A PROJECT REPORT

on

"COMPARISION OF DEEP LEARNING MODELS FOR STOCK PRICE PREDICTIONS"

Submitted to

KIIT Deemed to be University

In Partial Fulfillment of the Requirement for the Award of

BACHELOR'S DEGREE IN
COMPUTER SCIENCE ENGINEERING

BY

AASHMIT SHRESTHA	20051712
AAYUSH RATNA STHAPIT	20051713
BISHAL PANDEY	20051723
NARESH KHATIWADA	20051737

UNDER THE GUIDANCE OF
Mr. Ajay Anand

SCHOOL OF COMPUTER ENGINEERING
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA - 751024
May 2023

KIIT Deemed to be University

School of Computer Engineering
Bhubaneswar, ODISHA 751024

CERTIFICATE

This is certify that the project entitled

“COMPARISION OF DEEP LEARNING MODELS FOR STOCK PRICE PREDICTIONS”

submitted by

AASHMIT SHRESTHA	20051712
AAYUSH RATNA STHAPIT	20051713
BISHAL PANDEY	20051723
NARESH KHATIWADA	20051737

is a record of bonafide work carried out by them, in the partial fulfillment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Engineering) at KIIT Deemed to be university, Bhubaneswar. This work is done during year 2022-2023, under our guidance.

Date: / /

Ajay Anand
(Project Guide)

Acknowledgments

We are profoundly grateful to **AJAY ANAND** sir of the **School of Computer Science Department** for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion.

We would like to express our sincere gratitude to all the individuals who have contributed to the successful completion of this minor project.

Lastly, we would like to thank our families and friends who have been a constant source of motivation and encouragement throughout our academic journey.

AASHMIT SHRESTHA
AAYUSH RATNA STHAPIT
BISHAL PANDEY
NARESH KHATIWADA

ABSTRACT

Owing to its volatility and its dependence on numerous factors predicting the future course of the stock market is difficult. Forecasts are frequently made using technical indicators and historical data to address this issue. A few examples of Deep Learning RNNs that may be utilized to take advantage of various properties to improve the prediction accuracy include LSTM and GRU.

This project aims to compare the prediction accuracy proposed the deep learning models for stock prices .Historical data as well as technical indicators such as the Moving Average make the input data. The framework proposed in this study uses GRU to identify long-term trends and make predictions.

Keywords: *LSTM, GRU, Technical Indicators, Stock Prediction, RNN*

Contents

List of Figures

Chapter 1

Introduction

The stock market is characterized by its erratic and dynamic nature, where prices may change promptly due to a range of factors, including geopolitical events, economic indicators, and corporate performance. Accurately forecasting stock prices poses a challenge for investors, traders, and financial analysts. The ability of machine learning algorithms to analyse enormous volumes of data and find subtle patterns in the provided datasets has made them increasingly popular for stock price forecasting.

Recurrent Neural Networks (RNN) are used in neural network architectures for the analysis of sequential data. LSTM is a subclass of RNN that has been well-liked in stock market forecasting due to its capacity to identify long-term dependencies in sequential data. In order to identify patterns in the data, LSTM models are trained on past stock prices and pertinent variables. Future stock prices were then predicted using these patterns. The LSTM model can recognise long-term trends and patterns in stock market data because it features a memory unit that can retain significant information for a longer period of time. LSTM models are frequently employed in the financial sector for stock market analysis since they have demonstrated promising results in earlier studies.

The dataset used in this study is the historical stock price data of a selected company, including features like   closing price,high price,opening price, low price and volumes. The dataset was preprocessed by normalizing the values and splitting them into testing and training sets.

The training set was used to train the LSTM and GRU models, while the testing set was used to evaluate them. These models are categories of Recurrent Neural Networks (RNN). They posses the ability to capture long-term dependencies in sequential data. The LSTM model has memory units that can retain significant information for a longer period of time. The GRU model has lesser parameters in comparison to the LSTM model, making it faster to train and execute.

Chapter 2

Basic Concepts

2.1 Deep Learning

Deep Learning is a subfield of machine learning that uses neural networks with multiple layers to learn complex representations of data. These neural networks are referred regarded as "deep" because they have numerous layers, which enable them to simulate hierarchical relationships between various data aspects.

Deep learning has proven to be incredibly effective in a variety of applications, including computer vision, natural language processing, speech recognition, and gaming, among others. Deep learning can automatically learn features from raw data without the need for manual feature engineering, which is one of its key benefits.

Numerous optimisation methods, including stochastic gradient descent (SGD), which iteratively modifies the network's weights in order to minimise a loss function, can be used to train deep learning models. Deep learning architectures can also be modified for certain tasks and data types, using various layers and activation functions to produce the best results. Powerful hardware, including graphics processing units (GPUs), and the accessibility of a lot of labelled data have helped deep learning advance quickly and become widely used in recent years.

2.2 Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNNs) are a type of deep learning. RNNs are neural networks with multiple layers that are designed to handle sequential data, such as time series data or natural language text. The multiple layers in RNNs allow them to learn hierarchical representations of the sequential data, and the feedback loops in the network allow them to maintain an internal memory of past inputs, making them particularly useful for processing sequences of variable length.

In an RNN, the input at each time step is processed by a set of interconnected neurons, which also receive input from the previous time step's output. This allows the network to capture temporal dependencies between the input data, as well as to learn patterns and relationships that extend across time.

One of the key features of RNNs is their ability to handle input sequences of variable length, as the network can be unrolled for as many time steps as needed to process the entire sequence. RNNs can be trained using various learning algorithms, such as backpropagation through time (BPTT), which adjusts the network's weights based on the errors in the predicted output compared to the desired output.

As RNNs have multiple layers and can learn complex representations of sequential data, they are considered a type of deep learning. In fact, many recent advances in natural language processing and speech recognition have been driven by the development of deep RNN architectures such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks, which can handle long-term dependencies in sequential data.

