National College of Ireland

Autonomous Cloud Resource Allocation: A Hybrid Machine Learning Ensemble Approach

MSc Research Project
MSc Cloud Computing

Abhishek Malik
Student ID: x22165843

School of Computing
National College of Ireland

Supervisor: Ahmed Makki

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Abhishek Malik
Student ID:	x22165843
Programme:	MSc Cloud Computing
Year:	2023
Module:	MSc Research Project
Supervisor:	Ahmed Makki
Submission Due Date:	14/12/2023
Project Title:	Autonomous Cloud Resource Allocation: A Hybrid Machine Learning Ensemble Approach
Word Count:	5352
Page Count:	19

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:	Abhishek Malik
Date:	14th December 2023

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):	

Autonomous Cloud Resource Allocation: A Hybrid Machine Learning Ensemble Approach

Abhishek Malik
x22165843

Abstract

This research looks at Autonomous Cloud Resource Allocation using a Multiscale Deep Learning Architecture, comparing two models for forecasting CPU consumption. Deep learning was used to train and assess Bidirectional Long Short-Term Memory (BILSTM) and STACKED LSTM GRU models on performance measures. The BILSTM model emerged as the clear winner, with remarkable predictive ability. In a comparison examination, the BILSTM model produced a validation loss of 8.6205e-04, an RMSE of 0.0294, and an outstanding R-squared value of 0.9743. Meanwhile, the STACKED LSTM GRU model performed well, with a validation loss of 9.1896e-04, an RMSE of 0.0303, and an R-squared of 0.9726. The results demonstrate the BILSTM model's superior accuracy in estimating CPU consumption, recognising detailed patterns, and explaining data volatility. These findings highlight the effectiveness of deep learning models for improving cloud resource allocation. The research adds to our understanding of how to implement powerful predictive models for autonomous resource management in cloud computing, opening the path for more efficient and responsive cloud architecture.

1 Introduction

1.1 Background

Cloud computing has changed the current computer environment by providing scalable and on-demand access to computational (Singh et al. (2017)). However, the dynamic nature of cloud systems poses issues in successfully managing these resources. Traditional resource allocation strategies frequently fall short of handling the complexities and variety of workloads inside cloud infrastructures. As a result, the notion of Autonomic Cloud Computing evolved, based on human autonomic systems, to construct self-managing and flexible cloud environments. The cloud computing business model provides easy on-demand network access to a shared pool of programmable computer resources. The background section provides essential context and foundational information relevant to the study of Autonomous Cloud Resource Allocation using Multiscale Deep Learning Architecture. Autonomic systems emulate self-regulatory mechanisms seen in nature, aiming to automate decision-making processes and optimize resource utilization without constant human intervention. This approach aligns with the ever-evolving and diverse nature of cloud workloads, offering the promise of efficient and responsive resource allocation.

1.2 Need for Autonomous Cloud Computing

The growing complexity of cloud systems, as well as the requirement to operate them efficiently, drive the demand for Autonomic Cloud Computing (Erdil (2013)). Autonomic systems mimic human-like self-regulation by adapting and adjusting to changing situations without continual manual intervention. Because of the scale, variety, and dynamic nature of cloud infrastructures, this technique is critical in cloud computing. Autonomous systems promise improved resource allocation, proactive fault management, and self-optimization, which are critical in successfully managing large and diversified workloads. The QoS determines the cloud services provided by the diverse and dynamic nature of cloud resources. Fulfilling QoS criteria while optimising efficiency and minimising resource dispersion, heterogeneity, and uncertainty pose difficulties to cloud computing systems that cannot be successfully met using typical resource allocation rules in the cloud environment.

1.3 Aim of the study

The purpose of this research and report is to investigate and assess the effectiveness of a Multiscale Deep Learning Architecture in the context of Autonomous Cloud Resource Allocation. The purpose of this research is to look at the use of advanced deep learning models, namely the Bidirectional Long Short-Term Memory (BILSTM) and STACKED LSTM GRU models, in forecasting CPU consumption in cloud computing settings. The key goal is to evaluate these models' capacity to estimate resource needs efficiently, optimising the allocation of computing resources autonomously. This paper attempts to determine the most accurate and trustworthy forecast model for CPU utilisation by rigorous examination and comparison of different models, eventually giving significant insights into boosting autonomous resource management in cloud infrastructures.

1.4 Research Objectives

The research objectives of this report are:

- To evaluate the predictive power of the Bidirectional Long Short-Term Memory (BILSTM) and STACKED LSTM GRU models in forecasting CPU use.
- To compare and assess the performance of various deep learning models in the context of Autonomous Cloud Resource Allocation.
- Develop the most accurate and effective model for resource allocation optimization in cloud computing settings.

1.5 Research Questions

The research questions for this report are:

- How well do Bidirectional Long Short-Term Memory (BILSTM) and STACKED LSTM GRU models predict CPU usage in cloud environments?
- How do these deep learning models compare in terms of Autonomous Cloud Resource Allocation?

- Which model among BILSTM and STACKED LSTM GRU demonstrates superior accuracy for optimizing resource allocation in the cloud?
- How can Multiscale DL Architectures contribute to autonomous resource management in cloud infrastructures?

1.6 Research Gaps

The research gaps in this area of research lie in:

- Limited studies compare Bidirectional Long Short-Term Memory (BILSTM) and STACKED LSTM GRU models specifically for Autonomous Cloud Resource Allocation.
- Lack of exploration regarding the real-time adaptability of these models to dynamic changes in cloud environments.

2 Related Work

Table 1: Comparison Table for Traditional Resource Allocation Approaches.

Study	Focus	Approach	Main Challenge	Key Result
Grewal and Pateriya (2013)	Hybrid cloud scalability	Rule-Based	Scalability enhancement	Improved resource utilization and cost reduction
Tan et al. (2020)	Container-based clouds	Heuristic	Optimizing resource utilization	Enhanced resource allocation in container clouds
Haratian et al. (2019)	Adaptive resource management	Adaptive & Fuzzy	Dynamic workload challenges	Dynamic rule and membership updates for QoS
Chen et al. (2020)	Self-adaptive resource allocation	Iterative QoS prediction & PSO-based	QoS prediction accuracy & resource allocation	Improved QoS prediction accuracy & resource efficiency
Murad et al. (2022)	Job scheduling in cloud	Various Job Scheduling Techniques	Effective scheduling strategies	Emphasized the importance of effective scheduling

2.1 Traditional Resource Allocation Approaches

Cloud computing is a complicated but critical feature that seeks to improve resource allocation in constantly changing contexts. (Grewal and Pateriya (2013)) developed a Rule-Based Resource Manager for hybrid cloud environments, emphasising scalability enhancement and cost reduction using rule-based systems. (Tan et al. (2020)) proposed a Cooperative Coevolution Genetic Programming (CCGP) strategy to solve resource allocation issues in container-based clouds, concentrating on learning workload patterns and VM types for enhanced resource usage. In another study (Haratian et al. (2019)) proposed an Adaptive and Fuzzy Resource Management framework (AFRM) that dynamically updated rules and membership functions to meet QoS requirements, whereas (Chen et al. (2020)) contributed a self-adaptive resource allocation method that uses iterative QoS prediction models and PSO-based algorithms to improve prediction accuracy

and optimise cloud application resource allocation. (Murad et al. (2022)) also highlighted job scheduling methodologies, emphasising the need for good scheduling tactics in cloud systems.

Table 2: Comparison Table for Machine Learning Techniques in Resource Allocation.

