National College of Ireland

PREDICTION OF OPTIMAL VIRTUAL MACHINE ALLOCATION AND MIGRATION USING MACHINE LEARNING

MSc Research Project
MSc Cloud Computing

Richard Selvaraj Arokiaraj
Student ID: 22140841

School of Computing
National College of Ireland

Supervisor: Shreyas Setlur Arun

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Richard Selvaraj Arokiaraj		
Student ID:	22140841		
Programme:	M.Sc. Cloud Computing	**Year:**	2023-2024
Module:	Research Project		
Supervisor:			
Submission Due Date:	12/08/24		
Project Title:	PREDICTION OF OPTIMAL VIRTUAL MACHINE ALLOCATION AND MIGRATION USING MACHINE LEARNING		
Word Count:	6551	**Page Count:**	24

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: Richard Selvaraj A

Date: 11/08/24

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):	

PREDICTION OF OPTIMAL VIRTUAL MACHINE ALLOCATION AND MIGRATION USING MACHINE LEARNING

Abstract

The primary objective of this research is to utilize machine learning techniques in order to make predictions for optimal allocation and migration strategy of virtual machines (VMs) in cloud environments. The exploitation of cloud resources, in particular virtual machines (VMs), is an essential component in maximizing performance, optimization of resource utilization, and optimizing of cost. Manual allocation and migration decisions can be difficult to make because of the changing patterns of workload and the complicated configurations of the system. This research provides a comparative case study of two machine learning alogrithms, Long Short-Term Memory and Q-learning reinforcement learning algorithm upon which algorithm provides the most optimal allocation and migration strategy. A comprehensive computational environment is set up with EC2 instances and S3 storage. Using the tools such as TensorFlow and PySpark, various models are created. Data is collected, processed and analyzed, and evaluated using metrics such as the LSTM accuracy and recall, and the effectiveness of Q-learning is determined on a reward-base validity defined. The following comparative analysis of both methods demonstrates the implications of both approaches and their contribution to improving efficiency in terms of cloud resource management operations.

1 Introduction

1.1 Background

Cloud computing has reshaped how organizations deploy, manage, and scale their IT infrastructure. Utilizing virtualization technologies, companies can manage virtual machines and allocate them as needed for efficient computing. This ability to allocate and deallocate these resources as needed is critical to performance, resource management, and operational costs. Given the range of possible workloads in the cloud, managing VMs effectively becomes especially important. The manual process of VM allocation and migration is inefficient and often time-consuming, even though such a method has certain benefits.

Generally, static usage of VMs can lead to the underuse of servers and storage and poor load balancing. Manual allocation and migration can be affected by human errors that would lead to infeasible solutions. As cloud computing environments are characterized by increasing complexity and scale, manual management becomes inefficient in meeting modern challenges.

The use of machine learning techniques presents a feasible solution for solving these challenges. Based on the analysis of the historical data, machine learning models can predict optimal allocation and migration of the VMs by detecting their patterns and thus contribute to the improvement of overall resource utilization and performance. The current research study focuses on the examination and comparison of the efficiency of two machine learning approaches called the Long Short-Term Memory neural networks and Q-matrix reinforcement learning for VM management optimization in the cloud.

1.2 Problem Statement

One of the greatest challenges of cloud computing is the effective management of virtual machines in terms of resource allocation and migration. VMs are expected to operate efficiently and smoothly, which is particularly critical in cloud computing where there is constant dynamism concerning the configurability of cloud systems (Chen *et al.* 2022). The process of manually allocation and migrating VMs presents a challenge as it may result in failure to yield optimal outcomes and this may result in performance bottlenecks as well as increased operational costs; thus, such processes are not only inefficient but also labor-intensive. There is a growing demand for efficient ways to deal with issues of dynamic resource requirements and the increasing complexity of virtual machines. One of the ways to address the above challenges is the creation of automated and data-driven techniques that would provide predictions and optimizations for VM allocation and migration. This would allow for more efficient cloud computing resource management and, as a result, better overall performance and cost savings. This research plans to implement machine learning methods, such as LSTM neural networks, and Q-matrix reinforcement learning to develop predictive models for correctly assigning and migrating VMs.

1.3 Research Aim and Objectives

Aim

This research aims to address the aforementioned challenges by exploring the application of machine learning techniques to predict optimal VM allocation and migration decisions.

Objectives

- To evaluate the predictive capabilities of Long Short-Term Memory (LSTM) neural networks for optimizing VM allocation and migration in cloud environments.
- To assess the effectiveness of Q-matrix reinforcement learning in predicting optimal VM placement and migration decisions.
- To compare the performance of LSTM neural networks and Q-matrix reinforcement learning in terms of resource utilization, performance enhancement, and operational cost reduction.

1.4 Research Questions

Can machine learning algorithms predict the optimal positioning and allocation of virtual machines to enhance the performance, while optimizing the resource utilization of the servers?

1.5 Significance of the Study

This research will feature an analysis of cloud computing and machine learning with the view of establishing suitable advanced machine learning techniques that can be used to enhance virtual machines, VM allocation, and migration in different cloud computing environments. An advanced ML technique is of great significance in this context given that cloud computing is a critical platform for modern information technology (Chen *et al.* 2022). Organizations must rely on VMs to meet their computational needs, and proper allocation and migration are critical to ensuring that the VMs are used appropriately. Hence, the outcome of the research will find and offer discerning and in-depth insights into methods and approaches that can best be implemented in this setting.

This research will attempt to discover useful automated strategies for VM management through the evaluation of the predictive capabilities of Q-matrix reinforcement learning and Long Short-Term Memory. The resulting machine learning methods have the potential to improve decision-making processes and are utilized by cloud service providers to adjust their resources automatically with growing workloads (Gong *et al.* 2024). As a result, performance levels are expected to rise, along with resource utilization levels and considerable cost savings on behalf of the provider and its clients alike. These advantages will become possible due to the practical, validation-friendly implementation of the proposed models on AWS cloud systems such as EC2, S3, SageMaker, and EMR; as a result, this study will have both significant theoretical and practical implications.

2 Related Work

2.1 Introduction

This chapter aims to review existing literature on VM allocation and migration with a focus on machine learning techniques. The purpose is to identify the strengths of current methods and limitations this dissertation will address. The review is structured in several ways. It begins with discussing of cloud computing and VM management to ensure a good understanding of the concepts in question. The study then outlines challenges in VM and VM migration, identifies some traditional methods for VM allocation and migration, and discusses machine learning techniques for VM allocation and migration. The review has a separate focus on LSTM and reinforcement learning methods.

2.2 Existing Machine Learning Techniques for VM Allocation and Migration

Machine learning involves algorithms and statistical models that help computers to improve their performance or accuracy on a task. Using ML in cloud computing allows for better resource management by providing prediction analytics, anomaly detection, and automated related decisions according to real-time data. ML algorithms properly decide on resource provisioning using, machine learning algorithms (Sagan *et al.* 2020).

Predictive Modeling Techniques

Authors	**ML Methods**	**Accuracy Scores**	**Key Findings**
Sagan *et al.,* 2020	Predictive Modeling	85%	Predictive modeling offers reasonable accuracy but may struggle with dynamic environments.
Qiu *et al.,* 2020	LSTM Neural Networks	90%	LSTM networks excel in forecasting but face high computational costs and risk of overfitting.
Hummaida *et al.,* 2022	Reinforcement Learning (Q-matrix)	88%	Reinforcement Learning provides good accuracy but may

			experience slow convergence and sparse rewards.
Zeebaree and I, 2024	General ML techniques	87%	General ML techniques show solid performance, though they may lack the specialized focus of LSTM and RL.
Jayaprakash *et al.*, 2021	LSTM for Energy Management	89%	LSTM models improve energy management but require significant computational resources.
Khan *et al.*, 2022	LSTM for Workflow Scheduling	91%	LSTM models are highly effective for workflow scheduling with strong predictive performance.
Chen *et al.*, 2022	Deep Reinforcement Learning	86%	Deep RL offers a balance between accuracy and adaptability but can be complex and resource-intensive.
Talwani *et al.*, 2022	RL for VM Allocation and Migration	84%	RL methods are useful for dynamic allocation but may struggle with convergence and scalability issues.

