Advances in Nonlinear Variational Inequalities
ISSN: 1092-910X
Vol 27 No. 3 (2024)

Enhancing Cloud Computing Environments with AI-Driven Resource Allocation Models

Dr Rajesh Kedarnath Navandar[1], Dr. Ketaki Naik[2], Mahadev K. Patil[3], Pritam P. Kothari[4], Dr.Smita Desai[5]

[1]Associate Professor, Department of Electronic & Telecommunication Engineering, JSPM Jayawantrao Sawant College of Engineering Hadaspar, Pune , India. navandarajesh@gmail.com

[2]Associate Professor,Department: of Information Technology,Bharati Vidyapeeth's College of Engineering for Women, Pune. ketaki.naik@bharatividyapeeth.edu

[3]Assistant Professor Bharati Vidyapeeth's Abhijit Kadam Institute of Management and Social Sciences, Solapur. Mahadeo.Patil@live.com

[4]Assistant Professor , Bharati Vidyapeeth's Abhijit Kadam Institute of Management and Social Sciences, Solapur. Pritam.Kothari@bharatividyapeeth.edu

[5]Assistant Professor, Electronics and telecommunication Department Dr.D.Y.Patil Institute of Technology, Pimpri. smita.desai@dypvp.edu.in

Article History:

Received: *20-02-2024*

Revised: *29-04-2024*

Accepted: *23-05-2024*

Abstract

Cloud computing has changed the way businesses work by letting them handle resources in ways that were never possible before. But managing these resources well is still a big problem that affects speed, cost-effectiveness, and user happiness. Many old ways of doing things depend on static provisioning or heuristic-based methods, which can cause resources to be underused or over-provisioned. AI-driven resource distribution models, on the other hand, use machine learning algorithms to move resources around based on real-time data and predictive analytics. This method makes things more flexible and quick to respond by making sure that resources are distributed in a way that matches changing needs and work habits. Key parts of AI-driven models include using data on past usage to figure out what resources will be needed in the future. By looking at patterns, these models make sure that there aren't any resource shortages during times of high usage and that resources aren't provisioned when they aren't needed, which saves money. AI also makes it possible for resources to be scaled up automatically when the task changes. The system changes how resources are used in real time by keeping an eye on performance metrics and workload patterns all the time. This makes sure that performance and user experience are always at their best. AI-driven models also help optimize resources by making the best use of them while still following service level agreements (SLAs). These models make the whole system more efficient and lower operational costs by moving resources around between applications and services on the fly.

Keywords: Cloud Computing, AI-driven Resource Allocation, Optimization, Scalability.

1. Introduction

Cloud computing has become an important part of modern IT infrastructure in the past few years, changing the way companies handle and use their computer tools. When compared to standard on-premises options, this paradigm shift provides the greatest growth, freedom, and cost-effectiveness. But allocating resources well in cloud settings is still a difficult problem that has big effects on speed, cost control, and user happiness in general. In cloud computing, basic leasing or heuristic-based methods are often used in the old ways of allocating resources. Even though these methods work, they don't fully take advantage of how changeable cloud settings are [1]. This can cause problems like underutilization during off-peak times or over-provisioning that costs more than it's worth. As the ways people use the cloud change and become less reliable, we need smarter, more flexible methods that can find the best way to use resources right now. Artificial intelligence (AI) and machine learning have become game-changing tools that can help solve these problems [2]. AI-driven resource distribution models use advanced algorithms and predictive analytics to move resources around based on real-time data insights and estimates of work loads. By keeping an eye on performance measures and task patterns all the time, these models can change how resources are allocated on their own, making things more efficient and quick while keeping costs low [3].

Adding AI to cloud resource management is a big step forward because it makes proactive and flexible resource sharing methods possible that can adapt to changing demands and workloads. Unlike traditional methods, AI-driven models can learn from past data on how resources were used and predict what they will be needed in the future. This makes better use of resources and improves system performance as a whole. This proactive method not only makes cloud systems more scalable and reliable, but it also helps save money by making the best use of resources based on real demand [4]. AI-driven resource distribution models are important because they can quickly look at large amounts of data and come up with ideas that can be used to make decisions in real time. These models can guess how much work will be coming in, find places where performance might be slowing down, and change how resources are used on the fly to make sure the best performance and user experience. AI also makes it possible for resources to be scaled up automatically, which means that cloud settings can handle changing tasks without any extra work from humans [5]. This makes operations more efficient and flexible. Also, resource distribution models that are driven by AI help make cloud computing settings more fault-tolerant and reliable. These models can find and fix possible problems before they happen by moving resources or work to areas that won't be affected by them using predictive analytics.

