UNIVERSITY OF GHANA
COLLEGE OF BASIC AND APPLIED SCIENCES
FACULTY OF PHYSICAL AND MATHEMATICAL SCIENCES

CONSTRUCTION OF AN ENSEMBLE SCHEME FOR STOCK PRICE PREDICTION USING DEEP LEARNING TECHNIQUES

BY
ISMAIL WAFAA DENWAR
(INDEX NO. 10807273)

A THESIS SUBMITTED TO THE SCHOOL OF GRADUATE STUDIES IN PARTIAL FULFILMENT OF THE AWARD OF DEGREE OF MASTER OF SCIENCE IN COMPUTER SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

OCTOBER 2020

UNIVERSITY OF GHANA
COLLEGE OF BASIC AND APPLIED SCIENCES
FACULTY OF PHYSICAL AND MATHEMATICAL SCIENCES

CONSTRUCTION OF AN ENSEMBLE SCHEME FOR STOCK PRICE PREDICTION USING DEEP LEARNING TECHNIQUES

BY
ISMAIL WAFAA DENWAR

DEPARTMENT OF COMPUTER SCIENCE
OCTOBER 2020

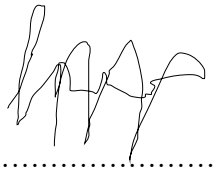
DECLARATION

I hereby declare that except where specific references are made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this or any other University.

ISMAIL WAFAA DENWAR

October 2020

This dissertation was under the supervision of the Supervisory Committee as detailed below:



…… Date: 02/10/2020 …….

ISMAIL WAFAA DENWAR (10807273)

…… Date: …….

DR. JUSTICE K. APPATI (Principal Supervisor)

…… Date: …….

DR. EBENEZER OWUSU (Co-Supervisor)

DEDICATION

This thesis is dedicated to my father, Dr. Nicholas Denwar, mother, Mrs. Sawda Denwar, sister, Hawa Junuoh Denwar, uncles, Baba Seidu, Wahab Ibrahim, James Vuyon, John Weelnibeh (deceased), and to the Department of Computer Science, UG.

ACKNOWLEDGMENTS

I am appreciative of the Almighty Allah for His favour, love, and gift upon my life all through my one year stay at the University of Ghana Campus. I express gratitude toward Him for giving me knowledge and shrewdness from above to finish my research work.

I can't thank my folks enough, Dr. also, Mrs. Denwar, for their help and supplications all through my stay at the University of Ghana Campus.

I want to offer my sincerest gratitude to Dr. Justice K. Appati, my Supervisor, for the dynamic help on my task. To him, I say, God bless you.

Likewise, my much gratitude goes to Dr. Ebenezer Owusu (Co-Supervisor), DR. Jamal-deen Abdulai, head of the department, and all faculty members of the Computer Science Institute.

To my course, mates, friends, and relatives, thank you, and may God richly favor you for your help.

ABSTRACT

This study proposes a deep learning approach for stock price prediction; the study sought to predict the closing price for the next day using the S & P 500 dataset. The methods employed were the Long Short Term Memory, Gated Recurrent Unit, and Autoencoder. These are variants of the Recurrent Neural Network. The metrics used for evaluating the models were the Mean Absolute Error and Mean Squared Error. Two models were proposed. From the observation, the first proposed model performed poorly with higher errors than the existing models. The second proposed model performed well; it showed the ability of the output to model the input. The first proposed model uses the LSTM, GRU, and Autoencoder, where the LTSM's output is passed through the GRU and then to the Autoencoder for the final prediction. The second proposed model uses the Bidirectional LSTM & Bidirectional GRU; the model extends the study of Hossain et al. (2018), which had an MSE of 0.00098 as their lowest. The second proposed model had 0.00000008 MSE, which was the lowest error recorded. The LSTM model recorded 0.00000025 MSE, the GRU model recorded 0.00000077 MSE , the LSTM & GRU model recorded 0.00000023 MSE. Other combinations of the existing models such as the Bi-directional LSTM model recorded 0.00000019 MSE, Bi-directional GRU recorded 0.00000011 MSE, Bidirectional LSTM & GRU recorded 0.00000027 MSE, LSTM & Bi-directional GRU recorded 0.00000020 MSE.

Keywords: Autoencoder, Gated Recurrent Unit (GRU), Long Short Term Memory (LSTM), Mean Square Error (MSE), S&P 500

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1 - Introduction

1.0 OVERVIEW OF STOCK MARKET

The trading pool involving the buying or selling of shares is referred to as a stock market. The dealers make profits if the trading goes in their favor (Pun & Shahi, 2018). There is a relationship between the financial market and the economy. It is a means for raising capital for the firms involved (Billah, Waheed, & Hanifa, 2017). The stock market impact can significantly affect the global economy, where it is currently revolving. Publicly listed companies and businesses thrive in the financial markets. They analyze the trends, either the profits generated or losses incurred. The markets are analyzed from different dimensions ranging from stocks, foreign exchanges, bonds to market indicators. Therefore, research in this field has become a hot cake; business owners and investors need information (Samarawickrama & Fernando, 2018). Share or stakeholders find the financial markets fascinating as returns and risk could be extremely high. Investors crave for insights about the stock market to decide either buying or selling portions of stocks and looking to swell profit on investment capital. However, forecasting stock prices can be hard since it is highly volatile; some of the factors range from the global economy, events, politics, to the investor(s) involved(Oncharoen & Vateekul, 2018). According to Hoseinzade & Haratizadeh (2019), financial markets are presumed a vital part of the world's economy in which, daily, billions of dollars trade. Accurately forecasting patterns of the stocks can come in handy in all aspects. An economy's growth can be triggered by stocks depending on how well it is doing. The insights will, therefore, be useful to businesses that revolve around the market's performance.

1.1 BACKGROUND OF STUDY

In stock price forecasting, the objective is to foresee the future estimation of the financial stocks of an organization. A right prediction of stocks can prompt huge benefits for the brokers involved. Every now and again, estimating draws out the thought that it is noisy instead of arbitrary; it very well may be predicted via cautiously breaking down the historical backdrop of the separate financial stock market (Parmar et al., 2018). Predictions on stock prices have been the object of studies for many decades. However, due to its chaoticness and dynamism, studies have concluded that forecasting stock price is a difficult task (Nelson, Pereira, & Oliveira, 2017). AI has demonstrated to be a proficient method to incorporate such procedures, especially in the area of machine learning. Its presentation in the zone of the stock forecast has captivated numerous kinds of research due to its dynamic and exact estimation showcase (Parmar et al., 2018). The imperative piece of machine learning is the dataset utilized. The dataset must be cleaned since a minute distortion of the data can influence the outcome negatively; the predictions will be rendered inaccurate (Parmar et al., 2018). AI has increased wide consideration because of its capacity to gain from huge information and recognize stock patterns; it is dynamical.(Kumar, Dogra, Utreja, & Yadav, 2018). In order to aid prospective owners of shares to make appropriate decisions, predicting changes in stock prices in the future can be done by studying the patterns of an earlier time. One of the methods with the potential to resolve this problem is the Neural Network (NN). The design of NN is to mimic how the human mind processes information; it is one of the information systems for solving a problem (Prastyo, Junaedi, & Sulistiyo, 2017). Recurrent Neural Networks (RNN) learn from output information as it is inputted recursively, consequently, alluded to as a feedback network. In the financial domain, a lot of RNN models are utilized for the stock forecast (Samarawickrama and Fernando, 2018). A neural networks' ability to gain recursively

from a chronicled input arrangement is alluded to as the Long Short-Term Memory (LSTM). It flourishes with input successions (past) of the vast information it is given. LSTM presents the memory cell, entryway (gate) structure, which has demonstrated to have the option to associate memories and input in time adequately. Over time the LSTM has the capacity to dynamically understand the long-term structure of data (Yao et al., 2018). Gated Recurrent Units (GRU) is also a subtype of RNN. The gating mechanism is a core component of the GRU recurrent neural networks. A GRU can refer to as an LSTM, but the disparity between the LSTM and GRU is that a GRU lacks the presence of an output gate (Samarawickrama & Fernando, 2018).

1.2 PROBLEM STATEMENT

Techniques adopted for the prediction of stock prices, such as fundamental and technical analysis, is a constraint since it is not able to learn the dynamics of the previous data in detail; these conventional methods lack the capabilities (Selvin, Vinayakumar, Gopalakrishnan, Menon, & Soman, 2017). Entire life savings could be lost as the dynamic and chaotic attribute of the stock market makes the forecasting a gamble (Nayak et al., 2016). Numerous approaches proposed presently cannot yield the outcome and execution expected, as it is difficult forecasting in the stock market field (Misra & Chaurasia, 2019).

1.3 RESEARCH QUESTION

Is it conceivable to construct a model to precisely foresee stock price with the S and P 500 dataset utilizing recurrent neural systems?

1.4 OBJECTIVE

This study's main objective is to investigate and propose models to improve stock prediction accuracy with fewer errors. Specifically, the objectives are to:

- To construct an extended version of the study of Hossain et al. (2018)
- To evaluate the performance of the extended model
- To forecast future stocks through the existing models
- To analyze the behavior of the proposed model on future stocks

1.5 OUTLINE OF METHODOLOGY

The methods adopted for the prediction of the S & P 500 dataset are the Long Short Term Memory (LSTM) and the Gradient Recurrent Unit. The metrics for evaluating the hybrid models (proposed) are the MSE for Mean Square Error and the MAE for Mean Absolute Error. The study employs PYTHON. Python 3.7 language (Anaconda), together with the Keras and Tensorflow library, this was to run the algorithms; the Adam optimizer also adopted to optimize the models. The following is the outline of the methodology to be used in achieving the objectives in five steps:

- The first step is downloading the dataset in CSV format from Yahoo Finance
- Data processing takes place in the second step. Data is normalized using the MinMax scalar, which scales the values from 0 to 1.Training and testing dataset is split
- Performing feature extraction is observed in the third step, which is the selection of the features for training the neural network. The features to be extracted (S & P 500) are volume, high price, close price, low price, open price, and date
- The fourth stage uses the LSTM & GRU network for training the dataset. It uses extracted features in step three

- The last step is the output generation. The predicted values will be compared with the real values to determine the Accuracy. It makes use of the MAE, and MSE metrics to give insights on the level of Accuracy

1.6 JUSTIFICATION

This study will be relevant in the field of academics for further studies to improve the Accuracy. This study will also be of relevance to stakeholders of the Bank of Ghana (BOG) as they rely on the stocks to make business decisions. Lastly, it will be of significance to the population, such as the banks, as they also rely on the shares to make business decisions.

1.7 CONTRIBUTION

The proposed models have shown that it is conceivable to predict stock prices with fewer errors. The Second proposed model (Bidirectional LSTM and Bidirectional GRU model) trains faster and records fewer errors than the existing models after training. However, though the first proposed model (LSTM, GRU & Autoencoder), which recorded higher errors, can be used in the field of Natural Language Processing, the model could be trained to translate from one language to the other.

1.8 ORGANIZATION OF CHAPTERS

There are five chapters in this thesis describing the studies carried out.

Chapter one contains the introduction and discussion of stock prediction, includes the background, the problem statement, outline of the methodology as well as the objective of the study. In this chapter, the study is justified.

Chapter two contains a literature review of the various studies conducted regarding this project; it includes the techniques and approaches employed.

Chapter three is the methodology; this discusses the stages for the prediction in detail; it includes methods and metrics.

Chapter four from the analysis examines the level of Accuracy. The number of layers, epoch, and batch size is adjusted to observe the patterns.

Chapter five eventually concludes this study with some recommendations and future works.

Chapter 2 - Literature Review

2.1 INTRODUCTION

This section delves into various articles reviewed in the course of the research work. One major area with numerous ongoing research studies in the domain of machine learning is stock price prediction. The purpose of this section is to get insights into the recent work done to identify gaps and improve on them.

2.2 REVIEW OF RELATED STUDY

The study of Soni, Agarwal, Agarwel, Arora, & Gupta (2018) sought to build a model for stock price prediction. The methods employed were Black-Hole, Decision Tree, PSO, Naïve Bayes. The proposed model was a "nature-inspired technique" that used a matrix with values 1 and 0 for prediction. The database for training and testing the models was the Nifty stock index dataset. The metric used for evaluating the models was mean Accuracy. The proposed model had a higher mean accuracy of 96.10% than the rest of the models. Decision Tree had a mean accuracy of 81.16%, PSO 83.57%, Naïve Bayes 85.56%, Black Hole 95.10 %. The study of Rajput & Kaulwar (2019) attempted to predict the closing price of a stock in the National Stock Exchange of India. Artificial Neural Networks (ANN) and Support Vector Machine (SVM) were the techniques utilized. The database for training and testing the models was Larsen & Toubro (LT) and State Bank of India (SBIN). The metrics the models were assessed on were the Normalized Mean Square Error (NMSE), which was additionally changed over to a percentage. The findings demonstrate that SVM conveyed better outcomes contrasted with ANN with PCA. SVM gave a 0.01 NMSE and a 1.11 % for the SBIN dataset, and 0.04 NMSE and an error level of 1.91 % with an LT dataset.

ANN (PCA) report gave 0.03 NMSE and a 2.12 % error level for the SBIN dataset and 0.09 NMSE and a 2.41 % error level for the LT dataset.

The study of Nayak, Pai, & Pai (2016) intended to predict stock market trends on a daily and monthly basis for the oil, mining, and banking sector. The methods employed in the study were Support Vector Machine (SVM), Logistic Regression (LR), and Boosted Decision Trees (BTS) models. The datasets for training and testing the models were from Yahoo Finance. The metric for evaluating the models was Accuracy. In their findings, BTS performed better than LR and SVM. BTS had an accuracy 0.548, 0.76, 0.769 for banking, mining and oil respectively; for LR 0.654, 0.61, and 0.442 and, lastly for SVM 0.51, 0.59, and 0.442. The study of Ouahilal, Mohajir, Chahhou, & El Mohajir (2017) tried to optimize the prediction of the stock price with a novel Hybrid approach. The methods employed were Support Vector Regression (SVR) with the Hodrick-Prescott filter (Optimizing prediction). Christiano Fitzgerald (CF) filter and Band-Pass (BP) filter using the Fourier transformation were also used with SVR to compare results. The database for training and testing their models was Maroc Telecom (IAM) financial time series. The metric for training and testing the models was the Mean Average Percentage Error (MAPE).In their findings, the proposed model gives better results in terms of stock price predictions. SVR had a MAPE of 0.29, SVR+HP 0.11, SVR+CF 0.51, SVR+BP and 0.22. No future work was stated in this paper.

The study of Pun and Shahi (2018) sought to forecast the stocks for the following day from the Nepal Stock Exchange. The techniques/methods utilized in this appraisal were Support Vector Regression (SVR) and Artificial Neural Networks (Back-Propagation Neural Network) models. The information utilized for preparing and testing the models was the Nepal Stock Exchange (NEPSE). Min-Max and Z-score are utilized to standardize the data. The metrics/measurements

for assessing the models are mean absolute error (MSE), mean absolute error (MAE), root mean square error (RMSE), and Coefficient of Determination is utilized. In their discoveries, SVR utilizing the min-max scale for cleaning outperformed BPNN in all segments, with the exception of the mutual fund, finance, and development bank. The two models discovered reduced precision in Trade and Factory divisions. BPNN was additionally seen as preferred utilizing z-score over utilizing min-max in these parts. Be that as it may, SVR performed better than BPNN in all divisions on the average. The authors in this paper expect to up the dataset size for future research.

The study of Rasel, Sultana, & Hasan (2017) attempted to predict stock prices and trends using time series data of 1-day ahead market. The methods/techniques employed were Artificial Neural Network (ANN), K-Nearest Neighbor (KNN), and Support Vector Machine (SVM). The database for training and testing the models was the Wal-Mart Stores Inc. (WMT) dataset. The attributes of the dataset were high, open, low, and volume. The metrics for evaluating the model was the Mean Average Percentage Error (MAPE) and Root Mean Square Error (RMSE).In their findings, ANN performed better than KNN and SVM. ANN had a MAPE of 0.75 and an RMSE of 0.60. SVM had a MAPE of 2.75 and an RMSE of 1.90. KNN had a MAPE of 2.71 and RMSE of2.28. The authors in this paper intend to use different models and datasets of stock markets to build a universal model. The study was limited to ANN, KNN, and SVM. The study of Selvin, Vinayakumar, Gopalakrishnan, Menon, & Soman (2017) sought to identify the underlying existing dynamics of the dataset to be utilized with deep learning for the price prediction rather than fitting the information in a specific model. The methods employed were Long-Short Term Memory (LSTM), Recurrent Neural Network (CNN), and Concurrent Neural Network with a sliding window. The databases used were Infosys, TCS, and Cipla. The metric used to evaluate the models was an error percentage (EP). In their findings, CNN outperformed LSTM and ANN.