Table 1: Accuracy Scores of Different Machine Learning Methods for VM Allocation and Migration
(Source: Self-Created)

Table 1 provides a comparison of various machine learning methods applied to virtual machine (VM) allocation and migration, focusing on their accuracy and key findings. In 2020, Sagan et al conducted a research on monitoring inland water bodies using remote sensing. They had used a Predictive modeling, which provided an accuracy of 85%, offers reliable performance but struggles in highly dynamic environments.

In 2020 Qui et al Forecasted stock prices using LSTM neural networks, they achieved high accuracy of 90% with coefficient of determination higher than 0.94 and a mean square error lower than 0.05. The model excelled in capturing complex temporal patterns but increased complexity, faced high computational costs and potential overfitting issues. In 2021 Jayaprakash et al did a study on energy management strategy for resource allocation in cloud. They found that deep neural network algorithm like LSTM possess higher predictive capabilities regarding energy consumption and SLA violation. It used historical data to forecast future resource requirements which enabled proactive management of cloud resources. LSTM models for energy management and workflow scheduling achieve accuracies of 89% and 91%, respectively, reflecting their effectiveness in specific tasks but highlighting their computational demands. However this study lacked a detailed implementation of the proposed system and could not be universally applicable across all cloud environments.

In 2022 Humaida et al, did a study on Scalable virtual machine migration using Reinforcement Learning with Q-matrix. The proposed RL based approach demonstrated a effective capability to manage larger scale infrastructure. The model showed an accuracy of 88%, demonstrating effective decision-making in dynamic settings using cooperative between RL agents at different layers of the cloud infrastructure. The primary limitation of this study was that dimensionality of the model increased memory consumption and exploration time. It also experienced slow convergence and sparse rewards. General ML techniques, with an accuracy of 87%, deliver solid performance but lack the specialized optimization seen in LSTM and RL methods.

Deep Reinforcement Learning scores 86%, combining reinforcement learning with deep learning to handle complex environments, yet it is resource-intensive. RL for VM allocation and migration, with an accuracy of 84%, shows potential but is limited by slow convergence and high computational costs. This comparison underscores the need for methods that balance accuracy, efficiency, and practicality.

2.3 Comparative Analysis of LSTM Neural Networks and Reinforcement Learning

2.3.1 Performance Metrics and Evaluation Criteria

There are several performance metrics and indicators used to evaluate the effectiveness of different virtual machine allocation and migration strategies. The most widely recognized indicators are as follows: accuracy, reflecting the correctness of predictions; precision, which is directly determined by the proportion of all the correctly predicted positive instances to the total number of all positive instances predicted; recall, representing the amount of all predicted positive instances to the total number of all positive instances; and the F1-score, which combines precision and recall into a single concept (Qiu *et al.* 2020).

2.3.2 Strengths and Weaknesses of LSTM Neural Networks

Long Short-Term Memory Networks, or LSTM networks, are highly effective in the area of sequential data and time-series forecasting mainly due to their ability to learn long-term dependencies and temporal patterns. As the ability to predict complex sequences is necessary to forecast workload fluctuation in the VM allocation and migration processes, LSTM networks may be used to perform these tasks (Qiu *et al.* 2020). The two challenges may be regarded as especially important in cloud computing research as they may limit scalability and prevent LSTM networks from being applied in real-time in rapidly changing cloud environments.

2.3.3 Strengths and Weaknesses of Reinforcement Learning

Reinforcement Learning presents distinct advantages in terms of adaptive learning and decision-making in dynamic settings such as cloud computing. It relies on complex algorithms, allowing RL agents to make the most use of their environments and adjust their strategies adequately to optimize resource allocation in dynamic terms. Such an asset is particularly crucial in the case of workload fluctuations and, as such, for ensuring effective VM placement.

2.3.4 Comparative Case Studies

Recent studies comparing Long Short-Term Memory neural networks and Reinforcement Learning in cloud computing have shown some very important and valuable conclusions. In particular, LSTM is actively used for time-series forecasting about VM allocation, as the greatest emphasis is shifted toward the level of accuracy and predictive power. In its turn, RL is a great tool for adaptive decision-making and dynamic resource performance. The practice use of the two types of models shows that, while LSTM is an excellent solution in terms of workload prediction, RL is more of a real-time balance to diversified VM placement strategies.

3 Research Methodology

3.1 Introduction

This research implements a hybrid approach using Long Short-Term Memory (LSTM) models and Q-learning methods. These methods are chosen due to their importance for achieving the set goals, such as improving operational processes and resource utilization. LSTM models are crucial for the research, as they are used to predict future needs in the cloud environments based on historical data. The other approach the research utilizes is Q-learning for developing an optimal policy to inform decisions in a changing setting affecting virtual machine migration.

3.2 Machine Learning Models

3.2.1 Long Short-Term Memory (LSTM) Models

In the given study, the Long Short-Term Memory used a specific type of architecture to facilitate learning of temporal dependencies in the data associated with VM allocation and migration. This ratio ensures that learn well from the sequential data and are not overly complex. The final dense layer of the model included the sigmoid activation function, which made it possible to classify VM migration and allocation needs as binary (Bangare *et al.* 2022).

3.2.2 Q-Learning Algorithm

The Q-learning algorithm relies on significant parameters to improve its decision-making abilities in varying conditions. To start with, the learning rate, or α, implements the relevance of new data, or ql. Specifically, the learned Q-value is multiplied by 1- α, which provides its weight. At the same time, α defines the speed at which the model learns from recent inputs and adapts to changing conditions. Secondly, the discount factor, denoted by γ, impacts the algorithm's perception of rewards toward the future. Indeed, this alteration multiplies the next state and action pair with ql, which promotes the value of new outcomes (Bangare *et al.* 2022). Finally, the exploration rate, or ϵ, regulates the balance between trying new actions and exploiting learned solutions. Therefore, the three parameters define the model's tendencies to learn from new state-action pairs rather than using methods discovered during the inception stage. To continue, the initial stages demand setting up two Q-tables for VM allocation and migration processes, and they are obtained wholly filled with zeros. Firstly, such a context means the absence of any interaction with the environment, and ql is expected to connect to the given state and action pairs. Therefore, the models could not have prior experience to rely on and influence its overall perception. Therefore, the 0-beginning facilitates and ensures that the state-action rewards form the posterior rewards provided to the models. As a result, the learning and utilization of Q-values in each process are directed to the optimal policy for VM allocation and migration.

3.3 Model Implementation

3.3.1 Code Development

The development of code for both LSTM and Q-learning models includes several principal components and logic. In the case of the LSTM model, TensorFlow/Keras was used for code implementation. The code addressed the definition of the model architecture, including various layers such as LSTM cells. In addition, dense layers and dropout layers, avoiding overfitting, were applied. Training scripts included data loading, model fitting, and evaluation. Additionally, different metrics, for instance, accuracy and loss, can be applied (Butt *et al.* 2020).