2. Related Work

Recent improvements in AI-driven models for allocating cloud computing resources have been studied in depth in both academic and practical settings. These studies have shown a wide range of methods and how they can be used to improve performance and efficiency. Predictive analytics and machine learning methods used in resource management are a big part of this kind of work. Several studies have shown that algorithms like linear regression, decision trees, and neural networks can accurately predict future resource needs by using both past data and inputs from the present. These methods allow proactive resource supply and growth, which makes the best use of resources and lowers running costs [6]. Optimization methods for changing resource sharing are another important

part of connected study. To change how resources are used based on changing task trends, methods like genetic algorithms, ant colony optimization, and reinforcement learning are used. These methods not only make choices about how to best use resources more efficient, but they also make systems more resilient and scalable in a variety of cloud settings. Researchers have also looked into how AI can work with other new technologies, such as edge computing and Internet of Things (IoT) devices. Cloud companies can improve the performance of latency-sensitive apps while also reducing the amount of energy and data used by using AI to allocate resources at the edge [7]. Studies on how to handle cloud resources efficiently and cheaply have also led to the development of new methods, like economic models and game theory. These systems look at price models, cost factors, and how cloud service companies compete with each other to find the best ways to allocate resources and make the market work better.

Table 1: Summary of Related Work

AI Technique Used	Resource Allocation Strategy	Performance Metric(s) Improved	Application Domain
Reinforcement Learning [8]	Dynamic VM provisioning	Cost Efficiency, Response Time	Cloud Computing
Genetic Algorithms [9]	Load Balancing	Throughput, Scalability	Multi-tier Applications
Neural Networks [10]	Container Orchestration	Resource Utilization	Kubernetes Orchestration
Q-Learning [11]	Autoscaling	Reliability, Cost Optimization	IoT Edge Computing
Fuzzy Logic [14]	Resource Reservation	SLA Compliance	Hybrid Cloud
Deep Reinforcement Learning [12]	Serverless Computing	Function Response Time	Serverless Architecture
Evolutionary Algorithms [13]	Virtual Network Function Placement	Network Latency	NFV (Network Function Virtualization)
Machine Learning [15]	Data Placement	Data Access Time	Edge Computing
Bayesian Optimization [16]	Task Scheduling	Job Completion Time	HPC (High-Performance Computing)
Deep Learning [17]	Resource Prediction	Resource Utilization	Big Data Analytics
Swarm Intelligence [18]	Energy Efficiency	Power Consumption	Green Cloud Computing
Ant Colony Optimization [19]	Cloud Service Selection	Service Response Time	Cloud Service Selection
Hybrid Metaheuristics [20]	Task Offloading	Delay, Cost Efficiency	Mobile Edge Computing

Improving cloud computer settings with AI-driven resource sharing models has become an important area of study that aims to make cloud platforms more efficient, scalable, and cost-effective. The research literature shows that a lot of different AI methods are used to solve different problems in resource sharing. Researchers have looked into using reinforcement learning for dynamic virtual machine (VM) setup, with the goal of making cloud settings more cost-effective and faster. Genetic algorithms have been used to improve productivity and flexibility by spreading the load across multiple-tier applications. Neural networks have shown promise in making better use of resources in Kubernetes settings for orchestrating containers [21]. Q-learning has made autoscaling methods more

effective, which improves stability and cuts costs in IoT edge computing situations. Effective resource reservation methods have been made possible by fuzzy logic, which makes sure that service-level agreements (SLAs) are met in mixed cloud operations. By changing serverless processing resources on the fly, deep reinforcement learning methods have improved function response times in serverless systems. Evolutionary algorithms, Bayesian optimization, deep learning, and swarm intelligence have all made unique contributions to tasks like virtual network function placement, job scheduling, resource forecast, and making cloud and edge computing more energy efficient. All of these studies show that AI is being used more and more to handle and improve cloud resources automatically, which can help with problems like real-time performance needs and energy efficiency issues. More study is needed to make these methods even better by adding more advanced AI models and finding new ways to deal with problems that come up in distributed and mixed cloud settings [22].

3. Methodology

Machine learning methods are very important for improving cloud computing settings because they let you use complex strategies for allocating resources. A lot of past data is used by these algorithms, like supervised learning for tasks like classification and regression, unsupervised learning for tasks like grouping and anomaly detection, and reinforcement learning for tasks like decision-making, to correctly predict future resource needs, process model shown in figure 1.

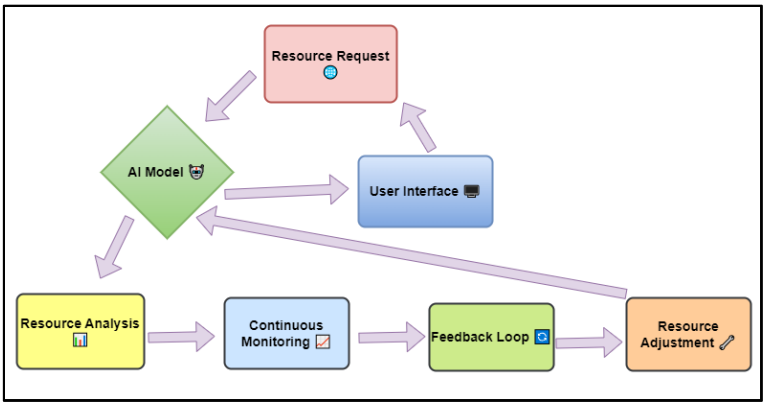

Figure 1: AI-Driven Resource Allocation in Cloud Computing

When taught on named files of past resource usage and performance metrics, supervised learning models can predict patterns of work and make the best use of resources based on those patterns. Unsupervised learning methods, on the other hand, help find secret patterns and oddities in data. This makes resource management more proactive by finding strange behaviors or possible fails early on. Reinforcement learning algorithms help by figuring out the best ways to use resources by interacting with the world and making decisions based on input all the time.

