

Article

MambaNet0: Mamba-Based Sustainable Cloud Resource Prediction Framework Towards Net Zero Goals

Thananont Chevaphatrakul , Han Wang  and Sukhpal Singh Gill * 

School of Electronic Engineering and Computer Science, Queen Mary University of London, London E1 4NS, UK; t.chevaphatrakul@se24.qmul.ac.uk (T.C.); hanwang@qmul.ac.uk (H.W.)

* Correspondence: s.s.gill@qmul.ac.uk

Abstract

With the ever-growing reliance on cloud computing, efficient resource allocation is crucial for maximising the effective use of provisioned resources from cloud service providers. Proactive resource management is therefore critical for minimising costs and striving for net zero emission goals. One of the most promising methods involves the use of Artificial Intelligence (AI) techniques to analyse and predict resource demand, such as cloud CPU utilisation. This paper presents MambaNet0, a Mamba-based cloud resource prediction framework. The model is implemented on Google's Vertex AI workbench and uses the real-world Bitbrains Grid Workload Archive-T-12 dataset, which contains the resource usage metrics of 1750 virtual machines. The Mamba model's performance is then evaluated against established baseline models, including Autoregressive Integrated Moving Average (ARIMA), Long Short-Term Memory (LSTM), and Amazon Chronos, to demonstrate its potential for accurate prediction of CPU utilisation. The MambaNet0 model achieved a 29% improvement in Symmetric Mean Absolute Percentage Error (SMAPE) compared to the best-performing baseline Amazon Chronos. These findings reinforce the Mamba model's ability to forecast accurate CPU utilisation, highlighting its potential for optimising cloud resource allocation in contribution to net zero goals.

Keywords: cloud computing; Artificial Intelligence; time-series forecasting; Mamba; net zero emissions



Academic Editor: Paolo Bellavista

Received: 30 August 2025

Revised: 16 October 2025

Accepted: 17 October 2025

Published: 21 October 2025

Citation: Chevaphatrakul, T.; Wang, H.; Gill, S.S. MambaNet0: Mamba-Based Sustainable Cloud Resource Prediction Framework Towards Net Zero Goals. *Future Internet* **2025**, *17*, 480. <https://doi.org/10.3390/fi17100480>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Cloud computing has become an integral component of the information technology landscape, providing on-demand access to virtually infinite computing power to businesses and individuals alike [1]. Its flexible and scalable nature makes it the perfect foundation for a modern digital infrastructure. The paradigm has driven innovation, becoming a crucial part in the development of multiple fields such as Artificial Intelligence (AI), Big Data, and the Internet of Things (IoT) [2]. As a result, its widespread adoption has transformed traditional Information Technology (IT) practices and reshaped how companies of all sizes operate. However, this transformation has brought to the forefront new challenges that must be addressed to ensure the long-term sustainability and efficiency of this essential technology [3]. The volatile nature of cloud workloads makes it difficult to estimate the required resources, often leading to a mismatch between the provisioned resources and their actual usage [4]. These underutilised servers consume significant amounts of energy, resulting in unnecessary financial costs and carbon emissions [5].

1.1. Motivation

As the adoption of cloud computing continues to experience exponential growth worldwide, it becomes increasingly difficult to ensure optimal utilisation of provisioned resources in the cloud [6]. With constant flux in the demand for computational resources at any given time, it is difficult to accurately gauge the specifications needed. This inability to perfectly predict the demands runs the risk of either under-provisioning or over-provisioning resources. Under-provisioning can potentially lead to a drop in application quality or complete service failure. Conversely, over-provisioning results in unnecessary operational costs and poses significant environmental concerns. The massive adoption of cloud computing has led to data centres taking up to 2% of global electricity consumption, in turn, making them responsible for massive amounts of carbon emissions [7]. The practice of over-provisioning resources exacerbates this, as unutilised computational power represents an unnecessary waste of both cost and energy [8]. Although cloud service providers offer dynamic provision scaling, it is a rather reactive approach that relies on a threshold being exceeded before additional resources are requested. Addressing this inefficiency is a priority, as many nations, such as the United Kingdom, have committed to achieving net zero emission goals within the next few decades [9].

Nowadays, AI has become a pivotal force driven by breakthroughs in Generative AI (GenAI) and related digital technologies, especially Large Language Models (LLMs), and has also seen a massive surge in popularity for different domains such as computing, education, healthcare, finance, transportation, Natural Language Processing (NLP), and retail [10–12]. The growing demand for training and operating these complex AI models has made them enormous consumers of cloud resources [13]. While AI contributes a significant part of cloud energy consumption and resource waste, its predictive capabilities offer a promising solution to these problems [14]. By accurately forecasting resource usage, AI models can proactively scale resource provisioning in real time, minimising both operational costs and environmental impact [15,16]. This has led to multiple studies focusing on predicting CPU utilisation as the main feature to reduce inefficiency, given that CPU is a significant energy consumer in cloud servers [17].

1.2. Main Contributions

The main objective of this paper is to address inefficiencies in cloud resource management by proposing an AI framework that delivers accurate predictions of cloud resource usage, which would result in reduced energy waste and carbon emissions. Therefore, this paper proposes **MambaNet0**, a novel framework for CPU utilisation time-series forecasting. The proposed system leverages the Mamba State Space Model (SSM), an architecture designed to efficiently handle long sequential data. The main contributions of this paper are

- To propose the new MambaNet0 framework that utilises the Mamba model to forecast cloud resource usage, optimising its efficiency to reduce waste of energy.
- To deploy and validate the framework on the Google Cloud's Vertex AI environment.
- To evaluate the performance of the proposed framework using error metrics against other baseline models.