CNN had an EP OF 2.36, LSTM had 4.18, and RNN had 3.90 with Infosys data. CNN had an EP of 3.63, LSTM had 3.94, and RNN had 3.83 with the Cipla dataset. However, CNN performed poorly with the TCS dataset; it had an EP of 8.96, LSTM had 7.82, and RNN had 7.65. No future works were stated in this paper.

The study of Weng, Lu, Wang, Megahed, & Martinez (2018) sought to build an expert system for predicting short-term financial prices of historical data. The methods employed were four ensemble models. They were Support Vector Regression, Random Forest Regression, Boosted Regression Trees, and Neural Network Regression Ensemble. The database for training and testing the models was the Citi Group stock ($C) stock price data set. The metric for evaluating the models was the Mean Absolute Percent Error (MAPE), Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE).In their findings, the Boosted Regression Tree performed better than Support Vector Regression, Random Forest Regression, and Neural Network Regression of the 19 stocks.

The study of Parmar et al. (2018) wanted to predict the future prices of the stocks of an organization. The methods employed were Linear Regression and Long Short Term Memory (an advanced form of RNN). The dataset the models trained and tested from was downloaded from Yahoo Finance. The metrics for evaluating the models were Mean Squared Error (MSE) and Root Mean Squared Error (RMSE). In their findings, the model led to a 0.00106 MSE (0.03 RMSE) train score and a 0.00875 MSE (0.09 RMSE) test score. The models improve as they are trained, and also the size of the dataset is a factor. Higher the Accuracy, is noticed when trained on large inputs sequences. The LSTM model yielded better Accuracy than the Regression Model. The study of Du, Liu, Chen, & Wang (2019) aim was to predict stock prices using the single feature (univariate) and multi-feature input variables to appraise the model's prediction ability on stock

time series. The method employed was the Long Short Term Movement (LSTM) from Recurrent Neural Networks (RNN). The dataset used for training and testing the models was the Apple Stocks dataset. The attributes of the database were. The metrics for evaluating the model was Mean Square Error (MSE) and Mean Absolute Error (MAE). In their findings, the multivariate feature input outperformed the univariate feature input. The univariate feature had an MSE of 0.024, and the univariate feature had an MSE of 0.035. The multivariate feature had an MAE of 0.033, and the univariate feature had an MSE 0.155. Future works were not stated in this paper. The study was limited to Long Short Term Movement (LSTM) from Recurrent Neural Networks (RNN), and a single stock dataset was used.

The study of Samarawickrama & Fernando (2018) goal was to build models to predict the daily stock prices of the Colombo Stock Exchange (CSE). The methods employed were Feedforward Multi-Layer Perceptron (MLP), Simple Recurrent Neural Network (SRNN), Gated Recurrent Unit (GRU), and Long Short Term Memory (LSTM) architecture. The dataset for training and testing the models was the Colombo Stock Exchange (CSE) dataset. The metrics for evaluating the models were Mean Absolute Deviation (MAD) and Mean Absolute Percentage Error (MAPE). In their findings, MLP outperformed LSTM. SRNN and GRU performed averagely. MLP had the lowest errors (MAD – 1.7636; MAPE – 0.86%), LSTM (MAD – 2.2701; MAPE – 1.13%), Simple Recurrent Neural Network (SRNN) achieved a MAD ranging from 0.72 – 6.57 for each company (MAPE ranging from 0.67% - 5.60%). Gated Recurrent Units (GRU) yielded higher errors. The study of Hossain, Karim, Thulasiram, Bruce, & Wang (2019) sought to predict the closing price for the next day for the regression-based problem with a proposed model. The methods employed were Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU). The database for training and testing the models was the S and P 500 historical time series. The metrics used for

evaluating the models were Mean Square Error (MSE), Mean Absolute Error (MAE), and Mean Absolute Percent Error (MAPE).In their findings, the proposed hybrid model recorded a 0.00098 MSE, which outperformed all the existing neural network models utilizing the same dataset with a higher gap. It had an MAE of 0.023 and a MAPE of 4.13.

The study of Yao et al. (2018) aimed to predict a short-term price movement. The method employed was the Long Short Term Memory (LSTM), which is an alternate in the area of Recurrent Neural Network (RNN) and a Random Method. The database for training and testing the models was the CSI 300 constituent stocks dataset. The metrics used for evaluating the models were e precision (AccRF), the recall rate (RecRF), and the critical error rate (CerRF). In their findings, the LSTM performed better than the random method. The average AccRF, RecRF, and CerRF for LSTM are 28.54%, 38.15%, and 16.94%, respectively, while that of the Random method yielded 20.96%, 20.42%, and 23.67%, respectively. The study of Murkute & Sarode (2015) attempted to predict stock price. The database for training and testing the models was the Infosys dataset from yahoo finance. The method employed was the Backpropagation algorithm from Artificial Neural Network. The metrics for evaluating the model was Mean Square Error (MSE). In their findings, the backpropagation neural network had an MSE of 0.1830. The outcome for the data forecast after comparing to the actual data demonstrated huge contrasts. The outcome of the forecast took the patterns of actual data trends extending to two days. However, after the forecast during the third day, the outcome did not take the patterns of the real data. The difference in the stock's daily closing price could be a factor.

The study of Kumar et al. (2018) aimed to predict the stock market trend. The methods they proposed were Support Vector Machine (SVM), Random Forest, K-Nearest Neighbor (KNN),

Naïve Bayes, and Softmax. The databases for training and testing the models were downloaded from data sources such as Quandl, NSE-India, and YCharts. The metrics for validating the models were F-measure and Accuracy. In their findings, the Random Forest algorithm performed well on large datasets while Naïve Bayes Classifier performed well on small datasets. The investigation of Labiad, Berrado, and Benabbou (2016) sought to foresee 10 minutes ahead of varieties of the Moroccan financial exchange. The Random Forest (RF), Gradient Boosted Trees (GBT), and Support Vector Machine (SVM) were techniques used. Intraday costs (tick-by-tick information) of Maroc Telecom (IAM) stocks is utilized as a trial database to assess the exhibitions of the chose model. The metrics/measurements utilized for evaluating the models were Mean Absolute Deviation (MAD). For the technical analysis, Moving Average (MA), Rate of Change (ROC), Standard Deviation (SD), Psychological Line (PSL), Stochastic Oscillator, Relative Strength Index (RSI) and Observations/value varieties (Up, Down). The discoveries show that RF and GBT are better than SVM from the dataset. Moreover, the low computational complexities and decreased preparation time of RF and GBT are appropriate for short-term forecasting.

The study of Usmani, Ebrahim, Adil, & Raza (2019) goal was to predict the performance of the Karachi Stock Exchange (KSE) with a proposed Hybrid model. The methods employed were Radial Basis Function (RBF), Support Vector Machine, and Artificial Neural Network (ANN). The ANN was of two variants, including Single Layer Perceptron (SLP) and Multi-layer Perceptron (MLP), for prediction of the stock price. The database used to training and testing the models was the KSE-100 index of the Pakistan Stock Market. The metrics used for validation were Auto-Regressive Integrated Moving Average (ARIMA), and Simple Moving Average (SMA). In the discoveries, there were two variations of the Hybrid model proposed; Variant 1 gave about

72.8% precision, while Variant 2 gave 95.7% accuracy. The Hybrid model could not outperform the MLP based sub-model alone.

The study of Oncharoen & Vateekul (2018) objective was to improve the stock market predictions of historical price data. The methods employed were Convolutional Neural Network (CNN) and Long Short-term Memory (LSTM) architectures for the proposed model. The databases used for training and testing the models were Intrinio, Standard & Poor's 500 Index (S&P500), Dow Jones Industrial Average (DJIA), Reddit, and Reuters. The data were in numerical and textual format. The metrics for evaluating the models were Accuracy and Annualized Return. For technical analysis, the metrics used were Stochastics, Rate of Change, Disparity 5, Momentum, A/D Oscillator, and William's %R. In the findings, the proposed model is predicted with high precision against the existing models. The study of Li, Bu, & Wu (2017) desired to improve forecasting performance by combining market price information and investor sentiment. The methods employed were Long Short-Term Memory (LSTM) neural network for the stock prices and investor sentiments. Naïve Bayes sentiment classifier was used. The dataset used to training and testing the proposed model was CSI300 index values in the Chinese stock market. The metrics used for evaluating the model were Accuracy, Precision, Recall, and F-measure. In the findings, the proposed model gave a prediction accuracy of 87.86%, outperforming Support Vector Machine (SVM) models by 6%.

The study of Jiao & Jakubowicz (2018) focused on evaluating the performance of four classification algorithms for stock movement direction. The method employed were Random Forest, Gradient Boosted Trees, Artificial Neural Network, and Logistic regression. The database used for testing and training the model was the S&P 500 index. The metrics for validating the models were standard cross-validation, sequential Validation, and single validation. In their

findings, it was difficult to predict stocks from the past. Additional data such as recently closed European and Asian indexes to predict S and P 500 can improve the forecast. Moreover, it is comparatively less complicated to forecast than other industries among the various financial sector. The study of Ballings, Van Den Poel, Hespeels, & Gryp (2015) tried to predict stock prices. The methods employed were ensemble (Adaboost, Random Forest, Kernel Factory and Random Forest) and classifier models (Logistic Regression, Support Vector Machines, Neural Networks, and K-Nearest Neighbor). The databases used for training and testing the models was the Amadeus Database. The metric for validating the models was the area under the curve (AUC), Cross-Validation. Their findings show that Random Forest was the best performing algorithm, followed by Support Vector Machines (SVM), Kernel Factory, Adaboost, Neural Networks (NN), K-Nearest Neighbors (KNN), and Logistic Regression.

The study of Misra & Chaurasia (2019) purpose was to predict daily prices. The methods employed were Random Forest (RF), Support Vector Machine (SVM), and Artificial Neural Network (ANN). The dataset used to training and testing the models was the S&P BSE Sensex index. The data was discretized. The metrics for evaluating the models were Precision, Recall, F-Score, and Accuracy. The technical indicators used for predicting movements were the Relative Strength Index (RSI), Accumulation/Distribution (AD), William%R, Stochastic%K, Momentum, and Commodity Channel Index (CCI). In their findings, RF recorded the best Accuracy, followed by SVM and ANN. It is observed that using ensembles perform better than using a single model. The study of Cakra (2015) attempted to predict the Indonesian stock market using sentiment analysis. The methods employed were Naïve Bayes and Random Forest algorithm for classifying sentiments and the Linear Regression technique for the prediction model. The database for training and testing the models were Yahoo Finance CSV API (Indonesian companies) and twitter feeds. The metric

for evaluating the models was Accuracy. Tweets were being classified into three classes (positive, negative, and neutral) for sentiment analysis. In their findings, Random Forest had an accuracy of 60.39%, and Naïve Bayes had an accuracy of 56.50% for sentiment analysis. Using past (previous) stock price and hybrid feature gave the best prediction with 0.9989 and 0.9983 coefficient of determination. Further presumes that individuals' sentiments via social media, for example, Twitter, can be used for predicting the Dow Jones Industrial Average (DJIA) esteem with 87.6% precision.

The study of Jeevan, Naresh, & Vijaya (2018) aimed to predict share price. The methods employed were Long Short Term Memory (LSTM) and Recurrent Neural Networks (RNN). The database for training and testing the models was NSE data. The metric used for evaluating the models was the sliding window. In their discoveries, RNN based model demonstrated extremely viable in foreseeing the stocks. It utilized a backpropagation system while assembling and gathering information to avoid data from distorting. The study of Vargas, Dos Anjos, Bichara, & Evsukoff (2018) attempted to predict stock prices with technical indicators and financial news with a proposed model (SI-RCNN). The methods employed were Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM. The databases for training and testing the models were Reuters for business report and CVX stock price from Yahoo Finance. Two sets of metrics were used, set1: Stochastic %K, Stochastic %D, Momentum, William's %R, Rate of Change, Accumulation/Distribution (A/D) oscillator and Disparity 5; set 2: Relative Strength Index, Exponential Moving Average, Moving Average Convergence-Divergence On Balance Volume and Bollinger Bands. In their findings, SI-RCNN models forecasted profit (13.94% in 8 months) when measured up to the buy-and-hold strategy, which was 3.22% throughout the period. Also, it

showed that news and that of technical indicators as inputs yielded the desired forecast than using a single input such as LSTM for technical indicators alone.

The study of Omar et al. (2018) sought to predict the stock market price for the next day based on historical data of 15 years. The methods employed were multilinear regression algorithms, optimized with particle swarm Metaheuristic, Particle Swarm Optimization Algorithm, and Artificial Neural Network. The database used for training and testing the models were Brazilian and Chilean currency exchange dataset. The metric used for evaluating the models was the Mean Squared Error (MSE). In their findings, the Algorithms used proved outperformed the classic econometrics models. The ANN and PSO minimized errors after iterating for some time, which led to the prediction of the stocks with a coefficient of determination of 87% for the Brazilian data set and the Chilean market 92%.

The study of Zhang et al. (2018) goal was to predict the price trend for 30 days. The method employed were SVM (support vector machine), Naive Bayesian Classifier, Random forest, and Neutral Network. The dataset for training and testing the models was the Shanghai Stock Exchange (SSE) 50 index. The metric for evaluating the models was Accuracy. In their findings, the result shows that ANN outperformed the other three models; there is potential in finding some profitable insights. The study of Tang & Chen (2018) sought to predict stock price with historical price data and new as inputs with a hybrid model. The methods employed were Recursive Neural Network (RNN), Long Short-Term Movement (LSTM), Convolutional Neural Network (CNN), and Feed Forward Neural Network (FFNN). The Hybrid model was made of RNN and LSTM. The datasets used to train and test the models were Yahoo Finance and Reddit World News Channel. Yahoo Finance. The metrics used for evaluating the models were mean Accuracy. In their findings, the

proposed model drastically outperformed the existing models. The Hybrid algorithm had a mean accuracy of 54.45%, LSTM 52.64%, FFNN 50.33%, and CNN 51.38.

The study of Ta, Liu, & Addis (2018) attempted to predict stock price movement. The methods employed were Linear Regression and Support Vector Regression (SVM). The datasets the models trained and tested on were S&P 500 ETF-SPY of ten years of daily historical data using the Quandl API. The metrics for evaluating the models were Mean square error (MSE), Accuracy, and the Error rate was the metric. In their findings, the linear regression model outperforms the support vector regression in the short-term forecast, whereas the SVM model outperformed the regression in the long-term forecast. The study of Qian & Xiaoxia (2019) sought to predict stock price reversal points by including Independent Component Analysis. The method employed was SVM for Support Vector Machine and ICA for Independent Component Analysis. The proposed model combined ICA and SVM and was evaluated against the regular SVM. The database for training and testing the models was the Shanghai Stock Exchange. The metrics for evaluating the models were Recall, Precision, and F-measure. In their findings, ICA-SVM has proven to be an excellent method to predict stock price (reversal points) than the conventional SVM.

Chapter 3 - Methodology

3.1 INTRODUCTION

Chapter three investigates the current approach used in stock prediction and discusses the various techniques used in detail. This study employs the S&P 500 dataset the models are trained and tested on. It uses the Gated Recurrent Unit short for GRU and Long Short Term Memory short for LSTM for the prediction. These methods are recurrent neural networks and do not require supervision to learn. The individual forecast of the stock goes beyond sixty-five percent. Gated Recurrent Units (GRU), the recurrent neural networks, or a merge of the LSTM and GRU need further studies in the field of stock price prediction; it shows potentials.

3.2 DATA INFORMATION

The period of the S&P 500 set is January 01, 1950- December 31, 2019, that makes it seventy years. It comprises 17,611 rows, whereby the training set consists of 13,333 rows, and the testing set consists of 4,278 rows. The entire features present on the data are volume, date, low price, high price, close price, open price, and Adjusted close price. The S&P 500 dataset are the inputs (CSV format) to the system. The closing price is selected as the feature for conducting the studies.

3.3 DATA PREPROCESSING

The MinMaxScale is the preprocessing scale utilized before splitting the S & P 500 dataset into training and testing sets for cross-validation. In this investigation, the data is scaled into the length of [zero, one], where x is the feature vector, x_i is an individual element of feature x, and X_p is the rescaled element illustrated in Eqn 3.1. Preprocessing of data occurs to normalize the data values.

$$X_p = \frac{x_i - \min(x)}{\max(x) - \min(x)} \tag{3.1}$$

The train set data ranges from 1950 to 2002, and the test dataset is from 2003 to 2016. used For validating the S&P 500, ten percent of the training data is used, which implies that the train set is assigned eighty percent of the S & P 500 data, while the test is assigned twenty percent of the data. After downloading the file in CSV, it is transformed into a data frame using the Pandas. The investigation leaves twenty percent of the data available for testing.

