

Energy-Efficient Cloud Usage Forecasting Using SE-GRU for Power and Carbon Emission Reduction

MSc Research Project
Msc in Cloud Computing

Rishabh Raj
Student ID: x23336315

School of Computing
National College of Ireland

Supervisor: Rashid Mijumbi

National College of Ireland
MSc Project Submission Sheet



School of Computing

Student Name: Rishabh Raj
Student ID: X23336315
Programme: Msc in cloud Computing **Year:** 2025
Module: Research Project
Supervisor: Rashid Mijumbi
Submission Due Date: 28-01-2026
Project Title: Energy-Efficient Cloud usage Forecasting Using SE-GRU for Power and Carbon Emission Reduction
Word Count: 8960 **Page Count:** 23

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Rishabh Raj
Date: 28-01-26

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Energy-Efficient Cloud Usage Forecasting Using SE-GRU for Power and Carbon Emission Reduction

Rishabh Raj

Student ID : x23336315

Abstract

Cloud data centers face growing energy consumption and carbon emissions due to quickly fluctuating CPU workloads by making accurate forecasting which is important for sustainable and cost-efficient cloud operations. There are some existing deep learning approaches like LSTM, BiLSTM and GRU provide reasonable predictions but struggle with sudden workload spikes, higher latency and insufficient linkage which is in between CPU forecasting and subsequent power and carbon estimation. This study addresses these limitations by proposing a Squeeze-and-Excitation GRU (SE-GRU) model to improve workload sensitivity and forecasting accuracy. Also the trained SE-GRU model was deployed on an AWS EC2 instance to validate real-time inference capability. Synthetic workload data has generated every five minutes to simulate live CPU behavior and the system produced forecasts. When high utilization was predicted then an automated SNS notification was triggered which shows its potential for future integration with AWS CloudWatch metrics. There are some experimental evaluations which shows that SE-GRU outperforms all baseline models by achieving the lowest MSE ($9.48e-06$), highest R^2 (0.980), reduced latency (61.03 ms) and improved throughput. It also gives the lowest estimated power and carbon outputs by showing stronger environmental performance. The findings advance current state-of-the-art CPU forecasting by using attention-enhanced lightweight modeling with sustainability analysis.

Keywords: CPU forecasting, SE-GRU model, cloud energy, power and carbon prediction

1 Introduction

This chapter presents the background and importance of the study which is been followed by the research question and clearly defined research objectives. It also outlines the research contributions which is showing the proposed SE-GRU model.

1.1 Background

Cloud computing refers to the delivery of computing resources such as servers, storage, databases, networking, software, and analytics over the internet to offer flexible and scalable

computing power (Sharifzadeha et al., 2022). Data centers, which form the core infrastructure of cloud computing, host thousands of interconnected servers that process and store enormous volumes of data (Xie, 2023). However, this growth has resulted in substantial energy consumption, leading to increased operational costs and environmental concerns. Energy efficiency in cloud data centers can be defined as the optimal use of computing resources to minimize energy usage while maintaining performance and service quality (Long et al., 2022). To achieve this, Machine Learning (ML)—a subset of artificial intelligence that enables systems to learn from data without explicit programming—plays a crucial role in analyzing and forecasting resource utilization trends.

1.2 Importance of the Study

The increasing reliance on cloud computing has led to a significant rise in energy consumption within data centers, making energy efficiency a critical concern for both economic and environmental sustainability (Saxena and Singh, 2021). This study holds great importance as it aims to optimize power utilization by predicting CPU usage using advanced deep learning models, thereby reducing unnecessary energy wastage while maintaining system performance. Accurate forecasting of resource consumption enables proactive management and efficient workload scheduling, which directly contributes to lowering operational costs and carbon emissions. The outcomes of this study not only promote sustainable computing but also support the development of green cloud infrastructure, aligning with global efforts toward energy conservation and environmental responsibility. Ultimately, this work contributes to building a more resilient, efficient, and eco-friendly cloud computing ecosystem, benefiting both service providers and end users.

1.3 Research Question

RQ1: How accurately can deep learning models forecast CPU usage under variable cloud workloads?

RQ2: To what extent does SE attention improve forecasting accuracy and downstream power-carbon estimation?

1.4 Research Objectives

1. To develop and evaluate deep learning models which includes LSTM, BiLSTM, GRU and the proposed SE-GRU architecture for accurately forecasting CPU usage in cloud data centers under dynamic and volatile workload patterns.
2. To design an end-to-end energy-aware forecasting pipeline that converts predicted CPU usage into power consumption and carbon emissions by enabling sustainability analysis for cloud infrastructures.
3. To assess the computational performance and real-time applicability of the proposed SE-GRU model by comparing latency, throughput and prediction performance against baseline models by securing suitability for scalable cloud deployment.

1.5 Research Contributions

This study suggests a new SE-GRU forecasting model in cloud computing that will increase the capability of predicting CPU utilization with spikes of workloads and decreasing noise by

combining Squeeze-and-Excitation attention with GRU. It provides full chain sustainability pipeline transforming CPU predictions into power and carbon estimates, making decisions energy-conscious. The study offers an in-depth performance comparison of LSTM, BiLSTM, GRU, and SE-GRU in terms of accuracy, latency, and throughput, and has shown that the proposed model is better than the rest. The framework is also lightweight, cloud-deployable, and can be used in real-time resources management, which is where the gaps in the extant literature are addressed.

1.6 Structure of the Report

The report is structured into seven chapters. It begins with an introduction outlining the background, importance, research question, objectives and contributions. The related work chapter reviews energy-efficient cloud computing, resource scheduling and DL approaches while identifying key research gaps. The methodology chapter explains the research design, dataset, preprocessing, feature engineering, visualisation, rolling-window creation and model-training process. The design specification presents the system architecture, tools and justification of the SE-GRU model. The implementation chapter details all DL models and forecasting procedures. The evaluation chapter analyses accuracy and all results. Finally the conclusion summarises findings and suggests future improvements.

2 Related Work

This chapter reviews existing work on Energy Efficiency in Cloud Computing, examining foundational strategies and emerging machine-learning-based approaches. It also explained critical analysis and identified gaps.

2.1 Energy Efficiency in Cloud Computing

Energy efficiency has become an important concern in modern cloud computing infrastructures due to the exponential growth of computational workloads driven by artificial intelligence, big data and continuous online services (Katal et al., 2023). Cloud data centers form the foundation of digital operations, which still take a huge share of electricity usage in the world (Al Kez et al., 2022). It has been shown that operational energy cost of large-scale data centers has the potential to add up to 50 -60 percent of overall maintenance cost, thus energy optimization is a critical aspect of sustainable cloud computing. The conventional cloud management strategies were mostly based on fixed allocation of resources and over provisioning to ensure that the performance (Maenhaut et al., 2020) was always upheld but with these techniques, the energy was usually wasted as all the servers were not fully utilized and the redundancy of the computational process. The recent progress has changed the focus to dynamic workload balancing, adaptive scheduling and resource allocation using machine learning to optimize power usage without affecting the quality of the services (Gures et al., 2022). Things like virtual machine (VM) consolidation, dynamic voltage and frequency scaling (DVFS), and predictive load models have been considered in order to minimize the total energy cost of a data center. An example is the use of machine learning models to predict the needs of resources and dynamically allocate the computational resources, thus reducing power wastage on idle resources and enhancing the efficiency of the system. The current goal of energy-conscious scheduling algorithms is to allocate workloads rationally among the accessible nodes with the minimum energy consumption and a reasonable level of latency. Moreover, the use of green computing framework and integration of renewable energy in cloud computing is another emerging trend of realizing carbon-neutral operation (Paul et al., 2023). The problems with

these innovations remain in terms of predictability of performance as well as the management of the heterogeneous workload and scalability in response to the changing demand. Thus, energy-efficient cloud computing has become a research priority that involves the development of smart, self-adaptive cloud ecosystems with the ability to anticipate workload behavior, utilize resources more efficiently (Xu et al., 2021) and eventually minimize power consumption in the distributed cloud ecosystem. This changing environment forms the basis of the implementation of machine learning algorithms in workload management systems that are energy aware in order to be sustainable in cloud computing.

Along with recent deep learning methods, classical time-series methods including ARIMA, Holt-Winters and Prophet have been applied to workload forecasting on CPUs and VM and have set their foundational studies in cloud workload prediction (Masdari and Khoshnevis, 2020). SE-attention mechanisms are also tested in various time-series-based applications - including SE-LSTM which was found to demonstrate that channel recalibration is capable of enhancing the weighting of features, though have not been commonly investigated in cloud CPU forecasting (Zheng et al., 2024). In addition to linear power modeling, nonlinear and server-specific models such as SPECpower, DVFS-aware energy models, and PUE-adjusted consumption models have been examined by previous research, demonstrating the necessity to have a more realistic power-carbon mapping in forecasting literature.