Study	**Main Focus**	**Key Techniques**	**Key Contributions**
Wang et al. (2018)	Dynamic resource allocation	Historical data analysis, similarity identification	Introduction of data-driven approaches, historical data analysis for allocation decisions
Jixian Zhang (2018)	Resource allocation prediction	Linear and logistic regression	Prediction models for optimal resource allocation
Bal et al. (2022)	Task scheduling & resource allocation	Cat swarm optimization, group optimization-based DNN	Enhanced schedulers, DNN for resource allocation
Goodarzy et al. (2020)	Machine learning survey	Workload estimation, task scheduling, VM consolidation, energy optimization	Comprehensive review of ML projects in resource management
Khan et al. (2022)	Challenges & future directions	Various ML techniques in resource management analysis	Identification of challenges, proposed future research directions
Huang et al. (2013)	Resource allocation in cloud computing	Support Vector Regressions (SVRs)	Effective resource redistribution based on response time estimation
Jayaprakash et al. (2021)	Energy efficiency in cloud computing	Clustering, Optimization, ML methods	Demonstrated the efficacy of ML in enhancing energy efficiency
Demirci (2015)	Energy conservation in cloud computing	ML-based solutions for resource management	Surveyed ML applications, highlighting energy efficiency in the cloud
Duc et al. (2019)	Reliable resource provisioning in joint edge-cloud environments	ML for workload prediction, component placement	Explored ML's role in enhancing reliability in joint edge-cloud setups
Kumar et al. (2022)	ML and cloud computing integration for various optimizations	ML for load allocation, task scheduling, energy optimization	Demonstrated ML's potential in enhancing multiple aspects of cloud computing

2.2 Machine Learning Techniques in Resource Allocation

Initially, (Wang et al. (2018)) and (Jixian Zhang (2018)) focus on the transition from static policies to data-driven methods, emphasising machine learning's potential to tackle resource allocation difficulties. They offer the idea of guiding resource allocation decisions with historical data and similarity analysis. (Jixian Zhang (2018)) focus on linear and logistic regression models in predicting optimal resource allocation. Building on this

foundation, (Bal et al. (2022)) offer an improved cat swarm optimization-based scheduler and a group optimization-based deep neural network for job scheduling and resource allocation, respectively, stressing the application of optimization methods and deep learning in resource management. Moreover, (Goodarzy et al. (2020)) broaden this subject by undertaking a thorough evaluation of machine learning initiatives in cloud resource management, with an emphasis on workload estimate, task scheduling, VM consolidation, and energy efficiency. Finally, (Khan et al. (2022)) extensively examine the existing ML-based resource management issues and limits and recommend future research options to develop this topic.

Various research has focused on resource allocation in cloud and edge computing systems, with machine learning (ML) approaches emerging as critical tools for maximising efficiency and dependability. These studies investigate the complexities of resource provisioning, including components such as workload classification, application placement, and overall system orchestration. (Huang et al. (2013)) pioneered the use of Support Vector Regressions (SVRs) to predict response times, allowing for resource redistribution across virtual machines (VMs) on physical servers. (Jayaprakash et al. (2021)) expanded on this by examining clustering, optimization, and machine learning approaches, highlighting their significance in energy efficiency and performance improvement. (Demirci (2015)) examined ML's use in energy efficiency inside cloud computing in-depth, assessing numerous ML-based resource management systems. Duc et al. (2019) and Kumar et al. (2022) focus on combined edge-cloud systems, emphasising the need of machine learning to guarantee reliable resource delivery. To overcome the challenges of distributed applications in heterogeneous networks, their research focused on workload prediction, component placement, and application flexibility.

2.3 Autonomous Systems in Cloud Computing

Both (Alqefari and Zaghloul (2013)) and (Werkhoven et al. (2018)) anticipate a future in which intelligent systems will play a critical role in constructing technological landscapes. (Alqefari and Zaghloul (2013)) offer a network system architecture for service management that leverages cloud computing, with an emphasis on the integration of autonomic Service-Oriented Architecture (SOA) and distributed autonomous control frameworks. In contrast, (Werkhoven et al. (2018)) emphasize the appropriate integration of intelligent autonomous machines into human society, stressing self-awareness in machines, human-defined aims, and comprehension of human behaviours for effective collaboration. Both studies anticipate a future with autonomous systems powered by sophisticated architectures and artificial intelligence. Within Industry 4.0, (Kovács et al. (2019)) expanded on these ideas by adding autonomous systems known as multi-systems, emphasising decentralised control and cooperation via intelligent robots powered by artificial intelligence. (Nasr (2022)) investigated the use of cloud computing in robot control systems, exploiting its heavy computational capability to overcome embedded component restrictions. Their suggested system included clustering, allocation, and path planning phases to allow for effective job execution by several robots. (Singhal et al. (2022)) broadened the horizon by providing an advanced heuristic scheduling approach for cloud infrastructure based on enhanced ant colony optimization.

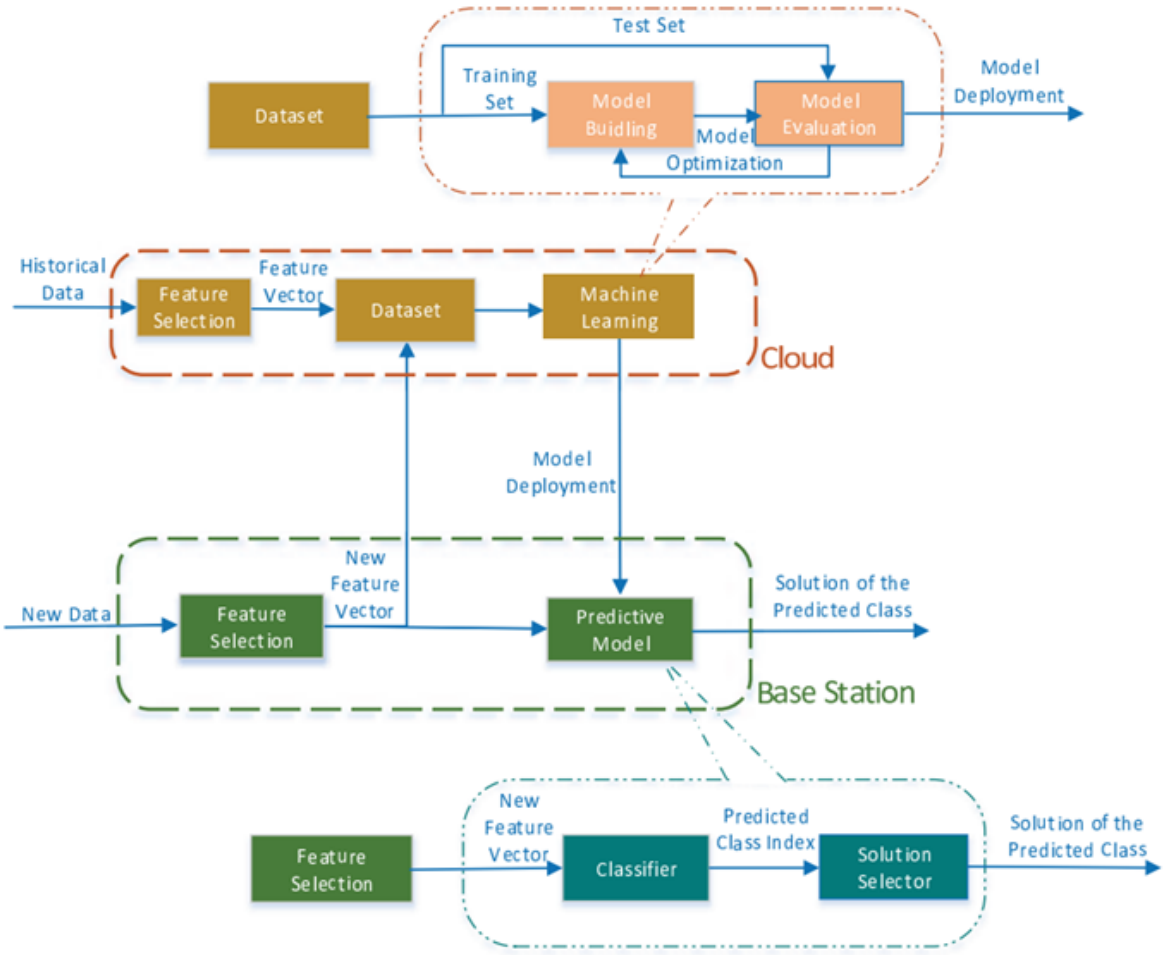

Figure 1: Machine Learning Framework of Resource Allocation by (Wang et al. (2018))

Table 3: Autonomous Systems in Cloud Computing.