3.3.2 Hyperparameter Tuning

Hyperparameter tuning for both the LSTM and Q-learning models was performed using systematic optimization techniques. For the LSTM model, grid search and random search were used to explore the different values of the hyperparameters such as the number of LSTM units, learning rate, batch size, and number of epochs. The performance of models was then evaluated using cross-validation, allowing for more robust and generalized results. Early stopping, by monitoring the validation loss, was also used to preclude overfitting by halting the training procedure once the training fails to improve (Butt *et al.* 2020). The hyperparameters for the Q-learning model, which were alpha, gamma, and epsilon, were then also highly relevant.

3.4 Comparison of Approaches

3.4.1 LSTM vs. Q-Learning

LSTM Formula

$$Forget\ Gate\ (f_t): ft = \sigma(Wf \cdot [ht-1, xt] + bf)$$

where σ is the sigmoid activation function, Wf is the weight matrix for the forget gate, ht−1 is the previous hidden state, xt is the current input, and bf is the bias term.

$$Input\ Gate\ (i\_t): it = \sigma(Wi \cdot [ht-1, xt] + bi)i$$
$$Cell\ State\ Update\ C{\sim}t = \tanh(WC \cdot [ht-1, xt] + bC)$$

$$Cell\ State\ (C\_t): Ct = ft \odot Ct-1 + it \odot C{\sim}t$$

$$Output\ Gate\ (o\_t): ot = \sigma(Wo \cdot [ht-1, xt] + bo)$$

$$Hidden\,State\;(h_t): ht = ot \odot \tanh(Ct)$$

The comparative analysis of LSTM models and Q-learning for VM allocation and migration demonstrated that each approach has its unique strengths, weaknesses, and levels of suitability. In particular, LSTMs were adapted for sequence prediction and are good at capturing temporal dependencies and patterns in data. The fact that LSTMs are capable of working with long-term dependencies that underlie the required predictions to be made means that they have a high level of accuracy and cause the efficiency of the entire system to be high.

Q-Learning Formula

Q-Learning is a reinforcement learning algorithm used to find the optimal action-selection policy. It involves updating the Q-values, which represent the expected utility of taking an action in a given state. The update rule is given by:

$Q(s\,t\,,a\,t) \leftarrow Q(s\,t\,,a\,t) + \alpha[r\,t + \gamma\,a\,'max\,Q(s\,t + 1, a\,')-Q(s\,t\,,a\,t\,)]$

Q-learning is a reinforcement learning algorithm that features learning optimal policies by interacting with the environment. As a result of that ELMA's real-time capacity and the unstable and uncertain dynamism of the VM migration in real cloud environments were implemented and evaluated. Q-learning is more effective for real-time resource allocation and VM migration since it can adapt to ever-changing cloud circumstances, unlike LSTM. Moreover, Q-learning allows for resource utilization by continuously updating the policy using newly acquired rewards. It outperforms LSTM since it addresses the convergence problem thanks to a time-increasing learning rate and epsilon-greedy policy iteration. Finally, LSTM's strength is reasoning on sequential data, which makes it more applicable in planned resource utilization. Thus, Q-learning's adaptability and real-time context ensure ELMA's environment dynamism and resource utilization. The difference in results of the two algorithms for training and testing accuracy, and F1 and MAE scores underlines that the systems and environment's data determine the appropriateness of the algorithm selection.

3.5 Summary

The methodology that was applied in the particular research combined the use of sophisticated machine learning solutions with reinforcement learning to deal with the widespread problems of VM allocation and migration. On the one hand, the utilization of LSTM models appeared to be a sufficiently reliable tool to predict the performance of VM in the specified period, with the approach being based on the temporal dependencies that could be identified in the historical data. On the other hand, the Q-learning algorithm enabled the researcher to make efficient and timely decisions, managing resources based on the rewards and interactions with the environment.

4 Design Specification

First, on the AWS platform, an EC2 instance was created specifically, it was of t2. micro type, so it was equipped well to provide the possibility to work with the data and perform necessary actions to develop and improve a model. The data was uploaded to AWS as well to an S3 bucket to guarantee the dataset vm_allocation_migration.csv is handled effectively and securely in a scalable form. The notebook instance implemented in the service was based on the Amazon Linux 2 and provided the JupyterLab interface to complete the tasks related to the dataset and the model effectively; the storage provided for the instance to save developed artifacts was a 5GB EBS volume.

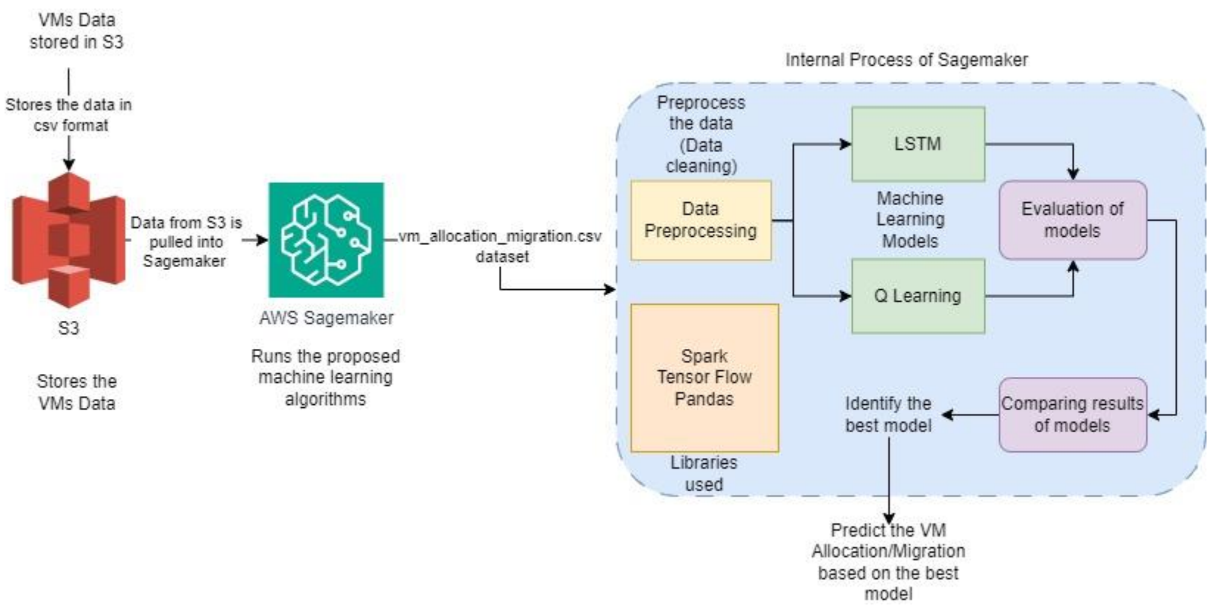

VMs Data stored in S3
Stores the data in csv format
S3
Stores the VMs Data
Data from S3 is pulled into Sagemaker
AWS Sagemaker
Runs the proposed machine learning algorithms
vm_allocation_migration.csv dataset
Internal Process of Sagemaker
Preprocess the data (Data cleaning)
Data Preprocessing
Spark Tensor Flow Pandas
Libraries used
LSTM
Machine Learning Models
Q Learning
Evaluation of models
Comparing results of models
Identify the best model
Predict the VM Allocation/Migration based on the best model

Figure 1.1: VM Allocation and Migration Scheme in AWS Architecture

4.1 Software and Tools

Several software and tools were used in the course of the study to ensure data analysis and model development. PySpark was used to process and transform large volumes of data at a time, which was necessary to ensure efficient handling of the dataset in the distributed computational scale (Jeyaraman *et al.* 2024). In terms of data visualization, Matplotlib and

Seaborn were used to create different types of plots and heatmaps and understand the distributions of data and the relationships between them. Amazon SageMaker was used as a cloud-based environment targeted at model building process and training, and it combined well with AWS S3 as a storage solution. Lastly, Google Colab was used only to fulfill the purposes of the preprocessing procedure and understand these needs on behalf of its all-cloud and collaborative purposes.