2.2 Resource Scheduling and Load Balancing Mechanisms

In cloud and edge computers, resource scheduling and load balancing are critical processes in order to guarantee the optimal use of the computational resources, lower latency, and Quality of Service (QoS). A number of researches have suggested new frameworks and algorithms to overcome the issue of resource imbalance and inefficient scheduling. A load-balancing Docker scheduler provided by OpenStack was proposed in (Qian et al., 2023), and it created exclusive weights of containers by filtering, weight adaptation and NUMA optimization steps. This approach was high in minimizing the intra-and inter-node imbalance but was less flexible to real-time workload changes. On the same note, (Bal et al., 2022) also suggested a hybrid machine learning framework (RATS-HM) that fuses cat swarm optimization enhanced with better task scheduling, a group-based optimization deep neural network with resource allocation, and a lightweight authentication system that ensures data security. Even though it recorded better performance in utilization and response time, it has a high level of computational complexity. In (Mahdizadeh et al., 2024), the authors discussed cloud and software-defined networking (SDN) architectures and applied such algorithms as IRMTS, reinforcement learning, and PSG to optimize QoS and network performance in simulators, including Mininet and CloudSim, and reported scalability and integration problems. Moving even further to the paradigm of fog computing, (Baburao et al., 2023) introduced an Enhanced Dynamic Resource Allocation Method (EDRAM) based on Particle Swarm Optimization so as to minimize the latency, waiting time and bandwidth usage by dynamically reclaiming unused services, but the adaptability to dynamic workloads was limited by the convergence of PSO. Lastly, (Vijay and Sree, 2023) conceptually analyzed the resource scheduling and load balancing methods with the focus on allocating the workloads and enhancing the stability of the system without experimentation. Taken together, these pieces of work emphasize the fact that even though different algorithmic and hybrid strategies have provided better efficiency and functionality in cloud, edge, and fog settings, critical research areas that should be tackled to make step towards intelligent and autonomous management of resources include scalability, computational overhead, and adaptability under dynamism.

2.3 Deep Learning in Energy Efficiency in Cloud

The growing need of computational power, which is provided by the spread of the Internet of Things, cloud computing, and applications that consume data, has increased the amount of energy consumed in the world, requiring more effective and smart approaches to managing energy resources. Deep learning has risen to be a disruptive technology towards enhancing energy efficiency of cloud infrastructures, data centers, and renewable energy systems. A model to predict short-term solar irradiance based on a deep learning architecture based on Long Short-Term Memory (LSTM) and optimized by Bayesian Optimization (BO) was developed in (Zuo et al., 2022). The model used meteorological variables together with cloud coverage characteristics to forecast global horizontal irradiance (GHI) with a normalized RMSE of 15.25 which is 8.23% lower than the baseline models and therefore, will help to better integrate solar energy in smart grids. In (Ounifi et al., 2022), the authors concentrated on prediction of Power Usage Effectiveness (PUE) in data centers with the help of three machine learning models, i.e. MLP, Resilient Backpropagation DNN, and an Attention-based LSTM where the latter demonstrated the best performance in terms of nonlinear dependencies prediction of data center systems and its lowest prediction error and maximum PUE accuracy. On the same note, (Sireesha et al., 2024) proposed a Layer Optimized LSTM (LOLSTM) model to predict cloud resource needs during dynamic workloads to ensure 97.6% training and 95.9% validation accuracy thus forecasting the future needs and violating Service Level Agreement (SLA) to achieve QoS as shown in Table 1.

In (Pan et al., 2024), the authors discussed the latency and energy inefficiency of the IoT systems by implementing Lyapunov-based Dynamic LSTM and Particle Swarm Optimization (LDLPSO) algorithm in the fog computing system. This model enhanced the speed of response by 25% and energy consumption by 9.44, which is better than other algorithms and allows rechargeable networks to be scaled. In the meantime, (Tabrizchi et al., 2023) suggested a CNN-stacked BiLSTM hybrid architecture to forecast the temperatures of hosts in a hyperscale data center, which was effective in overcoming the problem of hotspots formation and cooling overhead. This model had a high R2 of 97.15 percent, and the RMSE was reduced to 0.2892, which is better than the conventional models in predicting thermals. Lastly, (Alizadegan et al., 2025) introduced Bi-LSTM time series forecasting model of energy consumption prediction in the smart grids. The model showed better results when compared to ARIMA, SARIMA, and traditional LSTM models, with the highest improvement of 50.6% in RMSE which will help to manage energy responsibly and efficiently. Altogether, these researches emphasize that deep learning, in particular, LSTM variations is paramount in improving energy efficiency, predictive performance, and sustainability in cloud settings. Scalability, however, computational overhead and scaling to a heterogeneous or mobile environment are some of these challenges that need to be addressed in future studies.

Table 1: Summary Table

Study	Focus Area	Proposed Approach	Challenge Addressed	Key Results	Limitations
(Zuo et al., 2022)	Solar irradiance forecasting	LSTM optimized with Bayesian Optimization	Cloud-induced irradiance variability	nRMSE = 15.25%, 8.23% better than persistence model	Limited generalization to different climatic regions
(Ounifi et al., 2022)	Data center energy efficiency	Attention-based LSTM vs. MLP & DNN	Nonlinear subsystem dependencies	Best PUE prediction	Limited dataset diversity for broader validation

	(PUE prediction)			accuracy among models	
(Sireesha et al., 2024)	Cloud resource prediction	Layer Optimized LSTM (LOLSTM)	Dynamic workloads and SLA violations	Accuracy: 97.6% (train), 95.9% (val), RMSE = 0.127	Scalability and data dependency issues
(Pan et al., 2024)	Fog computing & IoT energy optimization	Lyapunov-based Dynamic LSTM + PSO (LDLPSO)	Latency and energy inefficiency in edge devices	25% faster response, 9.44% lower energy use	Ignores user mobility and cloud collaboration
(Tabrizchi et al., 2023)	Data center thermal prediction	CNN + Stacked BiLSTM	Hotspot formation and cooling inefficiency	$R^2 = 97.15\%$, RMSE = 0.2892	Scalability in large-scale DCs
(Alizadegan et al., 2025)	Smart grid energy forecasting	Bi-LSTM time series model	Nonlinear and imbalanced energy data	RMSE reduced by 50.6% over SARIMA	Computational complexity, risk of overfitting

2.4 Critical Analysis

Even though the previous LSTM and GRU-based prediction systems resolve general temporal trends, they often fail to react to sharp increases in workloads as they tend to smooth out sharp fluctuations and use sequential memory as opposed to feature-weighting based on their finer details. The existing researches also rely on smaller or snorted datasets which are not representative of real non-stationary fluctuations and hence may not be applicable in real cloud settings. Also, BiLSTM models though useful in learning bidirectional dependencies have higher latency and computational costs, which means that they are not as well suited to real time deployment where operationally reasonable GRU architectures that are light weight and respond quickly are more favored.

2.5 Identified Gaps

In spite of the fact that the previous works used the LSTM, BiLSTM, and GRU models in resource forecasting, the majority of them are dealing with stable or fluctuating workloads and, thus, cannot easily handle sudden LCPUs bursts characteristic of a real cloud. It is also common that many works use small, or noise-filtered datasets, which do not allow them to be robust in non-stationary conditions. Moreover, although there is a body of literature on attention mechanisms in LSTM based designs, less research has been done on attention-enhanced GRU models in cloud CPU forecasting, specifically the use of Squeeze-and-Excitation (SE) recalibration blocks that can emphasize significant temporal features. This presents a chance of analyzing lightweight GRU models with SE attention to achieve better workload sensitivity and reduced latency.

3 Methodology

This chapter explains the entire methodological workflow, beginning with the research design framework and dataset description, followed by data preprocessing, feature engineering, and data visualisation. it then details the creation of rolling-window sequences, scaling and reshaping of data, and the preparation of model inputs for deep learning forecasting.

3.1 Research Design Framework

This study seeks to come up with an energy efficient cloud usage forecast system able to forecast the CPU load, power and carbon emission in real time. The study conforms to the

CRISP-DM model that consists of Data Understanding, Preparation, Modeling, Evaluation, and Deployment. The design focuses on precise time-series prediction, limited latency, and sound future environmental impact prediction. The processed data with 100 VM files of the Bitbrains dataset consists of preprocessing, creating rolling-windows, scaling, and training deep learning models (LSTM, BiLSTM, GRU, and SE-GRU). The computations of power and carbon are computed afterward. All the piping was created in Python so that it would be reproducible and cost-effective.

3.2 Dataset Scope and Statistical Characteristics

The 100 VM traces that will be used in this study are taken out of the Bitbrains GWA-T-12 collection of traces, consisting of 1,750 VMs. The sampled files capture the usage of the CPU at a frequency of one observation every five minutes which is equivalent to a temporal resolution that is sufficient to make short-horizon forecasts. The CPU MHz distribution of these 100 VMs varies approximately between 2,000 MHz (minimum) and more than 7,000 MHz (maximum), median of approximately 4,500 MHz. This sub-set maintains a large enough amount of workload variability to model and the computation scales to explore iterative deep learning experiments.

