National College of Ireland

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: 23336315

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: 23336315

Programme: MSc in Cloud Computing **Year:** 2025

Module: Research Project

Supervisor: Rashid Mijumbi

Submission Due Date: 11-12-25

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

Word Count: 777 **Page Count** 5

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: 11-12-25

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):	

Configuration Manual: Energy-Efficient Cloud Usage Forecasting Using SE-GRU for Power & Carbon Emission Reduction

1. Introduction

As we know Modern cloud datacentres host thousands of virtual machines (VMs) running enterprise applications, scientific workloads, and real-time services. These environments require intelligent resource management to reduce operational costs, energy consumption, and associated carbon emissions. However, CPU load in cloud systems is highly dynamic—fluctuating due to varying workloads, user activity patterns, and background services—which makes accurate forecasting a challenge.

Traditional statistical methods struggle to capture these complex temporal patterns. Deep learning architectures such as LSTM, BiLSTM, and GRU offer improved capabilities for sequential data modeling. Yet, even these models may fail to fully capture subtle temporal dependencies.

This project introduces an enhanced **SE-GRU (Squeeze-and-Excitation GRU)** architecture, which integrates channel-wise attention to strengthen sequence learning. The objective is not only to forecast CPU usage (in MHz) accurately but also to translate predictions into:

- power consumption (W),
- energy consumption (kWh), and
- carbon emissions (g, kg).

This contributes toward greener cloud computing and sustainability-driven system optimization.

2. Research Question

Primary Research Question

How effectively can deep learning models forecast CPU usage in a cloud environment, and does incorporating a Squeeze-and-Excitation (SE) attention mechanism into a GRU network significantly improve prediction accuracy and reduce errors in estimating power consumption and carbon emissions?

Explanation of the Research Question

Cloud resource usage depends on:

- temporal workload patterns (hourly, daily variability),
- VM-specific performance characteristics,

- underlying hardware performance capacity.

This study investigates two core aspects:

(A) Temporal Learning Performance

Does the SE-GRU model enhance the identification of important historical CPU load patterns using channel-wise recalibration?

(B) Impact on Energy & Emission Estimation

Does improved forecasting accuracy lead to:

- more reliable power usage estimation,
- more accurate energy consumption modeling,
- reduced error in carbon footprint calculations?

3. Environment Setup

Correct environment configuration ensures consistent and reproducible experimentation.

3.1 Recommended Hardware

Component	Recommended
GPU	NVIDIA T4 / RTX 3060 / A100
RAM	Minimum 12 GB, recommended 16–32 GB
CPU	Quad-core or better
Storage	~10 GB for dataset + models

Deep learning models benefit greatly from GPU acceleration, especially when training on time series windows.

3.2 Software Requirements

Operating System:

- Windows 10/11
- macOS
- Linux (Ubuntu recommended)

3.3 Python Version

Python **3.10 or 3.11** recommended.
(Your notebook is compatible with both.)

3.4 Required Libraries

Major libraries used:

Core:

- numpy
- pandas
- matplotlib
- seaborn
- warnings
- datetime
- os
- glob

Deep Learning:

- tensorflow
- keras
- keras.layers (LSTM, GRU, BiLSTM, Attention, Lambda, Dense)

Machine Learning Utilities:

- scikit-learn (MinMaxScaler, metrics)

Performance Measurement:

- time
- psutil (optional for memory usage)

3.5 Installation Commands

```
pip install numpy pandas matplotlib seaborn scikit-learn
pip install tensorflow keras
pip install psutil
```

Optional GPU:

```
pip install tensorflow-gpu
```

Ensure CUDA & cuDNN versions match TensorFlow GPU requirements.

4. Dataset Description

Primary Dataset: Bitbrains Workload Trace (GWA-T-12)

Used subset: **100 CSV files** selected from 1,750 VM performance traces.

Each file contains timestamped performance measurements, including:

- CPU usage (MHz) → **target column**
- CPU capacity provisioned
- CPU usage (%)
- Memory usage (KB)
- Disk throughput (read/write)
- Network throughput (tx/rx)

Important Notes:

- Timestamps recorded in milliseconds.
- Traces originate from enterprise application workloads.
- Only **CPU usage in MHz** is used for forecasting.

