

*Industrial Training and Defence(ITD) Report*

Bachelor of Technology

*in*

## **Electronics and Communication Engineering**

### **Anomaly Detection in Stock Price Time-Series Using an LSTM Autoencoder Project**

Supervised by

**Prof. Siddhartha S. Borkotoky**

**Name: Gorthy Hari Krishna Sai  
Roll No: 22EC01011**



SCHOOL OF ELECTRICAL AND COMPUTER SCIENCES

INDIAN INSTITUTE OF TECHNOLOGY BHUBANESWAR

# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Overview</b>	<b>3</b>
2.1	Auto Encoders . . . . .	3
2.2	Long Short-Term Memory (LSTM) . . . . .	3
2.3	LSTM Autoencoder . . . . .	4
<b>3</b>	<b>Implementation</b>	<b>6</b>
3.1	Dataset and Preprocessing . . . . .	6
3.1.1	Dataset . . . . .	6
3.1.2	Data Exploration . . . . .	6
3.1.3	Preprocessing . . . . .	6
3.2	Model Architecture . . . . .	7
3.2.1	Encoder . . . . .	7
3.2.2	Decoder . . . . .	7
3.2.3	Autoencoder Objective . . . . .	7
3.3	Training Procedure . . . . .	8
3.3.1	Hyperparameters . . . . .	8
3.3.2	Training Monitoring . . . . .	8
3.3.3	Training Summary . . . . .	8
3.4	Evaluation Methodology . . . . .	8
3.4.1	Anomaly Score . . . . .	8
3.4.2	Threshold Selection . . . . .	8
3.4.3	Evaluation Metrics . . . . .	9
<b>4</b>	<b>Results and Analysis</b>	<b>10</b>
4.1	Results . . . . .	10
4.2	Discussion . . . . .	13
4.3	Future Work . . . . .	14
4.4	Conclusion . . . . .	14
4.5	References . . . . .	14

# 1. Introduction

## Abstract

Financial time-series such as stock prices show non-linear and fluctuating patterns influenced by several market and economic factors. Detecting unusual price movements is useful in identifying market instability and supporting trading decisions. In this work, an LSTM-based Autoencoder is trained on normal closing price sequences of Google (GOOG) stock. The model learns typical temporal behavior and reconstructs it. When abnormal variations occur, the reconstruction error increases, which is used to mark anomalies. This method is fully unsupervised and does not require labeled anomaly examples, making it practical for real-time market monitoring.

## Introduction

Stock price variations depend on investor activity, global events, and market conditions, leading to complex behavior that traditional statistical models often fail to capture. LSTM networks are effective for modeling sequential data due to their ability to preserve historical dependencies. When combined with an autoencoder framework, the model learns normal patterns during training and identifies deviations based on reconstruction quality. This makes the LSTM autoencoder suitable for anomaly detection in time series of stock prices, where abnormal changes may indicate volatility spikes or unusual trading activity.

## Objectives

The primary objectives of this project are:

1. Collect and preprocess the GOOGLE stock price dataset for time-series analysis.
2. To design and train an LSTM autoencoder capable of modeling normal closing price patterns.
3. Compute reconstruction error and define a threshold to identify anomalous price movements.
4. Visualise detected anomalies along the price timeline for interpretability.
5. Evaluate the effectiveness of the LSTM autoencoder in real financial anomaly detection scenarios.

# 2. Overview

## 2.1 Auto Encoders

Autoencoders are a class of neural network architectures designed for unsupervised representation learning. They work by encoding the input data into a lower-dimensional latent space and then reconstructing the original data from this compressed representation. The goal of an autoencoder is to learn meaningful internal features that capture the essential patterns present in the data.

An autoencoder consists of two main components:

- **Encoder:** The encoder compresses the input data into a compact latent vector. This process forces the network to extract the most relevant and informative patterns from the input.
- **Decoder:** The decoder attempts to reconstruct the input data from the latent representation. The reconstruction quality reflects how well the model has learned the underlying structure of the data.

The model is trained by minimizing reconstruction loss, typically Mean Squared Error (MSE):

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^N (x_i - \hat{x}_i)^2$$

where  $x_i$  is the input and  $\hat{x}_i$  is the reconstructed output.

Autoencoders are widely used for anomaly detection because they learn only the normal patterns during training. When abnormal or unseen patterns occur, the reconstruction error increases, which can be used to detect anomalies.

