# SPAL: Stock Prices Forecast with Stacked Autoencoders and LSTM

Shantanu Mandal [*], Jee Su Byun [†] and Zebo Xiong [‡]

Texas A&M University

## Abstract

The application of neural network (NN) methods in finance has received a great deal of attention from both industry and academia [2]. This study presents an integrated analysis which combines stacked autoencoders, long short term memory (LSTM) and feature analysis with embedded NN. LSTM networks are originally designed for sequential learning but are less commonly used in financial time series predictions; yet, they are inherently suitable for this domain [5]. We deployed LSTM networks for predicting out-of-sample directional movements for the constituent stocks of the S&P500 from 2009 until 2018. According to the result, outperformance relative to the general market is very clear from 1992 to 2009, but as of 2010, excess returns seem to have been arbitraged away with LSTM profitability fluctuating around zero after transaction costs. We further unveil sources of profitability, thereby shedding light into the black box of artificial neural networks. The neural network has three stages. *First*, the stock price time series is decomposed by WT to eliminate noise. *Second*, SAEs is applied to generate deep high-level features for predicting the stock price. *Third*, high-level denoised features are then fed into LSTM to forecast the next day's closing price. S&P500 index and their corresponding index futures are chosen to examine the performance of the proposed model.

## 1 Introduction

Stock market prediction is usually considered as one of the most challenging issues among time series predictions [11] because of its volatile properties. How to correctly predict stock movement is still an heated topic with respect to the economic and social organization. During the past decades, machine learning techniques, such as Artificial Neural Networks [3] and the Support Vector Regression (SVR) [6], have been widely used. In the paper, however, we try to do experiments on an improved model which models complex real-world data by extracting robust features that capture the relevant information [2]. Considering the complexity of financial time series, neural network is regarded as one of the most charming topic [2]. However, this field still has much to explore.

Currently we have three main neural network approaches. First, convolutional neural networks. Second, deep belief networks. Third, stacked autoencoders. Previously, the relevant methods used for finance were the former two of these. For example, Ding et al. [4] combine the neural tensor network and the deep convolutional neural network to predict the short-term and long-term influences of events on stock price movements. Also, certain works use deep belief networks.

[*] shanto@tamu.edu

[†] jb733@tamu.edu

[‡] dior@tamu.edu

Preprint. Under review.

However, there has been only one attempt, by Bao et al. [2], that incorporates stacked autoencoders. This paper will continue to explore Bao's paper to refine the stock prediction. Bao's model has three parts: wavelet, stacked autoencoders(SAEs) and LSTM. SAEs play a main role in that model. It is used to learn the deep features of a stock price in an unsupervised way, as SAEs model can successfully learn invariant and abstract features [3].

After SAE is used to find the invariants, LSTM is then used to enhance the prediction accuracy. It is one of the most advanced deep learning architectures for sequential learning tasks, such as handwriting recognition, speech recognition, or time series prediction [5]. Surprisingly, its application in financial area has yet been little and there has not been any previous attempt to deploy LSTM networks on a large, liquid, and survivor-bias-free stock universe to assess its performance in large-scale financial market prediction tasks [5]. Unlike conventional RNN, LSTM is well-suited to learn from experience to predict time series when there are time steps with arbitrary size. In addition, it can solve the problem of a vanishing gradient by having the memory unit retain the time related information for an arbitrary amount of time [7]. Previous work has shown that LSTM is more effective than RNN. At the same time, WT is used to fix the noise feature of financial time series. We use it to denoise the input financial time series. We'll call this model SPAL hereafter [2].