Study	Focus	Key Techniques/ Approach	Main Challenges Addressed	Key Results/Findings
Alqefari and Zaghloul (2013)	Network system architecture	Autonomic SOA, distributed control	Complexity, dynamism, autonomous restructuring	Enhanced service management via cloud-based architecture
Werkhoven et al. (2018)	Integration of autonomous machines	Self-awareness, human-defined objectives	Handling failures, human-machine collaboration	Pathway to responsible integration of AI machines
Kovács et al. (2019)	Industry 4.0 and robotics	Multi-agent systems, decentralized control	Intelligent robots, Industry 4.0 principles	Decentralized control in smart factories with AI robots
Nasr (2022)	Cloud-based robot control systems	Leveraging cloud computing for control	Embedded system limitations, efficient control	Improved task allocation, efficiency using cloud power
Singhal et al. (2022)	Heuristic scheduling for cloud infra.	Enhanced ant colony optimization	Nonlinear loads, processing complexities	Optimized resource allocation, rapid solution times

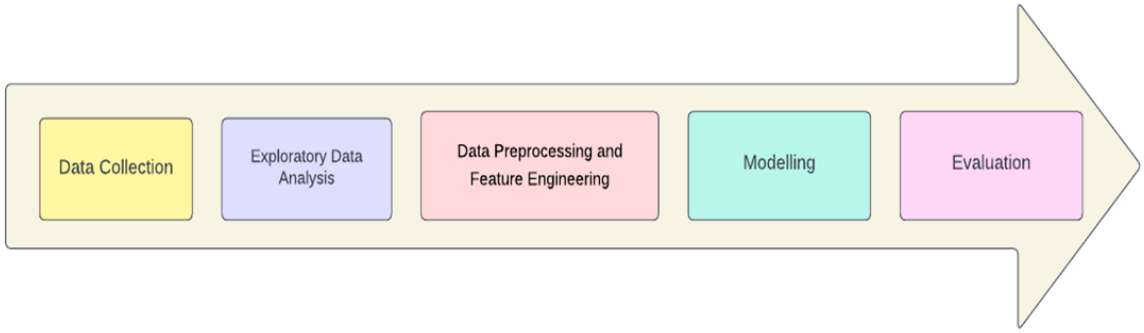

Figure 2: Flow Diagram

3 Methodology

3.1 Data Collection

Performance measurements from 1,750 virtual machines (VMs) in a dispersed data centre hosted by Bitbrains, a managed hosting and business computing service provider, were collected throughout the project's data-gathering phase. Bitbrains serves big financial institutions and insurers by hosting apps used mostly in the solvency sector for financial reporting after fiscal quarters. The dataset, divided into two tracks (fastStorage and Rnd), comprises files representing virtual machines (VMs) linked to either fast storage area network (SAN) or slower Network Attached Storage (NAS) devices. Because of the greater storage performance, the fastStorage trace has 1,250 VMs, with a higher proportion of application servers and compute nodes. In contrast, the Rnd trace, which has 500 VMs, has a greater proportion of management machines without lower storage performance needs. The files in the Rnd trace are organised into monthly subdirectories, and each file, which is formatted in rows, contains observations of performance metrics including such CPU cores, CPU capacity, CPU usage, memory provisioned and using, disc read and write throughput, and network received and transmitted throughput. The timestamp represents the number of milliseconds since January 1, 1970. To handle resource limits, only 100 CSV files were used out of the whole dataset, with an emphasis on estimating CPU utilisation in megahertz using date-time information.

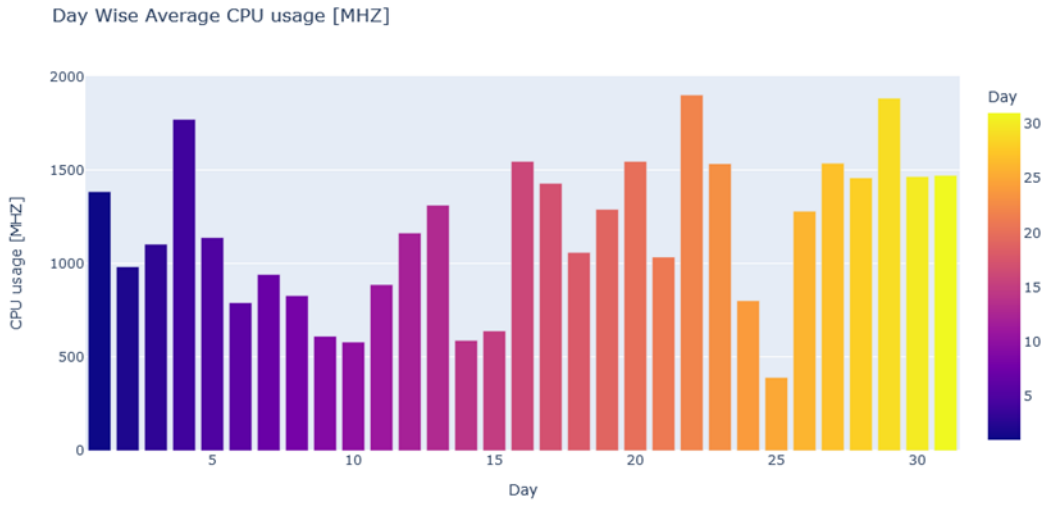

Figure 3: Day-wise Average CPU Usage Over Time (MHZ)

3.2 Exploratory Data Analysis

EDA is a detailed assessment of the performance metrics gathered for 1,750 virtual machines (VMs) in Bitbrains' distributed data centre. The goal of the analysis is to acquire insight into the data's structure, trends, and qualities. An in-depth review of the distribution of VMs between the fastStorage and Rnd traces, realising the composition of VM types based on storage connectivity (fast SAN or slower NAS), or rather assessing any discernible patterns concerning the nature of applications hosted by Bitbrains for major banks and insurers would be part of the EDA process. Furthermore, EDA would entail investigating the variation in performance parameters such as CPU utilisation, memory usage, disc throughput, and network throughput across the two traces.

3.3 Data Preprocessing and Feature Engineering

The goal of the data pretreatment and feature engineering phase, which corresponds to the transformation phase of the KDD technique, is to refine and enhance the performance metrics dataset from Bitbrains' distributed data centre. Converting timestamps to a uniform datetime format allows for consistent temporal analysis. Furthermore, feature engineering is used to extract relevant temporal information from timestamps, such as month and year. These derived characteristics improve the dataset's interpretability and aid in visualisation efforts, allowing for a more in-depth knowledge of performance patterns over time.

