Learning to predict cryptocurrency price using artificial neural network models of time series

by

Sneha Gullapalli

B.Tech., Jawaharlal Nehru Technological University, India, 2016

A REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computer Science
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2018

Approved by:

Major Professor
Dr. William H. Hsu

Copyright

Abstract

Cryptocurrencies are digital currencies that have garnered significant investor attention in the financial markets. The aim of this project is to predict the daily price, particularly the daily high and closing price, of the cryptocurrency Bitcoin. This plays a vital role in making trading decisions. There exist various factors which affect the price of Bitcoin, thereby making price prediction a complex and technically challenging task. To perform prediction, we trained temporal neural networks such as time-delay neural networks (TDNN) and recurrent neural networks (RNN) on historical time series – that is, past prices of Bitcoin over several years. Features such as the opening price, highest price, lowest price, closing price, and volume of a currency over several preceding quarters were taken into consideration so as to predict the highest and closing price of the next day. We designed and implemented TDNNs and RNNs using the *NeuroSolutions* artificial neural network (ANN) development environment to build predictive models and evaluated them by computing various measures such as the MSE (mean square error), NMSE (normalized mean square error), and r (Pearson's correlation coefficient) on a continuation of the training data from each time series, held out for validation.

Table of Contents

List of Figures

List of Tables

Acknowledgements

First and foremost, I would like to express my heartfelt gratitude to my major professor Dr. William H. Hsu of the Department of Computer Science at Kansas State University for his excellent guidance on this project. I am extremely indebted to him for being supportive of my career goals and encouraging me in every way possible. My sincere thanks to my committee members Dr. Mitch Neilsen and Dr. Daniel Andresen for taking time to serve on my committee.

I would like to extend huge thanks to all my family members without whom this journey would not have been possible. I am extremely grateful to my mother Prameela Rani Gullapalli and my sister Sindhu Gullapalli for always inspiring me to follow my dreams and for all the love, encouragement, and emotional support in my life.

Special thanks to my roommates, Poojitha Bikki, Sharmila Vegesana, Sindhu Reddy Velumula, and Sravani Donepudi, who have always been there for me. I wish to thank my good friends Poojitha Maganti, Ravi Raja Mallala, Pruthvidhar Reddy Dhodda, Nithin Kumar Kakkireni, and especially Pavan Harshit Manepalli for all the help and support. Last, by no means least, I would like to thank my fellow lab mates at the Laboratory for Knowledge Discovery in Databases (KDD Lab) with whom I had the pleasure of working during this period.

Chapter 1 - Introduction

1.1 Motivation and Project Overview

Cryptocurrency has become increasingly popular over the few years. Bitcoin is one of the most popular and widely known cryptocurrency. It is mainly designed to remove the need of any third-party entities or financial institutions and thereby eliminate the possibility of fraud during the transactions. One significant and unique characteristic of Bitcoin is the multi-version concurrency control that allows safe concurrent transactions without any significant delay.

Cryptocurrency has a total market cap of around $600 billion USD by the end of December 2017 with Bitcoin having a market cap of around $300 billion USD share in the cryptocurrency market. Not only the investors but also brokers and private investors are finding cryptocurrency as an investment tool. In this regard, it is very much necessary to predict the future values of cryptocurrency so as to take correct trading decisions. In the past years, traditional statistical methods such as linear regression were popular. However, due to uncertainty in the trends of financial markets, computing techniques such as neural networks have become quite popular. Artificial Neural networks can adjust by itself based on the information given to it. They have the capability to capture the non-linear trends of the financial markets.

1.2 Goal and Technical Objective

The goal of this project is to predict the highest and closing price of Bitcoin on a given day based on the Bitcoin data of several preceding quarters. It is technically challenging to predict the accurate price, mainly due to lack of seasonality and highly volatile nature of the cryptocurrency market. This is basically a time series prediction problem. Artificial neural network (ANNs) models of time series is used to perform the prediction task, mainly due to the ability of ANNs to deal with non-linearities in the data such as lack of seasonality. Temporal neural network architectures such as time-delay neural networks (TDNN) and recurrent neural networks (RNN) are used for prediction task in this study. These two models are trained and tested on Bitcoin data starting from 2012 till the first quarter of 2018.

In order to make the one day ahead prediction of highest and closing price of Bitcoin, features such as open price, high price, low price, close price and volume of currency (USD) are taken into consideration. To predict the highest and closing price on a day of quarter, both the neural network models are trained with data over the past eight quarters and it is tested over the next quarter. Comparisons between both the TDNN and RNN models are presented with reference to MSE (mean square error), NMSE (normalized mean square error), and r (Pearson's correlation coefficient).

The document explains the data preparation steps followed by the neural network models and their functionality. Quantitative measures like to MSE (mean square error), NMSE (normalized mean square error), and r (Pearson's correlation coefficient) are taken and compared for TDNN and RNN models (Billard, A. et al., 2016). The predicted high and closing price using these two neural networks are presented in tabular format. At the end, the report discusses possible improvements that can be made to increase the scope of the experiment.

Chapter 2 - Background Study and Related Work

This chapter discusses related work done in the area of predicting the Bitcoin price using various approaches. Also, we will look at how the current approach evolved based on past motivating use cases.

2.1 Literature Survey

Various approaches have been used in the past to carry out the price prediction task. There are mainly two sets of literature that are highly relevant to this work. One is financial data analysis; the other, time series data analysis.

2.1.1 Financial Data Analysis

Several approaches are described in the literature including, one called technical analysis also known as "charting" that forecasts future prices (Lo et al., 2000). According to it, stock market prices do not follow random walks, that is – the price movements follow a set of patterns. These price movements can be used to predict the future price (Lo & MacKinlay, 1988, 1999). There exist some other empirically designed patterns such as heads-and-shoulders, double-top-and-bottom that can be used to predict future prices. We refer the interested reader to the work of Lo et al. (2000). In this paper, authors have used kernel based regression techniques to find out the patterns in historical data, that is – price is predicted based on past data. This work (Lo et al., 2000) is theoretically close to the current project work. However, it does not employ the same strategies followed in the current project.

2.1.2 Time Series Data Analysis

In the context of future price predictions, classical methods are quite popular. Autoregressive integrated moving average (ARIMA) models are a popular choice for forecasting over a short term. It works very well when the data exhibits consistent or stable pattern over time with least possible outliers. The ARIMA methodology works well only when the data exhibits “stationarity”, which means that the series remains almost constant. But this is not always possible in the real time scenario, where the data fluctuates drastically, and it is highly volatile. Ediger and Akar used the seasonal ARIMA model to estimate the future fuel energy demand in Turkey over certain years. However, the similar scenario is not guaranteed to work for unseasonal or non-linear data. To solve the real time prediction problems, artificial neural networks are very much useful to increase the speed of computation due to its ability to handle nonlinearities in the data. Examples of nonlinear data include psychological data (Scheier & Tschacher, 1996).

In one of their research papers, Greaves et al., 2015 predicted the price of Bitcoin using support vector machine(SVM) and artificial neural networks (ANNs), concluded that probability of predicting the price in block chain market is challenging and is scope is limited. Despite of all this, neural networks have become a valuable tool for prediction of time series problems due to their ability to handle non-linear data and nonstationary data.

2.2 Established Methods and Approaches

The technical details and approaches used for prediction of the Bitcoin closing and the highest price are discussed in this section.

2.2.1 Artificial Neural Networks for Prediction

Artificial neural networks(ANNs) are inspired by the working of human nervous system, that is, the neurons of a human brain. It consists of multiple nodes, each of which takes input data and performs some basic operations and results are passed to the other neurons. Neurons can transmit information among themselves using links between them. Each of the links has a **weight (**synaptic weight**)** associated with it. An activation function is used to calculate whether the neuron fires.

This should be a non-linear function in order to be able to express complex patterns in the data. The following diagram illustrates a simple ANN.

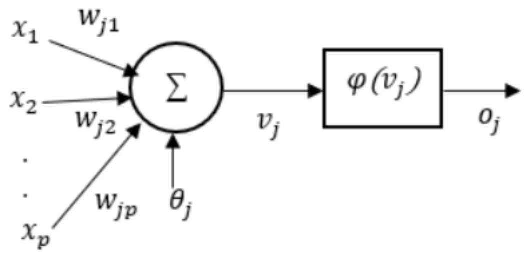

Figure 1. Representation of a simple neuron j

(Source: Marques et al., 2005)

input signals = $\{x_1, x_2, \ldots, x_p\}$

synaptic weights = $\{w_{j1}, w_{j2}, \ldots, w_{jp}\}$

bias value = θ_j

activation potential = v_j

activation function (generally non-linear sigmoid function) = $\varphi(v_j)$

o_j is the Output signal defined as follows:

$$\boldsymbol{o_j = \varphi(\theta_j + \sum_{i=1}^{p} w_{ji} x_i)} \quad (1)$$

A typical neural network has an input layer (leftmost layer) which sends stimulus to the network, output layer (rightmost layer) and all the layers between input and output layers are called hidden layers. Once the neural network is trained with data, it the weights are adjusted, and knowledge is stored. When a new input signal arrives that is not used for testing, based on the pre-calculated weights, output will be generated.

The current work focuses on two temporal neural networks, that is – Time-Delay Neural Networks (TDNN) and Recurrent Neural Networks (RNN).

2.2.2 Time-Delay Neural Network(TDNN)

Time-delay neural networks is one popular and potential methods for prediction of time series data. A simple time-delay neural network with one hidden layer can be illustrated as follows:

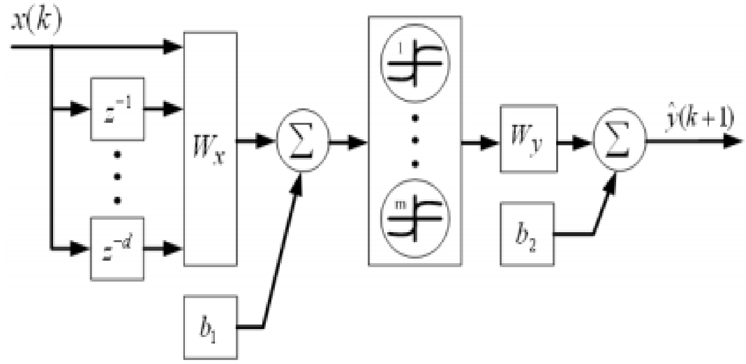

Figure 2. Simple TDNN model with one hidden layer

(Source: Molina et al., 2011)

In this study, we have used a simple time-delay neural network with one hidden layer. Past observations of a variable serve as input data, which will be used to perform one-step ahead prediction. Assuming ŷ(k+1) is the output from TDNN, it can be calculated as follows:

$$\hat{y}(\boldsymbol{k}+\mathbf{1}) = \boldsymbol{g}(\boldsymbol{W_x} * [\boldsymbol{x}(\boldsymbol{k})\ \boldsymbol{x}(\boldsymbol{k}-\mathbf{1})\ldots\boldsymbol{x}(\boldsymbol{k}-\boldsymbol{d})]^{\boldsymbol{T}} + \boldsymbol{b_1}) *\ \boldsymbol{W_y} + \boldsymbol{b_2} \qquad (2)$$

where W_x and W_y are weight vectors, b_1 and b_2 represents bias value, d denotes the time delays applied to input vector $x(k)$, g represents the activation function.

2.2.3 Recurrent Neural Network (RNN)

Recurrent neural networks make use of one or feedback connections. The idea behind recurrent neural networks is that to find the next value that might occur in the sequence, better we have an idea of which ones come before it. RNN are called “recurrent” because they perform same task on each element of the sequence and also the output of this step depends on the previous computations. The input to an RNN is based on all the previous inputs, that are used for feedback connections. From this it can be stated that the output of an RNN is function of current external input along with inputs and outputs of the previous state.

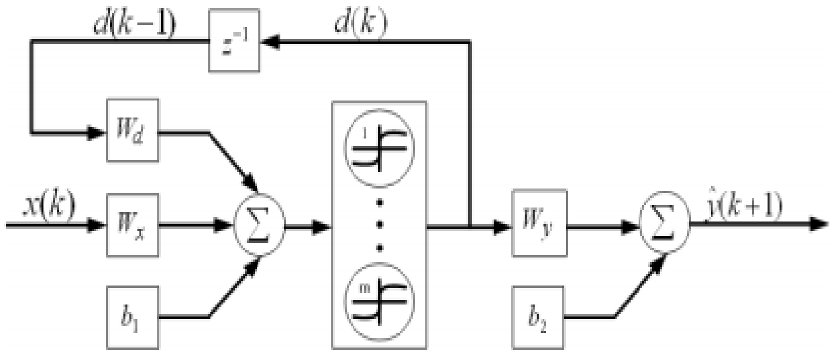

Figure 3. Simple RNN model with one hidden layer

(Source: Molina et al., 2011)

Recurrent networks are the state of the art in nonlinear time series prediction, temporal pattern classification. It can be illustrated with the following equation:

$$\hat{y}(\boldsymbol{k}+\mathbf{1}) = \mathbf{g}(\boldsymbol{W_x} * \boldsymbol{x}(\boldsymbol{k}) + \boldsymbol{W_d} * \mathbf{d(k\text{-}1)} + \boldsymbol{b_1}) * \boldsymbol{W_y} + \boldsymbol{b_2} \tag{3}$$

Here, W_x, W_y and W_d represent weight vectors, *b1* and *b2* represent bias value, *d* denotes the time delays applied to input vector $x(k)$, *d(k) and d(k-1)* represents recurrence over $x(k)$.

