Task Failure Prediction in Cloud Computing

A Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of
Bachelor of Technology
in
Computer Science & Engineering

by
Himanshi Sharma 20204083
Hitakshi Jhamnani 20204084
Naman Mittal 20204121
Nityam Gupta 20204132

to the
COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
MOTILAL NEHRU NATIONAL INSTITUTE OF TECHNOLOGY
ALLAHABAD PRAYAGRAJ
May, 2022

UNDERTAKING

I declare that the work presented in this report titled "Task Failure Prediction in Cloud Computing", submitted to the Computer Science and Engineering Department, Motilal Nehru National Institute of Technology Allahabad, Prayagraj, for the award of the Bachelor of Technology degree in Computer Science & Engineering, is my original work. I have not plagiarized or submitted the same work for the award of any other degree. In case this undertaking is found incorrect, I accept that my degree may be unconditionally withdrawn.

May, 2022
Allahabad

Himanshi Sharma 20204083
Hitakshi Jhamnani 20204084
Naman Mittal 20204121
Nityam Gupta 20204132

CERTIFICATE

Certified that the work contained in the report titled "Task Failure Prediction in Cloud Computing", by Himanshi Sharma (20204083), Hitakshi Jhamnani (20204084), Naman Mittal (20204121), Nityam Gupta (20204132), has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Dr. Ashish Kumar Maurya
Computer Science and Engineering Dept.
M.N.N.I.T, Allahabad

May, 2022

Preface

A good B.Tech. thesis is one that helps you in furthering your interest in a specific field of study. Whether you plan to work in an industry or wish to take up academics as a way of life, your thesis plays an important role.

Your thesis should judiciously combine theory with practice. It should result in a realization of reasonably complex system (software and/or hardware). Given various limitations, it is always better to extend your predecessor’s work. If you plan it properly, you can really build on the experience of your seniors.

Acknowledgements

We would begin by thanking Ashish sir for all the guidance offered and all the faith he has shown in us. We cannot thank him enough for all the motivation he has provided us throughout our BTech Mini Project. We would like to also thank him for helping us in finding our interest and regularly helping us to pursue the field of work in which our interests lies. We are very thankful for all their help and support and owe them for all the belief they showed in us. Finally, we would like to acknowledge with gratitude, the support and love of our family, friends and seniors. They all kept us going and the completion of our project would not have been possible without them.

Contents

Chapter 1

Introduction

The cloud is a method of managing and providing software, platform, and infrastructure services over the internet. Task failure is an inevitable challenge in cloud computing due to the inherent complexity and dynamic nature of the cloud environment. Task failure can be defined as the point at which the system is no longer able to meet the task execution demand. When a task fails, it can disrupt the entire workflow, resulting in degraded performance, delayed completion times, and increased costs. Task failure can occur for several reasons, including hardware or software failures, resource constraints, network disruptions, and overload conditions.

In order to further improve the accuracy of the previous failure prediction algorithms based on machine learning and deep learning, a multi-layer Bidirectional Long Short Term Memory (BiLSTM) based failure prediction algorithm was developed in order to identify tasks and jobs that have failed in the cloud from a failure prediction standpoint.

1.1 What is Machine Learning?

Machine learning is a subset of artificial intelligence that focuses on the development of algorithms and statistical models that enable computer systems to learn from data, identify patterns and make predictions or decisions without being explicitly programmed. Machine learning algorithms can be divided into three main categories:

1. Supervised Learning: In this type of learning, the algorithm is trained on a labeled dataset, meaning that the desired output for each input is already known.
2. Unsupervised Learning: In this type of learning, the algorithm is trained on an unlabeled dataset, meaning that there is no predefined output for each input. The algorithm learns to identify patterns and relationships in the data without any guidance.

3. Reinforcement Learning: In this type of learning, the algorithm learns through trial and error, by receiving feedback in the form of rewards or punishments for its actions.