3.4 Modelling

The focus of the modelling step in the KDD methodology and in the area of Bitbrains' distributed datacenter performance measurements is on building and training predictive models to forecast CPU utilisation. This stage involves installing advanced deep learning models including such STACKED LSTM GRU and BILSTM to capture subtle patterns

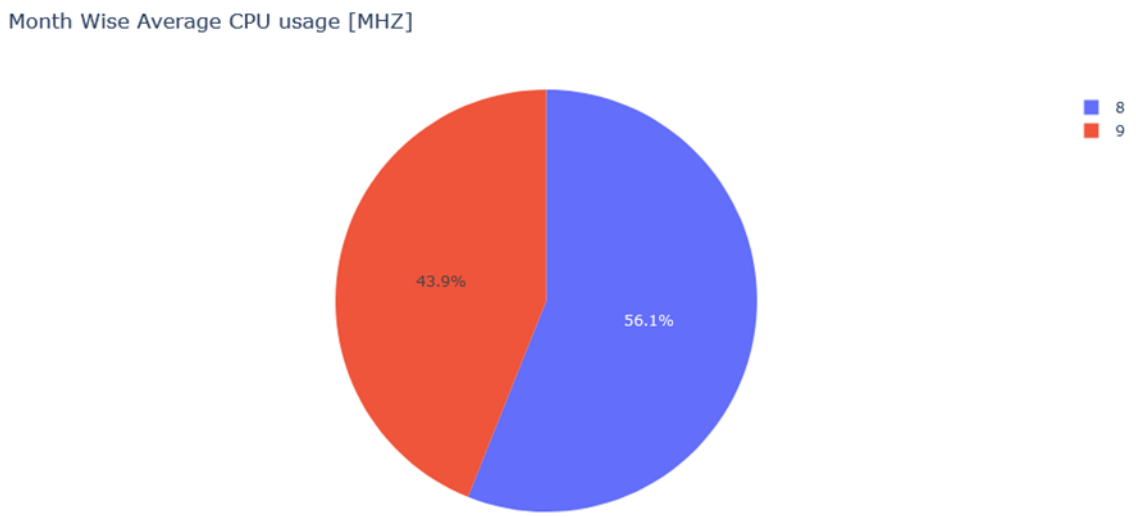

Figure 4: Monthly Distribution of Average CPU Usage

and relationships in the data, leveraging features acquired during the transformation phase. These algorithms are evaluated on the heavily processed dataset to discover correlations between performance parameters and timestamped CPU consumption, allowing future CPU usage patterns to be predicted. With a 70:30 split ratio, the dataset is partitioned into two subgroups. The training set receives 70% of the data, allowing the model to learn from this significant amount. The remaining 30% is the testing set, which acts as a separate dataset to assess the model's performance and generalizability.

3.5 Evaluation

The major goal of the assessment phase of the Knowledge Discovery in Databases (KDD) approach, implemented to Bitbrains' distributed data centre performance metrics, is to evaluate the performance and efficacy of the built forecasting models. To assess the prediction capabilities of the models, many metrics such as accuracy and loss graphs, RMSE values, and indeed the comparison of real and anticipated values are used. The study is to test how effectively the STACKED LSTM GRU and BILSTM models can anticipate CPU use based on the extracted features by measuring the accuracy and loss measures.

3.6 Data Visualization

A graphical depiction of the day-to-day average CPU consumption is shown in 3. The graph's x-axis shows individual days, demonstrating the chronological element of the data, whilst the y-axis depicts average CPU utilisation in megahertz (MHz). Each point on the graph reflects the average CPU utilisation on a particular day. The graphical representation is useful for visually examining daily fluctuations and trends in CPU utilisation over a certain time period.

A pie chart is shown in 4 to graphically display the distribution of monthly average

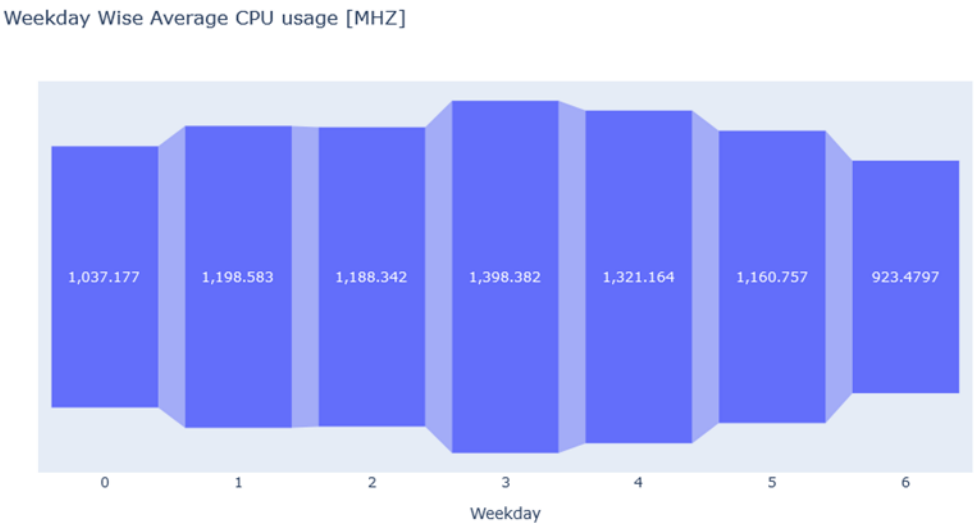

Figure 5: Weekday-wise Average CPU Usage Distribution

CPU consumption. The figure is broken into two sections: one in red which represents 43.9 percent and one in blue that represents 56.1 percent. Each section represents a percentage of the average CPU utilisation for a given month. The red segment, which accounts for 43.9 percent, depicts the relative percentage of CPU usage for one month, while the blue section, which accounts for 56.1 percent, represents CPU consumption for another month.

A graphical depiction of the weekday average CPU utilisation is shown in 5. The graph's x-axis ranges from 0 to 6, signifying each day of the week, with 0 generally referring to Sunday and 6 typically corresponding to Saturday. The y-axis represents the average CPU utilisation.

6 depicts a graphical representation of the link between the day, the number of CPU cores, and the average CPU consumption measured in megahertz (MHz). The graph's x-axis depicts individual days, indicating the data's temporal component, while the y-axis indicates average CPU utilisation. Each data point on the graph represents a unique day and the average CPU utilisation measured for a certain number of CPU cores, which ranges from 0 to 30.