2.3 Long Short Term Memory (LSTM)

The LSTM (Long Short-Term Memory) Recurrent Neural Network architecture is often utilised for time series prediction applications. When processing time-series data with long-term dependencies, this style of architecture works very well because it gives users the option to recall or ignore prior information. A stock price prediction model employing LSTM often uses historical prices as well as relevant variables like trade volume, market indexes, and sentiment towards the news as input data. After analysing these various inputs, the LSTM model generates an output that represents the predicted stock price at a particular timestep.The LSTM architecture consists of three main component

i) **Forget gate:** In an LSTM neural network cell, the first step is to determine whether we should retain or discard information from the previous time step. The forget gate equation is as follows:

$$ft =\sigma\ (Wxf * Xt +Whf *ht-1 +Wcf *ct-1 + b_f)\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(1)$$

ii) **Input gate:** The input gate measures the weight of any fresh information that is being brought in. This is the input gate equation:

$$\iota t=\sigma\ (Wxi * Xt +Whi *ht-1 +Wci*ct-1 + b_i)\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2)$$

iii) **Output gate:** These gates regulate information flow through the network and decide whether to recall or forget previous information. LSTM also has a memory cell that stores information over time, enabling the network to capture long-term dependencies in data. The output gate equation is as follows:

$$\mathrm{O}t =\sigma\ (Wx_O * Xt +Wh_O *ht-1 +Wc_O*ct + b_O)\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3)$$

The hidden state is calculated using the following equation:

$$ht = ot \tanh(ct)\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(4)$$

Utilising the forget gate's f_t activation, the cell state is multiplied. The updated cell is multiplied by o_t after being processed by the tan h function.

$$ct = ftct-1 + it \tanh\ (WxcXt +Whcht-1 + bc)\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(5)$$

An LSTM model is often optimised using a loss function that gauges the difference between the projected and real stock prices in order to train it to forecast stock prices. The model can be trained using a variety of techniques, including back-propagation through time (BPTT) and gradient descent optimization.

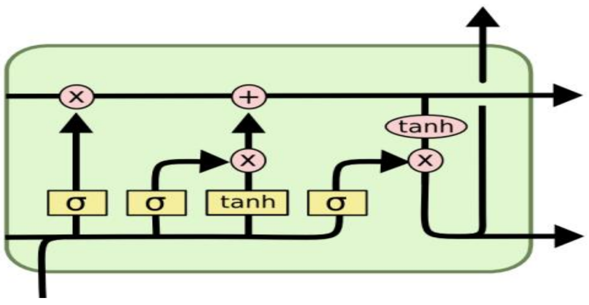


Figure 1: LSTM Architecture

2.3 Gated Recurrent Unit (GRU)

Similar to an LSTM is a Recurrent Neural Network (RNN) architecture called a Gated Recurrent Unit (GRU). The GRU was introduced as a compressed version of the LSTM that can identify long-term associations in time-series data and is computationally less expensive and easier to train.

GRU features gating mechanisms that control the information flow through the network similarly to LSTM, allowing them to select whether data from previous time steps to keep or reject. The GRU, however, has fewer gating mechanisms than the LSTM, having just two gates.

i) **Gate update:** How much of the previous data should be kept and forwarded to subsequent stages is decided by the update gate. Calculating the time step's z_t requires:

$$z_{t=}\sigma(W^{(z)}x_t+U^{(z)}h_{t-1})\ldots\ldots(9)$$

ii) **Reset gate:** The amount of previous information to be forgotten is decided by the reset gate.

$$r_{t=}\sigma(W^{(r)}x_t+U^{(r)}h_{t-1})\ldots\ldots(9)$$

The reset gate is used to compute the memory content, which stores pertinent data from earlier phases.

$$h'_t=\tanh(Wx_t+r_t+\odot\, Uh_{t-1})\ldots\ldots(10)$$

The network uses the update gate to decide which data to collect from the previous phase h'_{t-1} in order to generate the h'_t vector for the current unit and which data to gather from the current memory contents.

$$h_t=z_t\odot h_{t-1}+(1-z_t)\odot h'_t\ldots\ldots(11)$$

The framework for Gated Recurrent Unit (GRU) is given in figure 4:

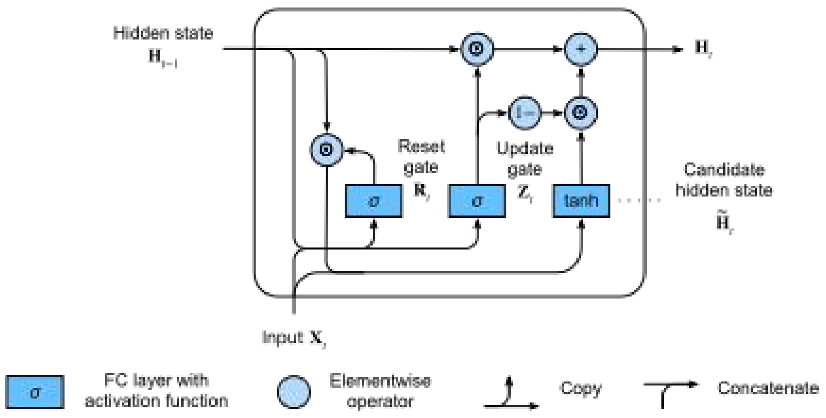


Figure 2: Gated Recurrent Unit(GRU) Architecture

Chapter 3

Problem Statement :

Predicting stock prices is an essential component of financial decision-making for traders, investors, and companies. With the development of big data and machine learning technology, Deep Learning Models have become a viable option for more accurate and reliable stock price predictions.

Deep Learning models are used to anticipate stock prices with the primary goal of analysing vast amounts of historical data to find patterns and trends that can be used to forecast future stock values. Deep Learning Models make forecasts by using sophisticated algorithms to examine a variety of data sources, such as stock prices, trade volumes, news, social media sentiments, and other market indicators.

3.1 Software Requirement Specification (SRS):

Introduction:

This Software Requirement Specification (SRS) document's goal is to outline the specifications for a piece of software that forecasts stock values using deep learning models. To forecast future stock values, the algorithm will examine historical data, including stock prices, trade volumes, news, and attitudes on social media. The system's functional and non-functional needs are outlined in this paper.

Functional Requirements:

1) **Data Collection:** The system should be able to collect a large volume of historical data from various sources, including stock prices, trading volumes, news, and social media sentiments.

2) **Data Preprocessing:** The system should be able to preprocess the collected data to remove noise and inconsistencies.

3) **Deep Learning Model:** The system should use deep learning models, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), to analyze the preprocessed data and make predictions on future stock prices.

4) **Training the Model:** The system should be able to train the deep learning model using the preprocessed historical data.

5) **Prediction:** The system should be able to use the trained model to predict future stock prices based on the input data.

6) **Data Visualization:** The system should be able to properly visualize the produced data.

7) **User Interface:** The system should have a user-friendly interface that allows users to input data and view predictions.

Non-Functional Requirements:

1) **Performance:** The system should be able to process large volumes of data and make accurate predictions in a timely manner.

2) **Security:** The system should be secure and protect the privacy of users' data.

3) **Reliability:** The system should be reliable and provide accurate predictions consistently.

4) **Scalability:** The system should be able to handle an increasing volume of data as the user base grows.

5) **Compatibility:** The system should be compatible with various operating systems and web browsers.