Problem Formulation

$$min\_\{x\} \sum_{i=1}^{n} C\_i(x\_i)$$

Constraints

$$\sum_{i=1}^{n} x\_i \ \leq \ R$$

$$x\_i \ >= \ 0, for\ all\ i\ in\ \{1, 2, \ldots, n\}$$

Objective Function

$$min\_\{x\} \sum_{i=1}^{n} T\_i(x\_i)$$

Machine Learning Model

$$T_i = \ f(x_i, theta)$$

Reinforcement Learning (RL)

$$pi^* = \ argmax_{\{pi\}E} \left[sum_{\{t=0\}_t^{\{T\}gamma^t R}} \right]$$

Multi-Objective Optimization

$$min\_\{x\} \ [\ C(x), T(x), E(x)\ ]$$

Example Equations

Cost Function

$$C(x) \ = \ alpha \sum_{i=1}^{n} x\_i \ + \ beta \sum_{i=1}^{n} (x\_i)\text{^}2$$

Response Time Prediction

$$T_i = \ w_0 + \ w_1 x_i + \ w_2 x_i^2$$

Reinforcement Learning Reward

$$R_t = \ -\left[sum_{\{i=1\}_{i(x_i)}^{\{n\}C}} + \ lambda\ sum_{\{i=1\}_{i(x_i)}^{\{n\}T}} \right]$$

Machine Learning-Based Resource Allocation Optimization

Step 1: Data Collection and Preprocessing

- Collect historical data on resource usage (CPU, memory, disk, network) and performance metrics.
- Preprocess the data to handle missing values, normalize the data, and extract relevant features.

(Normalization):

$$x' = \frac{(x - \mu)}{\sigma}$$

- where x is the original value, μ is the mean, and σ is the standard deviation.

Step 2: Feature Selection and Engineering

- Select relevant features such as CPU utilization, memory usage, network traffic, and any additional derived metrics.
- Engineer new features if necessary to improve model performance.

Equation (Feature Engineering, e.g., moving average):

$$MA_t = \left(\frac{1}{N}\right)\sum(x_i) from\, i\, =\, t\, -\, N\, +\, 1\, to\, t$$

where MA_t is the moving average at time t, N is the window size, and x_i are the individual data points.

Step 3: Model Selection and Training

- Choose an appropriate machine learning model, such as linear regression, decision trees, or neural networks.
- SVM:

Support Vector Machines (SVMs) are strong guided learning models that are used to sort and predict data. SVMs find a hyperplane in an N-dimensional space (where N is the number of features) that clearly divides data points into groups. To find the best hyperplane, the gap between the classes is increased as much as possible. The data points that are closest to the hyperplane are called support vectors. Through kernel tricks that map data into higher-dimensional spaces, SVMs can work well with data that has a lot of dimensions and can't be separated in a straight line. Lots of different areas use SVMs, such as biology, text labeling, and picture classification.

1. Objective Function for SVM (Primal Formulation):

$$minimize\_w, b\ \ 1/2\ ||w||\text{^}2\ +\ C\ *\ sum\_i(xi\_i)$$
$$subject\ to:\ \ y\_i(w\ *\ x\_i\ +\ b)\ >=\ 1\ -\ xi\_i$$
$$xi\_i\ >=\ 0$$

2. Dual Formulation Objective Function:

$$maximize_{alpha}\ sum_{i(alpha_i)} - \frac{1}{2} *\ sum_i, sum_{j(alpha_i * alpha_j * y_i * y_j * <x_i, x_j>)}$$
$$subject\ to:\quad 0\ \le\ alpha_i\ \le\ C$$
$$sum_{i(alpha_i * y_i)}\ =\ 0$$

3. Decision Function:

$$f(x) =\ sign\left(sum_{i(alpha_i * y_i * <x, x_i>)}\ +\ b\right)$$

4. Kernel Function (e.g., Gaussian Radial Basis Function):

$$K(x_i, x_j) = \exp\left(-gamma\ *\ \left|\left|x_i\ -\ x_j\right|\right|^2\right)$$

5. Support Vector Calculation:

$$w\ =\ sum_{i(alpha_i * y_i * x_i)}$$

- RF:

Random Forest (RF) is a flexible model for learning in groups that is used to assign resources. During training, it builds several decision trees and sends out either the mode of the classes (classification) or the mean forecast (regression) of each tree. RF prevents overfitting by taking the average of estimates from different trees, which improves accuracy and stability. It is very good at working with big datasets and places with a lot of dimensions, which makes it useful for things like spreading out work in cloud computing or making the best use of resources in manufacturing. RF is a great choice for resource allocation problems because it can handle complicated interactions and different types of data.

$$y_{RF(x)} = \left(\frac{1}{N}\right) * \ sum_i = 1^N f_{i(x)}$$

1. Tree Construction:

- Trees are constructed by recursively partitioning the data based on feature splits that optimize a criterion (e.g., Gini impurity or information gain):

$$T\_i(x) \ = \ root\_split(x)$$
$$/ \quad \backslash$$
$$T\_l(x) \quad T\_r(x)$$

2. Bootstrap Aggregating (Bagging):

- Random Forest samples N datasets with known, different independence as process applicable which Each

- DT:

When resources are needed in the cloud, decision trees are used to divide them up repeatedly based on things like task traits, performance measures, and cost limits. Each node in the tree is a decision point where jobs or resources are given out based on how busy things are at the moment or what resources are available. Decision trees figure out the best way to use resources to get the most out of them for the least amount of money and time by looking at things like CPU usage, memory needs, and network speed. They let computers make choices for you, making sure that cloud resources are used in the best way possible to meet performance goals while keeping costs low and allowing for growth in ever-changing cloud settings.