2. Related Work

This section explores relevant works on the topic of cloud resource management optimisation through AI-based time-series forecasting.

Due to the inefficient nature of resource provisioning in the cloud, there has been a multitude of studies on improving the predictive accuracy of AI models to forecast resource demands. Past studies in time-series forecasting for cloud resource management have used models such as Autoregressive Integrated Moving Average (ARIMA), Recurrent Neural

Network (RNN), and Long Short-Term Memory (LSTM), with more recent studies exploring pre-trained time-series models like Amazon Chronos [18]. These works serve as a baseline to evaluate the performance of the proposed framework of this paper. Table 1, presented below, highlights the differences between the proposed model and previous works.

Table 1. Comparison of MambaNet0 with existing AI-driven time-series forecasting approaches.

Work	Model	Environment	Metrics	Dataset
Calheiros et al. [19]	ARIMA	CloudSim	RMSD, NRMSD, MAD, MAPE	Wikipedia Foundation
Dunggan et al. [20]	RNN	CloudSim	MAE, MSE	PlanetLab
Tang et al. [21]	LSTM	Kubernetes	RMSE	Web and database service load trace
Velu et al. [6]	DeepAR	Google Colab	MAE, MSE, MAPE	Bitbrains
Wang et al. [18]	Amazon Chronos	Amazon SageMaker	MAE, MSE, MAPE	Bitbrains
MambaNet0 (This Paper)	Mamba	Vertex AI	MAE, RMSE, SMAPE	Bitbrains

Abbreviations: RMSD: Root Mean Squared Deviation; NRMSD: Normalised Root Mean Square Deviation; MAD: Mean Absolute Deviation; MAPE: Mean Absolute Percentage Error; MAE: Mean Absolute Error; MSE: Mean Squared Error; RMSE: Root Mean Squared Error; SMAPE: Symmetric Mean Absolute Percentage Error.

Calheiros et al. [19] utilised the traditional ARIMA model to create a cloud workload prediction module. The study focuses on achieving good Quality of Service (QoS) for cloud-based applications with proactive provisioning of cloud resources, while maintaining cost-effectiveness. This was achieved by having the ARIMA model predict the estimated resource required by cloud applications and allocate it accordingly. An average accuracy of 91% was achieved while preserving QoS according to established standards [19]. However, due to the ARIMA model being a linear model, it is limited in its ability to capture non-linear dependencies and unexpected spikes that are ever-present in highly volatile workloads found in everyday cloud servers. Additionally, due to the study's reactive approach, it can only respond to fluctuations after they occur, potentially degrading QoS.

Dunggan et al. [20] investigated the ability of RNN models to provide accurate short-term predictions for CPU utilisation. The paper states that AI and Machine Learning (ML) methods are more favourable at predicting non-stationary data due to their robustness and adaptability, especially compared to traditional methods. Moreover, as the volume of available internet traffic data continues to grow, these methods are expected to improve even further with respect to their prediction accuracy [20]. This research focuses primarily on predicting CPU utilisation within short periods, using only two timesteps to forecast future values. Although this proved sufficient in achieving their target application, this configuration limits the model's ability to learn long-range temporal patterns, which could lead to a significant increase in accuracy [20].

Tang et al. [21] address the limitations of the ARIMA model with regard to its reliance on static patterns in its historical data through Fisher, a cloud load prediction model designed to provide efficient and accurate predictions. Their model utilises Bidirectional LSTM for load predictions, as its non-linear nature allows it to better handle dynamic patterns. The framework also contains a metric selection module for dimensionality reduction, filtering irrelevant metrics in order to achieve efficient results. Despite its improved accuracy compared to the model discussed previously, this research also focuses on short-term accuracy, as it limits its forecast horizon to the next time slot. This shortcoming would be problematic for proactive resource management, as it requires a longer forecast horizon to allow time for resource allocation.

Velu et al. [6] developed CloudAIBus, a testbed for optimising resource allocation in cloud environments using the DeepAR model. This study underscores the importance of proactive resource allocation methods by evaluating the framework's QoS and energy costs, along with the predictive accuracy of the model. The goal of this approach is to ensure efficient use of resources in handling fluctuating cloud workloads [6]. However, the model is computationally intensive, especially since the processing time scales with the size of the data due to its RNN architecture [6]. This scalability issue could limit the model from utilising the long sequential data available from cloud workloads.

Wang et al. [18] introduced the AmazonAICloud framework, using Amazon's pre-trained Chronos model to forecast CPU utilisation, which improves resource efficiency and promotes cloud sustainability. Their study explored the potential of pre-trained models through their ability to outperform traditional baseline models like ARIMA, LSTM, and DeepAR. The findings of the paper highlighted the limitations of traditional models, which struggle with non-stationary time-series data that can often be found in real-world cloud environments [18]. Due to Chronos having a transformer-based architecture, the model will lead to scalability issues when dealing with extremely long sequential data. As the attention mechanism will consider all its inputs when processing, its computational complexity will scale with the length of the data used as input [22].

