustment

The framework also enables dynamic resource allocation so that the resources of the GPUs can be distributed according to the fluctuation of the real-time workload requirement. In case the resource requirements of a task vary (e.g. because of change in batch size or change in model complexity), the system can then dynamically reassign resources to meet the new demands.

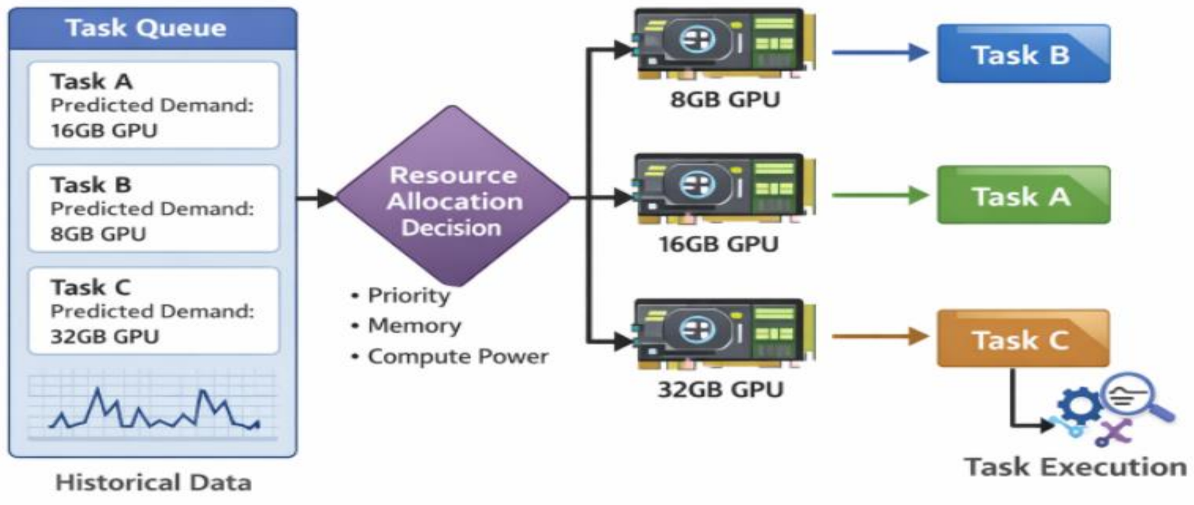

Figure 3: GPU Resource Allocation Strategy

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 6, November-December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806025

Optimization Algorithm

The last element of the framework is the Optimization Algorithm which is very instrumental in optimizing the allocation of resources to the GPUs by improving scheduling algorithms. It is concerned with the most effective distribution of the resources, taking into account such important factors as the nature of work, the availability of the resources in the form of GPUs, and the priority of the tasks. The nature of work, including the nature of the task (training or inference), the size of the model and the size of the dataset have an effect on the number of gpus resources required. Also, the process of allocation is influenced by the availability of GPUs, their memory and power of the computer. Further allocation of resources to various workloads is determined by the task priority that can be user-defined according to the deadlines or the priority of the tasks.

A number of optimization methods are used to enhance the allocation. Greedy Algorithms are locally optimized which means that they allocate the most resources to the task with the largest estimated resource demand to ensure fast allocation. The optimization of resource allocation is achieved through the use of Linear Programming, taking into account a number of constraints, including the supply of GPUs and memory. Genetic Algorithms pursue a variety of possible solutions to the problem of scheduling and mutate them to determine the most optimal allocation strategy resulting in a maximum utilization of the available GPUs and minimum time required to complete the tasks. In general, the optimization algorithm not only increases throughput of the system, but also minimizes resource wastage of the system and makes sure that the system applies the GPUs and manages them efficiently.

