Journal of IoT in Social, Mobile, Analytics, and Cloud (ISSN: 2582-1369)
www.irojournals.com/iroismac/

Prediction of Workloads in Cloud using ARIMA-ANN

Suriya S.[1], Surya Arvindh M.[2]

[1]Associate Professor, Department of Computer Science and Engineering, PSG College of Technology, Coimbatore, Tamil Nadu, India.

[2]PG Scholar, Department of Computer Science and Engineering, PSG College of Technology, Coimbatore, Tamil Nadu, India.

E-mail: [1]ss.cse@psgtech.ac.in, suriya84@gmail.com, [2]23mz04@psgtech.ac.in

Abstract

This study introduces an innovative hybrid ARIMA-ANN model personalized for cloud workload prediction. Unlike existing models that focus solely on linear or nonlinear patterns, the approach combines the strengths of ARIMA for time-series linear trends and ANN for nonlinear data complexities. This integration ensures higher accuracy, as validated using the MIT Supercloud dataset. The methodology leverages data pre-processing, sensitivity analysis, and advanced validation techniques, demonstrating improved accuracy in scenarios of high workload variability. This model supports cloud providers in resource optimization and dynamic load management.

Keywords: ARIMA, ANN, Load Balancing, Time Series data, MIT Supercloud

1. Introduction

Cloud computing, a foundation of distributed computing, relies on dynamic resource management to deliver optimal Quality of Service (QoS) and meet Service Level Agreements (SLAs). Infrastructure as a Service (IaaS) providers face challenges managing volatile CPU and memory demands. Accurate workload prediction ensures resource optimization, reducing over-provisioning and cost.

Existing methods, such as ARIMA and ANN, handle specific workload patterns but lack the versatility required for diverse and unpredictable workloads. This study introduces a

DOI: https://doi.org/10.36548/jismac.2024.4.003
Received: 23.11.2024, received in revised form: 17.12.2024, accepted: 04.01.2025, published: 16.01.2025

hybrid ARIMA-ANN model, blending linear forecasting with nonlinear learning capabilities. The novelty lies in addressing complex workload scenarios, validated using a real-world dataset. This approach achieves superior prediction accuracy and robustness, essential for dynamic resource allocation in cloud systems.

2. Related Work

The study focuses on the prediction models for efficient cloud resource provisioning, emphasizing the role of auto-scaling algorithms and comparing QoS parameters between conventional and efficient provisioning methods. It highlights designing prediction mechanisms and utilizing methods for workload determination [1]. In their research study [2], the authors discuss the shift to cloud computing, the rise of edge computing to complement cloud deployments, and challenges in orchestrating edge-cloud applications. It underscores the significance of machine learning in workload characterization, component placement, and application elasticity, classifying algorithms into supervised and unsupervised categories. This study [3] uses a hybrid ARIMA–ANN model to forecast future CPU and memory utilization in cloud resource provisioning. The model detects linear and nonlinear components in cloud traces, while the artificial neural network (ANN) uses residuals. The Savitzky–Golay filter finds a range of forecast values, reducing forecasting error by introducing a range of values. The accuracy of the prediction is tested using statistical analysis using Google's 29-day trail and BitBrain. Cloud computing is gaining popularity, requiring accurate prediction of computing resource usage for efficient management. However, excessive costs can be a concern. The study [4] presents a novel approach that uses data-driven prediction algorithms to generate short- and long-term cloud resource usage predictions. The solution readjusts to different load characteristics and usage changes. Preliminary tests showed 36% better prediction quality, and real-life historical data showed 9.28% to 80.68% better prediction quality. This research study [5] is a systematic survey and comparative study of machine learning-driven cloud workload prediction models. It discusses the importance of predictive resource management, operational design, motivation, and challenges. The study classifies different prediction approaches into five categories, focusing on theoretical concepts and mathematical functioning. The proposed method in [6] presents COSCO2, a workload prediction framework optimized with a sheep flock optimization algorithm. It accurately

