Report for Machine Learning Final Project

--ImprovinA'g Stock Price Prediction Using PCA and LSTM

Zicheng He, Shengsong Qu, Wenyuan Zhou

Abstract

Followed the ideas from our research paper, this study focuses on predicting stock closing prices by using Recurrent Neural Network (RNNs). A Long Short-Term Memory(LSTM) model, a type of RNN coupled with stock basic trading data and technical indicators, is introduced as a novel method to predict the closing price of the stock market. We realize dimension reduction for the technical indicators by conducting Principal Component Analysis(PCA). To train the model, we used optimization strategies like adaptive moment estimation(Adam), Glorot uniform and orthogonal initialization. This case study is conducted on APPLE's (AAPL) stock price dating from 2000-2019. We used a series of evaluation criteria to assess our model. Besides reconstructing models utilized by previous researchers, our team commits to 1) Adjusting LSTM's hyperparameters. Looking for change in hyperparameters and its effects in predicting the stock price; 2) Build up strategies to exam the profitability of our model; 3) Add more new factors(like Fama-French 5 factors) and test them on different data, check and analyze the performance and conclude this report.

1 Introduction

At present, ANNs are regarded as the state-of-the-art theory and technique for regression and classification applications. A recurrent neural network (RNN) is a special kind of ANN, designed to learn sequential or time-varying patternsLong short-term memory (LSTM) is an important kind of RNN that excels at remembering values for either long or short periods of time RNN has many advantages in regressions, however, the biggest problem is that it can bring along the vanishing or exploding gradient to our learning process, which can invalidate training model. To solve this problem, LSTM use multiple gates to control the input information and improve its capabilities of remembering long-term information. Therefore, LSTM has been shown to outperform other RNNs on tasks involving long time lags.

This study proposes and validates a novel stock prediction model on the basis of LSTM, stock basic trading data, stock technical indicators, and principal component analysis (PCA). The model is designed to predict the closing price of the next day. Our first major contribution is that we effectively reconstruct previous stock prediction system using LSTM and improve the model with a method of combining basic stock trading data and technical indicators as the input variables by PCA. Only technical indicators are associated with the PCA unit.

The remainder of this letter is organized as follows. In section 2, we will briefly introduce the data of our case study. Section 3 Shows our model inputs and how we adjust data according to different models. Sections 4 goes detailed with model architecture, the core method is LSTM and PCA, we will talk about optimization strategies like Adaptive moment estimation(Adam), Glorot uniform and orthogonal initialization. In Section 5, we will evaluate our training result by looking at model's forecasting performance. Finally, we will conclude our research and provide our reflections in improving the model.

2 Data Description

Our case study is presented on the basis of APPLE's (AAPL) historical stock price. The data set has 4882 historical observations from January 3, 2000, to December 3rd, 2019. The dataset looks as listed in Table 1.

Table 1: Model Original Data Set

Date	OP	HI	LO	CL	AD	VO
2000-01-03	3.7455	4.0178	3.6317	3.9978	3.4785	133949200
2000-01-04	3.8660	3.9508	3.6319	3.6607	3.1852	128094400
2000-01-05	3.7053	3.9486	3.6786	3.7143	3.2318	194580400
2000-01-06	3.7901	3.8214	3.3929	3.3929	2.9521	191993200
2000-01-07	3.4464	3.6071	3.4107	3.5536	3.0920	115183600

3 Model Inputs

3.1 Input Variable Section

Our model's input variables include basic historical trading data and technical indicators. There are six variables in the basic trading data set. The open price (OP) is the price at which a security first trades when the exchange opens on a given trading day. The closing price (CL) is the final price at which a security is traded on a given trading day. The high price (HI) is the highest price at which a stock trades over the course of a trading day. The low price (LO) is the lowest price at which a stock trades over the course of a trading day. The adjusted price (AD) is a stock's CL on any given day of trading that has been amended to include any distributions and corporate actions that occurred at any time prior to the next day's open. The volume (VO) is the total quantity of shares or contracts traded for a specified security.

Stock technology indicators can be adopted to predict the performance of company's stock price (Thomas, 2001). Technical indicators are mathematical calculations on the basis of stock basic trading data. They have been proved to have abundant and latent information about stock markets. Employing technique indicators has had better effects, as some technique indicators contain information of that very day and information on previous days. For instance, Price Rate Of Change(PROC) provides the closing price information for the past 12 days.