## 2.2 Long Short-Term Memory (LSTM)

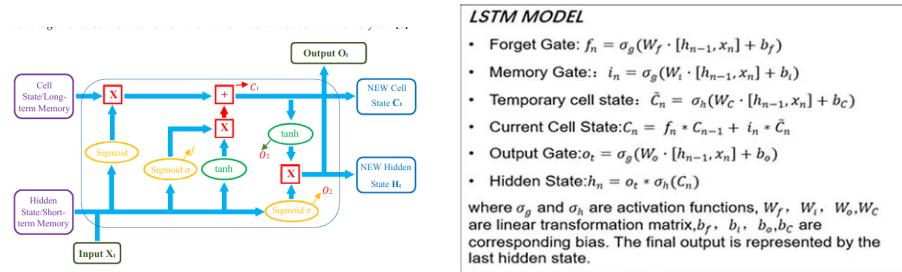
LSTM is a specialized type of Recurrent Neural Network (RNN) designed to handle sequential or time-dependent data. Standard RNNs struggle with long-term dependencies due to vanishing and exploding gradients. LSTMs address this issue by introducing memory cells and gating mechanisms that control information flow across time steps.

An LSTM cell is composed of:

- **Forget Gate:** Determines which information should be discarded.

- **Input Gate:** Decides what new information should be stored.
- **Cell State:** The memory that carries important information across long sequences.
- **Output Gate:** Controls the final output at each time step.

The key advantage of LSTM is its ability to remember long-term patterns in data such as trends, cycles, or temporal dependencies. This makes it highly suitable for financial time-series analysis, speech processing, and natural language modeling.



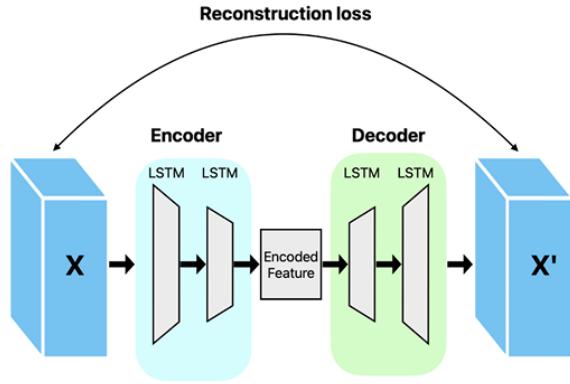
## 2.3 LSTM Autoencoder

An LSTM Autoencoder combines the strengths of autoencoders and LSTMs to learn compressed representations of sequential data. The encoder is implemented using LSTM layers that compress the input time-series into a fixed-length latent representation. The decoder, also built using LSTM layers, reconstructs the original series from this latent vector.

The architecture typically follows these steps:

- The input time-series is passed through the **LSTM Encoder**, which captures temporal patterns and encodes them.
- The resulting latent vector is repeated to match the time dimension using a **RepeatVector** layer.
- A **LSTM Decoder** reconstructs the original time-series sequence.
- A **TimeDistributed(Dense(1))** layer produces the final output for each time step.

### Why LSTM Autoencoder for Anomaly Detection?



- The model learns only the normal behavior present in the training data.
- When abnormal or unexpected patterns occur, the reconstruction error increases significantly.
- A threshold on reconstruction error can be used to classify anomalies.

Thus, LSTM Autoencoders are highly effective for anomaly detection in financial time-series where capturing temporal trends and deviations is essential.

# 3. Implementation

## 3.1 Dataset and Preprocessing

### 3.1.1 Dataset

The dataset used for this work consists of historical stock price records of Google (GOOG), downloaded from **Yahoo Finance**. The dataset contains the following columns: Date, Open, High, Low, Close, Adjusted Close, and Volume. For the purpose of anomaly detection, only the **Close** price was extracted and used in the modeling process, since it reflects the end-of-day market valuation and contains critical information about trend and volatility.

### 3.1.2 Data Exploration

The time span of the closing price data was visualized to understand the long-term market trend and fluctuations. The dataset was checked for missing values, and no inconsistencies were found. Basic descriptive statistics were analyzed to observe seasonality and volatility behavior.