predicts workloads, reducing energy consumption and outperforming existing methods for datasets like NASA and Saskatchewan HTTP traces. The study [7] proposes an autonomic prediction suite to improve the accuracy of cloud computing's auto-scaling system. It suggests that selecting the right time-series prediction algorithm based on the incoming workload pattern can increase the prediction accuracy. The research conducts theoretical investigations and empirical validations, designing a self-adaptive prediction suite that automatically chooses the most suitable algorithm. The research study [8] discusses designing cloud client prediction models using machine learning, identifying Support Vector Regression (SVR) as the best model for non-linear workload patterns. The study [9] addresses challenges in volatile resource demands and proposes a meta-algorithm for algorithm selection based on past performance, with insights into dynamic regret and optimal solutions. The method proposed in [10] introduces a DCRNN, a deep learning model for accurate workload prediction, to improve forecasting accuracy and minimize the error between the predicted and the actual workloads. The method put forth in [11] describes a framework for provisioning virtual machines using Kalman filter-based data preprocessing, enhancing service quality by reducing provisioning latency. Experimental results show that the framework reduces latency and improves cloud service quality, using Alicloud as an experimental infrastructure. The study presents RPPS, a Cloud Resource Prediction and Provisioning scheme, which predicts future demand and performs proactive resource provisioning for cloud applications. It uses the ARIMA model, combines coarse-grained and fine-grained resource scaling, and adopts a VM-complementary migration strategy. The prototype has high prediction accuracy and good resource scaling, making it a valuable solution for enterprises facing demand fluctuations in cloud data centers [12]. The research [13] presents a machine learning-based solution for cloud resource optimization, using anomaly detection to reduce costs while maintaining QoS. It emphasizes the importance of initial training and future research in model reuse without system duplication. The study [14] focuses on QoS-based resource provisioning to optimize allocation, reduce execution time and costs, and improve resource scheduling. It emphasizes the role of workload analysis in understanding QoS requirements. The method put forth in [15] presents the RRAU approach for resource management, improving metrics such as Job Completion Time (JCT) and monetary cost compared to conventional methods. It emphasizes optimizing VM runtimes and asset utilization. The researchers in [16] propose a capacity planning approach utilizing hybrid spot instances for improved utilization and reliability. It enhances throughput during out-of-bid situations and suggests future integration of reactive modules for better QoS impact

analysis. The study in [17] introduces a container-based video surveillance cloud platform with predictive resource allocation, optimizing service density and cost prediction. It demonstrates the effectiveness of Docker technology over VM-based platforms in supporting microservices like video on demand. This article [18] presents a method for elastic resource provisioning using date clustering in cloud service platforms. The method consists of tasks clustering, task prediction, dynamic resource provisioning, and scheduling. It partitions tasks based on similarity and forecasts future tasks. The method achieves high guarantees, resource utilization, and total energy consumption in the Google cloud traces dataset. The study [19] addresses monitoring challenges in hybrid clouds, proposing a solution that automates management across hybrid environments. The study [20] introduces PCA-BPN for estimating simulation workloads in cloud manufacturing, achieving superior accuracy in execution time prediction, especially in factory simulation contexts.

3. Comparison of Various Computing Types

3.1 Centralized Computing

Centralized computing relies on a central server for all computations and data storage. This model is simple to implement and administer, with centralized management and security, but it struggles with scalability and is prone to single points of failure. Examples include traditional mainframe systems.

3.2 Distributed Computing

Distributed computing involves interconnected nodes working collaboratively. It offers scalability and reliability with fault tolerance and optimized resource usage. However, performance can be limited by network bandwidth. Examples include peer-to-peer networks and Hadoop.

3.3 Grid Computing

Grid computing coordinates geographically dispersed resources to function as a unified system. It allows scalability and redundancy but may face latency and bandwidth challenges. Control is typically centralized, with shared resource utilization. Examples include SETI@home and the Globus Toolkit.

3.4 Cluster Computing

Cluster computing connects multiple nodes to work together, providing fault tolerance and scalability by adding nodes. Management can be centralized or distributed, with shared resources tailored for specific tasks. Examples include HPC and Beowulf clusters.

3.5 Cloud Computing

Cloud computing provides elastic, on-demand resource sharing through virtualized environments. Resources are distributed but centrally managed, with data stored across multiple regions. Prominent platforms include AWS, Google Cloud Platform, and Microsoft Azure.

4. Types of Workload

4.1 Static Workload

Static workloads involve a fixed number of requests, with constant memory, processor capacity, and bandwidth. These workloads follow consistent usage patterns and do not scale dynamically. Examples include web servers and email services. Figure 1 represents the pattern of static workload.

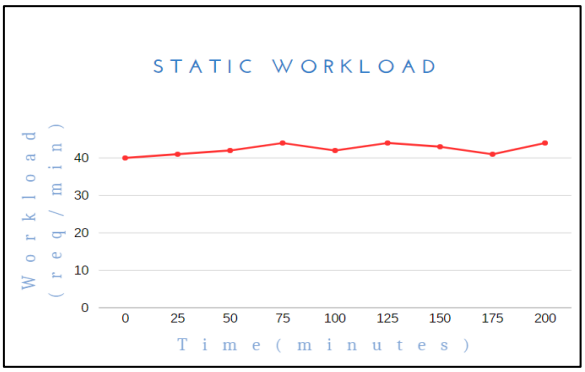

Figure 1. Static Workload Pattern

4.2 Periodic Workload

Periodic workloads involve recurring cycles, such as daily, weekly, or seasonal peaks. They require scalability to handle peak loads efficiently while avoiding resource

underutilization during off-peak periods. These workloads demand flexible resource allocation. Figure 2 represents the periodic workload pattern.