3.7 List of Models

This section involves the implementation of advanced neural network architectures for forecasting CPU usage in the context of Bitbrains' distributed datacenter performance metrics. Two specific models are employed:

3.7.1 STACKED LSTM GRU MODEL:

This model employs a layered architecture that combines layers of Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). LSTMs and GRUs are recurrent neural network types that are designed to capture long-term relationships in sequential

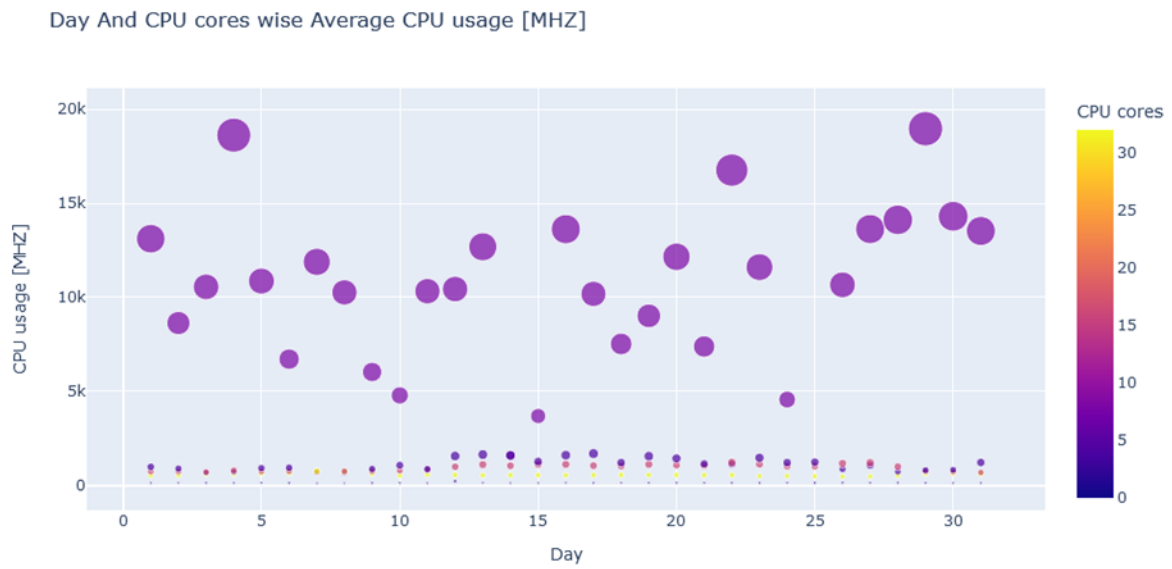

Figure 6: Day and CPU Cores-wise Average CPU Usage (MHz)

data, making them suited for time series forecasting problems. The stacking of these layers provides for a more comprehensive representation of the dataset's temporal patterns, improving the model's capacity to identify delicate correlations between performance measures and CPU utilisation.

3.7.2 BILSTM MODEL:

Another deep learning architecture used for forecasting is the Bidirectional LSTM (BiLSTM) model. Bidirectional LSTMs analyse input data both forward and backward, allowing the model to collect information from both past and future contexts. Because of this bidirectional approach, the BiLSTM model is well-suited for projecting CPU utilisation based on previous performance measurements.

4 Design Specification

The fundamental approaches, systems, and frameworks enabling the implementation of the project are fully defined and detailed in the "Design Specification" chapter. The selection and description of sophisticated deep learning models optimised for estimating CPU utilisation in Bitbrains' distributed data centre performance measures is the primary topic. The implementation, in particular, makes use of two significant models: the STACKED LSTM GRU MODEL and the BILSTM MODEL. The STACKED LSTM GRU MODEL is an advanced architecture that combines layers of Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). Recurrent neural networks such as LSTMs and GRUs are notable for their capacity to detect long-term relationships in sequential data. By layering these layers, the model acquires a better knowledge of the dataset's temporal patterns, allowing it to discover complicated correlations between various performance measures and CPU utilisation.

In parallel, the BILSTM MODEL, which employs a Bidirectional Long Short-Term Memory (BiLSTM) architecture, is an important component of the design. The BiLSTM model bidirectionally analyses input data, incorporating information from both past and future contexts. This bidirectional technique is very useful for understanding temporal connections in time series data, and it aligns perfectly with the forecasting goals of estimating CPU consumption based on previous performance measures. The implementation requirements are built around the Amazon Sagemaker laptop environment, guaranteeing model training and assessment compatibility and efficiency. The use of deep learning frameworks like TensorFlow or PyTorch is described, highlighting their importance in assisting with the creation of sophisticated neural network topologies.

This chapter also emphasises the need for a curated dataset, especially picking 100 CSV files from a bigger dataset to properly manage resource restrictions. The data pretreatment processes are specified to ensure that the dataset is optimal for training and assessment, including managing missing values, feature engineering, and datetime conversion. As a result, the "Design Specification" chapter serves as a thorough guide, offering an in-depth explanation of the models, frameworks, and dataset processing procedures that have been chosen. This systematic method promotes project openness and clarity, setting the groundwork for effective execution and subsequent review.

5 Implementation

The last stage of bringing the suggested solution to actuality is discussed in the "Implementation" chapter, concentrating on the outputs generated as well as the tools and languages used. The building of trained deep learning models for estimating CPU utilisation in Bitbrains' distributed datacenter is a crucial outcome of the implementation phase. The curated dataset, which contains day-by-day average CPU consumption, month-by-month distribution, and weekday-by-weekday trends, was used to effectively train the STACKED LSTM GRU MODEL and the BILSTM MODEL.

The outputs include altered data that went through a careful preparation pipeline, including missing value management, feature engineering, including datetime conversion. The dataset has now been adjusted for use as input into deep learning models, ensuring ensuring temporal patterns and relationships are well recorded. The processed data is useful in training models that can make accurate predictions about CPU consumption patterns.

This implementation phase's code complies to recommended practises in deep learning model building. Popular frameworks such as TensorFlow or PyTorch have been used to write the architecture of the STACKED LSTM GRU MODEL and the BILSTM MODEL, with Python as the major programming language. The Amazon Sagemaker notebook environment is heavily used during the implementation phase, offering a smooth and efficient platform enabling model training and validation.

The models created are the result of iterative design, training, and assessment, and they align with the suggested solution detailed in previous chapters. Specific code listings and extensive user manual explanations, on the other hand, are purposefully left out of this part to retain a simple and high-level perspective. Instead, the emphasis is on completing the implementation successfully, emphasising the strategic use of tools, languages, and processed data to fulfil the project's objectives. This chapter acts as a link between both the conceptual framework and the actual consequences, illustrating