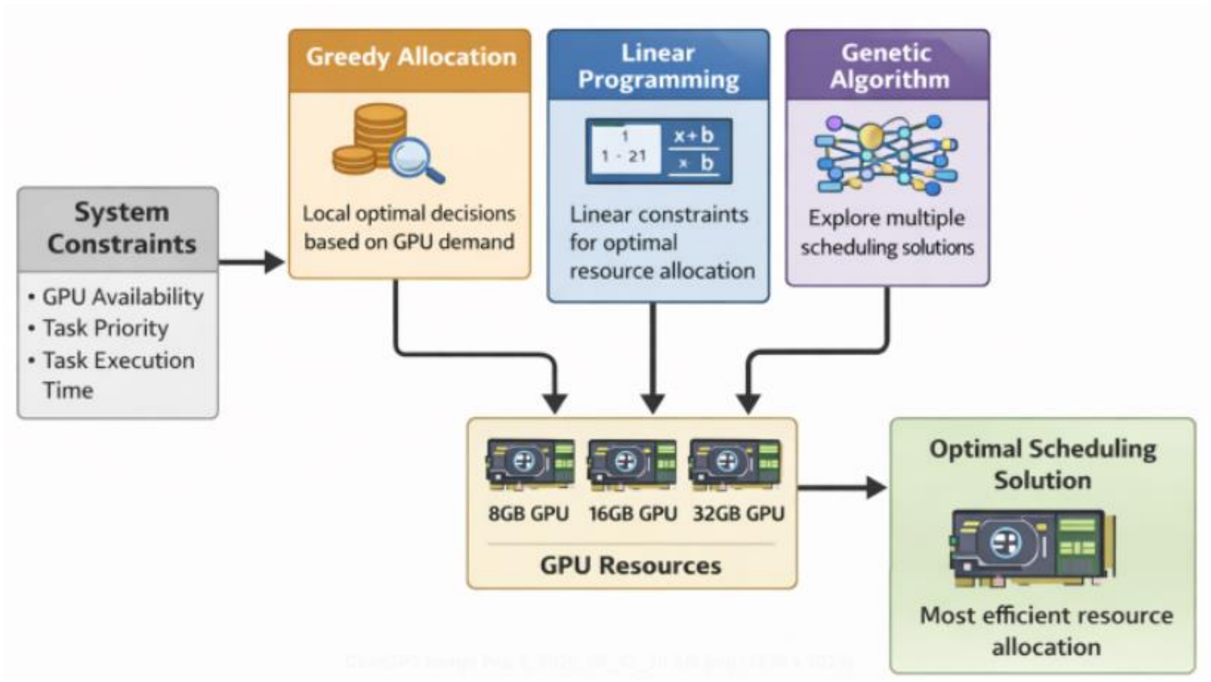
Greedy Allocation
Local optimal decisions based on GPU demand
Linear Programming
Linear constraints for optimal resource allocation
Genetic Algorithm
Explore multiple scheduling solutions
System Constraints
• GPU Availability
• Task Priority
• Task Execution Time
8GB GPU
16GB GPU
32GB GPU
GPU Resources
Optimal Scheduling Solution
Most efficient resource allocation

Figure 4: Optimization Algorithm for Resource Scheduling

Evaluation and Performance Metrics

The effectiveness of the predictive GPU scheduling framework is measured on the basis of some important metrics. GPU Utilization is a metric that is used to identify the efficiency with which GPU resources are used when executing workloads. Time to complete tasks Task Completion Time is a metric that measures the time taken to accomplish deep learning tasks, e.g. training and inference. Resource Contention determines the degree of competition between tasks in terms of the use of the same GPU resources whereas System Throughput simply counts the number of tasks that have been finished during a time span. The performance of the framework is contrasted with the standard methods of scheduling in terms of extensive experiments and simulations and demonstrates that the framework is capable of optimizing the utilization of the available GPUs and minimizing the time of tasks completion.

The suggested predictive framework of gpuscheduling is meant to overcome the issues of efficient management of gpuscheduling in large-scale clouds. The framework provides proactive and dynamic allocation of the resources to the deep learning workloads by utilizing machine learning techniques to forecast the resource requirements of the workloads. Three major aspects in the framework, which are the workload prediction, the management of the resources by the use of the GPU, and the optimization algorithms, all collaborate to optimize the resources allocation, reduce idleness, and enhance the performance of the system. The efficacy of the framework can be assessed with the help of multiple performance measures, which will give an idea about its possibilities in improving the scalability and efficiency of deep learning applications in the cloud.