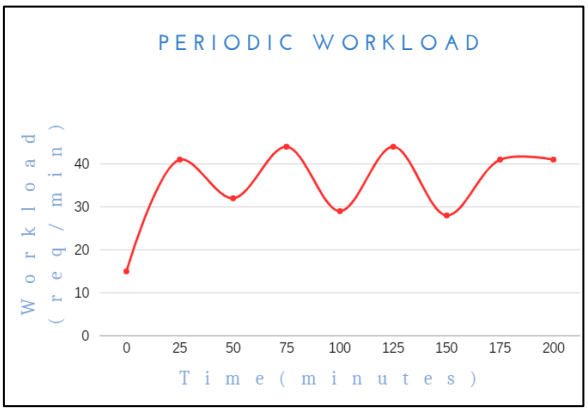

Figure 2. Periodic Workload Pattern

4.3 Unpredictable Workload

Unpredictable workloads are dynamic and lack regular patterns, making resource planning challenging. These workloads experience random surges due to factors like unexpected traffic or environmental conditions. Cloud providers must address variability through dynamic resource allocation. Figure 3 depicts the unpredictable workload pattern

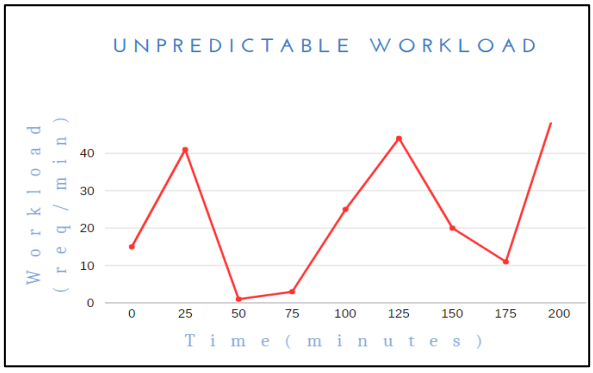

Figure 3. Unpredictable Workload Pattern

4.4 Continuously Changing Workload

Continuously changing workloads exhibit gradual increases or decreases in resource demand over time. Resource allocation adjusts dynamically to align with the evolving scope and scale of tasks. Figure 4 represents continuously changing workload pattern.

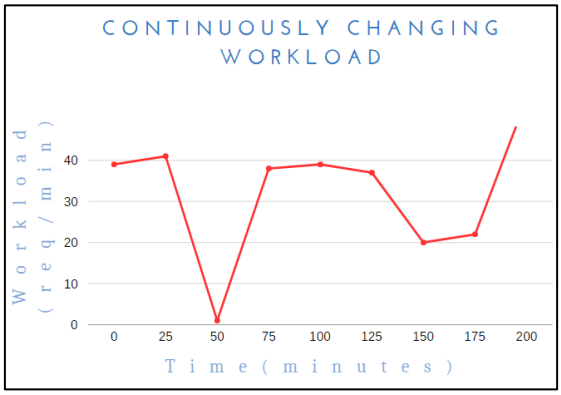

Figure 4. Continuously Changing Workload Pattern

4.5 Once in a Lifetime Workload

This workload represents peak utilization events unlikely to recur, requiring manual provisioning or decommissioning of resources. Such workloads often involve one-time high resource consumption. Figure 5 represents the Once in a Lifetime Workload pattern

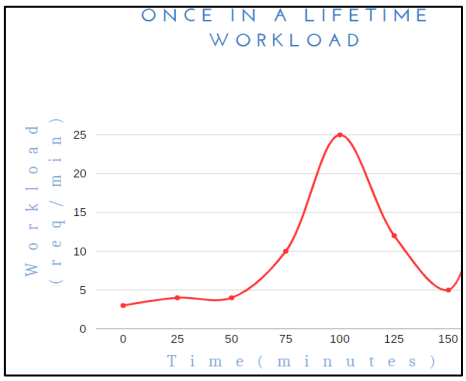

Figure 5. Once in a Lifetime Workload Pattern

5. Proposed Methodology

5.1 Time series forecasting Models

Time series can be defined with data for y(t) it's a series of dispersed data values taken at successive time intervals and t is time and time is a variable that increases continuously from zero to one and one to two in equal intervals. It is a manner that they arrange the observations right from the observation that is observed most to the observation observed least. As usual,

the single variable has a time series while more than one variable is encoded in the multivariate time series. However, time series could be of two types: and these are: Continuous time series and Discrete time series. When the time series is continuous, this is because the observations the events occur at a continuous time point, while for the data recorded on discrete time series, the observations are taken at equal time steps. It has four components: These are seasonal working shifts, regular working shifts, at random working shift and periodic working shifts. On the basis of classification of numerous time series, economists divided them into four sub-series. They are fashionable, oscillatory, periodic, and non-periodic and are feasible from the observed facts. With reference to the impacts of the four elements, as well as the multiplicative and the additive models, use the time series computations [3].

