# Dhaka Stock Market Analysis With ARIMA-LSTM Hybrid Model

by

Raisul Islam Arnob
15301117
Rafatul Alam
15101099
Alvi Ebne Alam
15101062

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
August 2019

# Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

**Student's Full Name & Signature:**

Raisul Islam
15301117

Rafatul Alam
15101099

Alvi Ebne Alam
15101062

# Abstract

This paper proposes the forecasting of correlation coefficients of Dhaka Stock Exchange market assets required for portfolio optimization using an ARIMA-LSTM hybrid model. We have developed a robust model that encompasses both linearity and non-linearity within the datasets of the Dhaka stock market with a hybrid combining ARIMA model and a Recurrent Neural Network called LSTM. Our hybrid model tries to utilize the unique properties of both the ARIMA model and the LSTM model. We have filtered the linear components in the datasets using the ARIMA model and passed the residuals obtained onto the LSTM model which deals with the nonlinear components and random errors. We have compared the empirical results of this model with several other traditional statistical models used in portfolio management namely the Single Index model, Constant Correlation model and Historical Model. We have also predicted the correlation coefficients using the ARIMA model to see how one of the model in our hybrid performs individually. The test results show that the hybrid model excels the other models in accuracy and indicates that the ARIMA-LSTM hybrid model can be an effective way of predicting correlation coefficients required for portfolio optimization.

**Keywords:** LSTM, ARIMA, hybrid, Portfolio, Dhaka Stock Exchange, linear, non-linear

# Table of Contents

# List of Figures

# List of Tables

# Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

| | |
|---|---|
| ACF | Auto Correlation Function |
| ADF | Augmented Dicky Fuller |
| AIC | Akaike Information Criterion |
| ANN | Artificial Neural Network |
| ARIMA | Auto-Regression Integrated Moving Average |
| CAPM | Capital Asset Pricing Model |
| DSE | Dhaka Stock Exchange |
| LSTM | Long Short-Term Memory |
| MAD | Mean Absolute Deviation |
| MAE | Mean Absolute Error |
| MLE | Maximum Likelihood Estimator |
| MSE | Mean Squared Error |
| PACF | Partial Auto Correlation Function |
| ReLU | Rectified Linear Unit |
| RMSE | Root Mean Squared Error |
| RNN | Recurrent Neural Network |

# Chapter 1

# Introduction

The stock market is volatile and unpredictable in nature, the investors' main interest lies on maximizing the profitable return on investment and reducing the risk. In stock market asset analysis, a group of monetary assets consisting of stocks, commodities, currencies and cash equivalents along with mutual, exchange traded, closed funds and other items is termed as a portfolio. Correlation coefficients can be found between the stock market companies based on certain parameter like closing price. These coefficients are used to define strong or weak relations between pair of companies. This in turn helps in portfolio optimization. A portfolio serves as a guide for potential return and risk in investment.

In 1952, Markowitz proposed the Modern Portfolio Theory in his paper 'Portfolio Selection' [1]. This theory shows how investors can form a portfolio that will give them maximum returns for a certain level of risk. It suggests that a group of optimal portfolios termed as 'Efficient Frontier' can be built which offer maximum possible return depending on investor's desired level of risk. The Modern Portfolio Theory contends that the traits of an investment's risk and return should be assessed based on how the investment affects the overall portfolio's risk and return. Finding correlations between company pairs help to find an investment's merit in context to the entire portfolio. Lower or negative correlation means the companies are loosely related. So, investors averse to risk can form portfolios with companies with lower or negative correlation. On the other hand high risk can many times reward the investor with greater returns. Building a portfolio with positively correlated companies would be preferable for such investors. For better portfolio optimization, the correlation coefficients in the near future need to be forecasted well enough and this can be done using the historical data.

Several statistical methods are used for predicting correlations, but the most commonly used statistical models for forecasting the correlation coefficients between the stocks in a portfolio are Single Index model, Constant Correlation model and Historical Model. Beside these traditional models, several other different models such as the ARIMA and along the lines of deep learning, RNN have the potential to work better on forecasting correlation coefficient values between stock assets. This can be justified given that these models have proven to work well on forecasting short term as well as long term stock market data.

The ARIMA model is particularly good at forecasting from linear trending datasets but has not performed well enough on datasets showing non-linear trend behavior. This limits the accuracy of the predictions provided by this model since stock market data tends to exhibit both linear and nonlinear trend. The RNN (Recurrent Neural Network) involving the use of LSTM (Long Short-Term Memory) cells has been proven in several other research papers to forecast well from non-linear trending data. Each model has its own set of drawbacks while interpreting from either of linear and non-linear trending datasets. Most of the researches done on financial market, have particularly focused on forecasting or predicting either long-term or short-term stock market prices.

Our aim is to propose an ARIMA-LSTM hybrid model to forecast the correlation coefficients between stock market assets. For this research we have used 15 registered companies from Dhaka Stock Exchange (DSE). We then analyze and compare the performance of our proposed model against the traditional statistical models and also the widely used ARIMA model. Even though such model has been used in other time series data, forecasting stock market correlation coefficients with this model is still relatively a new thing. Especially, we have not come across any previous research work which has aimed at encompassing both linearity and non-linearity of the Dhaka Stock Exchange dataset. In this paper, we have tried to implement this idea.

# Chapter 2

# Literature Review

Many researchers have studied and found that majority of the developed countries' stock markets exhibit random walk movement [17]. It has been found that inefficiency of any emerging market happens due to the market's size, thinness of trading and authentication of the information disclosed. Early studies performed on Dhaka Stock Exchange for examining the weak form efficiency were found to be inconclusive. However, more recent studies investigating the weak form inefficient confirmed the market inefficiency [21].

Alam et. al (2007) investigated like other researchers, on performing an empirical case study on Dhaka Stock Exchange to test its market efficiency. They used a dataset of 3209 daily observations from a period of 1994 to 2005 in order to analyze risk and return ratio followed by CAPM modelling to find the risk-return relationship in the market. They deduced that there was not enough liquidity in the stock market data and the market is of weak form inefficient evidently [27].

Chakraborty (2018) performed a comparative study on analyzing and predicting the stock market trend movement amongst the companies in Dhaka Stock Exchange using ARIMA model and LSTM model respectively. He used historical stock market data and split it into train and test sets. He used the training set to identify an appropriate ARIMA model for forecasting stock price of a chosen company for the next 60 days approximately. He used the same training dataset to train his proposed LSTM model and perform the prediction for the same number of future days [17].

Jia (2016) performed a study on determining the prediction accuracy of LSTM networks on stock markets' prices particularly on high, low, open or close prices respectively. He used the open source dataset of Google's daily stock market data ranging from the 1st of January 2005 till 31st of December 2015. He split the resultant dataset into train and test sets. Instead of feeding the whole train dataset on the proposed neural network at once, the length of the dataset was segmented into several portions, each of size 256 lengthwise. The dataset was trained on an optimal neural network of 100 epochs, batch size of 20 samples and a value of 0.001 on Adam optimiser. The accuracy of the model was deduced in the form calculating the RMSE on a varying range of layers followed by a different number of units on each layer. The model's performance was also compared against a simplified neural network [5].

Salem et. al (2019) tested the learning performance of the various forms of LSTM types. The different types of LSTM cells were embedded within a proposed neural network in order to test for its performance. The LSTM variants within the custom neural network model, were tested with several forms of activation functions in them such as Tanh, Linear, Sigmoid, ReLu etc. The learning rate of the LSTM variant cells, together with their validation accuracy and loss, were measured after iterating them up to 100 epochs precisely.

Zhang (2001) investigated the forecasting performance of a custom-made model by combining ARIMA and ANN. This model was then compared with an individual ARIMA model and an ANN model respectively. He used three different types of datasets to investigate the performance of his proposed model and measured its accuracy in the form of calculating MSE and MAD values. In every dataset experimentation, his proposed hybrid model has performed significantly well enough to identify the trending patterns from the data [18].

Roondiwala et. al (2017) used LSTM only to make predictions of certain features of stock market data, particularly on the stock market return of Nifty50 company. They trained and predicted for features using a custom LSTM model. This model is made up of a sequential input layer followed by 2 layers of LSTM. To add more, a ReLU activation enabled dense layer was included. Finally as an output, a dense layer with linear activation function was added. After performing several iterations with different number of pricing types and epochs, their LSTM model has performed better on 4 features mainly High, Low, Open and Close prices respectively with an epoch value of around 500 and achieved less value on RMSE [20].

# Chapter 3

# Methodology

Our hybrid ARIMA-LSTM model entails separate processing of the data. The data we obtained is first preprocessed to remove gaps in the data to make it continuous before it is given as an input to our model. The series of steps involved has been outlined below.

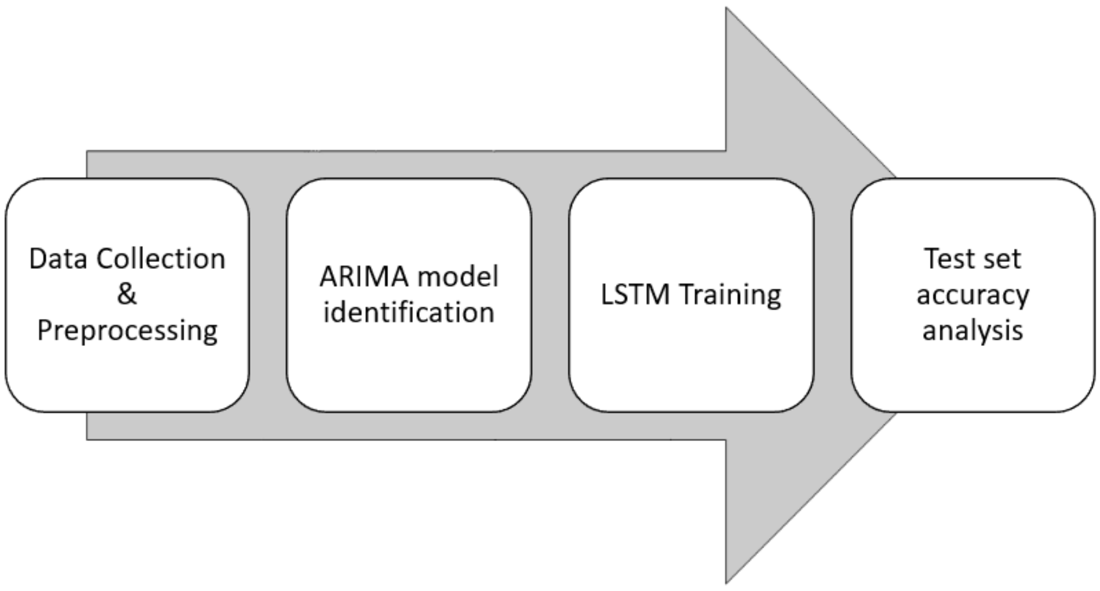

Data Collection & Preprocessing
ARIMA model identification
LSTM Training
Test set accuracy analysis

Figure 3.1: Workflow

After preprocessing, the data is given as an input to the ARIMA model where the precise order of the model is determined and parameters are identified. The residual values from the ARIMA model mostly consists of nonlinear data. This residual value acts as an input for the LSTM model where the input layer gates pass the data to the hidden layers.Both the forget gate and the input candidate gate within each LSTM cell manipulates the data and propagates it through the cells of each layer. Finally, we compared the accuracy of the results of our model with prevalent traditional model values using statistical error checking measures like MSE and MAE.