We select S&P500 Index for our experiments from the past 10 years, up until 2018 and use SPAL to forecast the movements of each stock index and check how well our model is performing in predicting stock moving trends. In the future, we hope to test the performance of the model in multiple financial markets instead of only one market for better results. This is because, according to the efficient market hypothesis (EMH), the efficiency of a market affects the predictability of its assets. In other words, even though we may achieve satisfactory predictive performances in one market, it is still difficult to attribute it to the role of the proposed model [2] in other financial markets. Since we will eventually test the model in multiple market conditions, we hope it will help us to solve this problem and proves the robustness of the model. We evaluate the model's performance by measuring how predicted prices are numerically close to the actual prices. There are four possible ways to predict the accuracy: Mean absolute percentage error (MAPE), correlation coefficient (R), Theil's inequality coefficient (Theil U), Root Mean Square Error (RMSE) [2]. They are widely used to measure the predicted values [8].

## 2 Problem Statement

In this section we will describe

- Basic concepts of the stock bar and pattern and describe why it is hard to get the features.
- Why we use NN in detecting stock pattern.

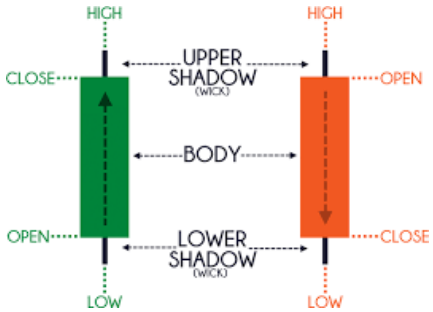

Figure 1: Stock bar for any timeframe [1]

### 2.1 Stock series data

A 'tick' is defined as a single movement in price at any given time frame. As it is very difficult and expensive to collect the tick data history, we used data with larger timeframes in our experiments. A candle bar is commonly used to depict price movement behaviors in a unit timeframe. A bar has 4 data points. Open, high, low, close, as shown in Figure 1.

### 2.2 Chart pattern

When seen in multiple timeframes, stock prices follow different chart patterns that indicate certain future price movements with high likelihoods. Before the advent of machine learning in finance, statisticians made predictions by recognizing these patterns. Some of the well known examples of chart patterns are given in figure 2. Since they are not always easily detected with bared eyes, we are motivated to use NN to detect the patterns. As NN is known to capture fuzzy features from large datasets, we can conclude that NN can also predict results with empirical evidence from the past price movements such as these patterns.

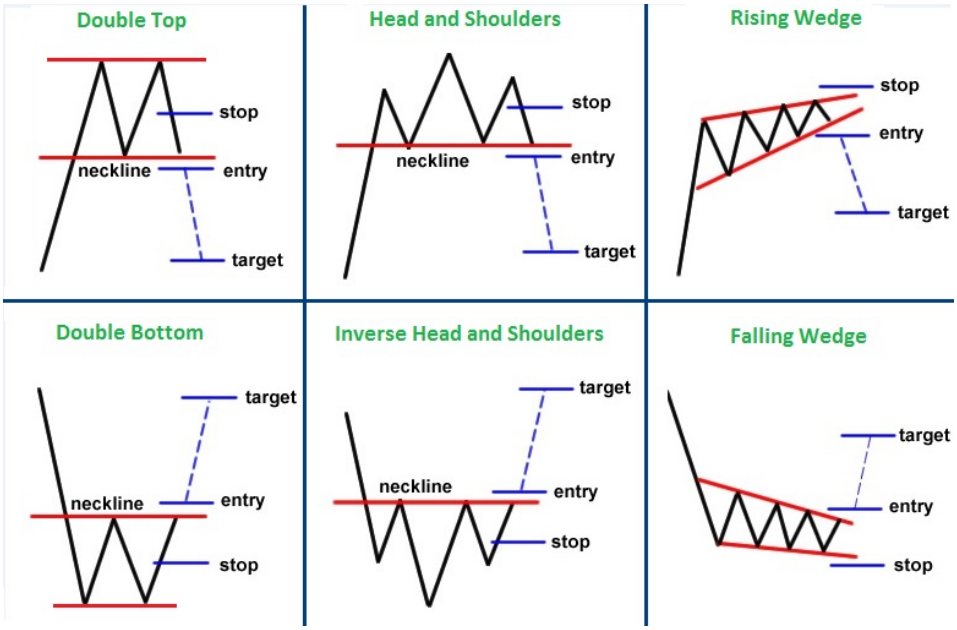


Figure 2: Different stock pattern across timestep [1]

## 3 Overall Architecture

In the following sections we describe two approaches with different model architecture and features. *First*, we describe the SPAL model that tries to analyse prices with discrete wavelet transform with LSTM.
*Second*, in addition to this, we describe stacked NN trained with multiple features with embedding layer for performance comparison.