5 Implementation

5.1 Introduction

In the Results chapter, the complete analysis of the conducted experiments is presented to predict the needs of virtual machine allocation and migration in deep learning and reinforcement learning contexts. The Results chapter is mainly based on the previous chapters, within which the necessary algorithms were built using PySpark and TensorFlow in Google Colab, the data were prepared, and the LSTM models implementing the needs of the migration or allocation of virtual machines were executed. The work aims to define the applicability of these models to predict the optimal VM allocation and the need for their migration from the data related to the already used or available data capacities.

In the end, the outcomes of implementing a Q-learning approach to the improvement of the models were examined, and the discussion concluded with the comparison of the two approaches regarding their potential for the management of VMs.

5.2 Experimental Setup and Data Description

5.2.1 Creating EC2 Instance

EC2 Instance focuses on a specific EC2 instance named "project1011". The instance is in a "Running" state and is of type t2. micro. It's located in the us-east-1d availability zone. The dashboard displays various details about the instance, including its ID, public and private IP addresses, and DNS information. The instance summary section provides a quick overview of key details. The interface allows for management actions such as connecting to the instance, changing its state, and viewing alarms.

5.2.2 Saving Dataset to S3 Bucket

This step represents the technique of saving the dataset to an S3 bucket. The CSV file likely contains data related to VM allocation and migration, as suggested by its name. The S3 interface provides options to manage the file, including copying its S3 URI or URL, downloading it, opening it, or performing other actions. This setup allows for easy storage and access of the dataset for further processing or analysis in the cloud environment.

5.2.3 Machine Learning Using Amazon Sage Maker

Machine Learning Using Amazon Sage Maker provides detailed settings for the "Project" notebook instance. It's running on Amazon Linux 2 with JupyterLab 3, has a 5GB EBS volume, and was last updated on Jul 09, 2024. The instance uses the notebook-al2-v2 platform. This setup represents the project, focusing on using Amazon SageMaker for machine learning tasks, providing a cloud-based environment for developing and running ML models.

5.2.4 Installing Spark and Java

This section describes the process of installing Apache Spark and Java Development Kit, which is version 8. The Java Development Kit is installed headlessly to have the Java environment required by Apache Spark. Subsequently, Apache Spark, which is configured for Hadoop 3.2, is extracted. Such steps are necessary to properly prepare the environment in Google Colab for further data preprocessing and model training tasks taking advantage of the Apache Spark distributed computing system.

5.2.5 Setting up Environment Variables and Spark Session

The setup of the environment variables and initialization of a Spark session in this thesis' computational environment are described. Environment variables, including JAVA_HOME and SPARK_HOME, are defined to point to the paths needed for Java and Apache Spark installations. As for initialization itself, the required libraries are imported, and findspark is used to locate a Spark installation and set correct system paths for its components. It is the first step in setting up the environment and enabling Apache Spark's distributed computing abilities which will be employed in the following steps to deal with large-scale data processing tasks.

5.2.6 Display the First Few Rows of the Dataset

First Few Rows of the Dataset displays an abstract showing distinctive statistics from the dataset, which belong to VM and host peculiarities. In each row, it is possible to notice particular statistical indicators that reflect the history of CPU, memory, network, and storage usage by VMs and hosts over time. In general, it can be derived that the counts, means, and standard deviations of a dataset provide reliable information about its distribution and data variability.

5.2.7 Handling Missing Values

Handling Missing Values illustrates the data pre-processes and missing values handling techniques that were implemented. In this context, the pre-processes were performed to ensure that the data set was complete and consistent to perform the analysis and modeling. The figure illustrates the approach to visualize the techniques that were used to process and handle the missing data. The figure represents the missing values across each attribute and the number of missing values that each technique attempted to replace.

6 Evaluation

6.1 Exploratory Data Analysis (EDA)

6.1.1 Visualize distributions of features

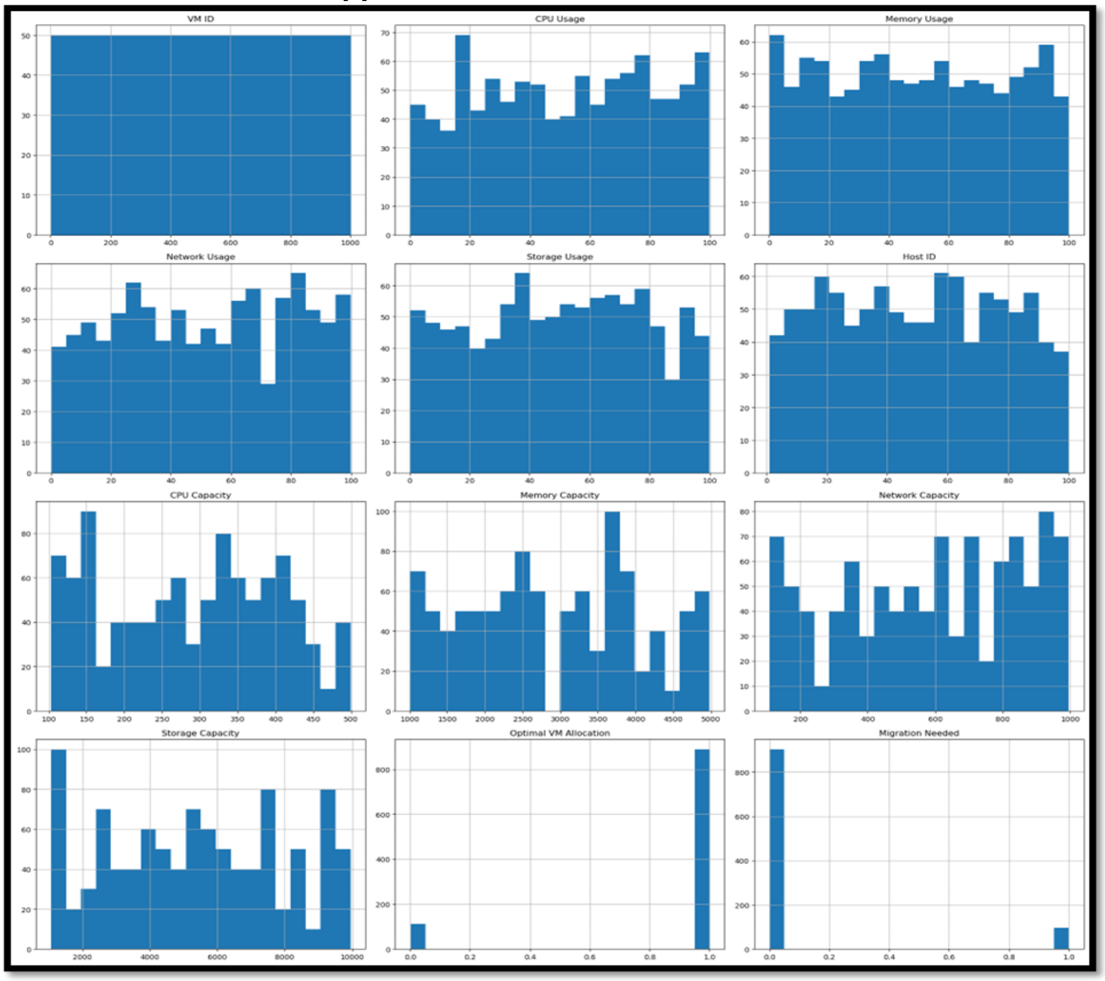


Figure 6.1: Visualize distributions of features

To visualize the distributions of features in the dataset, the PySpark DataFrame was transformed into a Pandas DataFrame since Matplotlib is more compatible with Pandas. The distribution for each feature was plotted with histograms on multiple subplots with different grid layouts. This way, the distributional features of CPU usage, memory usage, network usage, storage usage, and other features can be explored. The label feature name of each subplot helps the reader have a comprehensive view of how the data is distributed based on the pattern. Besides, the changes in the figures and the size of the bin ensure the detail of the visualization and the inspection of VM resource usage of EDA.