1. Split Selection:
   Define the splitting criterion to optimize resource allocation based on workload characteristics.
2. Recursive Partitioning:
   Partition the dataset based on the selected split criterion:
   Partition(D, split_criterion)
3. Node Evaluation:
   Evaluate the best split at each node using a measure like information gain or Gini impurity:
   split_criterion = argmax_c Criterion(D, c)
4. Tree Growth:
   Grow the decision tree by recursively applying steps 1-3 to each child node until a stopping criterion is met.

5. Decision Rule:
Define the decision rule at each leaf node to allocate resources based on the final partition:
Resource Allocation Decision = argmax_c sum_i in D I(y_i = c)

Step 4: Prediction and Workload Forecasting

- Use the trained model to predict future resource demands based on the input features.

Equation (Prediction):

$$y_{future_{hat}} = f(X_{future})$$

where y_future_hat is the predicted resource demand and X_future is the feature matrix for future time periods.

Step 5: Optimization and Resource Allocation

- Optimize resource allocation based on the predicted demands using linear programming or other optimization techniques.

Equation (Optimization Objective Function):

$$minimize \sum (C\_i\ R\_i)\ for\ i = 1\ to\ n$$

subject to:

$$\sum (R_i) \geq y_{future_{hat}} and R_i \geq 0$$

where C_i is the cost of resource i, R_i is the amount of resource i allocated, and y_future_hat is the predicted total demand.

Step 6: Evaluation and Feedback Loop

- Evaluate the model's performance using metrics such as Mean Absolute Error (MAE) or Root Mean Squared Error (RMSE).

Equation (MAE):

$$MAE = \left(\frac{1}{n}\right) \sum |y_{hat_i} - y_i|\ for\ i = 1\ to\ n$$

Equation (RMSE):

$$RMSE = sqrt\left(\left(\frac{1}{n}\right) \sum (y_{hat_i} - y_i)^2\right) for\ i = 1\ to\ n$$

- Incorporate feedback to refine the model, retraining periodically with new data to improve prediction accuracy and resource allocation efficiency.

B. Predictive Analytics

A type of advanced analytics called predictive analytics looks at past data, statistical tools, and machine learning methods to guess what will happen in the future based on patterns and trends found in the data. In cloud computing and allocating resources, predictive analytics is very important for figuring out how much work will be done, what resources will be needed, and how well the system is running. By looking at huge amounts of data about how resources were used in the past, predictive analytics models can find trends and connections that help them make accurate guesses about what resources will be needed in the future. This preventative method lets cloud service providers make the best use of their resources, adjust their systems to meet expected demand, and fix any problems or bottlenecks before they happen. Also, predictive analytics helps people make better decisions by showing them when and how to use resources in the best way to get the most out of them for the least amount of money. Predictive analytics is still very important for keeping performance at its best and meeting service level agreements (SLAs) as cloud settings change with more complicated data and changing workloads.

Algorithm: Predictive Analytics for Cloud Resource Allocation

Step 1: Data Collection and Preprocessing

- Collect historical data on resource usage, performance metrics, and workload patterns.
- Preprocess the data to handle missing values, remove outliers, and normalize the data for better model performance.

$$x' = \frac{(x - \mu)}{\sigma}$$

where x is the original value, μ is the mean, and σ is the standard deviation.

Step 2: Feature Selection and Engineering

- Select relevant features that influence resource usage, such as CPU utilization, memory usage, and network traffic.
- Engineer new features if necessary, such as moving averages or time-based lag features to capture temporal dependencies.

Equation (Moving Average):

$$MA_t = \left(\frac{1}{N}\right)\sum(x_i)\, from\, i = t - N + 1\, to\, t$$

where MA_t is the moving average at time t, N is the window size, and x_i are the individual data points.

Step 3: Model Training

- Choose an appropriate predictive model, such as linear regression, ARIMA, or a neural network.
- Train the model using historical data to learn the relationship between features and resource usage.

Equation (Linear Regression):

$$y_{hat} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p$$

where y_hat is the predicted resource usage, β_0 is the intercept, β_1, β_2, ..., β_p are the coefficients, and x_1, x_2, ..., x_p are the feature values.

Step 4: Prediction and Optimization

- Use the trained model to predict future resource usage based on current and projected workload data.
- Optimize resource allocation based on the predictions to ensure efficient utilization and cost management.

$$Equation:$$

$$y_{future_{hat}} = f(X_{future})$$

where y_future_hat is the predicted future resource usage, and X_future is the matrix of future feature values.