Critical Analysis

As seen in Table 1, there have been multiple works that address inefficient resource management through the use of AI models. With research from Calheiros et al. [19], Dungan et al. [20], and Tang et al. [21], promising and highly accurate results have been achieved from their predictions with traditional models. However, their assumption of time-series data being stationary could limit the performance of their models in real-world cloud scenarios. Velu et al. [6] and Wang et al. [18] addressed these issues with their frameworks by catering more to non-stationary data, utilising pre-trained deep learning models with the ability to better recognise trends and seasonality. However, the models used in both studies introduce significant computational struggles that could limit their scalability to improve their predictions in large cloud environments. This research aims to further improve forecast accuracy and scalability by taking advantage of the Mamba's state space architecture, which is designed to handle complex sequential data with superior efficiency, making it suitable for proactive real-time resource management. With the MambaNet0 framework implemented into Google Cloud's Vertex AI, it can proactively forecast the CPU utilisation of cloud applications and directly adjust their configuration to reduce inefficient resource allocation [23].

3. Methodology: MambaNet0 Framework

This section details the MambaNet0 framework and the methodology used to train the model. This includes a description of the Mamba model, the architecture of the proposed framework, the dataset used for training, the model training process, and the evaluation metrics.

3.1. Mamba

Mamba is an SSM that was developed to address the inefficiency of attention-based models such as the Transformer architecture when it comes to long sequential data [24]. Unlike its attention-based counterparts, Mamba's architecture employs a selective state-space layer that allows it to decide whether to maintain or discard specific information [24]. This mechanism helps the model avoid the performance and memory constraints of considering all tokens simultaneously. Although Mamba was developed for language modelling, its

core strength in handling long-range dependencies and linear scaling computation showed promise in predicting time-series data, which are long and sequential by nature [24].

3.2. MambaNet0 Framework

The proposed MambaNet0 framework is a novel system for proactive CPU utilisation forecasting designed to minimise inefficient cloud resource allocation. The architecture and workflow of the framework in a real-world scenario are illustrated in Figure 1.

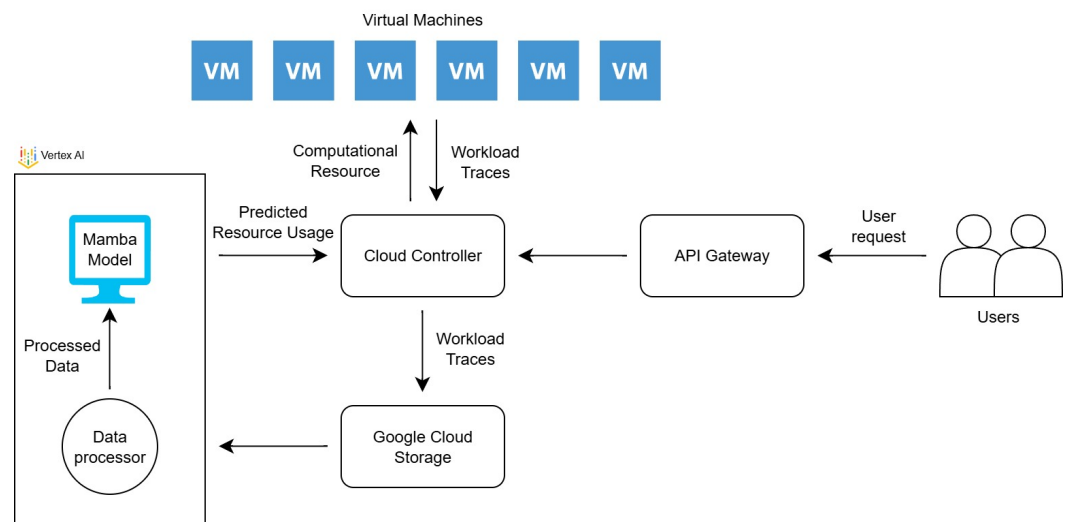


Figure 1. The architecture of the MambaNet0 framework.

The operation begins with users interacting with the services provided by the cloud applications. These interactions are sent in as requests through Amazon’s Application Programming Interface (API) Gateway, which is then relayed to the cloud controller. The cloud controller provides computational resources to virtual machines that offer these services to users while monitoring resource usage simultaneously. The cloud controller would then store these workload traces in a bucket on Google Cloud Storage (GCS), where they can be easily retrieved. The CPU usage forecaster would then take the data into the Vertex AI environment, where the data is analysed, cleaned, and resampled for the model. The trained Mamba model then takes the processed data and makes predictions for future resource usage. Afterwards, the predictions are sent back to the cloud controller, which then provides the predicted resource requirements to the virtual machines.

3.3. Parameters and Configuration

Table 2 lists the configurations used for model training. This set of configurations represents the optimal parameters found after a series of experiments that maximise the model’s performance within the available hardware constraints. The model utilises three Mamba layers, each containing 16 hidden dimensions. The input shape for each training batch is [16, 10, 1], where 16 represents the batch size, 10 is the timesteps used for training, and 1 is the number of features used. The model was configured with a forecast horizon of 30 timesteps, which means it will predict the subsequent 30 future CPU utilisation values. This results in the size of the model output having the shape [16, 30], providing 30 predictions for each of the 16 batches. For optimisation, the Adam optimiser was used with an initial learning rate of 0.001. A learning rate scheduler was also implemented, adjusting the learning rate by a factor of 0.5 if the validation loss did not improve for 10 consecutive epochs.

Table 2. Parameters and configuration used in training.