3.3 Clarification of Preprocessing and Feature Engineering Choices

To enhance the preprocessing decision transparency, only VMs that had 32 CPU cores were filtered to the study to ensure hardware consistency as well as eliminating variation in performance in heterogeneous VM configurations. The empirical testing, as well as the usual short-horizon forecasting practice, led to selecting the rolling-window size of 30 timesteps as the short-term fluctuations of the CPU can be well represented by 30 past observations without adding extra computational load. Though some of the temporal features like weekday, weekend, month and year were also engineered to aid the exploratory analysis, they were not provided as inputs by the neural networks. The models were only trained using univariate CPU usage sequences in order to have a consistent and steady forecasting structure.

3.4 Data Preparation Limitations

The forecasting results are affected by a number of limitations related to data. To begin with, the modelling is only based on univariate CPU consumption since memory, disk and network data points have not been taken into consideration and therefore the model can not reflect the correlations amongst the resources. Second, the limitation of the analysis to 32-core VMs is more suitable in enhancing uniformity but minimises generalisability with heterogeneous cloud environments. Third, a time split train-test split was used so that old samples were used as the training sample and new samples were used as the test sample. This means that no data is leaked although it might cause difficulties when workload changes occurring over time because of concept drift. These are aspects that need to be kept in mind in interpreting the results.

3.5 Data Visualisation

This bar chart is a daily average in the use of CPU in MHz on various days observed as shown in Figure 1. The visualization represents a certain day with the values of CPU utilization approximately between 2,000 and 7,000 MHz. As the chart shows, the CPU consumption trends are highly uneven throughout the tracked period with some days having maximum

utilization of around 6,500 MHz, and some having the low utilization of around 2,500-3,000 MHz.

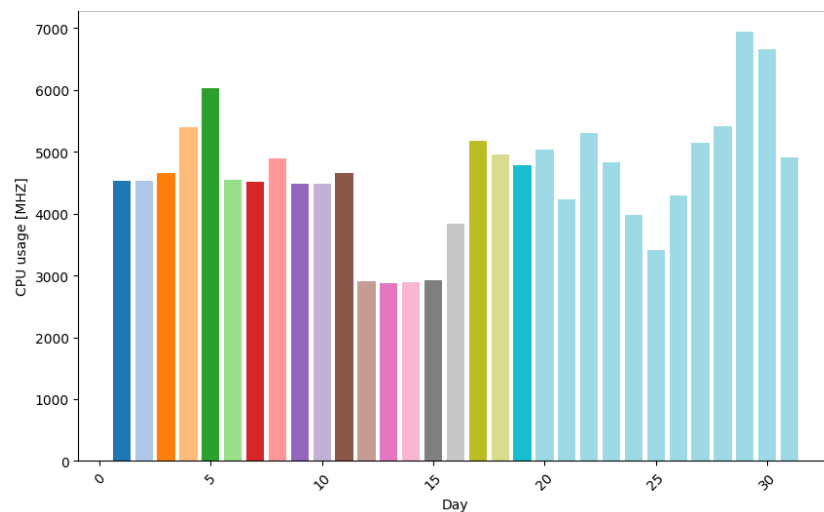


Figure 1: Bar Chart Visualization of Daily Average CPU Usage Patterns

This pie chart shows the distribution of average CPU usage in a two different months in the period of observation as shown in Figure 2. The comparatively equal distribution between these two months shows the consistent CPU demand in the consequent months meaning that there are consistent workload patterns with little deviations. This analysis has a wider time view of resource utilization patterns than the daily level of analysis, which is suitable in interpreting seasonal or cyclical diseases in VM performance measures.

Month Wise Average CPU usage [MHZ]

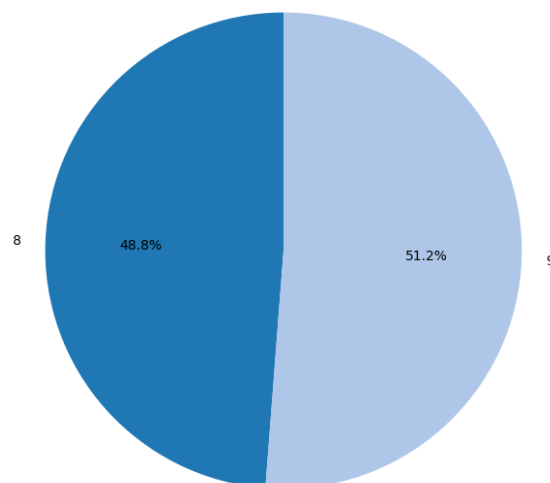


Figure 2: Pie Chart Representation of Monthly Average CPU Usage Distribution

It is a horizontal bar chart that shows average CPU usage in MHz on weekdays of seven days of a week as shown in Figure 3. This workday cycle indicates a cyclical workload cycle where business weekdays will see the highest utilization and weekends will see the lowest cloud enterprise computing patterns and schedules.

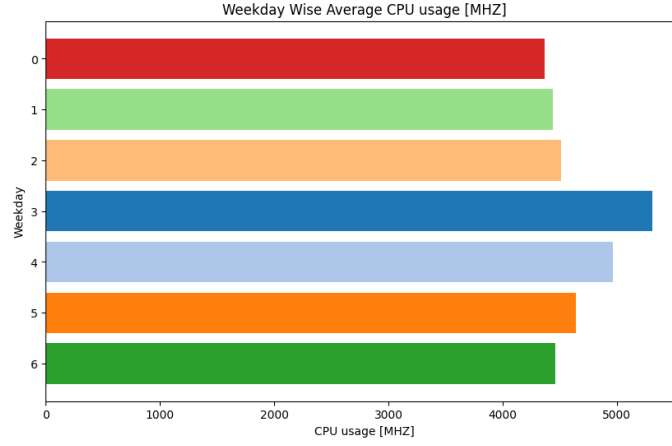


Figure 3: Horizontal Bar Chart of Weekday-Wise Average CPU Usage Analysis

This scatter plot represents the individual days and CPU core-wise average CPU usage in MHZ as shown in Figure 4. The chart shows the points of data involved over 30 days observation in which the range of CPU usage value is between around 2,500 and 7,000 MHZ on the y-axis. The days 20-30 prove to be moderate to high utilization (4,000-6,500 MHZ) indicating the presence of heterogeneous workload allocation throughout the monitoring period and performance discrepancy between CPU cores.

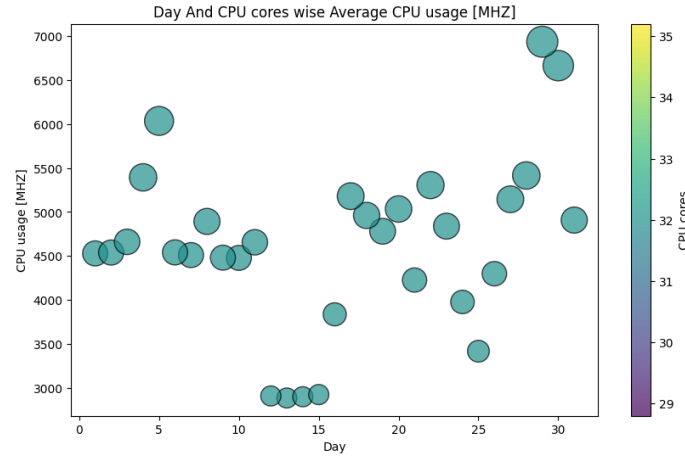


Figure 4: Scatter Plot of Day and CPU Core-Wise Average CPU Usage Distribution

3.6 Rolling Window Sequence Generation for Time-Series Forecasting

A rolling-window technique was applied to convert the time-series data into supervised learning sequences as shown in Figure 5. The custom function `create_dataset()` generated input-output pairs by sliding a fixed window across the dataset. For each position, the function extracted a sequence of past values using dataset and stored it in `dataX`, while the corresponding target value dataset was stored in `dataY`. Using a `time_step` of 30, rolling windows were created separately for both training and testing sets, resulting in `X_train`, `y_train` and `X_test`, `y_test` for model training and evaluation.