### 3.1.3 Preprocessing

The following preprocessing steps were carried out:

1. The Date column was converted to *datetime* format and set as the index for time-series operations.
2. Only the Close price sequence was selected for further processing.
3. The data was normalized using the **StandardScaler** to stabilize variance and improve model training:

$$x_{scaled} = \frac{x - \mu}{\sigma}$$

4. The dataset was split into training and testing sets in an 80:20 ratio.
5. Sliding window segmentation was applied to convert the 1D sequence into overlapping sequences of fixed length:

$$\text{TIME\_STEPS} = 30$$

This segmentation enabled the model to learn temporal dependencies across multiple time intervals rather than predicting individual values independently.

## 3.2 Model Architecture

### 3.2.1 Encoder

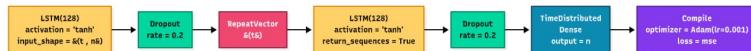
The encoder consists of an LSTM layer with 128 units and `tanh` activation, followed by a dropout layer with dropout probability of 0.2. The encoder compresses the input time sequence into a compact latent state representation:

$$\text{Encoder: } \text{LSTM}(128) \rightarrow \text{Dropout}(0.2)$$

### 3.2.2 Decoder

The decoder reconstructs the original sequence from the encoded latent representation. First, the latent vector is repeated to match the time dimension using a `RepeatVector` layer. Then, another LSTM layer reconstructs the sequence, followed by a `TimeDistributed(Dense(1))` layer to output one value per time step:

$$\text{Decoder: } \text{RepeatVector} \rightarrow \text{LSTM}(128) \rightarrow \text{Dropout}(0.2) \rightarrow \text{TimeDistributed(Dense(1))}$$



### 3.2.3 Autoencoder Objective

The model is trained using Mean Squared Error (MSE) loss, which measures the difference between the original input sequence and its reconstructed output:

$$\mathcal{L} = \frac{1}{T} \sum_{t=1}^T (x_t - \hat{x}_t)^2$$

### 3.3 Training Procedure

#### 3.3.1 Hyperparameters

Parameter	Value
Optimizer	Adam (learning rate = 0.001)
Loss Function	Mean Squared Error (MSE)
Batch Size	32
Epochs	100(Early stop at 12)
Sequence Length	30

#### 3.3.2 Training Monitoring

Training and validation losses were monitored across epochs to ensure convergence and avoid overfitting. The loss curves indicated good learning behavior, confirming that the model learned the underlying temporal structure of normal stock price movement.

#### 3.3.3 Training Summary

The trained autoencoder successfully reconstructed normal price patterns with low reconstruction error, demonstrating effective capture of temporal dependencies.

### 3.4 Evaluation Methodology

#### 3.4.1 Anomaly Score

Anomaly detection is based on the reconstruction error computed for each time window:

$$\text{Reconstruction Error (MAE)} \text{ for both train and test sets}(X) = \frac{1}{T} \sum_t (x_t - \hat{x}_t)$$

#### 3.4.2 Threshold Selection

A statistical threshold was chosen using the mean and standard deviation of reconstruction errors from the training set:

$$\theta = \mu_E + 2\sigma_E = (0.95 \text{ of samples}) \text{reconstruction error in training}$$

Any sequence whose reconstruction error exceeds this threshold is labeled as anomalous.

### 3.4.3 Evaluation Metrics

The evaluation involved:

- The steady decrease of train and validation loss.
- Visual comparison of reconstruction errors against the threshold.
- Highlighting time periods in the price series where anomalies were detected.
- The no of Anomalies depend on the threshold as we take 95th percentile of the samples reconstruction losses. Which also depend on how the reconstruction is going on.

These visual assessments confirmed that detected anomalies corresponded to periods of high volatility and abrupt price shifts.

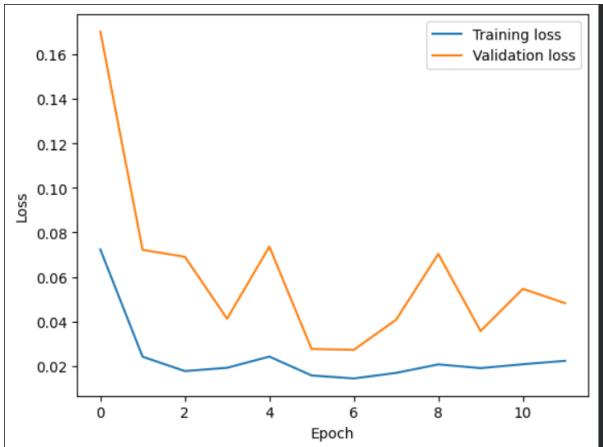
# 4. Results and Analysis

## 4.1 Results

The performance of the LSTM Autoencoder was evaluated using both training and testing datasets. The goal of the model was to learn the underlying structure of normal stock price sequences and identify significant deviations based on reconstruction error.