Parameter	Configuration
Hidden Layer:	3 Mamba layers
Hidden Dimension:	16
Input Size:	$16 \times 10 \times 1$
Forecast Horizon:	30
Output Size:	16×30
Optimiser:	Adam
Learning rate:	0.001
Scheduler:	Patience = 10 and factor = 0.5

3.4. Model Training Process

Algorithm 1 illustrates the pseudocode of the process. The model training process is conducted using a supervised learning approach, in which a Mamba-based neural network is trained to forecast future CPU utilisation values based on historical data. First, historical cloud data D is obtained from GCS and undergoes preprocessing, where the missing data is filled and divided into training and test data, $D_{training}$ and D_{test} . Afterwards, both $D_{training}$ and D_{test} are loaded onto a data loader, assigning each value in batches. The model is then initialised with the configuration seen in Table 2. The process can be formulated as a univariate time-series forecasting problem, with CPU usage from the dataset as the input feature. The training data with the shape $[16, 10, 1]$ is passed into the model through an input layer. The input vector can be denoted as a batch of input sequences X_t :

$$X_t = [x_{t-9}, x_{t-8}, \dots, x_t] \quad (1)$$

where X_t is the sequence of the last 10 timesteps of CPU utilisation values. This layer projects the single-feature data into a higher-dimensional space defined by the hidden dimension parameter d_{model} . The shape of the data becomes $[16, 10, 16]$ ($d_{model} = 16$), which is passed sequentially through the three Mamba layers. Inside each Mamba block, the information is processed through the SSM, which filters and compresses the sequential data into a hidden state vector. The output of the final Mamba layer is then passed onto an output layer that projects the hidden dimensions onto the forecasts, resulting in predictions with the shape $[16, 30]$. The output vector can be denoted as a batch of output sequences Y_t :

$$Y_t = [x_{t+1}, x_{t+2}, \dots, x_{t+30}] \quad (2)$$

where Y_t is the sequence of the predicted CPU utilisation values for the next 30 timesteps. The model's parameters are optimised during training by minimising the MSE loss function. The MSE is the average squared error from the predicted values and the actual values. The equation for MSE can be seen below as Equation (3).

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (3)$$

MSE was chosen as the optimisation function due to its ability to indicate the performance of a model through its sensitivity towards large errors [25]. The Adam optimiser is used to fine-tune the model's parameters along with a learning rate scheduler to help adjust the learning rate during the training process. Once training is complete, the model is evaluated in the test set using the MAE, RMSE, and SMAPE metrics.

Algorithm 1 Pseudocode of MambaNet0's training algorithm.

```

1: Load data  $D$  from GCS
2:  $D_{\text{training}}, D_{\text{test}} \leftarrow \text{DataProcessor}(D)$ 
3:  $\text{Training}_{\text{batch}}, \text{Test}_{\text{batch}} \leftarrow \text{DataLoader}(D_{\text{training}}, D_{\text{test}})$ 
4: Initialise model  $M$  with parameters  $P$ 
5: for epoch in range( $E$ ) do
6:    $\text{Input}_{\text{batch}}, \text{Target}_{\text{batch}} \leftarrow \text{Training}_{\text{batch}}$ 
7:    $\text{Predictions}_{\text{training}} \leftarrow M(\text{Input}_{\text{batch}})$ 
8:    $\text{Loss}_{\text{batch}} \leftarrow \text{Loss}(\text{Predictions}_{\text{training}}, \text{Target}_{\text{batch}})$ 
9:   Update model parameters using Adam
10:  Adjust learning rate if necessary
11: end for
12:  $\text{Input}_{\text{batch}}, \text{Target}_{\text{batch}} \leftarrow \text{Test}_{\text{batch}}$ 
13:  $\text{Predictions}_{\text{test}} \leftarrow M(\text{Input}_{\text{batch}})$ 
14:  $\text{Results}_{\text{test}} \leftarrow \text{EvaluationMetrics}(\text{Predictions}_{\text{test}}, \text{Target}_{\text{batch}})$ 

```

3.5. Dataset

The dataset used was the Bitbrains GWA-T-12 dataset, which contains the performance metrics of 1750 virtual machines, spanning over a period of three months. The available metrics consist of CPU usage, provisioned CPU capacity, provisioned memory, memory usage, disk read/write throughput, and network received/transmitted throughput. A sample of the dataset can be seen in Table 3.

To highlight the gap between the used CPU capacity and the provisioned CPU capacity, the values of these two features from the 1750 available virtual machines were aggregated and plotted in Figure 2. The graph displays a stark difference between the two values, with the provisioned capacity being constantly higher than needed, indicating an almost constant state of over-provisioning. This is further supported through data analysis, where the average CPU usage was approximately 4,699,483 MHz, while the average provisioned CPU capacity was around 42,553,551 MHz, highlighting the extremely inefficient nature of resource utilisation in cloud computing.

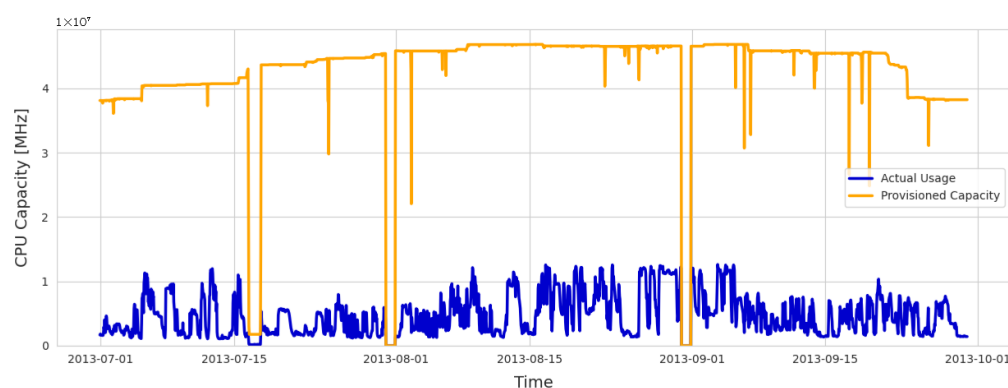


Figure 2. Comparison between provisioned CPU capacity and actual CPU usage.