3.4 NEURAL NETWORK TECHNIQUES

This section discusses the deep learning techniques that are used. They are the Long Short Term Memory, Gated Recurrent Unit, and Autoencoder. This section expatiate on these concepts and mathematically expresses how they work.

3.4.1 Long Short Term Memory

Regression-based problems are often associated with stock price prediction; LSTM and GRU are robust recurrent neural networks that, in terms of Accuracy, can perform better and quicker (Hossain et al., 2018). Samarawickrama & Fernando (2018) have stated in their paper that the LSTM architecture has three gates, namely the forget gate, input gate, an output gate, as illustrated in Figure3.1. LSTM is a potent antidote to the vanishing gradient problem the simple recurrent network cannot solve. The study claimed the LSTM was able to preserve the error that propagates through time and layers. Similarly, Parmar et al. (2018) state that the financial forecasting blossoms with predicting huge datasets; the gradient concerning the weight matrix may turn out to be little. It might compromise the learning rate, which compares to the Vanishing Gradient issue. LSTM keeps this from occurring. The LSTM is shaped by a forget gate, input gate, output gate, and a

remembering cell. The cell tracks the values propagated in the longterm and the gates filters. The LSTM employs linear units called Constant Error Carousels (CEC) to quash gradient vanishing and explosion issues plaguing previous RNNs. Each CEC has a fixed self-connection, which three gating units responsible for regulating the flow of data in and out of the CEC surround them (Yao, Luo, & Peng, 2018). Hossain et al. (2018) reiterate that the LSTM has a remembering cell (memory unit) that can hold a specific amount of training data. The input sequences in bidirectional LSTM (BLSTM) go forward and backward (direction) in timestep to secure the desired effect in learning during the process; it exploits all the input data. The third form of RNN is to stack several LSTM layers to build a Stacked LSTM (SLSTM) network to perform deep learning. It captures higher-order patterns in the time series at different dimensions (Althelaya, El-Alfy, & Mohammed, 2018). In the paper of Hossain et al. (2018), the LSTM transition equation is shown in Eqn 3.2

$$
\begin{aligned}
i_t &= \sigma\left(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)}\right) \\
f_t &= \sigma\left(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}\right) \\
o_t &= \sigma\left(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}\right) \\
u_t &= \tanh\left(W^{(u)}x_t + U^{(u)}h_{t-1} + b^{(u)}\right) \\
c_t &= i_t \odot u_t + f_t \odot c_{t-1} \\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}
\tag{3.2}
$$

Where time step is denoted as t, i_t is represented as the input gate. The f_t is identified as the forget gate. The o_t is denoted as the output gate. The c_t identifies as the cell. The u_t is a function for activation, the hidden state is defined as h_t. The weights are *W*, and *U*. Figure 3.1 presents the connections between the units.

3.4.2 Gated Recurrent Unit

The Gated Recurrent Unit neural network, featured in Figure 3.2 is an LSTM. The difference is the absence of the output gate; it is a gating mechanism in the neural network (Samarawickrama & Fernando, 2018). To discern between the LSTM and GRU is that GRU brings together the forget

gate and the input gates as an update gate. It fuses both the cell and the hidden states. The GRU model executes faster than the conventional LSTM models. However, the primary reason for using GRU is to serve the same goal as the LSTM (Hossain et al., 2018).

$$
\begin{aligned}
z_t &= \sigma\left(W^{(z)}x_t + U^{(z)}h_{t-1} + b^{(z)}\right) \\
r_t &= \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1} + b^{(r)}\right) \\
a_t &= \tanh\left(Wx_t + r_t \odot Uh_{t-1} + b^{(h)}\right) \\
h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot a_t
\end{aligned}
\tag{3.3}
$$

Eqn 3.3 shows the math behind the GRU, the z_t is an update gate. The a_t denotes the reset gate, a_t is the activation function. The h_t is a hidden state output gate. *W* and *U* denote the weights. Figure 3.3 illustrates the formation of the GRU network.

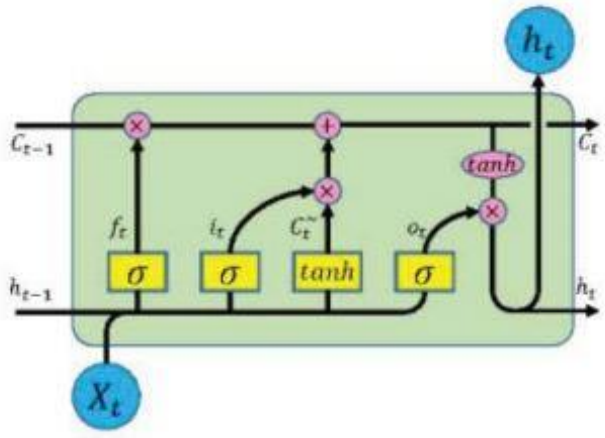


Figure 3. 1: The LSTM network (Hossain et al., 2018)

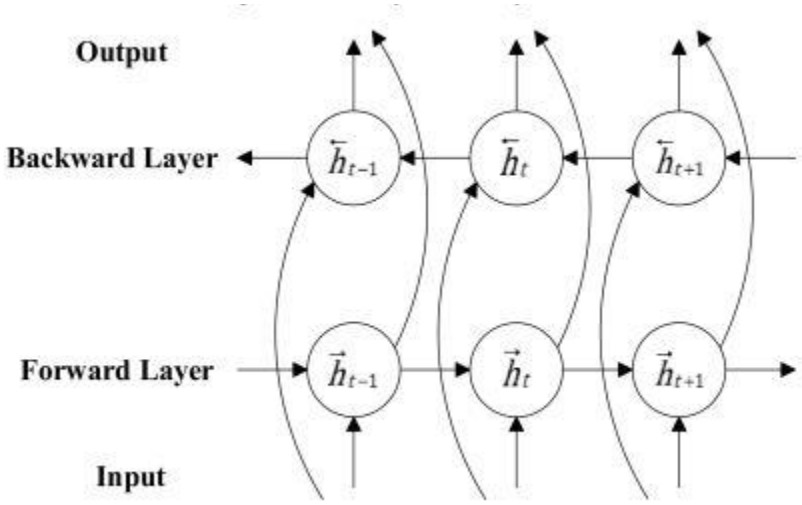


Figure 3. 2: The Bidirectional LSTM network (Althelaya et al., 2018)

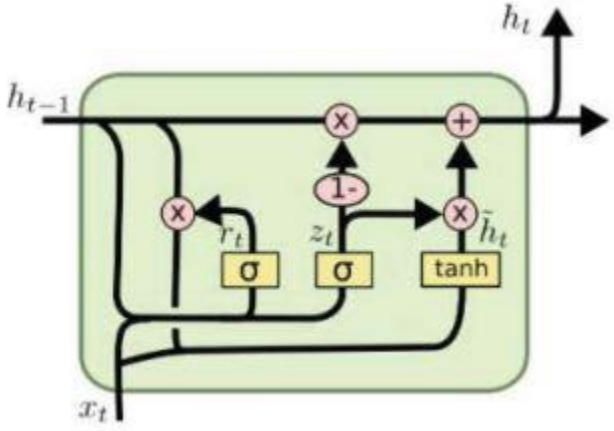


Figure 3. 3: The GRU network (Hossain et al., 2018)

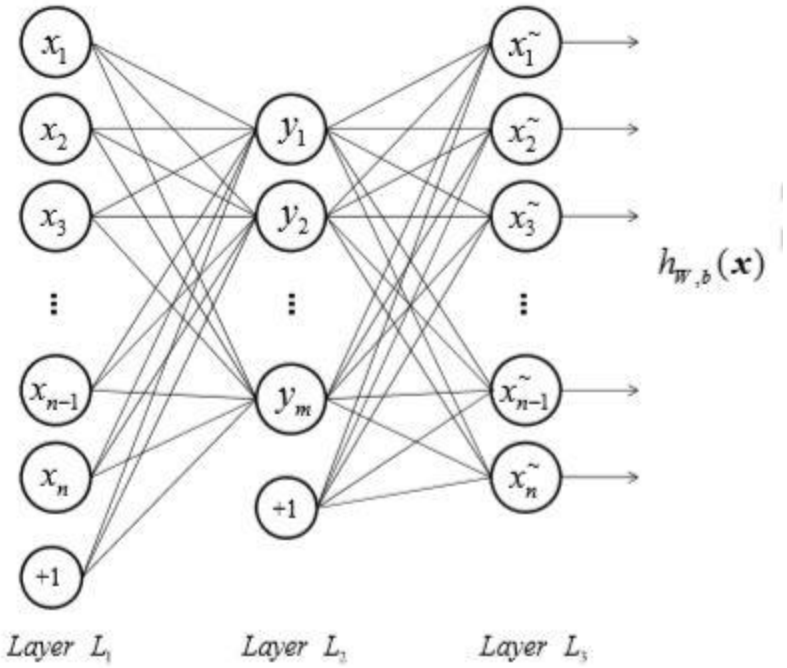


Figure 3. 4: The Autoencoder network (Wang, Yao, & Zhao, 2016)

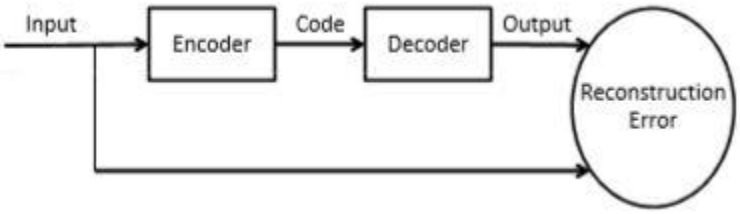


Figure 3. 5: The visualization description of the Autoencoder (Wang et al., 2016)

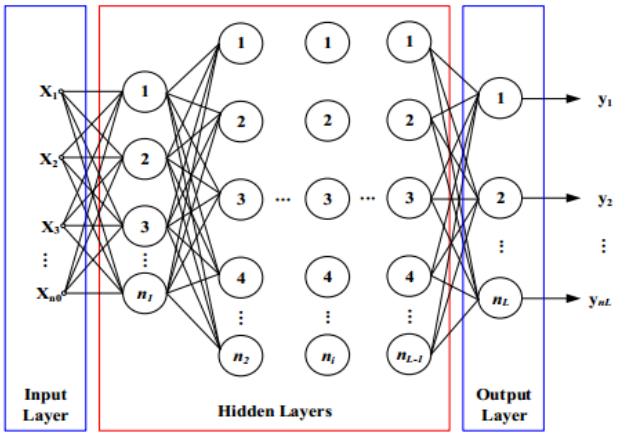


Figure 3. 6: The architecture of the MLP (Alameer, Elaziz, Ewees, Ye, & Jianhua, 2019)

3.4.3 Bidirectional LSTM

Bidirectional RNN is another variation of RNN, designed to train the network using input data sequences in the past and future. Two associated layers are utilized to process the input data. In a reverse time step direction, each layer performs its operations. The outcomes would then be able to be joined utilizing various sorts of combining techniques such as adding, multiplying, concatenating, and average. Also, Bidirectional LSTM utilizes two layers to such an extent that one layer executes the operations following the same direction of the data sequence, and the other layer takes a reverse course to run based on the data sequence it receives, as illustrated in Figure3.2. From experiments carried out, BLSTM has been discovered to be more proficient than unidirectional LSTM in some applications. Unidirectional LSTM goes in only one direction (Althelaya et al., 2018). According to Hossain et al. (2018), applying the Bidirectional Gated Recurrent Unit (BGRU) for prediction of stock prices could achieve a sixty percent accuracy using the S & P 500 index.

3.4.4 Autoencoders

According to Pereira, Junior, & Caloba (2018), autoencoders are shallow Feed Forward Neural Network models in which the anticipated output is just as the input sequence. Similarly, an autoencoder belongs to a class of network (feedforward) whereupon the input is analogous to the output. Case in point, in autoencoders, the input vector is compressed into a lower-dimensional code; it attempts to reconstruct the output from that which has been given (Essien & Giannetti, 2019). According to Pereira, Junior, & Caloba (2018), autoencoders are shallow Feed Forward Neural Network models in which the anticipated output is equivalent to the input. Furthermore, an autoencoder encompasses a class of feedforward networks at which the input is the same as the output. Viz, autoencoders compress the input sequence into a lower-dimensional code as it endeavors to reassemble the output from this given model (Essien & Giannetti, 2019). The magnitude of the hidden layer is constrained to be smaller to reduce the errors of the output during training. Figure 3.4 illustrates a typical example of this architecture. Figure 3.5 shows the visualization. The hidden layer output translates to be a dense form of the actual features. Feeding the resulting compressed features as input to a second autoencoder leads to another compression being made. It reduces the dimensionality of features (Pereira, Junior, & Caloba, 2018). The study of Essien & Giannetti (2019) also states that the elements encoder, the code, and the decoder functions according to what their respective names suggest. The encoder compresses the input to a latent space (representation of compressed data) to generate the code, which is then decoded by the decoder. In other words, the output is reassembled from the code, putting to work the decoder.

According to (Gu, Kelly, & Xiu, 2019), in autoencoders, hidden layers are denoted as L and are expressed as a recursive formula. $K^{(l)}$ are neurons represented for each layer l = 1, ..., L. Represent the neuron's output k in layer l as $r_k^{(l)}$, and the vector of the entire outputs as $r^{(l)} = (r_1^{(l)}, \ldots, r_{K^{(l)}}^{(l)})$.

The inputs from the earlier layers are encoded according to an activation function that is nonlinear g(·); it is sent to the layer that follows to initialize the network expressed in Eqn 3.4, the cross-section of returns as the hidden layer employs raw predictors, $r^{(0)} = r = (r_1, \dots, r_N)'$. The recursive output formula for the neural network at each neuron in layer $l > 0$ is then

$$r_k^{(l)} = g\left(b^{(l-1)} + r^{(l-1)'} W^{(l-1)}\right) \tag{3.4}$$

with final output

$$G(r, b, W) = b^{(L-1)} + r^{(L-1)'} W^{(L-1)} \tag{3.5}$$

Where $W^{(l-1)}$ is a $K^{(l)} \times K^{(l-1)}$ matrix of weight parameters, and $b^{(l-1)}$ is a $K^{(l)} \times 1$ vector of so-called bias parameters. The rectified linear unit$(ReLU), g(y) = \max(y, 0)$. The parameters for each hidden layer l is$K^{(l)}(1 + K^{(l-1)})$, plus another $1 + K^{(L-1)}$ weights for the output layer. In summary, an autoencoder employs G(r, b, W) to approximate r itself, as shown in Eqn 3.5.

3.5 PERFORMANCE METRICS

The training of the network is based on the mean square error loss as the stock price randomly changes. The resulted difference between the forecasted value and the real value is the Euclidean distance for a particular symbol class (Hossain et al., 2018). The elucidation of the loss function is shown in Eqn 3.6:

$$L(\Theta) = \frac{1}{N} \sum_{i=1}^{N} (F(X_i; \Theta) - Y_i)^2 \tag{3.6}$$

Where Θ is a collection of learnable input arguments in the hybrid method proposed, N is the vector which comprises of the training data. X_i is the ith input that is the initial prices to be predicted, and Y_i is the real price for X_i. F(X_i; Θ) denotes the expected price generated by the model proposed, which is parameterized with Θ for sample X_i. L(Θ)is the loss between the expected price and real prices.

$$MAE = \frac{1}{N}\sum_{t=1}^{n} |y_t - y_t| \tag{3.7}$$

$$MSE = \frac{1}{N}\sum_{n=1}^{N} (y_t - y_t)^2 \tag{3.8}$$

Where $\hat{y}_t$ represents the predicted value, and y_t represents the real value for the t_{th} day.

3.6 BATCH SIZES

Deep neural network training conventionally is based on stochastic gradient optimization (mini-batch). Splitting the training set into batches allows the CPU/GPU cores to train on different batches in parallel. This gives an incredible speed boost (Masters & Luschi, 2018). According to Radiuk, (2018), SGD and its variants are employed in a small-batch regime, where $|B| \in X$ and typically $|B| \in \{16,32, \dots ,512\}$.