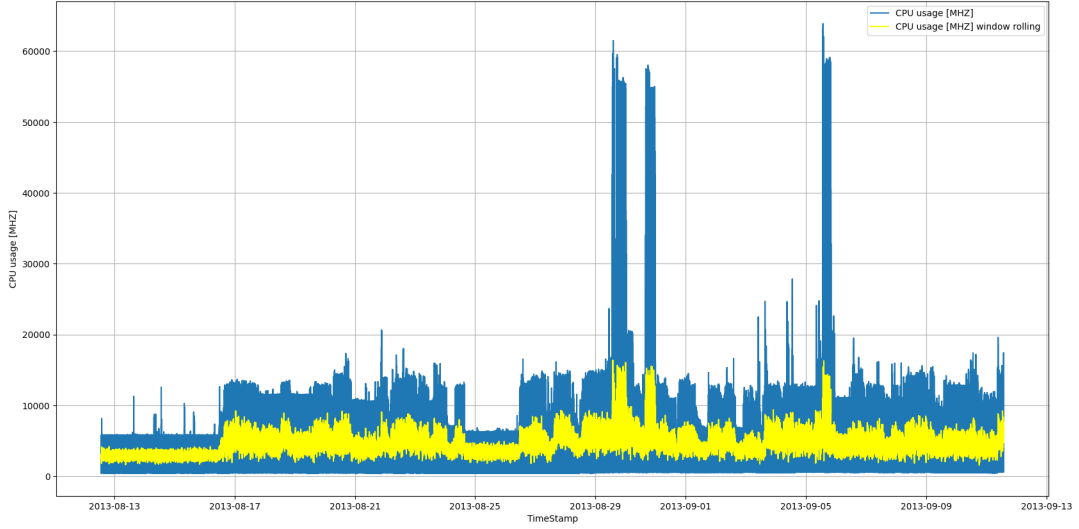


Figure 5: Temporal CPU Usage Trends

3.7 Data Scaling and Input Reshaping for Model Training

In this stage, MinMaxScaler was applied to scale both the feature set and target values within the fixed range of 0 to 1, ensuring uniform numerical behaviour for the deep learning models. Two independent scalers were used: `feature_scaler` for scaling X and `target_scaler` for scaling y . The dataset was divided into a training size of 85,873 samples and a testing size of 36,804 samples, after which the actual scaled shapes obtained for model input were X_{train} (85,842, 30) and y_{train} (85,842, 1), while the testing set produced X_{test} (36,773, 30) and y_{test} (36,773, 1). These dimensions represent the rolling window sequences used for forecasting. After scaling, the input arrays were reshaped into the required three-dimensional format with deep learning architectures such as LSTM, GRU, and SE-GRU that require sequential input representation.

3.8 Training Optimization Using Early Stopping and Learning Rate Scheduling

The forecasting model training set-up was tuned to a standard set of supervised learning to the rolled-window input prepared. The model was initially trained through the Adam optimizer, the loss was mean squared error (MSE) and the evaluation measure was mean absolute error (MAE). The `model.fit` was used with 20 epochs and a batch size of 64 to train the model and (X_{test} , y_{test}) was used to do the validation on the test set.

Two optimisation callbacks were used in order to improve the stability of the models and reduce overfitting: Early Stopping and ReduceLROnPlateau. Early Stopping steadily checked validation loss and terminated training in case of observing no further progress in three epochs, having automatically restored the optimal model weights. This not only made generalisation optimum but also prevented useless computation. Also, ReduceLROnPlateau used the dynamic learning rate, decreasing the learning rate by half when the validation loss flattened, which encouraged a better convergence. All of these methods enhanced the efficiency of training, avoided the divergence and stabilised the learning process, and provided substantial contribution to the overall performance of the SE-GRU forecasting model.

4 Design Specification

This chapter presents the overall system architecture, showing how the components of the forecasting framework interact. It then outlines the tools, technologies, and development environment used to implement the system. Finally, it justifies the proposed SE-GRU model by explaining why it is chosen over traditional DL architectures.

4.1 System Architecture Overview

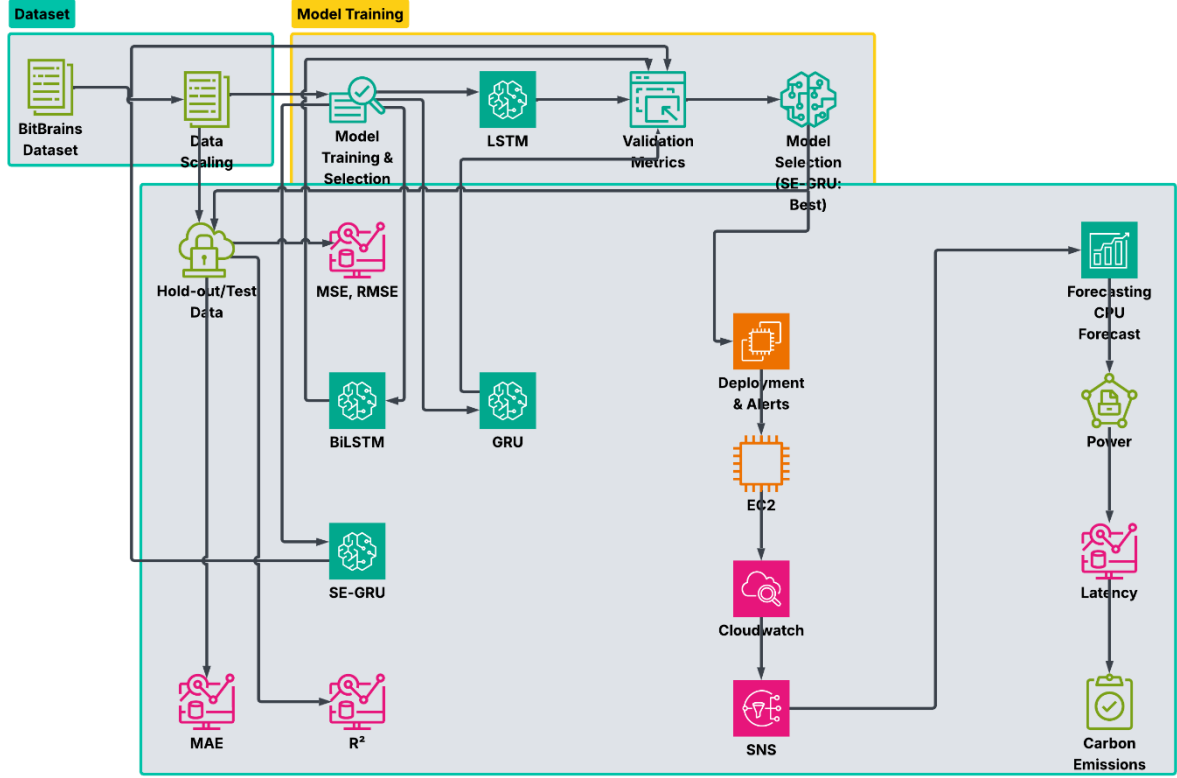


Figure 6: System Architecture Diagram

The proposed system architecture illustrates the end-to-end pipeline for energy-efficient cloud usage forecasting as shown in Figure 6. The BitBrains dataset containing 100 CSV files undergoes comprehensive preprocessing including data cleaning, feature engineering, window rolling with 30 previous timesteps, and min-max scaling. Four deep learning models (LSTM, BiLSTM, GRU, and the proposed SE-GRU) are trained for CPU usage prediction in MHz. The forecasted CPU values are subsequently converted to estimate power consumption in Watts and carbon emissions in kg CO₂. Finally, model performance is rigorously evaluated using multiple metrics to validate forecasting accuracy and environmental impact reduction.

4.2 Tools, Technologies, and Development Environment

This study was developed and experimented with the help of Google Colab that provided a GPU-powered environment that is needed to effectively train deep learning models. This has been implemented in Python and with the help of major libraries like Pandas, NumPy, and Glob used in the management and merging of the Bitbrains dataset files. The Matplotlib, Seaborn, and Plotly were applied to visual analyse the CPU behaviour and explore the trends.

The proposed SE-GRU architecture was built using TensorFlow and Keras, with the addition of LSTM and Bidirectional LSTM, GRU, Dense, Dropout, and attention-related components for the proposed SE-GRU architecture. Other tools such as MinMaxScaler, train-test-split, EarlyStopping and TensorFlow backend functions provided efficiency in preprocessing, stable optimisation in the training process and clean up sessions.

4.3 Justification of the Proposed Model

The proposed SE-GRU model is introduced to overcome the limitations observed in traditional deep learning architectures such as LSTM, BiLSTM, and standard GRU when predicting highly dynamic cloud CPU usage patterns. While these models capture sequential behaviour, they often treat all time-step features equally, resulting in reduced sensitivity to sudden workload spikes and irregular fluctuations common in VM environments. By integrating a Squeeze-and-Excitation (SE) attention mechanism with GRU, the proposed model adaptively recalibrates feature importance, enabling it to highlight relevant temporal patterns and suppress noise. This targeted feature weighting strengthens the model's ability to learn complex CPU behaviour more efficiently than conventional architectures. Additionally, the GRU backbone ensures faster training and lower latency, making the model suitable for real-time forecasting scenarios. The improved accuracy, stability, and responsiveness of SE-GRU justify its use as a more effective approach for cloud workload prediction and sustainability analysis.