Table 3. A sample of the Bitbrains dataset.

Timestamp	CPU Cores	CPU Provisioned [MHZ]	CPU Usage [MHZ]	CPU Usage [%]	Memory Provisioned [KB]	Memory Usage [KB]	Disk Read [KB/s]	Disk Write [KB/s]	Network Received [KB/s]	Network Transmitted [KB/s]
1372629804	2	5851.998	87.779	1.500	8,218,624.0	1.034×10^6	160.866	21.733	0.266	1.467
1372630104	2	5851.998	29.259	0.500	8,218,624.0	4.585×10^5	0.000	2.333	0.200	1.000
1372629804	2	5851.998	27.309	0.466	8,218,624.0	1.845×10^5	32.066	4.200	0.133	1.067

Table 3. Cont.

Timestamp	CPU Cores	CPU Provisioned [MHZ]	CPU Usage [MHZ]	CPU Usage [%]	Memory Provisioned [KB]	Memory Usage [KB]	Disk Read [KB/s]	Disk Write [KB/s]	Network Received [KB/s]	Network Transmitted [KB/s]
1372629804	2	5851.998	23.407	0.400	8,218,624.0	7.829×10^4	0.000	0.866	0.067	1.000
1372629804	2	5851.998	19.506	0.333	8,218,624.0	1.677×10^5	0.000	0.200	0.133	1.000

3.6. Data Preprocessing

To prepare the historical cloud workload dataset for model training, data preprocessing is performed. First, missing values in the time-series data are identified and addressed using forward filling, where if a value d_i from time step i is missing, the value d_{i-1} from the previous timestep will be used to fill in to maintain continuity. Subsequently, the data is normalised using standard scaling, which transforms the data to have a mean of zero and a standard deviation of one ($\mu = 0, \sigma = 1$). The dataset is then resampled into a consistent and uniform frequency, as the Bitbrains dataset has an inconsistent sampling rate. This ensures an appropriate time-series structure for model training, while also reducing noise in the data.

4. Performance Evaluation

This section explains the experimental setup, the baseline models used, the evaluation metrics, and the results.

4.1. Experimental Setup

The proposed MambaNet0 framework was implemented on Google Cloud's Vertex AI, along with the baseline models, to ensure a consistent and adjustable experimental environment. Vertex AI is an AI development platform that offers scalable computational resources and Machine Learning Operations tools (MLOps) to help streamline the model management and deployment process [26]. The training and evaluation of this experiment were conducted using Jupyter Notebook version 6.5.7 within the Vertex AI Python 3.10.18 environment. GCS was also used for storing the historical data that will be used by the models to train and predict cloud resources.

4.2. Baseline Models

This section outlines the baseline models used, along with the parameters utilised on each model. The performance of the Mamba model that was implemented in Vertex AI is compared with three baseline models: ARIMA [19], LSTM [21], and Amazon Chronos [18].

4.2.1. Autoregressive Integrated Moving Average (ARIMA)

ARIMA is the combination of the autoregressive model and the moving average model, along with the inclusion of a differencing operator, which helps make the time series stationary by reducing the reliance on past values [27]. The model is widely used in time-series forecasting due to its proven capabilities to handle non-stationary data [27]. The parameters used for the ARIMA(p, d, q) model are ARIMA(10,1,0), where the autoregressive term ($p = 10$) is the number of timesteps used for making prediction, the degree of differencing ($d = 1$) denotes the number of times data was differenced to achieve stationarity, and the moving average term ($q = 0$) indicates the size of the moving average.

4.2.2. Long Short-Term Memory (LSTM)

LSTM is a type of RNN that specialises in long sequential data due to its ability to maintain information [27]. Each LSTM layer contains cells that capture and store data, enabling it to learn and filter relevant information over long sequences [27]. This is achieved

through gates, a sigmoidal neural network layer that determines the flow of information between each cell [27]. The parameter of the model utilised consists of an LSTM layer, configured with four cells, and processed input sequences with a window size of 10.

4.2.3. Chronos

Chronos is a set of pre-trained time-series forecasting models that utilise transformer-based architecture, taking a probabilistic approach to time-series forecasting and tokenising their inputs similar to Large Language Models (LLMs) [28]. As Mamba's design was to address the limitations of transformer-based models, Chronos serves as a great baseline for comparison when performing on long sequential data. To ensure computational feasibility without utilising expensive hardware such as a Graphics Processing Unit (GPU), the Chronos small model was used.

4.3. Evaluation Metrics

This section discusses the evaluation metrics used to measure the accuracy of the proposed models and the baseline models. The metrics used consist of MAE, RMSE, and SMAPE. MAE is the average sum of errors from the predicted values and the actual values [29]. This is a straightforward metric that represents the accuracy of the forecast. The equation for MAE can be seen below as Equation (4).

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (4)$$

where y_i is the value at i , $\hat{y}_i$ is the predicted value at i , and n is the number of data points. RMSE is the square root of the average squared error [29]. Due to the fact that the error values are squared, the RMSE is more sensitive to large errors, which helps highlight incorrect forecasts. The equation for RMSE can be seen below as Equation (5).

