AI for Predictive Maintenance

Duration of the project: 4 months

Project Mentors: Hari Chetan Kurapati, Prasanna PM

Team Members: Soumya Sangam Jha, Tanushree R Ranjanagi, Violina Doley

Introduction

Project statement: Develop a deep learning model utilising LSTM network architecture and implement an autoencoder that can identify anomalies for the purpose of predictive maintenance.

Inspiration: Machine failure causes substantial financial losses globally each year, leading to organisations across various industrial sectors to revamp maintenance processes to enhance efficiency and reduce costs. Artificial intelligence (AI) and machine learning have emerged as potential solutions for predictive maintenance by processing vast amounts of sensor data to identify potential equipment failures before they happen.

The demand for data storage has resulted in the proliferation of data centres, with hard disk drives being the most common data storage device that underpins the functioning of data centres and other data-reliant institutions. A hard disk drive failure would have severe implications for businesses, with an estimated average loss of approximately $70,000 per hour for mid-size companies. Therefore, predicting such failures is crucial to mitigate potential losses.

Background info: To achieve the goal of developing a predictive maintenance solution, we have utilised two deep learning techniques: Long Short-Term Memory (LSTM) and autoencoders.

LSTM is a type of recurrent neural network (RNN) that can process sequential data such as time series, making it ideal for predicting hard disk drive failures as disk failures occur over time. Autoencoders are neural networks that can learn to compress and reconstruct data. We have used them to extract relevant features from the sensor data, allowing us to reduce the dimensionality of the input data and identify important patterns and hence, detect anomalies.

We have used a dataset from Backblaze, a cloud backup and storage provider. Specifically, we utilised data from 2021 and 2022 that contained information on the make and model of hard drive disks, as well as various SMART attributes that provide information on the health and performance of the hard drives.

The project has shown promising results in detecting potential disk failures before they occur, allowing organisations to take proactive measures to avoid costly downtime and data loss. By using deep learning techniques such as LSTM and autoencoders, the models can process huge amounts of sensor data and detect equipment failure before it happens, thereby improving efficiency and minimising costs.

Methodology

The following methodology was used to predict failures using the LSTM model.

ST4000DM000 was chosen as the hard disk model as it experienced the most failure. The dataset was filtered to contain the records of only this model. The data was then preprocessed by sorting it by serial number and date, filling in missing data using the fill-forward technique, and normalising values using the StandardScaler() method from the preprocessing module.

The training dataset consisted of the entire backblaze dataset for 2022, while the testing dataset was the 2021 Quarter 1 dataset. To prepare the data for the LSTM model, a sequence length of 5 days and a lookahead period of 1 day were selected. The data was then formatted into sequences of the selected length, with the target variable being the next value in the sequence.

The LSTM model architecture was designed with 3 stacked LSTM layers, followed by 3 dense layers and a dropout layer to prevent overfitting. The sequential model was used to define the architecture. Hyperparameter tuning was performed to optimise the model's hyperparameters, such as the number of LSTM units, dense units, batch size, and learning rate. The LSTM model was trained on the prepared training data using the binary_crossentropy loss function and Adam optimiser with balanced class weights. Early stopping and model checkpoints were applied to avoid overfitting and save the best model. The trained LSTM model was evaluated on the testing set using metrics such as accuracy and recall.

The autoencoder model used in this study was trained on the Backblaze dataset of 2022 Quarter 4. The data preprocessing for autoencoders was similar to LSTM, involving model selection, sorting, normalizing, and filling missing data. The dataframe was sequenced with a sequence length of 16 days. Since unsupervised learning was involved, there was no y label. The model attempted to reconstruct the input data by selecting the best features. Only non-failure sequences were used for training, so whenever the model encountered a failure sequence, it would treat it as an 'anomaly.'