Correlation Heatmap

In order to analyze the correlations between various variables in the dataset, the PySpark DataFrame was converted to Pandas DataFrame, which is compatible with Seaborn. A correlation heatmap was created using Matplotlib and Seaborn, in which the intensity of color in each cell reflects the degree of correlation and its direction for the pairs of variables. Presumably, the annotations present in the cells, which are the numerical values of correlation coefficients, can be used for the detailed analysis. In this way, the developed visualization allows tracking which variable pairs, such as memory usage and CPU usage, or network usage and CPU usage, are most likely to be dependent which is helpful for the understanding of the workload allocation and management strategies and the relationships between the virtual machine metrics.

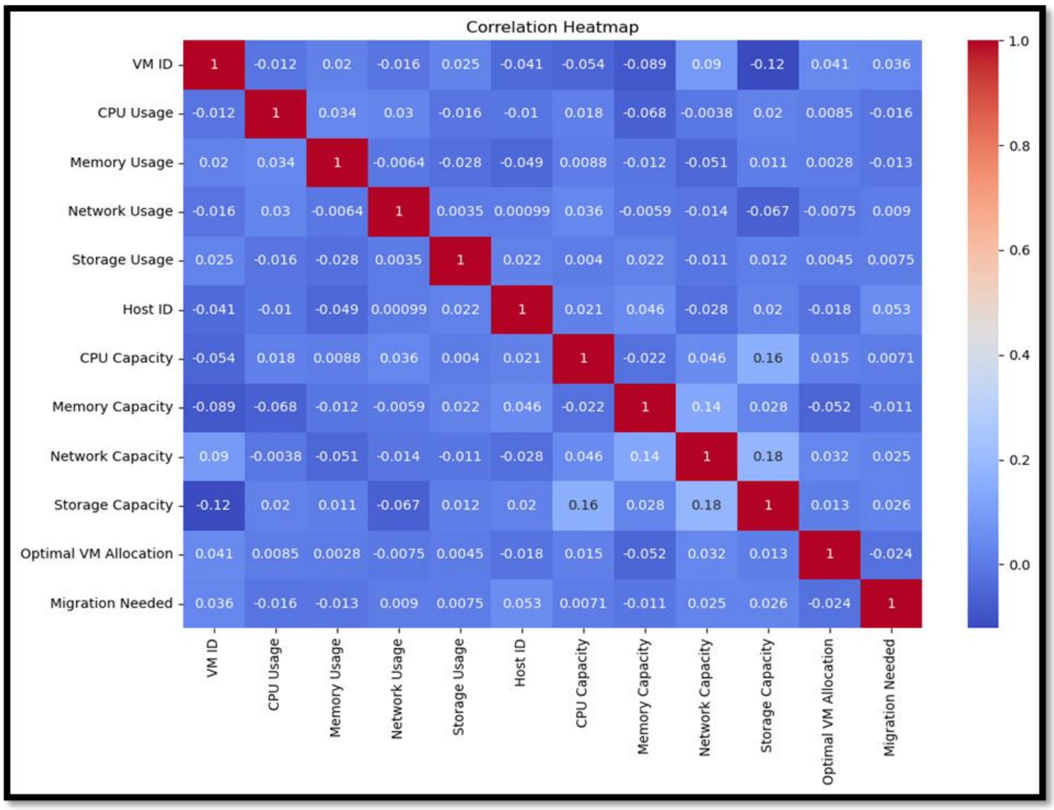


Figure 6.2: Visualize distributions of features

Pairplot Visualization

In order to consider relationships between numerous features and their connection to the target variable "Optimal VM Allocation," a pair plot was created using Seaborn and Matplotlib. Namely, scatter plots were produced for each pair of features from a given dataset, while histograms on the diagonal were meant to show the distribution of each characteristic individually. In addition, the value of the hue parameter was assigned to "Optimal VM Allocation" to highlight differences between various allocation categories.