1.2 Deep Learning

Deep learning is a subset of machine learning that utilizes artificial neural networks with multiple layers to perform complex tasks such as image recognition, natural language processing, speech recognition, and more.

Deep learning models consist of multiple layers of interconnected artificial neurons that process and analyze data at each layer to make predictions or classifications. These neural networks can learn to recognize patterns and features in large datasets, making them particularly effective in solving complex problems.

1.3 Cloud Data Centres

A cloud data center is a facility that provides cloud computing services and hosts the hardware and software infrastructure required to support those services. These data centers typically house a large number of servers, storage systems, networking equipment, and other resources necessary to deliver cloud-based applications and services to users over the internet.

A cloud data center with a heterogeneous infrastructure and intensive workloads can be vulnerable to different types of failures, such as hardware failures, software failures, and disk failures as mentioned in Fig 1. These failures can have a significant impact on the availability, reliability, and performance of the cloud services provided by the data center.

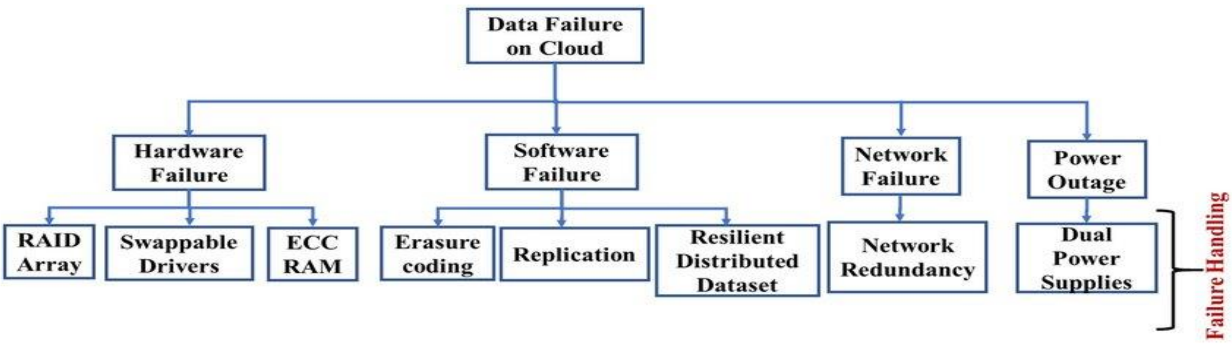


Fig:1 Cloud Failure Reasons [1]

1.4 Recurrent Neural Network

Recurrent Neural Network (RNN) is a type of neural network architecture that is particularly useful for processing sequential data. Unlike feedforward neural networks, which process inputs in a single pass and do not have any internal memory, RNNs have a "memory" that enables them to process sequences of inputs and maintain a representation of the past.

The basic building block of an RNN is a recurrent neuron, which takes an input and an internal state as inputs and produces an output and a new internal state as outputs. The output of the previous neuron in the sequence is used as the input to the next neuron, along with the current input and the current state.

1.5 Support Vector Machine

Support Vector Machine (SVM) is a type of supervised learning algorithm used for classification and regression analysis. In SVM, the goal is to find a hyperplane that best separates the data into different classes. The hyperplane is a decision boundary that maximizes the margin between the two classes. The margin is the distance between the hyperplane and the nearest data points from each class.

1.6 Long Short-Term Memory Network

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture that is designed to handle the vanishing gradient problem that can occur when training traditional RNNs.

LSTMs are particularly useful for processing sequential data, such as speech and natural language, and are designed to remember information over longer periods of time.

1.7 Bi-Long Short Term Memory Network

Bi-Directional Long Short-Term Memory (Bi-LSTM) is an extension of the LSTM architecture that allows the network to process sequential data in both forward and backward directions. The two sets of hidden units in a Bi-LSTM are connected by a special layer called a merge layer, which combines the outputs of the forward and backward LSTMs. This allows the network to capture information from both past and future contexts, making it particularly useful for tasks where the prediction of the current state depends on both preceding and following information.