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (5)$$

where y_i is the value at i , $\hat{y}_i$ is the predicted value at i , and n is the number of data points. SMAPE is the average absolute percentage error, where the denominator used is the average of the predicted and actual values, rather than just the actual value. This approach provides a symmetrical balance to both under-prediction scenarios and over-prediction scenarios, while also addressing the issue when the actual value is zero [30]. The equation for SMAPE can be seen below as Equation (6).

$$SMAPE = \frac{1}{n} \sum_{i=1}^n \frac{|y_i - \hat{y}_i|}{(|y_i| + |\hat{y}_i|)/2} \quad (6)$$

where y_i is the value at i , $\hat{y}_i$ is the predicted value at i , and n is the number of data points.

4.4. Results

Table 4 presents the performance of the Mamba model in comparison to the chosen baseline models. Each model was configured with a forecast horizon of 30 timesteps ahead to accommodate the requirements for the forecast and the cloud server configuration process. Figure 3 displays the predictions made by the MambaNet0 framework compared to the true values.

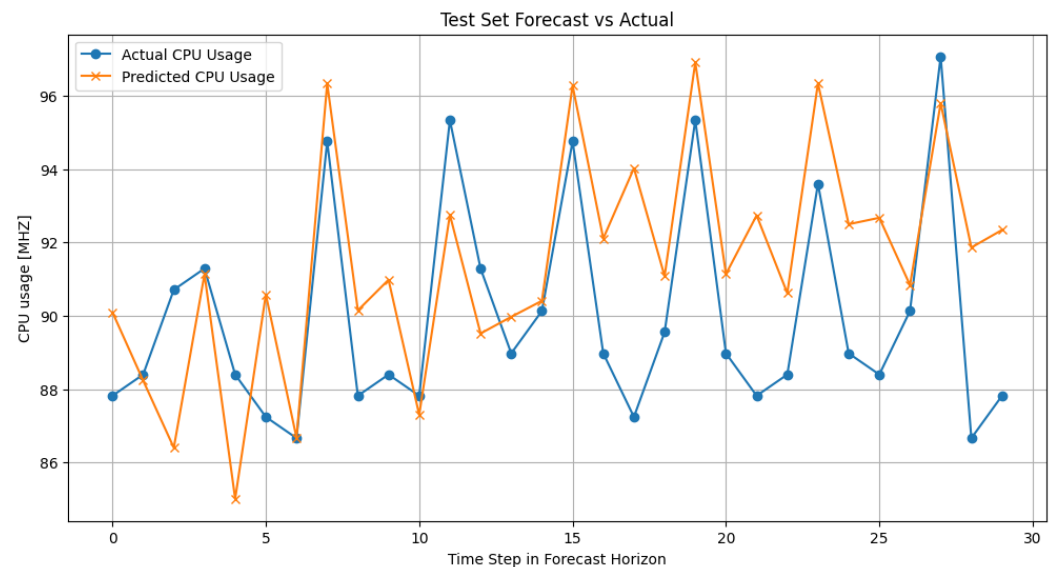


Figure 3. Resulting CPU usage prediction from the model compared to the actual CPU usage.

Table 4. Comparison of predicted and observed CPU utilisation across the forecast horizon.

Work	Model	MAE	RMSE	SMAPE
Calheiros et al. (2014) [19]	ARIMA	136.693	216.295	24.82
Tang et al. (2018) [21]	LSTM	132.06	214.13	23.84
Wang et al. (2025) [18]	Chronos	5.987	9.796	6.411
MambaNet0 (this paper)	Mamba	3.932	4.758	4.477

The traditional baseline models, ARIMA and LSTM, displayed a similar level of accuracy, with LSTM slightly outperforming ARIMA on all metrics. Although LSTM showed stronger performance than ARIMA, its resulting MAE of 132.06 suggests poor overall accuracy. This is further supported by its RMSE of 214.13, which likely indicates significant forecast errors caused by sudden spikes or unexpected fluctuations within the data. With SMAPE being relatively high for both ARIMA and LSTM at 24.82 and 23.84, respectively, they exhibit a significant deviation in the predictions from the true values, which suggests their unsuitability in cloud resource forecasting.

Chronos, on the other hand, demonstrated a remarkable leap in performance compared to the other baseline models. Its forecast results with a lower MAE of 5.987 and an RMSE of 9.796, underscoring its ability to provide accurate predictions. With a lower SMAPE of 6.411, Chronos solidifies itself as an effective approach to cloud resource forecasting, highlighting the effectiveness of transformer-based architectures with lengthy sequential data.

Likewise, Mamba showcased even superior performance, outperforming all baseline models across all metrics. It achieved the lowest MAE of 3.932, RMSE of 4.758, and SMAPE of 4.477. This performance gap represents a significant leap, seeing an approximate 34% reduction in MAE and 51% in RMSE when compared to the best-performing baseline model, Chronos. This magnitude of improvement could suggest an advantage introduced by the advances made from Mamba's architecture. It could stem from Mamba's selective SSM, which allows the model to filter and compress relevant information while discarding noise better than the transformer-based architecture of Chronos. This might have enabled the Mamba model to have better retention of relevant historical patterns, thus providing more accurate predictions. Furthermore, similar MAE (3.932) and RMSE (4.758) demonstrate Mamba's robustness in error distribution. Since RMSE amplifies large error values, the small difference between MAE and RMSE indicates the success of Mamba in

mitigating large prediction errors. This highlights the model's ability to forecast sudden spikes and troughs that traditional models like ARIMA and LSTM consistently struggle with. Furthermore, the low SMAPE value of 4.477 reinforces its effectiveness at providing accurate and consistent predictions, demonstrating its potential in precise and proactive resource allocation.