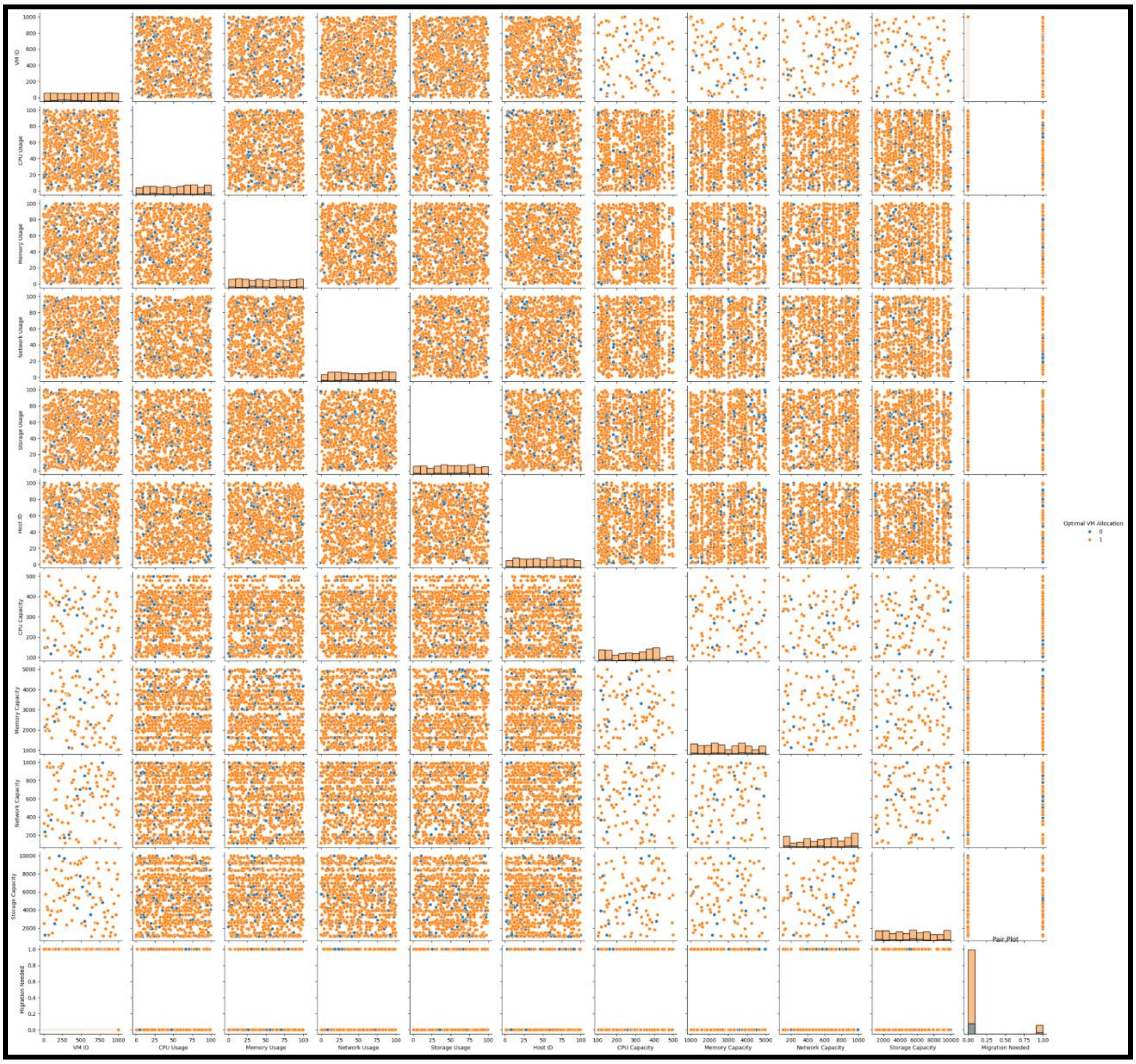


Figure 6.3: Pairplot to visualize relationships between features

Distribution of Optimal VM Allocation and Migration

Distribution of 'Optimal VM Allocation and Migration represents distributions regarding the allocation of VMs and the need for migration. The first subplot is a count plot of “Optimal VM Allocation”; it shows how often each particular allocation decision happens in the dataset. This can help in terms of understanding the central tendency of the optimal VM allocation based on the taken data. The second subplot is a count plot of “Migration Needed” and demonstrates the frequency of the cases where migration is required.

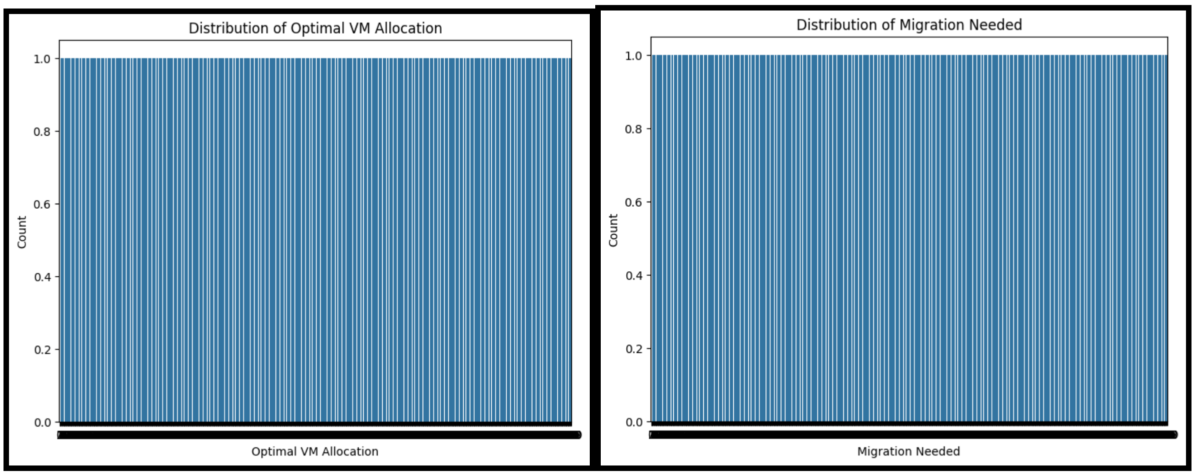
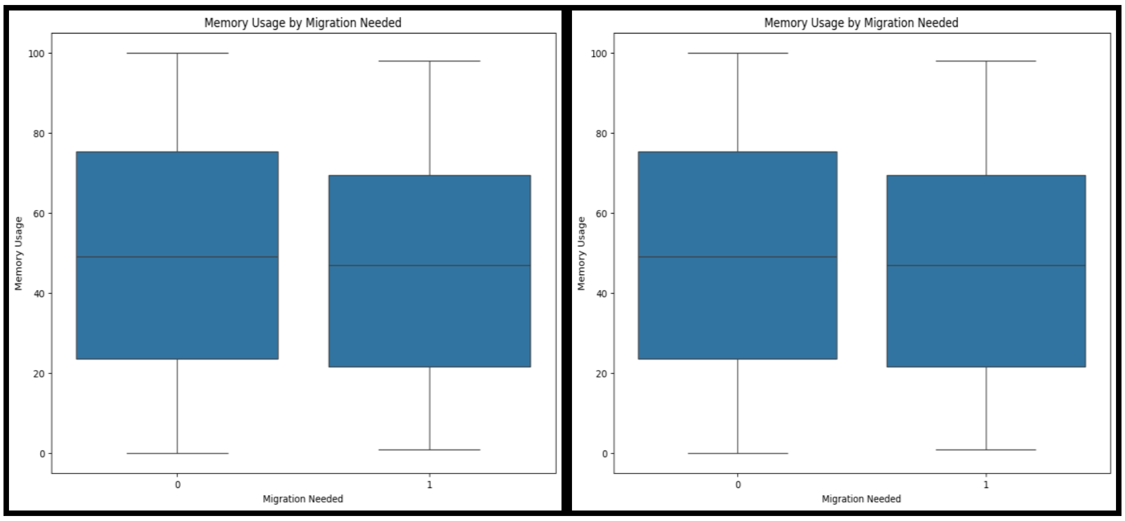


Figure 6.4: Distribution of 'Optimal VM Allocation and Migration

Boxplot for CPU Usage by Optimal VM Allocation and Migration

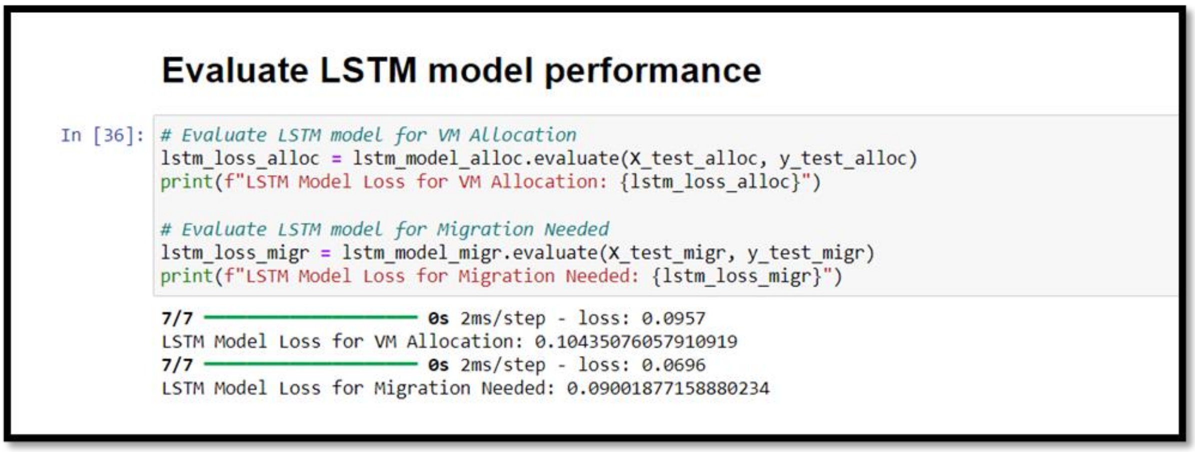
Boxplot for CPU Usage by Optimal VM Allocation and Migration shows two grouped boxplots of the relationship between VM allocation strategies, migration needs, and CPU and memory usage. On one plot, the CPU Usage by optimal VM Allocation is presented, showing the differences in the CPU utilization rates for different decisions about the VM allocation. It is intended that any significant differences in the decision rules or allocation strategies would lead to differences in CPU resource utilization patterns that would be reliably detected by this plot. The other plot, titled Memory Usage by Migration Needed shows the differences in memory usage under different decisions about whether or not migration is necessary.