5 Implementation

This chapter explains the implementation of all models which includes detailed architectures for LSTM, BiLSTM, GRU and the proposed SE-GRU model. It then describes the procedure used for generating future forecasts of CPU usage, power consumption and carbon emissions.

5.1 Deep Learning Model Implementation

5.1.1 LSTM Model

LSTM model was trained on a Sequential architecture composed of two layers of stacked LSTM to learn the patterns of temporal CPU (Patel and Kushwaha, 2022) utilisation based on Bitbrains dataset. The initial model unit had 128 units, and the return-sequences=True option was added to propagate the entire sequence to the second unit and then a dropout unit (0.5) was added so that overfitting could be minimized. A second LSTM layer of 64 units and sequence return was introduced to enhance further temporal learning which was again regularized using dropout. The Layer Normalization was employed to stabilize the internal activations and GlobalAveragePooling1D layer was employed to reduce the temporal output into one representative vector (Zafar et al., 2024). This representation was processed by a dense layer with 64 units and ReLU activation and then a dropout was performed to achieve robustness. Lastly, the output of the regression was produced by a single-neuron dense layer. The model has been obtained with the Adam optimizer, where MSE was set as a loss factor and MAE was set as an extra performance measure. EarlyStopping automatically recovered the optimal weights in the course of training.

5.1.2 BiLSTM Model

The BiLSTM model was designed in such a way that both forward and reverse temporal relationships found in the time-series workload of CPUs. An initial architecture was based on a Bidirectional wrapper on top of an LSTM layer of 128 units (Veerabaku et al., 2023) so that the model can simultaneously learn the patterns in the past and in the future. This was repeated

by 0.5 dropout layer to avoid overfitting. The second Bidirectional LSTM layer containing 64 units also improved the temporal representation learning, and dropout was also used to stabilize the training. Layer nonmonotropy enhanced the gradient motion and minimized internal covariate shift. The sequence output was condensed into one aggregate feature vector by a GlobalAveragePooling1D layer which was then processed by a 64-unit dense layer using ReLU as the activation function and dropout as a regularizer. The prediction of the next CPU usage value was made by the output layer, which had a single neuron. The model was trained on the Adam optimizer whereby the loss function is MSE and the evaluation metric MAE. EarlyStopping terminated training in the event of failure and reinstated the best weights.

5.1.3 GRU Model

GRU model was used as stacked architecture that is effective at learning sequential patterns of CPU usage at reduced computational price than LSTMs (Ullah et al., 2022). The model started with a GRU layer that has 128 units and with the return sequence set to true to pass the output into further GRU layers. A dropout layer (0.5) was added in order to reduce overfitting. A second GRU layer comprising 128 units further discovered temporal information which was once again regularized using dropout. The final sequence encoding was produced by another GRU layer having 64 units by returning sequences with return sequences=False, producing a fixed length feature representation. This vector went through a dense layer of 64 units to refine learned patterns and this was followed by a dropout to stabilize it. The last thick layer having one neuron gave back the regression output. The Adam minimizer compiled this model, and the training loss was MSE, and another metric was the MAE. EarlyStopping was used to monitor model training in order to achieve the best convergence and also avoid redundant epochs.

5.1.4 GRU+ squeeze and excitation Attention Model

SE-GRU model combines deep GRU layers and Squeeze-and-Excitation (SE) attention block (Wang et al., 2024) to enhance feature recalibration and bettering of CPU usage forecasting. The model starts with a GRU layer of 128 units, which produces complete sequences and a dropout layer concludes the model. The sequence is again run through a second 128-unit GRU layer that is also regularized with dropout. The output is then fed into the custom SE attention block which is the GlobalAveragePooling1D which does the squeeze operation. Two thick layers (the reduction and the expansion) calculate channel-wise weights with the help of the sigmoid activation. These weights are redesigned and used to emphasis significant temporal properties together with the GRU output. The last GRU layer of 64 units consists of the recalibrated sequence, which is condensed into one single vector and followed by dropout, as well as a densely connected final representation learning layer of 64 units. The output of the regression is generated by an individual dense neuron. The model employs Adam and MSE and MAE and it performs with EarlyStopping in the training process.

Pseudocode 1: SE Block Mechanism

```

s = GlobalAveragePooling(X)
z = Dense(features/16, relu)(s)
z = Dense(features, sigmoid)(z)
for each channel i in features:
    if z[i] > 0.5:
        weight[i] = z[i]
    else:
        weight[i] = z[i] * 0.5

```

```
# Recalibrate GRU outputs
X_recalibrated = X * weight
```

5.2 Future Forecasting of CPU Usage, Power Consumption, and Carbon Emissions

The future forecasting stage involved generating short-term predictions for CPU usage and translating these predictions into corresponding power consumption and carbon emission values. The process began by extracting the last 30 actual CPU usage values from the dataset, which were scaled using the previously fitted target scaler. This 30-step sequence served as the initial input for the model. Using an auto-regressive prediction loop, the model produced 15 future CPU usage values, each time updating the input sequence by sliding the window forward and appending the newly predicted value. All predictions were then inverse-scaled to obtain actual CPU usage in MHz. New timestamps were generated at 5-minute intervals to align with the forecasting horizon, creating a structured output dataframe. Next, predicted CPU usage was converted into CPU percentage relative to the maximum recorded MHz. Using the linear power model the corresponding power consumption for each timestep was calculated. Energy usage was derived by converting each 5-minute interval into kWh, and carbon emissions were computed using a factor of 708 g CO₂ per kWh. Finally, all results were visualized through individual plots for CPU forecast, power consumption, and carbon footprint.

5.3 Real-Time Cloud Inferencing Setup Using AWS

To test the model in a live environment there is a real-time inferencing workflow was implemented using AWS EC2. Since real user workloads were not available so synthetic workload data was generated on the EC2 instance to replicate real-time CPU fluctuations. This synthetic stream acted as input and was checked every 5 minutes which aligns with the forecasting interval used during model training. Figure 7 shows the configuration of the Amazon SNS used in the alerting workflow of the system. The trained SE-GRU model has integrated with this live data flow and CPU forecasts were computed along with predicted power consumption and carbon emissions. If the predicted CPU utilization exceeded a defined high-load threshold then the system triggered an AWS SNS notification on email as shown in Figure 8. This setup simulates a CloudWatch-like operational environment and shows how the model can be used for automated monitoring and alert-based decision making.

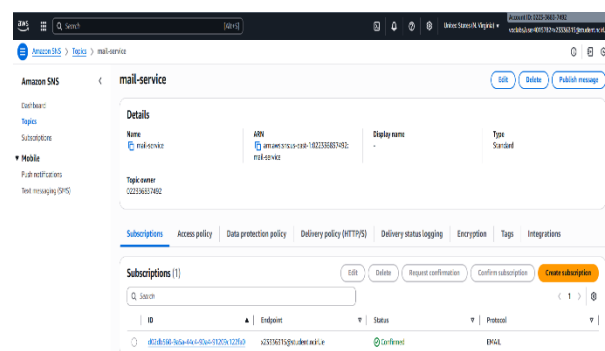


Figure 7: AWS SNS Setup

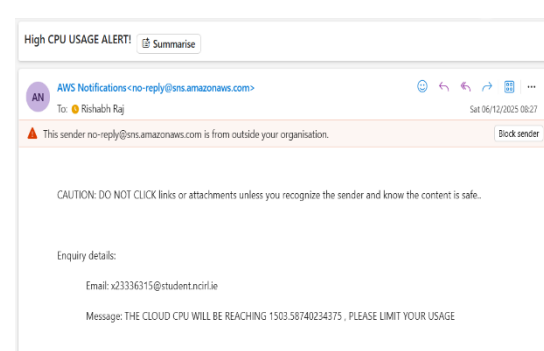


Figure 8: Email Alert Triggered

6 Evaluation

This chapter presents the evaluation of the proposed forecasting system through power–carbon calculations, accuracy testing, latency–throughput analysis, and visual comparison of actual versus predicted CPU usage. It reports results from three experiments and compares all models, highlighting the superior performance of SE-GRU across environmental, predictive, and computational metrics.

6.1 Power Consumption and Carbon Emission Calculation Method

To extend the forecasting capability beyond CPU usage, this study converts the model’s predicted CPU load into corresponding power consumption and carbon emission estimates. This analysis helps evaluate the environmental impact of future server workloads. The process begins by transforming the predicted CPU usage values (in MHz) into normalized CPU percentages. For this, the maximum CPU usage recorded in the dataset is used as a reference benchmark. The conversion follows the expression:

$$\text{CPU \%} = (\text{Predicted MHz} / \text{Maximum MHz}) \times 100,$$

ensuring all values are within a valid 0–100% range through clipping.