Multiplicative Model

$$Y(t) = T(t) * S(t) * C(t) * I(t)$$

Additive Model

$$Y(t) = T(t) + S(t) + C(t) + I(t)$$

T(t)-Trend Component

S(t) Seasonal Component

C(t) Cyclic Component

I(t)-Irregular component

5.2 ARIMA – Autoregressive Integrated Moving Average

Autoregression represents a concept suggesting that a variable that varies is regressed on preceding or lag values. Integrated (I) refers to the level of the simple transformation of an immature form of data in the time series in the smoothing process that aims at replacing each of the value of time series with the variation of the data value from the preceding data value. MA employs the stochastic component of an observation in a moving average model fitted to lag observations.

The parameters of this model are:

P: the number of lag observations in the model more precisely identified as the lag order.

D: the number of times the raw observations are differenced; also the degree of differencing.

Q: the second parameter that could be changed is the size of the moving average window or simply the order of the moving average.

$$y\,(t) = \theta_0 + \phi_1 y_{t-1} + \phi_2 y_{t-2} \ldots + \phi_p\, y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} \ldots + \theta q\, \varepsilon_{t-q}$$

y_t and ε_t are the actual value and the random error at time period t respectively, $\phi_i = 1, 2, \ldots p)$ and $\theta_i = 1,2, \ldots q)$ are the model parameters [3].

5.3 ANN

The ANN structure is very similar to the neuron structure of the human brain. ANNs approximate the various nonlinearities in the data. To model the time series data, the relationship between the output (y_t) and the inputs ($y_{t-1}, y_{t-2}, \ldots\ y_{t-p}$) is expressed as

$$y_t = W_0 + \sum_{j=1}^{q} w_j . g(w_{0j} + \sum_{i=1}^{p} w_{ij} . y_{t-i}) + \varepsilon_t$$

w_{ij} $(i = 0, 1, 2, \ldots p; j = 1, 2, \ldots q)$ represents the model parameters also called the connection weights; p is the number of input nodes and q is the number of hidden nodes and g is the transfer function of the hidden layer. Indeed, the ANN model can be considered almost a nonlinear autoregressive model. This model's coefficient is then computed from the input of the ANN where the latter is fed with the observed data sequence and the latter trained with this sample sequence. The performance is checked on a sample after the training process is over [3].

5.4 Data Collection and Preprocessing

Dataset: MIT Supercloud Dataset

Source and Samples: Kaggle - 96,893 entries with 23 columns.

Features: Memory usage, and GPU memory utilization.

Preprocessing:

- Empty values are replaced by NaN and then NaN value elimination.
- Time-series transformation for sequential analysis.
- Scaling and normalization for feature consistency.

5.5 Model Training and Optimization

- All data is scaled to a range between 0 and 1 using MinMaxScaler. This is essential for neural networks to ensure uniform weight updates during training.
- ANN uses lagging to transform the time-series data into a supervised learning problem:
  - X (Features): The data at time t.
  - y (Targets): The data at time t+1.
- Loss function: Mean Squared Error (MSE).
- Optimizer: Adam optimizer with learning rate 0.05. Adam (Adaptive Moment Estimation) combines the benefits of momentum and RMSprop optimizers. It dynamically adjusts learning rates for each parameter, making it efficient for non-stationary objectives.
- Epochs: 100.
- Batch size: 32.

5.6 Evaluation Metrics and Tools

The code evaluates the models using metrics and tools specific to the methods applied. For the ANN model, Mean Squared Error (MSE) is used to measure the average squared difference between predicted and actual values, computed using sklearn.metrics.mean_squared_error, with lower MSE indicating better performance. For the ARIMA model, evaluation relies on statistical summaries provided by statsmodels.tsa.arima.model. ARIMA, which assess model fit and complexity, includes metrics like Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), and Log-Likelihood. Tools like sklearn.neural_network.MLPRegressor handles ANN training,

while sklearn.preprocessing. MinMaxScaler scales features to improve model convergence. Visualization is performed using Matplotlib to plot actual vs. predicted values for ANN. Additional metrics such as Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), or R^2 Score could enhance the analysis further, providing more comprehensive insights into model performance.

6. Architecture

The ARIMA-ANN model, to predict the area of confidence values for CPU and memory utilization is presented in Figure. 6. The measurement for the analysis was done using the Google cluster data collection. Note that pre-processing of time series data requires null elimination. Finally, the time series is passed to the ARIMA model, where the implementation of predictions of CPU and memory usage of each request for the ensuing N requests is executed using the data. ARIMA cannot model the residues which are described as the nonlinear components of the model. The residuals, which were collected, are fed along with the upcoming data into the network model. It constructs the model of the prognosis of CPU and memory demands.