2.3 NeuroSolutions

To build the predictive models using which experiments are conducted in this work, *NeuroSolutions* software, an object-oriented environment for designing and deploying artificial neural networks (ANNs) solutions, is used. The environment is modular, as a neural network breaks down into fundamental set of neural components. It supports large number of neural network models starting with basic to complex neural network models. One very good thing about *NeuroSolutions* is that it provides an icon-based user interface from which the neural components are interconnectable.

Chapter 3 - Data Set

Several Bitcoin data sets are available online to download for free. Most of them provide the data related to price of Bitcoin on a minute to minute basis. However, the end goal of the project is to make one-day ahead prediction of highest and closing price of Bitcoin. So, we will need data such as highest and closing price of Bitcoin for each day over period of several years. The **Quandl API** provides the Bitcoin price data set, starting from September 2011 – 2018 (present). This API gives access to Bitcoin exchanges and daily Bitcoin values. It allows users to customize the query while using the interface to download the historical Bitcoin prices.

The data is available in three different formats i.e JSON, XML and CSV. Data is downloaded in the .csv format. Size of data is around 200KB. It has a total of 2381 data records (each record corresponds to a day) consisting of Bitcoin open, high, low, closing price and volume of Bitcoin (USD) starting from Sept 2011 – 2018 (present). However due to inconsistencies in the data from September 2011 to December 2011, this data has been discarded and data records starting from January 2012 – March 2018 are taken into consideration for this project. So, after the data is cleaned, the final data set has a total of 2271 data records.

The total data records are divided into three (3) sets, namely: Y12-13 –2012 and 2013 data, Y14-15 – 2014 and 2015 data, Y16-17 – 2016 and 2017 data. Y12-13 has eight quarters and the neural networks (TDNN, RNN) will be trained on this data and tested on the first quarter of 2014. Similarly, Y14-15 has eight quarters and the neural networks will be trained on this data and tested on the first quarter of 2016. In the same way, Y16-17 has eight quarters and neural networks are trained on this data and tested on the first quarter of 2018.

Data Set Name	Training Data	Test Data
Y16-17	2016, 2017 data	Q1 of 2018
Y14-15	2014, 2015 data	Q1 of 2016
Y12-13	2012, 2013 data	Q1 of 2014

Table 1. Partitioning of total records into three (3) sets

To predict the highest and closing price of Bitcoin one day ahead, in each of the sub data sets, columns high and close are shifted up by one (1) unit. In the three sub data sets, it should be noted that the testing data is from 1st January to 18th March and it is predicted on 19th March (of years 2014, 2016, 2018) for three sets respectively.

The data set has limited features and in the current project almost all these features are considered valuable for the prediction task. To be clear, for predicting the highest and closing price of Bitcoin one step ahead, features such as open, high, low, closing price and volume of Bitcoin (USD) are used.

Chapter 4 - Implementation and Experimental Design

4.1 Implementation Steps

The diagram below illustrates the steps followed to perform one-day ahead prediction of highest and closing price of Bitcoin.

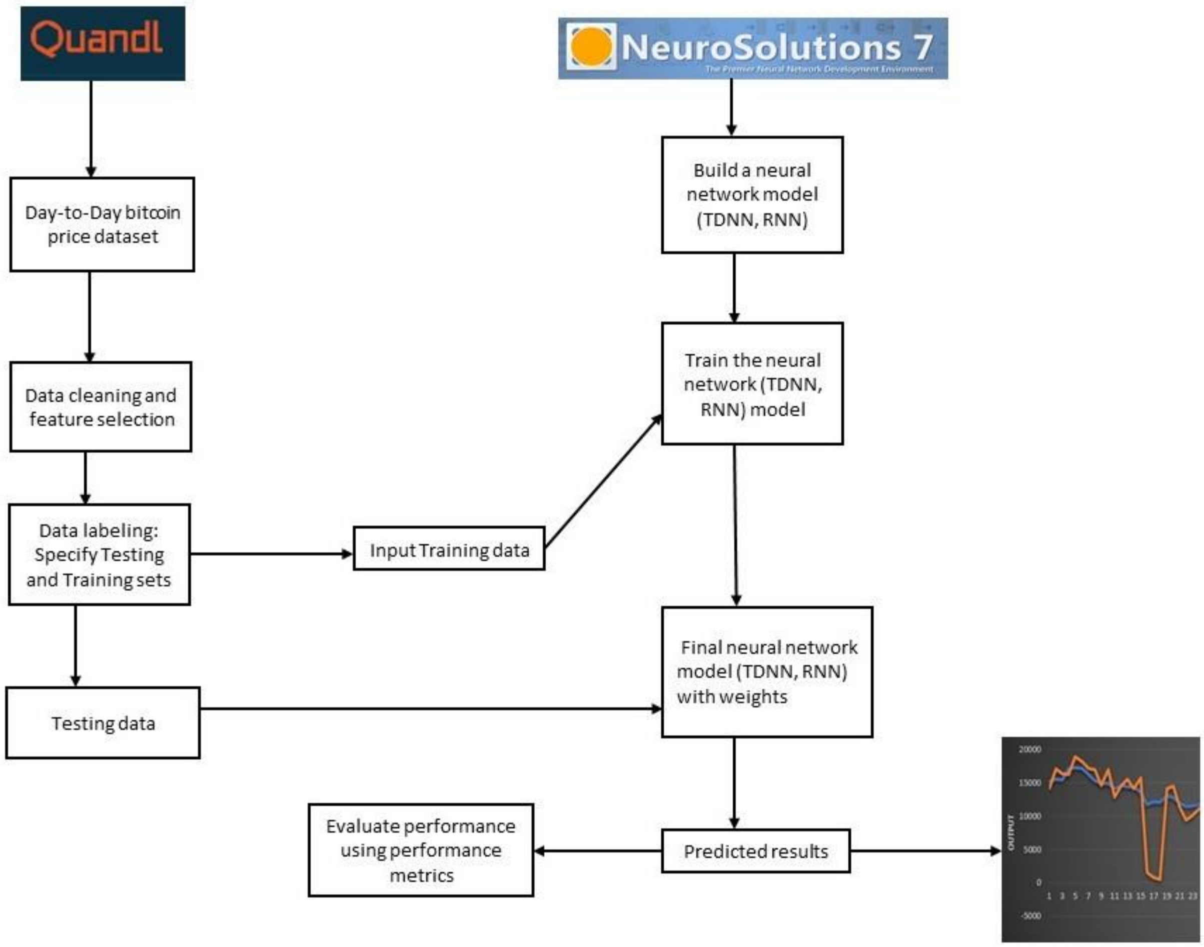

Figure 4. Implementation pipeline using neural networks to predict Bitcoin price

The work flow diagram above depicts the steps of the process, and their order for downloading and cleaning the data set (explained in chapter - 3) to evaluating the results.

4.2 Experimental Design

This section explains the various experiments performed on the Bitcoin data set using the neural networks such as Time-delay neural networks (TDNN) and RNN (Recurrent neural networks). These networks are built in the *NeuroSolutions* development environment. It provides user-friendly icon-based interface to build the neural networks. While training the neural networks, an efficient training scheme is achieved by using *Levenberg-Marquardt* learning rule and *tanh* non-linearity function is used.

4.2.1 Levenberg-Marquardt Algorithm (LMA)

An important step during the training process is the optimization of network parameters such as weights based on the input-output sets being trained on the neural network. There exist schemes like backpropagation algorithm, Momentum etc. but the most efficient one is Levenberg-Marquardt algorithm (LMA) well known for minimizing the mean square error (MSE) of the network. It determines best direction to move weights so as to minimize the error. LMA uses *Gauss-Newton* approximation for minimizing sum of squares of non-linear functions.

According to Newton's method, weight update rule is given by:

$$\boldsymbol{w_{n+1}} = \boldsymbol{w_n} - \boldsymbol{H_n^{-1} g_n} \tag{4}$$

Where w is matrix of network weights, n is the step of iterations, H is the Hessian matrix and g is called gradient matrix. Usually hessian matrix is approximated in terms of Jacobian matrix J, that has first order derivatives of network errors in context of weights and biases. Therefore, H can be expressed in terms of J as follows:

$$\boldsymbol{H} \cong \boldsymbol{J^T J} \tag{5}$$

By substituting H in above equation of weights, we get the Gauss-Newton method as follows:

$$\boldsymbol{w_{n+1}} = \boldsymbol{w_n} - [\boldsymbol{J_n^T J_n}]^{-1} \boldsymbol{g_n} \tag{6}$$

One drawback in Gauss-Newton method is that it is possible that the matrix $[J^TJ]$ does not have an inverse. This can be overcome by making slight modification as shown below, which leads to LMA:

$$\boldsymbol{w_{n+1}} = \boldsymbol{w_n} - [\boldsymbol{J_n^T J_n} + \boldsymbol{\mu_n I}]^{-1} \boldsymbol{g_n} \tag{7}$$

Here, I is an identity matrix and μ is a scalar that has an important role in LMA. If $\mu_n = 0$, the weight update is same as Gauss – Newton method. If μ_n is large, it is similar to gradient descent with small step size. So, it is important to choose a proper value of μ in LMA which helps to optimize the great performance of Gauss-Newton method and great convergence of gradient descent method.

4.2.2 Transfer/Activation function: tanh

The hyperbolic tangent function maps the range of each neuron in the layer to between -1 to +1. This allows the network to apply a value to the node (negative) instead of node not having to fire at all. This provides network ability to make soft decisions. Illustration is shown below

Activation function: It is given by the following equation

$$\boldsymbol{f(x_i, w_i)} = \boldsymbol{tanh(\beta x_i)} \tag{8}$$

The value of β is set to one (1) for both TDNN and RNN models.

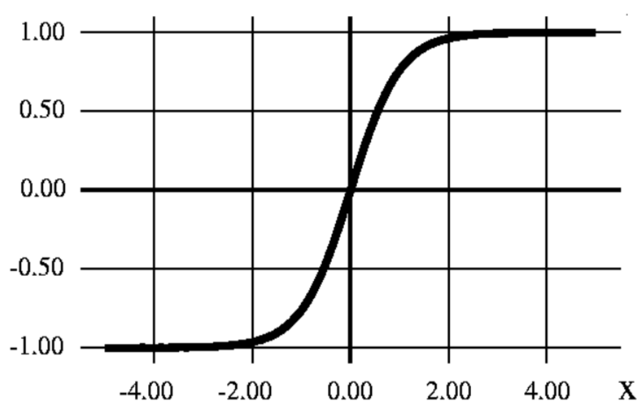

Figure 5. Illustration of tanh activation function

(Source: www.towardsdatascience.com)

Its mathematical equation is given as follows:

$$\boldsymbol{tanh}(\boldsymbol{x}) = \frac{(\mathbf{1}-\boldsymbol{e}^{-\mathbf{2}\boldsymbol{x}})}{(\mathbf{1}+\boldsymbol{e}^{-\mathbf{2}\boldsymbol{x}})} \tag{9}$$

This function is smooth, continuous i.e. differentiable, monotonic. The drawback with sigmoid function is that a larger input gives almost zero output, preventing the next nodes from learning. This is different in case of tangent function which gives -1 for negative values allowing the subsequent nodes to learn from it. Both the networks TDNN, RNN used in this project apply the tangent(tanh) as the activation function that adds a bias variance of 0.5 to each neuron in the hidden and output layer.

4.2.3 Time-Delay Neural Network

TDNN will only have feedforward connections. The time-delay neural network is built with five (5) input processing elements(PE's), one (1) output unit and one (1) hidden layer with four (4) PE's. Both the hidden and output layer uses the Levenberg-Marquardt learning rule with tanh (hyperbolic tangent) activation function mentioned above. A ***tap delay line*** represents the input vector with current and past time steps and a ***tapped delay*** represents number of samples or delay between successive taps. This model uses three (3) time delay inputs with a tapped delay of one (1). Weights vary during the training process to minimize the error and best weights are saved. Following figure is captured when weights vary across TDNN model.

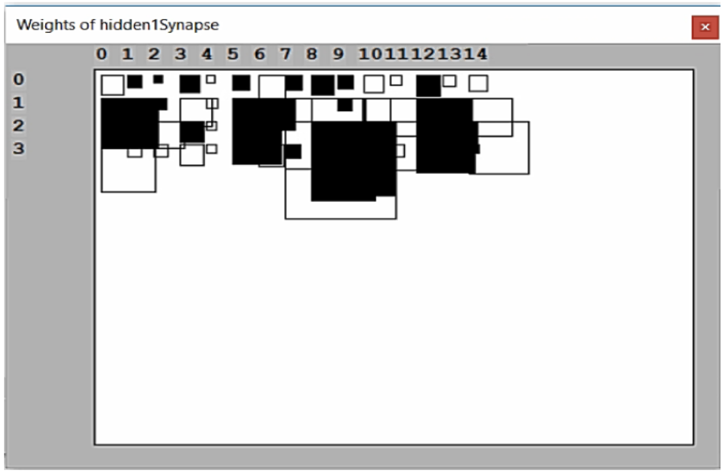

Figure 6. Instance showing variation of weights while TDNN model is being trained

Hinton probe in the probe components creates a view for observing numeric data as a matrix of squares. The size of the squares is proportional to the magnitude (the absolute value) of the data being probed. Positive values are displayed as solid squares while negative values are displayed as outlined squares. The scale used to display the squares may be changed manually, or automatically. This probe is often used to display a weight matrix. Its design makes it easy to detect patterns and symmetries in the data. Predicted results are presented in the results section below.