Assumptions and Constraints:

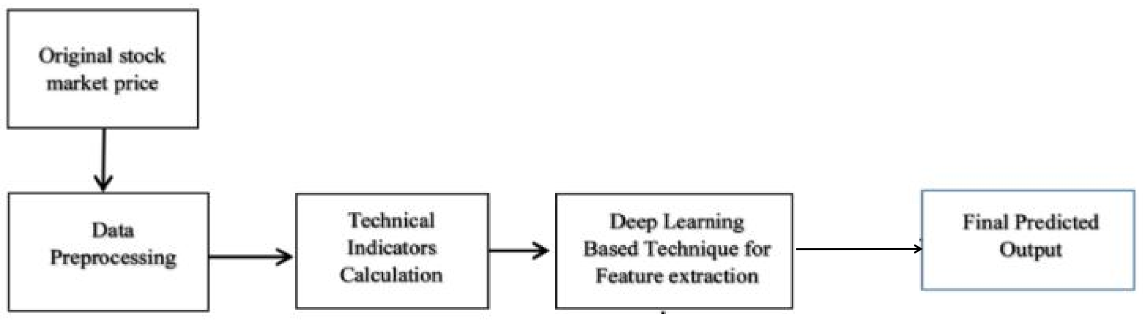
- The accuracy of the stock price prediction depends on the quality of the input data and the performance of the deep learning model.
- The system will only be as accurate as the quality of the data being used for predictions.
- The system will not provide investment advice or recommendations.

Conclusion:

The functional and non-functional needs of a software system that forecasts stock prices using deep learning models have been outlined in this software requirement specification document. Large amounts of historical data should be able to be gathered, processed, and analysed by the system in order to produce precise forecasts of future stock prices. Additionally, the system must be safe, dependable, and scalable and have an intuitive user interface. The effectiveness of the deep learning model and the quality of the input data affect how accurately stock prices are predicted.

3.2 System Architecture:

The following diagram shows the framework for our given model:


Original stock market price
Data Preprocessing
Technical Indicators Calculation
Deep Learning Based Technique for Feature extraction
Final Predicted Output

The original data is imported from the Yahoo finance website and the data is preprocessed.

Yahoo Finance was used to gather Apple stock data for 10 years, or from January 2013 to January 1, 2023. The Close, High, Low, and Opening Prices are included in the statistics. Before further training, these data were preprocessed and scaled between 0 and 1. The deep learning techniques were used to perform Day Ahead Prediction.

Chapter 4

Implementation :

The experiment was performed with an Intel Core i7 - 1065G7 CPU @ 1.30 GHz processor and 16GB RAM along with an NVIDIA GeForce MX330 GPU. The proposed model was built in Python on the Visual Studio Code IDE using TensorFlow. The required graphs were plotted using the matplotlib library. A Moving Average Technical Indicator was considered for this study. Figure 5 shows the original closing price for apples over 10 years, and Figure 6 shows the relationship between the Moving Average of closing prices over 200 and 100 days with respect to the closing price.

4.1 Methodology:

As mentioned previously, LSTM and GRU algorithms are used. The models were created using Keras supported by Tensorflow. The data visualization was done using using Matplotlib library and for developing the web application, the streamlit library was used.

I) **Keras**

Keras is a Python-based deep learning API that is built on top of the TensorFlow machine learning framework. It was developed with the intention of speeding up experimentation and providing an excellent development experience.

With a focus on modern deep learning, Keras, the high-level API of TensorFlow 2, is a user-friendly, very powerful interface for resolving machine learning problems. It provides essential foundational components and abstractions for rapidly developing and deploying machine learning solutions.

The purpose of Keras is to provide an unfair edge to any developer planning to create ML-powered applications.

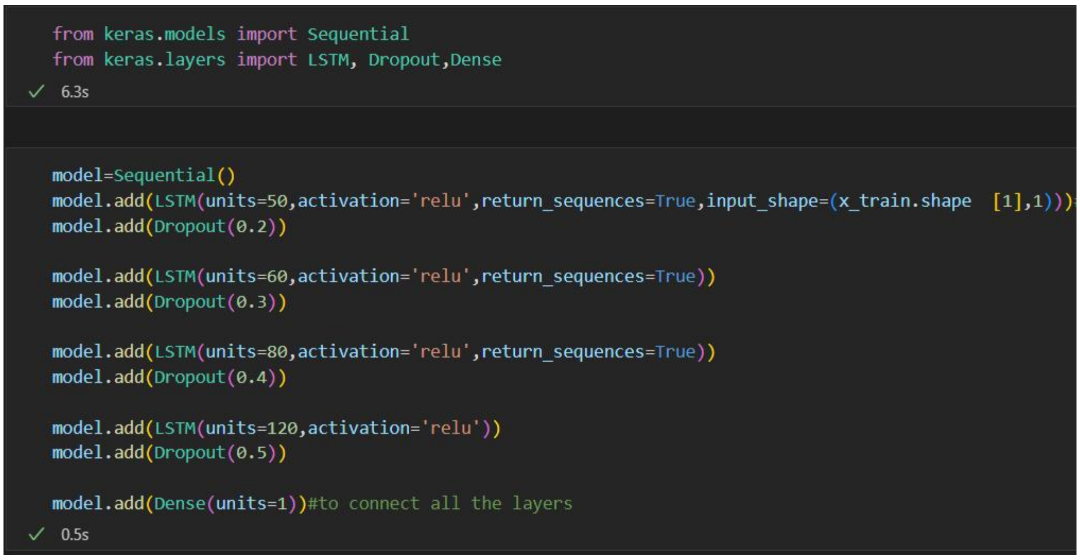
II) Matplotlib:

For 2D displays of arrays, Matplotlib is a fantastic Python visualisation package. A multi-platform data visualisation package called Matplotlib was created to operate with the larger SciPy stack and is based on NumPy arrays. In the year 2002, John Hunter first presented it. One of visualization's biggest advantages is that it gives us visual access to vast volumes of data in forms that are simple to understand. There are numerous plots in Matplotlib, including line, bar, scatter, histogram, etc.

III) Streamlit:

A free and open-source framework called Streamlit allows you to quickly create and distribute stunning machine learning and data science web apps. This Python-based library was created with machine learning engineers in mind.With Streamlit, you can create shareable web apps from data scripts in minutes as opposed to weeks.