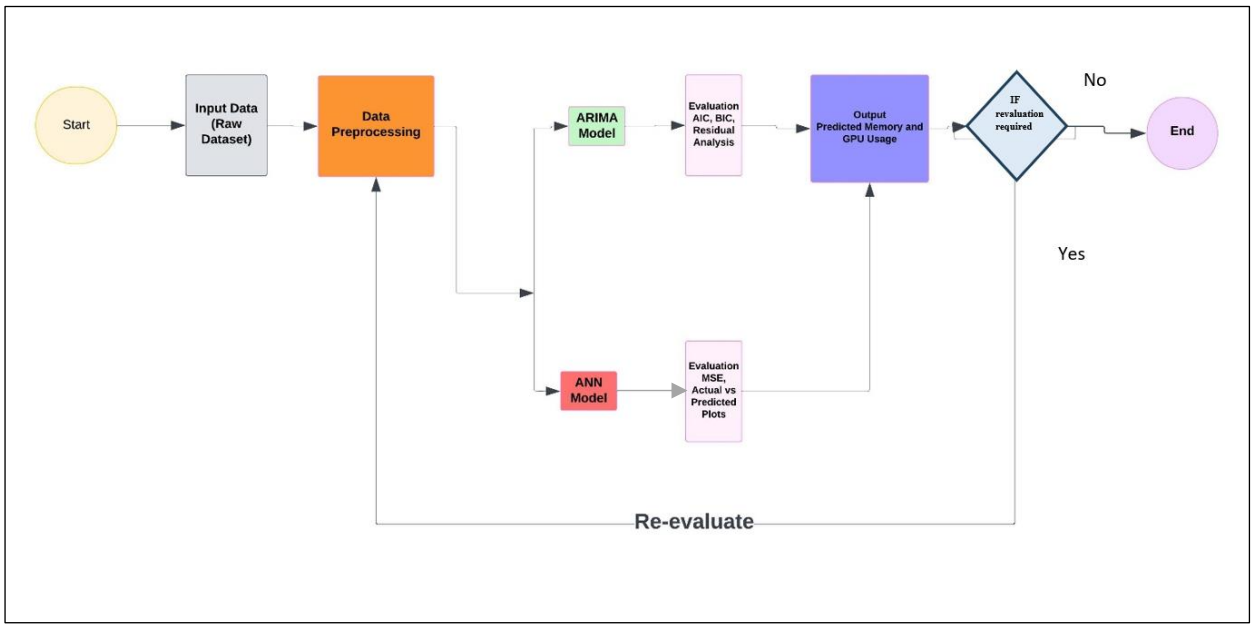

Figure 6. Architecture of Hybrid ARIMA – ANN Model [3]

The final values of the prediction are achieved from the sum of the values derived from both models of prediction. Therefore, the time series plots are derived from taking values from these two times and overlaid with the other times series. It appears that recent data, in fact, especially tempting for the knowledge of the past than other special kind of data from the distant past are. The error bars for the n-step ahead CPU and memory use estimation are located on the data which were smoothened.

7. Experimental Setup

The workload traces provide detailed information on jobs and tasks, including their CPU and memory usage, over one month. The dataset consists of timestamped entries (microseconds as 64-bit integers) and other resource utilization metrics, which allow the creation of time-series data for analysis. In the implementation, memory utilization (`avgmemoryutilization_pct`) and maximum GPU memory usage (`maxgpumemoryused_bytes`) are the primary focus. Data preprocessing involves replacing zero values with NaN and dropping rows with missing values. For modeling, an Artificial Neural Network (ANN) and ARIMA model are used. The ANN is trained and validated using 70% and 30% of the data, respectively, and Min-Max scaling is applied for feature normalization. For n-step predictions, the model processes lagged features, with evaluation based on Mean Squared Error (MSE). The ARIMA model captures temporal trends and autocorrelations, with its performance assessed via statistical summaries. The implementation and training are carried out in Python, providing a robust framework for GPU workload prediction**.** The sample of data used is depicted in Figure 7**.**