Once CPU percentage is derived, a linear server power model is applied, assuming 50W as the idle power and 150W as the full-load power. Power consumption for each forecasted point is calculated as:

$$\text{Power (W)} = P_IDLE + (\text{CPU\%} / 100) \times (P_MAX - P_IDLE).$$

This formula reflects the proportional relationship between CPU load and power usage in typical cloud servers.

Next, the predicted power values are converted into energy consumption in kWh by considering the 5-minute forecasting interval. Since 5 minutes = 5/60 hours, energy for each step is computed using:

$$\text{Energy (kWh)} = (\text{Power} \times \text{Time in hours}) / 1000.$$

To estimate the environmental impact, a carbon emission factor of 708 grams of CO₂ per kWh is applied, representing average coal-based electricity generation. Carbon output is calculated as:

$$\text{CO}_2 \text{ (grams)} = \text{Energy (kWh)} \times 708,$$

followed by conversion into kilograms.

This conversion pipeline allows the model’s CPU forecasts to translate into actionable sustainability insights regarding power demand and carbon footprint.

6.2 Train–Test Split Method

This is a time-series forecasting task so the dataset has split to avoid any data leakage. The first 70% of the time-ordered data has used for training while the remaining 30% has kept strictly as for future test data. There is no random shuffling has applied. This secures that the model is evaluated only on unseen future timestamps which shows a real cloud forecasting scenario.

6.3 Experiment 1: Power & Carbon Consumption Analysis of Forecasted Workloads

The table 2 summarizes the predicted energy and carbon impact of the CPU usage estimates of the four models under analysis. The behaviour of LSTM and BiLSTM is similar whereby both generate relatively higher power and carbon values, which implies that they predict with moderate precision. GRU is showing definite progress and its average energy and carbon levels decreased due to the enhanced ability to learn about the temporal patterns. The proposed SE-GRU model is the most optimal one with the lowest power and carbon outputs which is indicative of smooth and more accurate workload forecasting. The Squeeze-and-Excitation attention mechanism is also significant as it highlights the features that are relevant in time-series so that the model is not overestimated about the CPU load. Consequently, SE-GRU produces forecasts that are translated into efficient power use and carbon reduction, and thus makes it a strong prediction tool with respect to its ability to forecast sustainably as well as its ability to predict effectively. The experiment proves the excellence of the proposed model regarding the environmental-conscious computing.

Table 2: Summary of Power and Carbon Consumption Predicted by All Models

Model	Avg Power (Wh)	Max Power (Wh)	Avg Carbon (kg)	Max Carbon (kg)
LSTM	51.2597	51.2898	0.003024	0.003026
BiLSTM	51.2433	51.2639	0.003023	0.003025
GRU	50.6441	51.0132	0.002988	0.003010
SE-GRU (Proposed)	50.0792	50.6599	0.002955	0.002989

6.4 Experiment 2: Model Accuracy Evaluation

This table 3 summarizes quantitative prediction accuracy of all the four deep learning models that were been tested on the prediction of CPU usage. The performance of LSTM is reasonable, and it still demonstrates higher error values, which means that it has weaknesses in the representation of long-term dependencies. BiLSTM enhances the accuracy of the baseline by getting to know two-way temporal patterns, which lead to lesser MSE, RMSE and MAE. GRU also improves performance through the provision of a more efficient and easier to manipulate gating mechanism which results in better generalization and increased R. The best results are provided by the proposed SE-GRU model that shows the lowest values of MSE and MAE and the highest value of R2. This enhancement can be explained by the fact that the Squeeze-and-Excitation attention block is integrated to focus on beneficial temporal features and eliminate noise in an adaptive way. This will help the model to make more stable predictions with very

little variation with the real values. All in all, Experiment 2 supports the idea that SE-GRU is the most robust architecture in terms of accuracy of all tested architectures.

Table 3: Model Performance Metrics (MSE, RMSE, R², MAE)

Model	MSE	RMSE	R ² Score	MAE
LSTM	2.65026e-05	0.005148	0.94479	0.002063
BiLSTM	2.13827e-05	0.004624	0.95545	0.001491
GRU	1.58983e-05	0.003987	0.96688	0.002755
SE-GRU (Proposed)	9.48341e-06	0.003080	0.98024	0.001049

6.5 Interpretation of Results

The SE-GRU model has been reduced prediction noise by temporal features through the SE attention block focusing on relevant type of patterns while suppressing less informative ones. However this can slightly over-smooth sudden workload spikes which potentially underestimate extreme CPU peaks. The high R² (0.98) shows strong overall fit but may suggest minor overfitting to the training set. Baseline models (LSTM, BiLSTM, GRU) mainly fail to capture abrupt spikes which leads to larger errors during high-utilization events.

6.6 Experiment 3: Latency and Throughput Analysis

This table 4 compares the computational efficiency of all models based on their latency and throughput which play a crucial role in real time prediction of the cloud resources. The LSTM model has a high throughput because of its simpler architecture although it has moderate latency. BiLSTM is the slowest since it requires processing in both directions to enhance its calculations which make its throughput to be minimal. GRU strikes a balance between speed, accuracy, more than LSTM and BiLSTM, has better throughput at quite low latency. The proposed SE-GRU model has one of the lowest latencies, and it has high throughput hence is applicable in real time implementations. Attention mechanism is not very heavy and it brings a great improvement in prediction. These findings indicate that SE-GRU is more accurate and at the same time computationally lightweight which makes it an ideal option in large-scale cloud forecasting systems where speed and consistency are also a major consideration.

Table 4: Model Latency and Throughput Comparison

Model	Latency per Prediction (ms)	Throughput (Pred/sec)
LSTM	61.1828	11982.04
BiLSTM	77.1841	7884.32
GRU	69.6403	10185.83
SE-GRU (Proposed)	61.0366	10504.09

6.7 Experiment 4: Real-Time Cloud Inferencing and Alert Validation

This experiment evaluates the real-time deployment behavior of the forecasting system when integrated with an AWS EC2 instance. Synthetic CPU usage values have generated on EC2 to replicate a realistic fluctuating workload pattern. Each batch of incoming workload data was passed through the deployed SE-GRU model which produced predictions for CPU usage along with corresponding power consumption and carbon emission values.

A predefined threshold has used to represent a high-CPU event. When the SE-GRU model predicted CPU utilization levels which is very high then AWS SNS triggers. Email alerts have successfully triggered during high-forecast intervals by confirming correct integration between prediction logic and cloud alerting. The deployment workflow also reflects the behavior of Amazon CloudWatch by simulating a continuous monitoring cycle driven by synthetic workload generation.

The results confirm that the model is capable not only of accurate offline forecasting but also of functioning autonomously in a cloud environment with live inferencing, periodic evaluation and proactive alerting. This shows that the SE-GRU forecasting framework can be operationalized for real-time resource optimization and supports the potential for future full-scale CloudWatch metric integration and automated scaling decisions.

6.8 Actual vs Predicted Graphs

The actual vs Predicted graphs of the CPU usage prediction of the LSTM model are shown in Figure 9. The visualization shows time-changing trends in the dataset with real values and forecasted values. The above graph indicates that LSTM is good at capturing the trends at baseline, however, there were instances of prediction spikes, which is an indication that it cannot accurately predict abrupt changes in the CPU load. The model is overall found to be reasonably accurate during stable periods but it fails at responding to sudden workload pattern shifts.

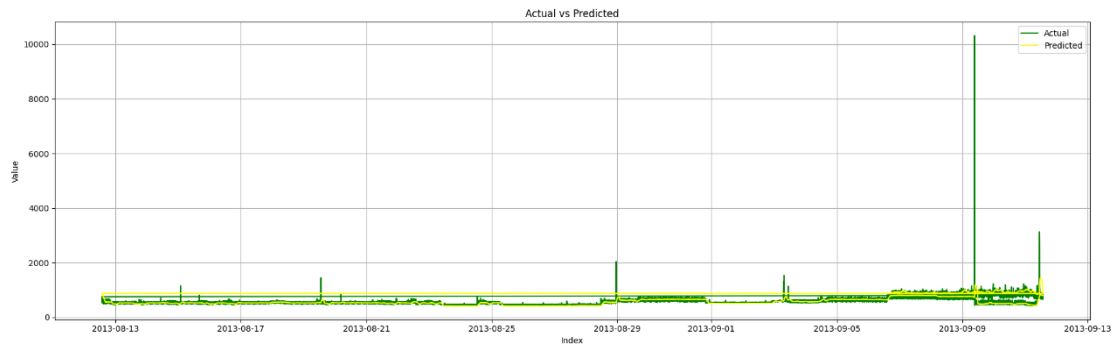


Figure 9: Actual vs Predicted Graphs

Figure 10 shows the Actual vs Predicted graphs of the CPU usage forecasting of the BiLSTM model. The bidirectional architecture exhibits better representation of temporal relationships than that of unidirectional LSTM with closer approximations of actual and predicted values. BiLSTM demonstrates higher performance in case of moderate fluctuations and more precise tracking of the baseline. Nevertheless, sharp CPU spikes are still difficult, implying that the use of bidirectional context is inadequate to record high levels of workload deviations without other systems.