5. Conclusions and Future Directions

This study proposes a new AI-driven cloud resource forecasting framework called MambaNet0, a system for efficient cloud resource provisioning. Through its utilisation of the Mamba model to predict future resource usage, MambaNet0 enables proactive resource allocation for cloud servers to help mitigate inefficient resource provisioning. The framework was implemented and evaluated in Google Cloud's Vertex AI environment, where it outperforms established baseline models like ARIMA, LSTM, and Chronos by achieving 29% improvement in SMAPE against the top baseline model. These results underscore the potential of the Mamba model in providing accurate predictions for cloud resource utilisation, which would lead to a reduction in operational costs, along with supporting environmental and net zero goals.

Future Directions

This section discusses possible future directions that could be expanded upon in MambaNet0. While this study has successfully demonstrated the effectiveness of the MambaNet0 framework in providing accurate CPU utilisation forecasts, there are multiple avenues of future work to explore:

- The natural next step for this research would be the integration of the MambaNet0 predictive framework with a resource allocation system. While this paper focuses on validating the model's forecasting capabilities, a complete solution requires a system that would deploy these predictions to the cloud environment. Future work could explore a resource allocation scheme, such as the online auction mechanism [31]. This integration could take advantage of MambaNet0's predictions to formulate a complete solution that could deliver an improvement in cloud resource utilisation, reduce over-provisioning costs, and improve social welfare in its distributions.
- The framework's generalisability can be confirmed by evaluating its performance on diverse time-series datasets from various cloud providers, which would validate its robustness and adaptability in real-world environments.
- The framework's versatility could be further validated by extending its forecasting capabilities to other cloud resources, such as memory usage, disk I/O, and network traffic. This would confirm its effectiveness beyond CPU utilisation prediction.
- Moreover, the framework's environmental impact could be quantified using a digital twin of a cloud environment. By simulating daily energy consumption based on MambaNet0's forecasts, this approach would precisely measure potential reductions in energy waste and carbon emissions, directly assessing the framework's contribution to cloud sustainability [32].
- The Mamba architecture's flexibility presents an opportunity to create hybrid models. Integrating Mamba with complementary forecasting techniques could leverage their combined strengths to achieve even greater predictive accuracy.

Author Contributions: T.C.: writing—original draft, writing—review and editing, visualisation, software, methodology, formal analysis, data curation, and conceptualisation. H.W.: writing—original draft, writing—review and editing, methodology, investigation, formal analysis, and conceptualisation.

tion. S.S.G.: writing—original draft, writing—review and editing, supervision, methodology, and conceptualisation. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The code and reproducibility scripts are available on GitHub: <https://github.com/Thananont/mamba-forecasting>.

Conflicts of Interest: The authors declare no conflicts of interest.

References

- Islam, R.; Patamsetti, V.; Gadhi, A.; Gondu, R.M.; Bandaru, C.M.; Kesani, S.C.; Abiona, O. The future of cloud computing: Benefits and challenges. *Int. J. Commun. Netw. Syst. Sci.* **2023**, *16*, 53–65. [\[CrossRef\]](#)
- Mistry, H.K.; Mavani, C.; Goswami, A.; Patel, R. The impact of cloud computing and AI on industry dynamics and competition. *Educ. Adm. Theory Pract.* **2024**, *30*, 797–804. [\[CrossRef\]](#)
- Galaz, V.; Centeno, M.A.; Callahan, P.W.; Causevic, A.; Patterson, T.; Brass, I.; Baum, S.; Farber, D.; Fischer, J.; Garcia, D.; et al. Artificial intelligence, systemic risks, and sustainability. *Technol. Soc.* **2021**, *67*, 101741. [\[CrossRef\]](#)
- Mahida, A. Comprehensive review on optimizing resource allocation in cloud computing for cost efficiency. *J. Artif. Intell. Cloud Comput.* **2022**, *232*, 2–4. [\[CrossRef\]](#)
- Yadav, R.; Zhang, W.; Kaiwartya, O.; Singh, P.R.; Elgendy, I.A.; Tian, Y.C. Adaptive energy-aware algorithms for minimizing energy consumption and SLA violation in cloud computing. *IEEE Access* **2018**, *6*, 55923–55936. [\[CrossRef\]](#)
- Velu, S.; Gill, S.S.; Murugesan, S.S.; Wu, H.; Li, X. CloudAIBus: A testbed for AI based cloud computing environments. *Clust. Comput.* **2024**, *27*, 11953–11981. [\[CrossRef\]](#)
- Wang, Y.; Yang, X. Cloud Computing Energy Consumption Prediction Based on Kernel Extreme Learning Machine Algorithm Improved by Vector Weighted Average Algorithm. In Proceedings of the 2025 5th International Conference on Artificial Intelligence and Industrial Technology Applications (AIITA), Xi'an, China, 28–30 March 2025; IEEE: Piscataway, NJ, USA, 2025; pp. 1355–1361.
- Yadav, R.; Zhang, W.; Li, K.; Liu, C.; Laghari, A.A. Managing overloaded hosts for energy-efficiency in cloud data centers. *Clust. Comput.* **2021**, *24*, 2001–2015. [\[CrossRef\]](#)
- UK Government. *Net Zero Strategy: Build Back Greener*; Technical Report; UK Government: London, UK, 2021.
- Zhou, Q.; Sheu, J.B. The use of Generative Artificial Intelligence (GenAI) in operations research: Review and future research agenda. *J. Oper. Res. Soc.* **2025**, 1–21. [\[CrossRef\]](#)
- Lee, R. Large Language Models (LLMs) and Generative Artificial Intelligence (GenAI). In *Natural Language Processing: A Textbook with Python Implementation*; Springer: Singapore, 2025; pp. 241–273.
- Gill, S.S.; Jackson, J.; Qadir, J.; Parlikad, A.K.; Alam, M.; Jarvis, L.; Fuller, S.; Morris, G.R.; Shah, R.; Zawahreh, Y.I. et al. A Toolkit for Sustainable Educational Environment in the Modern AI Era: Guidelines for Mitigating the Misuse of GenAI in Assessments. *TechRxiv* **2025**, 1–18. [\[CrossRef\]](#)
- Van Der Vlist, F.; Helmond, A.; Ferrari, F. Big AI: Cloud infrastructure dependence and the industrialisation of artificial intelligence. *Big Data Soc.* **2024**, *11*, 20539517241232630. [\[CrossRef\]](#)
- Egbuhuzor, N.S.; Ajayi, A.J.; Akhigbe, E.E.; Agbede, O.O. Leveraging AI and cloud solutions for energy efficiency in large-scale manufacturing. *Int. J. Sci. Res. Arch.* **2024**, *13*, 4170–4192. [\[CrossRef\]](#)
- Chen, Z.; Ma, M.; Li, T.; Wang, H.; Li, C. Long sequence time-series forecasting with deep learning: A survey. *Inf. Fusion* **2023**, *97*, 101819. [\[CrossRef\]](#)
- Prangon, N.F.; Wu, J. AI and computing horizons: Cloud and edge in the modern era. *J. Sens. Actuator Netw.* **2024**, *13*, 44. [\[CrossRef\]](#)
- Katal, A.; Dahiya, S.; Choudhury, T. Energy efficiency in cloud computing data centers: A survey on software technologies. *Clust. Comput.* **2023**, *26*, 1845–1875. [\[CrossRef\]](#)
- Wang, H.; Mathews, K.J.; Golec, M.; Gill, S.S.; Uhlig, S. AmazonAICloud: Proactive resource allocation using amazon chronos based time series model for sustainable cloud computing. *Computing* **2025**, *107*, 77. [\[CrossRef\]](#)
- Calheiros, R.N.; Masoumi, E.; Ranjan, R.; Buyya, R. Workload prediction using ARIMA model and its impact on cloud applications' QoS. *IEEE Trans. Cloud Comput.* **2014**, *3*, 449–458. [\[CrossRef\]](#)
- Duggan, M.; Mason, K.; Duggan, J.; Howley, E.; Barrett, E. Predicting host CPU utilization in cloud computing using recurrent neural networks. In Proceedings of the 2017 12th International Conference for Internet Technology and Secured Transactions (ICITST), Cambridge, UK, 11–14 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 67–72.