## 3.1 Concept of ARIMA

The ARIMA is a regressive model used for detection of linearity in stationary data. It is expressed generally as (p, d, q) where the p, q determines the definite order of AR and MA models,whereas d specifies the iterations for differencing required to make the input data stationary. Mathematically ARIMA can be modelled as,

$$\begin{aligned} x_t&=\mathrm{c}+{}_1x_{t-1}+{}_2x_{t-2}+....+{}_px_{t-p}\text{-}\theta_1\ \epsilon_{t-1}\text{-}\ \theta_2\ \epsilon_{t-2}\text{-}.....\text{-}\theta_q\ \epsilon_{t-q} \\ &=\mathrm{c}+\textstyle\sum_{k=1}^{p}\ {}_kx_{t-k}\text{-}\ \sum_{l=1}^{q}\theta_l\ \epsilon_{t-l} \end{aligned} \tag{1}$$

Symbol c is constant and the coefficients ${}_k$, $\theta_l$ belong to $\mathrm{x}_{\text{t-k}}$ of AR and $\epsilon_{t-1}$ of MA. $\epsilon_{t-1}$ denotes loss term encompassing zero mean and unvaried variance. A standard methodology to build the ARIMA model consists of an iterative three-step procedure.

- Model identification and model selection.
- Parameter estimation.
- Model checking.

The model identification involves precisely determining the values of p, q which in turn deduces the order of AR and MA model. Stationary data points are required for this step meaning it is mandatory to have constant mean, covariance, variance or autocorrelation over given time periods. Non stationary data is converted to stationary data by differencing once or twice usually. Then the ACF and PACF is used to select the appropriate ARIMA model.
Parameter estimation involves an optimization process that uses AIC. The AIC equation is given below.

$$\text{AIC} = \text{-2(log-likelihood)} + 2\text{k} \tag{2}$$

Here, using different variants of the likelihood function, we opted to use the maximum for calculation purpose. The lower the value of AIC the better a given model. If the result of the analysis is not satisfactory, then the three steps are repeated until the optimal model is acquired.

## 3.2 LSTM

Neural networks are cell structures that take vector of inputs and process the inputs within the hidden layers using activation functions and different gating techniques in a series of propagation steps from one cell state to the other until the output of the vectors are produced. Traditionally, the neural networks involve RNN but it has the limitation that it cannot propagate data from previous time steps which are considerably large because of its smaller memory. LSTM is an improved and extended variant of RNN that can remedy this problem.

LSTM is good at processing non-linear data because it employs several non-linear activation functions like sigmoid, hyperbolic tangent. Generally, neural networks perform better and have higher accuracy but have some overlapping regions with other models showing that it does not always outperform other models. Hence, we decided to use a hybrid ARIMA-LSTM structure can handle linearity and non-linearity to improve overall accuracy.

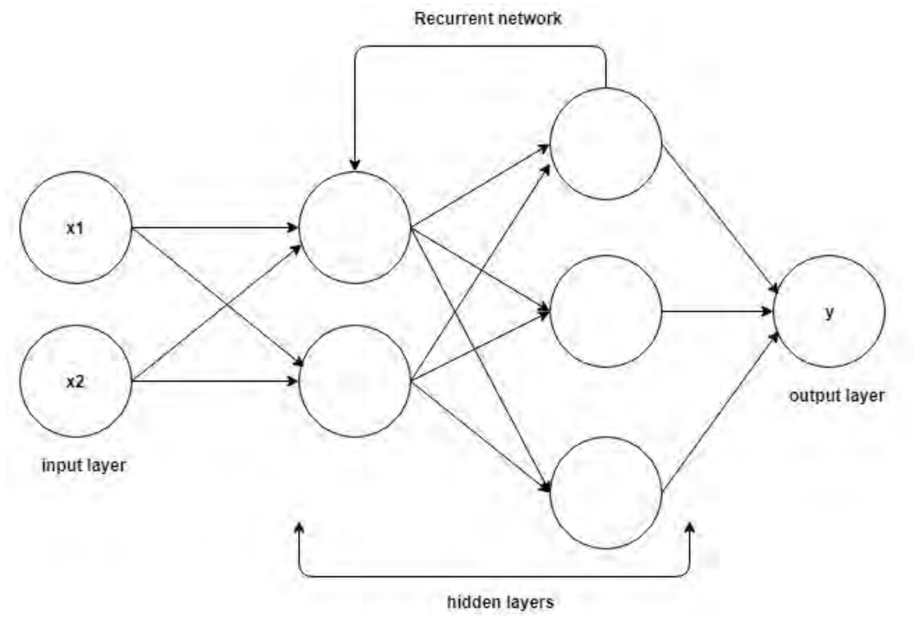


Figure 3.2: RNN Architecture

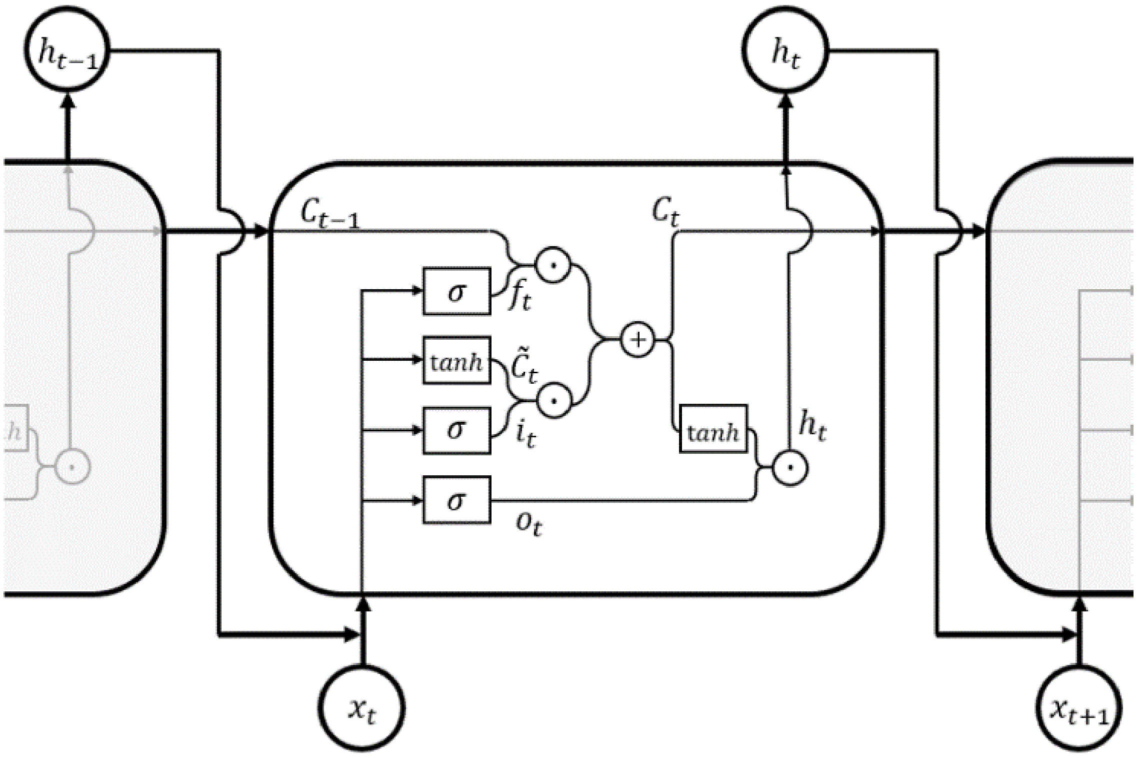


Figure 3.3: LSTM cell structure

Each cell comprises of input, next likely input, forget gate and output. The gates use two types of non-linear functions namely sigmoid and hyperbolic tangent. These functions evades exploding gradient problem by keeping inputs between -1 to 1 respectively.

The forget gate employs sigmoid activation function to discard values that range from 0 and 1.

$$f_t = \sigma\ (W_f\ .[_{ht-1}, _{x_t}] + b_f) \tag{3}$$

$W_f$ is weight function over $h_{t-1}$ and $x_t$, the current input. Next both the input and next likely input determine new cell state $C_t$. The input gate uses sigmoid activation function while the latter uses hyperbolic tangent function giving $i_t$ and $C_{t\sim}$ .

$$i_t = \sigma\ (W_i\ .[h_{t-1}, x_t] + b_i) \tag{4}$$

$$C_{t\sim} = \tanh\ (W_C\ .[h_{t-1}, x_t] + b_C) \tag{5}$$

Output gate then determines the current cell state output $h_t$ by combining the current state tanh of $C_t$ with function $o_t$.

$$o_t = \sigma(W_0[h_{t-1}.x_t]+b_0) \tag{6}$$

$$C_t\ f_t.C_{t-1}+i_t.C_{t\sim} \tag{7}$$

$$h_t = o_t.\tanh(C_t) \tag{8}$$

This output then acts as the input for the next cell state and the above-mentioned steps are repeated for all the cell states.