4.2.4 Recurrent Neural Network

RNN's have feedback connections. In this project, a partially recurrent neural network(RNN) with five (5) input processing elements(PE's), one (1) output unit and one (1) hidden layer with twelve (12) PE's. The partially recurrent structure adds a feedforward connection, through a synapse, from the input axon to the layer after the 1st hidden layer. In this case, the recurrent structure acts as a state for the feedforward structure. Both the hidden and output layer uses the Levenberg-Marquardt learning rule with tanh (hyperbolic tangent) activation function mentioned above. Weights vary during the training process to minimize the error and best weights are saved. Results predicted using this network are presented in the results section below.

Chapter 5 - Results

5.1 Performance Metrics

5.1.1 MSE (Mean Square Error)

Mean squared error is the arithmetic mean of the squares of an error function computed between estimated and actual (target) values, where the estimates here are predictions. To be clear, it can be defined as the average of square of the difference between each output processing element and the desired output. It is measure to determine how good the network output fits the desired output. If 'n' is the number of exemplars in the data set, Ŷ represents vector of n predictions, Y represents vector of observed values being predicted, the mean square error (MSE) of the predictor is given by:

$$\boldsymbol{MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2} \tag{10}$$

5.1.2 NMSE (Normalized Mean Square Error)

Normalized mean square error is one more metric to evaluate the performance of the predictor. It uses MSE during its calculation. NMSE can be simply stated as the MSE normalized by the variance of observed values. It can be formulated as follows:

$$\boldsymbol{NMSE = \frac{MSE}{VAR(Y)} = \frac{\frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2}{\frac{1}{n-1}\sum_{i=1}^{n}(Y_i - \mu)^2}} \tag{10}$$

In the above formula, n is the number of exemplars in the data set, μ is the mean of observed values. It is given by:

$$\boldsymbol{\mu = \frac{1}{n}\sum_{i=1}^{n} Y_i} \tag{12}$$

5.1.3 r (Pearson's Correlation Coefficient)

The Pearson's correlation coefficient depicts the linear association between two variables. It helps to figure if two sets of data move in the same direction. It is denoted by *r*. It can take the values from -1 to +1. If X is the network output and D is the desired output, then *r* is given by:

$$r = \frac{\sum_i (xi - \bar{x})(di - \bar{d})}{\sqrt{\sum_i \frac{(di - \bar{d})^2}{N}}\sqrt{\sum_i \frac{(xi - \bar{x})^2}{N}}} \tag{11}$$

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} xi \quad \text{and} \quad \bar{\mathrm{d}} = \frac{1}{N}\sum_{i=1}^{N} di \tag{14}$$

If $r = 0$, it indicates there is no correlation between X and D

If $r > 0$, it indicates there is positive association between X and D i.e if value of one variable increases, the other variable increases.

If $r < 0$, it indicates there is negative association between X and D i.e if value of one variable increases, the other variable decreases.

The following diagram shows the possible correlation between two variables.

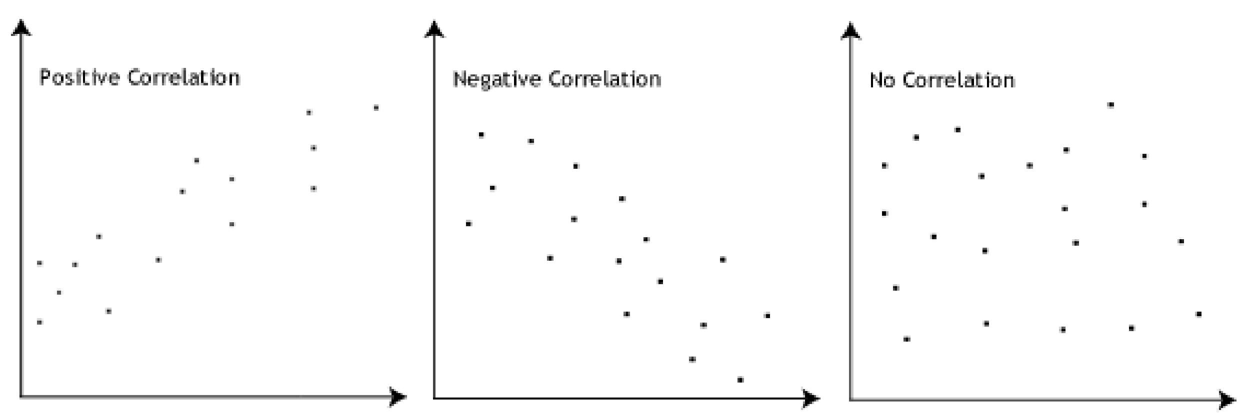

Figure 7. Illustration showing Pearson's correlation coefficient

5.2 Experimental Results

This section discusses the results which includes predicted highest and closing price for three (3) test sets, performance metrics obtained in each case and corresponding graphs.

5.2.1 Models trained on 2016-17 data and tested on first quarter of 2018

The results of TDNN and RNN model trained on data set from first quarter of 2016 to last quarter of 2017(starting from January 2016 to December 2017) and tested on first quarter of 2018 (January 2018 – March 2018). One day ahead prediction of highest and closing price is done on March 19, 2018. Both models use 1000 epochs during training and testing.

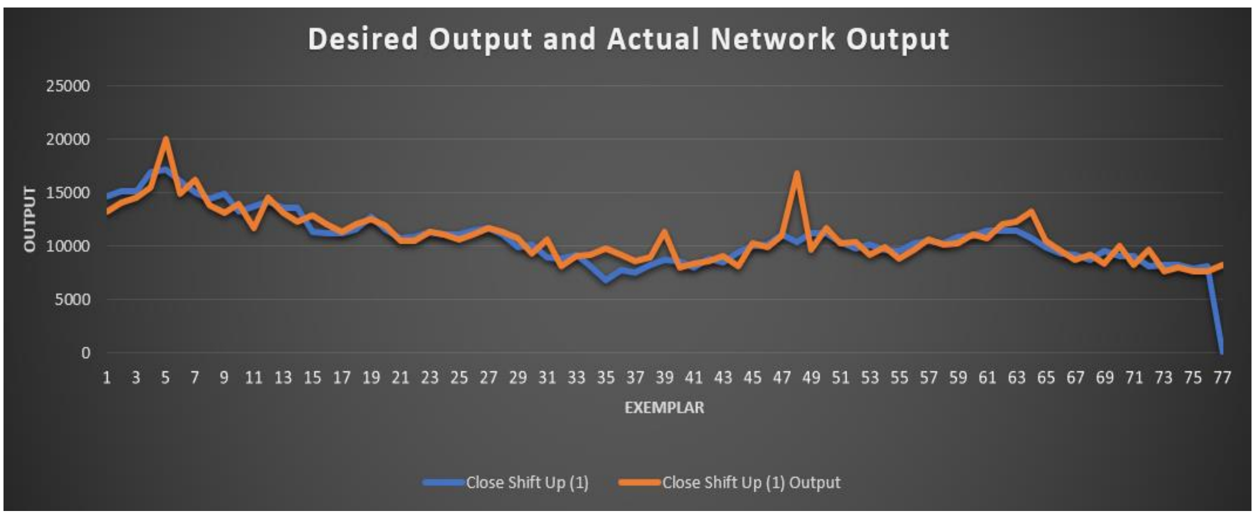

Figure 8. TDNN one-day ahead prediction of closing price (March 19, 2018)

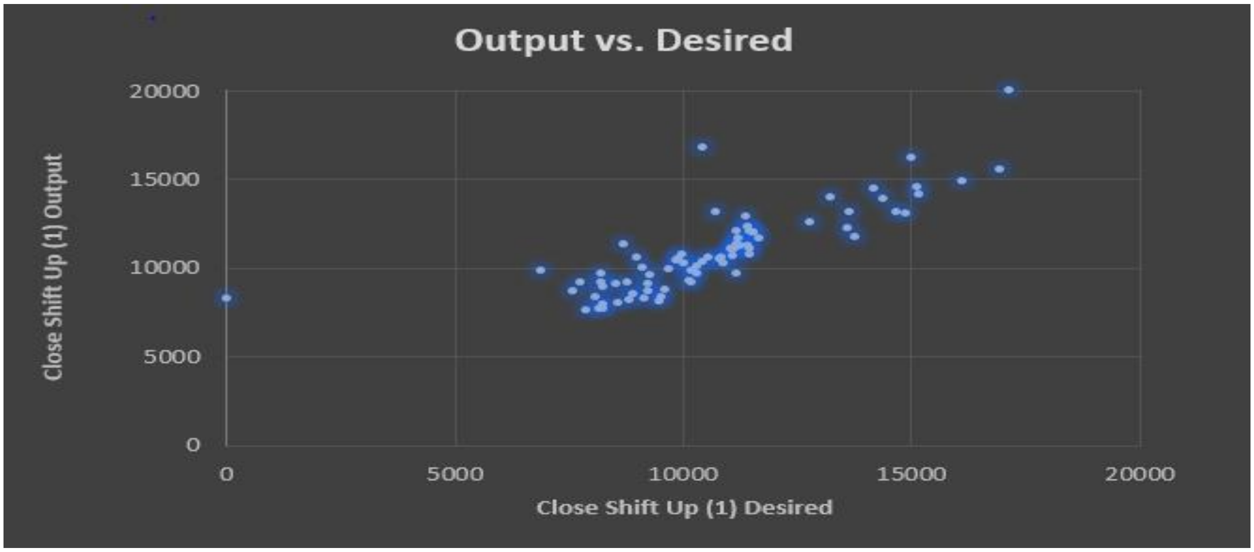

Figure 9. Scatter plot representing output vs. desired closing value using TDNN

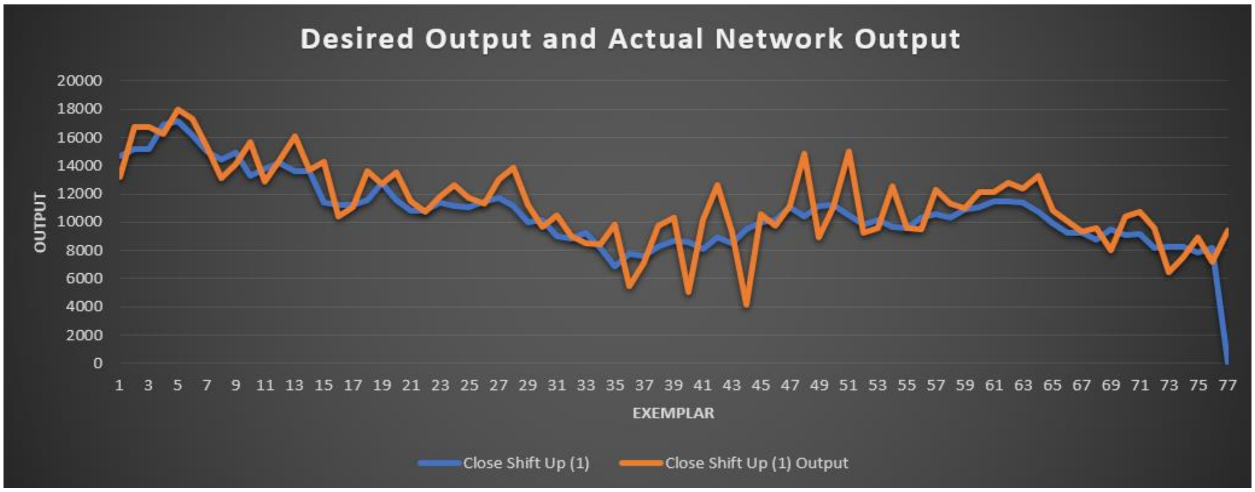

Figure 10. RNN one-day ahead prediction of closing price (March 19, 2018)