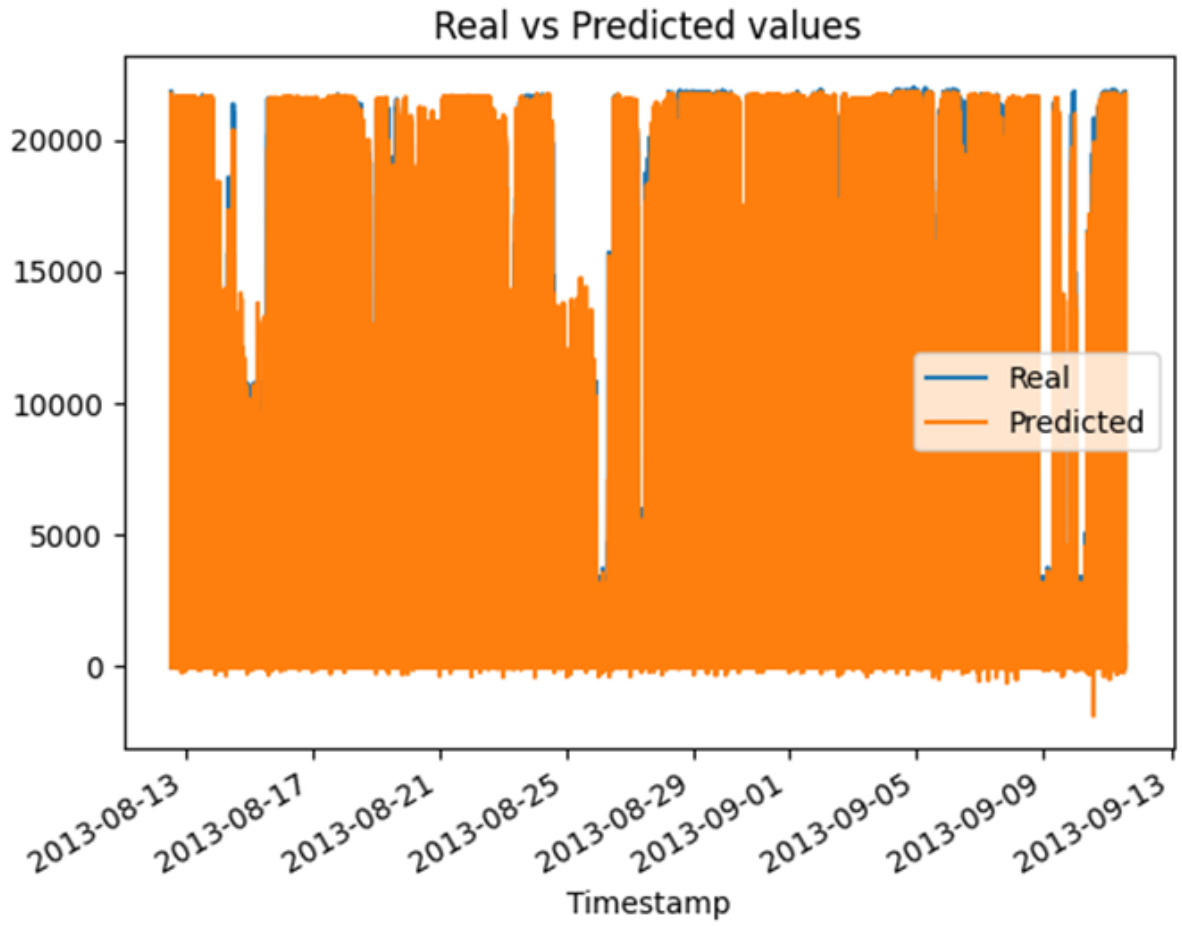

Figure 7: Actual & Predicted Graph - STACKED LSTM GRU MODEL

how the suggested approach may be used to anticipate CPU utilisation in a real-world datacenter scenario.

6 Evaluation

6.1 STACKED LSTM GRU MODEL

The STACKED LSTM GRU MODEL is an important element of a multiscale deep learning architecture meant to improve computational resource allocation in a cloud setting. Stacked Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) layers are used in the model to capture complicated temporal correlations within performance metrics data. The LSTM layers aid in the comprehension of long-term patterns, but the GRU layers improve the model's ability to catch short-term fluctuations. This stacking design enables multiscale analysis, allowing for both broad trends and fine-grained changes in resource utilisation. The STACKED LSTM GRU MODEL, trained on historical data, learns to recognise patterns that indicate optimal resource allocation, allowing the autonomous system to adapt dynamically to altering computing demands.

The "validation loss" measure, with a value of 9.1896e-04, represents the validation dataset's mean squared error. This low number indicates that the model's predictions

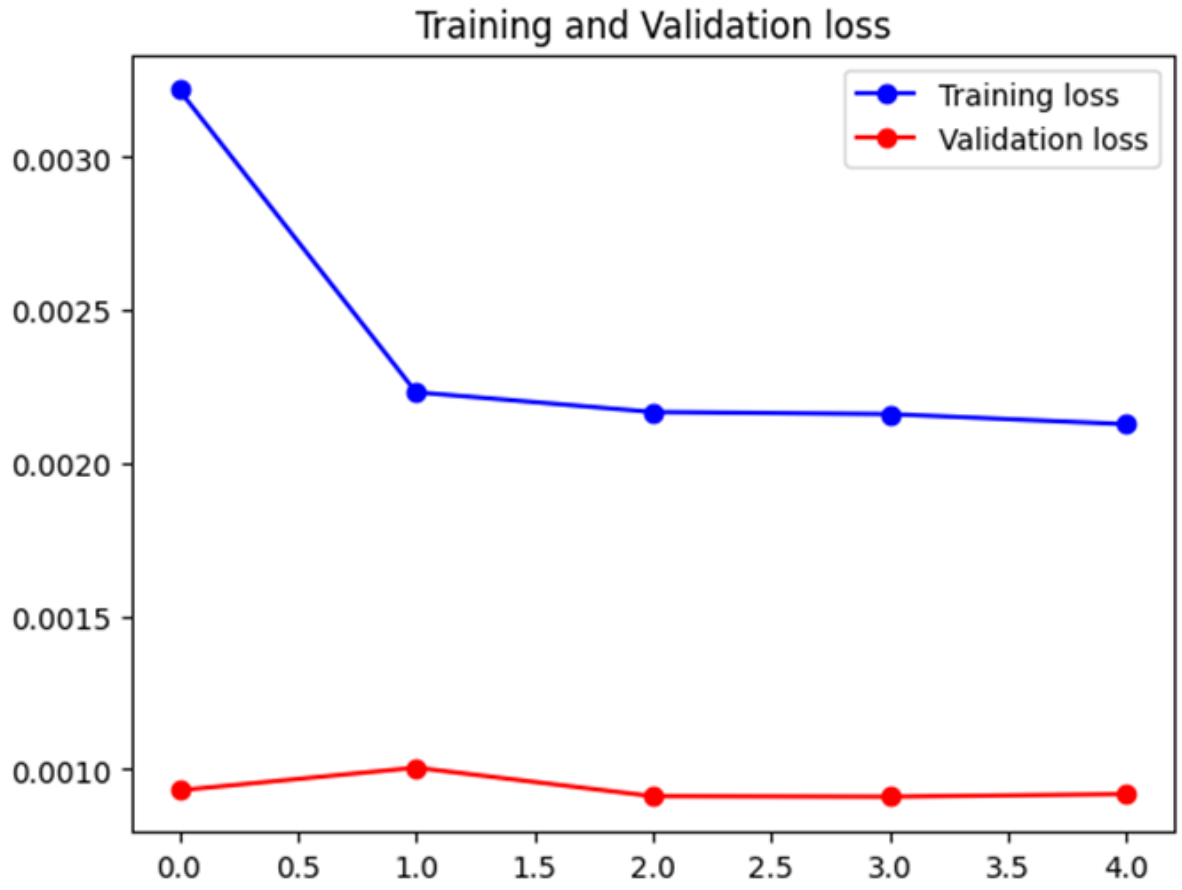

Figure 8: Loss Graph

are extremely near to the actual values in the validation set, indicating the model's efficacy in reducing prediction errors. At the same time, the Root Mean Squared Error (RMSE) is reported to be 0.030314197955657814. The root mean square error (RMSE) is a measure of the average size of errors between anticipated and actual values, with a smaller RMSE suggesting more precision. In this situation, the low RMSE indicates that the STACKED LSTM GRU MODEL predicts CPU use well, since the average error in predictions is rather modest. Furthermore, the R-squared value, reported as 0.9726457693339003, indicates how well the model explains the variation in the data.