The model architecture consisted mainly of an encoder and a decoder. The encoder layer comprised a LSTM layer with appropriate dropout and recurrent dropout values, while the decoder layer reconstructed the minimized data based on important features. The first layer of the decoder part had a repeat vector, which repeated the compressed dataframe to make its size equal to the original dataframe. Two more LSTM decoder layers were added after the repeat vector to minimize the loss by modifying the weights. The inner layers used relu as the activation function, as it showed a fast growth. In the last decoder layer, the tanh activation function was used as the output had to be bounded. The loss function used was mae (Mean Absolute Error), and early stopping and model checkpoints were applied.

The testing process involved feeding the trained autoencoder model with pre-processed test data, which consisted of both failure and non-failure sequences. It is important to note that the model was not trained to reconstruct failure sequences, and as a result, it gives a considerable error for those sequences after testing. To differentiate between normal and anomalous data, a specified threshold for the error was set. If the error for a given sequence is greater than the threshold, the sequence is classified as a failure sequence (anomaly). This approach allows the model to effectively detect anomalies in the input data, which can be useful in predictive maintenance applications.

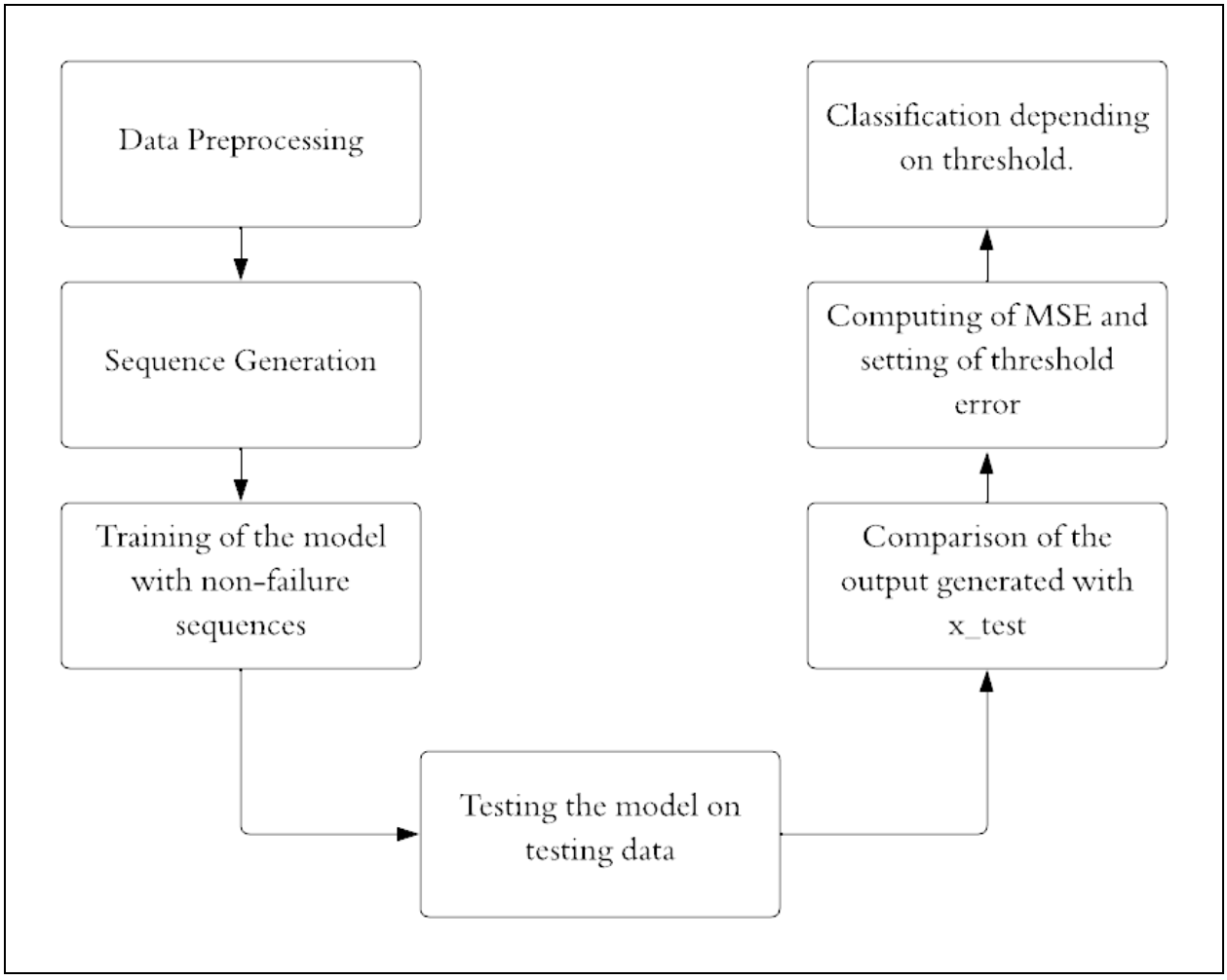

Figure 1. Flowchart depicting the methodology for training the autoencoder model for anomaly detection