In our research, 13 stock market technical indicators are selected as the input variables (we list the calculation formula of technical indicators in Table 2):

Table 2: Technical Indicator and Their Formulas

Technical Indicator	Formula
MACD	$MACD = EMA_{(CL,12)} - EMA_{(CL,26)}$
CHO	$CHO = EMA_{(AD,3)} - EMA_{(AD,10)}$
Highest	$Highest(t) = \max\,(CL_i, where\ i\ in\ [1,20])$
Lowest	$Lowest(t) = \min\,(CL_i, where\ i\ in\ [1,20])$
SO-%K	$SO - \%K = \frac{\left(CL - Lowest(5)\right)}{Highest(5) - Lowest(5)} * 100\%$

SO-%D	$SO-\%D = MA(STOS-\%K, 3)$
W-R%	$W-R\% = \frac{(Highest(n)-CL)}{Highest(n)-Lowest(n)} * 100\%$
AC	$AC(t) = AO - MA(AO, t)$
PROC	$PROC = \frac{CL - CL_{t-12}}{CL_{t-12}} * 100\%$
VROC	$VROC = \frac{VO - VO_{t-12}}{VO_{t-12}} * 100\%$
OBV	$If\ CL \geq CL_{previous-day}, OBV = OBV_{previous-day} + VO$ $If\ CL < CL_{previous-day}, OBV = OBV_{previous-day} - VO$
Ultimate_Osc	$UO = \frac{A_7 * 4 + A_{14} * 2 + A_{28}}{4+2+1} * 100$ where A=Average; Buying Pressure(BP)=Close-Min(Low, PC); PC=Prior Price; True Range(TR) =Max(High, Prior Close)-Min(Low, Prior close) $A_i = \frac{\sum_{p=1}^{i} BP}{\sum_{p=1}^{i} TR}$
MFI	$MFI = 100 - \frac{100}{1 + Money\ Flow\ Ratio}$ here $Money\ Flow\ Ratio = \frac{14\ period\ positive\ money\ flow}{14\ period\ negative\ money\ flow}$ Raw Money Flow=Typical Price*Volume $Typical\ Price = \frac{High + Low + Close}{3}$

- The moving average convergence divergence (MACD) shows the relationship between two moving averages of prices and reveals changes in strength, direction, and duration of a trend in a stock's price.
- The Chaikin oscillator(CHO)is used to measures the AD line of the MACD.
- Highest(t) is the highest closing price value during the past *20* trading days.
- Lowest(t) is the lowest closing price value within the past *20* trading days.

- The stochastic oscillator (SO) attempts to compare the closing price of a security to the range of its prices over a certain period of time.
- The Williams %R(W-R%) measures over bought and oversold levels.
- Acceleration (AC) measures the acceleration and deceleration of price.
- The price rate of change (ROC) measures the percentage change in price between the current price and the price *n* periods in the past.
- The Volume rate of change(VROC)is used to gauge the volatility in a security's volume.
- On-balance volume (OBV) uses volume ow to predict changes in stock price.
- Ultimate Oscillator is a notion of buying or selling "pressure" represented by where a day's closing price falls within the day's true range.
- Money Flow Index identifies overnight or oversold conditions in an asset, and can be used to spot divergences which warn of a trend change in price

3.2 PCA for Technical Indicators

The reason we use PCA with our technical indicators is to extract the most critical information from the data set. The technical indicators have some redundant information and highly correlated features. It is generally known that correlatedi information presented to the neural network will have an adverse impact on the learning ability of neural network, for example, by easily falling into local minima solutions, a heavier training burden, or reducing the generalization ability of the network (Mohamadsaled & Hoyle, 2008; Pan, Rust, & Bolouri, 2000). The relevant data will decrease the distinctiveness of data representation, thus introducing confusion in the neural network. There are various methods to reduce dimensions, follow the logic from the research paper, they choose PCA because they consider it as the most suitable method. For example, linear discriminant analysis(LDA) can decrease dimension, but it pertains to supervised learning (which means our data need a label) and is applied for classification, in this case, it's not suitable for our research. PCA here is used to reduce the obvious correlation between input variables, such as MFI and William%R. PCA can decrease the dimension of the data and extract the intrinsic features of data.

In our study of reconstructing the model in the paper, only some of the input variables are processed by PCA. The basic trading input variables are not processed by

PCA, while technical indicator data are decreased to extract the crucial and intrinsic features from high-dimensional data space using PCA.

Specifically, in LSTM models, there are 4 types of Input Method(IM): IM-A merely involves basic trading data; IM-B involves basic trading data and technical indicators; IM-C involves basic trading data and technical indicators. All in- put variables are processed by PCA. IM-D involves basic trading data and technical indicators. Only technical indicators are processed by PCA. Thus, LSTM coupled with IM-D represents the model we propose. These different input methods are shown in Figure 1.