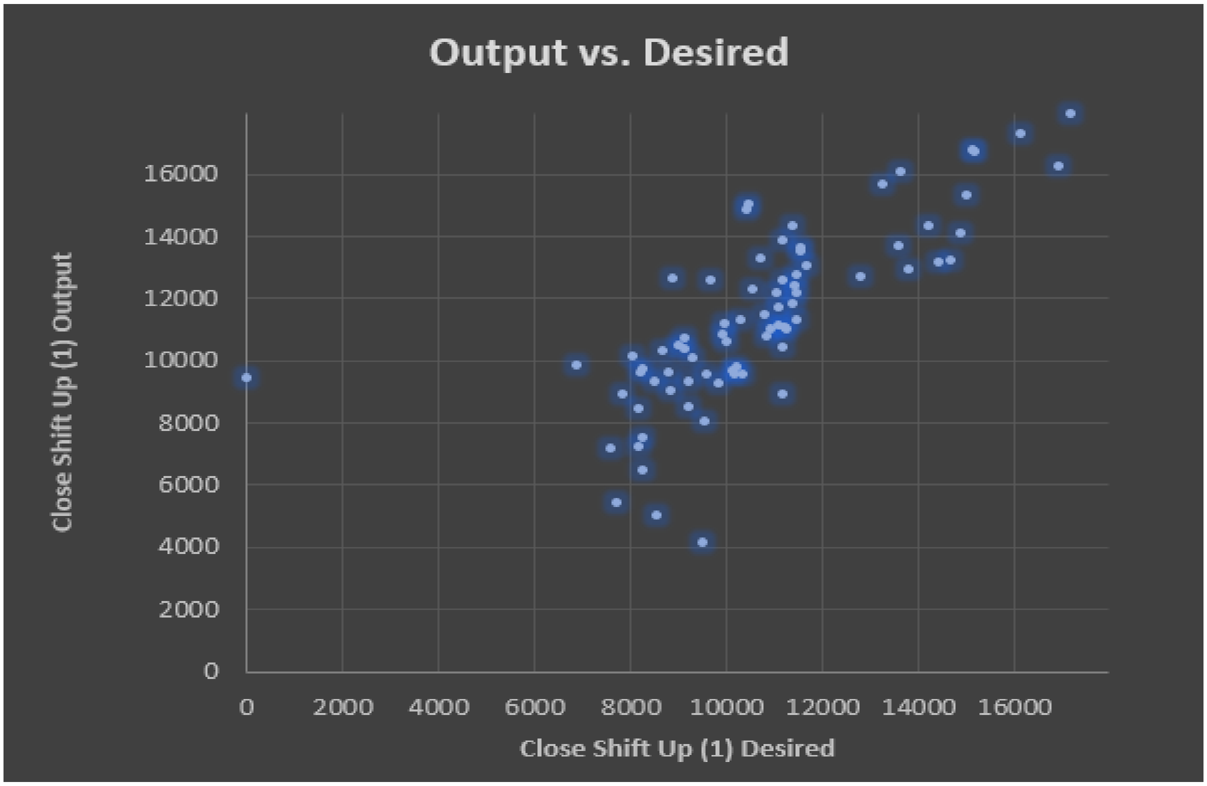

Figure 11. Scatter plot representing output vs. desired closing value using RNN

S.NO	Neural Network MODEL(close)	Models trained on 2016-17 data and tested on first quarter (Q1) of 2018		
		MSE	**NMSE**	**R**
1.	TDNN	0.0042×10^{-2}	0.083	0.809
2.	RNN	0.1267×10^{-2}	0.251	0.469

Table 2. Y16-17 Comparison of Performance metrics for models predicting closing value

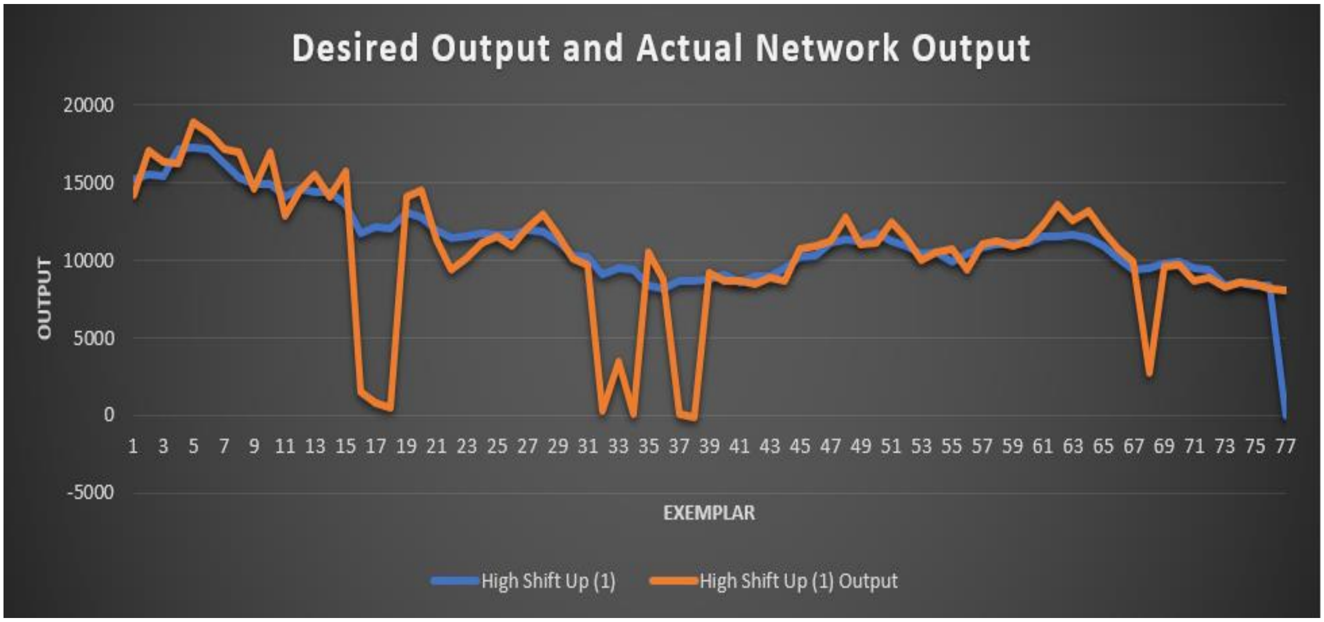

Figure 12. TDNN one-day ahead prediction of highest price (March 19, 2018)

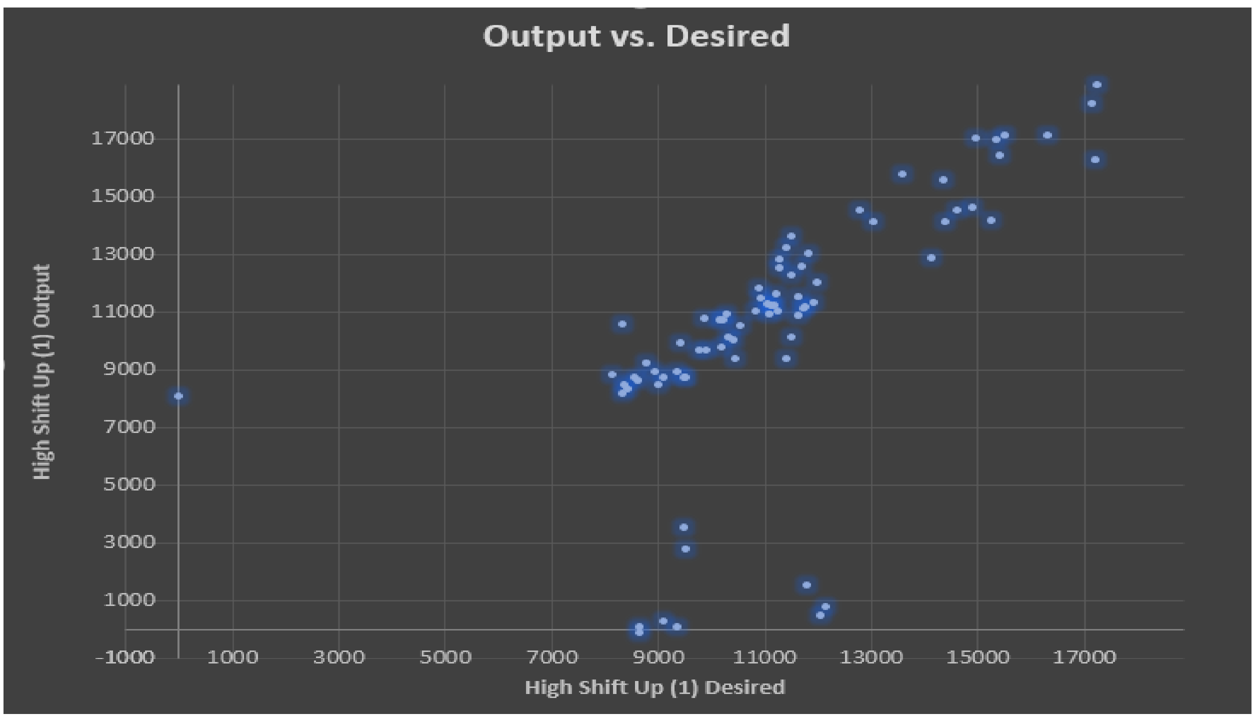

Figure 13. Scatter plot representing output vs. desired highest value using TDNN

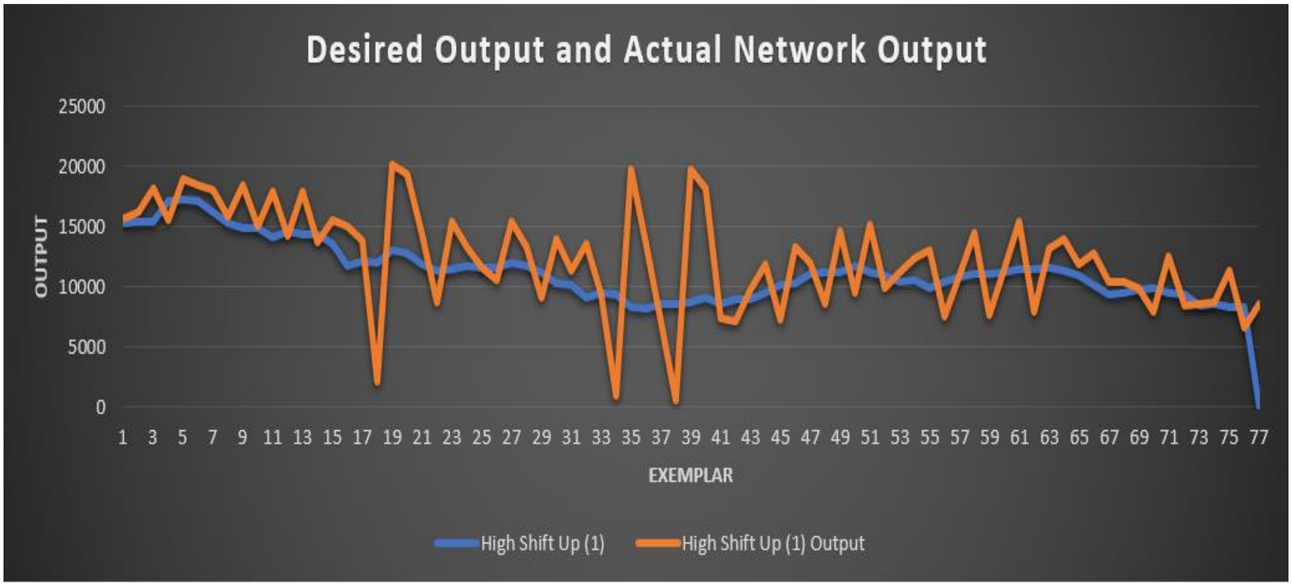

Figure 14. RNN one-day ahead prediction of highest price (March 19, 2018)

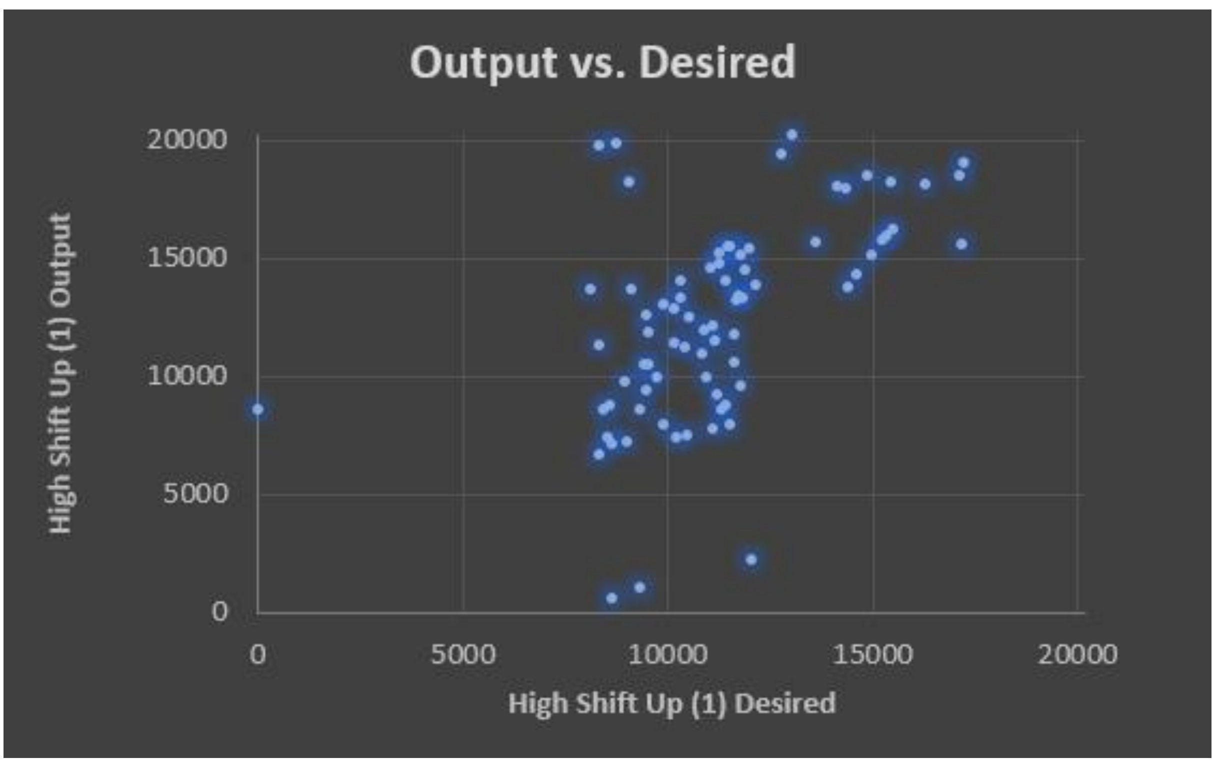

Figure 15. Scatter plot representing output vs. desired highest value using RNN

S.NO	Neural Network MODEL(high)	Models trained on 2016-17 data and tested on first quarter (Q1) of 2018		
		Final MSE	NMSE	r (Pearson's correlation coefficient)
1.	TDNN	0.024×10^{-2}	0.176	0.641
2.	RNN	2.52×10^{-2}	0.197	0.528

Table 3. Y16-17 Comparison of Performance metrics for models predicting highest value

S.NO	Neural Network MODEL	Model trained on 2016-17 data to predict 2018 Q1			
		High (predicted)	High (actual)	Close (predicted)	Close (actual)
1.	TDNN	8040.67	8718.74	8274.27	8492.91
2.	RNN	8554.15	8718.74	9423.16	8492.91

Table 4. Y16-17 One-day ahead predicted values of highest and closing price of Bitcoin

5.2.2 Models trained on 2014-15 data and tested on first quarter of 2016

The results of TDNN and RNN model trained on data set from first quarter of 2014 to last quarter of 2015(starting from January 2014 to December 2015) and tested on first quarter of 2016(January 2016 – March 2016). One day ahead prediction of highest and closing price is done on March 19, 2016. Both models use 1000 epochs during training and testing.