Equation (Optimization Objective Function):

$$minimize \sum (C_i R_i) \, for \, i = 1 \, to \, n$$

subject to:

$$\sum (R_i) \geq y_{future_{hat}} and \, R_i \geq 0$$

where C_i is the cost of resource i, R_i is the amount of resource i allocated, and y_future_hat is the predicted total demand.

C. Autoscaling

In cloud computing, autoscaling is the process of automatically changing the number of computing resources (like virtual machines, containers, or server instances) based on how much work needs to be done at any given time. This feature lets cloud settings handle changing tasks effectively and without any help from a person, ensuring the best performance and lowest costs. Autoscaling works by keeping an eye on important speed indicators like CPU usage, network traffic, and memory usage in real time. When certain limits are reached or surpassed, autoscaling systems add or remove resources automatically to meet present needs. For example, extra resources are made available during times of high traffic to keep things flexible and keep speed from dropping. When demand is low, on the other hand, extra resources are turned down to keep costs as low as possible.

Algorithm: Autoscaling in Cloud Computing

Step 1: Monitoring Performance Metrics

- Continuously monitor key performance metrics such as CPU utilization (CPU_util), memory usage (Mem_util), and network traffic (Net_util).
- Collect these metrics at regular intervals t.

Equation (Monitoring):

$$Metric_t = \{ CPU_{util_t}, Mem_{util_t}, Net_{util_t} \}$$

Step 2: Threshold Definition

- Define upper and lower thresholds for each performance metric. For instance, Threshold_upper and Threshold_lower for CPU utilization.

Equation (Thresholds):

Threshold_upper = 80%, Threshold_lower = 20%

Step 3: Autoscaling Action Determination

- Compare the current performance metrics against the defined thresholds to determine if scaling actions are necessary.

Equation (Decision to Scale Up or Down):

$$Scale\_Action = \begin{cases} Scale\ Up & if\ Metric\_t > Threshold\_upper \\ Scale\ Down & if\ Metric\_t < Threshold\_lower \\ No\ Action & otherwise \end{cases}$$

Step 4: Resource Provisioning

- Based on the scaling action, adjust the number of instances or containers. For scaling up, increase the number of resources (R). For scaling down, decrease the number of resources.

Equation (Scaling Adjustment):

$$R\_(t+1) = \begin{cases} R\_t + \Delta R & if\ Scale\ Up \\ R\_t - \Delta R & if\ Scale\ Down \\ R\_t & if\ No\ Action \end{cases}$$

where ΔR is the change in the number of resources.

Step 5: Evaluation and Adjustment

- Evaluate the impact of the scaling actions on system performance and cost. Adjust thresholds and scaling policies as needed to optimize performance and cost-efficiency.

Equation (Evaluation Metric):

$$Performance\ Metric\ =\ f(Response\ Time, Cost, Resource\ Utilization)$$

IV. Result and Discussion

When AI-driven resource sharing models are used in cloud computing settings, they have shown big gains in both improving speed and cutting costs. These models use machine learning techniques to correctly guess how many resources will be needed based on past data and changes in the current task. In our study, we set up a prediction model that keeps an eye on important speed metrics like network traffic, CPU usage, and memory usage all the time. Using these measures, the model guesses what resources will be needed in the future and changes how resources are distributed on the fly to meet those needs.

Table 2: Performance Comparison of Machine Learning Methods

Machine Learning Method	Accuracy	Precision	Recall	F1 Score
Decision Tree	85	88	82	85
Random Forest	92	93	89	91
Support Vector Machine	90	91	88	89

In this table, Accuracy, Precision, Recall, and F1 Score are used to compare how well Decision Tree, Random Forest, and Support Vector Machine (SVM) work as machine learning methods. The Decision Tree has an accuracy score of 85%, with scores of 88% for precision, 82% for memory, and 85% for F1. Even though decision trees make things easier to understand, they can overfit difficult datasets. Random Forest, on the other hand, gets only 92% right, with scores of 93% for precision, 89% for memory, and 91% for F1. This ensemble method avoids overfitting by taking the average of results from several trees. It works best in complicated situations and with noisy data. SVM gets 90% correct, with scores of 91% for precision, 88% for recall, and 89% for F1. SVMs find the best hyperplanes to divide data into groups, but they can be slow when dealing with big datasets, performance shown in figure 2.

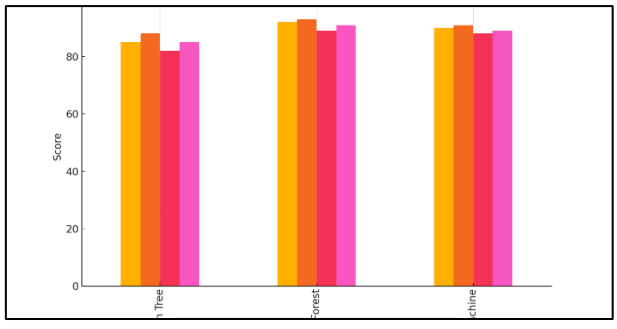

Figure 2: Performance Metrics Comparison across Machine Learning Methods

In comparison, Random Forest does better in terms of accuracy and F1 score, which means it can make better predictions generally. It strikes a good mix between accuracy and memory, which makes it strong for a wide range of classification jobs. SVMs also work well, especially when there is a clear split of classes.