6.2 BILSTM MODEL

Bidirectional Long Short-Term Memory (BiLSTM) MODEL appears as a critical component in a multiscale deep learning architecture. The BiLSTM MODEL presents a bidirectional method to sequence processing, which is tailored to improve the allocation of computer resources in a cloud setting. The model develops a thorough grasp of temporal relationships within performance metrics data by processing information in both forward and backward directions. This bidirectional feature is especially useful for recording intricate patterns and correlations, providing a more comprehensive picture of resource utilisation changes. The BiLSTM MODEL excels at understanding temporal connections when trained on historical data, allowing it to estimate CPU consumption

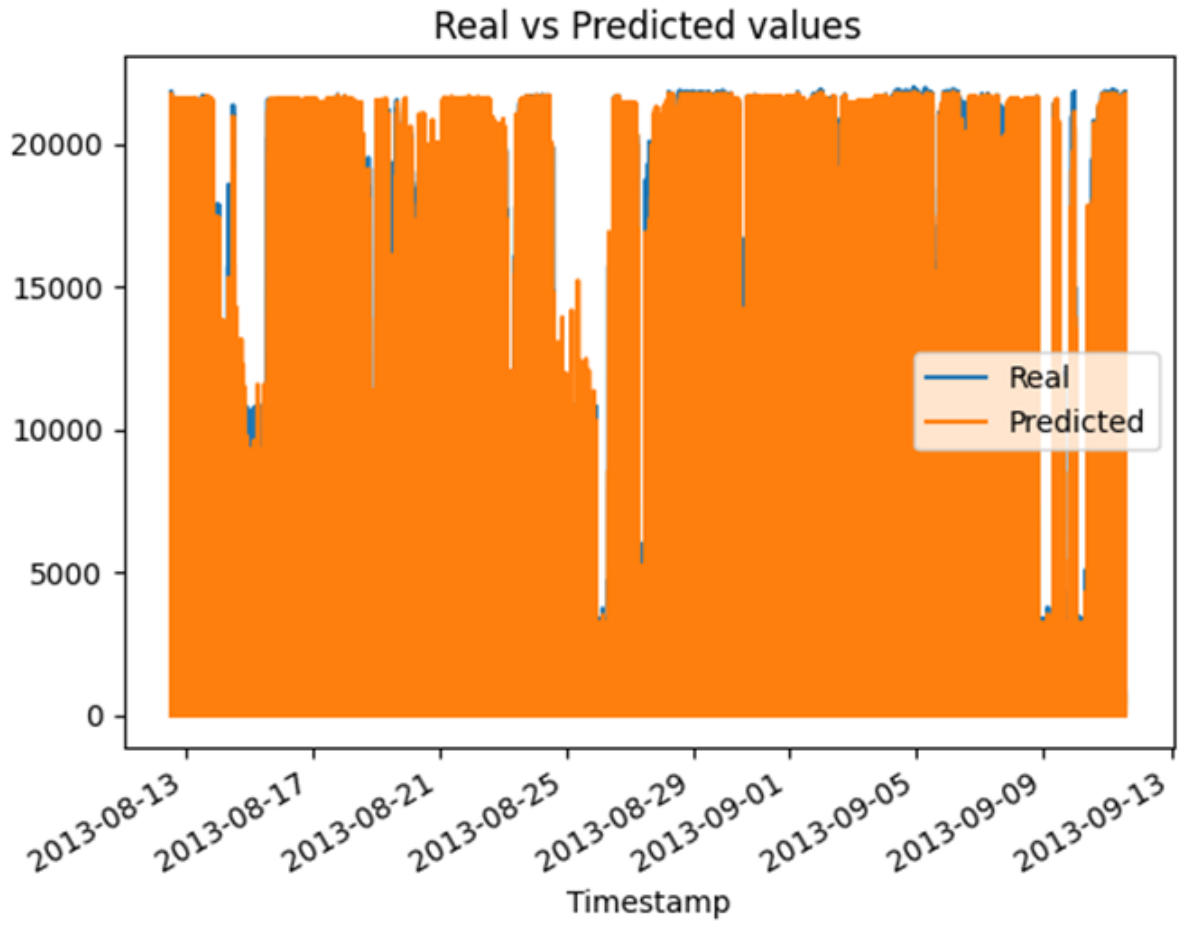

Figure 9: Actual & Predicted Graph - BILSTM MODEL

based on a bidirectional study of past and future contexts.

The "validation loss" measure, with a value of 8.6205e-04, represents the validation dataset's mean squared error. This low result implies that the BiLSTM model's predictions closely match the actual values in the validation set, demonstrating the model's ability to minimise prediction errors. The Root Mean Squared Error (RMSE), which was reported as 0.02936068540477952, is a measure of the average size of differences between expected and actual values. Furthermore, the R-squared value of 0.9743395233305695 indicates how well the model explains the variation in the data. The Best Model is BILSTM with best RMSE and R-squared Value.

6.3 Discussion

The table 4 shows that the BILSTM model is more efficient than the STACKED LSTM GRU Model as the RMSE value approaches 0 and the R-Squared approaches 1.

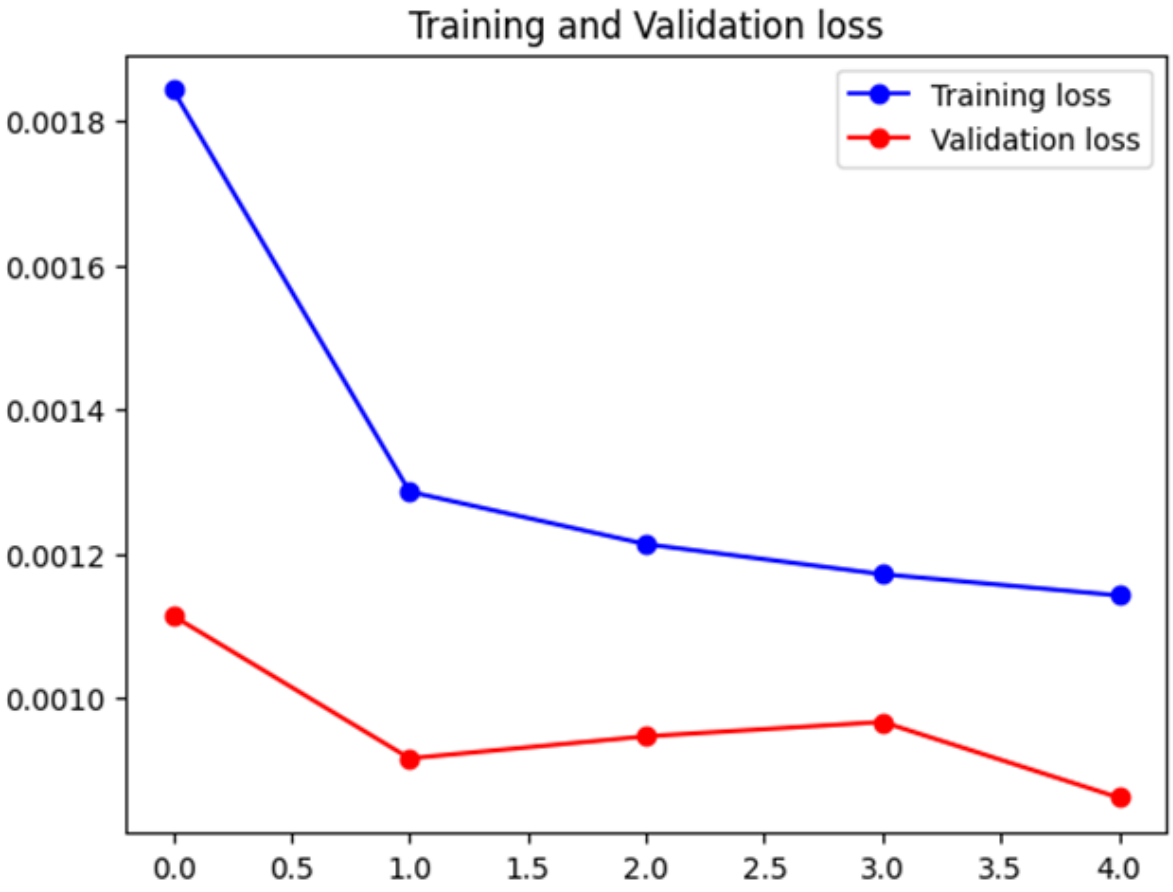

Figure 10: Loss Graph - BILSTM MODEL

Table 4: Comparison of Performance Metrics for BILSTM and STACKED LSTM GRU Models in Autonomous Cloud Resource Allocation.