avgmemory	avgmutiliza	energyconsumed	gpu_id	maxgpumemoryused_b	memoryutili	memoryutili	memoryutili	pcierxbandwidt	pcierxbandwidt	pcierxbandwidt	pcietxbandwidt	pcietxbandwidt	pcietxbandwidt	powerusage_watts	powerusage_watts	powerusage_watts	smutilization_	smutilization_	smutilization_	totalexecutiontime	id_job
0	1	157521	0	2645557248	3	24	0	1646	1748	1613	971	1530	676	44.0546	156.962	25.468	11	64	0	4452.47	4393785332
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.54	1817701425
0	3	94816	0	29171712	0	0	0	1846	1849	1838	1785	1872	1346	26.3725	27.483	25.979	6	43	0	8398.3	3952468078
0	0	69451	0	1527586816	0	1	0	539	542	536	314	337	293	37.5481	52.039	26.676	1	25	0	1851.72	1999833952
0	0	0	1	0	0	0	0	1991	1991	1991	1109	1109	1109	25.032	25.032	25.032	0	0	0	0.99	397881706
0	0	22587	0	0	0	0	0	1740	1740	1739	1722	1722	1721	27.622	27.821	27.532	0	0	0	819.64	56459319806
0	3	196461	1	1143996416	4	5	0	955	1683	206	1087	2147	0	54.6094	61.091	38.101	88	97	0	74636.6	13878440310
0	2	46532	1	2645557248	1	14	0	199	205	191	1674	1701	1649	38.6441	126.934	24.203	3	39	0	1200.81	3247893276
0	0	6706	0	1253048320	0	0	0	2130	2131	2129	1991	2015	1968	26.1606	36.257	25.42	4	9	0	257.37	3113016482
0	0	90161	0	6889472	0	0	0	1695	1697	1693	1361	1362	1359	25.0819	25.412	24.761	0	0	0	34740.8	14306435439
0	9	11385	1	1366294528	0	3	0	1105	1108	1101	210	210	209	40.1786	51.193	26.256	9	32	0	284.02	7641411333
0	9	197442	0	1143996416	4	5	0	1004	2147	0	700	2147	0	54.8724	60.57	37.849	93	99	0	68662.4	41584639356
0	0	36302	1	0	0	0	0	1069	1069	1068	1311	1311	1310	25.527	25.697	25.392	0	0	0	1424.6	34850826363
0	0	7075	1	0	0	0	0	31	31	31	1486	1487	1486	23.8464	23.964	23.753	0	0	0	298.18	51916549454
0	0	6992	0	1055916032	0	0	0	1931	1932	1930	1829	1850	1802	27.2866	41.168	26.229	4	10	0	257.06	33697999212
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.59	55294358298
0	0	0	0	0	0	0	0	21	21	21	404	404	404	27.449	27.449	27.449	0	0	0	1.07	55403326389
0	0	686	0	0	0	0	0	1838	1838	1838	1801	1801	1801	27.4212	27.454	27.346	0	0	0	26	7228147637
0	0	141189	1	2658140160	1	14	0	172	191	152	1566	1649	1478	39.1102	128.177	36.453	3	39	0	24430.8	17697620496
0	0	3156	1	1135607808	0	5	0	1700	1713	1686	1336	1338	1334	39.9009	48.445	27.167	11	96	0	80.17	17780987506
0	0	303	1	0	0	0	0	101	102	102	1862	1862	1862	27.5371	27.639	27.49	0	0	0	12.32	49127934589
0	0	92380	1	0	0	0	0	438	440	436	1286	1288	1284	25.6923	26.082	25.386	0	0	0	3706.21	29809496593
0	0	91499	0	0	0	0	0	1454	1456	1452	660	662	659	25.4521	25.611	25.298	0	0	0	86749.1	7490639208
0	0	68099	0	1527586816	0	1	0	1930	1933	1927	1746	1769	1735	36.8209	50.128	28.208	1	10	0	1851.77	3311219662
0	0	0	0	0	0	0	0	1212	1212	1212	1023	1023	1023	26.295	26.357	26.233	0	0	0	1.64	53504729747
0	0	95793	0	6889472	0	0	0	669	671	667	1283	1285	1281	26.6449	27.034	26.21	0	0	0	15078	27380150765
0	0	192	0	0	0	0	0	2053	2053	2053	1432	1432	1432	27.387	27.488	27.313	0	0	0	7.9	62476106594
0	0	555	0	0	0	0	0	682	682	682	1035	1035	1035	26.3967	26.534	26.334	0	0	0	22.22	7326405924
0	0	483538	0	2658140160	32	38	0	1434	2147	0	1312	2147	0	134.787	161.908	41.649	89	100	0	50715.1	29891429222
0	0	0	1	0	0	0	0	0	0	0	25	25	25	0	0	0	0	0	0	0.56	5108535984
0	0	40090	0	0	0	0	0	1746	1746	1745	1727	1727	1726	27.5645	27.743	27.436	0	0	0	1456.4	32404091771
0	0	944	1	0	0	0	0	1615	1615	1615	25	25	24	24.1201	24.192	24.084	0	0	0	40.34	2261495240
0	2	40555	0	1389363200	0	3	0	126	138	112	766	768	764	33.279	50.084	24.7	7	33	0	1219.71	1278009669