1.8 Motivation

There have been several real-life examples of cloud data center failures that have impacted businesses and users. In January 2015, Yahoo and Microsoft's Bing search engine experienced a 20-minute outage, which cost an estimated $180,000 in lost revenue [2]. In another case a software failure in a cloud data center occurred in February 2017 when Microsoft's Azure cloud platform experienced a massive outage. In August 2012, Knight Capital Group, a financial services firm, experienced a major software failure in their cloud-based automatic stock trading software, which resulted in a trading loss of $440 million.These examples demonstrate the potential impact of cloud data center failures on businesses and users who rely on cloud services for their daily operations.

1.9 Uses of ML in different areas

Here are some points about the use of machine learning (ML) in different areas:
ML algorithms are used to assist with medical diagnoses, patient monitoring, drug development, and medical image analysis. ML is used in fraud detection, credit risk assessment, investment portfolio management, and algorithmic trading. ML is used to improve customer experience, enhance supply chain management, and personalize product recommendations. ML is used to personalize learning experiences, identify student learning difficulties, and improve educational outcomes.

1.10 Problem Statement

"Cloud data centers are critical to the operation of modern computing systems, and task failures can have a significant impact on their performance and reliability. Traditional approaches to detecting and predicting task failures are often reactive and based on rule-based systems, which may not be effective in handling the dynamic and complex nature of cloud computing environments. Therefore, there is a need to develop more accurate and efficient techniques for task failure prediction in cloud data centers.”

Chapter 2

Related Work

We classify the previous related work into two parts: failure analysis in cloud data centers and failure prediction.

i. Failure Analysis in Cloud Data Centers:

- Ford et al. [3] studied the impact of correlated failures on availability of distributed storage systems for Google clusters.

ii. Failure Prediction Methods:

- Zhao et al. [4] used HMM and Hidden Semi-markov Model to predict disk failures in cloud storage systems.
- Pitakratet al.[5] proposed a hierarchical online failure prediction approach called Hora. Hora employed a combination of a failure propagation model and software system failure prediction techniques based on Bayesian networks.
- Zhang et al. [6] designed and implemented a new tool based on Random Forest (RF) called PreFix, for accurately predicting whether there will be a switch failure in the near future.
- Das et al. [7] provided a powerful technique to process High Performance Computing (HPC) logs using LSTM for efficient failure prediction. They used a three-phase deep learning approach to first train to recognize chains of log events leading to a failure, second re-train chain recognition of events augmented with expected lead times to failure, and third predict lead times during testing/inference deployment to predict which specific node fails in how many minutes.
- Du et al. [8] presented DeepLog, which is used to predict task and job failures by using Long Short-Term Memory (LSTM) and density clustering approaches. The authors first preprocessed the log data generated by the tasks and jobs in a Hadoop distributed file system. They then used LSTM to model the log sequences and predict future task and job failures. In addition

to LSTM, the authors also used a density clustering approach to group similar log events together.

- Xu et al. [9] proposed an RNN-based model for predicting hard disk drive failure and giving health degrees. The proposed model treats the observed SMART (Self-Monitoring, Analysis and Reporting Technology) features such as disk usage and disk I/O time as time-sequence data. The RNN model used in the proposed approach is able to capture the long-term dependencies among the time-series data and make accurate predictions about the future health of the hard disk drive.

Chapter 3

Work done

3.1 Data Set

We hadn't used any standard dataset as none of the standard dataset consists of all the required attributes. We had collected data from several sources and had integrated that to prepare a complete dataset. Our dataset consists of 280896 records. The database is split into two sub-datasets: Training Data(80%) and Test Data(20%) which in numbers contain 224716 training records and 56180 test records.