3.7 PROCESS FLOW OF THE PROPOSED MODELS

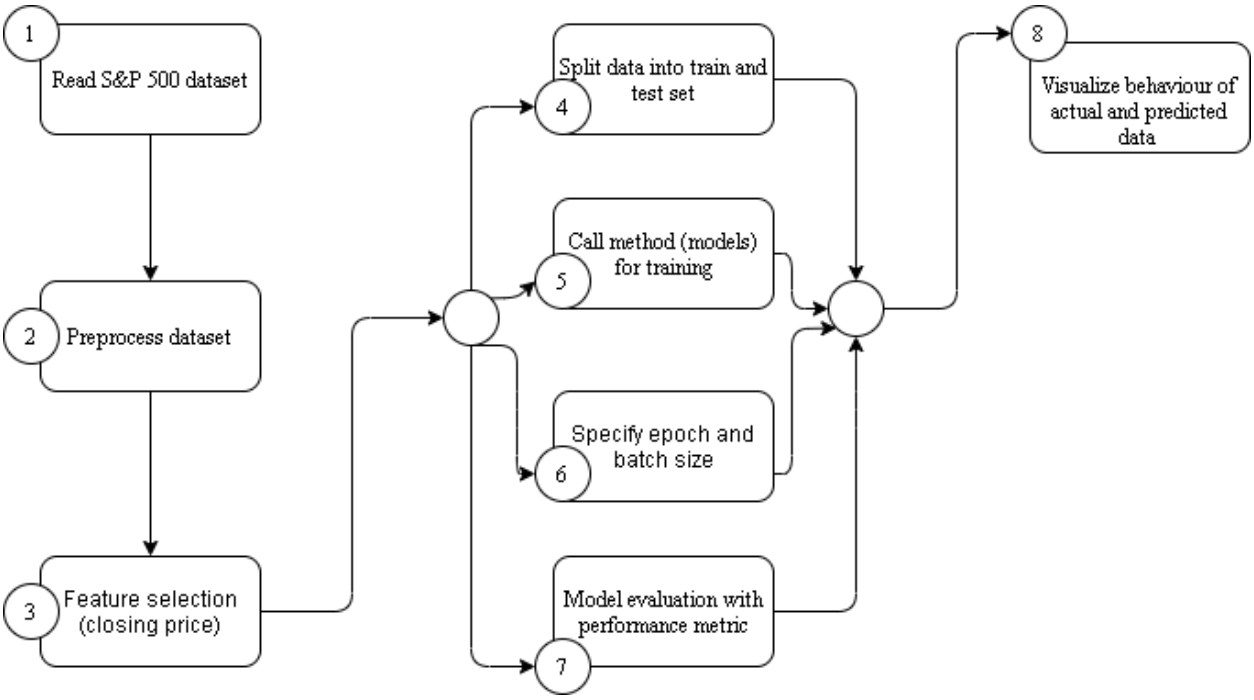


Figure 3. 7: The process flow of the proposed models

The process flow, as illustrated in Figure3.7, is the processes taken for the prediction of stock prices. The first step involved reading the dataset for the training. The second was to preprocess the data, and this was to transform the data into a state that machine can parse it; it scaled in a range of zero to one using the MinMax scalar as shown in Eqn 3.1. The third process was to select the closing price feature as the target variable for the prediction. The fourth process involved splitting the data into training and testing dataset, where 80% of the dataset was used for training, and 20% was used for the testing. The fifth process was calling the function (of a specific model) for training on the data. The sixth process specified the number of epoch (cycle) and the batch size

to determine the best accurate point. In the seventh process, the model is evaluated. MAE and MSE performance metrics were used, as shown in Eqn 3.7 and Eqn 3.8, respectively. The last process gave a visualization of the actual and the predicted data.

3.8 LSTM, GRU& AUTOENCODER MODEL (PROPOSED I)

$$
\begin{aligned}
i_t &= \sigma\left(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)}\right)\\
f_t &= \sigma\left(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}\right)\\
o_t &= \sigma\left(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}\right)\\
u_t &= \tanh\left(W^{(u)}x_t + U^{(u)}h_{t-1} + b^{(u)}\right)\\
c_t &= i_t \odot u_t + f_t \odot c_{t-1}\\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}
$$

$$
\begin{aligned}
z_t &= \sigma\left(W^{(z)}x_t + U^{(z)}h_{t-1} + b^{(z)}\right)\\
r_t &= \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1} + b^{(r)}\right)\\
a_t &= \tanh\left(Wx_t + r_t \odot Uh_{t-1} + b^{(h)}\right)\\
h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot a_t
\end{aligned}
\qquad (3.9)
$$

$$
r_k^{(l)} = g\left(b^{(l-1)} + r^{(l-1)'}W^{(l-1)}\right)
$$

$$
G(r, b, W) = b^{(L-1)} + r^{(L-1)'}W^{(L-1)}
$$

Where time step is denoted as t, i_t is represented as the input gate. The f_t is identified as the forget gate. The o_t is denoted as the output gate. The c_t identifies as the cell. The u_t is a function for activation, the hidden state is defined as h_t. The weights are *W*, and *U*.

Where $W^{(l-1)}$ is a $K^{(l)} \times K^{(l-1)}$ matrix of weight parameters, and $b^{(l-1)}$ is a $K^{(l)} \times 1$ vector of so-called bias parameters. The rectified linear unit$(ReLU), g(y) = \max(y, 0)$. The parameters for each hidden layer l is$K^{(l)}(1 + K^{(l-1)})$, plus another $1 + K^{(L-1)}$ weights for the output layer. In summary, an autoencoder employs G(r, b, W) to approximate r itself

3.9 BIDIRECTIONAL LSTM & BIDIRECTIONAL GRU MODEL (PROPOSED II)

$$i_t = \sigma\left(W^{(i)}x_t + W^{(i)}h_{t-1} + b^{(i)}\right)$$
$$f_t = \sigma\left(W^{(f)}x_t + W^{(f)}h_{t-1} + b^{(f)}\right)$$
$$o_t = \sigma\left(W^{(o)}x_t + W^{(o)}h_{t-1} + b^{(o)}\right)$$
$$u_t = \tanh\left(W^{(u)}x_t + W^{(u)}h_{t-1} + b^{(u)}\right)$$

$$\vec{h}_t = \tanh\left(W_{x\vec{h}}x_t + W_{\vec{h}\vec{h}}\vec{h}_{t-1} + b_{\vec{h}}\right)$$

$$\overleftarrow{h}_t = tanh\left(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t-1} + b_{\overleftarrow{h}}\right)$$

$$y_t = W_{\vec{h}y}\vec{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t + b_y \qquad (3.10)$$

$$c_t = i_t \odot u_t + f_t \odot c_{t-1} + y_t$$

$$h_t = o_t \odot \tanh(c_t) + y_t$$

$$z_t = \sigma\left(W^{(z)}x_t + W^{(z)}h_{t-1} + b^{(z)}\right)$$
$$r_t = \sigma\left(W^{(r)}x_t + W^{(r)}h_{t-1} + b^{(r)}\right)$$
$$a_t = \tanh\left(Wx_t + r_t \odot Wh_{t-1} + b^{(h)}\right)$$

$$\vec{h}_t = tanh\left(W_{x\vec{h}}x_t + W_{\vec{h}\vec{h}}\vec{h}_{t-1} + b_{\vec{h}}\right)$$

$$\overleftarrow{h}_t = tanh\left(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t-1} + b_{\overleftarrow{h}}\right)$$

$$y_t = W_{\vec{h}y}\vec{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t + b_y$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot a_t + y_t$$

Where time step is denoted as t, i_t is represented as the input gate. The f_t is identified as the forget gate. The o_t is denoted as the output gate. The c_t identifies as the cell. The u_t is a function for activation, the hidden state is defined as h_t. The weights are *W,* and *U*. The past context and future context of a specific list of elements (sequence). It is made of two separate hidden layers; it first computes the hidden forward sequence $\vec{h}_t$; then, it computes the backward hidden sequence $\overleftarrow{h}_t$; finally, it combines $\vec{h}_t$ and $\overleftarrow{h}_t$ to generate the sum (output) y_t.

Chapter 4 – Data Analysis and Results

4.1 INTRODUCTION

This section discusses the results of the various techniques employed. It shows the observations after making modifications to the recurrent layers, epochs, and batch size. The first function is the LSTM function, which uses only the LSTM layers for prediction. The second function is the GRU function, which uses only the GRU layers for prediction. The third function combines the LSTM and GRU layers, where the LSTM passes its output to the GRU layer for the final prediction. Similarly, the fourth function includes the LSTM and GRU layer, but the LSTM layer was made Bidirectional. In other words, the function (Python) is a Bidirectional LSTM layer and the GRU layer making the final prediction. The next function, the proposed model, comprises the LSTM, GRU, and Autoencoder Recurrent Neural Network. The LSTM output is shifted to the GRU layer and, lastly, moved to the Autoencoder for the final prediction. Last but not least is the second proposed model, which is the bidirectional LSTM & bidirectional GRU; similarly, LSTM passes its output to the GRU for the final prediction. There is a discussion of an article with a similar objective employing the same dataset, which leads to a comparative study between the models reviewed and the proposed models in this section. The technology to be applied is PYTHON. Python 3.7 language (Anaconda), together with the Keras and Tensorflow library, will be used to run the algorithms; the Adam optimizer will be adopted to optimize the models (Hossain et al., 2018).

4.2 COMPUTATIONAL EXPERIENCE

The recurrent neural network is trained and tested for the forecasting executions in Intel ® Xeon (R) CPU E3-122-v5 processor of three GigaHertz speed and eight GigaByte RAM (Random

Access Memory). Hossain et al. (2018) trained and tested their models on an Intel Core i7 processor with speed four GigaHertz and thirty-two GigaByte RAM. The neural network takes the input sequences and prepares it for random prediction biases as weights will have to be assigned.

4.3 LSTM MODEL ANALYSIS

This section discusses the findings using only LSTM layers (function). The epoch is tweaked several times to observe the Accuracy of the model using the mean squared error metrics. The batch size is set 32 and 64.

4.3.1 Training with batch size 32

The observation after training recorded a score (test) of 0.00001509 MSE (0.00388481 RMSE), and the duration indicated a test time of 3.327608585357666, as illustrated in Figure 4.1. The observation after training recorded a score (test) of 0.00006038 MSE (0.00777051 RMSE), and the duration indicated a test time of 3.763849973678589, as illustrated in Figure 4.2. The observation after training recorded a score (test) of 0.00000161 MSE (0.00126843 RMSE), and the duration indicated a test time of 5.173045873641968, as illustrated in Figure 4.3. The observation after training recorded a score (test) of 0.00000039 MSE (0.00062648 RMSE), and the duration indicated a test time of 4.242125511169434, as illustrated in Figure 4.4. After 20 epochs, the errors rose.

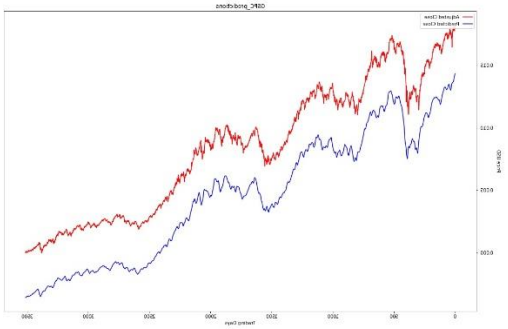


Figure 4. 1: Epoch 3 and batch size 32

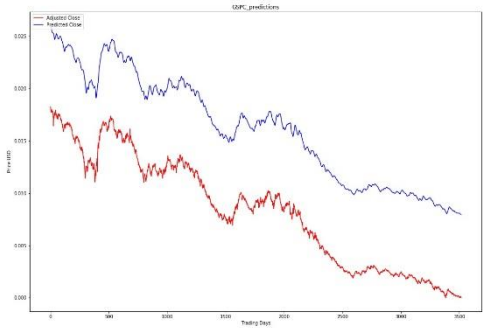


Figure 4. 2: Epoch 5 and batch size 32

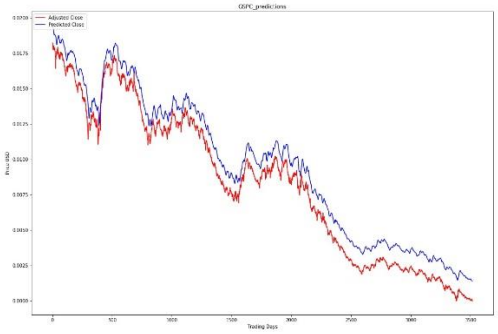


Figure 4. 3: Epoch 10 and batch size 32

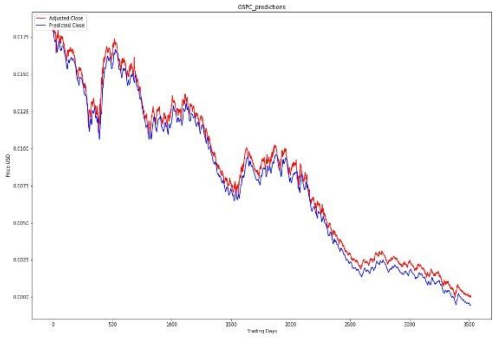


Figure 4. 4: Epoch 15 and batch size 32

4.3.2 Training with batch size 64

The observation after training recorded a score (test) of 0.00000044 MSE (0.00066559 RMSE), and the test time was 1.0312933921813965, as illustrated in Figure4.5. The observation after training recorded a score (test) of 0.00001143 MSE (0.00338124 RMSE), and the test time was 1.2031669616699219, as illustrated in Figure4.6. The observation after training recorded a score (test) of 0.00000031 MSE (0.00055519 RMSE), and the test time was 0.9844167232513428, as illustrated in Figure4.7. The observation after training recorded a score (test) of 0.00000168 MSE (0.00129452 RMSE), and the test time was 1.7657201290130615, as illustrated in Figure 4.8.

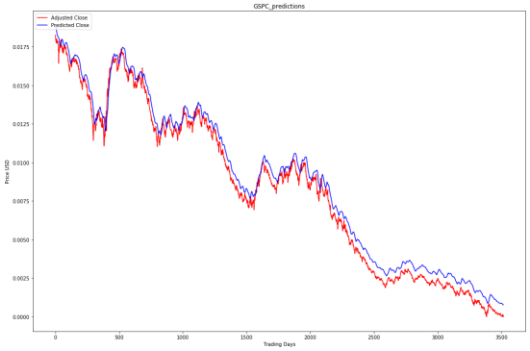


Figure 4. 5: Epoch 3 and batch size 64

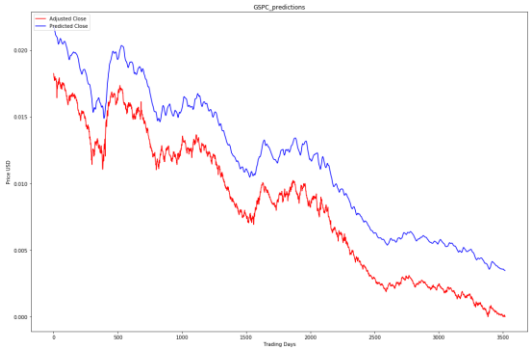


Figure 4. 6: Epoch 5 and batch size 64

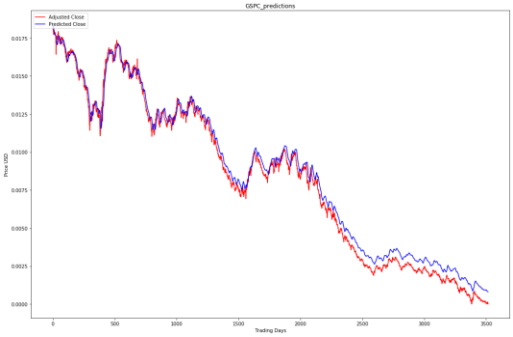


Figure 4. 7: Epoch 5 and batch size 64

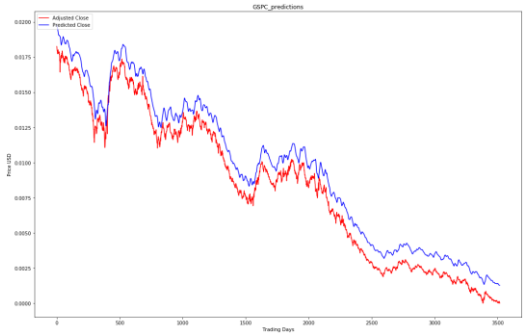


Figure 4. 8: Epoch 5 and batch size 64

4.4 GRU MODEL ANALYSIS

This section discusses the findings using only GRU layers (function). The epoch is tweaked several to observe the Accuracy of the model using the mean squared error metrics.

4.4.1 Training with batch size 32

The observation after training recorded a score (test) of 0.00000807 MSE (0.00284000 RMSE), and the test time was 5.256996154785156, as illustrated in Figure4.9. The observation after training recorded a score (test) of0.00000484 MSE (0.00220022 RMSE), and the test time was

4.241576671600342, as illustrated in Figure4.10. The observation after training recorded a score (test) of 0.00000158 MSE (0.00125513 RMSE), and the duration indicated a test time of6.262423038482666, as illustrated in Figure4.11. The observation after training recorded a score (test) of 0.00002140 MSE (0.00462575 RMSE), and the duration indicated a test time of5.211022138595581, as illustrated in Figure4.12. The observation after training recorded a score (test) of 0.00000192 MSE (0.00138480 RMSE), and the duration indicated a test time of7.371108531951904 illustrated in Figure 4.13.