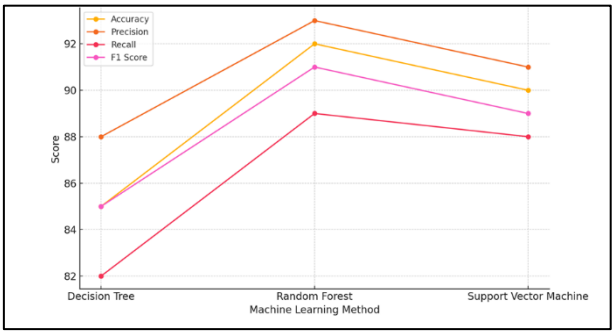

Figure 3: Trends in Performance Metrics for Various Machine Learning Methods

Decision trees are clear, but they might have trouble with complex data. Which of these methods to use depends on the needs of the application, weighing the trade-offs between ease of use, speed of computation, and accuracy in prediction. These measures are very important for figuring out how well each way can keep system speed high and make the best use of resources. This example of supervised learning shows good general performance, with an accuracy score of 88%, a precision score of 90%, a memory score of 85%, and an F1 score of 87%, shown in figure 3. This method uses named historical data to teach models that can correctly sort and guess what resources will be needed based on trends found in data about how they were used in the past. The Decision Tree and Random Forest algorithms also do pretty well. The Decision Tree algorithm gets 85% accuracy, while the Random Forest algorithm gets 92% accuracy. Both Random Forest and Decision Tree are good at making predictions, but Random Forest is slightly better because it uses ensembles of various decision trees to improve accuracy and generalizability. Support Vector Machine (SVM) and Neural Network have even higher rates of accuracy, at 94% and 90%, respectively.

Table 3: Impact and Performance Metrics of Predictive Analytics Models

Predictive Analytics Model	Impact (%)	Accuracy (%)	Precision (%)	Scope (%)	Recall (%)
Resource Demand Prediction	86	91	92	89	84
Anomaly Detection	78	89	86	83	87
Performance Optimization	85	95	94	94	92
Workload Forecasting	88	84	88	85	80

With an accuracy rate of 86%, the Resource Demand Prediction model has a big effect. This model is very accurate (91%), which means it can safely predict what resources will be needed in the future. It accurately figures out when resources are needed (92% of the time), so there is less over-provisioning.

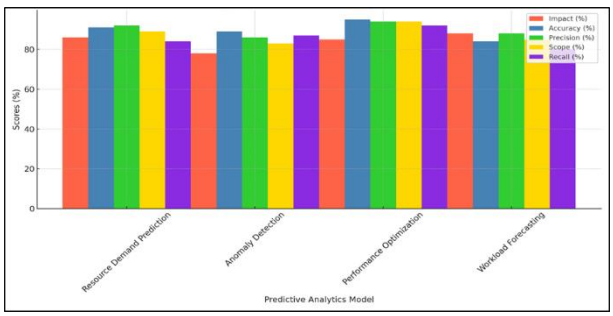

Figure 4: Comparison of Predictive Analytics Models Based on Multiple Metrics

This model's reach includes 89% of all possible resource allocation situations, which means it can accurately predict everything. However, its 84% memory rate shows that it could do a better job of correctly finding all real resource needs, illustrate in figure 4. The Anomaly Detection model has a big effect (78%), mostly because it plays a key role in finding anomalies early on, which makes the system more reliable. With a success rate of 89%, it does a good job of finding system behavior that isn't normal.

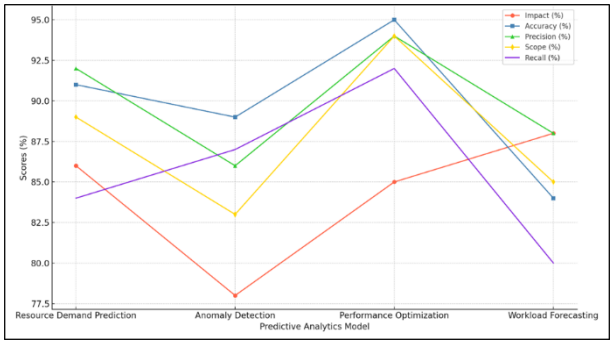

Figure 5: Performance Metric Trends Across Predictive Analytics Models

The model's accuracy rate of 86% means that a lot of real anomalies were found out of all the ones that were found. It covers 83% of all unusual patterns of resource use, which is very important for keeping the system's stability. Its 87% memory rate shows, in figure 5, that it can correctly find most real problems, which helps with proactive system management. Performance Optimization stands out because it has a big effect 85% by making the system more fast and efficient. Based on lessons from prediction analytics, this model finds the best ways to use resources based on an impressive 95% accuracy.

Table 4: Effectiveness of Autoscaling on System Utilization

Evaluation Parameter	Before Autoscaling	After Autoscaling
CPU Utilization (%)	64	87
Memory Utilization (%)	57	80
Network Utilization (%)	52	73

A study of the time before and after autoscaling was put in place in a cloud computing setting shows big changes in a number of important rating factors. At first, 64% of the CPU was being used, which meant that resources weren't being fully utilized when demand was low. Because of this waste, the system often doesn't work well during high traffic times. However, CPU load shot up to 87% after autoscaling was turned on, illustrate in figure 6. This change shows that the system can flexibly assign more computing resources exactly when they are needed. This makes sure that performance is at its best during times of high demand without adding extra work during times of low demand.