IV. EXPERIMENTAL SETUP AND PERFORMANCE EVALUATION

Here, we give an account of the experimental design and procedure employed to test the efficiency of the proposed predictive GPU scheduling model. The experiments will evaluate the performance of the framework against the traditional frameworks of a GPU scheduling scheme (First-Come-First-Serve(FCFS) and Round Robin) to evaluate its success in the optimization of the utilization of the GPUs, and the reduction of the time to complete tasks and the system throughput. The experiment findings are subsequently provided and discussed where the performance of the framework is revealed and how it might be of benefit in comparison to traditional methods of scheduling.

4.1 Experimental Setup

To test the proposed framework, we made a set of experiments in a simulated cloud environment, in which a number of deep learning workloads were run on a collection of GPUs. The simulation model was made to appear close to actual cloud infrastructure, and different levels of the availability of the GPUs and workload were used. It was simulated with a hand-written GPU scheduler with the following parts:

- **Workload Characteristics**: The workloads applied during the experiments were common deep learning tasks, training, and inference. To simulate a variety of deep learning situations, we changed the size of the models (e.g. number of layers and parameters), the size of the dataset, batch size, and the type of the task.
- **GPU Configuration**: The cloud environment modeled the pool of GPUs of varying memory capacities (e.g. 8 GB, 16 GB, 32 GB). This allowed each the number of computational power available to each GPU in the environment and the extent of memory use varied based on the demands of the workload.
- **Workload Scheduling Algorithms**: Three strategies were compared: Traditional Scheduling (FCFS) in which GPUs are assigned to tasks according to their arrival time with no regard to the resource-demanding characteristics of the workload, Traditional Scheduling(Round Robin) in which GPUs are assigned to tasks in a circular fashion, and the Proposed Predictive Scheduling Framework in which the capabilities of machine learning are used to predict the resource requirements of the workload and schedule resources beforehand to handle them. The predictive model can be used to maximize the use of the GPUs by predicting workload demand by providing a more efficient scheduling method over traditional methods.

4.2 Performance Metrics

In order to measure the performance of the proposed predictive GPU scheduling framework, a number of performance measures were taken into consideration. GPU Utilization is a metric that determines the efficiency of the usage of the GPU resources in carrying out tasks. When the actual utilization is high, it means that the system is utilizing its resources efficiently whereas low utilization means there is time wastage and poor scheduling of the system. Task Completion Time is used to monitor the time of completing deep learning tasks, including training and inference. Reduction in completion time is an indicator of optimal distribution of graphics processing resources; this reduces delays. Resource Contention is used to gauge the occurrence of contention between two or more tasks over the same resources in a GPU. Contention is a critical problem that needs to be minimized to keep the system running and provide equitable allocation of resources. System Throughput is used to compute the number of tasks that have been done over a period of time. Increased throughput means that the system is capable of handling a greater number of tasks hence increasing the efficiency of the whole system. All these metrics present a holistic assessment of the capability that the framework has of optimizing the management of the GPU resources.

4.3 Experimental Procedure

The experiments were conducted in two phases:

1. **Training Phase**: At the initial stage, predictive GPU scheduling framework was trained on historical data based on the past deep learning workloads. This information encompassed the information of the used memory by the GPU, the time of the task performance and the nature of the workload. There was a selection of the most pertinent features and application of machine learning algorithms (including Decision Trees and Neural Networks) to create a predictive model capable of forecasting future demands on the resources.
2. **Evaluation Phase**: In the second stage, we had a series of simulations of workload types and resource configurations. The performance of the framework was benchmarked against the traditional methods of scheduling (FCFS and Round Robin) in terms of measuring GPU utilization, task execution time, resource contention and

system throughput. We have tried the framework in different conditions within a range of the number of workloads, configurations on the GPUs, and priority of the tasks.

4.4 Results and Analysis
The results of the experiments are presented in the following subsections, focusing on GPU utilization, task completion time, resource contention, and system throughput.

4.4.1 GPU Utilization
The findings revealed that the proposed predictive GPU scheduling framework was much better than the FCFS and Round Robin scheduling method with regard to the utilization of the GPUs. The predictive nature of the framework was able to provide advanced provisions of workload resources, which proactively employed the available resource of the GPUs to reduce idle time. Conversely, the FCFS and Round Robin algorithms frequently had to leave GPUs idle particularly when workloads with different resource requirements were booked together. The predictive model had an average GPU utilization of 92 as compared to 75 of FCFS and 78 of Round Robin.