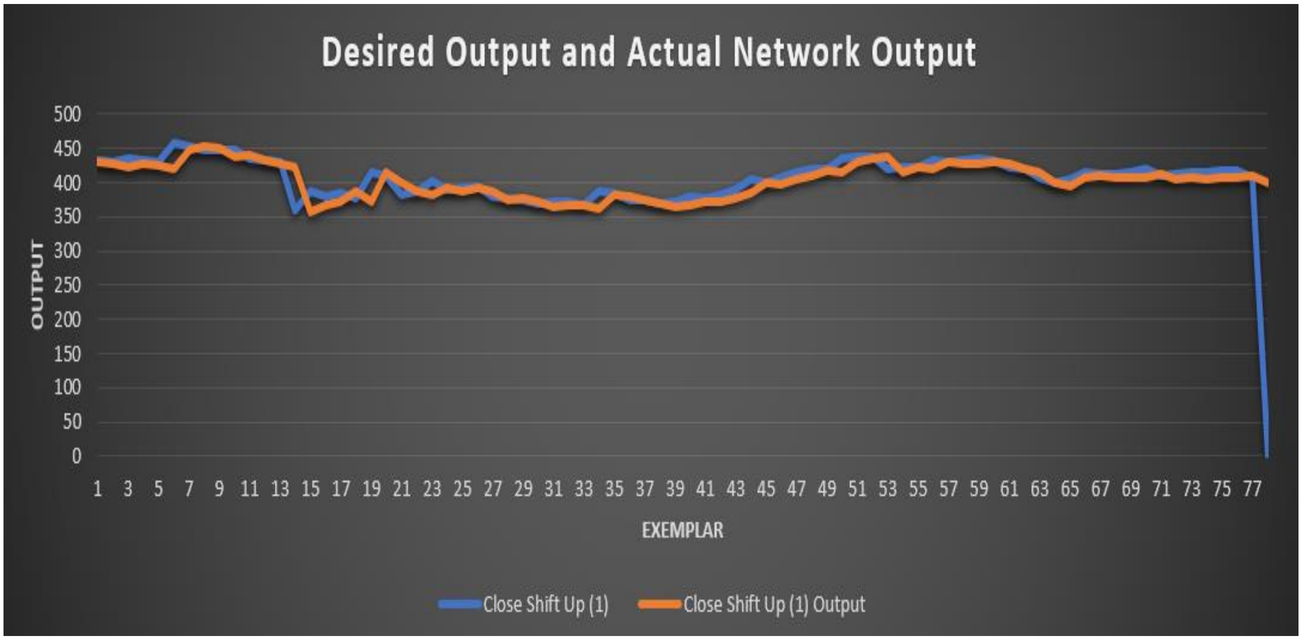

Figure 16. TDNN one-day ahead prediction of closing price (March 19, 2016)

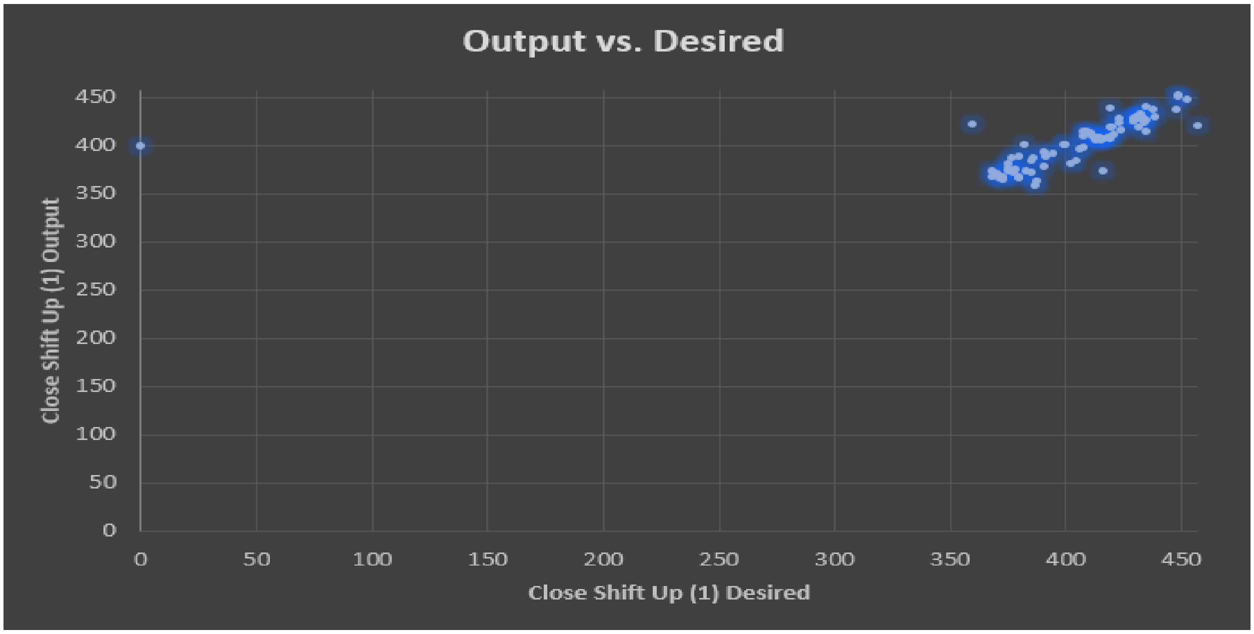

Figure 17. Scatter plot representing output vs. desired closing value using TDNN

S.NO	Neural Network MODEL(close)	Models trained on 2014-15 data and tested on first quarter (Q1) of 2016		
		Final MSE	NMSE	r (Pearson's correlation coefficient)
1.	TDNN	0.0504×10^{-2}	0.051	0.419
2.	RNN	0.898×10^{-2}	0.102	0.027

Table 5. Y14-15 Comparison of Performance metrics for models predicting closing value

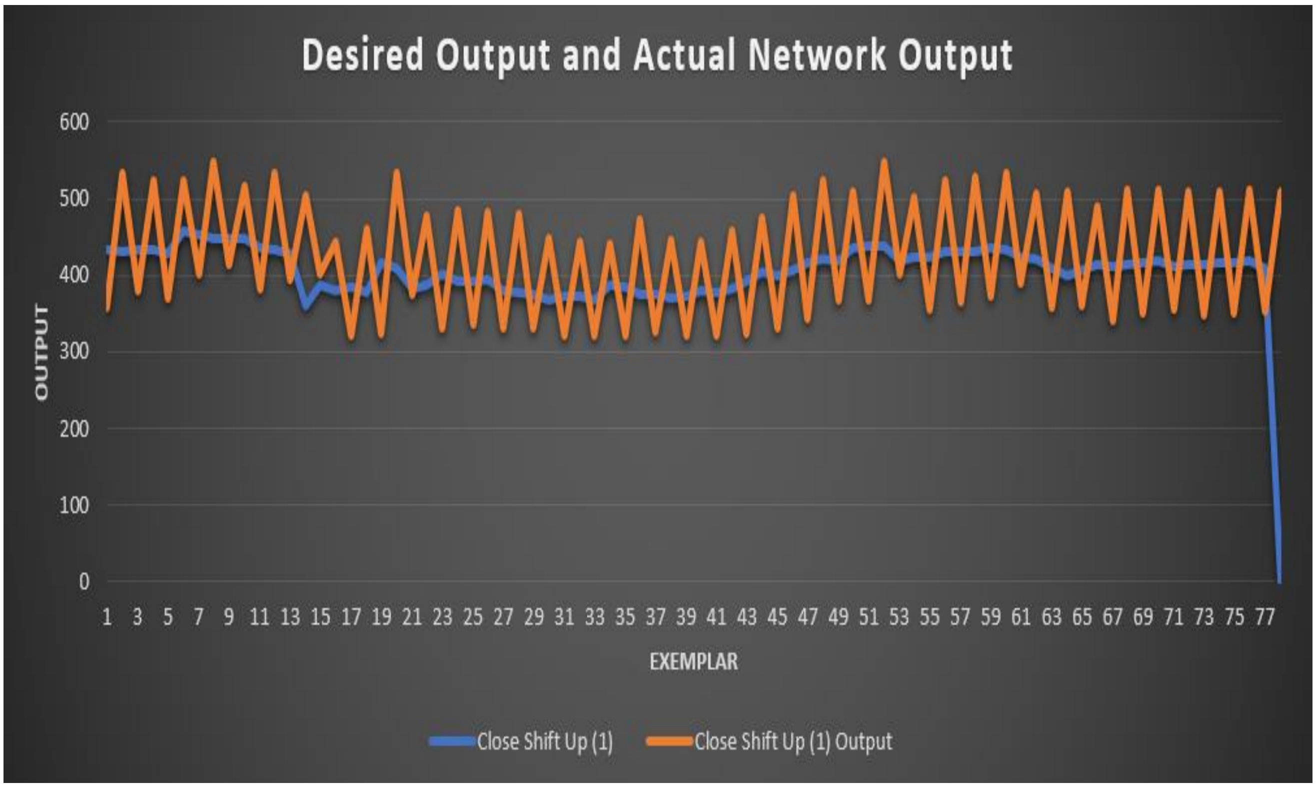

Figure 18. RNN one-day ahead prediction of closing price (March 19, 2016)