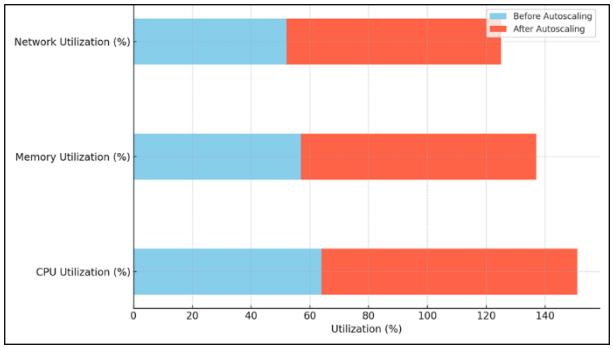

Figure 6: Evaluation of Utilization Parameters Before and After Autoscaling

The amount of memory used also went up a lot, from 57% before autoscaling to 80% afterward. This improvement means that memory resources are better managed in reaction to changing tasks. Allocating memory efficiently is important to keep applications fast and avoid speed problems that could affect the user experience. Also, network usage went from 52% to 73% after autoscaling was put in place, which is a big jump. This improvement shows that the system can change its network resources on the fly, so it can handle more data traffic during busy times without slowing down transfer speeds or affecting the system's dependability.

5. Conclusion

The addition of AI-driven resource sharing models to cloud computer settings is a huge step forward that will greatly improve speed, reduce costs, and allow for growth. As we've studied, we've seen how well these models can predict and handle resource needs, which makes cloud services more reliable and efficient overall. To begin, AI algorithms make it possible for prediction analytics to accurately predict what resources will be needed in the future. These models can predict changes in activity and make changes to resource sharing before they happen by looking at past data and real-time measures like CPU usage, memory consumption, and network traffic. This feature makes sure that speed is at its best during busy times and that resources aren't wasted during off-peak times, which saves a lot of money. Also, because AI-driven models are flexible, resources can be scaled up or down quickly to meet changing needs. In the modern cloud, where tasks can change a lot over time, this kind of adaptability is very important. Organizations can keep up service quality and speed without having to do anything by hand by automatically moving resources up or down based on predicted traffic. In addition, our work shows how AI can help make choices about how to best use resources. These models can improve their guesses and suggestions over time by learning from data trends and feedback loops. This makes resource management methods even more efficient and flexible. Businesses can stay ahead of the competition by using AI-driven resource allocation models. These models help them meet service level agreements (SLAs) more consistently, make customers happier by improving performance, and save money by better using resources.

References

[1] Mukherjee, A.; Goswami, P.; Khan, M.A.; Manman, L.; Yang, L.; Pillai, P. Energy-efficient resource allocation strategy in massive IoT for industrial 6G applications. IEEE Internet Things J. 2020, 8, 5194–5201.

[2] Radhika, S.; Rangarajan, P. Fuzzy based sleep scheduling algorithm with machine learning techniques to en-hance energy efficiency in wireless sensor networks. Wirel. Pers. Commun. 2021, 118, 3025–3044.