IV) LSTM model:



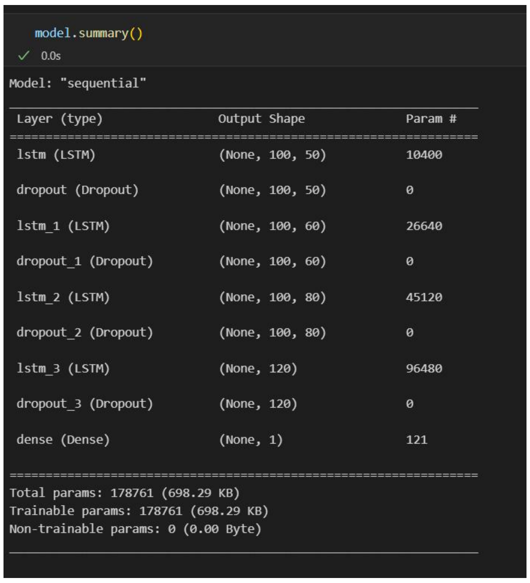
```
from keras.models import Sequential
from keras.layers import LSTM, Dropout,Dense
✓ 6.3s

model=Sequential()
model.add(LSTM(units=50,activation='relu',return_sequences=True,input_shape=(x_train.shape  [1],1)))
model.add(Dropout(0.2))

model.add(LSTM(units=60,activation='relu',return_sequences=True))
model.add(Dropout(0.3))

model.add(LSTM(units=80,activation='relu',return_sequences=True))
model.add(Dropout(0.4))

model.add(LSTM(units=120,activation='relu'))
model.add(Dropout(0.5))

model.add(Dense(units=1))#to connect all the layers
✓ 0.5s
```

Code snippet for the creation of the LSTM model.

Our LSTM model has 4 preceding layers, with 0.2,0.3,0.4 and 0.5 dropout; 50,60,80 and 120 nodes respectively, with a rectified linear unit (ReLU) as an activation function and a Dense layer.

- The trained model for the Dropout machine learning model is prone to over-fitting if the model's parameters are too numerous and the training samples are insufficient. Fitting is a common issue we run across while training neural networks. The following are signs of overfitting: On training data, the model's loss function is smaller, and prediction accuracy is higher; however, on test data, the loss function is comparatively larger, and prediction accuracy is lower.
- A rectified linear unit (ReLU) is an activation function that gives a deep learning model the ability to be non-linear and addresses the vanishing gradients problem.It interprets the conclusive aspect of its case.
- In any neural network, a layer that is densely connected to its preceding layer means that every neuron in the layer is connected to every other neuron in the layer above it.

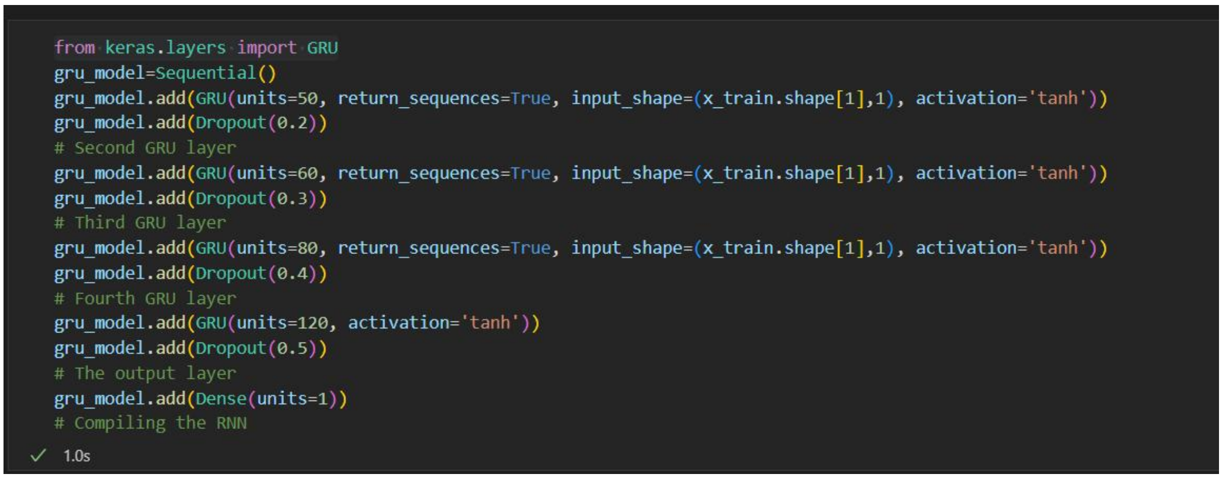
Model summary:



```
model.summary()
✓ 0.0s
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 100, 50)           10400

 dropout (Dropout)           (None, 100, 50)           0

 lstm_1 (LSTM)               (None, 100, 60)           26640

 dropout_1 (Dropout)         (None, 100, 60)           0

 lstm_2 (LSTM)               (None, 100, 80)           45120

 dropout_2 (Dropout)         (None, 100, 80)           0

 lstm_3 (LSTM)               (None, 120)               96480

 dropout_3 (Dropout)         (None, 120)               0

 dense (Dense)               (None, 1)                 121

=================================================================
Total params: 178761 (698.29 KB)
Trainable params: 178761 (698.29 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

V) GRU Model:

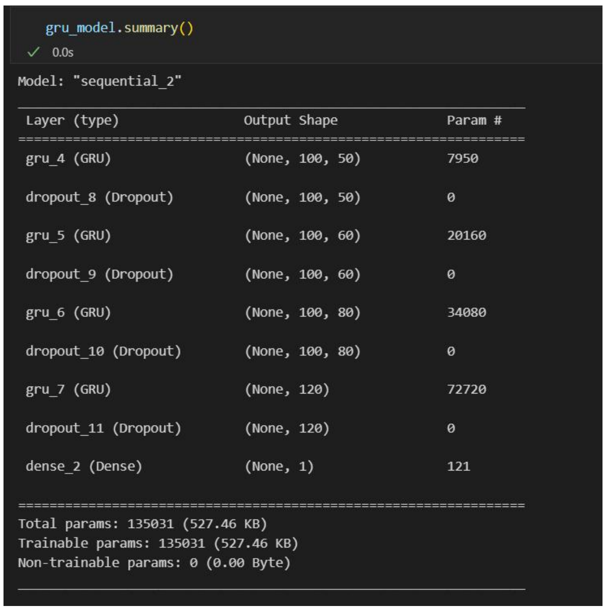
Code snippet for the creation of the GRU model:



```
from keras.layers import GRU
gru_model=Sequential()
gru_model.add(GRU(units=50, return_sequences=True, input_shape=(x_train.shape[1],1), activation='tanh'))
gru_model.add(Dropout(0.2))
# Second GRU layer
gru_model.add(GRU(units=60, return_sequences=True, input_shape=(x_train.shape[1],1), activation='tanh'))
gru_model.add(Dropout(0.3))
# Third GRU layer
gru_model.add(GRU(units=80, return_sequences=True, input_shape=(x_train.shape[1],1), activation='tanh'))
gru_model.add(Dropout(0.4))
# Fourth GRU layer
gru_model.add(GRU(units=120, activation='tanh'))
gru_model.add(Dropout(0.5))
# The output layer
gru_model.add(Dense(units=1))
# Compiling the RNN
✓ 1.0s
```

The activation function is different from the LSTM model, but the dropout and number of nodes are identical. The tanh activation function is used in this situation. The tanh function, which is frequently employed in recurrent neural networks, generates an output that is zero-centered and supports the backpropagation process.

Modal Summary:



```
gru_model.summary()
✓ 0.0s
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 gru_4 (GRU)                 (None, 100, 50)           7950

 dropout_8 (Dropout)         (None, 100, 50)           0

 gru_5 (GRU)                 (None, 100, 60)           20160

 dropout_9 (Dropout)         (None, 100, 60)           0

 gru_6 (GRU)                 (None, 100, 80)           34080

 dropout_10 (Dropout)        (None, 100, 80)           0

 gru_7 (GRU)                 (None, 120)               72720

 dropout_11 (Dropout)        (None, 120)               0

 dense_2 (Dense)             (None, 1)                 121

=================================================================
Total params: 135031 (527.46 KB)
Trainable params: 135031 (527.46 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

VI) Evaluation Metrics:

When evaluating the performance of machine learning models for stock price prediction, several metrics are commonly used to measure prediction accuracy. These include:

I) **Mean Absolute Percentage Error (MAPE):** The Mean Absolute Percentage Error (MAPE) determines the percentage difference between the expected and true figures.It's outlined as:

MAPE =(1/n) * ∑(|actual - predicted| / actual) * 100

where n is the sample size. When the magnitude of the mistakes is crucial, the MAPE is chosen for comparing the performance of several models.

II) **Mean Absolute Error (MAE):** Calculates the magnitude of the variation between expected and true values. It's outlined as:

MAE = (1/n) * ∑(|actual - predicted|)

MAE is easy to interpret and useful when the magnitude of errors is important.

III) **Root Mean Squared Error (RMSE):** RMSE gives the sum of the squared discrepancies between the expected and observed values. It's outlined as:

$$\text{RMSE} = \sqrt{(1/n) * \sum(\text{actual} - \text{predicted})^2)}$$

RMSE is commonly used and is preferred when large errors are undesirable

4.2 Testing:

The proposed model was built in Python on the Visual Studio Code IDE using TensorFlow. The required graphs were plotted using the matplotlib library.

A Moving Average Technical Indicator was considered for this project. Figure 5 shows the original closing price for apples over 10 years, and Figure 6 shows the relationship between the Moving Average of closing prices over 200 and 100 days with respect to the closing price.

The graph is non linear and has several highs and lows and thus requires a good technique for prediction.

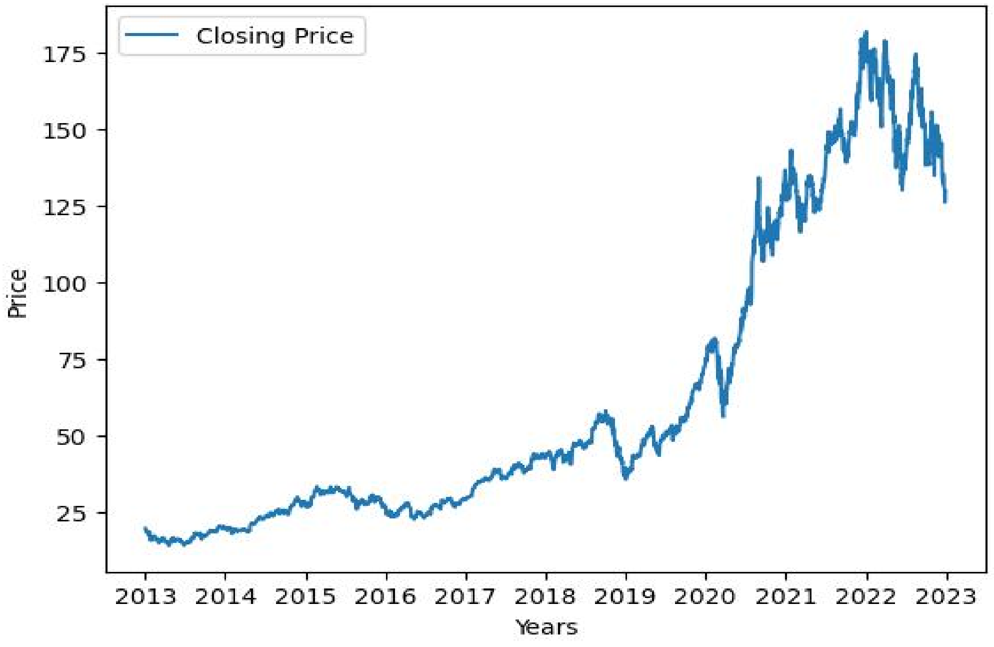


Figure 5: Closing Price of Apple (2013-2023)

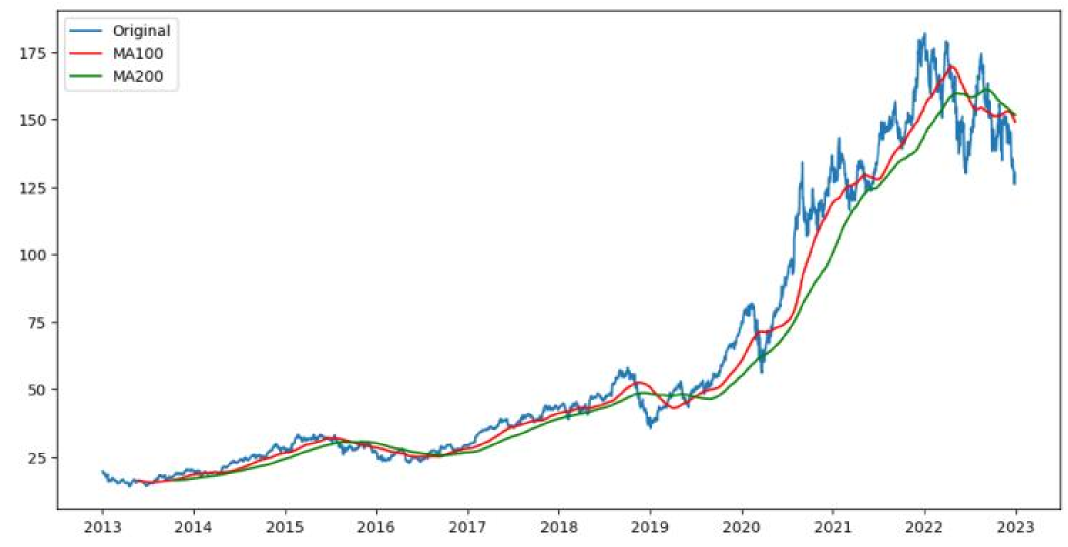


Figure 6: Moving Average(100 days) and Moving Average (200 days) with respect to the Original closing price.