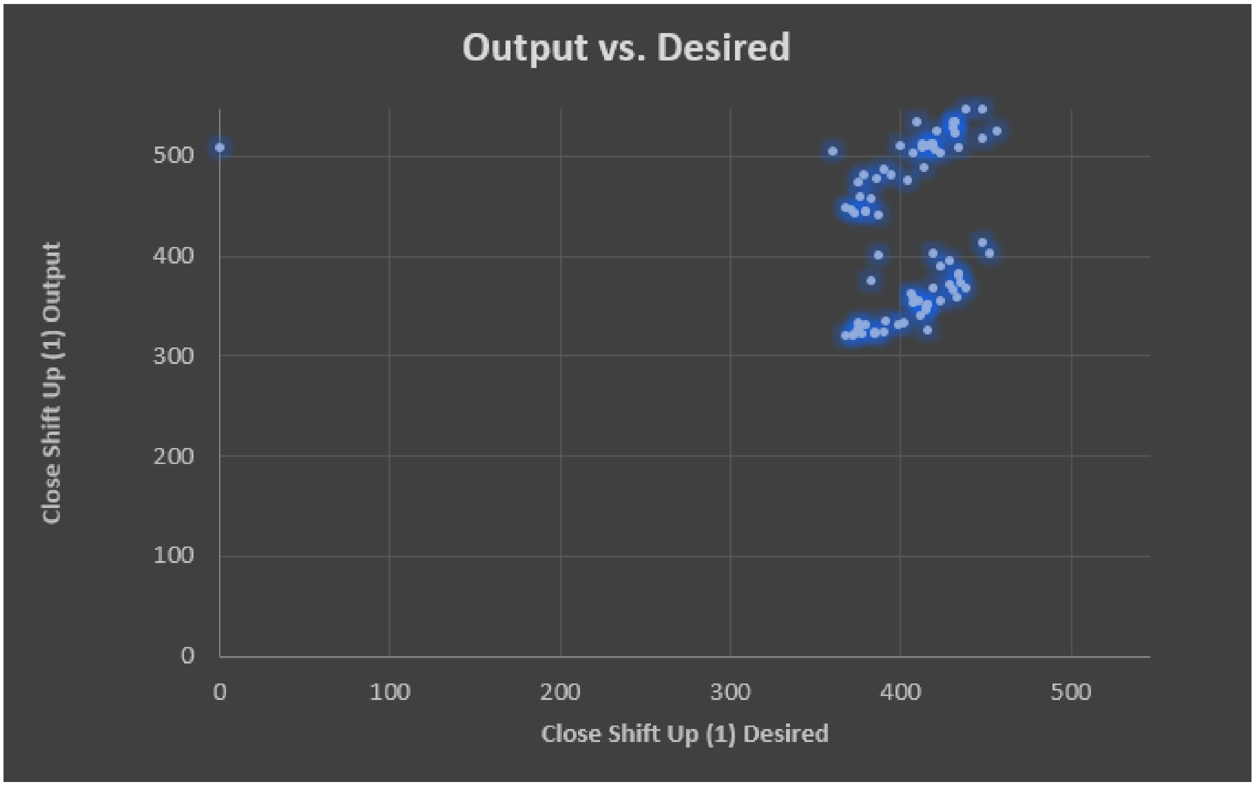

Figure 19. Scatter plot representing output vs. desired closing value using RNN

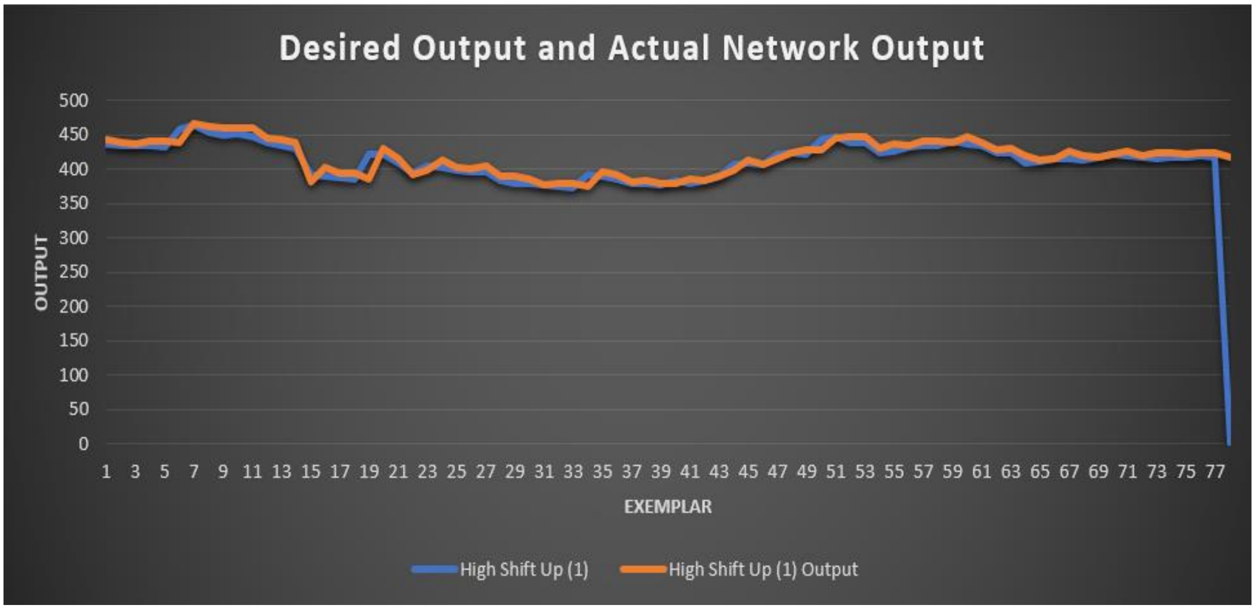

Figure 20. TDNN one-day ahead prediction of highest price (March 19, 2016)

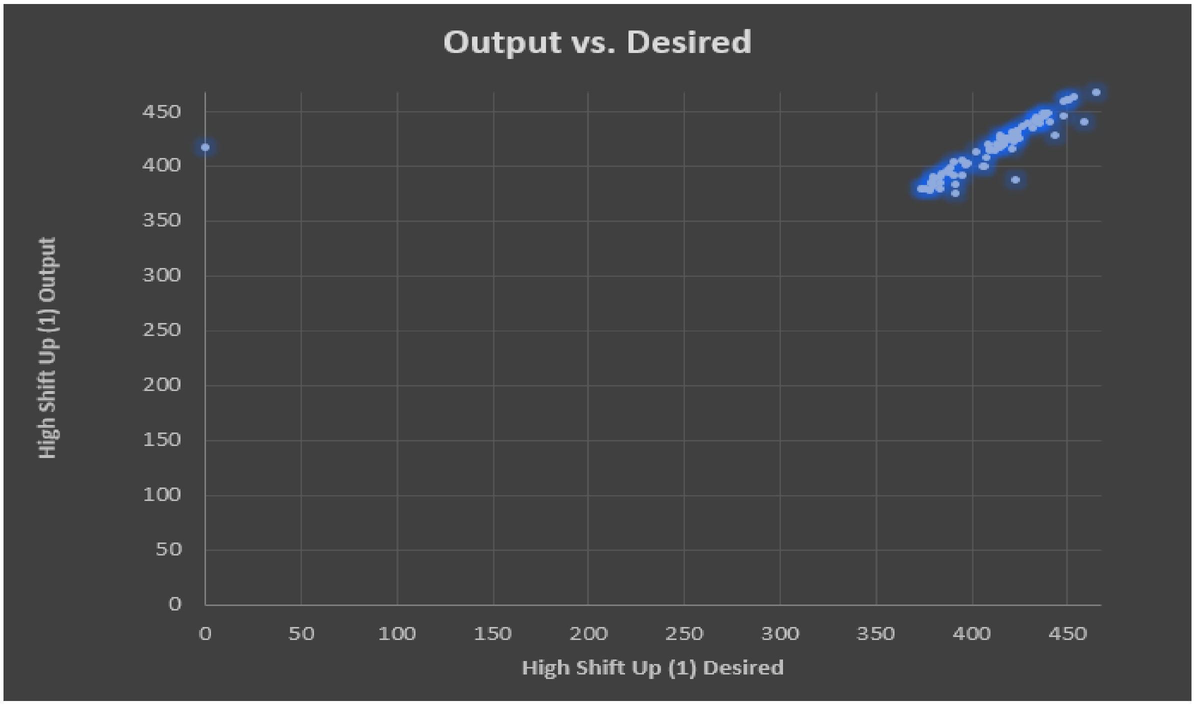

Figure 21. Scatter plot representing output vs. desired closing value using TDNN

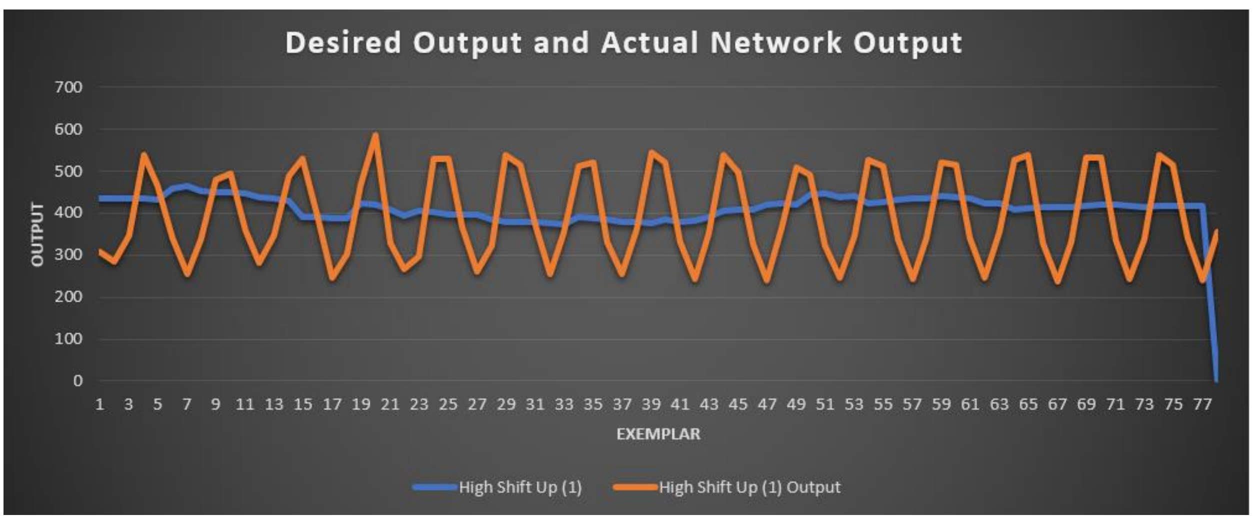

Figure 22. RNN one-day ahead prediction of highest price (March 19, 2016)

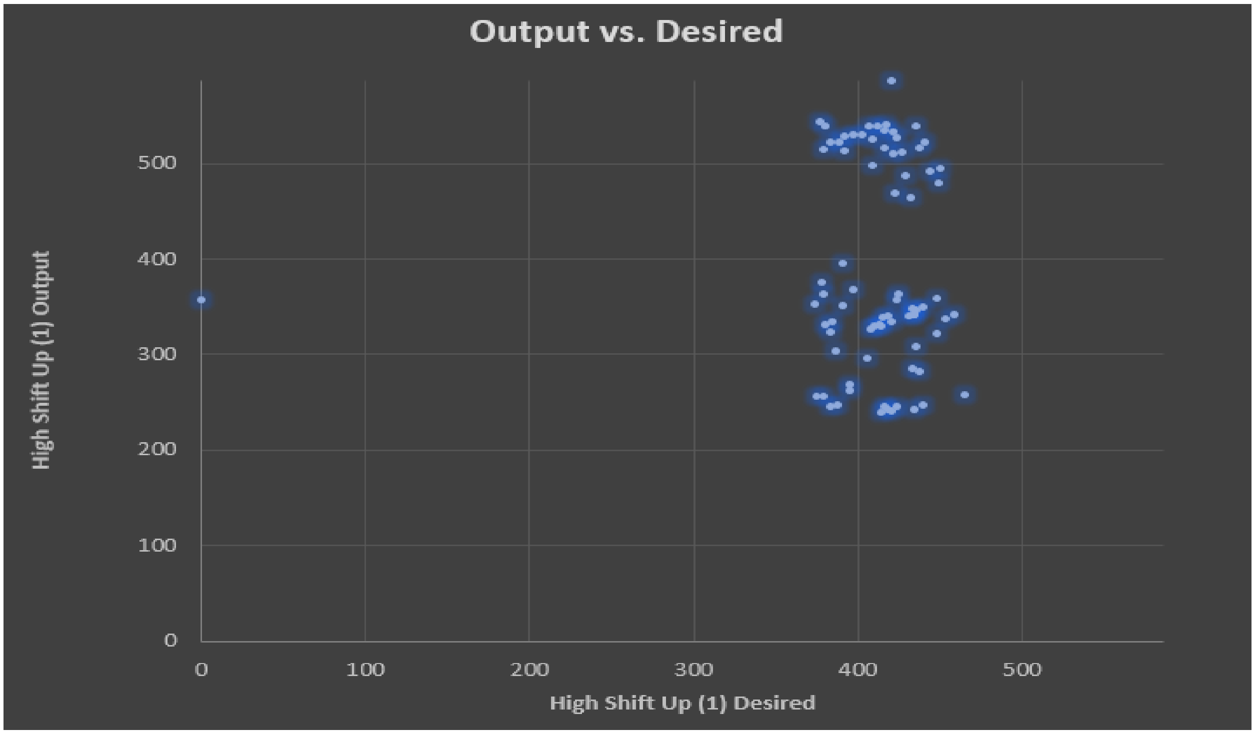

Figure 23. Scatter plot representing output vs. desired highest value using RNN

S.NO	Neural Network MODEL(high)	Models trained on 2014-15 data and tested on first quarter (Q1) of 2016		
		Final MSE	NMSE	r (Pearson's correlation coefficient)
1.	TDNN	0.065×10^{-2}	0.048	0.425
2.	RNN	29.937×10^{-2}	0.121	0.019

Table 6. Y14-15 Comparison of Performance metrics for models predicting highest value

S.NO	Neural Network MODEL	Models trained on 2014-15 data to predict 2016 Q1			
		High (predicted)	High (actual)	Close (predicted)	Close (actual)
1.	TDNN	416.83	409.95	399.54	409.04
2.	RNN	356.85	409.95	507.67	409.04

Table 7. Y14-15 One-day ahead predicted values of highest and closing price of Bitcoin

5.2.3 Models trained on 2012-13 data and tested on first quarter of 2014

The results of TDNN and RNN model trained on data set from first quarter of 2012 to last quarter of 2013(starting from January 2012 to December 2013) and tested on first quarter of 2014 (January 2014 – March 2014). One day ahead prediction of highest and closing price is done on March 19, 2014. Both models use 1000 epochs during training and testing.