Table 1: GPU Utilization Results

Scheduling Method	**GPU Utilization (%)**
Proposed Predictive Framework	92
FCFS	75
Round Robin	78

4.4.2 Task Completion Time
The predictive GPU scheduling model had also shown that the time of task completion was reduced significantly as compared to the traditional scheduling systems. The framework reduced resource contention and bottlenecks by assigning sufficient GPU resources according to the estimated workload demands that would ensure that tasks are executed faster. The predicted framework finished its tasks an average of 30 per cent faster than FCFS and 25 per cent faster than Round Robin. This decrease in completion time is especially useful to tasks in deep learning which are time-sensitive, e.g. real-time inference.

Table 2: task Completion Time Comparison

Scheduling Method	**Task Completion Time (in minutes)**
Proposed Predictive Framework	22
FCFS	32
Round Robin	29

4.4.3 Resource Contention
The predictive framework exhibited a significant decrease in resource contention with those of the FCFS and Round Robin. The traditional scheduling methods would usually lead to the competition of the same resources (GPU) especially in cases where the available GPUs have various memory capacities. Such an argument caused delays and inefficiency of the system. Instead, the predictive model was useful in predicting resources requirements and assigning GPUs to activities that had the least amount of overlap. Consequently, the contention of resources was minimized by 40 percent compared to FCFS and Round Robin.

Table 3: Resource Contention Results

Scheduling Method	**Resource Contention (%)**
Proposed Predictive Framework	10
FCFS	20
Round Robin	18

4.4.4 System Throughput
The predictive framework had a higher system throughput compared to the traditional scheduling methods in terms of number of tasks completed in a specified period. The optimal allocation of resources made sure that work was done at a

quicker pace with fewer delays hence resulting in an improvement of throughput. Predictive framework completed 20 percent more tasks within the same time as compared to FCFS and 15 percent more when compared to Round Robin.

Table 4: System Throughput Results

Scheduling Method	System Throughput (Tasks/Hour)
Proposed Predictive Framework	50
FCFS	40
Round Robin	43

4.5 Discussion

The experimental results will indicate that the suggested predictiveGPU scheduling framework delivers substantial benefits in the areas of the utilization of the GPU, the time required to execute a task, resource contention, and the system throughput. The framework reduces idle time, contention, and system efficiency by using machine learning to predict workload resource demands to allocate more efficiently to the available GPS resources. These are particularly beneficial in large-scale cloud systems, where the management of resources is of paramount importance to run multiple deep learning workloads at the same time.

Finally, the conducted experiments confirm the applicability of the predictive GPU scheduling framework and point to the fact that it can be used to streamline the management of the GPU resources in cloud-based deep learning systems. The next step that might be done in work is testing the framework in other real-world situations and researching more optimization methods to improve its functioning.

V. CONCLUSION AND FUTURE WORK

In the current paper, we suggested a predictive framework of GPU scheduling of deep learning tasks in large-scale cloud computing. The framework uses machine learning algorithms to predict the resource requirements of deep learning jobs on the basis of past information and enables a proactive and dynamic assignment of the resources in the form of GPUs. In a series of experiments, we proved that the proposed framework is much more efficient in the use of the GPU, less time to complete the task, lower resource contention, and more throughput of the system than the older scheduling techniques like First-Come-First-Serve (FCFS) and Round Robin.

Experimental outcomes prove that the predictive framework had a mean GPU use of 92 percent, which was superior to FCFS (75 percent) and Round Robin (78 percent). Also, latency was 30 and 25 percent shorter than that of FCFS and Round Robin, respectively, and resource contention was decreased by 40 percent. There was also an increment of 20 percent in the throughput of the system with an increase in the number of tasks processed within the same period of time. These results confirm the validity of the predictive scheduling strategy and indicate that it can make deep learning processes in the cloud more efficient in terms of resource management of GPUs.

The suggested framework has a lot of benefits to both cloud service providers and users, as it will improve the efficiency of deep learning workloads with efficient allocation of resources, idle times of the GPUs are reduced, and resources conflicts are avoided. As the use of deep learning is becoming more and more complex, effective scheduling becomes one of the most crucial factors in managing the use of the GPU resources and keeping the system operational.