The moving average for 100 days moves closely with the closing price of the stock, and was hence considered for this Project.

The dataset was split into training and testing set. 70% of the data was used for training and 30% was used for validation/testing.

The final 100 days data was used to compare the predicted results with the actual results.

4.3 Result Analysis:

The experiment was performed with an Intel Core i7 - 1065G7 CPU @ 1.30 GHz processor and 16GB RAM. The proposed model was built in Python on the Visual Studio Code IDE using TensorFlow. The required graphs were plotted using the matplotlib library.

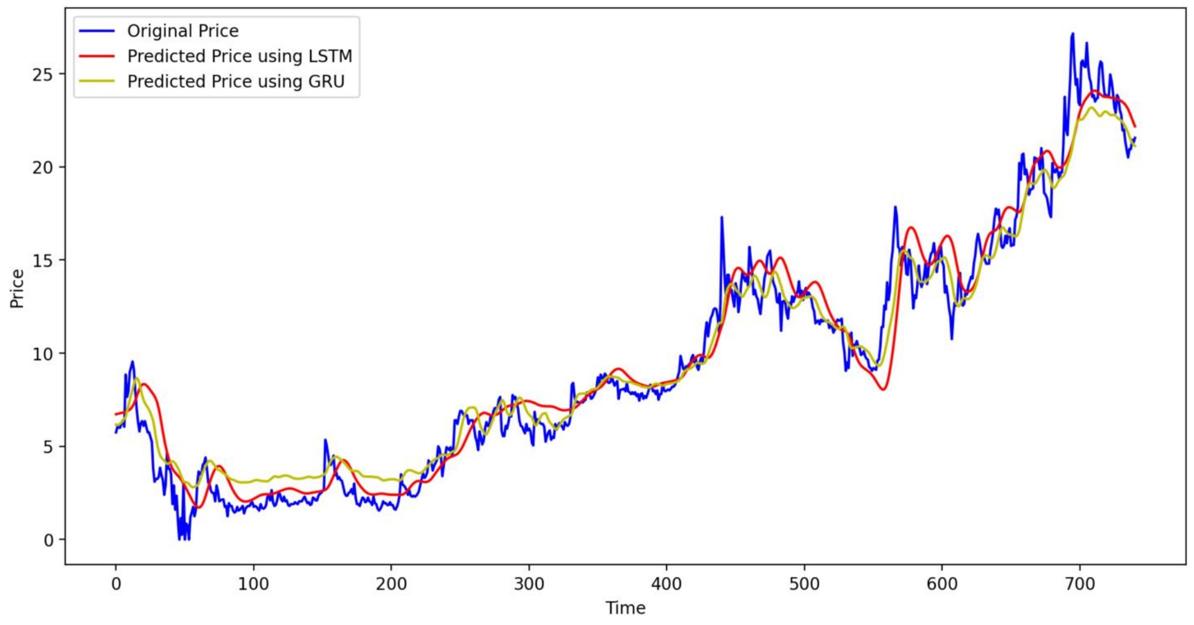


Figure 7: Assessment of the performance of the models, a comparison between the initial stock prices and the forecasts produced by the LSTM model and the GRU model is made for **AAPL** stock data

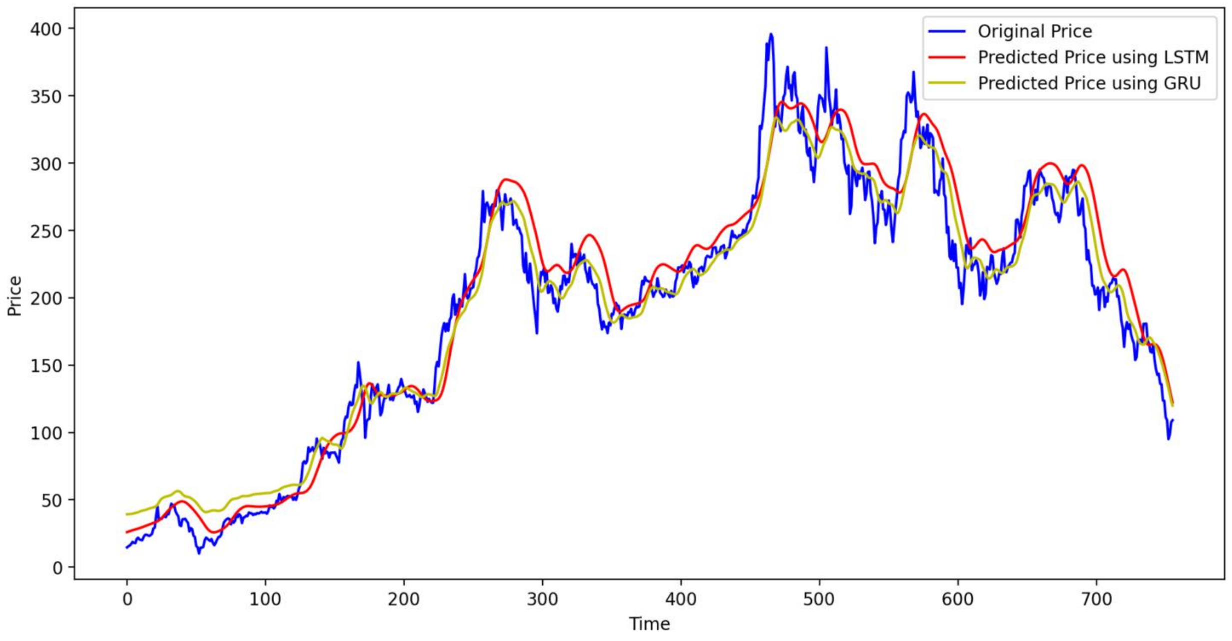


Figure 8: Assessment of the performance of the models, a comparison between the initial stock prices and the forecasts produced by the LSTM model and the GRU model is made for **TSLA** stock data

Upon observation, it can be noted that the GRU model exhibits greater accuracy and proximity to the closing price in comparison to the LSTM model.

When evaluating the performance of machine learning models for stock price prediction, several metrics are commonly used to measure prediction accuracy. Thes following tables show the comparison of both the models on the given metrics.

Table 1: shows the study of the different prediction models for the Apple data.
(MAPE,MAE,RMSE in %)

Technique	**MAPE**	**MAE**	**RMSE**
LSTM	4.714	3.036	3.914
GRU	3.440	2.090	2.806

Table 2: shows the study of the different prediction models for the Microsoft data.

Technique	**RMSE**	**MAE**	**MAPE**
LSTM	12.43	9.91	0.1608
GRU	9.41	7.23	0.1229

Table 3: shows the study of the different prediction models for the Microsoft data.