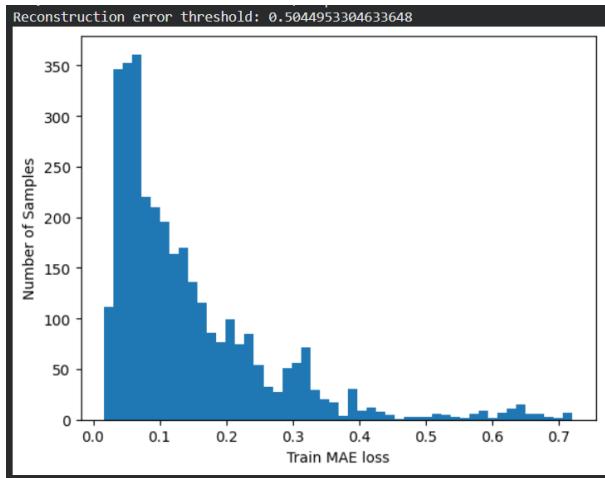
### Training and Validation Loss

During training, both the training and validation loss decreased steadily over the epochs, indicating that the model successfully learned to reconstruct normal patterns without overfitting but there was a significant surge in losses after 12 epochs, so we used early stopping and did callback.history for good weights. we got the training loss of 0.0167 and validation loss of 0.0389. The smooth convergence of the loss curves confirmed that the network parameters were updated consistently and the model generalised well to unseen data segments.



### Reconstruction Error on Training Data

The reconstruction error on the training data remained low and stable, showing that the autoencoder effectively captured the normal behavior of the GOOG closing price time-series. The distribution of training errors was used to compute the statistical threshold for anomaly detection.



The threshold was calculated as:

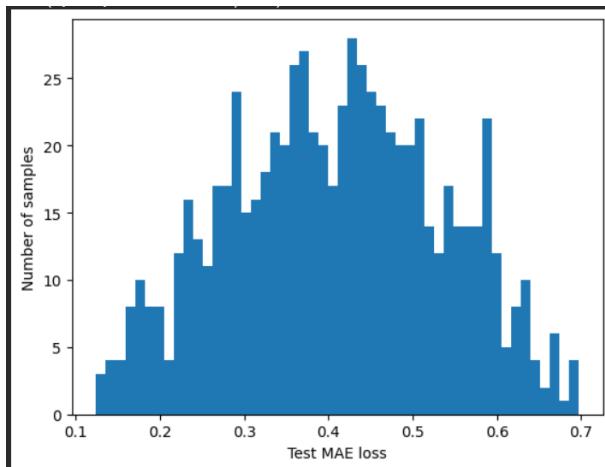
$$\theta = \mu_E + 2\sigma_E = (0.95 \text{ of samples}) \text{reconstruction error in training} = 0.5044$$

where  $\mu_E$  and  $\sigma_E$  are the mean and standard deviation of train reconstruction errors.

## Reconstruction Error on Test Data

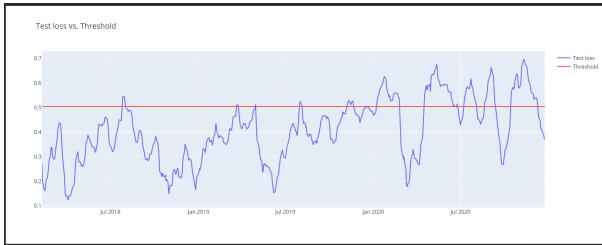
When the trained model was applied to the test dataset, the reconstruction error showed noticeable peaks during periods of unusual market activity. These peaks exceeded the threshold value  $\theta$ , indicating the presence of anomalies.

- Sequences with **smooth and stable price trends** exhibited **low reconstruction error** and Sequences associated with **sudden price jumps, drops, or volatility spikes** exhibited **high reconstruction error**.