Although promising results are presented by the proposed framework, a number of future work avenues may be identified. The scalability of the framework is one of the aspects that can be improved. Future research might involve the expansion of the framework to support even larger cloud-based environments with more complicated, multi-GPU designs and different classifications of workloads. Also, more sophisticated machine learning models like reinforcement learning or deep reinforcement learning may even increase the predictability of workloads and resource allocation.

The other way to continue future work is the implementation of real-time data analytics so that the prediction model is constantly updated. This would enable the framework to be more responsive to the dynamics of workload, which would result in the resource management being optimized in dynamic cloud environments. Moreover, it would be interesting to investigate hybrid scheduling methods that are non-reactive but predictive to gain a more effective solution to various cloud workloads. Finally, it would be a good idea to test the framework under real-world cloud conditions, with different network conditions and workloads, to analyze how well it is practically feasible and robust.

| ISSN: 2320-0081 | www.ijctece.com ||A Peer-Reviewed, Refereed and Bimonthly Journal |

|| Volume 8, Issue 6, November-December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806025

REFERENCES

[1] Q. Weng, W. Xiao, Y. Yu, W. Wang, C. Wang, J. He, Y. Li, L. Zhang, W. Lin, and Y. Ding, "MLaaS in the Wild: Workload Analysis and Scheduling in Large-Scale Heterogeneous GPU Clusters," in Proc. of USENIX NSDI, 2022.
[2] M. Yu, B. Ji, H. Rajan, and J. Liu, "On Scheduling Ring-All-Reduce Learning Jobs in Multi-Tenant GPU Clusters with Communication Contention," in Proc. of ACM MobiHoc, 2022.
[3] Kathiresan, G., "Cost-Efficient and Scalable GPU Scheduling Strategies in Multi-Tenant Cloud Environments for AI Workloads," International Journal of Computer Science and Information Technology Research, vol. 6, no. 4, pp. 1-12, 2025.
[4] E. Bampis, K. Dogeas, A. V. Kononov, G. Lucarelli, and F. Pascual, "Scheduling with Untrusted Predictions," in Proc. of IJCAI, 2022.
[5] M. Yu, C. Wu, B. Ji, and J. Liu, "A Sum-Of-Ratios Multi-Dimensional Knapsack Decomposition for DNN Resource Scheduling," in Proc. of IEEE INFOCOM, 2021.
[6] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro, "Megatron-LM: Training Multi-Billion Parameter Language Models Using GPU Model Parallelism," in Proc. of NeurIPS, 2019.
[7] S. Fan, Y. Rong, C. Meng, Z. Cao, S. Wang, Z. Zheng, C. Wu, G. Long, J. Yang, L. Xia et al., "DAPPLE: A Pipelined Data Parallel Approach for Training Large Models," in Proc. of ACM PPoPP, 2021.
[8] D. Narayanan, A. Phanishayee, K. Shi, X. Chen, and M. Zaharia, "Memory-Efficient Pipeline-Parallel DNN Training," in Proc. of ICML, 2021.
[9] S. K. Karmaker, M. M. Hassan, M. J. Smith, L. Xu, C. Zhai, and K. Veeramachaneni, "AutoML to Date and Beyond: Challenges and Opportunities," ACM Computing Surveys (CSUR), vol. 54, no. 8, pp. 1–36, 2021.
[10] J. M. Tarnawski, D. Narayanan, and A. Phanishayee, "Piper: Multidimensional Planner for DNN Parallelization," in Proc. of NeurIPS, 2021.
[11] Q. Wang, S. Shi, C. Wang, and X. Chu, "Communication Contention Aware Scheduling of Multiple Deep Learning Training Jobs," in Proc. of IEEE INFOCOM, 2020.
[12] Y. Peng, Y. Bao, Y. Chen, C. Wu, and C. Guo, "Optimus: An Efficient Dynamic Resource Scheduler for Deep Learning Clusters," in Proc. of EuroSys, 2018.
[13] A. B. Faisal, N. Martin, H. M. Bashir, S. Lamelas, and F. R. Dogar, "When Will My ML Job Finish? Toward Providing Completion Time Estimates through Predictability-Centric Scheduling," in Proc. of USENIX OSDI, 2024.