### 3.1 Model

#### 3.1.1 Overview

As mention, our dataset uses 10 years' daily price data of S&P500 index. Each row of daily stock price information includes OHLC (Open, High, Low, Close), technical indicators (MACD, CCI, ATR, BOLL, EMA20, MA10, MTM6, MA5, MTM12, ROC, SMI, WVAD), and macroeconomic variables (US Dollar Index, Federal Fund Rate). Instead of normalization, we applied manual scaling of these input features so that they can have similar orders of magnitude. For each 'sub-dataset training iteration', we divided the dataset into sub-datasets with each having 600 days of data, which is a bit more than 2 years of data. Our step size was set to 2 months as we thought it was a reasonable timeframe where each price data during that time can affect the opening price of that stock after 2 months. Specifically, for each sub-data, we used the most recent 60 days of data as the test data, the second most recent 60 days as the validation, and the rest as the training data.

As we initially planned, we implemented a model (Figure 3) that takes denoised inputs that are preprocessed via wavelet transform, and trains based on a 4-layer stacked autoencoder and subsequent Long-short term memory neural network.

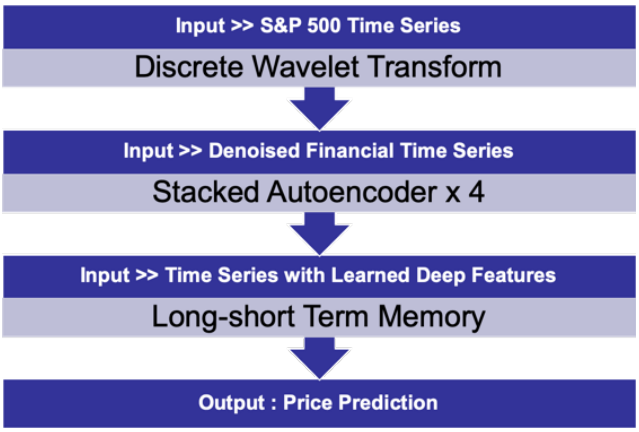


Figure 3: Overview of SPAL

#### 3.1.2 Denoising using DWT (Wavelet Transform)

First of all, we apply wavelet transform for data denoising. Wavelet transform has the ability to decompose complex information and patterns into elementary forms. It is applied for data denoising in this project because of its ability to handle non-stationary financial time series data, which is useful in handling highly irregular financial time series. We apply the Haar function as the wavelet basis function because it can not only decompose the financial time series into time and frequency domain but also reduce the processing time significantly. The wavelet transform with the Haar function as a basis has a time complexity of O(n) with n being the size of the time series. We first calculated wavelet coefficients for each data feature column, and designated a threshold to create the signals using thresheld coefficients.

#### 3.1.3 Stacked Autoencoder

After denoising, autoencoders are applied for layer-wise training for the OHLC variables and technical indicators. Single layer AE is a three-layer neural network, the first layer being the input layer and the third layer being the reconstruction layer, respectively. The second layer is the hidden layer, designed to generate the deep feature for this single layer AE. The aim of training the single layer AE is to minimize the error between the input vector and the reconstruction vector. The first step of the forward propagation of single layer AE is mapping the input vector to the hidden layer, while the second step is to reconstruct the input vector by mapping the hidden vector to the reconstruction layer. The activate function can have many alternatives such as sigmoid function, rectified linear unit (ReLU) and hyperbolic tangent. In this project, we set this to be a sigmoid function for all layers. The model learns a hidden feature from input by reconstructing it on the output layer.