## 3.3 Historical Model

The historical model is the most simplified statistical methodology associated with forecasting correlation between stock assets in order to ensure portfolio optimization. This model makes use of the past correlation values from historical data to forecast correlation coefficient values for the future. Therefore, the model expects the correlation coefficient for a future time frame to be equivalent to the correlation coefficient at a historical time frame.

$$\rho^t_{mn} = \rho^{t-1}_{mn} \tag{9}$$

*m, n : indices of a pair of assets in the correlation coefficient matrix

However, this model has received a fair share amount of criticism due to displaying poor forecasting quality compared to other similar statistical models.

## 3.4 Constant Correlation Model

By default, this model assumes that a complete historical model consists of the mean data of the correlation coefficient only. Any form of deviation from the associated mean value is assumed by the model to be a random noise. This model estimates that the correlation coefficients of all the stock assets in a portfolio is equal to the mean correlation coefficient. Thus, the implementation of this model results in all of the stock assets within a portfolio to have the same value of the correlation coefficient.

$$\rho_{mn}^{t}=\sum \rho_{mn}^{t-1} \div n(n-1)/2 \tag{10}$$

m,n : indices of a pair of assets in the correlation coefficient matrix.
n : total assets in the portfolio

## 3.5 Single-Index Model

This model assumes that returns from stock market assets tend to move in a systematic manner along the rate specified by the 'single-index' or in other words, the stock market return. In order to determine the systematic movement, it is mandatory to state clearly the market return beforehand. This specification is called the 'market model' and was initially discussed by H. M. Markowitz [1]. This 'market model' is associated with the return of an asset, m with the market return at a given time, t. The phenomenon has been described by the equation shown below:

$$R_{m,t}=\alpha_m+\beta_m+R_{j,t}+\epsilon_{m,t}$$

$R_{m,t}$: asset m's return at time t
$R_{j,t}$: market's return at time t
$\alpha_m$: asset m's risk adjusted excess return
$\epsilon_{m,t}$: residual return; error term

Here, we will be using the beta of assets m and n to determine the correlation coefficient. With the help of the equation,

$$C_{ov}(R_m, R_n)=\rho_{mn}\ \sigma_m\ \sigma_n=\beta_m\ \beta_n\ \sigma_j^2$$

$\sigma_m\ \sigma_n$: standard deviation of stock assets i / j's return
$\sigma_m$ : standard deviation of the market return
The estimated correlation coefficient $\rho_{m,n}$ would be,

$$\rho_{m,n}=\beta_m\ \beta_n\ \alpha_j^2/\alpha_m\ \alpha_n \tag{11}$$

# Chapter 4

# Data collection and preprocessing

## 4.1 Data fetching and API

One of the aims of our work was to build an API and connect it to an online repository for collecting stock data. The API would help us and other future researchers to get easy access to the consistent DSE datasets we would accumulate overtime. This section discusses the build procedure of the API.

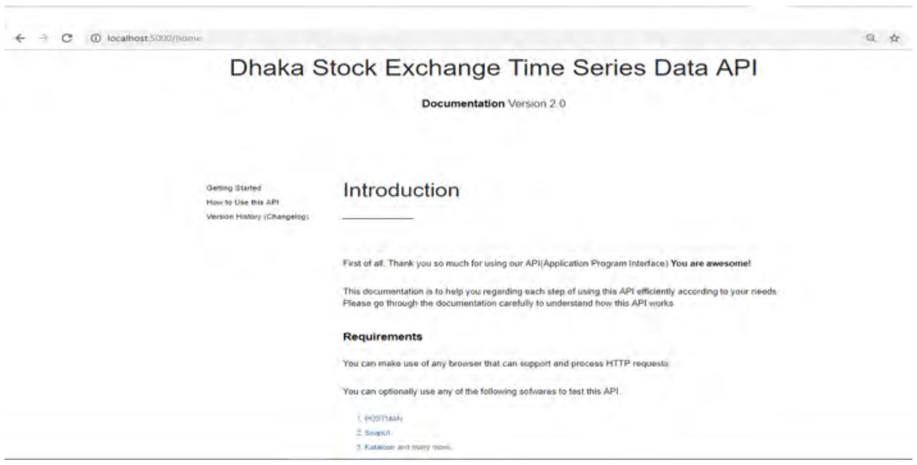
We have scraped off daily market data of 356 registered companies of Dhaka Stock Exchange by building a Python script. DSE provides data in its official website archive and also through third party websites such as AmarStock which uses data for plotting and visualization. However, the archive does not contain data before 1st of July 2017 and hence we collected the rest of the data from AmarStock. The scrapped off data was then stored in Google Firebase, which is an online real time database.

For our purpose we have collected daily index data under the parameters opening price, closing price, high, low, and trading volume. The collected data was originally fetched in UNIX time format, we converted the UNIX format to date time format. The data is of varying time range starting as early as January 2000 till February 2019.

Dhaka Stock Exchange does not have any API integrated with its website from where data can be easily viewed and accessed in simplified format. So, we aimed at building an API comprising of the historical data of each company which can be retrieved in JSON format. The API that we have designed is a Web API built with Python microframework Flask. In the API, the data has been arranged such that it can be accessed on a yearly basis for any specified company.

Our Web API has been documented precisely explaining how to trigger each HTTP GET request in order to receive historical data of certain stock market companies and use them to serve several research purposes.

The figures 4.2 to 4.4 respectively represent the http GET request pattern to trigger an appropriate API response.


localhost:5000/home

Dhaka Stock Exchange Time Series Data API

**Documentation** Version 2.0

Getting Started
How to Use this API
Version History (Changelog)

Introduction

First of all, Thank you so much for using our API(Application Program Interface) **You are awesome!**

This documentation is to help you regarding each step of using this API efficiently according to your needs. Please go through the documentation carefully to understand how this API works.

**Requirements**

You can make use of any browser that can support and process HTTP requests.

You can optionally use any of the following sofwares to test this API.

1. POSTMAN
2. SoapUI
3. Katalon and many more.

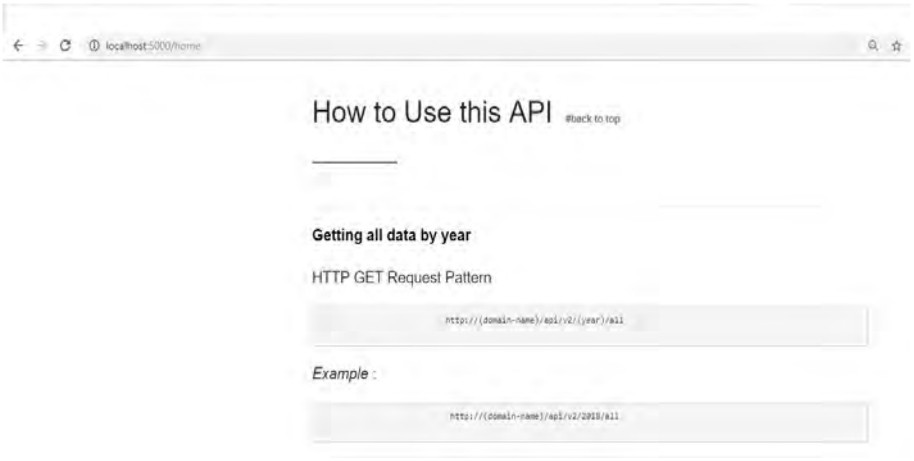
Figure 4.1: API Introduction Page

Figure 4.1 represents the website's homepage API documentation homepage.

Figure 4.2 represents the http GET request pattern to get all the historical data for all companies for a particular year. The year is specified inside the (year) tag within the http GET request.

Figure 4.3 represents the http GET request pattern to get all the historical data for all companies for a single month in a given year, as well as how to specify the range of the number of month historical data to receive for all companies. The month's name is specified inside the (month) tag within the http GET request.

Figure 4.4 represents the request arguments that can be passed on to the GET request pattern to get your desired form of filtered data. The request argument of the type 'company' requests the user to input the name of any of 356 registered companies in Dhaka Stock Exchange. To add more, the 'filter' argument type can be used to state precisely the types of data that the user wants to access for a given company.


localhost:5000/home

How to Use this API #back to top

**Getting all data by year**

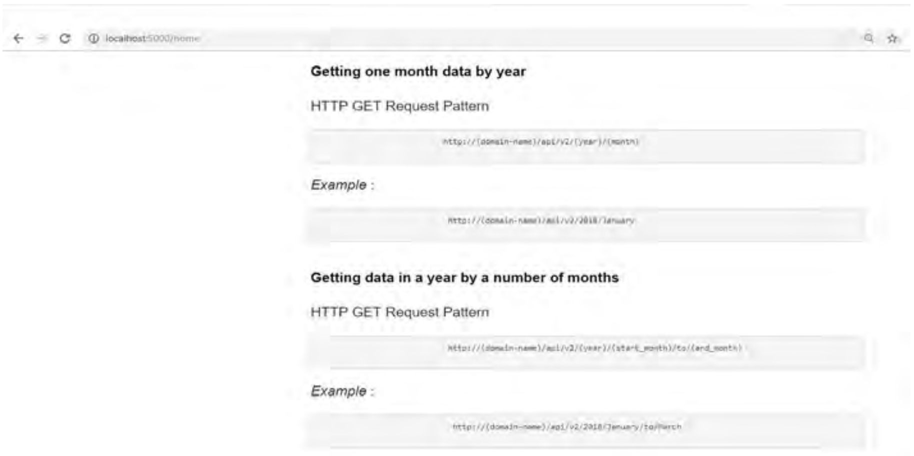
HTTP GET Request Pattern

```
http://(domain-name)/api/v2/(year)/all
```