Figure 7 Sample Data

8. Results and Evaluation

8.1 Prediction of Memory Utilization

The summary of the model used in this implementation is given in Figure 8. The Memory Utilization actual vs predicted plot is given in Figure 9.

```
Fitting ARIMA for avgmemoryutilization_pct...
                               SARIMAX Results
==============================================================================
Dep. Variable:     avgmemoryutilization_pct   No. Observations:                42190
Model:                     ARIMA(1, 1, 1)   Log Likelihood             -162314.629
Date:                    Sat, 11 Jan 2025   AIC                         324635.259
Time:                            12:41:46   BIC                         324661.208
Sample:                                 0   HQIC                        324643.452
                                    - 42190
Covariance Type:                      opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.1132      0.003     35.887      0.000       0.107       0.119
ma.L1         -0.9532      0.001   -921.211      0.000      -0.955      -0.951
sigma2       128.5996      0.445    289.231      0.000     127.728     129.471
===================================================================================
Ljung-Box (L1) (Q):                   0.02   Jarque-Bera (JB):             76615.74
Prob(Q):                              0.90   Prob(JB):                         0.00
Heteroskedasticity (H):               2.10   Skew:                             1.36
Prob(H) (two-sided):                  0.00   Kurtosis:                         9.02
===================================================================================
```

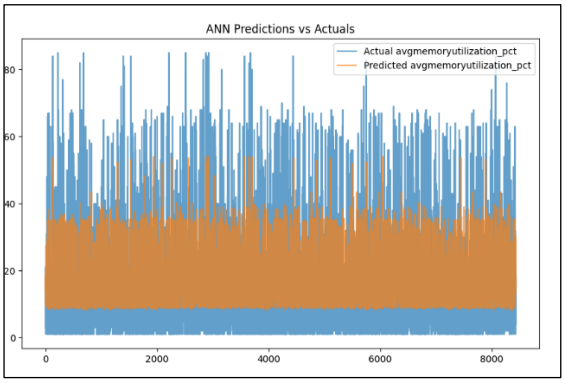

Figure 8 Model Summary

Figure 9 Memory Actual vs Predicted

The ANN mean square error between actual and predicted values are calculated and it was approximately 0.04122451258775828

8.2 Prediction of Memory

The summary of the model used in this implementation is given in Figure 10. The GPU actual vs predicted plot is given in Figure 11.

```
Fitting ARIMA for maxgpumemoryused_bytes...
                               SARIMAX Results
==============================================================================
Dep. Variable:     maxgpumemoryused_bytes   No. Observations:                42190
Model:                     ARIMA(1, 1, 1)   Log Likelihood            -1018640.537
Date:                    Sat, 11 Jan 2025   AIC                         2037287.073
Time:                            12:41:54   BIC                         2037313.023
Sample:                                 0   HQIC                        2037295.267
                                    - 42190
Covariance Type:                      opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.1258      0.004     34.324      0.000       0.119       0.133
ma.L1         -0.9285      0.002   -580.079      0.000      -0.932      -0.925
sigma2      6.252e+19        nan        nan        nan         nan         nan
===================================================================================
Ljung-Box (L1) (Q):                   0.06   Jarque-Bera (JB):             22363.21
Prob(Q):                              0.81   Prob(JB):                         0.00
Heteroskedasticity (H):               4.72   Skew:                             0.57
Prob(H) (two-sided):                  0.00   Kurtosis:                         6.38
===================================================================================
```

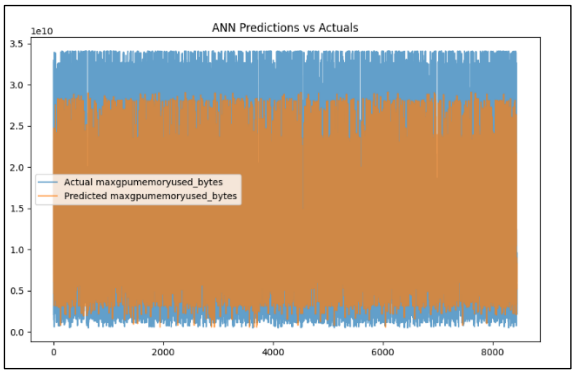

Figure 10. Model Summary

Figure 11. GPU Actual vs Predicted

9. Conclusion

However, to provide high QoS with optimum resource usage in clouds, it is imperative to predict workloads correctly. The workload characteristics of the hybrid model, which include linear and non-linear, afford cloud providers the tools necessary to manage inventories a priori and increase preparedness for varying workloads and demand. However, in case of any change in the workload pattern, it would be necessary to bring the model into the parameter estimation process to keep an accurate forecast. This open and flexible strategy also reduces the dangers of under-providing services, including service downtime, high energy expenses, and client loss. By using dynamic sliding windows and giving the latest data a higher weight, predictions are enhanced, supplying providers of cloud services with a powerful instrument to regulate resources for dynamic workloads.

References

[1] Gadhavi, L. J., and M. D. Bhavsar. "Prediction-Based Efficient Resource Provisioning and Its Impact on QoS Parameters in the Cloud Environment." International Journal of Electrical and Computer Engineering (IJECE) 8, no. 6 (2018): 5359–5370.