3.2 Technologies used

The programs and technologies crucial to the realization of our project were:

- Python : Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.
- Keras and Tensorflow: TensorFlow is an open-sourced end-to-end platform, a library for multiple machine learning tasks, while Keras is a high-level 7 neural network library that runs on top of TensorFlow.
- Google Collab: Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.
- Google Drive: Along with colab google drive was used to manage and save all the files such as model weights, model structure, image dataset, etc.
- Pandas: Pandas is a popular open-source library for data manipulation and analysis in Python.
- Sklearn: Scikit-learn (also known as sklearn) is a popular open-source library for machine learning in Python. It provides a wide range of tools for supervised and unsupervised learning, including classification, regression, clustering, and dimensionality reduction.

3.3 Project Structure

The project is based on Bi-LSTM to predict task failure in cloud data centers. There are two phases: training and testing. During the training phase, the model is fed with data to learn and adjust its parameters to minimize the difference between the predicted output and the actual output.

In order to predict failures, we follow the following practical process. One machine, with a trained failure prediction machine learning model, runs a centralized failure prediction application in cloud computing. The application collects and records all the system logs of all the machines, including CPU and memory usage. Based on the collected data in the previous step, the agent in the Bi-LSTM prediction model will predict whether the tasks will fail or succeed. Cloud monitoring kills a job or task when it is predicted to fail and restarts it when it is predicted to succeed.

3.3.1 Model Architecture

An example of the architecture of Bi-LSTM model is shown in Fig 2, which has an input layer, two Bi-LSTM layers, an output layer, and a Logistic Regression (LR) layer to determine whether a task or job has been successful or failed.

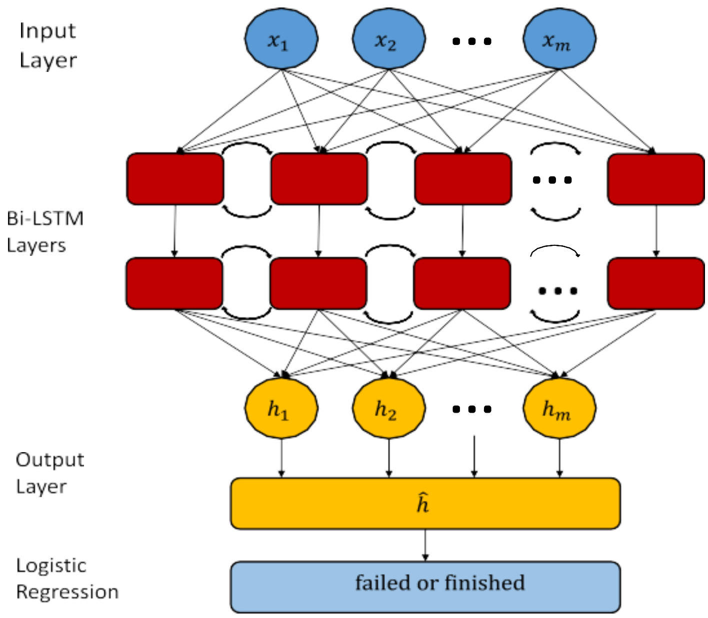


Fig 2: Architecture of multi-layer Bidirectional LSTM (Bi-LSTM) [10]

Input Layer

The input layer of a bidirectional LSTM (Bi-LSTM) model is responsible for receiving the input data and passing it to the subsequent layers for further processing. The input layer is composed of a sequence of neurons, each of which corresponds to one element of the input sequence.The number of neurons in the input layer would correspond to the length of the input sequence, and the dimensionality of the input data, such as the number of features, would determine the size of each neuron in the input layer.

Bi LSTM Layer

The bidirectional LSTM (Bi-LSTM) layer in the Bi-LSTM model has two parts as shown in Fig 3, the forward state and the backward state. Each sequence of inputs is presented to both the forward and backward states, which allows the model to capture information from both directions and hence better capture the context and dependencies within the sequence. The forward state processes the sequence from the beginning to the end, while the backward state processes the sequence from the end to the beginning. The outputs from both states are concatenated to produce the final output of the Bi-LSTM layer.