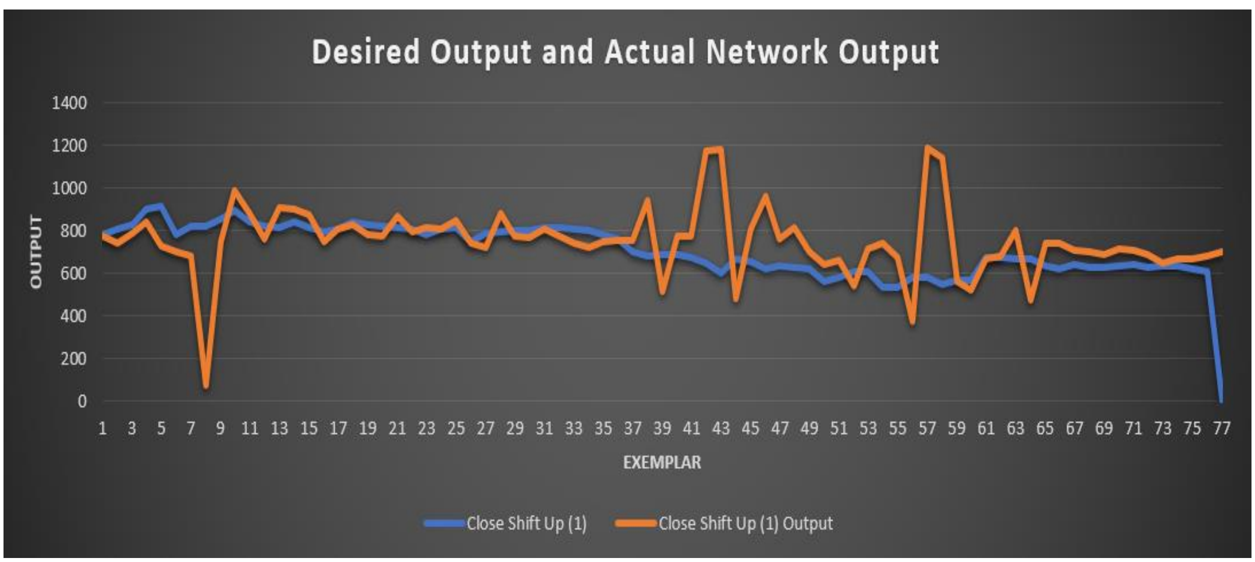

Figure 24. TDNN one-day ahead prediction of closing price (March 19, 2014)

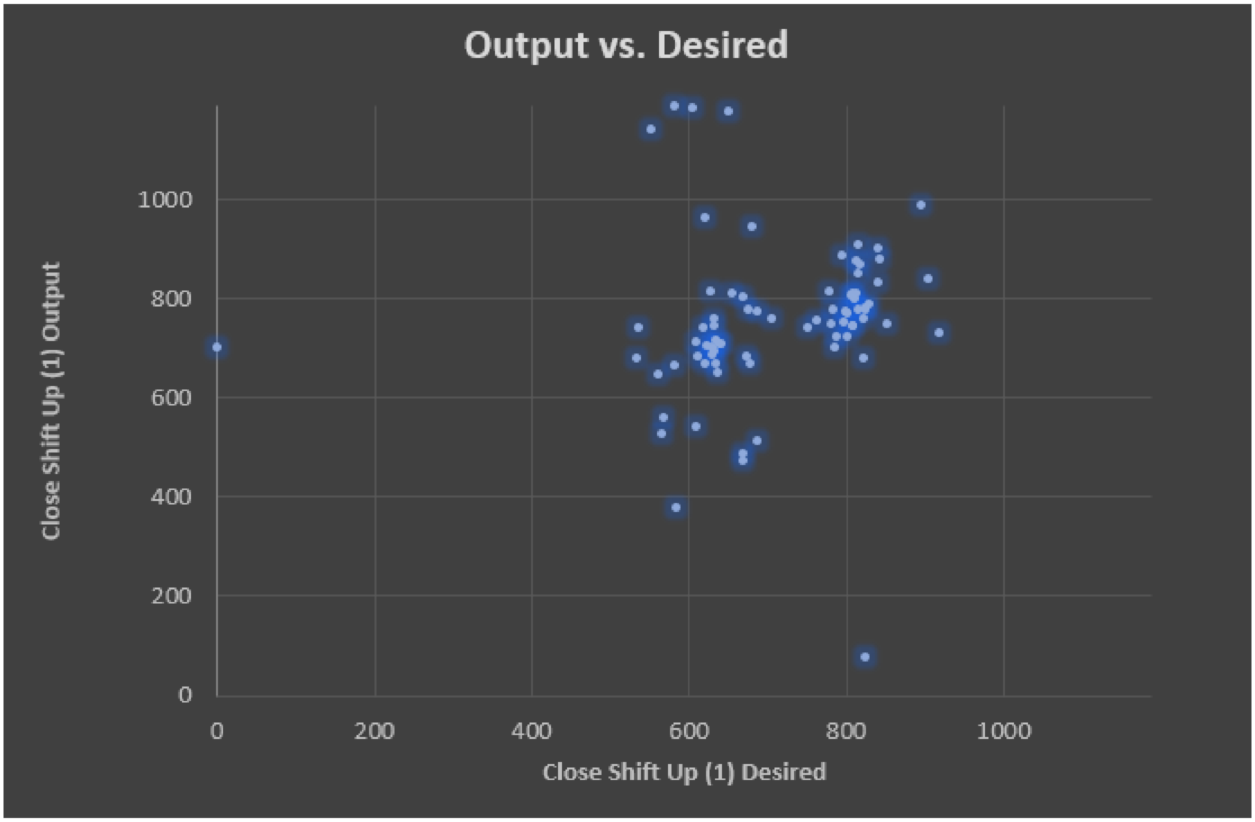

Figure 25. Scatter plot representing output vs. desired highest value using TDNN

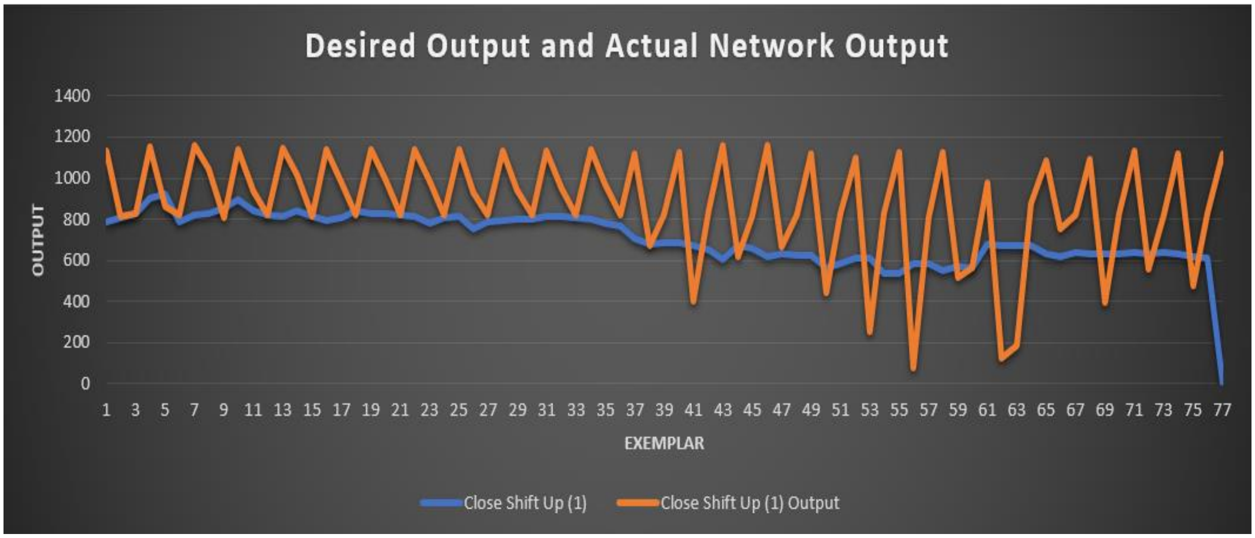

Figure 26. RNN one-day ahead prediction of closing price (March 19, 2014)

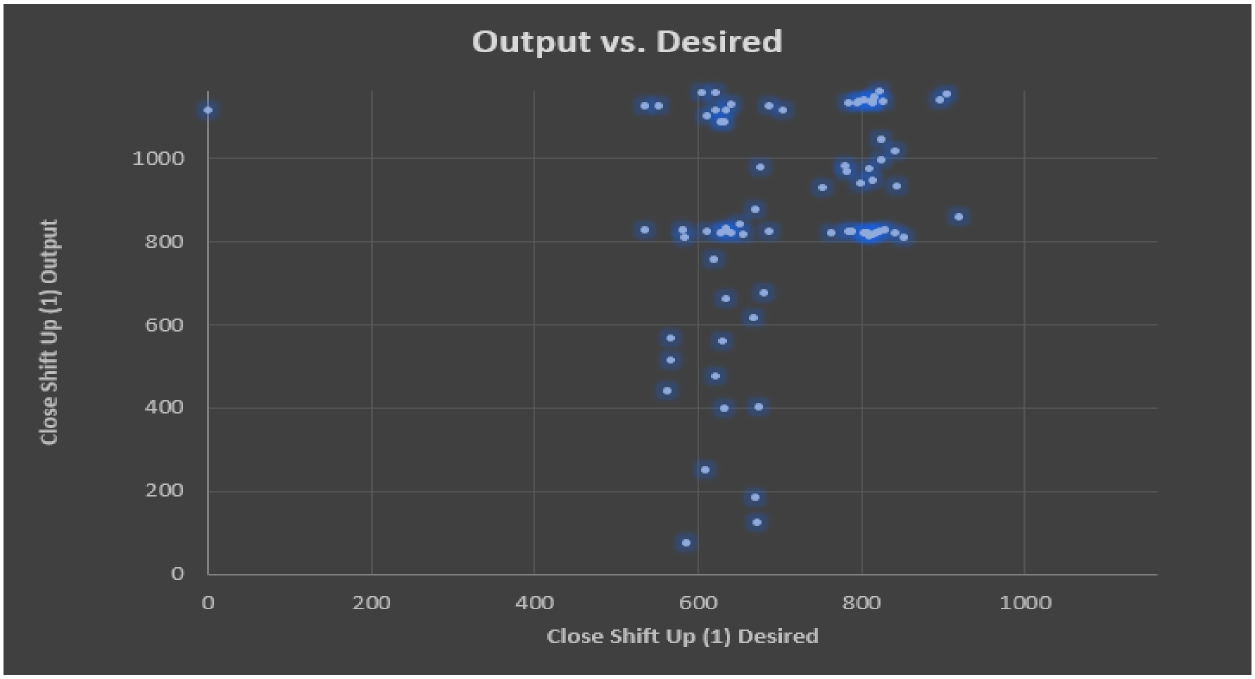

Figure 27. Scatter plot representing output vs. desired highest value using RNN

S.NO	Neural Network MODEL(close)	Model trained on 2012-13 data and tested on first quarter (Q1) of 2014		
		Final MSE	**NMSE**	**r (Pearson's correlation coefficient)**
1.	TDNN	$0.0079\text{x}10^{-2}$	0.179	0.115
2.	RNN	$1.565 \text{ x } 10^{-2}$	0.276	0.207

Table 8. Y12-13 Comparison of Performance metrics for models predicting closing value

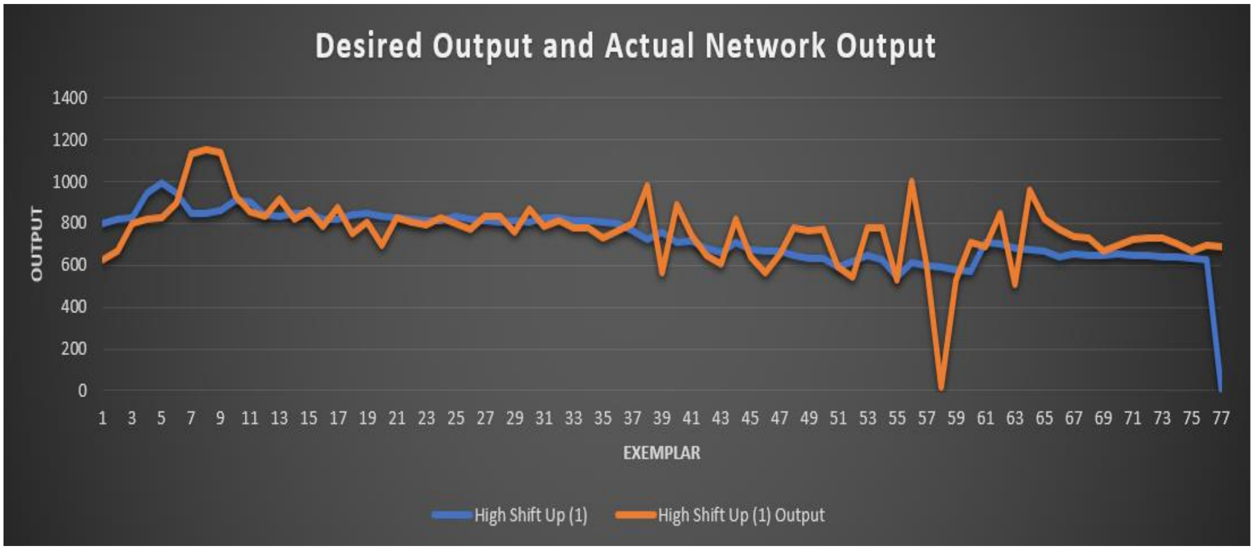

Figure 28. TDNN one-day ahead prediction of highest price (March 19, 2014)