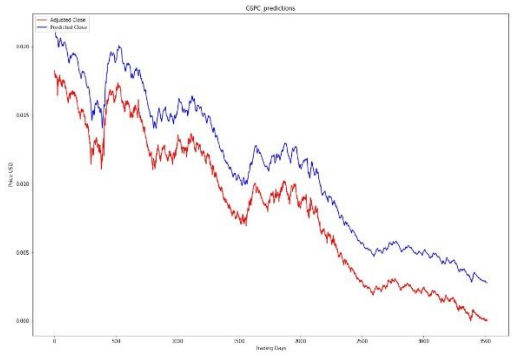


Figure 4. 9: Epoch 3 and batch size 32

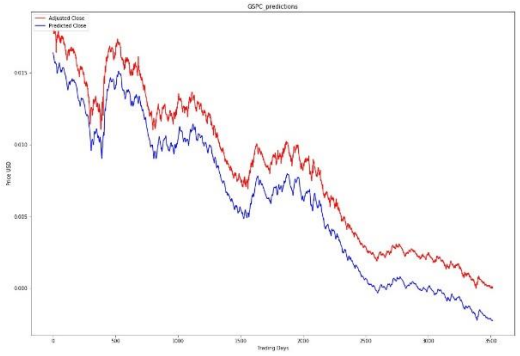


Figure 4. 10: Epoch 5 and batch size 32

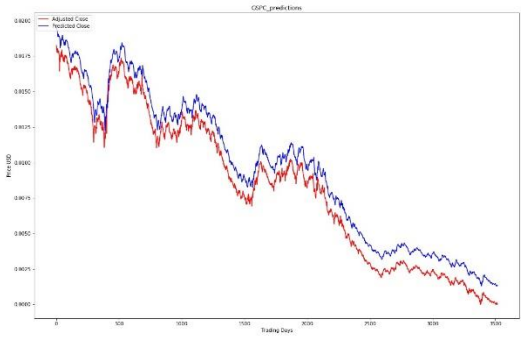


Figure 4. 5: Epoch 20 and batch size 32

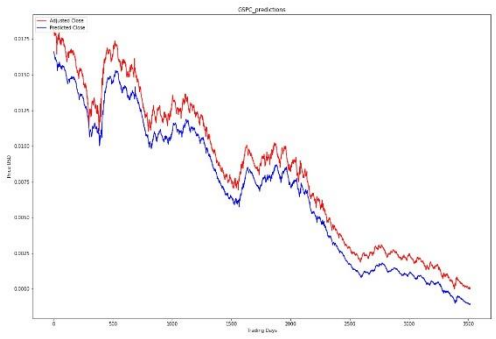


Figure 4.13: Epoch 50 and batch size 32

4.4.2 Training with batch size 64

The observation after training recorded a score (test) of 0.00000459 MSE (0.00214307 RMSE), and the test time was 0.9531586170196533, as illustrated in Figure4.14. The observation after training recorded a score (test) of 0.00000257 MSE (0.00160210 RMSE), and the test time was 0.9062821865081787, as illustrated in Figure4.14. The observation after training recorded a score (test) of 0.00000011 MSE (0.00033244 RMSE), and the test time was 1.2812957763671875, as illustrated in Figure4.16. The observation after training recorded a score (test) of 0.00000526 MSE (0.00229245 RMSE), and the test time was 1.2969143390655518, as illustrated in Figure4.17.

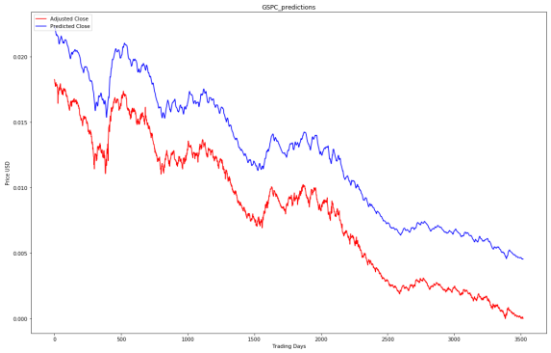


Figure 4.14: Epoch 3 and batch size 64

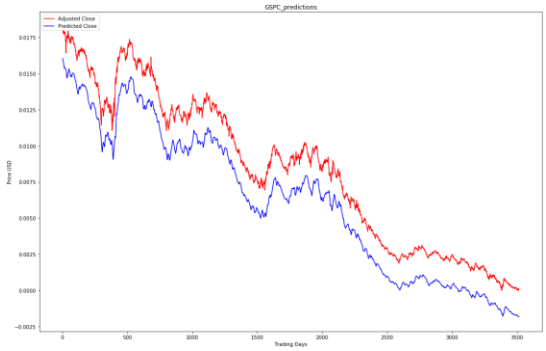


Figure 4.15: Epoch 5 and batch size 64

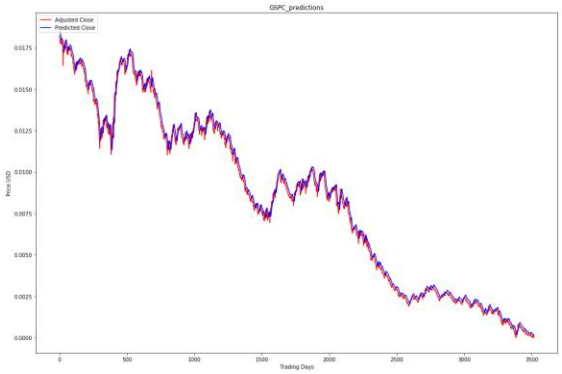


Figure 4.16: Epoch 15 and batch size 64

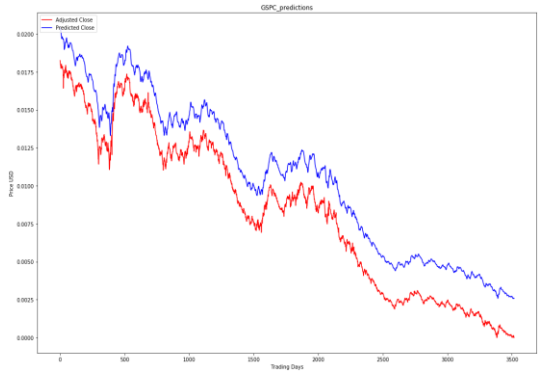


Figure 4.17: Epoch 15 and batch size 64

4.5 BIDIRECTIONAL LSTM MODEL ANALYSIS

This section discusses the findings using only bidirectional LSTM layers (function). The epoch is tweaked several to observe the Accuracy of the model using the mean squared error metrics.

4.5.1 Training with batch size 32

The observation after training recorded a score (test) of 0.00000440 MSE (0.00209655 RMSE), and the test time was 1.9688267707824707, as illustrated in Figure4.18. The observation after training recorded a score (test) of 0.00000215 MSE (0.00146604 RMSE), and the test time was 2.015683889389038, as illustrated in Figure4.19. The observation after training recorded a score (test) of 0.00001041 MSE (0.00322695 RMSE), and the test time was 2.7350988388061523, as illustrated in Figure4.20. The observation after training recorded a score (test) of 0.00001800 MSE (0.00424258 RMSE), and the test time was 2.921903371810913, as illustrated in Figure 4.21.

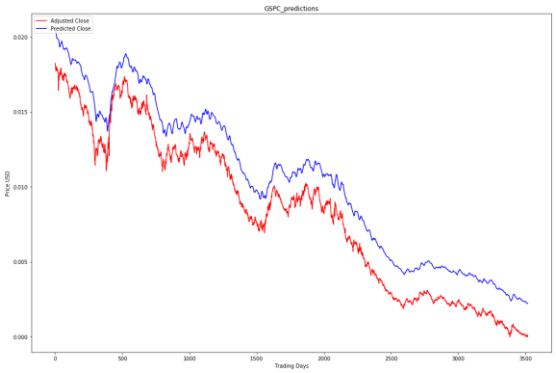


Figure 4.18: Epoch 5 and batch size 32

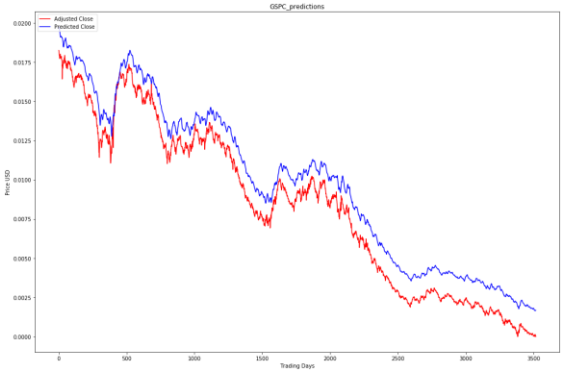


Figure 4.19: Epoch 10 and batch size 32

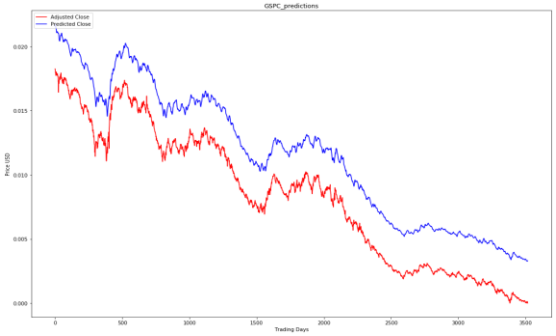


Figure 4.20: Epoch 15 and batch size 32

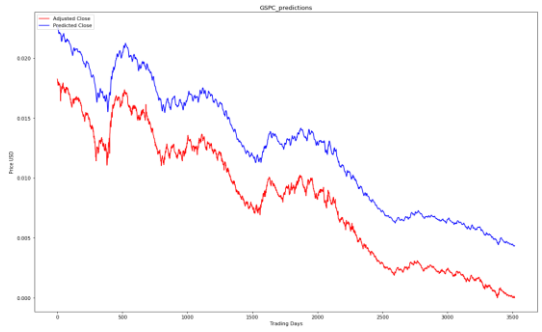


Figure 4.21: Epoch 20 and batch size 32

4.5.2 Training with batch size 64

The observation after training recorded a score (test) of 0.00000142 MSE (0.00119226 RMSE), and the test time was 1.7344341278076172, as illustrated in Figure4.22. The observation after training recorded a score (test) of 0.00001465 MSE (0.00382778 RMSE), and the test time was 1.5156781673431396, as illustrated in Figure4.23. The observation after training recorded a score (test) of 0.00000145 MSE (0.00120322 RMSE), and the test time was 2.236516237258911, as

illustrated in Figure4.24. The observation after training recorded a score (test) of 0.00000025 MSE (0.0005 RMSE), and the test time was 2.4375998973846436, as illustrated in Figure4.25. The errors rose after 11 epochs.

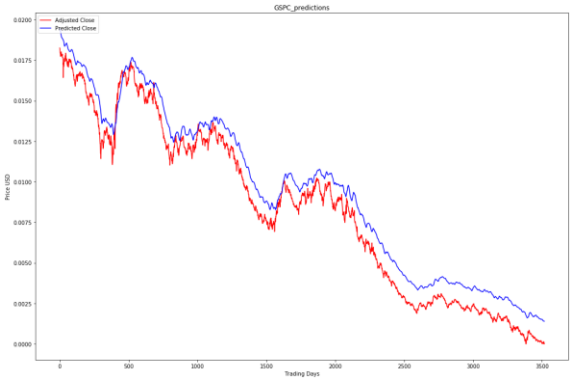


Figure 4.22: Epoch 3 and batch size 64

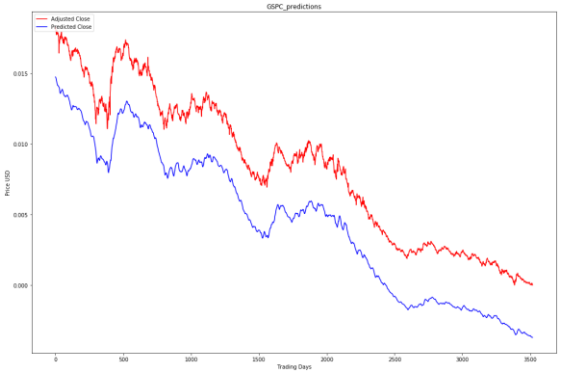


Figure 4.23: Epoch 5 and batch size 64

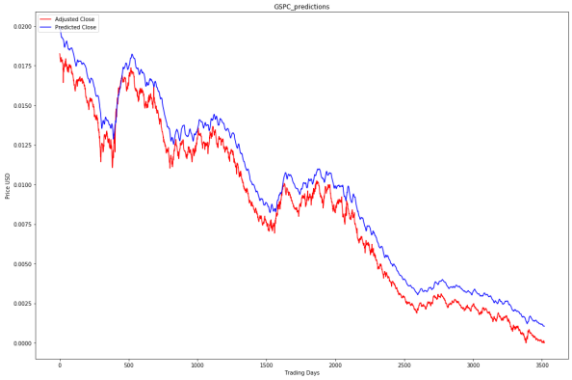


Figure 4.24: Epoch 10 and batch size 64

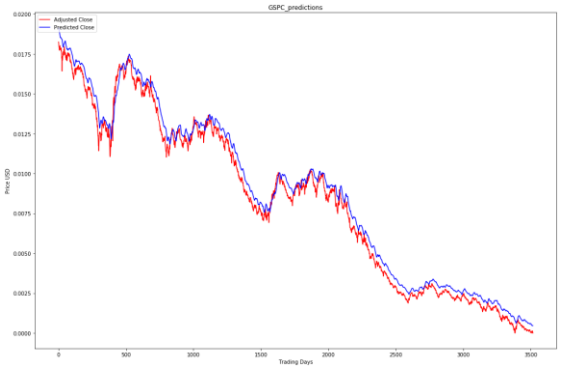


Figure 4.25: Epoch 11 and batch size 64

4.6 BIDIRECTIONAL GRU MODEL ANALYSIS

This section discusses the findings using only bidirectional GRU layers (function). The epoch is tweaked several to observe the Accuracy of the model using the mean squared error metrics.

4.6.1 Training with batch size 32

The observation after training recorded a score (test) of 0.00000016 MSE (0.00040012 RMSE), and the test time was 1.6250579357147217, as illustrated in Figure4.26. The observation after training recorded a score (test) of 0.00000101 MSE (0.00100448 RMSE), and the test time was 1.437558889389038, as illustrated in Figure4.27. The observation after training recorded a score (test) of 0.00001907 MSE (0.00436682 RMSE), and the test time was 2.313839912414551, as illustrated in Figure4.28. The observation after training recorded a score (test) of 0.00000485 MSE (0.00220227 RMSE), and the test time was 2.5000596046447754, as illustrated in Figure4.29.