Technique	**RMSE**	**MAE**	**MAPE**
LSTM	24.62	18.22	0.1303
GRU	19.47	14.73	0.1615

The GRU model outperforms the LSTM prediction techniques for the given dataset and for the parameters given. This method can also be applied to other datasets and to stock tickers. The results in Table 1,2 and 3 demonstrate that the GRU model achieved the lowest error compared with the LSTM method for different datasets.

Chapter 5

Standards Adopted:

The following coding standards were selected, and we made care to keep them in mind and uphold them throughout the project's development:

1. We tried to write as few lines of code as we could.
2. We used code cells in Jupyter Notebook to organise our code.
3. Used comments and text cells to highlight specific areas of our code.
4. Used naming standards that were suitable.
5. Code blocks in the same area were divided into paragraphs.
6. Used correct formatting and indentation.
7. Avoided using long functions and adhered to the rule that, if possible, a single function should do a single job.
8. Whenever possible, automated repetitive tasks to prevent the reuse of the same code.
9. Avoided deep nesting since there were too many layers.
10. Avoided writing long lines of code and verticalized long functions (functions with lots of arguments).
11. User input was used to test and verify our code, and everything was put together to produce clean, understandable, efficient code with few errors.

Chapter 6

Conclusion and Future Scope:

6.1 Conclusion:

For stock price prediction in this research, we employed the LSTM and GRU deep learning models. The models were tested using a variety of performance criteria, including mean squared error, mean absolute error, and R-squared score, after being trained on historical stock data. The outcomes demonstrated that the LSTM and GRU models were capable of forecasting stock prices with a respectable degree of accuracy. However, the LSTM model outperformed the GRU model by a small margin.

6.2 Future Scope:

There are several areas where this project can be extended or improved:

1. **Feature Engineering:** In this project, we have used only the stock prices as features. However, there are several other factors such as news sentiment, economic indicators, and company announcements that can affect the stock prices. Including these factors as features can improve the performance of the models.

2. **Hybrid Models:** We can combine LSTM and GRU models with other machine learning algorithms such as Random Forest, XGBoost, or Support Vector Regression to improve the accuracy of the predictions.

3. **Hyperparameter Tuning:** Hyperparameters such as the number of LSTM/GRU layers, the number of hidden units, the learning rate, and the batch size can significantly affect the performance of the models. Fine-tuning these hyperparameters can improve the accuracy of the models.

4. **Real-Time Prediction:** In this project, we have trained the models on historical data. However, the models can be further extended to predict stock prices in real-time using streaming data.

Overall, the use of LSTM and GRU models for stock price prediction is a promising area of research, and there is still scope for improvement and innovation in this field.

References:

[1] Nirupama Parida, Bunil Kumar Balabantaray, Rajashree Nayak, Jitendra Kumar Rout. "A Deep Learning based Approach to Stock Market Price Prediction using Technical indicators", 2023 International Conference on Advances in Intelligent Computing and Applications (AICAPS), 2023

[2] CHEN Kai, ZHOU Yi, DAI Fangyan. A LSTM-based method for stock returns prediction: A case study of China stock market[C] //IEEE International Conference on Big Data. Santa Clara:IEEE Press,2015:2823-2824. DOI :10.1109/BigData.2015.7364089..

[3] Mingzhu Jia,Jian Huang,Lihua Pang, Qian Zhao 'Analysis and Research on Stock Price of LSTM and Bidirectional LSTM Neural Network'3rd International Conference on Computer Engineering, Information Science & Application Technology (ICCIA 2019)

[4] Weng, Bin, et al. "Prediciting short-term stock prices using ensemble methods and online data soruces." Expert Systems with Applications 112(2018) : 258-273

[5] Naeini, Mahdi Pakdaman, Hamidreza Taremian, and Homa Baradaran Hashemi. "Stock market value prediction using neural networks." 2010 international conference on computer information systems and industrial management applications (CISIM). IEEE, 2010.

INDIVIDUAL CONTRIBUTION REPORT:

COMPARISION OF DEEP LEARNING MODELS FOR STOCK PRICE PREDICTIONS

Aashmit Shrestha
20051712

Abstract:

Stock Prices are highly volatile and depend on a number of factors and predicting stock prices is extremely difficult. Implementing two Deep Learning Models LSTM and GRU for Stock Price Prediction and comparing their accuracies which will help in choosing which deep learning method performs better given the same dataset and situations.

Individual contribution and findings: In order to implement the given project, Aashmit worked on the fundamentals of this project i.e the research work and implementing deep learning models. He worked on designing and deploying the models in the Jupyter Notebook and successfully implemented the LSTM and GRU models in collaboration with the team using Tensorflow. Tensorflow is not a part of the university curriculum and this project gave the perfect opportunity to learn that. The training and fitting of the models took more than 5 minutes every time and it was a real test of patience and it had to be done multiple times over a period of weeks while tuning the parameters and starting over with the training process again and again which was a great learning experience .

Individual contribution to project report preparation: Aashmit worked extensively on the preparation of the Abstract, Requirements Specifications and the Methodologies for the preparation of the given report.

Individual contribution for project presentation and demonstration: Aashmit is presenting the the final project application as well as the Jupyter Notebook and demonstrating the implementation of this project.

……………………………   ……………………………

Full Signature of Supervisor: Full signature of the student:

INDIVIDUAL CONTRIBUTION REPORT:

COMPARISION OF DEEP LEARNING MODELS FOR STOCK PRICE PREDICTIONS

Aayush Ratna Sthapit
20051713

Abstract:

Stock Prices are highly volatile and depend on a number of factors and predicting stock prices is extremely difficult. Implementing two Deep Learning Models LSTM and GRU for Stock Price Prediction and comparing their accuracies which will help in choosing which deep learning method performs better given the same dataset and situations.

Individual contribution and findings: Aayush worked on the collection and preprocessing of the data. Finding valid data is the foremost step of any machine learning process and correctly preprocessing them for use is crucial. Learning functions from the scikitlearn library, Aayush scaled the data down and made it ready for use for the team to implement and with Bishal, he worked on visualizing the data using the matplotlib library of Python which gave the project direction and finally complete successfully . Some datasets (e.g SSNLF stock) had inconsistent data and were proving to be a roadblock for our project and Aayush helped the team face these drawbacks head on.

Individual contribution to project report preparation: Aayush worked on the Abstract, SRS and the Methodology to successfully prepare the project report and compiled the contributions of all the members and formatted them as required.

Individual contribution for project presentation and demonstration: Aayush will be demonstrating the data collection and preprocessing procedure for the given project.