The two models have been deployed on a web application that uses the Flask framework to provide an interface for users to interact with predictive maintenance models. The application allows users to select between an LSTM model and an Autoencoder model. Users can upload pre-processed test data, the LSTM model predicts failure and the autoencoder model detects anomalies. The application's backend loads the trained models and performs the predictions using the uploaded test data. The web application also displays metrics such as accuracy and recall, as well as a scatter plot comparing predicted and expected labels. Overall, the web application provides an intuitive interface for users to interact with the models and analyse the performance of the models using different metrics.

Results

The LSTM model achieved an accuracy of 80.96% and a recall of 89.09% in the evaluation phase.

```
[[1372  328]
 [   6   49]]
Accuracy: 0.8096866096866097
```

Figure 2. Confusion matrix and accuracy of LSTM model on testing data.

The results for the autoencoder model showed that out of a total of 757 test sequences, which included 158 failure sequences and 599 non-failure sequences, the model was able to classify 155 sequences as failure. These 155 sequences were treated as anomalies, as they had higher reconstruction error compared to the non-failure sequences. The overall performance of the autoencoder model suggests that it was able to detect and classify failure sequences with a high degree of accuracy.

Both the models were successfully deployed on an interface made with Flask. The interface displays performance metrics such as accuracy and recall for the LSTM model. It also includes a scatter plot that allows for a comparison between predicted and expected labels. For the autoencoders, it displays the number of normal and anomalous disks.