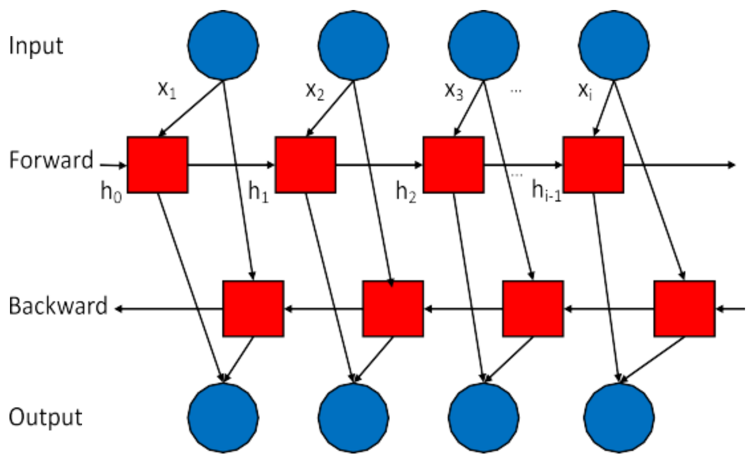


Fig 3: Structure of Bi-LSTM Layer [11]

Each memory cell has three gates to protect and control its state as shown in Fig 4, including the input gate, forget gate and output gate. The input gate determines which cell state should be updated. The forget gate decides what information should be overlooked. The output gate resolves which part of the cell state will be exported.

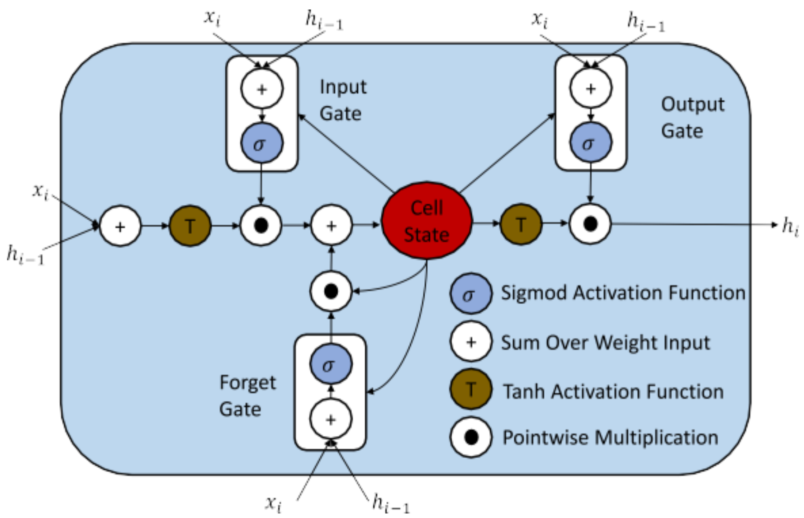


Fig 4: Structure of memory cell in Bi-LSTM [12]

Output Layer

In the output layer, from an input sequence {x1, x2 ... xm}, the memory cells in the Bi-LSTM layers will produce a representation sequence {h1, h2 ... hm}. We define the mean of these outputs as the mean pooling.

Logistic Regression Layer

The output of the Bi-LSTM layer is fed to the logistic regression layer, which uses a logistic regression function to classify the input sequence as either failed or finished. Setting up a threshold for the probability value of failure is a common approach in binary classification problems. The threshold is a value between 0 and 1 that determines the level of confidence required for the model to classify an instance as belonging to a particular class.

3.3.2 Bi-LSTM Failure Prediction Algorithm

Input: Data features of each time point for each task and job

Output: Termination statuses of the tasks and jobs

1. select Bi-LSTM prediction model parameters
2. foreach *job* do
3. for each task to do in job
4. extract task features into vectors in the order of time series take the vectors as input to the Bi-LSTM prediction model predict the task termination status
5. end
6. calculate the number of failed tasks n
7. if $n > 0$ then
8. termination status of job is failed
9. else
10. termination status of job is finished
11. end

Chapter 4

Experimental Setup and Results Analysis

4.1 Experiment Setting

We began by installing pycharm in our systems. Models were implemented on pycharm. Various libraries such as scikit learn, pandas and tensorflow were used to implement the model. Notebooks were created on google colab.
The experiments are deployed in our laptops. The dataset was stored in a csv file. We choose 280896 tasks in total for our experiments and divide them into training (80%) and testing (20%). The batch size is set as 32 which means we use 32 tasks for one training step. Number of training epochs was set to 50.