……………………………	……………………………
Full Signature of Supervisor:	Full signature of the student:

INDIVIDUAL CONTRIBUTION REPORT:

COMPARISION OF DEEP LEARNING MODELS FOR STOCK PRICE PREDICTIONS

Bishal Pandey
20051723

Abstract:

Stock Prices are highly volatile and depend on a number of factors and predicting stock prices is extremely difficult. Implementing two Deep Learning Models LSTM and GRU for Stock Price Prediction and comparing their accuracies which will help in choosing which deep learning method performs better given the same dataset and situations.

Individual contribution and findings: After training ,testing the dataset for the predictions made and visualizing the results is quintessential . Bishal worked on testing the dataset and analysing what the results for the predictions made by the models using various metrics (RMSE,MAE,MAPE) and visualized and helped the team in evaluating which model worked best. MAPE values were found to be arbitrarily high for both the models. On further reading about it, it was found that the test values 0 or close to 0 would give arbitrarily high values for MAPE which was an interesting finding of this project.

Individual contribution to project report preparation: Bishal worked extensively on chapter 4 , on the Testing and Result analysis section where 3 tables comparing the performance of the models on 3 different datasets has been presented.

Individual contribution for project presentation and demonstration: Bishal will be demonstrating the testing and result analysis procedure for the given project.

……………………………	……………………………
Full Signature of Supervisor:	Full signature of the student:

INDIVIDUAL CONTRIBUTION REPORT:

COMPARISION OF DEEP LEARNING MODELS FOR STOCK PRICE PREDICTIONS

Naresh Khatiwada
20051737

Abstract:

Stock Prices are highly volatile and depend on a number of factors and predicting stock prices is extremely difficult. Implementing two Deep Learning Models LSTM and GRU for Stock Price Prediction and comparing their accuracies which will help in choosing which deep learning method performs better given the same dataset and situations.

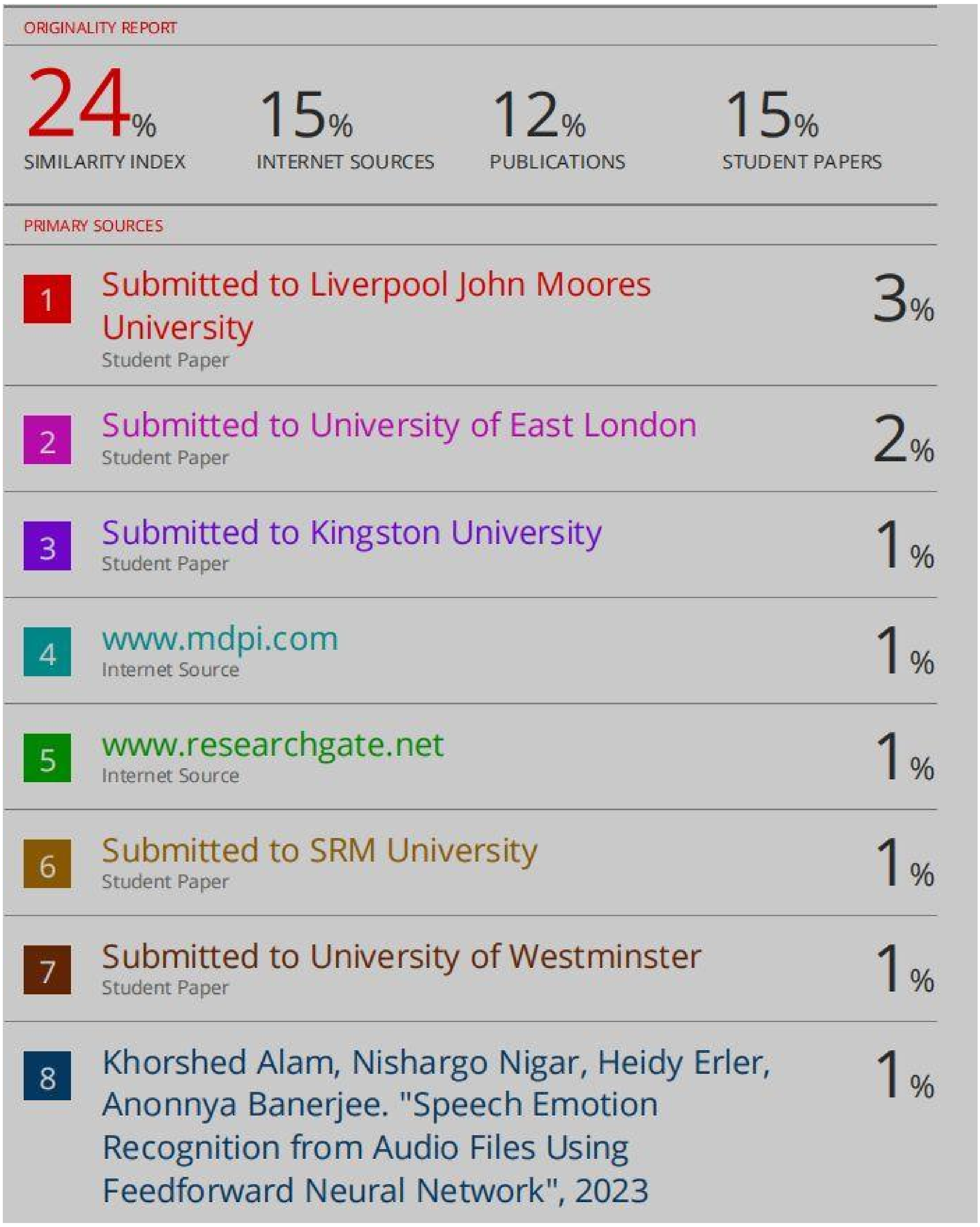
Individual contribution and findings: The models were trained in the jupyter notebook platform and exported. Naresh imported those trained models and deployed it on the web using Streamlit. The web app developed can take in any input from the user and produce all the visualized results as well the evaluation table for different stock datasets.

Individual contribution to project report preparation: Naresh worked extensively on all the sections related to Streamlit, the system architecture and the Standards adopted chapters of the report. He also helped produce the results for tables 2 and 3.

Individual contribution for project presentation and demonstration: Naresh will be demonstrating the Webapp produced using streamlit and result analysis on the web browser of the local host.

…………………………….	…………………………….
Full Signature of Supervisor:	Full signature of the student:

TURNITIN PLAGIARISM REPORT



ORIGINALITY REPORT

24%	15%	12%	15%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Liverpool John Moores University Student Paper	3%
2	Submitted to University of East London Student Paper	2%
3	Submitted to Kingston University Student Paper	1%
4	www.mdpi.com Internet Source	1%
5	www.researchgate.net Internet Source	1%
6	Submitted to SRM University Student Paper	1%
7	Submitted to University of Westminster Student Paper	1%
8	Khorshed Alam, Nishargo Nigar, Heidy Erler, Anonnya Banerjee. "Speech Emotion Recognition from Audio Files Using Feedforward Neural Network", 2023	1%