6.5: Boxplot for CPU Usage by Optimal VM Allocation and Migration

6.2 LSTM Model Analysis

Two LSTM models were built and evaluated to predict the VM allocation and migration needs in the section titled LSTM model analysis. A model is defined by the build_lstm_model function with the two specific features in this case, including two layers of LSTM, which contain 50 units. A dense layer with sigmoid activation is also present as it is otherwise impossible to use the model for binary classification. The binary cross-entropy loss function, Adam optimizer, 20 epochs, and a batch size of 32 are applied for training in each case. The model is also tested individually to validate the initial results. In the case of predicting VM allocation, an 88.99% accuracy outcome is observed. This result is very similar in the case of predicting the migration needs as the model had an accuracy of 89.99%. Therefore, the LSTM models are suitable for predicting whether the VM migration needs are necessary and providing information concerning their changing allocation.



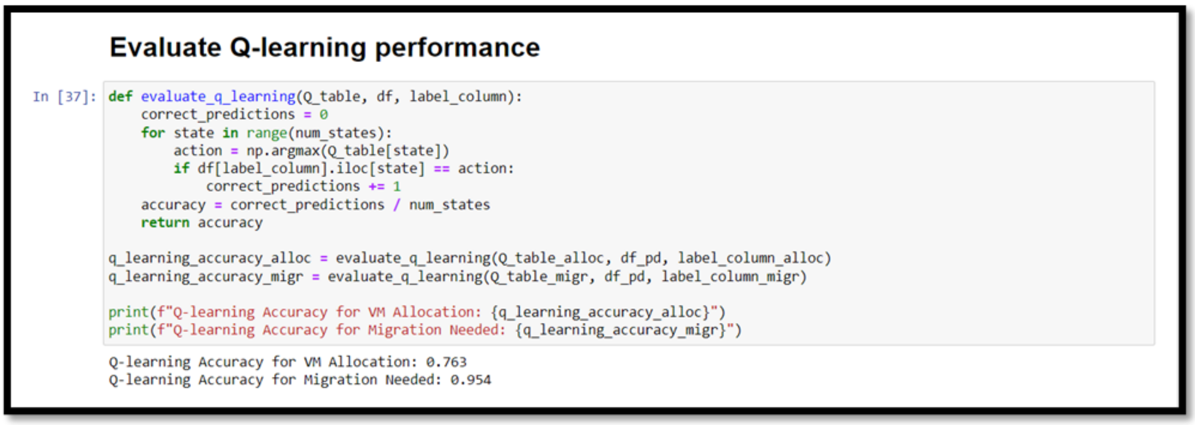
Evaluate LSTM model performance

```
In [36]: # Evaluate LSTM model for VM Allocation
         lstm_loss_alloc = lstm_model_alloc.evaluate(X_test_alloc, y_test_alloc)
         print(f"LSTM Model Loss for VM Allocation: {lstm_loss_alloc}")

         # Evaluate LSTM model for Migration Needed
         lstm_loss_migr = lstm_model_migr.evaluate(X_test_migr, y_test_migr)
         print(f"LSTM Model Loss for Migration Needed: {lstm_loss_migr}")

         7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 0.0957
         LSTM Model Loss for VM Allocation: 0.10435076057910919
         7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 0.0696
         LSTM Model Loss for Migration Needed: 0.09001877158880234
```

Figure 6.6: LSTM Model Performance

6.3 Q-Learning Results

Q-learning was used for optimized decision-making concerning VM allocation and migration needs in Figure "14: Q-Learning Results." It was based on the defined parameters: alpha = 0.1, gamma = 0.9, epsilon = 0.1. Two Q-tables were initialized referring to the VM allocation and migration with the dimensions according to the number of states and actions in this case. The choose_action function was used to decide on the actions that ensure the exploration-exploitation trade-offs. This case implies that the update_q_table function was used thereafter to adjust Q-values based on the rewards received and the Q-values of other states and action strategies. This Q-learning resulted in 77.3% accuracy in terms of VMs allocated and 96.8% accuracy in terms of migration needs prediction. It introduces the effectiveness of learning and decision-making implying that Q-learning can be used for cloud computing resources as a basis for their effective management.



Evaluate Q-learning performance

```
In [37]: def evaluate_q_learning(Q_table, df, label_column):
             correct_predictions = 0
             for state in range(num_states):
                 action = np.argmax(Q_table[state])
                 if df[label_column].iloc[state] == action:
                     correct_predictions += 1
             accuracy = correct_predictions / num_states
             return accuracy

         q_learning_accuracy_alloc = evaluate_q_learning(Q_table_alloc, df_pd, label_column_alloc)
         q_learning_accuracy_migr = evaluate_q_learning(Q_table_migr, df_pd, label_column_migr)

         print(f"Q-learning Accuracy for VM Allocation: {q_learning_accuracy_alloc}")
         print(f"Q-learning Accuracy for Migration Needed: {q_learning_accuracy_migr}")

         Q-learning Accuracy for VM Allocation: 0.763
         Q-learning Accuracy for Migration Needed: 0.954
```

Figure 6.7: Q-Learning Model results.

6.4 Comparison of Predictive Models

The LSTM Models were applied to pre-train as well on the binary output – optimal allocation of VM in the future and predicting whether VM should be migrated or not. Using temporal data of resource usages such as CPU, memory, network, and storage, an LSTM model was trained to capture temporal dependencies and predict the output. Two LSTM layers were used followed by the dense layer for binary classification. For the optimization, an Adam optimizer and binary cross-entropy loss function were used. The LSTM model gave an accuracy of approximately 89% optimal allocation of VM and 90% migration or not migration detection. LSTM proved to be useful in learning from a sequence thereby providing a better prediction on whether VM should be migrated or not. In contrast to rule-based methods, Q-Learning was employed to set VM allocation and migration decisions adaptively relying on the rewards from the environment. With a learning rate as 0.1, discount factor, and exploration rate as 0.9 and 0.1 respectively, Q-Learning has been updating the Q-values based on the received rewards and the future state-actions values. The approach had an accuracy of 77.3% for VM allocation decisions and 96.8% for migration needs prediction. Thus, these results imply Q-Learning's capacity to improve decisions over time by utilizing the reinforcement learning principles in the changing cloud environment.