5. Data Preprocessing

The dataset is loaded from ~100 VM CSV files, cleaned, and converted into a unified time-series format with proper datetime fields. Missing values, inconsistencies, and abnormal CPU readings are handled, followed by feature extraction and scaling. A rolling window of 30 previous CPU values is created to form model-ready sequential inputs.

6. Exploratory Data Analysis

EDA includes visualizing CPU usage trends, distributions, temporal cycles, and rolling statistics to understand workload behavior. Autocorrelation, seasonal patterns, and VM-level variations help identify temporal dependencies. These insights guide model selection and window sizing.

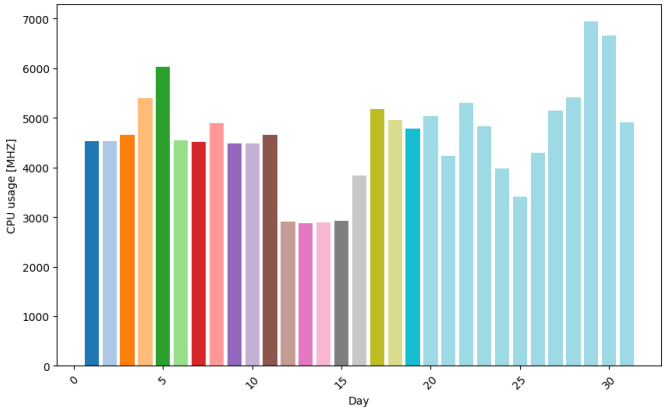

Figure 1: Bar Chart Visualization of Daily Average CPU Usage Patterns