[2] Duc, T. L., R. G. Leiva, P. Casari, and P. O. Östberg. "Machine Learning Methods for Reliable Resource Provisioning in Edge-Cloud Computing: A Survey." ACM Computing Surveys (CSUR) 52, no. 5 (2019): 1–39.

[3] Devi, K. L., and S. Valli. "Time Series-Based Workload Prediction Using the Statistical Hybrid Model for the Cloud Environment." Computing 105, no. 2 (2023): 353–374.

[4] Nawrocki, P., P. Osypanka, and B. Posluszny. "Data-Driven Adaptive Prediction of Cloud Resource Usage." Journal of Grid Computing 21, no. 1 (2023): 6.

[5] Saxena, D., J. Kumar, A. K. Singh, and S. Schmid. "Performance Analysis of Machine Learning Centered Workload Prediction Models for Cloud." IEEE Transactions on Parallel and Distributed Systems 34, no. 4 (2023): 1313–1330.

[6] Karthikeyan, R., V. Balamurugan, R. Cyriac, and B. Sundaravadivazhagan. "COSCO2: AI-Augmented Evolutionary Algorithm Based Workload Prediction Framework for

Sustainable Cloud Data Centers." Transactions on Emerging Telecommunications Technologies 34, no. 1 (2023): e4652.

[7] Nikravesh, A. Y., S. A. Ajila, and C. H. Lung. "An Autonomic Prediction Suite for Cloud Resource Provisioning." Journal of Cloud Computing 6 (2017): 1–20.

[8] Bankole, A. A., and S. A. Ajila. "Cloud Client Prediction Models for Cloud Resource Provisioning in a Multitier Web Application Environment." In 2013 IEEE Seventh International Symposium on Service-Oriented System Engineering,USA 156–161. IEEE, 2013.

[9] Comden, J., S. Yao, N. Chen, H. Xing, and Z. Liu. "Online Optimization in Cloud Resource Provisioning: Predictions, Regrets, and Algorithms." Proceedings of the ACM on Measurement and Analysis of Computing Systems 3, no. 1 (2019): 1–30.

[10] Al-Asaly, M. S., M. A. Bencherif, A. Alsanad, and M. M. Hassan. "A Deep Learning-Based Resource Usage Prediction Model for Resource Provisioning in an Autonomic Cloud Computing Environment." Neural Computing and Applications 34, no. 13 (2022): 10211–10228.

[11] Hu, Y., B. Deng, and F. Peng. "Autoscaling Prediction Models for Cloud Resource Provisioning." In 2016 2nd IEEE International Conference on Computer and Communications (ICCC),United States 1364–1369. IEEE, 2016.

[12] Fang, W., Z. Lu, J. Wu, and Z. Cao. "RPPS: A Novel Resource Prediction and Provisioning Scheme in Cloud Data Center." In 2012 IEEE Ninth International Conference on Services Computing,Honolulu, HI, USA 609–616. IEEE, 2012.

[13] Nawrocki, P., and P. Osypanka. "Cloud Resource Demand Prediction Using Machine Learning in the Context of QoS Parameters." Journal of Grid Computing 19, no. 2 (2021): 20.

[14] Singh, S., and I. Chana. "Q-Aware: Quality of Service Based Cloud Resource Provisioning." Computers & Electrical Engineering 47 (2015): 138–160.

[15] Baldoss, P., and G. Thangavel. "Optimal Resource Allocation and Quality of Service Prediction in Cloud." Computers, Materials & Continua 67, no. 1 (2021).

[16] Sadashiv, N., S. D. Kumar, and R. S. Goudar. "Cloud Capacity Planning and HSI Based Optimal Resource Provisioning." In 2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT),Coimbatore, India 1–6. IEEE, 2017.

[17] Zhang, H., H. Ma, G. Fu, X. Yang, Z. Jiang, and Y. Gao. "Container Based Video Surveillance Cloud Service with Fine-Grained Resource Provisioning." In 2016 IEEE 9th International Conference on Cloud Computing (CLOUD),San Francisco, CA, USA 758–765. IEEE, 2016.

[18] Fei, B., X. Zhu, D. Liu, J. Chen, W. Bao, and L. Liu. "Elastic Resource Provisioning Using Data Clustering in Cloud Service Platform." IEEE Transactions on Services Computing 15, no. 3 (2020): 1578–1591.

[19] Naik, V. K., K. Beaty, N. Vogl, and J. Sanchez. "Workload Monitoring in Hybrid Clouds." In 2013 IEEE Sixth International Conference on Cloud Computing, 816–822.Santa Clara, CA, USA IEEE, 2013.

[20] Chen, T. "A PCA-BPN Approach for Estimating Simulation Workload in Cloud Manufacturing." In 2015 Seventh International Conference on Ubiquitous and Future Networks, Sapporo, Japan 826–830. IEEE, 2015.