*Example :*

```
http://(domain-name)/api/v2/2018/all
```

Figure 4.2: API GET request by year


localhost:5000/home

**Getting one month data by year**

HTTP GET Request Pattern

```
http://(domain-name)/api/v2/(year)/(month)
```

*Example :*

```
http://(domain-name)/api/v2/2018/January
```

**Getting data in a year by a number of months**

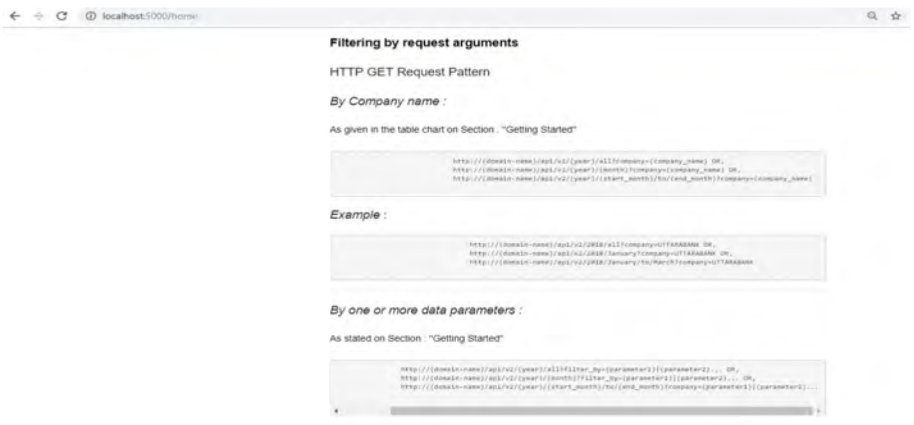
HTTP GET Request Pattern

```
http://(domain-name)/api/v2/(year)/(start_month)/to/(end_month)
```

*Example :*

```
http://(domain-name)/api/v2/2018/January/to/March
```

Figure 4.3: API GET request by months in year


localhost:5000/home

**Filtering by request arguments**

HTTP GET Request Pattern

*By Company name :*

As given in the table chart on Section : "Getting Started"

```
http://(domain-name)/api/v2/(year)/all?company=(company_name) OR,
http://(domain-name)/api/v2/(year)/(month)?company=(company_name) OR,
http://(domain-name)/api/v2/(year)/(start_month)/to/(end_month)?company=(company_name)
```

*Example :*

```
http://(domain-name)/api/v2/2018/all?company=UTTARABANK OR,
http://(domain-name)/api/v2/2018/January?company=UTTARABANK OR,
http://(domain-name)/api/v2/2018/January/to/March?company=UTTARABANK
```

*By one or more data parameters :*

As stated on Section : "Getting Started"

```
http://(domain-name)/api/v2/(year)/all?filter_by=(parameter1)|(parameter2)... OR,
http://(domain-name)/api/v2/(year)/(month)?filter_by=(parameter1)|(parameter2)... OR,
http://(domain-name)/api/v2/(year)/(start_month)/to/(end_month)?company=(parameter1)|(parameter2)...
```

Figure 4.4: API GET request filter by parameters

## 4.2 Data preprocessing

Before feeding the data to the hybrid model we have built, we needed to preprocess it as the initial datasets of companies were inconsistent. In this section we discuss the steps and procedures we went through to turn the scrapped data into a dataset suitable for our model.

In this paper we are taking 'close' price data of the Dhaka Stock Exchange companies. Though we had collected data of 356 companies, all companies were not listed in DSE at the same time. So, in order to create an equal dataset of the companies we decided to select those which have data from January 2007 onwards. We found 44 companies that fit the requirement. Next we fill in the missing data by imputing values at time t with t-1 values for selected assets. Out of these complete datasets we select 15 companies at random. The randomly selected 15 companies' tickers are listed at 'Appendices A'.

We take pairs of the companies and calculate the correlation coefficient between the imputed datasets with a 100-day rolling time window. We get 15C2 or 105 possible pairs and that with a 100-day stride results to 10500 sets of data each containing 24-time steps. In the final step of preprocessing we build the train, test1 and test 2 datasets with 10500x24 data.

# Chapter 5

# ARIMA Modelling

To deal with the linear components of our datasets we undergo through the ARIMA process. We discuss in detail the steps of this process and how the residual values help in forecast. Also, we determine MSE and MAE values of optimal ARIMA model for comparison with our hybrid model.

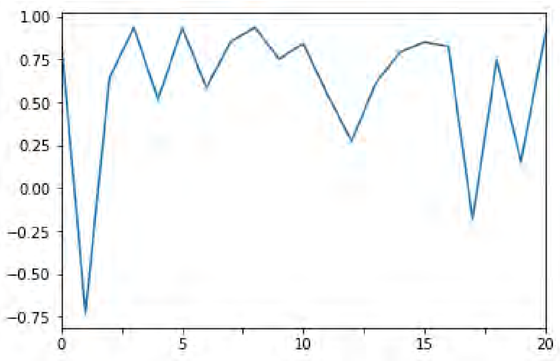
After optimizing the data for 20-time step model we get a new dataset with dimensions 525x24. Then we divide the data into train, test1 and test2 datasets. Train set is formed with columns of indexes 1-21, test1 set with indexes 2-22 and test2 set with indexes 3-23.

We transposed the datasets which gave us 525 columns and 20 rows for each dataset. For each of the columns we first determined whether the set was stationary or not and applied differencing accordingly. A unit root statistical test called ADF test helps us in this regard. We found that most of the columns were required to be differenced once in order to make them stationary.

ADF follows a null hypothesis which says that a series is probably defined by a trend if it can be represented by a unit root i.e. it is not stationary and contains time dependent structure. On the other hand, if this null hypothesis is rejected the series does not have a unit root, is stationary and contains no time dependent structure. In the ADF statistics the null hypothesis is rejected if the p-value is less than 0.05 or else we accept the null hypothesis. The more negative the ADF statistic value is the more likely the series is stationary.

Figure 5.1 and Figure 5.2 shows the graph and ACF, PACF plots of a column from train dataset which is already stationary and does not require any differencing. The p-value is much less than 0.05. ADF statistic is -5.08 which is less than the critical values at 1%, 5% and 10% and this means we can reject the null hypothesis with a significance level less than these percentiles.

Figure 5.3 shows graph of one of the columns within the train dataset which is non stationary initially and requires differencing. It also shows the pattern of the series after differencing once. In the first graph, the p-value is 0.3285 which is greater than 0.05 and so does not reject the null hypothesis. ADF statistic value is greater than the critical values at 1%, 5% and 10%.



ADF Statistic: 5.134777, p-value: 0.000012
Critical Values: 1%: -3.8095, 5%: -3.022

Figure 5.1: ADF statistics of stationary data with no differencing

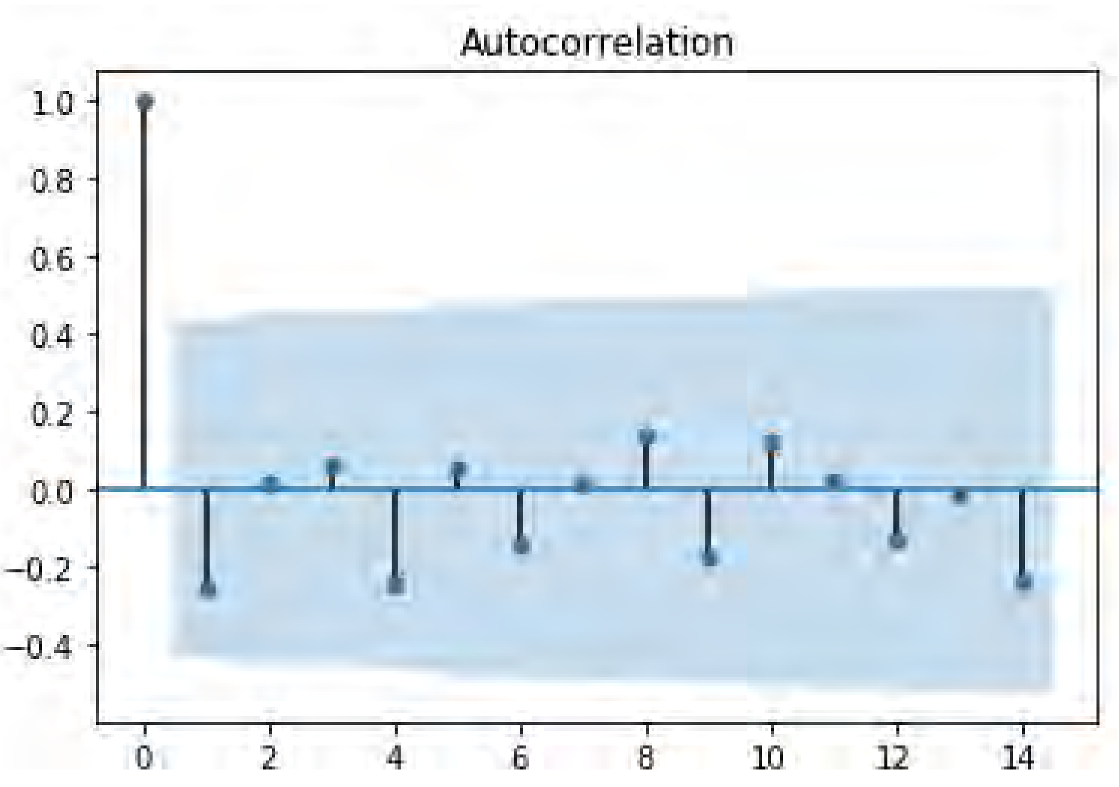
Since we found that majority of the columns in train, test1 and test2 datasets required differencing to make them stationary we decided to difference all the columns once. Then by plotting the ACF and PACF plots on random columns from each dataset we tried to figure out the possible orders of the ARIMA model that best fit those series of data within each column. During this process the models which we found multiple times were (p, d, q) = (0,1,1), (0,1,0), (1,1,1), (1,1,0), (2,1,1) which represents the AR(p), MA(q) and I(d) terms.

Figure 5.4 shows the ACF and PACF plots of order (0,1,1). We can derive the p terms from the PACF plot while the q terms are derived from the ACF plot. The ACF plot shows a significant spike at lag 1 while rest of the spikes are within the 95% confidence zone as represented by the blue region. But in the PACF plot we do not notice any significant lower order spikes. Hence, we can conclude that this data series has 1 q term but no p terms. The rest of the models were found in a similar manner.

We fit these four models with our train, test1 and test2 datasets and compared their AIC values for each column of the datasets. We found the ARIMA model of order (0,1,1) to be the best fit for majority of the datasets as it generated the least AIC value for most of the columns in the train, test1 and test2 datasets. In total the train, test1 and test2 datasets contain 1575 columns. Model (0,1,1) was found to be best fit for 1291 columns, model (1,1,1) was found best for 229 columns, model (0,1,0) for 51 columns and model (2,1,1) for 4 columns.

In the model fit, we used the MLE to compute the log likelihood function. AIC works on the amount of information lost by a model, the less the better. AIC deals with balancing between the goodness of fit and simplicity of the model and reduces risk of overfitting and underfitting.