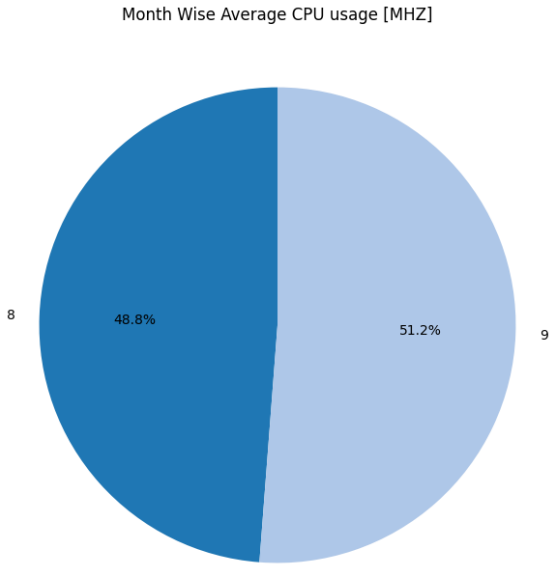

Figure 2: Pie Chart Representation of Monthly Average CPU Usage Distribution

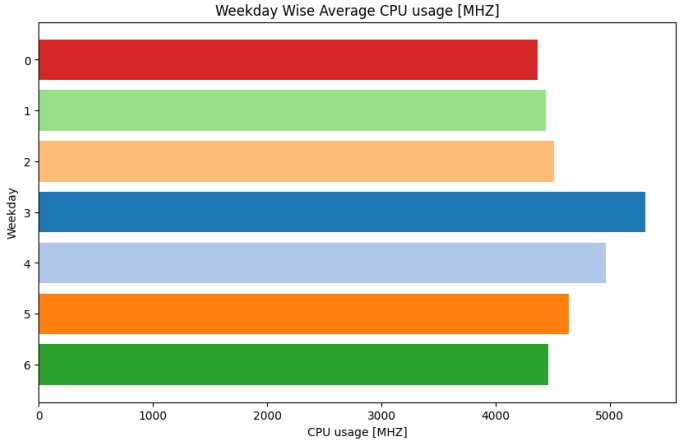

Figure 3: Horizontal Bar Chart of Weekday-Wise Average CPU Usage Analysis

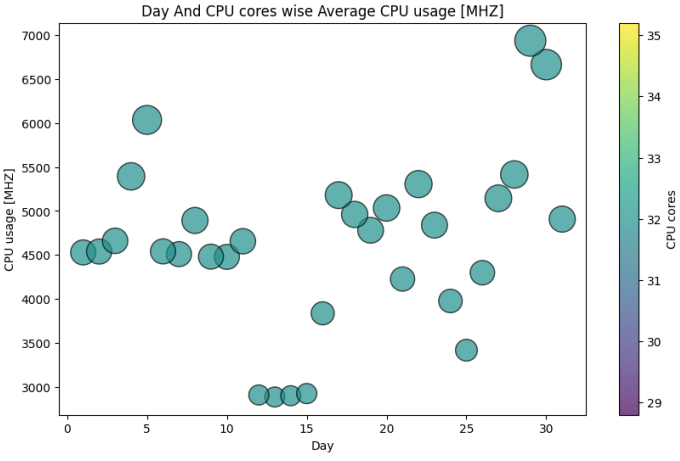

Figure 4: Scatter Plot of Day and CPU Core-Wise Average CPU Usage Distribution

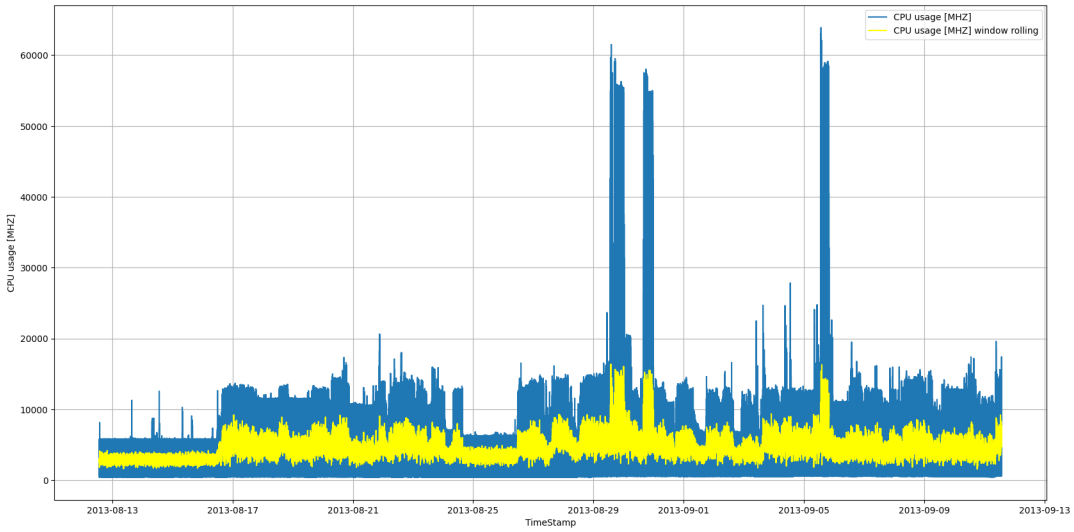

Figure 5: Temporal CPU Usage Trends

7. Model Architectures

The project implements LSTM, BiLSTM, and GRU as baseline sequence models, along with an enhanced SE-GRU architecture. SE-GRU integrates Squeeze-and-Excitation attention to emphasize important temporal features. This design improves representational strength and forecasting accuracy.