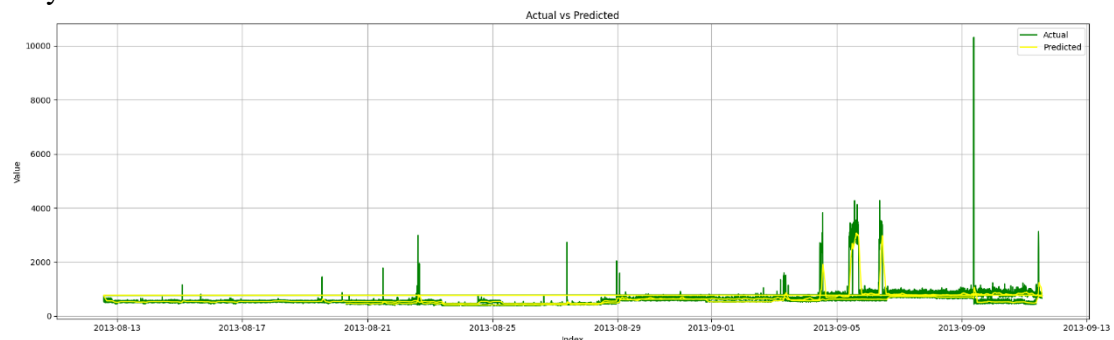


Figure 10: Actual vs Predicted Graphs

Figure 11 shows the Actual vs Predicted graphs of the forecasting performance of the CPU usage of the GRU model. GRU architecture has better efficiency in learning the temporal patterns with lesser complexity than the LSTM variants. Large prediction errors are observed when the load on the CPU suddenly increases, which suggests that GRU is not able to cope with the anomalies in the workload peaks regardless of the benefits associated with its gating mechanism.

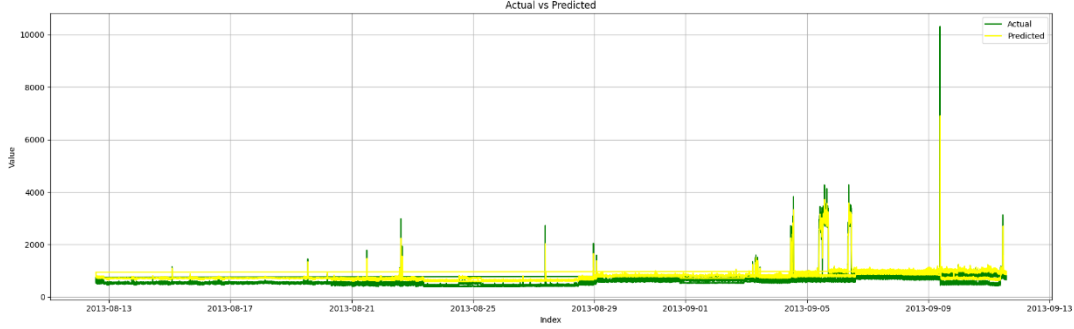


Figure 11: Actual vs Predicted Graphs

The Actual vs Predicted graphs of the GRU Squeeze-and-Excitation (SE) Attention model are shown in Figure 12. This improved architecture has a better forecasting accuracy with the predicted values closely matched with the true CPU usage at a variety of workload patterns. The SE attention mechanism makes good use out of major features and replicates noise to capture sudden load spike and fluctuations much better. The model significantly improves baseline models, which supports the usefulness of attention integration in predicting CPU usage in clouds.

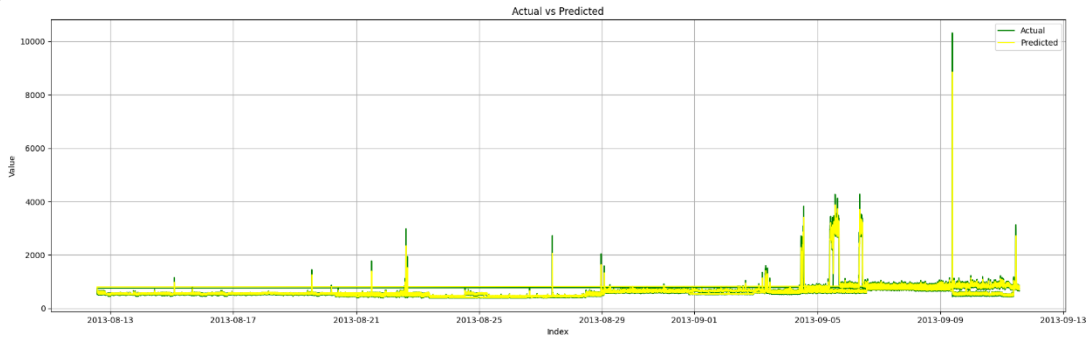


Figure 12: Actual vs Predicted Graphs

6.9 Impact of Squeeze-and-Excitation Block

An ablation study was done to determine the role played by the Squeeze-and-Excitation (SE) block by comparing regular GRU to SE-GRU. Findings indicate that SE-GRU represents a better score in terms of prediction error and higher R² scores, which prove that adaptive feature recalibration enhances sensitivity to workload changes. This confirms that performance improvement is largely due to the SE attention mechanism more than the architectural depth.

6.10 Comparative Discussion with Baseline

Compared to the baseline OM-FNN framework, which focuses on multi-resource prediction and autoscaling for VM placement, the proposed SE-GRU study provides a more accurate and

computationally efficient CPU-usage forecasting solution. The baseline paper given by (Saxena and Singh, 2021) improves utilization and power efficiency through multi-objective autoscaling, but it requires complex multi-stage processing and higher overhead. In contrast, SE-GRU achieves the lowest MSE, RMSE, and MAE with the highest R^2 , offering stable predictions and reduced latency as shown in Table 5. Furthermore, my proposed study extends forecasting benefits by converting CPU predictions into power and carbon estimations, enabling sustainability insights that the baseline paper does not provide.

Table 5: Comparison Table (Baseline vs Proposed Study)

Aspect	Baseline Paper given by (Saxena and Singh, 2021)	My Proposed Study (SE-GRU)
Prediction Scope	Predicts multiple resources (CPU, Memory).	Predicts CPU usage only.
Model Type	OM-FNN with autoscaling + VM placement.	SE-GRU deep learning model.
Accuracy Focus	Improves utilization & power by VM-level decisions.	Lowest error (MSE, MAE) & highest R^2 for CPU forecasting.
Sustainability Output	Reduces power via placement decisions.	Direct power & carbon prediction from CPU forecasting.
Complexity	Multi-stage, high overhead.	Lightweight, low-latency forecasting architecture.

6.11 Discussion

The proposed SE-GRU forecasting infrastructure is not specific to a CPU workload and a single cloud provider. As the model is trained on generic time series trends, it can be generalized to other cloud resources including memory, disk I/O, and network traffic using retraining. Whereas forecasting is region insensitive, carbon estimation must use region-specific emission factors. The architecture is not cloud-specific, and it can be used with AWS, Azure, or Google Cloud.

7 Conclusion and Future Works

7.1 Conclusion

The present study has shown that deep learning models can be useful in predicting CPU loads in clouds, and SE-GRU is the highest accuracy and lowest errors. Through analysis, easy patterns of workload are highlighted, such as weekday weekend cycles and stable and unstable CPU utilization. These results indicate that real-time decision-making can be assisted by energy-aware forecasting to enable proactive workload balancing, power optimization, and carbon emissions reduction that may be possible. Attention mechanisms are implemented in the SE-GRU model which enables more accurate detection of abrupt spikes in workload giving the model practical utility to automated monitoring systems. On the whole, the research holds an understanding of the short-term resources behavior and the operational usefulness of predictive modeling in terms of sustainable cloud management.

7.2 Limitations

There are a number of limitations in this study, which must be mentioned. The actual 100 VM traces were utilised as opposed to the entire 1,750 VM dataset, which can decrease the generalizability. The univariate forecasting was based only on CPU usage and did not use other metrics of resources like memory or network load. The power usage model used was simple

linear model, and this might not be representative of data center observed heterogeneous behavior or dynamic energy management. Hyperparameter training was not done and the model was also trained in only 20 epochs which could limit the convergence. This was limited to 15 steps of forecasting with a short term on the line and no statistical significance testing was done to validate model improvements on a formal basis. All of these limitations reduce the generalizability of the results to large or changing cloud settings, which are ways to take future research a step further.

7.3 Future Work

This study can be expanded in various ways in future studies. The holistic cloud resource management can be enhanced by multi-resource forecasting which involves memory, network and storage measures. SE-GRU multivariate can be used to predict more robustly with multivariate dependence mechanisms of resources. Monitoring systems Practical applicability will be further confirmed by deploying real-time with operational workloads. The comparison of recurrent architectures with transformer-based models could provide the existence of performance improvements in forecasting accuracy and latency. Also, nonlinear or data center specific power models will be utilized to improve the estimation of environmental impact. Generalization and reliability can be enhanced by increasing size of datasets and prediction horizon and hyperparameter tuning, and make energy-efficient cloud operations a viable development in large scale deployment.