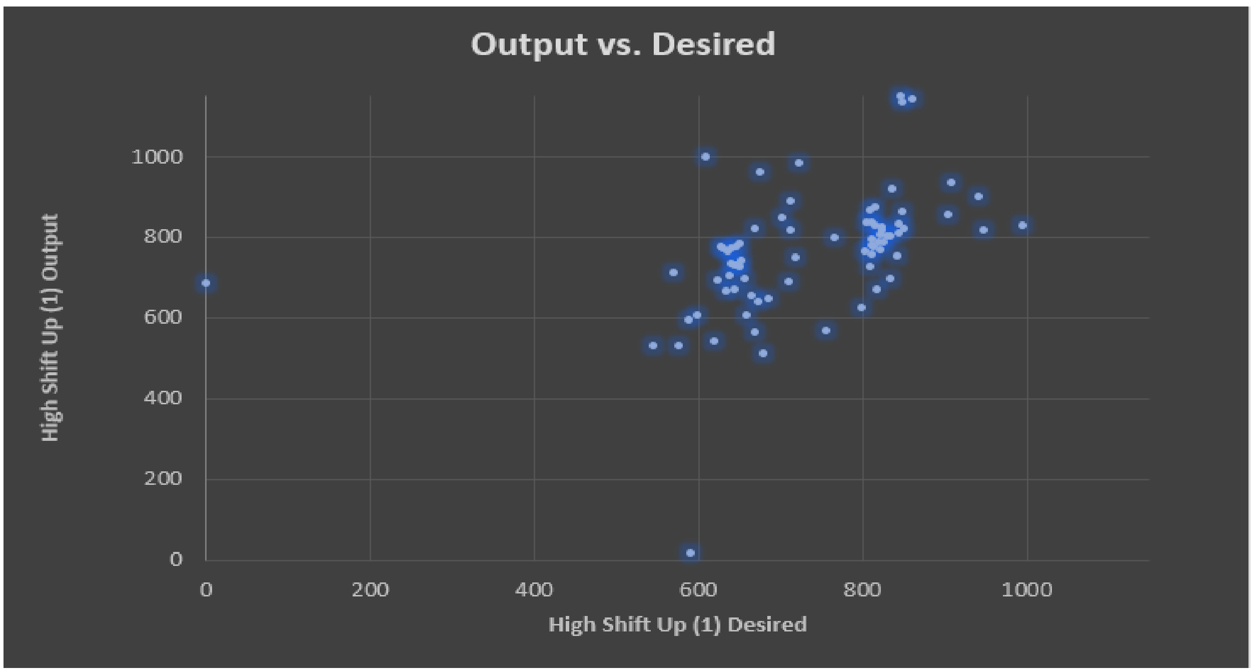

Figure 29. Scatter plot representing output vs. desired highest value using TDNN

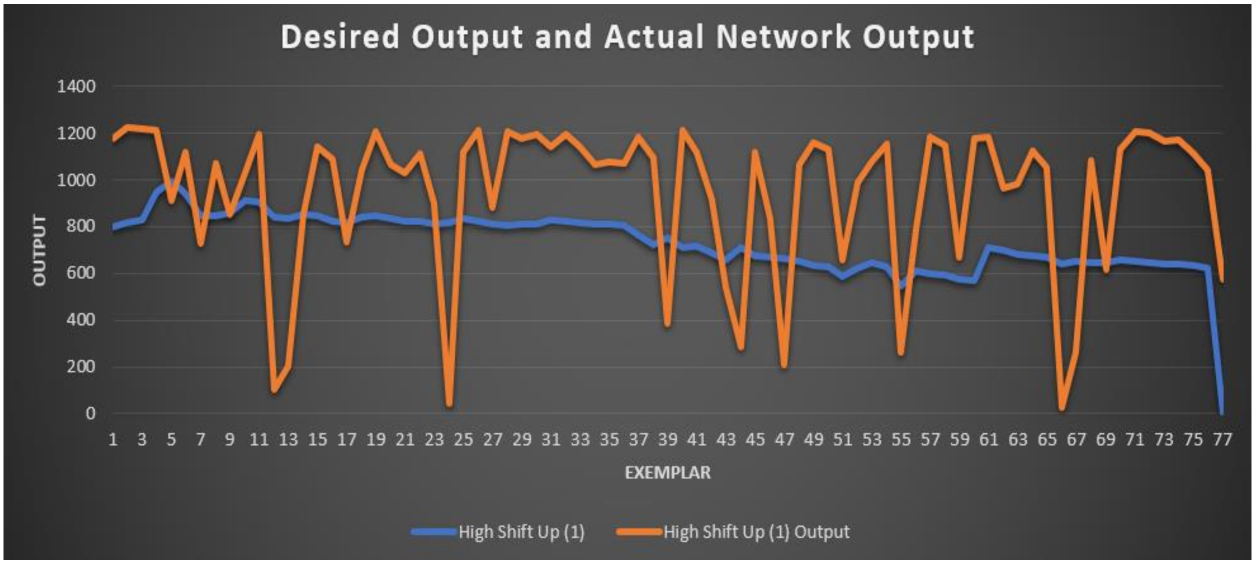

Figure 30. RNN one-day ahead prediction of highest price (March 19, 2014)

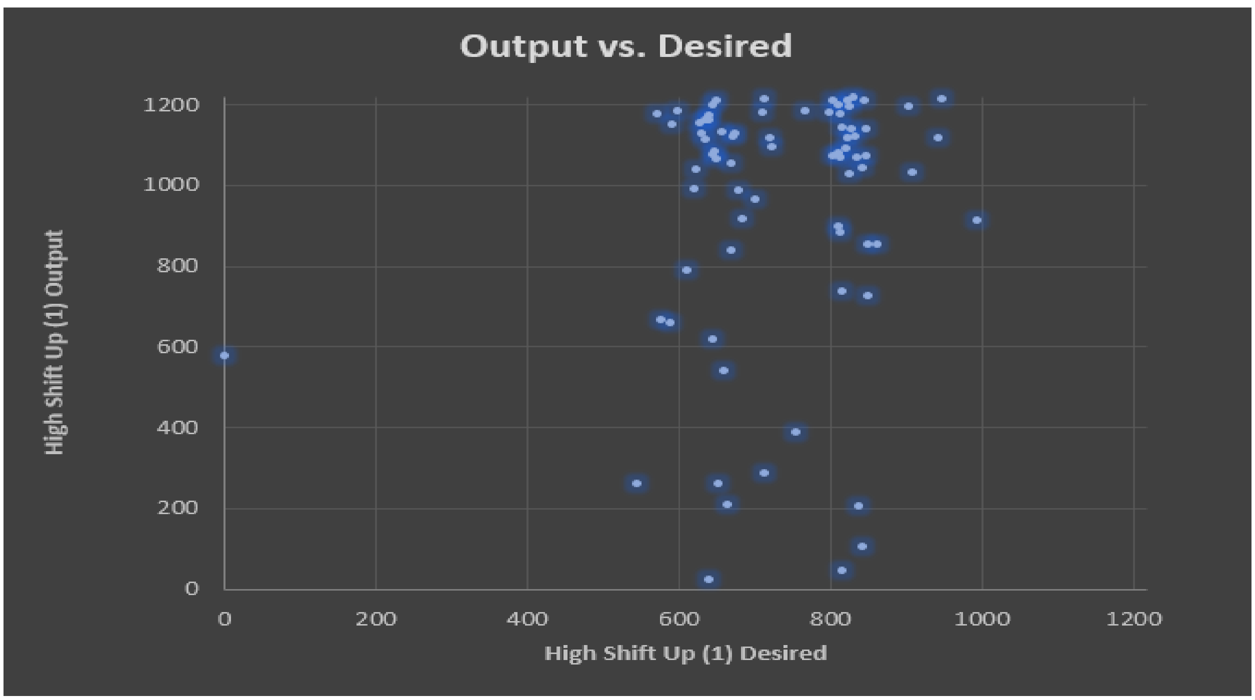

Figure 31. Scatter plot representing output vs. desired highest value using RNN

S.NO	Neural Network MODEL(high)	Model trained on 2012-13 data and tested on first quarter (Q1) of 2014		
		Final MSE	**NMSE**	**r (Pearson's correlation coefficient)**
1.	TDNN	0.024×10^{-2}	0.135	0.438
2.	RNN	5.306×10^{-2}	0.335	0.181

Table 9. Y12-13 Comparison of Performance metrics for models predicting highest value

S.NO	Neural Network MODEL	Model trained on 2012-13 data and tested on first quarter (Q1) of 2014			
		High (predicted)	**High (actual)**	**Close (predicted)**	**Close (actual)**
1.	TDNN	685.68	623.95	702.78	611.2
2.	RNN	577.09	623.95	1115.33	611.2

Table 10. Y12-13 One-day ahead predicted values of highest and closing price of Bitcoin

Chapter 6 - Summary and Future Work

6.1 Summary

This work presents an application of artificial neural networks for making one day ahead prediction of highest and closing price of cryptocurrency Bitcoin. Two temporal neural network architectures have been considered: a time-delay neural network (TDNN) and a recurrent neural network (RNN). Also, comparisons between TDNN and RNN have been presented.

It is observed that TDNN model is trained in lesser time compared to RNN for the Bitcoin data set. Results indicate that TDNN models predicts the values closer to the actual price compared to RNN. Also, the Pearson's correlation coefficient (r) is higher for TDNN model than RNN in almost all the cases. This shows that proposed TDNN model makes accurate prediction of Bitcoin price.

It is possible to increase the number of hidden layers used in these models, however it is set to one layer in the current study to avoid complexity of the networks. After training, both the network models provided satisfactory results.

6.2 Future work

Current work focuses on one-day ahead and partly on three-day ahead prediction (2016-17 data set) of Bitcoin price. However, we can follow a similar procedure to perform multi-step ahead prediction of Bitcoin price thereby increasing the scope of experiments. In addition to cryptocurrency Bitcoin, there are other cryptocurrencies such as Ethereum, Litecoin, Ripple, etc. that gained attention in the trading markets. Analysis on these cryptocurrencies can help investors decide which one to buy or sell so as to end up with profitable trades. Also, I have plans on using the GPU based accelerator that can significantly reduce the training time of the models. Deep learning techniques such as Long Short-Term Memory (LSTM) networks (Hochreiter &

Schmidhuber, 1997) can also boost the predictions, thereby helping to make better trading decisions. The scope of this project can be extended to use some deep learning techniques.

Bibliography

Amjad, M. J. & Shah, D. (2017). "Trading Bitcoin and Online Time Series Prediction". *Proceedings of the Time Series Workshop at NIPS 2016,* (pp. PMLR 55:1-15).

Billah, M. & Waheed, S. (2015). "Predicting Closing Stock Price using Artificial Neural Network and Adaptive Neuro Fuzzy Inference System (ANFIS): The Case of the Dhaka Stock Exchange",. *International Journal of Computer Applications (0975 - 8887),* Volume 129 - No.11.

Billard, A., de Chambrier, Guillaume., Figueroa, N. & Lamotte, D. (2016). *Advanced Machine Learning, Practical 4: Regression (SVR, RVR, GPR).*

Box, J. & Reinsel. (1994). *Time Series Analysis, Forecasting and Control* (3 ed.). Englewood Clifs, NJ: Prentice Hall.

Brockwell, P. J. & Davis, R. A. (2013). *Time series: theory and methods.* New York, NY: Springer.

Deshpande, R. R. & Khan, A. R. (2012). "On the spot time series prediction using Jordan Ellman artificial neural network". *International Journal of Advances in Engineering & Technology*, 611- 621.

Greaves, A. & Au, B. (2015). *"Using the Bitcoin Transaction Graph to Predict the Price of Bitcoin".* Stanford University. Department of Computer Science.

Hochreiter, S. & Schmidhuber, J. (1997). *Long Short-Term Memory.* Institute of Bioinformatics, Johannes Kepler University Linz.

Hsu, W. H. (1998). *Time Series Learning with Probabilistic Network Composites. Ph.D. dissertation.* University of Illinois at Urbana-Champaign (Technical Report UIUC-DCS-R2063). August, 1998.

Lo, A. W. & MacKinlay, A. C. (1988). Stock market prices do not follow random walks: Evidence from a simple specification test. *The Review of Financial Studies, Volume 1,* 41–66.

Marques, F. D., de Souza, L. F., Rebolho, D. C.,Caporali, A. S., & Belo, E. M. . (2005). "Application of Time-Delay Neural and Recurrent Neural Networks for the Identification of a Hingeless Helicopter Blade Flapping and Torsion Motions",. *J. of the Braz. Soc. of Mech. Sci. & Eng.*, 97-103.

Molina, D., Liang, J., Harley, R., & Venayagamoorthy, G. K. (2011). "Comparison of TDNN and RNN Performances for Neuro-Identification on Small to Medium-Sized Power Systems". *IEEE Symp. on Comp. Intelligence in Smart Grids, Proc. of the 2011.*

Scheier, C. & Tschacher, W. (1996). *Appropriate algorithms for non-linear time series analysis in psychology*.

Shumway, R. H. & Stoffer, D. S. (2017). *Time Series Analysis and It's Applications* (3 ed.). Springer International Publishing AG 2017.

Thomas, A. (2013). *API for Bitcoin Data*. Retrieved March 19, 2018, from Quandl Blog: https://blog.quandl.com/api-for-bitcoin-data

Ugiliweneza, B. (2007). User of arima time series and regressors to forecast the sale of electricity. *SESUG Proceedings.*