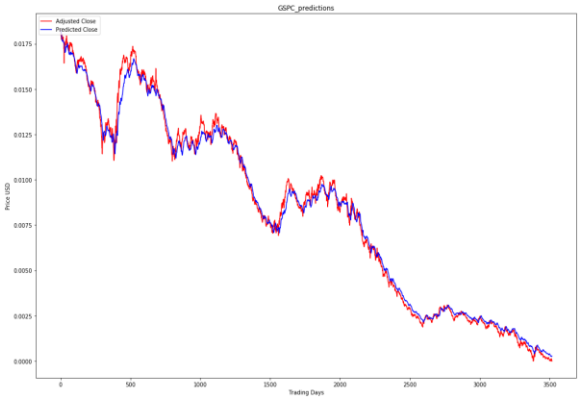


Figure 4.26: Epoch 3 and batch size 32

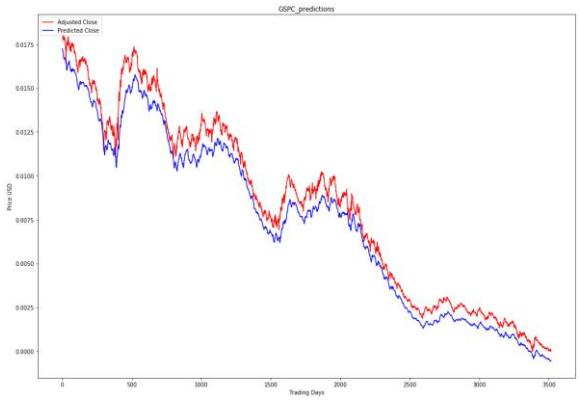


Figure 4.27: Epoch 4 and batch size 32

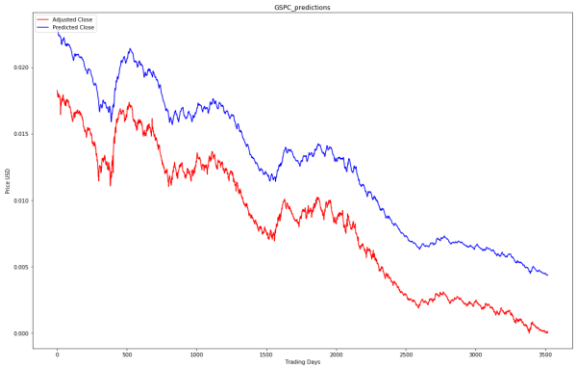


Figure 4.28: Epoch 5 and batch size 32

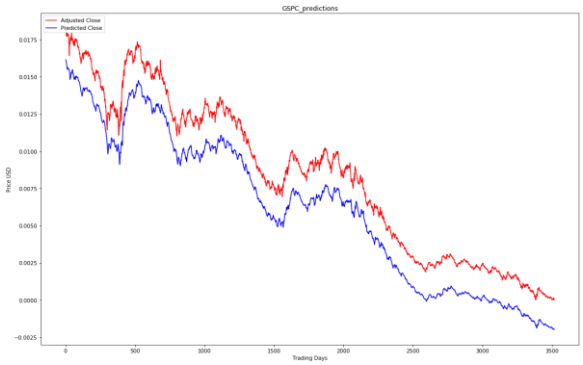


Figure 4.29: Epoch 10 and batch size 32

4.6.2 Training with batch size 64

The observation after training recorded a score (test) of 0.00000305 MSE (0.00174652 RMSE), and the test time was 1.250035047531128, as illustrated in Figure4.30. The observation after training recorded a score (test) of 0.00000016 MSE (0.00040017 RMSE), and the test time was 1.3888781070709229, as illustrated in Figure4.31. The observation after training recorded a score (test) of 0.00000011 MSE (0.00033166 RMSE), and the test time was 2.272493600845337, as illustrated in Figure4.32. The observation after training recorded a score (test) of 0.00000014 MSE (0.00037198 RMSE), and the test time was 2.156341314315796, as illustrated in Figure4.33.

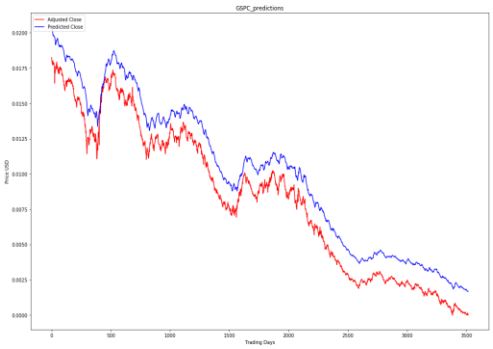


Figure 4.30: Epoch 3 and batch size 64

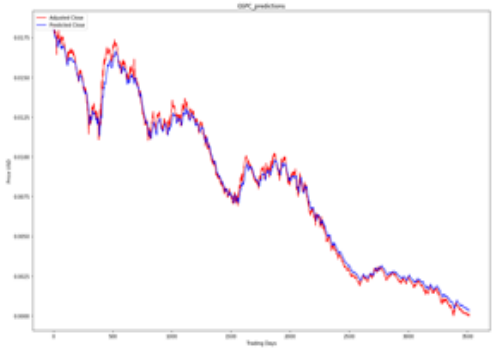


Figure 4.31: Epoch 10 and batch size 64

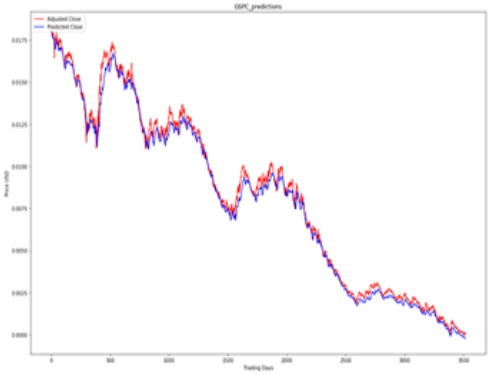


Figure 4.32: Epoch 11 and batch size 64

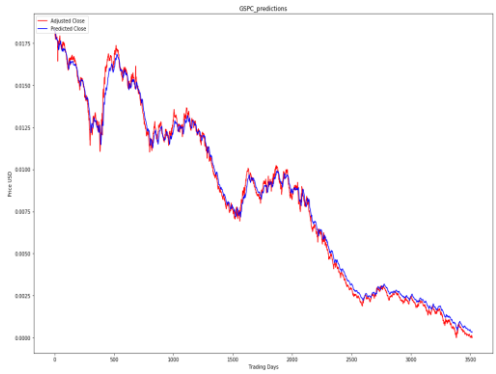


Figure 4.33: Epoch 12 and batch size 64

4.7 LSTM & GRU MODEL ANALYSIS

This section discusses the findings using only LSTM & GRU layers (function). The epoch is tweaked several to observe the Accuracy of the model using the mean squared error metrics.

4.7.1 Training with batch size 32

The observation after training recorded a score (test) of 0.00000186 MSE (0.00136563 RMSE), and the test time was 6.289473533630371. The observation after training recorded a score (test) of0.00005106 MSE (0.00714530 RMSE and the test time was 6.285409450531006. The

observation after training recorded a score (test) of 0.00000077 MSE (0.00087749 RMSE), and the duration indicated a test time of6.629211187362671. The observation after training recorded a score (test) of 0.00000023 MSE (0.00047958 RMSE), and the duration indicated a test time of8.620076894760132.

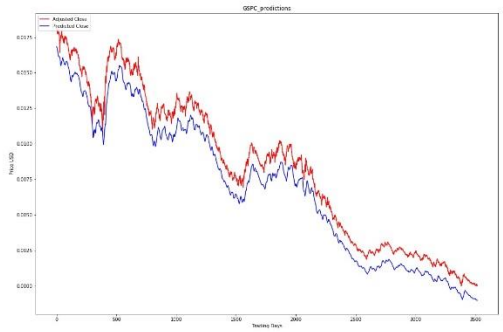


Figure 4. 34: Epoch 3 and batch size 32

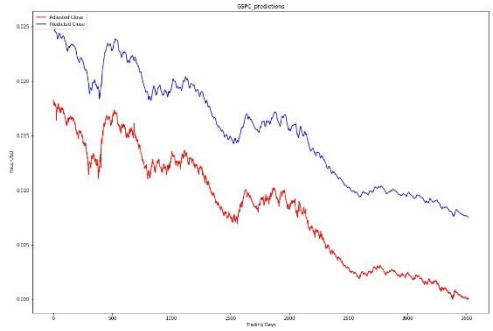


Figure 4. 35: Epoch 5 and batch size 32

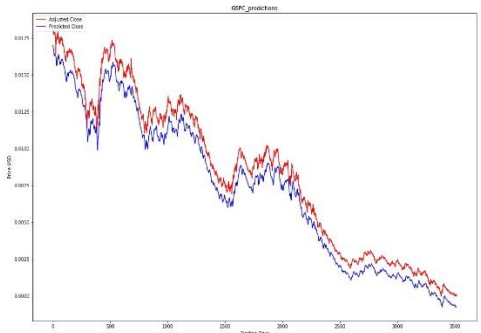


Figure 4. 36: Epoch 30 and batch size 32

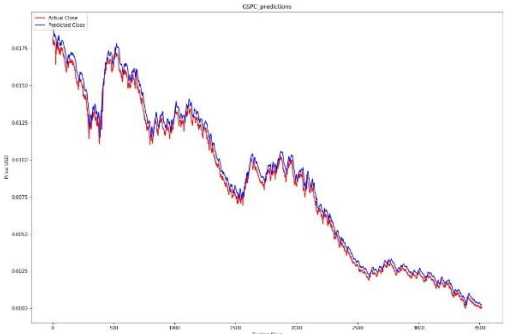


Figure 4. 37: Epoch 50 and batch size 32

4.7.2 Training with batch size 64

The observation after training recorded a score (test) of 0.00000559 MSE (0.00236389 RMSE), and the test time was 0.7344040870666504. The observation after training recorded a score (test) of 0.00001715 MSE (0.00414175 RMSE and the test time was 1.0312871932983398. The observation after training recorded a score (test) of 0.00000276 MSE (0.00166000 RMSE), and the duration indicated a test time of1.3679358959197998. The observation after training recorded a score (test) of 0.00004241 MSE (0.00651198 RMSE), and the duration indicated a test time of1.3906476497650146.

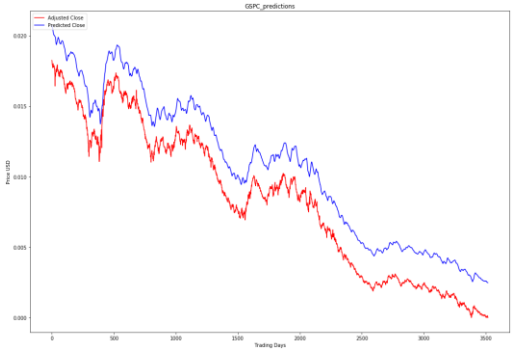


Figure 4. 38: Epoch 3 and batch size 64

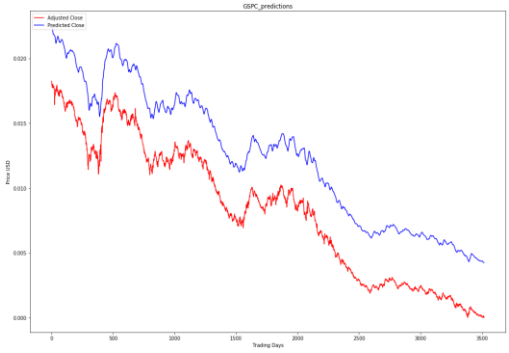


Figure 4. 39: Epoch 5 and batch size 64

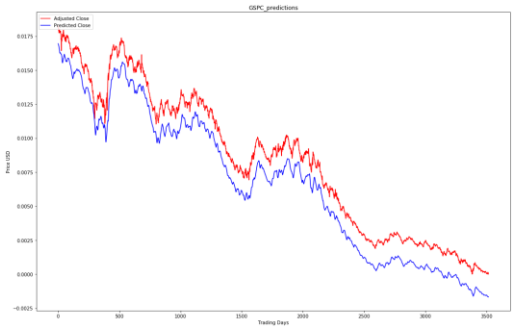


Figure 4. 40: Epoch 10 and batch size 64

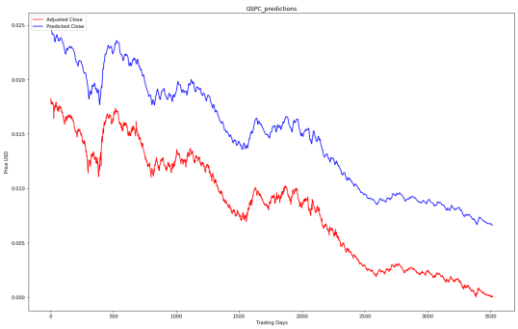


Figure 4. 41: Epoch 15 and batch size 64

4.8 BIDIRECTIONAL LSTM & GRU MODEL ANALYSIS

This section discusses the findings using only bidirectional LSTM & GRU layers (function). The epoch is tweaked several to observe the Accuracy of the model using the mean squared error metrics.

4.8.1 Training with batch size 32

The observation after training recorded a score (test) of 0.00000020 MSE (0.00044721 RMSE), and the test time was 1.0937883853912354. The observation after training recorded a score (test) of 0.00000870 MSE (0.00294895 RMSE), and the test time was 1.4062988758087158. The observation after training recorded a score (test) of 0.00000034 MSE (0.00058677 RMSE), and the test time was 2.053959608078003. The observation after training recorded a score (test) of 0.00001171 MSE (0.00342153 RMSE), and the test time was 2.125088930130005.

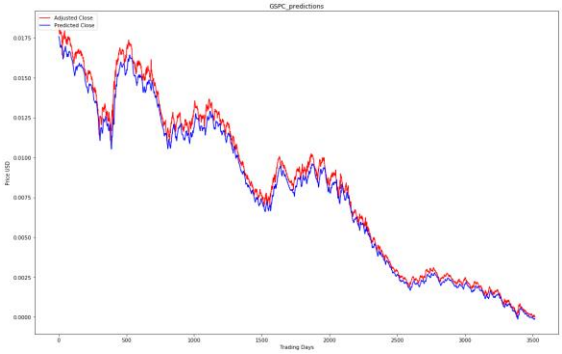


Figure 4. 42: Epoch 3 and batch size 32

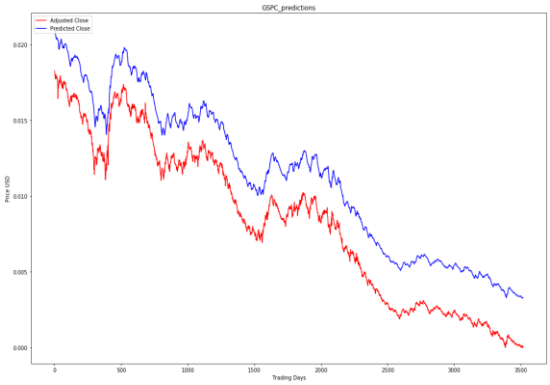


Figure 4. 43: Epoch 4 and batch size 32

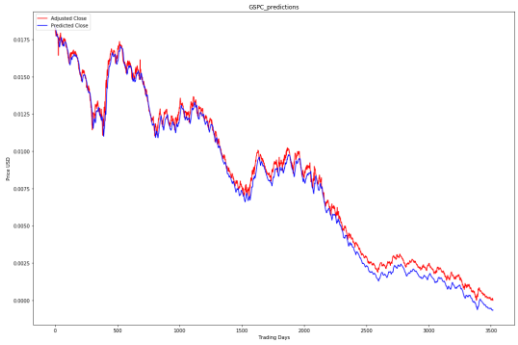


Figure 4. 44: Epoch 10 and batch size 32

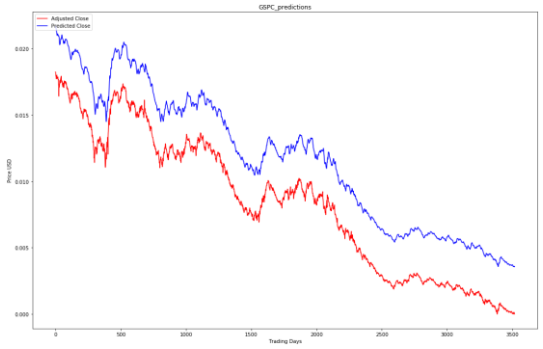


Figure 4. 45: Epoch 11 and batch size 32

4.8.2 Training with batch size 64

The observation after training recorded a score (test) of 0.00001005 MSE (0.00317069 RMSE), and the test time was 1.3781583309173584. The observation after training recorded a score (test) of 0.00000559 MSE (0.00236389 RMSE), and the test time was 0.7344040870666504. The observation after training recorded a score (test) of 0.00001077 MSE (0.00328115 RMSE), and

the test time was 1.5347676277160645. The observation after training recorded a score (test) of 0.00000328 MSE (0.00181034 RMSE), and the test time was 1.9688258171081543.

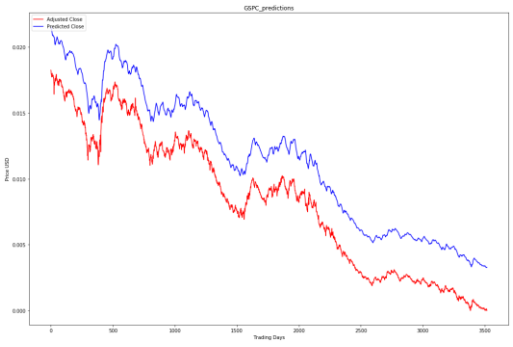


Figure 4. 46: Epoch 3 and batch size 64

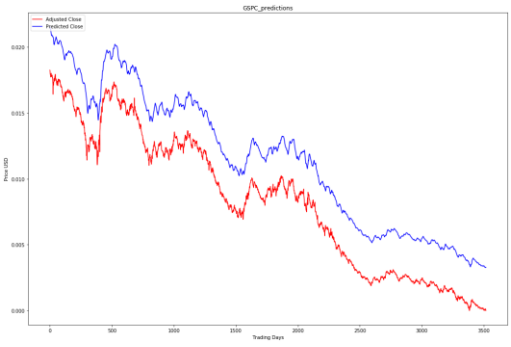


Figure 4. 47: Epoch 5 and batch size 64

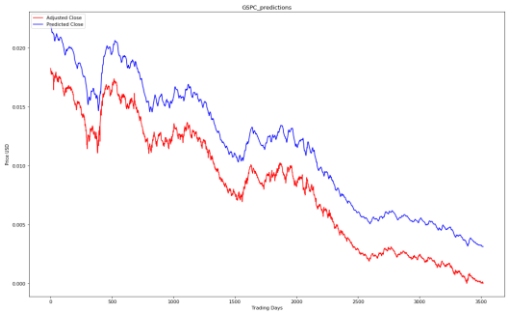


Figure 4. 48: Epoch 10 and batch size 64

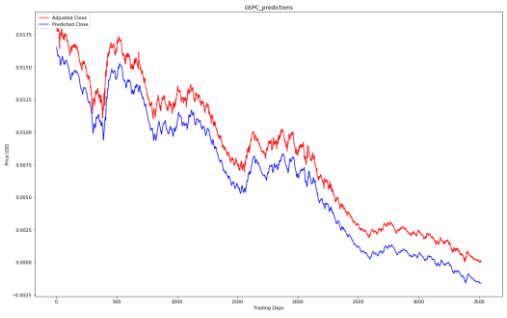


Figure 4. 49: Epoch 15 and batch size 64

4.9 LSTM & BIDIRECTIONAL GRU MODEL ANALYSIS

This section discusses the findings using only LSTM & bidirectional GRU layers (function). The epoch is tweaked several to observe the Accuracy of the model using the mean squared error metrics.