In the model fit we used the MLE to compute the log likelihood function. AIC works on the amount of information lost by a model, the less the better. AIC deals with balancing between the goodness of fit and simplicity of the model and reduces risk of overfitting and underfitting.For each of the columns in the train, test1 and test2, we make predictions and calculate the residual values. The residual values



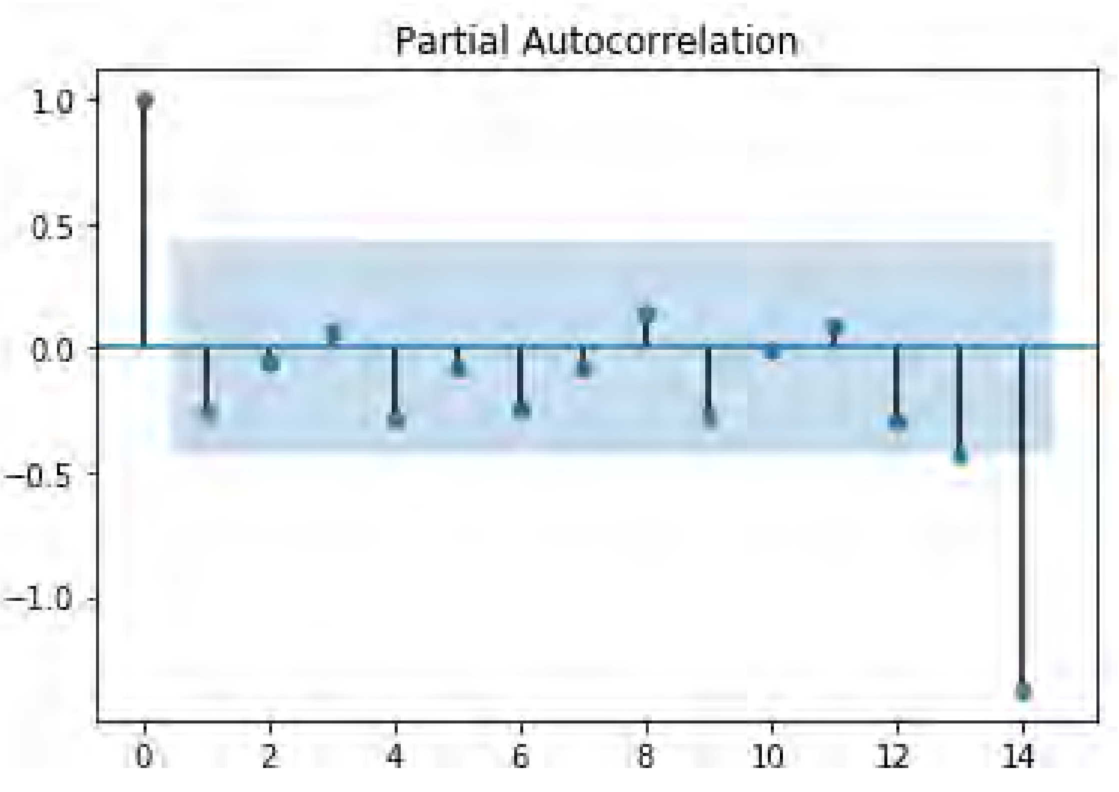
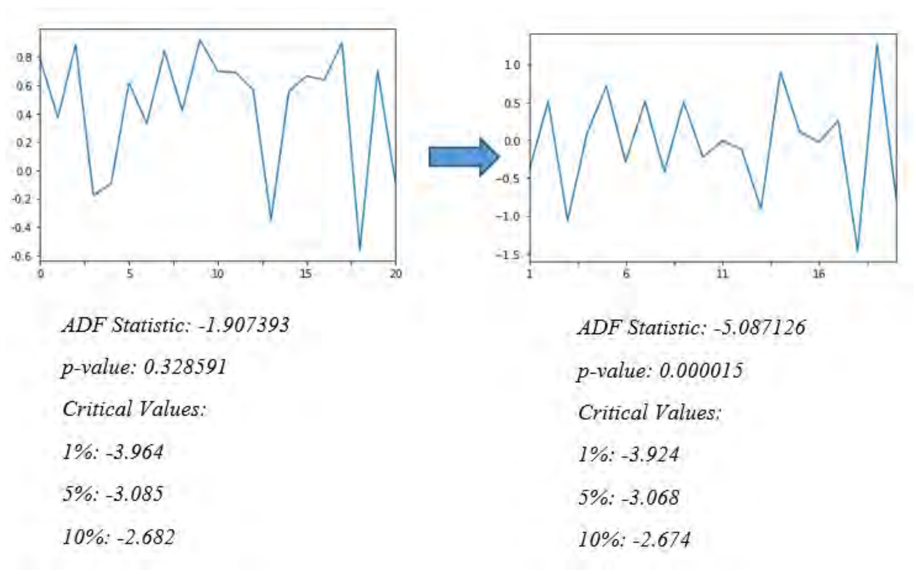


Figure 5.2: ACF and PACF plots for stationary data with no differencing



*ADF Statistic: -1.907393*

*p-value: 0.328591*

*Critical Values:*

*1%: -3.964*

*5%: -3.085*

*10%: -2.682*

*ADF Statistic: -5.087126*

*p-value: 0.000015*

*Critical Values:*

*1%: -3.924*

*5%: -3.068*

*10%: -2.674*

Figure 5.3: Transition of series from non-stationary to stationary

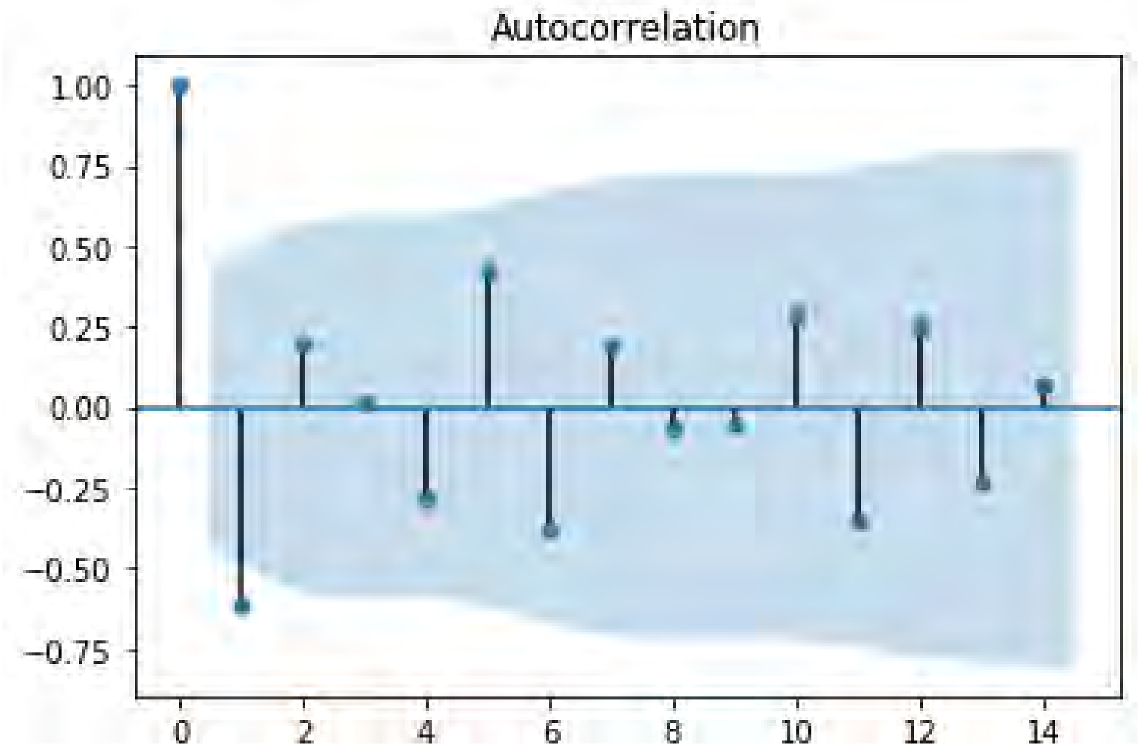
|  | MSE | MAE |
|---|---|---|
| TRAIN | 0.4661 | 0.5779 |
| TEST1 | 0.4560 | 0.5713 |
| TEST2 | 0.4539 | 0.5682 |

Table 5.1: MSE and MAE values for ARIMA model (0,1,1)

are obtained by subtracting the forecasted ARIMA values from the original time series values. We then calculated the MSE and MAE values from the forecasted and original datasets. For the (0,1,1) model the MSE and MAE values are as represented below in Table 1

Through ARIMA modelling we remove any linear tendencies present in our dataset. The residuals contain mostly non-linear components and random errors. ARIMA process is not capable of dealing with non-linear data. For this the obtained residuals are passed on to the LSTM process for analyzing the non-linear portion of the datasets.

Before residual values from the dataset are passed on to the LSTM model, a normal distribution graph is plotted beforehand to analyze the data points. Figure 5.5 below represents the data point distribution curve that appeared to be normally distributed without showing significant amount of skewness to the left or right.