Stacked autoencoders is constructed by stacking a sequence of single-layer AEs layer by layer. The single-layer autoencoder maps the input daily variables into the first hidden vector. After training the first single-layer autoencoder, the hidden layer is reserved as the input layer of the second single-layer autoencoder. Therefore, the input layer of the subsequent AE is the hidden layer of the previous AE.

For training, Adam optimizer is used for solving the optimization problem in SAEs, and completing parameter optimization. Each layer is trained using the same optimizer as a single-layer AE by and feeds the hidden vector into the subsequent AE. The weights and bias of the reconstruction layer after finishing training each single-layer AE are cast away. Depth plays an important role in SAE because it determines qualities like invariance and abstraction of the extracted feature. In this project, the depth of the SAE will be set to 4 with each layer having 10 hidden units.

#### 3.1.4 Long-short Term Memory

After SAEs, we use the time series data as sequence to train the LSTM. LSTM networks consists of an input layer, several hidden layers and an output layer. The number of input layers is the same as the number of features. We choose LSTM because it doesn't have the problem of vanishing gradients that a Recurrent Neural Network often has. The main drawback of RNN is that it cannot use previously

seen data as the size of input sequences becomes larger. LSTM is a gated version of recurrent neural network that solves this problem.

Another reason we use LSTM is because of the fact that in stock data pattern is the key factor to move the market in technical analysis. Hand-coded models such as clustering or feature extraction are sometimes difficult in time series data such as stock value. Therefore, the main purpose of the LSTM is to capture pattern throughout the time series data. For our LSTM model, the time step size is set to be 4, the batch size to be 60, and the hidden dimension to be 100. For training, we used the mean square error loss with Adam optimizer.

### 3.2 Applying NN with different stock features

SPAL uses different mathematical parameters as features. In stock price it is not enough to see only the price. There are several indicators that is used to predict different signals. SPAL uses those signal indicators as different features. However, features are more important for any machine learning model. If the features are good and it is learn-able then the model predictions are more accurate for financial data. Following sections describe the features first, then we describe how SPAL uses those in the NN model that we implemented in addition to SPAL

#### 3.2.1 Features

There are several other feature maps that we use in our experiments are described below.

- *Moving average convergence divergence (MACD):* MACD gives the convergence divergence of two period running exponential moving average (EMA). We use 12, 26 period EMA for calculating MACD. This features represents how price fluctuates over the period of time.
- *Relative strength index (RSI):* RSI is a momentum indicator that measures the magnitude of recent price changes to evaluate overbought or oversold conditions in the price of a stock or other asset. We choose this features to find out the market momentum at some particular point.
- *Bollinger Band (BB):* A Bollinger Band is a technical analysis tool defined by a set of lines plotted two standard deviations (positively and negatively) away from a simple moving average (SMA) of a stock price. It gives the range for a timeframe to fluctuate. This feature helps to get the highest and lowest point to fluctuate that helps to predict the director of the price.
- *Aroon Indicator:* Aroon is a statistical indicator that is used to identify the trend change in the stock.
- *Pivot points (PP):* PP gives several support and resistance level in price direction. Using this as a features helps to classify the direction of price trend.

#### 3.2.2 Model

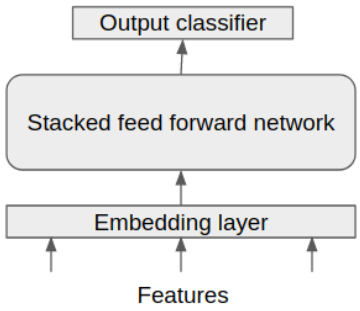


Figure 4: Stacked network with embedding layer for categorical bar prediction

In this experiment we use different features as input and output is different bar predictions. All the bars are divided into several categories according to their body, head and tail (Figure 4). At first we make a n-dimension embedding layer with all of our features together. Empirically we see that

self learned embedding works better with this model. Embedding layer is passed through n-stacked feed forward network. The model outputs the prediction with softmax layer with cross-entropy loss. Dropout is used to make the training faster.