4.9.1 Training with batch size 32

The observation after training recorded a score (test) of 0.00000902 MSE (0.00300312 RMSE), and the test time was 1.4375572204589844. The observation after training recorded a score (test) of 0.00000142 MSE (0.00119318 RMSE), and the test time was 1.5156829357147217. The observation after training recorded a score (test) of 0.00000162 MSE (0.00127209 RMSE), and the test time was 1.9341442584991455. The observation after training recorded a score (test) of 0.00000157 MSE (0.00125104 RMSE), and the test time was 2.046947956085205.

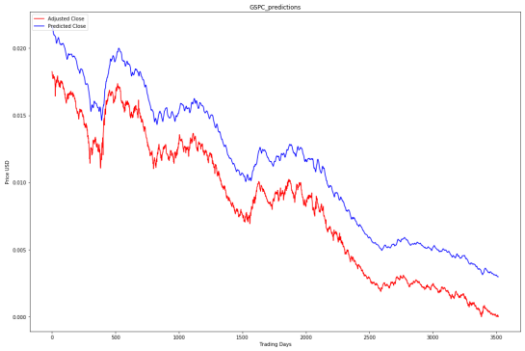


Figure 4. 50: Epoch 5 and batch size 32

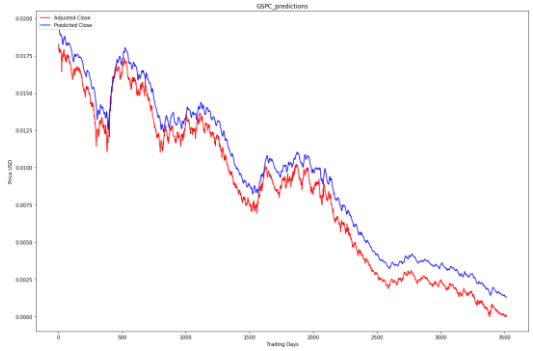


Figure 4. 51: Epoch 10 and batch size 32

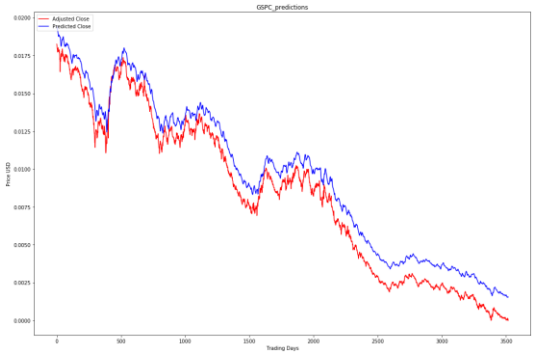


Figure 4. 52: Epoch 16 and batch size 32

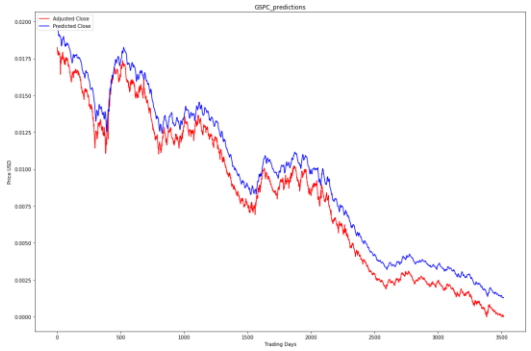


Figure 4. 53: Epoch 17 and batch size 32

4.9.2 Training with batch size 64

The observation after training recorded a score (test) of 0.00000269 MSE (0.00163939 RMSE), and the test time was 1.6223795413970947. The observation after training recorded a score (test) of 0.00000027 MSE (0.00051962 RMSE), and the test time was 1.8480885028839111. The observation after training recorded a score (test) of 0.00000061 MSE (0.00078102 RMSE), and the test time was 1.8032116889953613. The observation after training recorded a score (test) of 0.00001105 MSE (0.00332452 RMSE), and the test time was 2.2344515323638916.

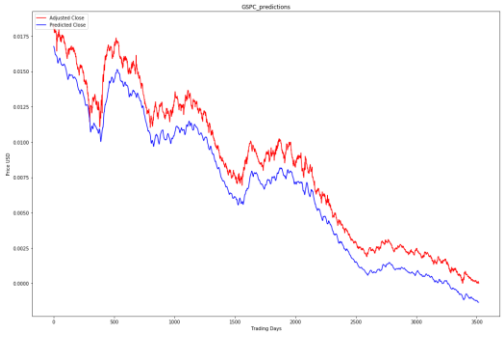


Figure 4. 54: Epoch 5 and batch size 64

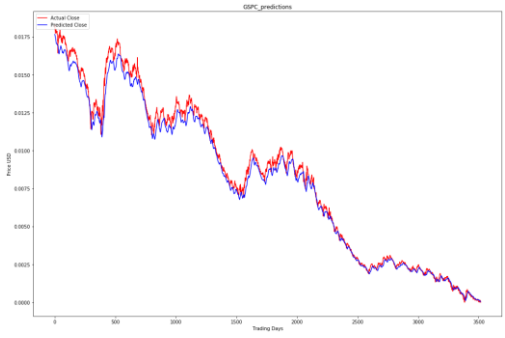


Figure 4. 55: Epoch 10 and batch size 64

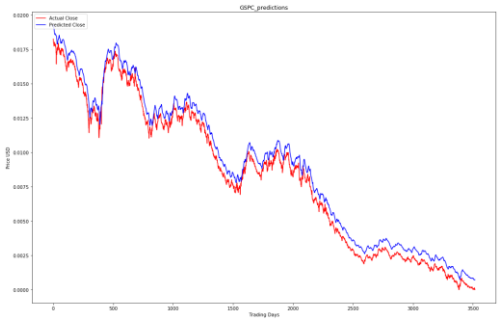


Figure 4. 56: Epoch 11 and batch size 64

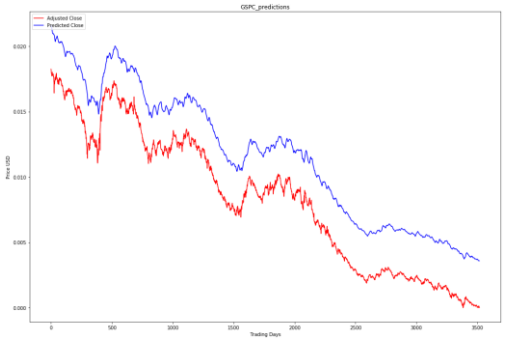


Figure 4. 57: Epoch 12 and batch size 64

4.10 BIDIRECTIONAL LSTM & BIDIRECTIONAL GRU MODEL ANALYSIS (PROPOSED II)

This section discusses the findings using only LSTM & GRU layers (function). The epoch is tweaked several to observe the Accuracy of the model using the mean squared error metrics.

4.10.1 Training with batch size 32

The observation after training recorded a score (test) of 0.00000205 MSE (0.00143250 RMSE), and the test time was 4.370504856109619. The observation after training recorded a score (test) of0.00000008 MSE (0.00028284 RMSE), and the test time was 7.263740539550781. The observation after training recorded a score (test) of0.00000272 MSE (0.00164810 RMSE), and the test time was 4.427470922470093. The observation after training recorded a score (test) of 0.00000062 MSE (0.00078730 RMSE), and the duration indicated a test time of6.846086740493774.

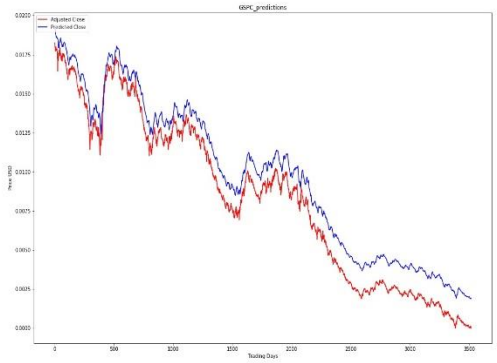


Figure 4.58: Epoch 3 and batch size 32

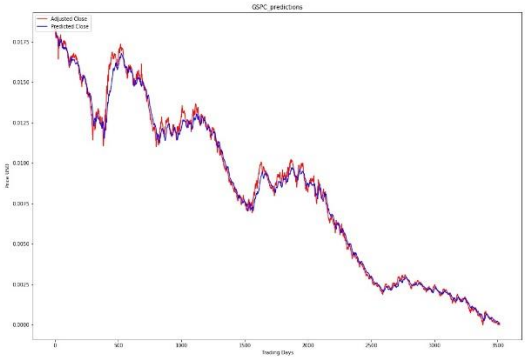
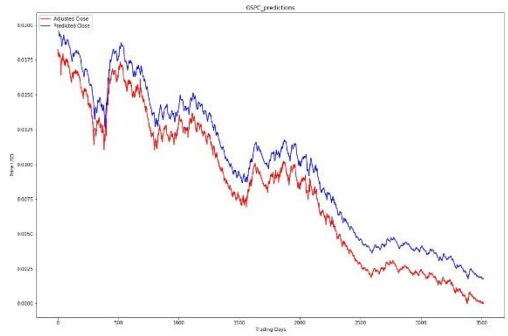


Figure 4.59: Epoch 5 and batch size 32



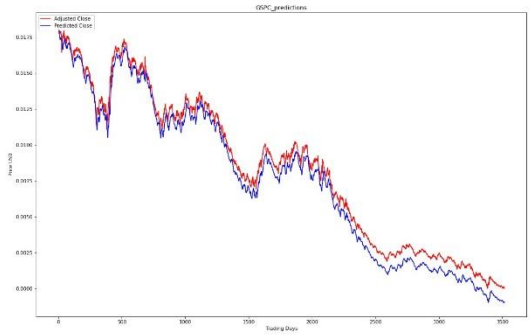


Figure 4.60: Epoch 10 and batch size 32

Figure 4.61: Epoch 50 and batch size 32

4.10.2 Training with batch size 64

The observation after training recorded a score (test) of 0.00000062 MSE (0.00079493 RMSE), and the duration indicated a test time of1.6566245555877686. The observation after training recorded a score (test) of 0.00000154 MSE (0.00124091 RMSE), and the test time was 1.7170436382293701. The observation after training recorded a score (test) of 0.00000955 MSE (0.00309110 RMSE), and the duration indicated a test time of2.3594772815704346. The observation after training recorded a score (test) of 0.00000294 MSE (0.00171543 RMSE), and the duration indicated a test time of3.4779343605041504.

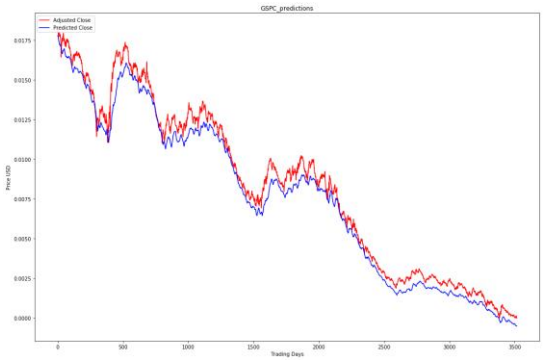


Figure 4.62: Epoch 5 and batch size 64

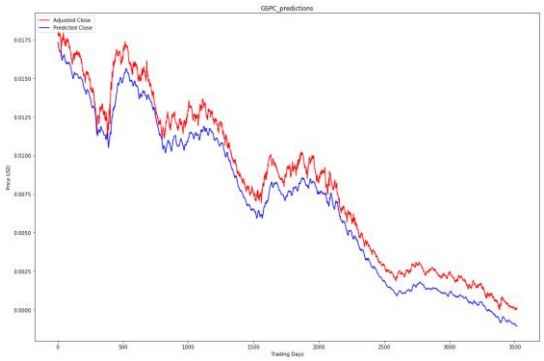


Figure 4.63: Epoch 6 and batch size 64

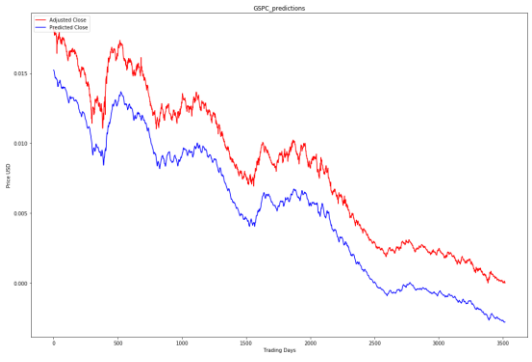


Figure 4.64: Epoch 15 and batch size 64

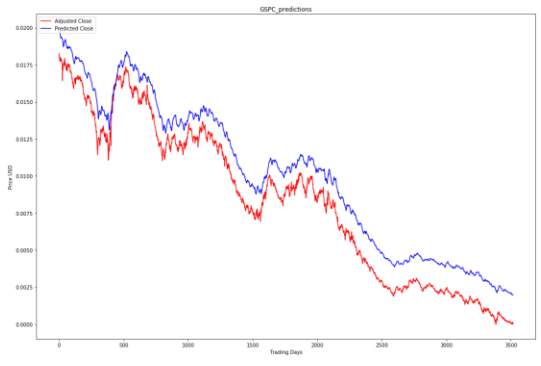


Figure 4.65: Epoch 20 and batch size 64

4.11 LSTM, GRU& AUTOENCODER MODEL ANALYSIS (PROPOSED I)

This section discusses the findings using only LSTM, GRU & AUTOENCODER layers (function). The epoch is tweaked several to observe the Accuracy of the model using the mean squared error metrics.

4.11.1 Training with batch size 32

The observation after training recorded a train loss of 0.22825898 MAE and a test loss of 0.15461363 MAE, as illustrated in Figure4.30. The observation after training recorded a train loss of 0.13314902 MAE and a test loss of 0.18256186 MAE, as illustrated in Figure4.31. The observation after training recorded a recorded train loss of 0.16936007 MAE and a test loss of 0.36932096 MAE as illustrated in Figure4.32. The observation after training recorded a recorded train loss of 0.13422944 MAE and a test loss of 0.14219181 MAE, as illustrated in Figure4.33.

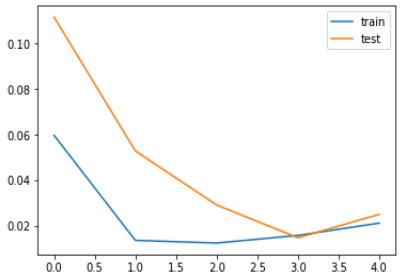


Figure 4.66: Epoch 5 and batch size 32

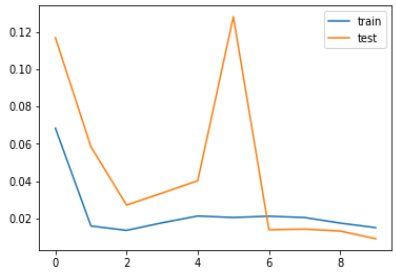


Figure 4.67: Epoch 10 and batch size 32

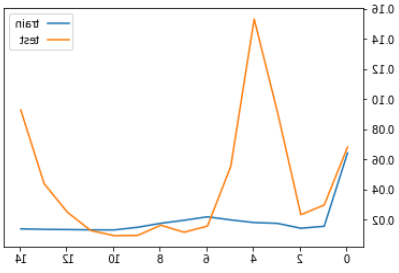


Figure 4.68: Epoch 15 and batch size 32

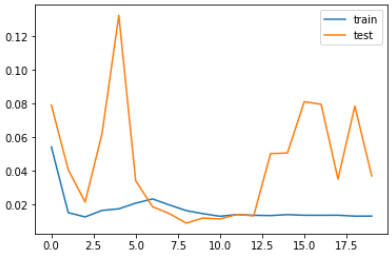


Figure 4.69: Epoch 20 and batch size 32

4.11.2 Training with batch size 64

The observation after training recorded a train loss of 0.14277580 MAE and a test loss of 0.24900658 MAE, as illustrated in Figure4.30. The observation after training recorded a train loss of 0.11990753 MAE and a test loss of 0.19246391 MAE, as illustrated in Figure4.31. The observation after training recorded a recorded train loss of 0.13499588 MAE and a test loss of 0.22265223 MAE, as illustrated in Figure4.32. The observation after training recorded a recorded train loss of 0.15386494 MAE and a test loss of 0.41768398 MAE, as illustrated in Figure4.33.