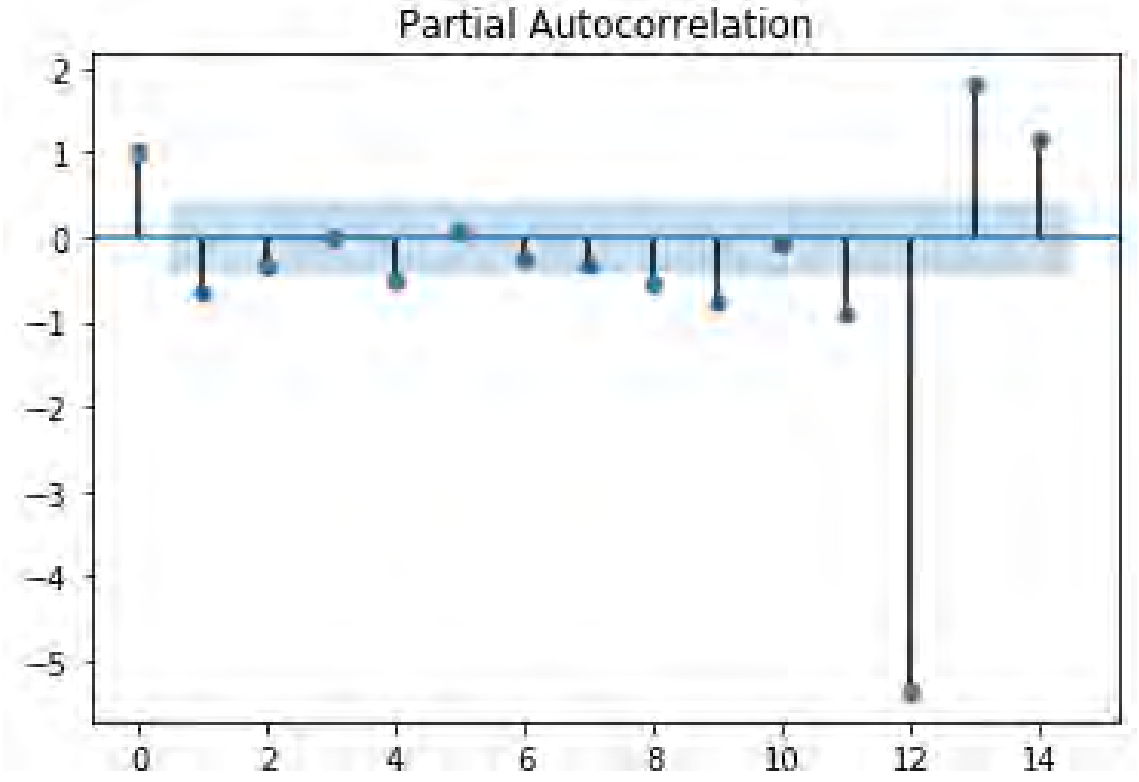


Figure 5.4: ACF and PACF plots showing order (0,1,1)

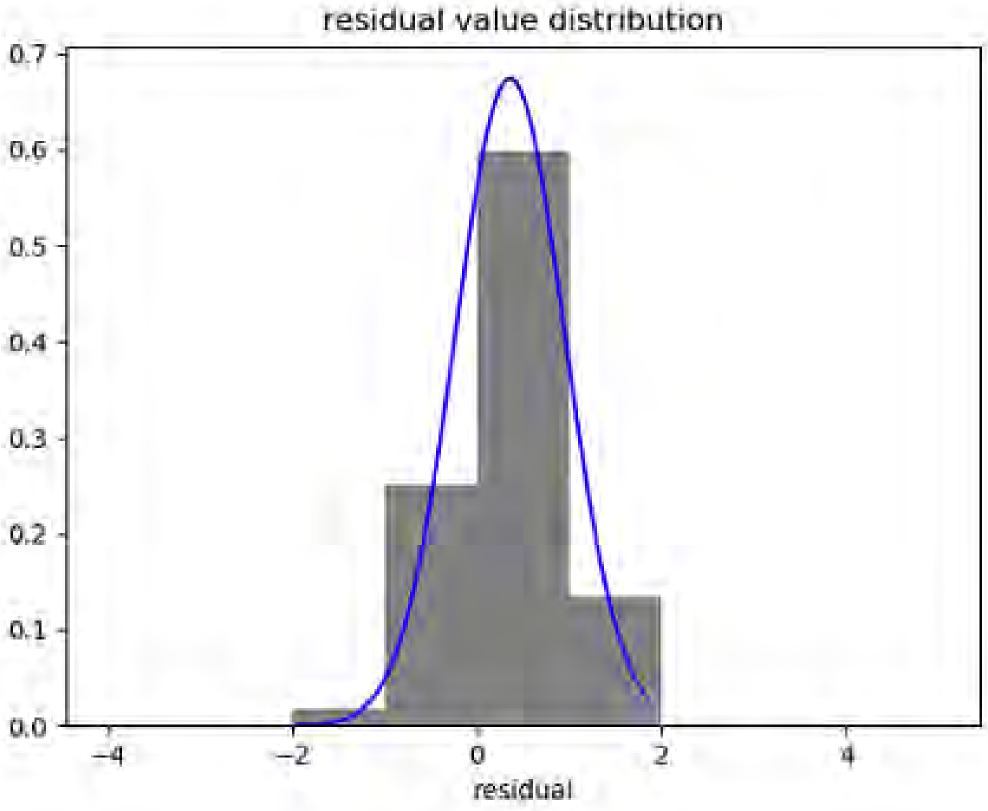


Figure 5.5: Normal Distribution of Residual values

# Chapter 6

# LSTM Modelling

Through LSTM we aim at removing the non-linearity in the residual value datasets for optimal prediction. In this way we would have dealt with both linear and non-linear components in our original datasets. In this section we try to find the correct parameters best fitted for our LSTM model through trial and error.

The residual values obtained from the ARIMA model are reshaped and used to form the Train X and Y, Test1 X and Y and Test2 X and Y datasets respectively. Each, X dataset is comprised of 525 rows of data with 20-time steps and each Y dataset contains the corresponding expected Y output. The splitting of the Train, Test1 and Test2 together with their X and Y components are given below:

·Train X : index 0-19

· Test1 X : index 1-20

· Test2 X : index 2-21

·Train Y : index 20

· Test1 Y : index 21

· Test2 Y : index 22

For training the model, we built the neural network layers with an instance of the Sequential class and use 50 LSTM units. For outputting the prediction, we use 1 dense layer. The model contains an input shape of 20-time steps and 1 feature. We fit the model with an optimal batch size of 105 samples.

Within this LSTM cell, we added a dropout value of 0.4. To prevent issues associated with the overfitting problem, we reduced the neuron interdependency within each LSTM unit by applying the dropout method to deactivate neurons with a certain value of probability. We tried building with no dropouts and another dropout size of 0.2. However, with these initial values we faced issues with overfitting over the train dataset.

We tried implementing a double tanh activation function for the model. Double tanh function is simply an activation function multiplied by a factor of 2. But the results were not satisfactory as the pair of test datasets performed poorly. As another option we tried using the ReLu function which is used widely for time series analysis nowadays. In the end the default tanh activation function performed better than the other two in case of our datasets.

To further assist with the overfitting problem we used Lasso regularization (L1) and Ridge regularization (L2). These regularization techniques keep the weight values from growing too large which in turn alters the data in the neural layers causing loss of information. However, during the trial and error run with our model, we have found that our model with Ridge regularization (L2) had the best output. The Ridge regularization had a value of 0.1 for our LSTM model.

Other details of the model include compiling the model with ADAM optimization algorithm, and MSE and MAE metrics. We iteratively ran our LSTM model up to around 80 epochs. An epoch is an arbitrary measurement of the number of times the whole sample is used in order to train the LSTM model. Our aim was to find the optimal epoch number where the MSE and MAE values of both test1 and test2 datasets converged closer to the train set. The process has been elaborated further in the next chapter. For further details on the LSTM modelling, see LSTM source code in 'Appendix B'

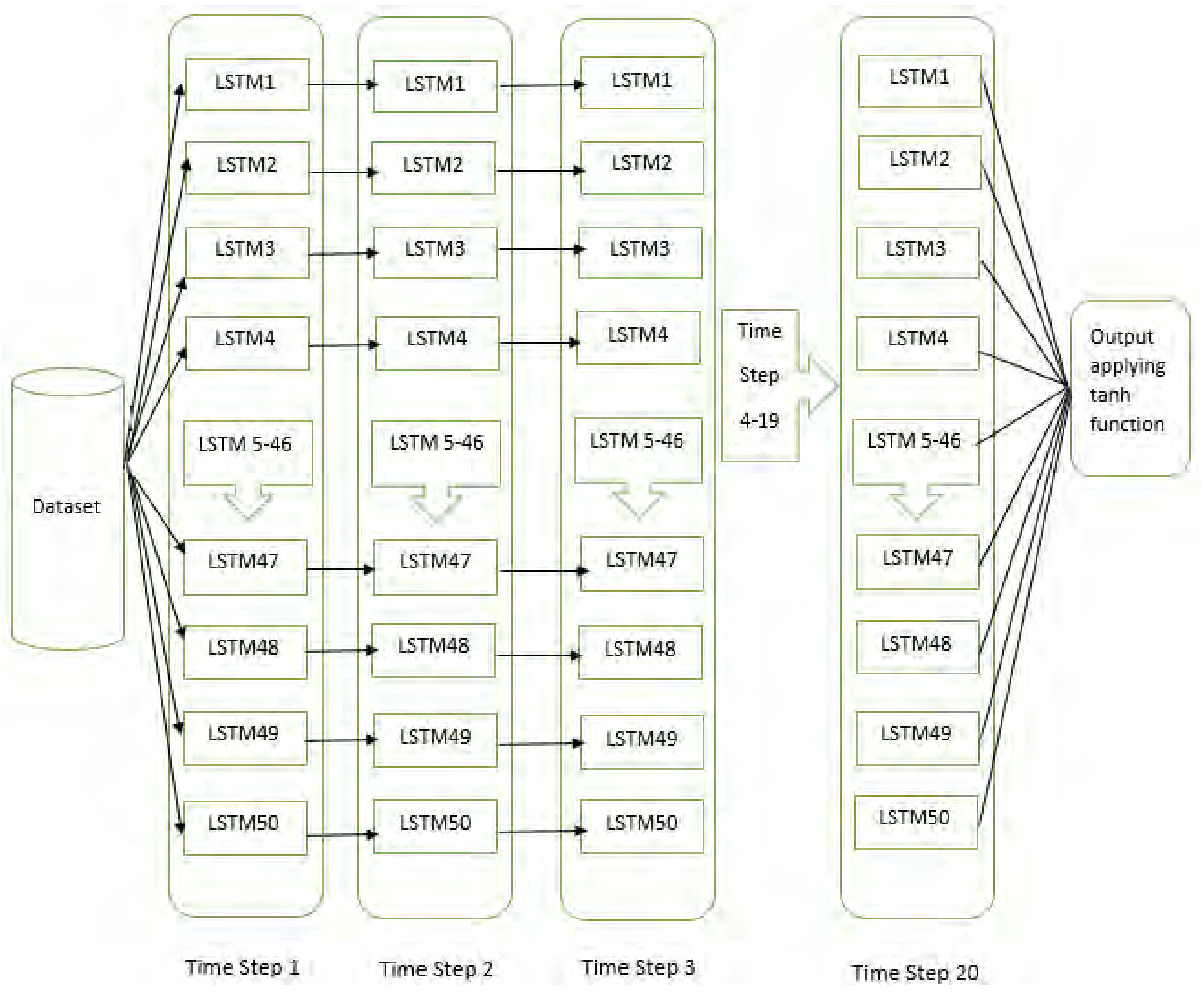


Figure 6.1: LSTM modular representation

# Chapter 7

# Evaluation and Results

For evaluating the ARIMA-LSTM hybrid model, we will be comparing its performance against some of the traditional statistical models, together with ARIMA model, often used in portfolio optimization. The walk forward validation method is used generally to build any model involving the use of neural network at every time step. However, using this procedure is time-consuming computationally. Instead of building a single model on every time step, we train our hybrid model on the first-time step from the train set and test it directly against the test1 and test2 datasets respectively or in other words against the two different time steps.