#### 3.2.3 A Glance on Implementation

Data preparation and handling are entirely conducted in Python 3.7 (Python Software Foundation, 2018), relying on the packages NumPy (Van Der Walt, Colbert, Varoquaux, 2011), Pandas (McKinney, 2010). The neural network with LSTM are developed with keras (Chollet, 2016) on top of Google TensorFlow, a powerful library for large-scale machine learning on heterogenous systems (Abadi et al., 2015). The AutoEncoder and Wavelet Tranformation are processed by PyWavelets and PyTorch library respectively.

## 4 Results

### 4.1 SPAL Results

As shown in Figure 5, both train and validation losses are improved at some point after 7th sub-data iteration. However, as training reaches the end of our dataset, at around 20th sub-data iterations, we observe spikes of loss increases which may be due to overfitting of the data. To achieve a high accuracy with time series with greater sizes, we will need further hyperparameter tuning to complement this by, for example, adjusting threshold to create signals in our wavelet transformer.

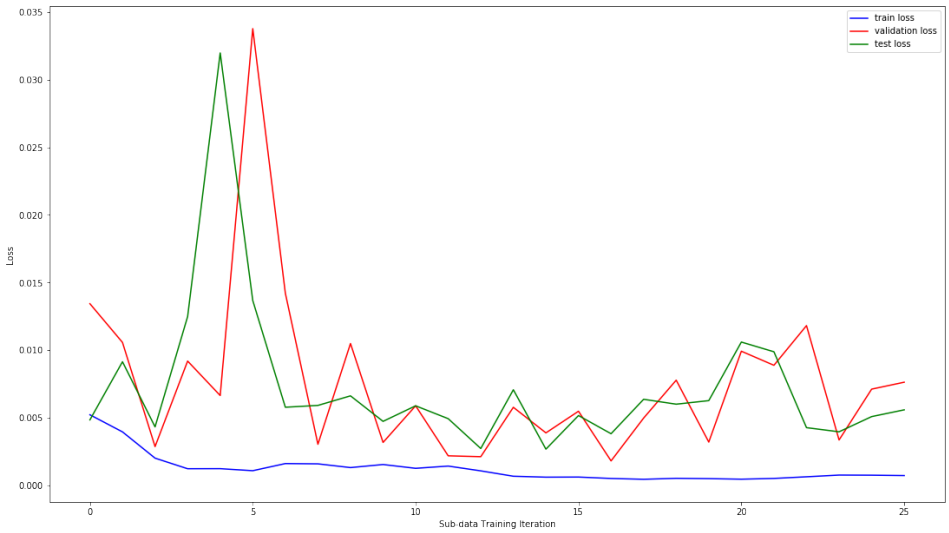


Figure 5: Loss per Sub-dataset Training Iteration

The actual and predicted price data for S&P500 are shown in Fig 5 and Fig 6. Our model predicts S&P500 index with reasonable accuracies as shown in Figure 8. At the start, the differences of predicted and actual prices are negligible. However, we observe exponential increase of inaccuracy of over 10 percent of the prices when the stock price plummeted at 2,500 to 3,000th timeframes. The model's accuracy dropping when the market sentiment is low is further proven by the most recent drop of index prices at around 4,700th timeframe.

Regarding our model and training schemes, we discovered normalization worsens accuracy compared to when features are scaled manually. We think that this is accountable by that train and test set are normalized at the same time. In other words, there are potential leaks of future information in each time step. Regarding the stacked autoencoder, the intuition was to derive hidden deep features out of the original raw dataset. However, what we think it actually does is to force our data into smaller dimensions with potential loss of important information. We will conduct accuracy comparison with the model without SAE in the future.