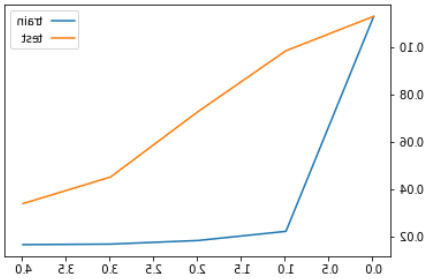


Figure 4. 70: Epoch 5 and batch size 64

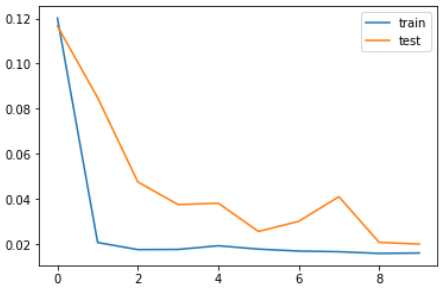


Figure 4. 71: Epoch 10 and batch size 64

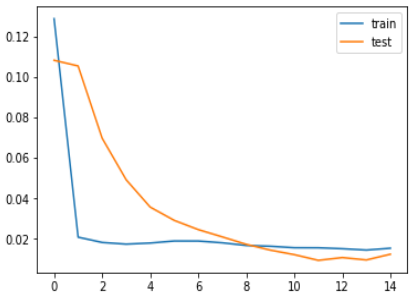


Figure 4. 72: Epoch 15 and batch size 64

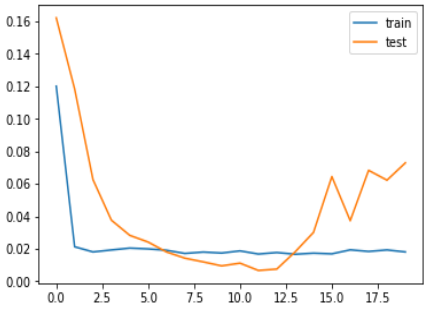


Figure 4. 73: Epoch 20 and batch size 64

4.12 DISCUSSION

The analysis in section 4.3- 4.11 shows that the number of layers and epochs affects predictive modeling performance. This study focused on two layers of the recurrent neural network; the epoch modified ranged from 3, 5, 10, 15, 20, 30 to 50. The batch size was varied from 32 to 64 for analysis. The LSTM model (function) at 15 epochs resulted in 0.00000025 mean squared error (MSE), which was the lowest error recorded among the epoch range specified. The GRUmodel (function) at 10 epochs resulted in 0.00000011 mean squared error (MSE), which was the lowest error recorded among the epoch range specified. The LSTM & GRU model (function) at 50 epochs resulted in 0.00000023 MSE, which was the lowest error recorded among the epoch range specified. The Bidirectional LSTM & Bidirectional GRU model (function) at 5 epochs, recorded 0.00000008 MSE, which was the smallest error recorded among the epoch range specified. Table 4.1 and Table 4.2 gives the details on the proposed and other existing models. The math behind the proposed models is shown in Eqn 3.9 and 3.10 for the first and second proposed models, respectively.

4.13 COMPARATIVE ANALYSIS AND RESULTS

This section shows all the errors recorded after training the models. From the observation, the first proposed model employing the Bidirectional LSTM & GRU model (function) performed than the other models. The second proposed model using LSTM, GRU, and autoencoders did not perform very well enough.

Table 4. 1: Evaluation of techniques

TECHNIQUES	MAE	MSE	RMSE
LSTM&GRU Hossain et al. (2018)	0.023	0.00098	0.03130495
Two LSTM layers	0.00036711	0.00000025	0.0005
Two GRU layers	0.00082809	0.00000077	0.0008775
Bi-directional LSTM layers	0.00030860	0.00000019	0.00043589
Bi-directional GRU layers	0.00027778	0.00000011	0.00033166
LSTM&GRU (one layer each)	0.00040942	0.00000023	0.00047958
Bi-LSTM&GRU (one layer each)	0.00039044	0.00000027	0.00051961
LSTM&Bi-GRU (one layer each)	0.00038123	0.00000020	0.00044721
Bidirectional LSTM& Bidirectional GRU (Proposed model with one layer each)	0.00022110	0.00000008	0.00028284

It is observed that the LSTM&GRU model Hossain et al. (2018) proposed was trained on a dataset ranging from 1950-2016, which recorded an error as low as 0.023 MAE. However, after simulating and retraining the model of Hossain et al. (2018) on data ranging from 1950-2019, an MAE of 0.00040942 was recorded, confirming that neural networks are data-hungry, the more the dataset, the better the performance. The second proposed (Bidirectional LSTM& Bidirectional GRU) model of this study outperformed all the existing models; it recoded an MAE of 0.00022110, which was the lowest recorded. In other observations show that Bi-directional LSTM layers and Bi-directional GRU layers outperformed Two LSTM layers and Two GRU layers, which indicates that including a bidirectional layer has the potential to predict better than models without the bidirectional layer. The Bi-LSTM&GRU (one layer each) and LSTM&Bi-GRU (one layer each) models also indicate similar behaviors (performance) with the bidirectional layer.

Table 4. 2: Evaluation of Autoencoder technique

LSTM,GRU& Autoencoder (Proposed model with one layer each)	Train MAE	Test MAE
Batch 32	0.22825898	0.15461363
Batch 64	0.14277580	0.24900658

Chapter 5 – Conclusion and Recommendation

5.1 CONCLUSION

This study sought to predict stock prices for the next day using the S & P 500 dataset with higher Accuracy and reliability using deep learning techniques (Recurrent Neural Network). This study led to the construction of two proposed models. The first proposed model makes use of the LSTM, GRU, and Autoencoder, where the LSTM's output is passed to the GRU and then to the Autoencoder for the final prediction; it performed poorly. The second proposed model makes use of a Bidirectional LSTM and Bidirectional GRU; the GRU executes the final prediction after the LSTM passes its output to the GRU. The second proposed model, as shown in table 4.1, outperformed the immediate models that were the LSTM model, GRU model, LSTM & GRU model, and other combinations and permutations of the techniques employed. The second proposed model had 0.00000008 MSE, which was the lowest recorded. The LSTM model recorded 0.00000025 MSE, the GRU model recorded 0.00000077 MSE ,and lastly the LSTM & GRU model recorded 0.00000023 MSE. The first proposed model outperformed the study of Hossain et al. (2018) from observation, which had an MSE of 0.00098 as their lowest, as shown in table 4.1. Other existing models such as the Bi-directional LSTM model recorded 0.00000019 MSE, Bi-directional GRU recorded 0.00000011 MSE, Bidirectional LSTM & GRU recorded 0.00000027 MSE, LSTM & Bi-directional GRU recorded 0.00000020 MSE. In general, these techniques have indicated improvement in the precision of forecasts, yielding positive outcomes with the LSTM and GRU model ending up being progressively proficient. The outcomes are very encouraging and have prompted the end that it is conceivable to foresee stock prices with more exactness and proficiency utilizing deep learning methods.

5.2 RECOMMENDATION

Rigorous research is still ongoing in the field of stock price forecast; different methods have experimented with the prediction. Although the results achieved were quite good, there is always more to be done to improve the accuracy of the models further. Using the much bigger dataset in a future study may enhance the forecasting power of the models than the current dataset for stock price prediction. Furthermore, other evolving models in deep learning could likewise be investigated to improve the precision rate. Sentiment analysis through AI in the machine learning field on how stories (news) impact an organization's stock prices is up-and-coming.

Reference

Alameer, Z., Elaziz, M. A., Ewees, A. A., Ye, H., & Jianhua, Z. (2019). Forecasting gold price fluctuations using improved multilayer perceptron neural network and whale optimization algorithm. *Resources Policy*, *61*(January), 250–260. https://doi.org/10.1016/j.resourpol.2019.02.014

Althelaya, K. A., El-Alfy, E. S. M., & Mohammed, S. (2018). Evaluation of Bidirectional LSTM for Short and Long-Term Stock Market Prediction. *2018 9th International Conference on Information and Communication Systems, ICICS 2018*, *2018-Janua*, 151–156. https://doi.org/10.1109/IACS.2018.8355458

Ballings, M., Van Den Poel, D., Hespeels, N., & Gryp, R. (2015). Evaluating multiple classifiers for stock price direction prediction. *Expert Systems with Applications*, *42*(20), 7046–7056. https://doi.org/10.1016/j.eswa.2015.05.013

Billah, M., Waheed, S., & Hanifa, A. (2017). Stock market prediction using an improved training algorithm of neural network. *ICECTE 2016 - 2nd International Conference on Electrical, Computer and Telecommunication Engineering*, (December), 8–10. https://doi.org/10.1109/ICECTE.2016.7879611

Cakra, T. (2015). Stock Price Prediction using Linear Regression based on Sentiment Analysis. *International Journal of Scientific & Engineering Research*, *6*(3), 1655–1659. Retrieved from https://www.ijser.org/researchpaper/Stock-Price-Prediction-Using-Regression-Analysis.pdf

Du, J., Liu, Q., Chen, K., & Wang, J. (2019). Forecasting stock prices in two ways based on LSTM neural network. *Proceedings of 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2019*, (Itnec), 1083–1086. https://doi.org/10.1109/ITNEC.2019.8729026

Gu, S., Kelly, B. T., & Xiu, D. (2019). Autoencoder Asset Pricing Models. *SSRN Electronic Journal*, 1–32. https://doi.org/10.2139/ssrn.3335536

Hossain, M. A., Karim, R., Thulasiram, R., Bruce, N. D. B., & Wang, Y. (2019). Hybrid Deep Learning Model for Stock Price Prediction. *Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018*, 1837–1844. https://doi.org/10.1109/SSCI.2018.8628641

Jeevan, B., Naresh, E., & Vijaya, B. P. (2018). Share Price Prediction using Machine Learning Technique. *2018 3rd International Conference on Circuits, Control, Communication and Computing (I4C)*, (1), 1–4.

Jiao, Y., & Jakubowicz, J. (2018). Predicting stock movement direction with machine learning: An extensive study on S&P 500 stocks. *Proceedings - 2017 IEEE International Conference on Big Data, Big Data 2017*, *2018-Janua*, 4705–4713. https://doi.org/10.1109/BigData.2017.8258518

Kim, S., & Kim, H. (2016). A new metric of absolute percentage error for intermittent demand forecasts. *International Journal of Forecasting*, *32*(3), 669–679. https://doi.org/10.1016/j.ijforecast.2015.12.003

Kumar, I., Dogra, K., Utreja, C., & Yadav, P. (2018). A Comparative Study of Supervised Machine Learning Algorithms for Stock Market Trend Prediction. *Proceedings of the International Conference on Inventive Communication and Computational Technologies, ICICCT 2018*, (Icicct), 1003–1007. https://doi.org/10.1109/ICICCT.2018.8473214

Li, J., Bu, H., & Wu, J. (2017). Sentiment-aware stock market prediction: A deep learning method. *14th International Conference on Services Systems and Services Management, ICSSSM 2017 - Proceedings*. https://doi.org/10.1109/ICSSSM.2017.7996306

Masters, D., & Luschi, C. (2018). Revisiting Small Batch Training for Deep Neural Networks, 1–18. Retrieved from http://arxiv.org/abs/1804.07612

Misra, P., & Chaurasia, S. (2019). Forecasting Direction of Stock Index Using Two Stage Hybridization of Machine Learning Models. *2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 533–537. https://doi.org/10.1109/icrito.2018.8748530

Nelson, D. M. Q., Pereira, A. C. M., & Oliveira, R. A. De. (2017). Stock Market ' s Price Movement Prediction With LSTM Neural Networks, (Dcc), 1419–1426.

Oncharoen, P., & Vateekul, P. (2018). Deep Learning for Stock Market Prediction Using Event Embedding and Technical Indicators. *ICAICTA 2018 - 5th International Conference on Advanced Informatics: Concepts Theory and Applications*, 19–24. https://doi.org/10.1109/ICAICTA.2018.8541310

Ouahilal, M., Mohajir, M. El, Chahhou, M., & El Mohajir, B. E. (2017). Optimizing stock market price prediction using a hybrid approach based on HP filter and support vector regression. *Colloquium in Information Science and Technology, CIST*, *0*, 290–294. https://doi.org/10.1109/CIST.2016.7805059

Prastyo, A., Junaedi, D., & Sulistiyo, M. D. (2017). Stock Price Forecasting Using Artificial Neural Network. *Fifth International Conference on Information and Communication Technology (ICoICT)*, *0*(c).

Pun, T. B., & Shahi, T. B. (2018). Nepal Stock Exchange Prediction Using Support Vector Regression and Neural Networks. *Proceedings of 2018 2nd International Conference on Advances in Electronics, Computers and Communications, ICAECC 2018*, 1–6. https://doi.org/10.1109/ICAECC.2018.8479456

Qian, Q., & Xiaoxia, W. (2019). Stock price reversal point prediction based on ICA and SVM. *ACM International Conference Proceeding Series*, (5), 101–104. https://doi.org/10.1145/3325730.3325760

Radiuk, P. M. (2018). Impact of Training Set Batch Size on the Performance of Convolutional Neural Networks for Diverse Datasets. *Information Technology and Management Science*, *20*(1), 20–24. https://doi.org/10.1515/itms-2017-0003

Rasel, R. I., Sultana, N., & Hasan, N. (2017). Financial Instability Analysis using ANN and Feature Selection Technique: Application to Stock Market Price Prediction. *2016 International Conference on Innovations in Science, Engineering and Technology, ICISET 2016*. https://doi.org/10.1109/ICISET.2016.7856515

Samarawickrama, A. J. P., & Fernando, T. G. I. (2018). A recurrent neural network approach in predicting daily stock prices an application to the Sri Lankan stock market. *2017 IEEE International Conference on Industrial and Information Systems, ICIIS 2017 - Proceedings*, *2018-Janua*, 1–6. https://doi.org/10.1109/ICIINFS.2017.8300345

Selvin, S., Vinayakumar, R., Gopalakrishnan, E. A., Menon, V. K., & Soman, K. P. (2017). Stock price prediction using LSTM, RNN and CNN-sliding window model. *2017 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2017*, *2017-Janua*, 1643–1647. https://doi.org/10.1109/ICACCI.2017.8126078

Soni, D., Agarwal, S., Agarwel, T., Arora, P., & Gupta, K. (2018). Optimised Prediction Model for Stock Market Trend Analysis. *2018 11th International Conference on Contemporary Computing, IC3 2018*, 2–4. https://doi.org/10.1109/IC3.2018.8530457

Ta, V. D., Liu, C. M., & Addis, D. (2018). Prediction and portfolio optimization in quantitative trading using machine learning techniques. *ACM International Conference Proceeding Series*, 98–105. https://doi.org/10.1145/3287921.3287963

Tang, J., & Chen, X. (2018). Stock market prediction based on historic prices and news titles. *ACM International Conference Proceeding Series*, 29–34. https://doi.org/10.1145/3231884.3231887

Usmani, M., Ebrahim, M., Adil, S. H., & Raza, K. (2019). Predicting Market Performance with Hybrid Model. *2018 3rd International Conference on Emerging Trends in Engineering, Sciences and Technology, ICEEST 2018*, 1–4. https://doi.org/10.1109/ICEEST.2018.8643327

Wang, Y., Yao, H., & Zhao, S. (2016). Auto-encoder based dimensionality reduction. *Neurocomputing*, *184*, 232–242. https://doi.org/10.1016/j.neucom.2015.08.104

Weng, B., Lu, L., Wang, X., Megahed, F. M., & Martinez, W. (2018). Predicting short-term stock prices using ensemble methods and online data sources. *Expert Systems with Applications*, *112*, 258–273. https://doi.org/10.1016/j.eswa.2018.06.016

Yao, S., Luo, L., & Peng, H. (2018). High-frequency stock trend forecast using LSTM model. *13th International Conference on Computer Science and Education, ICCSE 2018*, (Iccse), 293–296. https://doi.org/10.1109/ICCSE.2018.8468703

Zhang, C., Wang, Y., Ji, Z., Zhao, X., Zhang, J., & Yang, Y. (2018). Predicting Chinese stock market price trend using machine learning approach. *ACM International Conference Proceeding Series*, 6–9. https://doi.org/10.1145/3207677.3277966