4.2 Performance Metrics

To illustrate the performance of our method, we use accuracy as the metrics to determine the results.
In a confusion matrix, accuracy is defined as the ratio of correctly classified instances to the total number of instances. Mathematically, it can be expressed as:

$$\text{Accuracy} = (TP + TN) / (TP+TN+FP+FN)$$

where true positives (TP) are the number of instances correctly predicted as positive.
true negatives (TN) are the number of instances correctly predicted as negative.
false positives (FP) are the number of instances predicted as positive but actually negative.
false negatives (FN) are the number of instances predicted as negative but actually positive.

4.3 Experimental Results

We classify the termination statuses of task submissions based on the attributes and performance data. In all the target classes, the status finish is considered as one class, and the status failed is considered as other class.

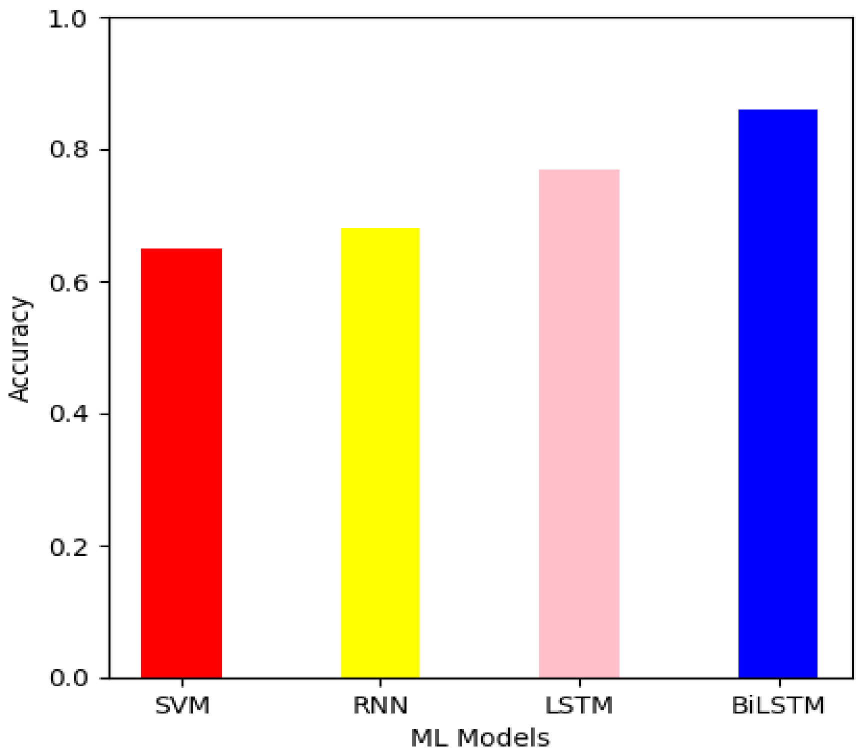
1. We took only 1000 tasks and accuracy of all the models were checked and results were obtained.