## Reconstruction Error vs Threshold Plot

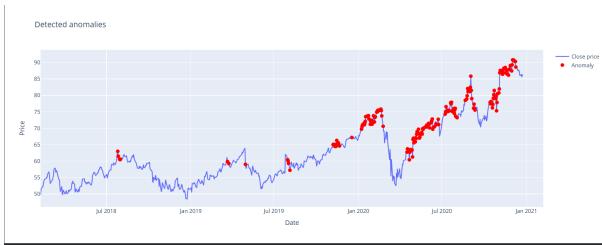
A graph plotting the reconstruction error over time against the anomaly threshold clearly showed that anomaly points correspond to instances where the error crossed the threshold boundary. This graphical comparison made anomaly detection visually intuitive and easy to interpret. We got around 165 Anomalies according to the threshold we have taken.



## Detected Anomalies in Price Series

The detected anomalies were highlighted on the original stock price plot. These points occurred primarily around:

- Periods of rapid upward or downward trend reversals,
- Sudden increases in price volatility,
- Market reaction events such as earnings announcements or macroeconomic news.
- The anomalies are very much less at the start but as we go on, the anomalies increase as the reconstruction is not that accurate at longer intervals.

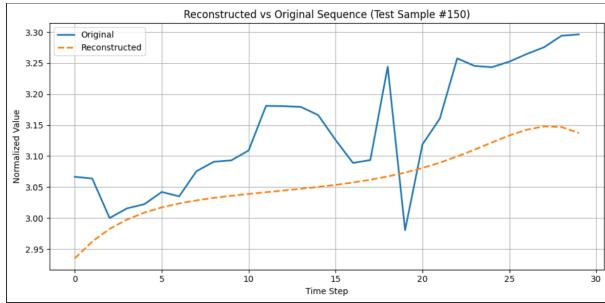


This correlation confirms that the model identified financially meaningful deviations rather than random noise.

## Reconstruction Plot

Comparisons between the original input sequences and their reconstructed outputs demonstrated that:

- For normal sequences, the model's reconstructed output closely matched the input.
- For anomalous sequences, the reconstruction deviated significantly, leading to large reconstruction error values.
- Reconstruction is not that perfect, but it is following the flow, because we did unsupervised learning without defining how anomalies are, so this is how we see the reconstruction happen.



These visualizations validate the effectiveness of the LSTM Autoencoder in distinguishing between normal and anomalous stock market behavior.

## 4.2 Discussion

The LSTM Autoencoder model effectively captured the temporal patterns associated with normal price behavior. Unlike statistical baseline models, which often assume stationarity, the LSTM architecture successfully modeled non-linear dependencies and temporal correlations.

Key observations include:

- The model does not require labeled anomaly data, making it suitable for unsupervised anomaly detection.
- The anomaly detection threshold based on the mean and standard deviation of training loss worked well but may require tuning for other financial instruments.
- Detected anomalies corresponded to real-world market events characterized by sudden price shifts or abnormal volatility.

However, the model reconstructs only one-dimensional closing price signals. Real market conditions often depend on multiple factors such as volume, high-low spread, and external economic indicators.

## 4.3 Future Work

The model can be further improved in several ways:

1. Incorporating multivariate inputs such as Open, High, Low, Volume, and technical indicators.
2. Using Attention-based LSTM for better feature emphasis and interpretability.
3. Implementing adaptive or dynamic thresholding based on market volatility.
4. Integrating real-time anomaly alerts with trading or risk management systems.
5. Applying explainability methods (e.g., SHAP) to understand detected anomalies.

## 4.4 Conclusion

This work demonstrated the application of an LSTM Autoencoder detecting anomalies in stock market time-series data using the GOOG closing price dataset. By training the model exclusively on normal behavior, reconstruction error served as an effective anomaly detection measure. The results show that the approach successfully identifies abnormal price movements associated with volatility spikes. The study highlights the strength of deep learning-based unsupervised models in understanding complex financial time-series patterns and sets the foundation for more advanced market anomaly detection systems.

## 4.5 References

- Lachekhab, F.; Benzaoui, M.; Tadjer, S.A.; Bensmaïne, A.; Hamma, H. (2024). *LSTM-Autoencoder Deep Learning Model for Anomaly Detection in Electric Motor*. Energies, 17(10), 2340. <https://doi.org/10.3390/en17102340>
- Hochreiter, S., & Schmidhuber, J. (1997). *Long Short-Term Memory*. Neural Computation.
- François Chollet, *Keras Documentation*. <https://keras.io>
- Yahoo Finance, *GOOGLE Historical Data*. <https://finance.yahoo.com>
- Homayouni et al., (2021). *IDEAL Framework for Time-Series Anomaly Detection*.