References

- Al Kez, D., Foley, A.M., Lavery, D., Del Rio, D.F., Sovacool, B., 2022. Exploring the sustainability challenges facing digitalization and internet data centers. *J. Clean. Prod.* 371, 133633. <https://doi.org/10.1016/j.jclepro.2022.133633>
- Alizadegan, H., Rashidi Malki, B., Radmehr, A., Karimi, H., Ilani, M.A., 2025. Comparative study of long short-term memory (LSTM), bidirectional LSTM, and traditional machine learning approaches for energy consumption prediction. *Energy Explor. Exploit.* 43, 281–301. <https://doi.org/10.1177/01445987241269496>
- Baburao, D., Pavankumar, T., Prabhu, C.S.R., 2023. Load balancing in the fog nodes using particle swarm optimization-based enhanced dynamic resource allocation method. *Appl. Nanosci.* 13, 1045–1054. <https://doi.org/10.1007/s13204-021-01970-w>
- Bal, P.K., Mohapatra, S.K., Das, T.K., Srinivasan, K., Hu, Y.-C., 2022. A Joint Resource Allocation, Security with Efficient Task Scheduling in Cloud Computing Using Hybrid Machine Learning Techniques. *Sensors* 22, 1242. <https://doi.org/10.3390/s22031242>
- Gures, E., Shayea, I., Ergen, M., Azmi, M.H., El-Saleh, A.A., 2022. Machine Learning-Based Load Balancing Algorithms in Future Heterogeneous Networks: A Survey. *IEEE Access* 10, 37689–37717. <https://doi.org/10.1109/ACCESS.2022.3161511>
- Katal, A., Dahiya, S., Choudhury, T., 2023. Energy efficiency in cloud computing data centers: a survey on software technologies. *Clust. Comput.* 26, 1845–1875. <https://doi.org/10.1007/s10586-022-03713-0>
- Long, S., Li, Yuan, Huang, J., Li, Z., Li, Yanchun, 2022. A review of energy efficiency evaluation technologies in cloud data centers. *Energy Build.* 260, 111848. <https://doi.org/10.1016/j.enbuild.2022.111848>

- Maenhaut, P.-J., Volckaert, B., Ongenae, V., De Turck, F., 2020. Resource Management in a Containerized Cloud: Status and Challenges. *J. Netw. Syst. Manag.* 28, 197–246. <https://doi.org/10.1007/s10922-019-09504-0>
- Mahdizadeh, M., Montazerolghaem, A., Jamshidi, K., 2024. Task scheduling and load balancing in SDN-based cloud computing: A review of relevant research. *J. Eng. Res.* S2307187724002773. <https://doi.org/10.1016/j.jer.2024.11.002>
- Masdari, M., Khoshnevis, A., 2020. A survey and classification of the workload forecasting methods in cloud computing. *Clust. Comput.* 23, 2399–2424. <https://doi.org/10.1007/s10586-019-03010-3>
- Ounifi, H.-A., Gherbi, A., Kara, N., 2022. Deep machine learning-based power usage effectiveness prediction for sustainable cloud infrastructures. *Sustain. Energy Technol. Assess.* 52, 101967. <https://doi.org/10.1016/j.seta.2022.101967>
- Pan, S., Huang, C., Fan, J., Shi, Z., Tong, J., Wang, H., 2024. Optimizing Internet of Things Fog Computing: Through Lyapunov-Based Long Short-Term Memory Particle Swarm Optimization Algorithm for Energy Consumption Optimization. *Sensors* 24, 1165. <https://doi.org/10.3390/s24041165>
- Patel, E., Kushwaha, D.S., 2022. A hybrid CNN-LSTM model for predicting server load in cloud computing. *J. Supercomput.* 78, 1–30. <https://doi.org/10.1007/s11227-021-04234-0>
- Paul, S.G., Saha, A., Arefin, M.S., Bhuiyan, T., Biswas, A.A., Reza, A.W., Alotaibi, N.M., Alyami, S.A., Moni, M.A., 2023. A Comprehensive Review of Green Computing: Past, Present, and Future Research. *IEEE Access* 11, 87445–87494. <https://doi.org/10.1109/ACCESS.2023.3304332>
- Qian, J., Wang, Y., Wang, Xiaoxue, Zhang, P., Wang, Xiaofeng, 2023. Load balancing scheduling mechanism for OpenStack and Docker integration. *J. Cloud Comput.* 12, 67. <https://doi.org/10.1186/s13677-023-00445-3>
- Saxena, D., Singh, A.K., 2021. A proactive autoscaling and energy-efficient VM allocation framework using online multi-resource neural network for cloud data center. *Neurocomputing* 426, 248–264. <https://doi.org/10.1016/j.neucom.2020.08.076>
- Sharifzadeha, M., Malekpoura, H., Shojab, E., 2022. Cloud Computing and Its Impact on Industry 4.0: An Overview, in: Sharifzadeh, M. (Ed.), *Industry 4.0 Vision for Energy and Materials*. Wiley, pp. 99–120. <https://doi.org/10.1002/9781119695868.ch4>
- Sireesha, P., S, V.P., Govindarajan, M., Rajan, S., Rajakumareswaran, V., 2024. Revolutionizing Cloud Resource Allocation: Harnessing Layer-Optimized Long Short-Term Memory for Energy-Efficient Predictive Resource Management. *EAI Endorsed Trans. Energy Web* 11. <https://doi.org/10.4108/ew.6505>
- Tabrizchi, H., Razmara, J., Mosavi, A., 2023. Thermal prediction for energy management of clouds using a hybrid model based on CNN and stacking multi-layer bi-directional LSTM. *Energy Rep.* 9, 2253–2268. <https://doi.org/10.1016/j.egyr.2023.01.032>
- Ullah, F.U.M., Ullah, A., Khan, N., Lee, M.Y., Rho, S., Baik, S.W., 2022. Deep Learning-Assisted Short-Term Power Load Forecasting Using Deep Convolutional LSTM and Stacked GRU. *Complexity* 2022, 2993184. <https://doi.org/10.1155/2022/2993184>
- Veerabaku, M.G., Nithiyanantham, J., Urooj, S., Md, A.Q., Sivaraman, A.K., Tee, K.F., 2023. Intelligent Bi-LSTM with Architecture Optimization for Heart Disease Prediction in WBAN through Optimal Channel Selection and Feature Selection. *Biomedicines* 11, 1167. <https://doi.org/10.3390/biomedicines11041167>
- Vijay, R., Sree, T.R., 2023. Resource Scheduling and Load Balancing Algorithms in Cloud Computing. *Procedia Comput. Sci.* 230, 326–336. <https://doi.org/10.1016/j.procs.2023.12.088>

- Wang, Y., Xie, W., Liu, C., Luo, J., Qiu, Z., Deconinck, G., 2024. Forecast of coal consumption in salt lake enterprises based on temporal gated recurrent unit network with squeeze-and-excitation attention. *Energy* 299, 131405. <https://doi.org/10.1016/j.energy.2024.131405>
- Xie, Z., 2023. Data Center Based on Cloud Computing Technology. *IJIIS Int. J. Inform. Inf. Syst.* 6, 31–37. <https://doi.org/10.47738/ijiis.v6i1.128>
- Xu, M., Toosi, A.N., Buyya, R., 2021. A Self-Adaptive Approach for Managing Applications and Harnessing Renewable Energy for Sustainable Cloud Computing. *IEEE Trans. Sustain. Comput.* 6, 544–558. <https://doi.org/10.1109/TSUSC.2020.3014943>
- Zafar, A., Saba, N., Arshad, A., Alabrah, A., Riaz, S., Suleman, M., Zafar, S., Nadeem, M., 2024. Convolutional Neural Networks: A Comprehensive Evaluation and Benchmarking of Pooling Layer Variants. *Symmetry* 16, 1516. <https://doi.org/10.3390/sym16111516>
- Zheng, H., Lu, Y., Sun, Z., Panneerselvam, J., Sun, X., Liu, L., 2024. Energy optimisation in cloud datacentres with MC-TIDE: Mixed Channel Time-series Dense Encoder for workload forecasting. *Appl. Energy* 374, 123903. <https://doi.org/10.1016/j.apenergy.2024.123903>
- Zuo, H.-M., Qiu, J., Jia, Y.-H., Wang, Q., Li, F.-F., 2022. Ten-minute prediction of solar irradiance based on cloud detection and a long short-term memory (LSTM) model. *Energy Rep.* 8, 5146–5157. <https://doi.org/10.1016/j.egyr.2022.03.182>