Model	**Validation Loss**	**RMSE**	**R-squared**
BILSTM	8.6205e-04	0.0294	0.9743
STACKED LSTM GRU	9.1896e-04	0.0303	0.9726

7 Conclusion and Future Work

7.1 Conclusions

In conclusion, this work investigated and implemented advanced deep learning models for predicting CPU use in the context of Autonomous Cloud Resource Allocation, especially the Bidirectional Long Short-Term Memory (BILSTM) and STACKED LSTM GRU models. The models performed well, with the BILSTM model emerging as the preferable option as the best model, with reduced Root Mean Squared Error (RMSE) and better R-squared values. The findings show that deep learning algorithms are effective in predicting and optimising CPU resource allocation in a dynamic cloud computing environment. The research provides important insights into the use of multiscale deep learning systems for improving autonomous resource management

7.2 Limitations

Despite the optimistic results, this study has certain drawbacks. The success of the models is strongly dependent on the quality and representativeness of the training data. Inaccuracies or biases in the dataset may have an influence on the models' ability to generalise to new data. Furthermore, the study assumes stationarity in the underlying patterns, which might pose problems in real-world circumstances with non-stationary patterns. Furthermore, the project's scope was limited to CPU consumption; extending the models to include additional resource measures might give a more thorough knowledge of resource allocation dynamics.

7.3 Future Works

Future study in this area might take numerous forms. To begin, expanding the dataset by include a larger variety of performance variables and considering longer time periods might improve the forecasting capabilities of the models. Investigating ensemble approaches or hybrid models that integrate deep learning with standard forecasting techniques might yield synergistic advantages. Adapting the models to real-time learning and including external elements like network conditions or application workloads may further increase their flexibility in dynamic cloud settings. Furthermore, evaluating the models' performance under various cloud architectures and deployment scenarios would add to their resilience and usefulness in a variety of circumstances. Finally, including energy efficiency as a fundamental optimization factor in resource allocation might match the models with cloud computing sustainability aims.

References

Alqefari, S. and Zaghloul, S. (2013). A new architecture of an autonomous system in cloud computing.

Bal, P. K., Mohapatra, S. K., Das, T. K., Srinivasan, K. and Hu, Y.-C. (2022). A joint resource allocation, security with efficient task scheduling in cloud computing using hybrid machine learning techniques, *Sensors* **22**(3).
URL: *https://www.mdpi.com/1424-8220/22/3/1242*

Chen, X., Wang, H., Ma, Y., Zheng, X. and Guo, L. (2020). Self-adaptive resource allocation for cloud-based software services based on iterative qos prediction model, *Future Gener. Comput. Syst.* **105**(C): 287–296.
URL: *https://doi.org/10.1016/j.future.2019.12.005*

Demirci, M. (2015). A survey of machine learning applications for energy-efficient resource management in cloud computing environments, *2015 IEEE 14th international conference on machine learning and applications (ICMLA)*, IEEE, pp. 1185–1190.

Duc, T. L., Leiva, R. G., Casari, P. and Östberg, P.-O. (2019). Machine learning methods for reliable resource provisioning in edge-cloud computing: A survey, *ACM Computing Surveys (CSUR)* **52**(5): 1–39.

Erdil, D. C. (2013). Autonomic cloud resource sharing for intercloud federations, *Future Generation Computer Systems* **29**: 1700–1708.

Goodarzy, S., Nazari, M., Han, R., Keller, E. and Rozner, E. (2020). Resource management in cloud computing using machine learning: A survey.

Grewal, R. K. and Pateriya, P. K. (2013). *A Rule-Based Approach for Effective Resource Provisioning in Hybrid Cloud Environment*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 41–57.
URL: *https://doi.org/10.1007/978-3-642-35461-8*$_5$

Haratian, P., Safi-Esfahani, F., Salimian, L. and Nabiollahi, A. (2019). An adaptive and fuzzy resource management approach in cloud computing, *IEEE Transactions on Cloud Computing* **7**(4): 907–920.

Huang, C.-J., Guan, C.-T., Chen, H.-M., Wang, Y.-W., Chang, S.-C., Li, C.-Y. and Weng, C.-H. (2013). An adaptive resource management scheme in cloud computing, *Engineering Applications of Artificial Intelligence* **26**(1): 382–389.

Jayaprakash, S., Nagarajan, M. D., Prado, R. P. d., Subramanian, S. and Divakarachari, P. B. (2021). A systematic review of energy management strategies for resource allocation in the cloud: Clustering, optimization and machine learning, *Energies* **14**(17): 5322.

Jixian Zhang, Ning Xie, X. Z. K. Y. W. L. D. K. (2018). Machine learning based resource allocation of cloud computing in auction, *Computers, Materials & Continua* **56**(1): 123–135.
URL: *http://www.techscience.com/cmc/v56n1/27810*

Khan, T., Tian, W., Zhou, G., Ilager, S., Gong, M. and Buyya, R. (2022). Machine learning (ml)-centric resource management in cloud computing: A review and future directions, *Journal of Network and Computer Applications* **204**: 103405.

Kovács, G., Benotsmane, R. and Dudás, L. (2019). The concept of autonomous systems in industry 4.0, *Advanced Logistic Systems - Theory and Practice* **12**: 77–87.

Kumar, Y., Kaul, S. and Hu, Y.-C. (2022). Machine learning for energy-resource allocation, workflow scheduling and live migration in cloud computing: State-of-the-art survey, *Sustainable Computing: Informatics and Systems* **36**: 100780.

Murad, S., Muzahid, A., Azmi, Z., Hoque, M. and Kowsher, M. (2022). A review on job scheduling technique in cloud computing and priority rule based intelligent framework, *Journal of King Saud University - Computer and Information Sciences* **34**.

Nasr, A. A. (2022). A new cloud autonomous system as a service for multi-mobile robots, *Neural Computing and Applications* **34**(23): 21223–21235.
URL: *https://doi.org/10.1007/s00521-022-07605-7*

Singh, A., Juneja, D. and Malhotra, M. (2017). A novel agent based autonomous and service composition framework for cost optimization of resource provisioning in cloud computing, *Journal of King Saud University - Computer and Information Sciences* **29**(1): 19–28.
URL: *https://www.sciencedirect.com/science/article/pii/S1319157815000841*

Singhal, A., Varshney, S., Mohanaprakash, T., Jayavadivel, R., Deepti, K., Reddy, P. C. S. and Mulat, M. B. (2022). Minimization of latency using multitask scheduling in industrial autonomous systems, *Wireless Communications and Mobile Computing* **2022**: 1–10.

Tan, B., Ma, H., Mei, Y. and Zhang, M. (2020). A cooperative coevolution genetic programming hyper-heuristics approach for on-line resource allocation in container-based clouds, *IEEE Transactions on Cloud Computing* **PP**: 1–1.

Wang, J.-B., Wang, J., Wu, Y., Wang, J.-Y., Zhu, H., Lin, M. and Wang, J. (2018). A machine learning framework for resource allocation assisted by cloud computing, *IEEE Network* **32**(2): 144–151.

Werkhoven, P., Kester, L. and Neerincx, M. (2018). Telling autonomous systems what to do, *Proceedings of the 36th European Conference on Cognitive Ergonomics*, ECCE '18, Association for Computing Machinery, New York, NY, USA.
URL: *https://doi.org/10.1145/3232078.3232238*