21. Tang, X.; Liu, Q.; Dong, Y.; Han, J.; Zhang, Z. Fisher: An efficient container load prediction model with deep neural network in clouds. In Proceedings of the 2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCLOUD/SocialCom/SustainCom), Melbourne, Australia, 11–13 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 199–206.
22. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017.
23. Wang, Z.; Kong, F.; Feng, S.; Wang, M.; Yang, X.; Zhao, H.; Wang, D.; Zhang, Y. Is Mamba effective for time series forecasting? *Neurocomputing* **2025**, *619*, 129178. [\[CrossRef\]](#)
24. Gu, A.; Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv* **2023**, arXiv:2312.00752. [\[CrossRef\]](#)
25. Hodson, T.O.; Over, T.M.; Foks, S.S. Mean squared error, deconstructed. *J. Adv. Model. Earth Syst.* **2021**, *13*, e2021MS002681. [\[CrossRef\]](#)
26. Introduction to Vertex AI. 2025. Available online: <https://cloud.google.com/vertex-ai/docs/start/introduction-unified-platform> (accessed on 27 July 2025).
27. Siami-Namini, S.; Tavakoli, N.; Namin, A.S. A comparison of ARIMA and LSTM in forecasting time series. In Proceedings of the 2018 17th IEEE international conference on machine learning and applications (ICMLA), Orlando, FL, USA, 17–20 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1394–1401.
28. Ansari, A.F.; Stella, L.; Turkmen, C.; Zhang, X.; Mercado, P.; Shen, H.; Shchur, O.; Rangapuram, S.S.; Arango, S.P.; Kapoor, S.; et al. Chronos: Learning the language of time series. *arXiv* **2024**, arXiv:2403.07815. [\[CrossRef\]](#)
29. Model Evaluation in Vertex AI. 2025. Available online: <https://cloud.google.com/vertex-ai/docs/evaluation/introduction#forecasting> (accessed on 27 July 2025).
30. Kreinovich, V.; Nguyen, H.T.; Ouncharoen, R. *How to Estimate Forecasting Quality: A System-Motivated Derivation of Symmetric Mean Absolute Percentage Error (SMAPE) and Other Similar Characteristics*; University of Texas at El Paso: El Paso, TX, USA, 2014.
31. Zhang, J.; Yang, X.; Xie, N.; Zhang, X.; Vasilakos, A.V.; Li, W. An online auction mechanism for time-varying multidimensional resource allocation in clouds. *Future Gener. Comput. Syst.* **2020**, *111*, 27–38. [\[CrossRef\]](#)
32. Henzel, J.; Wróbel, Ł.; Fice, M.; Sikora, M. Energy consumption forecasting for the digital-twin model of the building. *Energies* **2022**, *15*, 4318. [\[CrossRef\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.