   SVM: 65%

   RNN: 68%

   LSTM: 77%

   Bi-LSTM: 86%



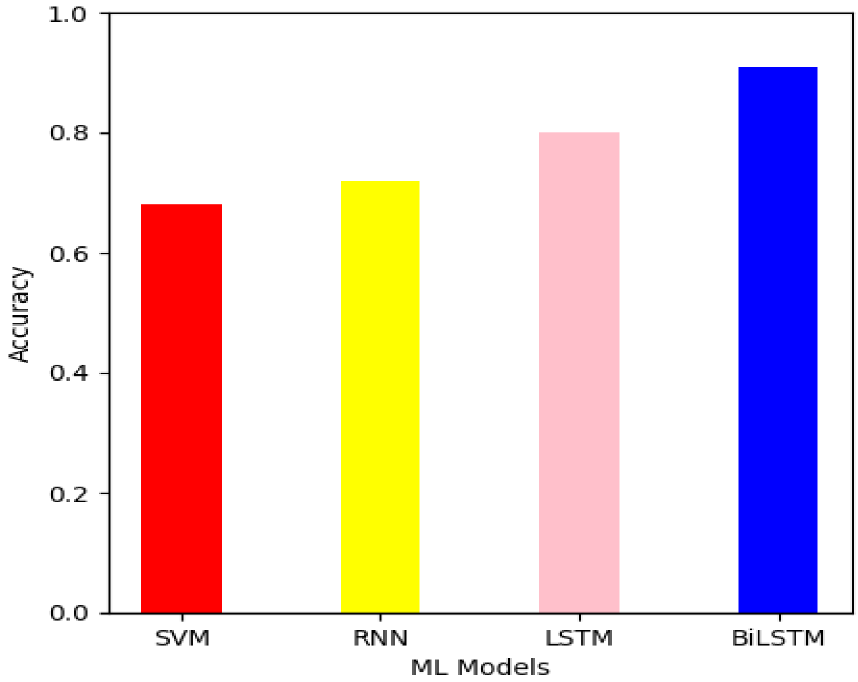
2. We took 20000 tasks and accuracy of all the models were checked and results were obtained.

   SVM: 68%

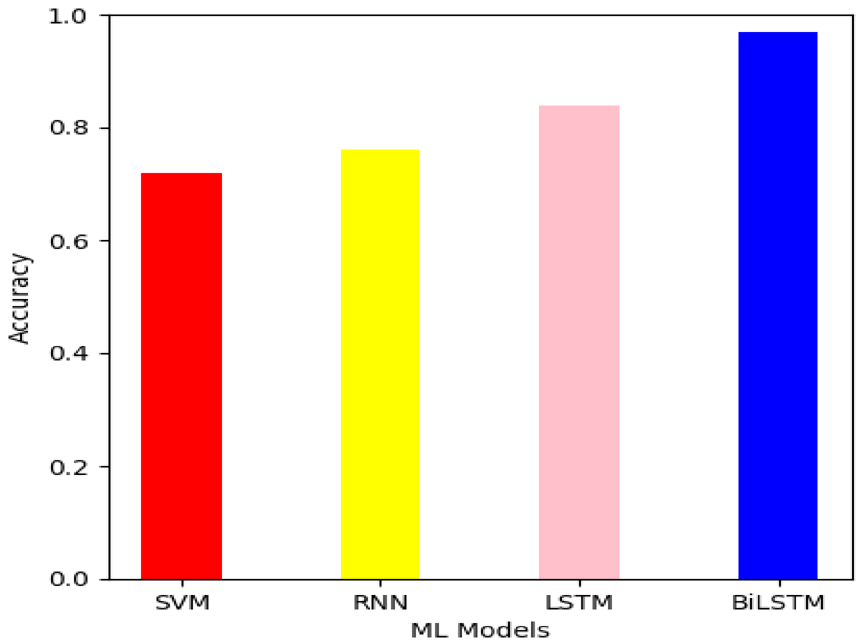
   RNN: 72%

   LSTM: 80%

   Bi-LSTM: 91%



3. We 280896 (entire dataset) tasks and accuracy of all the models were checked and results were obtained.
   SVM: 72%
   RNN: 76%
   LSTM: 84%
   Bi-LSTM: 97%



4.4 Results Analysis

We classify the termination statuses of task submissions based on the attributes and performance data. In all the target classes, the status finish is considered as one class, and the status failed is considered as other class.