Besides, we tested price history using point-by-point approach and sequence-by-sequence approach in our second architecture. The training and test rate is 0.8. We found that by using 8 epoch, we can

Figure 6: Actual Data

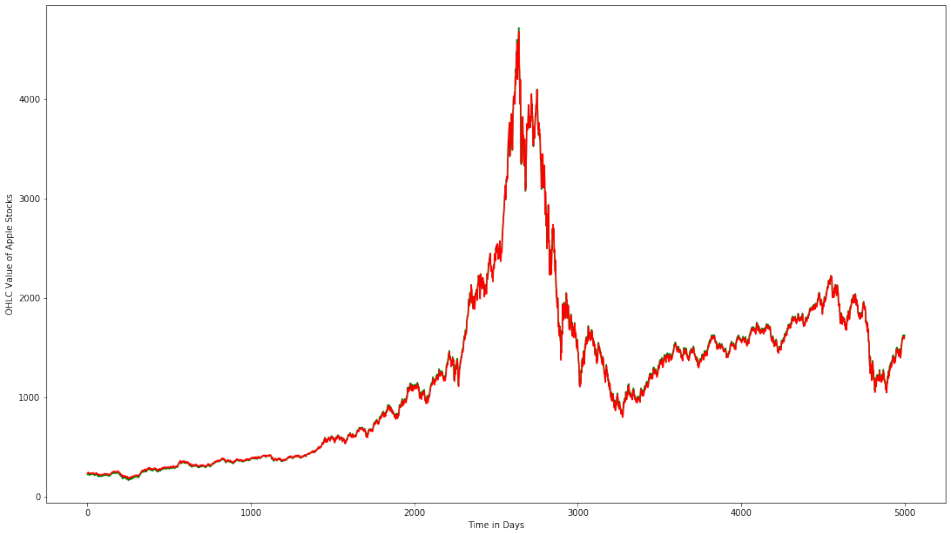

Figure 7: Predicted Data

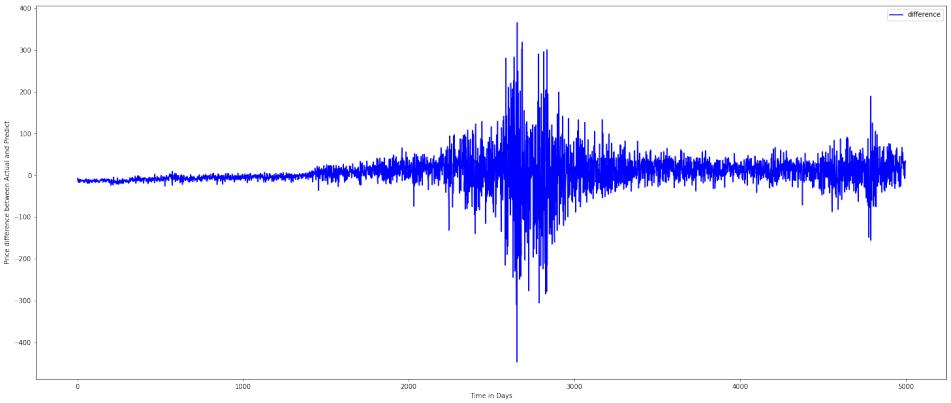

Figure 8: Difference between Predicted and Actual Prices

achieve the accuracy at 60%. Though we can find some trends by using the sequence by sequence, the prediction is not stable (Figure 1112). The point by point prediction figures looks very good but here it is a bit deceptive. We can only predict one data point ahead, based on its all previous history.

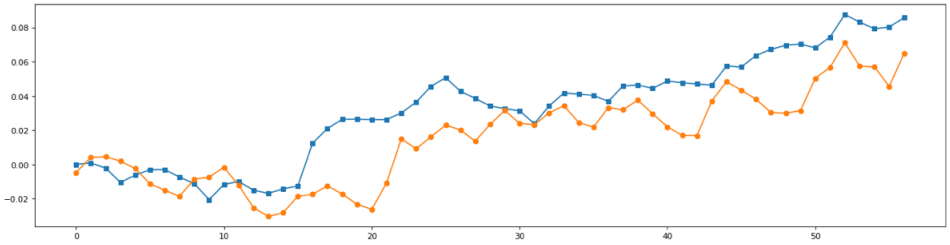

Figure 9: Point-by-Point prediction (2 Epoch, 50 Batch size)