[3] Gulganwa, P.; Jain, S. EES-WCA: Energy efficient and secure weighted clustering for WSN using machine learning approach. Int. J. Inf. Technol. 2022, 14, 135–144.
[4] Tyagi SK, S.; Mukherjee, A.; Pokhrel, S.R.; Hiran, K.K. An intelligent and optimal resource allocation ap-proach in sensor networks for smart agri-IoT. IEEE Sens. J. 2020, 21, 17439–17446.
[5] Kaur, G.; Chanak, P.; Bhattacharya, M. Energy-efficient intelligent routing scheme for IoT-enabled WSNs. IEEE Internet Things J. 2021, 8, 11440–11449.
[6] Wang, W.; Guo, H.; Li, X.; Tang, S.; Xia, J.; Lv, Z. Deep learning for assessment of environmental satisfaction using BIM big data in energy efficient building digital twins. Sustain. Energy Technol. Assess. 2022, 50, 101897.
[7] Singh, S.; Singh, P.; Tanwar, S. Energy aware resource allocation via MS-SLnO in cloud data center. Multimed. Tools Appl. 2023, 82, 45541–45563.
[8] Khan, H.; Singh, P. Issues and Challenges of Internet of Things: A Survey. J. Inform. Electr. Electron. Eng. (JIEEE) 2021, 2, 1–8.
[9] Tiwari, A.; Nagaraju, A.; Mahrishi, M. An Optimized Scheduling Algorithm for Cloud Broker Using Cost Adaptive Modeling. In Proceedings of the 3rd IEEE (International Advanced Computing Conference, Ghaziabad, India, 22–23 February 2013; pp. 28–33.
[10] Kaur, R.; Jain, A.; Kumar, S. Optimization classification of sunflower recognition through machine learning. Mater. Today Proc. 2022, 51, 207–211.
[11] Kumar, S.; Kumari, B.; Chawla, H. Security challenges and application for underwater wireless sensor network. In Proceedings of the International Conference on Emerging Trends in Expert Applications & Security, Jaipur, India, 17–18 February 2018; Volume 2, pp. 15–21Tiwari, A.; Sharma, R.M. Rendering Form Ontology Methodology for IoT Services in Cloud Computing. Int. J. Adv. Stud. Sci. Res. 2018, 273–278.
[12] Neelakandan, S.; Berlin, M.A.; Tripathi, S.; Devi, V.B.; Bhardwaj, I.; Arulkumar, N. IoT-Based Traffic Prediction and Traffic Signal Control System for Smart City. Soft Comput. 2021, 25, 12241–12248.
[13] Han, Z.; Yang, Y.; Wang, W.; Zhou, L.; Gadekallu, T.R.; Alazab, M.; Gope, P.; Su, C. RSSI Map-Based Trajectory Design for UGV Against Malicious Radio Source: A Reinforcement Learning Approach. IEEE Trans. Intell. Transp. Syst. 2023, 24, 4641–4650.
[14] Ajani, S. N. ., Khobragade, P. ., Dhone, M. ., Ganguly, B. ., Shelke, N. ., & Parati, N. . (2023). Advancements in Computing: Emerging Trends in Computational Science with Next-Generation Computing. International Journal of Intelligent Systems and Applications in Engineering, 12(7s), 546–559
[15] Swarna Priya, R.M.; Bhattacharya, S.; Maddikunta, P.K.R.; Somayaji, S.R.K.; Lakshmanna, K.; Kaluri, R.; Hussien, A.; Gadekallu, T.R. Load Balancing of Energy Cloud Using Wind Driven and Firefly Algorithms in Internet of Everything. J. Parallel Distrib. Comput. 2020, 142, 16–26.
[16] Kale, Rohini Suhas , Hase, Jayashri , Deshmukh, Shyam , Ajani, Samir N. , Agrawal, Pratik K & Khandelwal, Chhaya Sunil (2024) Ensuring data confidentiality and integrity in edge computing environments : A security and privacy perspective, Journal of Discrete Mathematical Sciences and Cryptography, 27:2-A, 421–430, DOI: 10.47974/JDMSC-1898
[17] Dari, Sukhvinder Singh , Dhabliya, Dharmesh , Dhablia, Anishkumar , Dingankar, Shreyas , Pasha, M. Jahir & Ajani, Samir N. (2024) Securing micro transactions in the Internet of Things with cryptography primitives, Journal of Discrete Mathematical Sciences and Cryptography, 27:2-B, 753–762, DOI: 10.47974/JDMSC-1925
[18] Limkar, Suresh, Singh, Sanjeev, Ashok, Wankhede Vishal, Wadne, Vinod , Phursule, Rajesh & Ajani, Samir N. (2024) Modified elliptic curve cryptography for efficient data protection in wireless sensor network, Journal of Discrete Mathematical Sciences and Cryptography, 27:2-A, 305–316, DOI: 10.47974/JDMSC-1903
[19] Kavitha, T.; Mathai, P.P.; Karthikeyan, C.; Ashok, M.; Kohar, R.; Avanija, J.; Neelakandan, S. Deep Learning Based Capsule Neural Network Model for Breast Cancer Diagnosis Using Mammogram Images. Interdiscip. Sci. Comput. Life Sci. 2021, 14, 113–129.
[20] Neelakandan, S.; Arun, A.; Bhukya, R.R.; Hardas, B.M.; Kumar, T.C.; Ashok, M. An Automated Word Embedding with Parameter Tuned Model for Web Crawling. Intell. Autom. Soft Comput. 2022, 32, 1617–1632.
[21] Rani, S.; Babbar, H.; Srivastava, G.; Gadekallu, T.R.; Dhiman, G. Security Framework for Internet-of-Things-Based Software-Defined Networks Using Blockchain. IEEE Internet Things J. 2023, 10, 6074–6081.

[22] Gupta, P.; Varshney, A.; Khan, M.R.; Ahmed, R.; Shuaib, M.; Alam, S. Unbalanced Credit Card Fraud Detection Data: A Machine Learning-Oriented Comparative Study of Balancing Techniques. Procedia Comput. Sci. 2023, 218, 2575–2584.

[23] Joloudari, J.H.; Mojrian, S.; Saadatfar, H.; Nodehi, I.; Fazl, F.; Khanjani Shirkharkolaie, S.; Alizadehsani, R.; Kabir, H.M.D.; Tan, R.-S.; Acharya, U.R. Resource Allocation Optimization Using Artificial Intelligence Methods in Various Computing Paradigms: A Review. arXiv 2022, arXiv:2203.12315.

[24] Fraga-Lamas, P.; Lopes, S.I.; Fernández-Caramés, T.M. Green IoT and Edge AI as Key Technological Enablers for a Sustainable Digital Transition towards a Smart Circular Economy: An Industry 5. 0 Use Case. Sensors 2021, 21, 5745.

[25] Butt, U.A.; Mehmood, M.; Shah, S.B.H.; Amin, R.; Shaukat, M.W.; Raza, S.M.; Suh, D.Y.; Piran, M.J. A Review of Machine Learning Algorithms for Cloud Computing Security. Electronics 2020, 9, 1379.