Our criteria for evaluating and comparing the model will be to find its MSE and MAE value. We calculate the MSE and MAE values for the other traditional statistical models and ARIMA model as well. We ran our model on different number of epochs numerically up to around 80 and determined the optimal epoch number where the MSE and MAE value converged to an acceptable value. From around 40 epochs, the MSE and MAE values started to converge. We made the use of an equation to derive the optimal epoch selection. The equation sums up the performance measure and the overfitting measure to give the outcome of the resultant epoch. The performance measure is deduced by the normalized value of the sum between the MSE of the train set and any of the test1 and test2 datasets divided by the standard deviation of the MSE. On the other hand, the overfitting measure is determined by normalizing the value of the difference between the MSE of the train set and any of the test1 and test2 datasets and dividing it by the standard deviation. We then add up both the performance metric and the overfitting metric to round up and find the Criterion value which is the value of the optimal epoch. This is where MSE values between the train, test1 and test2 datasets converged at a significant tolerance level.

Figures from 7.1 to 7.4 represents the learning curve pattern of our ARIMA-LSTM hybrid model where we compared the forecasting accuracy of our model against the train set, test1 set and test2 set respectively in terms of measuring their MSE and MAE values at a certain epoch number.
The equation below represents the criterion of selecting the optimal epoch number.

$$Criterion{=}\triangle MSE - mean(\triangle MSE)/stdev(\triangle MSE){+}\sum MSE - mean(\triangle MSE)/stdev(\sum MSE) \quad (12)$$

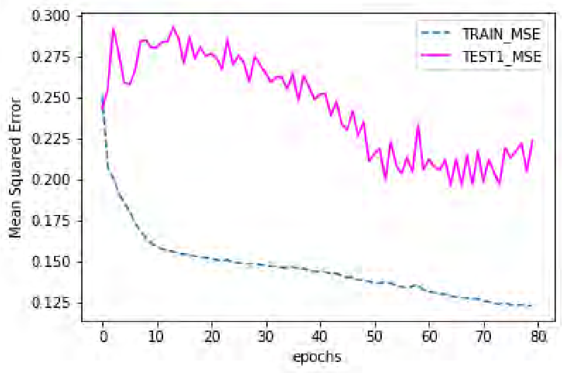


Figure 7.1: MSE distribution on Train and Test1 dataset

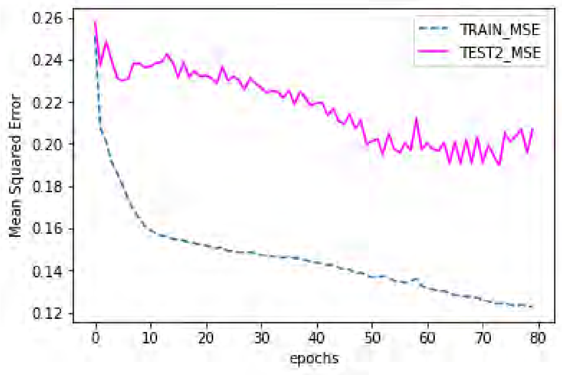


Figure 7.2: MSE distribution on Train and Test2 dataset

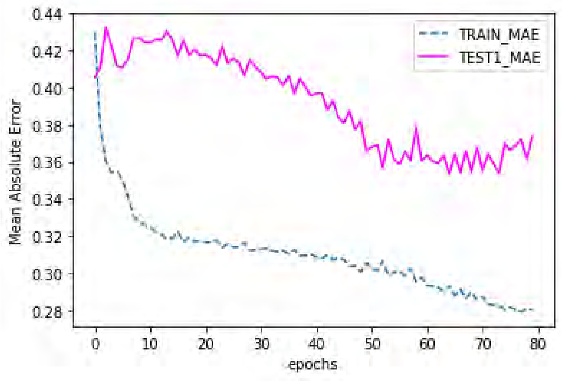


Figure 7.3: MAE distribution on Train and Test1 dataset

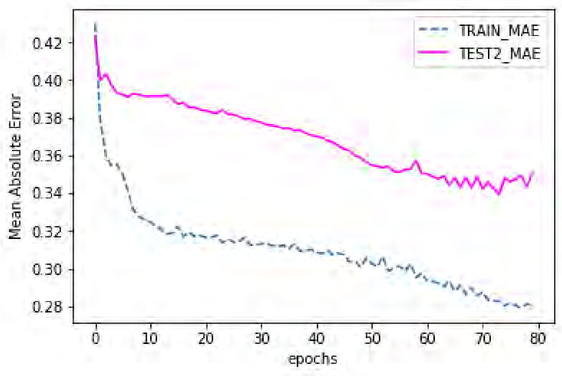


Figure 7.4: MAE distribution on Train and Test2 dataset

We performed 2 iterations to find out the optimal epoch number for our ARIMA-LSTM hybrid model. In the first iteration, we calculated both the performance metric and the overfitting metric and then calculated the overall Criterion value as per the equation in (11), using the train set and test1 set only. In the second iteration, we calculated the same metric values but using train set and test2 set only. For the first iteration, the Criterion value or the value of the optimal epoch is found to be approximately 65th epoch. For the second iteration, the Criterion value is found to be approximately 55th epoch. We then determined the mean optimal epoch, mean($\text{Criterion}_{\text{train-test1}}$, $\text{Criterion}_{\text{train-test2}}$) to give the optimal value of the epoch to be approximately equal to 60th epoch.

Our custom model produced the following MSE and MAE pairs of values of 0.205, 0.361, 0.197 and 0.350 from test1 and test2 datasets tested against the same train set respectively at around 60th epoch number. The small variations among the values support the claim that the values have been generalized sufficiently.

Table 7.1 shows the table for accuracy comparison of our model against ARIMA model as well as the other frequently used statistical models in stock market assets coefficient estimation for portfolio analysis and management.

| | MSE | MAE |
|---|---|---|
| ARIMA-LSTM Test1 | 0.205 | 0.361 |
| ARIMA-LSTM Test2 | 0.197 | 0.350 |
| Constant Correlation Test1 | 0.225 | 0.422 |
| Constant Correlation Test1 | 0.265 | 0.395 |
| Historical Model Test1 | 0.426 | 0.528 |
| Historical Model Test2 | 0.355 | 0.484 |
| Single Index Model Test1 | 0.338 | 0.488 |
| Single Index Model Test1 | 0.305 | 0.450 |
| ARIMA Test1 | 0.456 | 0.571 |
| ARIMA Test2 | 0.453 | 0.568 |

Table 7.1: Accuracy comparison of ARIMA-LSTM hybrid model against standard models

The study above shows the calculated MSE and MAE for our hybrid model performance against the standard ARIMA model as well as a few other traditional statistical models stated above. The traditional statistical models that we will use for comparing with our model are Historical Model, Constant Correlation Model and Single Index Model. Both the MSE and MAE values for our model has been significantly low enough against the other statistical models. This displays that our model has shown the best accuracy in forecasting correlation coefficients for a set of stock assets in a portfolio.

# Chapter 8

# Conclusion

The portfolio of stock market companies is an important aspect for investors to determine the risk and returns of investment. Modern Portfolio Theory suggests that a portfolio of stock companies optimizing returns for a certain level of risk can be built by an investor. Similarly, for a desired amount of return, a portfolio can also be constructed that will involve the lowest possible risk. By determining the correlation coefficients between pair of stock companies, we can find how strongly or weakly related they are. Depending on this an investor can then build a portfolio that serves his interest best. The ability to forecast the correlation coefficients in near future is thus of great significance. In this paper, we introduced ARIMA-LSTM hybrid model as an alternative to traditional statistical models for predicting correlation coefficients and with the help of empirical study showed that it performs better than the traditional models. Also we compared accuracy of our hybrid model with the ARIMA model alone and found the hybrid model to perform better.

However, there were some limitations to our study. As different companies were enlisted for stock trading at different times, the range of data available was not equal for all companies. For this inconsistency it was not possible for us to find correlations between companies which had huge difference in data. Therefore, we had to remove companies from the research that did not have data starting from our desired time. Also there was a good amount of data missing in some company datasets we selected. We had to fill the missing values with previous values in the dataset. If data was consistent we could have optimized our model more.

For future work, we plan to make a program capable of scraping daily stock index data in real time and storing it in our online repository. In this way we will able to build an API containing up to data stock prices and overtime construct a consistent dataset.

In this paper our goal was to find how the hybrid model performs compared to the other models. We did not build company portfolios here. For future research we plan to cluster the correlation coefficients obtained from prediction and group companies based on the correlation between them. For this we will take several time ranges into consideration and separate the companies based on data availability.

Furthermore, we only worked on Dhaka Stock Exchange data in this paper but in future we are aiming to work with Chittagong Stock Exchange data as well to develop a comprehensive cross stock exchange market asset portfolio to encompass any deviations in correlation between same assets in different stock market. If there are deviations, we would like to investigate the effect of averaging the correlations in cross market against individual market to better understand the portfolio optimization.

# Bibliography

[1] M. H. M, “Portfolio selection”, *The Journal of Finance*, vol. Vol, no. 9, pp. 77–91, 1952.

[2] S. W.F., “A simplified model for portfolio analysis”, *Management Science*, vol. Vol, no. 13, pp. 277–293, 1963.

[3] J. G. Box G.E.P., “Time series analysis, forecasting and control, holden-day”, *San Francisco, CA*, 1970.