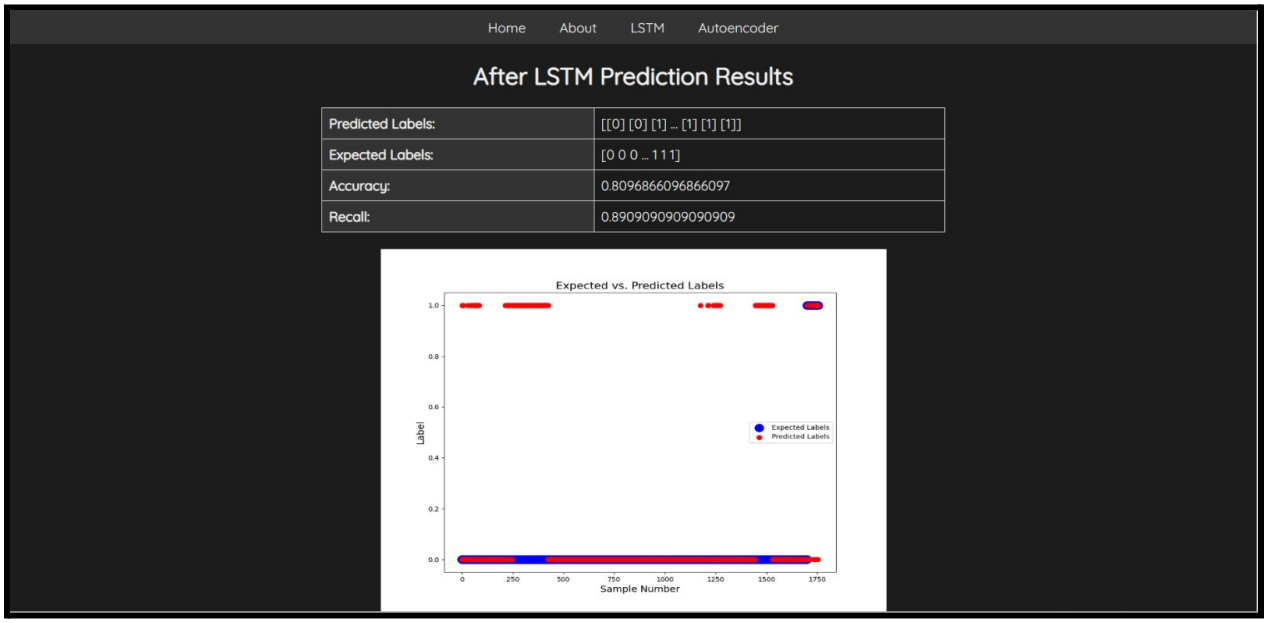

Figure 3. LSTM on the interface

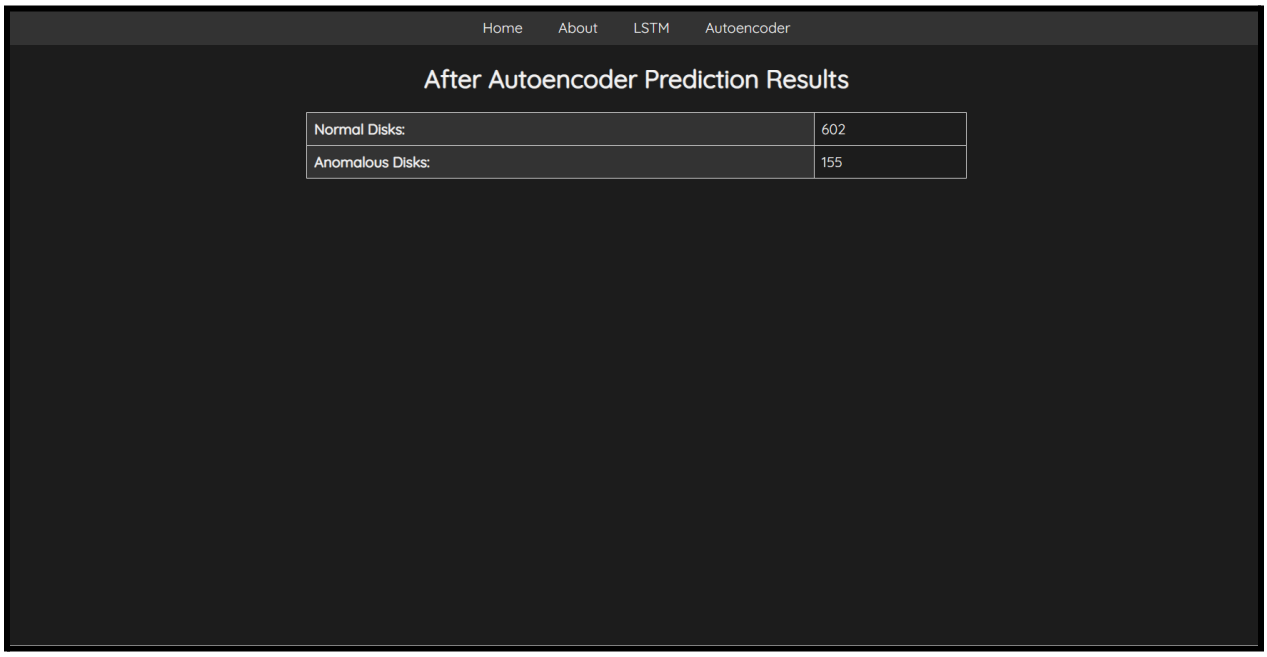

Figure 4. Autoencoders on the interface

Obstacles faced

During the course of the project, we encountered several obstacles. One of the primary challenges we faced was in merging the CSV files for each date, as the original dataset was large. We had to devise a solution to overcome this issue to proceed with the project.