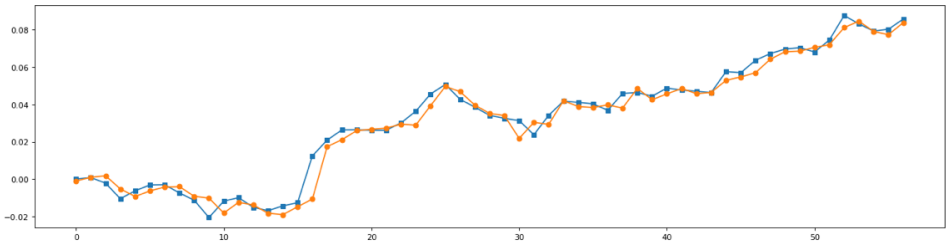

Figure 10: Point-by-Point prediction (8 Epoch, 50 Batch size)

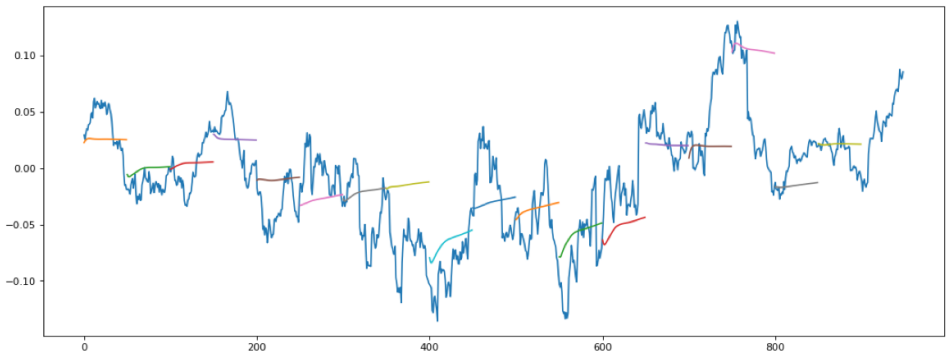

Figure 11: Sequence-by-Sequence prediction (2 Epoch, 50 Batch size)

### 4.2 Effectiveness of using multiple features in NN model

In table 1 we describe the precision, recall and f1-score for using the different feature maps to predict different bar category. We divided all the bars into 10 different categories according to their shape. From table 1, bar category 6 has the highest f1-score and category 3 and 7 has the lowest f1-score among all. Overall accuracy of our model is 67%.

## 5 Related Work

Initial evidence has been established that machine learning techniques are capable of identifying (non-linear) structures in financial market data, see Huck (2009, 2010), Takeuchi and Lee (2013) , Moritz and Zimmermann (2014) , Dixon, Klabjan, and Bang (2015) , and further references in Atsalakis and Valavanis (2009) as well as Sermpinis, Theofilatos, Karathana- sopoulos, Georgopoulos, and Dunis (2013). The relevant work on deep learning applied to finance has introduced the convolutional neural networks, deep belief networks. For example, Ding et al. [4] combine the neural tensor network and the deep convolutional neural network to predict the short-term and long-term influences of events on stock price movements. Other than that, certain works use deep belief networks in financial market prediction, for example, Yoshihara et al. [12], Shen et al. [10] and Kuremoto et al. [9].

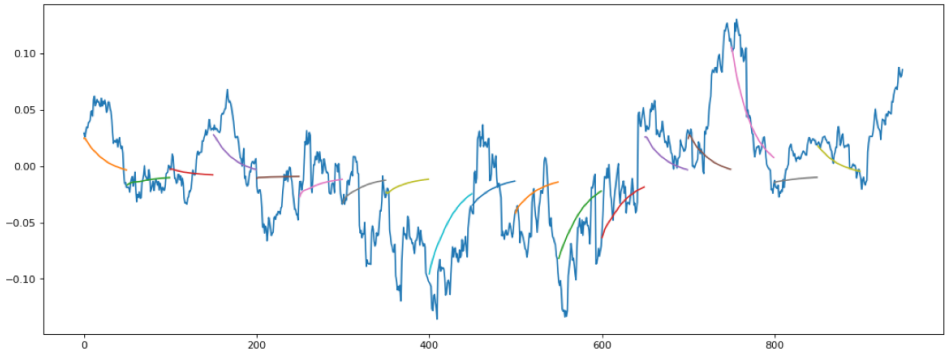

Figure 12: Sequence-by-Sequence prediction (4 Epoch, 50 Batch size)