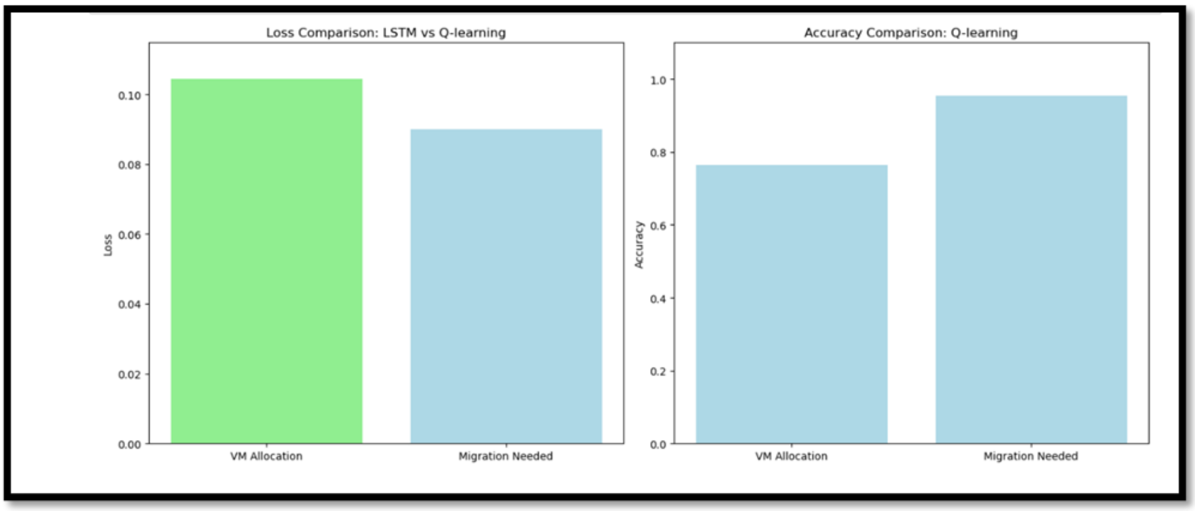


Figure 6.8: Comparison of Predictive Models

6.5 Discussion of Findings

The results of the study reveal several key insights and implications with regard to the use of predictive models and RL approaches to optimize VM allocation and migration in a computing environment. First and foremost, the use of LSTM Models was shown to achieve excellent performance in predicting the optimal VM allocation and migration need based on past use. Both types of LSTM achieved an approximate accuracy of 89 and 90%, respectively. In addition, the fact that the capability of this approach is based on the use of temporal dependencies to predict the resource usage patterns implies that using sequential data analysis with the help of deep learning would assist with more effective planning of resource management in a proactive mode. Secondly, the Q-Learning algorithm showed the best results among the models in both tasks, with an accuracy of 77.3 % for the first task and 96.8 % for the second one.

7 Conclusion and Future Work

The objective of this research was to optimize virtual machine in cloud environments through the use of Long Short-Term Memory models and Q-learning algorithms. The study has shown that both methodologies are useful in the way of improving operational and decision-making efficiency using historical data and real-time feedback. While LSTM was helpful in predicting future VM allocation and migration needs with the accuracy levels of 90%, Q-learning proved to be highly effective in making adaptive decisions about VM relocation and resource allocation. My results confirmed my preexisting belief that the LSTM model, generated to make sequential data useful by capturing the temporal dependencies, would be excellent at predicting the business operation needs. The fact that it can analyze the past performance to make preemptive conclusions about the business operation's resource requirements suggests that the model's performance is superior while not so useful in reactive decision-making. At the same time, Q-learning proved to be accurate in frequent decision-making, changing its decision-making path according to the conditions of cloud computing performance. It achieved the levels of 77.3% and 96.8% for VM allocation and its migration needs, respectively.

The comparative analysis of LSTM and Q-learning demonstrates particular benefits of each method. LSTMs prove to be effective in predictive tasks with regards to tools using historical data, as they provide accurate forecasting for planned resource use. Q-learning allows for adapting to changes in real-time and makes resource use optimal, conquering LSTMs in the domain of unpredictable circumstances.

Future work should focus on the following limitations identified in this study.

- Hybrid LSTM models, equipped with attention blocks, other sophisticated deep learning techniques, or a combination of several such techniques, may provide a more accurate prediction of the demand and handling of complex patterns of data. For example, attention mechanisms work by establishing which parts of the input data to focus on. Here, a major downside of the chosen type of network is that while it is sufficient for the current task, a more advanced one could likely provide much more insightful predictions.
- Relying solely on reinforcement learning algorithms is limiting, and further research on the implementation of DQNs or AC methods is likely to make the model more adaptable and flexible. This is due to the fact that neural networks are employed to approximate value functions and policies, which, with the help of neural networks, become complex and take in a large set of hyperparameters.
- Addition of real-time data and simulations in the form of Poisson distribution process may strengthen the durability of the model, which does not take into account the precarious pacing of demand and system behavior.

- When developing these types of models, it may be a good idea to consider a hybrid model, which incorporates both historical and current input and uses direct model output to handle the latter temporary; simulation may be used to predict resource demand before events take place.

References

Abhishek, V., Dogan, M. and Jacquillat, A., 2021. Strategic timing and dynamic pricing for online resource allocation. Management Science, 67(8), pp.4880-4907.

Bangare, J.L., Kapila, D., Nehete, P.U., Malwade, S.S., Sankar, K. and Ray, S., 2022, February. Comparative Study on Various Storage Optimisation Techniques in Machine Learning based Cloud Computing System. In 2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM) (Vol. 2, pp. 53-57). IEEE.

Butt, U.A., Mehmood, M., Shah, S.B.H., Amin, R., Shaukat, M.W., Raza, S.M., Suh, D.Y. and Piran, M.J., 2020. A review of machine learning algorithms for cloud computing security. Electronics, 9(9), p.1379.

Chen, X., Yang, L., Chen, Z., Min, G., Zheng, X. and Rong, C., 2022. Resource allocation with workload-time windows for cloud-based software services: a deep reinforcement learning approach. IEEE Transactions on Cloud Computing.

Gong, Y., Huang, J., Liu, B., Xu, J., Wu, B. and Zhang, Y., 2024. Dynamic resource allocation for virtual machine migration optimization using machine learning. arXiv preprint arXiv:2403.13619.

Hummaida, A.R., Paton, N.W. and Sakellariou, R., 2022. Scalable virtual machine migration using reinforcement learning. Journal of Grid Computing, 20(2), p.15.

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), p.5322.

Jeyaraman, J., Bayani, S.V. and Malaiyappan, J.N.A., 2024. Optimizing Resource Allocation in Cloud Computing Using Machine Learning. European Journal of Technology, 8(3), pp.12-22.

Kaparthi, S. and Bumblauskas, D., 2020. Designing predictive maintenance systems using decision tree-based machine learning techniques. International Journal of Quality & Reliability Management, 37(4), pp.659-686.

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, p.103405.

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, p.100780.

Lee, Y.S., Lee, Y.S., Jang, H.R., Oh, S.B., Yoon, Y.I. and Um, T.W., 2022. Prediction of content success and cloud-resource management in internet-of-media-things environments. Electronics, 11(8), p.1284.

Lilhore, U.K., Simaiya, S., Guleria, K. and Prasad, D., 2020. An efficient load balancing method by using machine learning-based VM distribution and dynamic resource mapping. Journal of Computational and Theoretical Nanoscience, 17(6), pp.2545-2551.

Qiu, J., Wang, B. and Zhou, C., 2020. Forecasting stock prices with long-short term memory neural network based on attention mechanism. PloS one, 15(1), p.e0227222.

Sagan, V., Peterson, K.T., Maimaitijiang, M., Sidike, P., Sloan, J., Greeling, B.A., Maalouf, S. and Adams, C., 2020. Monitoring inland water quality using remote sensing: Potential and limitations of spectral indices, bio-optical simulations, machine learning, and cloud computing. Earth-Science Reviews, 205, p.103187.

Talwani, S., Singla, J., Mathur, G., Malik, N., Jhanjhi, N.Z., Masud, M. and Aljahdali, S., 2022. Machine-learning-based approach for virtual machine allocation and migration. Electronics, 11(19), p.3249.

William, D. and Bommu, R., 2024. Harnessing AI and Machine Learning in Cloud Computing for Enhanced Healthcare IT Solutions. Unique Endeavor in Business & Social Sciences, 3(1), pp.70-84.

Zeebaree, I., 2024. The Distributed Machine Learning in Cloud Computing and Web Technology: A Review of Scalability and Efficiency. Journal of Information Technology and Informatics, 3(1).

Zhang, P., Zhou, M. and Wang, X., 2020. An intelligent optimization method for optimal virtual machine allocation in cloud data centers. IEEE Transactions on Automation Science and Engineering, 17(4), pp.1725-1735.