Another challenge we faced was related to data bias, which was causing the model to underfit and predict most of the sequences as non-failure. This was primarily due to the high imbalance in the dataset, which made it difficult for the model to train properly. We had to address this issue by using various techniques such as oversampling, undersampling, and adjusting the class weights to balance the dataset.

Additionally, as autoencoders reconstruct the original dataframe, the classification of sequences had to be done explicitly. This led to the need for a different approach to classify the sequences based on the reconstruction error.

The encountered challenges emphasize the significance of appropriate handling of imbalanced datasets and data preprocessing in predictive maintenance models.

Conclusion

In conclusion, the LSTM and Autoencoder models developed for predicting hard disk failures have demonstrated high accuracy rates, allowing for effective predictive maintenance. The models were trained on Backblaze's dataset, pre-processed, and deployed on a web application built on Flask, allowing for ease of use for end-users. Despite challenges faced in handling imbalanced data and merging CSV files for each date, the models were able to accurately classify failure sequences, providing valuable insights for proactive maintenance planning.

Deploying the models on a Flask-based web application has enabled end-users to easily access and utilize the models, providing a user-friendly interface for predictive maintenance. With the success of these models, there is potential for further development and expansion into other industries and datasets. In conclusion, the LSTM and Autoencoder models, in combination with Flask web framework, offer a promising solution for predictive maintenance and proactive planning in the field of cloud-based services.

Future work

There are several areas of future work that can be explored to improve the performance of the predictive maintenance model. One possible approach is to investigate different variations of LSTM architecture, such as bidirectional LSTM or attention-based LSTM, to see if they can provide better accuracy for the model. These architectures may be able to capture more complex patterns in the data, which could improve the overall performance of the model.

Another area of future work is to explore different types of autoencoders, such as convolutional or variational autoencoders. These types of autoencoders may be better suited to capturing different types of temporal patterns in the data, which could improve the accuracy of the model. Additionally, extending the autoencoder model to perform multistep forecasting, where the model predicts multiple time steps into the future instead of just one, could further improve the accuracy of the model.

Another potential area of future work is to integrate more features into the interface, such as feeding the raw data as the input or entering individual sequences. This could allow for more customized predictions based on specific parameters, which could be particularly useful for companies like Backblaze who may have unique requirements for their predictive maintenance models.

Overall, there are several potential avenues for future work that could further improve the accuracy and utility of the predictive maintenance model.

References

1. Cahyadi and M. Forshaw, "Hard Disk Failure Prediction on Highly Imbalanced Data using LSTM Network," 2021 IEEE International Conference on Big Data (Big Data), Orlando, FL, USA, 2021, pp. 3985-3991, doi: 10.1109/BigData52589.2021.9671555
2. Yufei Wang, Xiaoshe Dong, Longxiang Wang, Weiduo Chen, and Xingjun Zhang. 2022. Optimizing Small-Sample Disk Fault Detection Based on LSTM-GAN Model. ACM Trans. Archit. Code Optim. 19, 1, Article 13 (March 2022), 24 pages. https://doi.org/10.1145/3500917
3. https://flask.palletsprojects.com/en/2.2.x/
4. https://colah.github.io/posts/2015-08-Understanding-LSTMs/
5. https://www.analyticsvidhya.com/blog/2022/01/complete-guide-to-anomaly-detection-with-autoencoders-using-tensorflow/