| Bar category | Precision | Recall | f1-score |
|---|---|---|---|
| 1 | 0.82 | 0.73 | 0.77 |
| 2 | 0.74 | 0.64 | 0.68 |
| 3 | 0.62 | 0.67 | 0.64 |
| 4 | 0.59 | 0.65 | 0.61 |
| 5 | 0.74 | 0.83 | 0.78 |
| 6 | 0.89 | 0.82 | 0.85 |
| 7 | 0.62 | 0.68 | 0.64 |
| 8 | 0.75 | 0.73 | 0.73 |
| 9 | 0.82 | 0.83 | 0.82 |
| 10 | 0.83 | 0.74 | 0.78 |

Table 1: Accuracy of using multiple features in NN model

## 6 Future work

There are several path to look into in extending our work. Among them we thought of doing the following as our future work.

1. We want to show different deep neural network model comparison on stock analysis and find out the best fitted model for our case.

2. Different benchmark comparison with market alpha.

3. Make a framework that is easily extendable with different prediction model.

## 7 Conclusion

In this paper, we presented an stock prediction framework called SPAL which consists of various types of NNs. We attempted to predict the stock value with different network topologies with different features vector. Here we show how NN can be used as a good stock predictor that can easily trained compared to other machine learning model. We proposed different neural network architecture and contrasted them against different hand-based state-of-the art statistical indicator and machine learning technique. We shown that given enough data and different features vector neural network performs can better than other machine learning model and it also becomes much easier to train.

## References

[1] Chart Pattern. `https://medium.com/stockswipe-trade-ideas/chart-patterns-f0a7ceb2bcca`.

[2] W. Bao, J. Yue, and Y. Rao. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PloS one*, 12(7):e0180944, 2017.

[3] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu. Deep learning-based classification of hyperspectral data. *IEEE Journal of Selected topics in applied earth observations and remote sensing*, 7(6):2094–2107, 2014.

[4] X. Ding, Y. Zhang, T. Liu, and J. Duan. Deep learning for event-driven stock prediction. In *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[5] T. Fischer and C. Krauss. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669, 2018.

[6] Z. Guo, H. Wang, Q. Liu, and J. Yang. A feature fusion based forecasting model for financial time series. *PloS one*, 9(6):e101113, 2014.

[7] D. N. T. How, C. K. Loo, and K. S. M. Sahari. Behavior recognition for humanoid robots using long short-term memory. *International journal of advanced robotic systems*, 13(6):1729881416663369, 2016.

[8] T.-J. Hsieh, H.-F. Hsiao, and W.-C. Yeh. Forecasting stock markets using wavelet transforms and recurrent neural networks: An integrated system based on artificial bee colony algorithm. *Applied soft computing*, 11(2):2510–2525, 2011.

[9] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi. Time series forecasting using a deep belief network with restricted boltzmann machines. *Neurocomputing*, 137:47–56, 2014.

[10] F. Shen, J. Chao, and J. Zhao. Forecasting exchange rate using deep belief networks and conjugate gradient method. *Neurocomputing*, 167:243–253, 2015.

[11] B. Wang, H. Huang, and X. Wang. A novel text mining approach to financial time series forecasting. *Neurocomputing*, 83:136–145, 2012.

[12] A. Yoshihara, K. Fujikawa, K. Seki, and K. Uehara. Predicting stock market trends by recurrent deep neural networks. In *Pacific rim international conference on artificial intelligence*, pages 759–769. Springer, 2014.