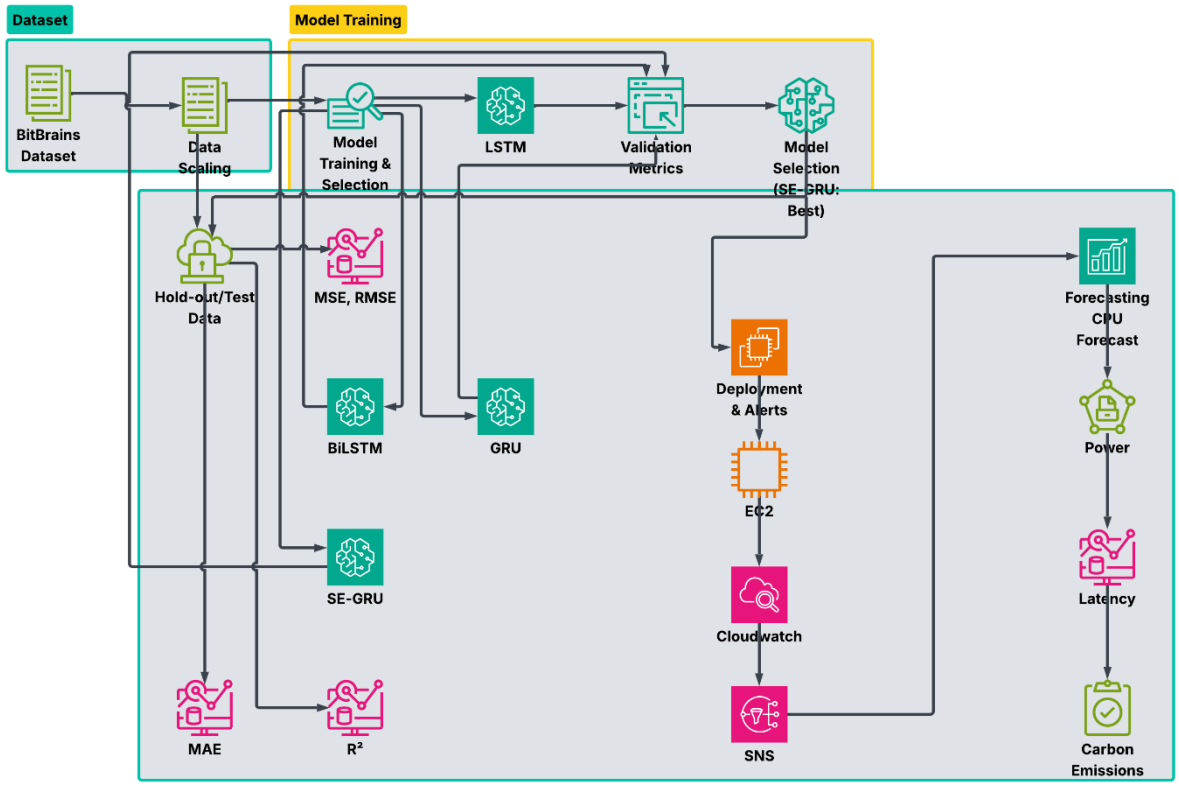

Figure 6: System Architecture Diagram

8. Training Pipeline

Data is split into 70% training and 30% testing, normalized, and transformed into windowed sequences before model training. Models are optimized using Adam with callbacks such as EarlyStopping and ReduceLROnPlateau. Performance metrics, latency, and throughput are tracked, and the best weights are saved.

9. Evaluation Metrics

Model performance is assessed using MAE, RMSE, R^2, and training–validation loss curves. Latency, throughput, and total execution time measure computational efficiency. Visual comparisons between actual and predicted CPU usage provide interpretability.