[4] B. Solnik, “Note on the validity of the random walk for european stock prices. journal of finance”, vol. Vol, no. 28, pp. 1151–1159, 1973.

[5] G. C. Newbold P., “Experience with forecasting univariate time series and the combination of forecasts”, *Journal of the Royal Statistical Society Series A*, vol. Vol, no. 137, pp. 131–164, 1974.

[6] K. K. Tanigawa T., “Stock price pattern recognition - a recurrent neural network approach”, *1990 IJCNN International Joint Conference on Neural Networks*, vol. Vol, no. 1, pp. 215–221, 1990.

[7] F. P. Tang Z. Almeida C., “Time series forecasting using neural networks vs box–jenkins methodology”, *Simulation*, vol. Vol, no. 57, pp. 303–310, 1991.

[8] K. K. De Gooijer J. G., “Some recent developments in non-linear time series modelling, testing, and forecasting”, *International Journal of Forecasting*, vol. Vol, no. 8, pp. 135–156, 1992.

[9] D. J.W., “How good are neural networks for causal forecasting?”, *The Journal of Business Forecasting Methods  Systems*, vol. Vol, no. 14, pp. 17–20, 1995.

[10] E. E. Gruber M. J., “Modern portfolio theory, 1950 to date”, *Journal of Banking and Finance*, vol. Vol, no. 21, pp. 1743–1759, 1997.

[11] Z. G. P, “Time series forecasting using a hybrid arima and neural network mode”, *Neurocomputing*, vol. Vol, no. 50, pp. 159–175, 2001.

[12] H. M. Y. Zhang G. P. Patuwo E. B., “A simulation study of artificial neural networks for nonlinear time-series”, *Computers  Operations Research*, vol. Vol, no. 28, pp. 381–396, 2001.

[13] G. F. Markowitz H. M. Fabozzi F. J., “The legacy of modern portfolio theory. the journal of investing”, vol. Vol, no. 11, pp. 7–22, 2002.

[14] H. J. Nelson R. D., “Time-series analysis with neural networks and arima-neural network hybrids”, *Journal of Experimental And Theoretical Artificial Intelligence*, vol. Vol, no. 15, pp. 315–330, 2003.

[15] H. E. Rahman S., “Weak-form efficiency: Testimony of dhaka stock exchange journal of business research”, vol. Vol, no. 8, pp. 1–12, 2006.

[16] A. M. M. Uddin M. G. S. Alam K. A., "Market depth and risk return analysis of dhaka stock exchange: An empirical test of market efficiency", *ASA University Review*, vol. Vol, no. 1, pp. 93–101, 2007.

[17] H. A. S., "Forecasting dhaka stock exchange (dse) return: An autoregressive integrated moving average (arima) approach", *North South Business Review*, vol. Vol, no. 3, pp. 36–54, 2009.

[18] S. M. Hossain M. M. Rajeb M., "Time series analysis of the general index of dhaka stock exchange in bangladesh: A comparative study of garch and arima models", *2nd International Conference on Business and Economic Research*, vol. Vol, no. 5, pp. 2756–2769, 2011.

[19] F. Z, "Forecast correlation coefficient matrix of stock returns in portfolio analysis", *UCLA Electronic Theses and Dissertations*, 2013.

[20] "Dropout: A simple way to prevent neural networks from overfitting", *Journal of Machine Learning Research*, vol. Vol, no. 15, pp. 1929–1958, 2014.

[21] J. H., "Investigation into the effectiveness of long short term memory networks for stock price prediction", 2016.

[22] T. D. C. Sen J., "Decomposition of time series data of stock markets and its implications for prediction – an application for the indian auto sector", *Advances in Business Research and Management Practices (ABRMP'2016)*, 2016.

[23] "Stock price prediction based on stock big data and pattern graph analysis", *International Conference on Internet of Things and Big Data*, 2016.

[24] P. A. C. M. de Oliveira R. A. Nelson D. M. Q., "Stock markets price movement prediction with lstm neural network", *International Joint Conference on Neural Networks (IJCNN)*, pp. 1419–1426, 2017.

[25] V. S. Roondiwala M. Patel H., "Predicting stock prices using lstm", vol. Vol, no. 6, pp. 1754–1756, 2017.

[26] N. A. S. Namini S. S. Tavakoli N., "A comparison of arima and lstm in forecasting time series", *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2018.

[27] C. P., "Forecasting market movement in dhaka stock exchange: Lstm vs. arima", 2018.

[28] "Stock price prediction using long short term memory", *International Research Journal of Engineering and Technology (IRJET)*, vol. Vol, no. 5, pp. 3342–3348, 2018.

[29] D. K. Salem F., "Performance of the three slim variants of the long short-term memory (lstm) layer", *Michigan State University*, 2019.

[30] G. S. Mondal P. Shit L., "Study of effectiveness of time series modeling (arima) in forecasting stock prices", vol. Vol, no. 4, pp. 13–29,

# Appendix A

Table 8.1: List of tickers of 15 randomly selected Dhaka Stock Exchange stocks

| | | |
|---|---|---|
| ABBANK | ACI | APEXFOODS |
| BEXIMCO | BGIC | EHL |
| GREENDELT | MONNOCERA | SINGERBD |
| SINOBANGLA | AFTABAUTO | AMBEEPHA |
| AMCL(PRAN) | APEXFOOT | PUBALIBANK |

# Appendix B

## 8.1 LSTM Model Source Code

import pandas as pd
import numpy as np
import os from keras.models
from keras.layersimport Sequential, load$_{m}odel$
$fromkerasimportDense, LSTM, Activation$
$fromkeras.utils.generic_{u}tilsimportbackendasK$
$fromkeras.callbacksimportget_{c}ustom_{o}bjects$
$fromkeras.regularizersimportModelCheckpoint$
$importl1_{l}2$
$Train - Dev - TestGeneration$
$train_X = pd.read_{c}sv(' /train_{d}ev_{t}est/after_{a}rima/train_X.csv')$
$dev_X = pd.read_{c}sv(' /train_{d}ev_{t}est/after_{a}rima/dev_X.csv')$
$test1_X = pd.read_{c}sv(' /train_{d}ev_{t}est/after_{a}rima/test1_X.csv')$
$test2_X = pd.read_{c}sv(' /train_{d}ev_{t}est/after_{a}rima/test2_X.csv')$
$train_Y = pd.read_{c}sv(' /train_{d}ev_{t}est/after_{a}rima/train_Y.csv')$
$dev_Y = pd.read_{c}sv(' /train_{d}ev_{t}est/after_{a}rima/dev_Y.csv')$
$test1_Y = pd.read_{c}sv(' /train_{d}ev_{t}est/after_{a}rima/test1_Y.csv')$
$test2_Y = pd.read_{c}sv(' /train_{d}ev_{t}est/after_{a}rima/test2_Y.csv')$
STEP = 20

$\mathrm{train}_X = np.asarray(train_X).reshape((int(10500/STEP), 20, 1))$

$_dev_X = np.asarray(dev_X).reshape((int(10500/STEP), 20, 1))$

$_test1_X = np.asarray(test1_X).reshape((int(10500/STEP), 20, 1))$

$_test2_X = np.asarray(test2_X).reshape((int(10500/STEP), 20, 1))$

$_train_Y = np.asarray(train_Y).reshape(int(10500/STEP), 1)$

$dev_Y = np.asarray(dev_Y).reshape(int(10500/STEP), 1)$

$_test1_Y = np.asarray(test1_Y).reshape(int(10500/STEP), 1)$

$_test2_Y = np.asarray(test2_Y).reshape(int(10500/STEP), 1)$

$ModelGeneration$

$model = Sequential()$

$model.add(LSTM(25, input_shape = (20, 1)))$

$model.add(Dense(1))$

$model.compile(loss =' mean_squared_error', optimizer =' adam',$

$metrics = ['mse','mae'])$

$model_scores =$

$epoch_num = 1$

$for_inrange(80) :$

$trainthemodel$

$dir =' \ /models/hybrid_LSTM' file_list = os.listdir(dir)$

$iflen(file_list)! = 0 :$

$epoch_num = len(file_list) + 1$

$recent_model_name =' epoch' + str(epoch_num - 1) +' .h5'$

$filepath =' \ /models/hybrid_LSTM/' + recent_model_name$

$model = load_model(filepath)$

$filepath =' \ /models/hybrid_LSTM/epoch' + str(epoch_num) +' .h5'$

$checkpoint = ModelCheckpoint(filepath, monitor =' loss', verbose = 1,$

$save_best_only = False, mode =' min')$

$callbacks_list = [checkpoint]iflen(callbacks_list) == 0 :$

$model.fit(_train_X,_train_Y, epochs = 1, batch_size = 500, shuffle = True)$

$else : model.fit(_train_X,_train_Y, epochs = 1, batch_size = 500, shuffle = True, callbacks =$

$callbacks_{l}ist)$

$testthemodel$

$score_{t}rain = model.evaluate(_{t}rain_{X},_{t}rain_{Y})$

$score_{d}ev = model.evaluate(_{d}ev_{X},_{d}ev_{Y})$

$getformerscoredata$

$df = pd.read_{c}sv('\ /models/hybrid_{L}STM.csv')$

$train_{m}se = list(df['TRAIN_{M}SE'])$

$dev_{m}se = list(df['DEV_{M}SE'])$

$train_{m}ae = list(df['TRAIN_{M}AE'])$

$dev_{m}ae = list(df['DEV_{M}AE'])$

$appendnewdata$

$train_{m}se.append(score_{t}rain[1])$

$dev_{m}se.append(score_{d}ev[1])$

$train_{m}ae.append(score_{t}rain[2])$

$dev_{m}ae.append(score_{d}ev[2])$

$organizenewlycreatedscoredataset$

$model_{s}cores['TRAIN_{M}SE'] = train_{m}se$

$model_{s}cores['DEV_{M}SE'] = dev_{m}se$

$model_{s}cores['TRAIN_{M}AE'] = train_{m}ae$

$model_{s}cores['DEV_{M}AE'] = dev_{m}ae$

$savenewlycreatedscoredataset$

$model_{s}cores_{d}f = pd.DataFrame(model_{s}cores)$

$model_{s}cores_{d}f.to_{c}sv('\ /models/hybrid_{L}STM.csv')$