The result follows SVM < RNN < LSTM < Bi-LSTM. SVM model shows the least accuracy while Bi-LSTM model shows the highest accuracy

.

Performance of SVM is worse than other models because SVM only has better performance when the dataset is not so big. The reason for not so high accuracy of RNN is the data with long term dependancies. LSTM cannot adjust the weight of further data point for a given time in the time series dataset. For Bi-LSTM, it has the forward and backward states which can more accurately determine the weights of data items that are closer and further to the given time in the prediction. Hence, Bi-LSTM shows highest accuracy.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

Bi-LSTM has shown better performance in predicting task and job failures compared to previous methods. It can capture the long-term dependencies in the time series data and also considers the data in both forward and backward directions, which helps to improve the accuracy of the predictions.

We have compared Bi-LSTM with other comparison methods including statistical, machine learning and deep learning based methods and evaluate the performance.

The results show that we achieved 97% percent accuracy in task failure prediction .

5.2 Future Work

In the future work, we will first try to implement our methods in real data center to examine the real-world performance and then work on how to use the failure prediction results from Bi-LSTM to build a failure recovery system in data center which can further improve the fault tolerance performance. We will also implement a job scheduling algorithm that can migrate the tasks and jobs to a suitable machine. The scheduler will be scalable and have the capability of dynamically adjusting its scheduling decisions based on the Bi-LSTM prediction results.

References

[1] Overcoming the cause of data center outrages, 2019. Accessed: Apr. 2019. [Online]. Available: https://www.365datacenters. com/portfolio-items/overcoming-causes-data-center-outages/

[2] 2015. Accessed: Apr. 2019. [Online]. Available: https:// techcrunch.com/2015/01/02/following-bing-coms-brief-outage search-yah oo-com-goes-down-too/

[3] D. Ford et al., “Availability in globally distributed storage systems,” in Proc. USENIX Symp. Operating Syst. Des. Implementation, 2010, pp. 61–74.

[4] Y. Zhao, X. Liu, S. Gan, and W. Zheng, “Predicting disk failures with HMM-and HSMM-based approaches,” in Proc. Ind. Conf. Data Mining, 2010, pp. 390–404.

[5] T. Pitakrat, D. Okanovic, A. van Hoorn, and L. Grunske, “Hora: Architecture-aware online failure prediction,” J. Syst. Softw., vol. 137, pp. 669–685, 2018.

[6] S. Zhang et al., “PreFix: Switch failure prediction in datacenter networks,” Proc. ACM Meas. Anal. Comput. Syst., vol. 2, 2018, Art. no. 2.

[7] A. Das, F. Mueller, C. Siegel, and A. Vishnu, “Desh: Deep learning for system health prediction of lead times to failure in HPC,” in Proc. 27th Int. Symp. High-Perform. Parallel Distrib. Comput., 2018, pp. 40–51.

[8] M. Du, F. Li, G. Zheng, and V. Srikumar, “DeepLog: Anomaly detection and diagnosis from system logs through deep learning,” in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., 2017, pp. 1285–1298.

[9] C. Xu, G. Wang, X. Liu, D. Guo, and T. Liu, “Health status assess ment and failure prediction for hard drives with recurrent neural networks,” IEEE Trans. Comput., vol. 65, no. 11, pp. 3502–3508, Nov. 2016

[10] Y. Cheng, H. Zhu, J. Wu, and X. Shao, “Machine health monitoring using adaptive kernel spectral clustering and deep long short term memory recurrent neural networks,” IEEE Trans. Ind. Informat., vol. 15, no. 2, pp. 987–997, Feb. 2019.

[11] P. Zhou et al., “Attention-based bidirectional long short-term memory networks for relation classification,” in Proc. 54th Annu. Meeting Assoc. Comput. Linguistics, 2016, pp. 207–212.

[12] Z. Huang, W. Xu, and K Yu, “Bidirectional LSTM-CRF models for sequence tagging,” 2015, arXiv:1508.01991.