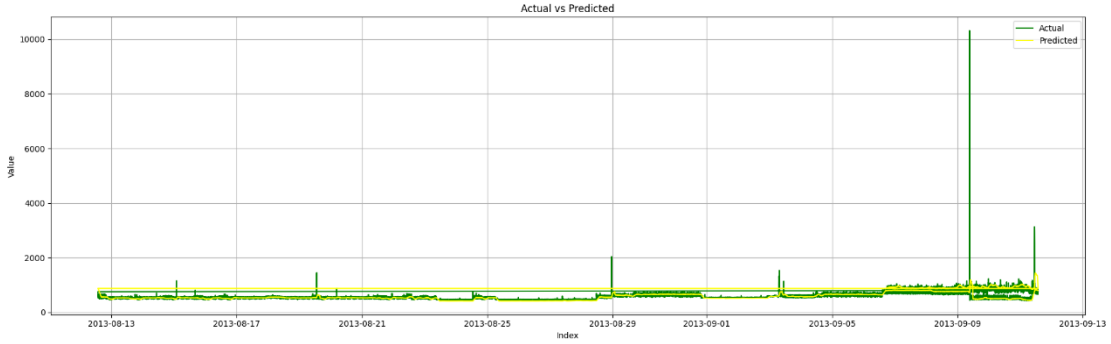

Figure 7: Actual vs Predicted Graph 1

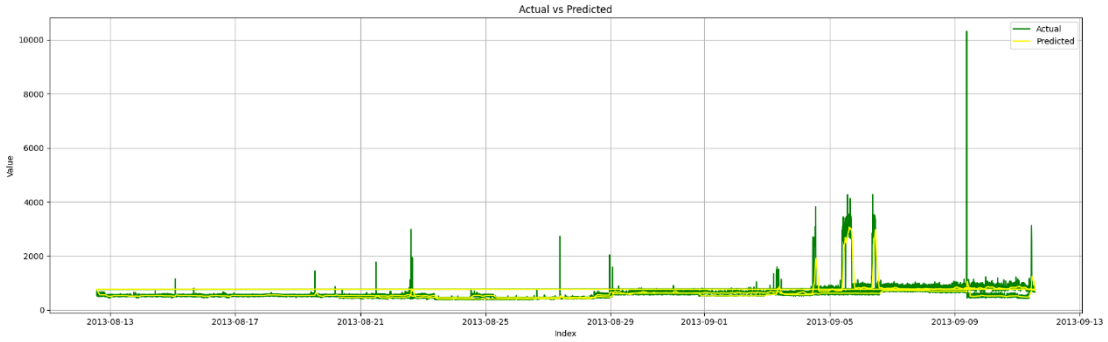

Figure 8: Actual vs Predicted Graph 2

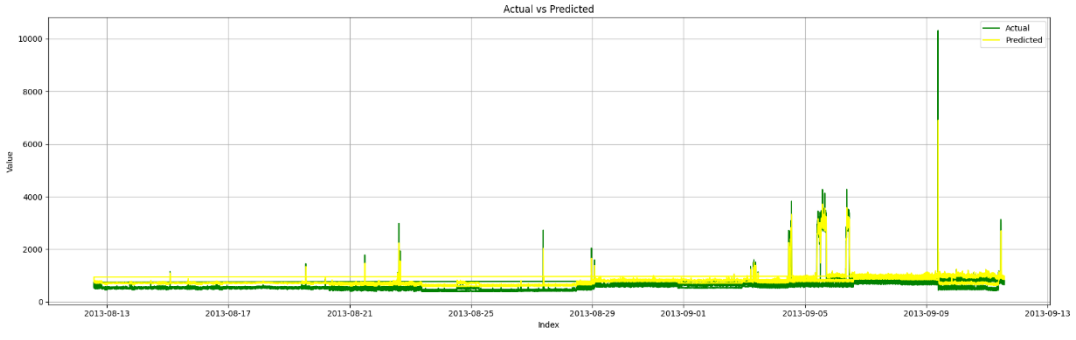

Figure 9: Actual vs Predicted Graph 3

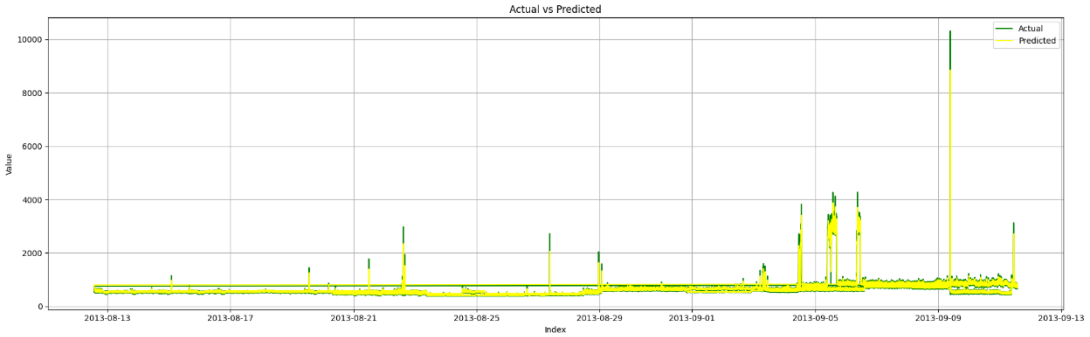

Figure 10: Actual vs Predicted Graph 4

10. Future Forecasting

The final model generates multi-step CPU usage forecasts, which are converted into CPU%, power (W), energy (kWh), and carbon emissions. Linear power models and standard carbon factors enable estimation of environmental impact. This supports energy-efficient cloud resource management.

References

TensorFlow Guide for RNN-based Models https://www.tensorflow.org/guide/keras/sequential_model

Google Cloud – Performance & Resource Management https://cloud.google.com/architecture

Carbon Footprint Calculator (General) https://www.carbonfootprint.com/calculator.aspx

The Green Grid – Power & Carbon Metrics (PUE, CUE) https://www.thegreengrid.org/

Original Squeeze-and-Excitation Networks (Paper & Code Links)https://arxiv.org/abs/1709.01507

Keras Official Documentation (LSTM, GRU, Attention)https://keras.io/api/layers/recurrent_layers/