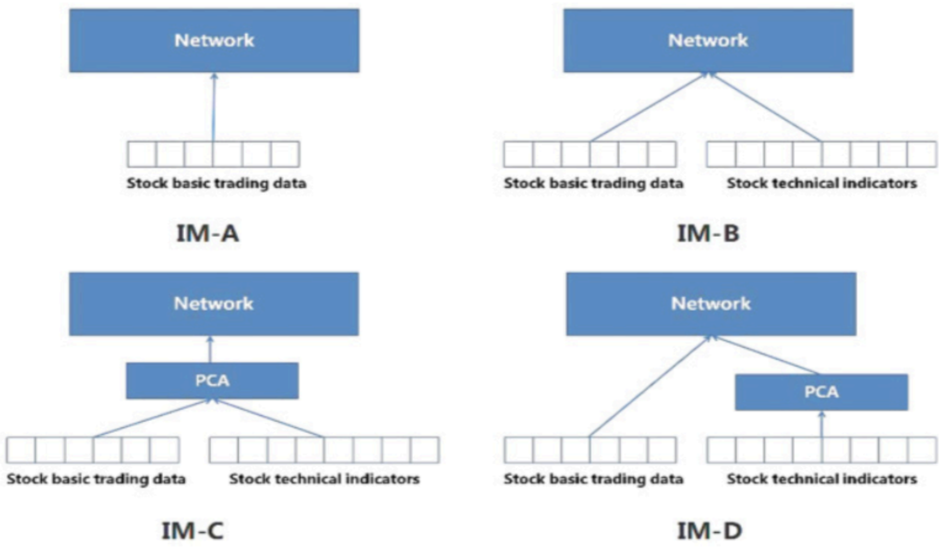

Figure 1: Different input methods

3.3 Inputs Sequence

The input variables need to be changed into sequence data. The input data segmentation is made by a sliding window of 20 days. This process is described in Figure 2. A sliding window is applied to the entire data set to extract the input data used by the forecasting model. After that, each input variable has 20 days of observation. The blue block represents the input data, which include data form 20 trading days. The white block represents the output data, the closing price of the next day. The training samples and testing samples are obtained sequentially. Thus, we obtain results on predictions based on stock market behavior over the previous 30 days.

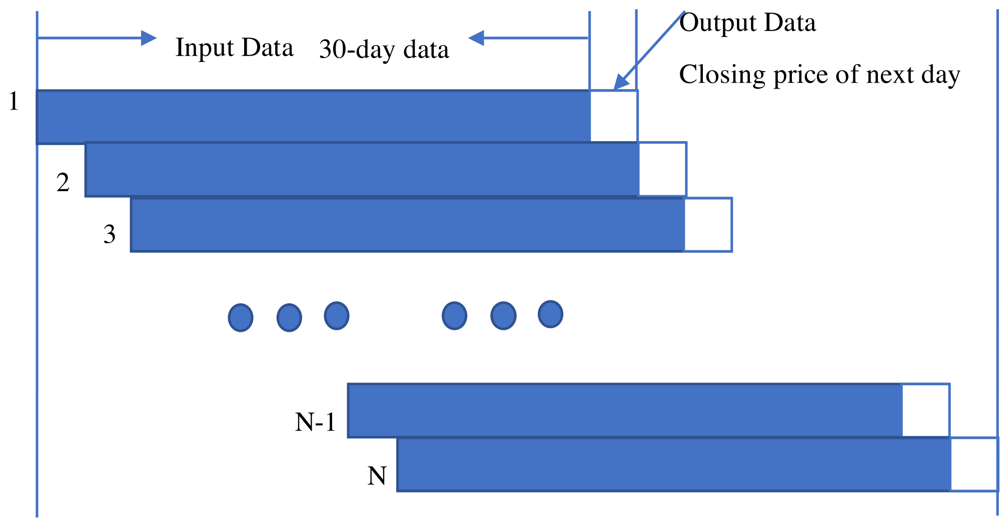

Figure 2: Diagram of building up the sequence data set

4 Model Architecture

4.1 Long Short-Term Memory Neural Networks

In feedforward neural networks (FFNNs), there are no connections among the neurons in the same layer, while an RNN a kind of ANN, has connections between neural units that form a directed cycle. LSTM has a novel structure, a memory cell, which contains four main elements: an input gate, a forget gate, an output gate, and a neuron unit. This special structure makes decisions about what information to store and when to allow reading, writing, and forgetting via the three gates that open and close. Figure 3 illustrates how data flow through a memory cell and are controlled by the gates. The behavior of memory cell can be described in the following ways:

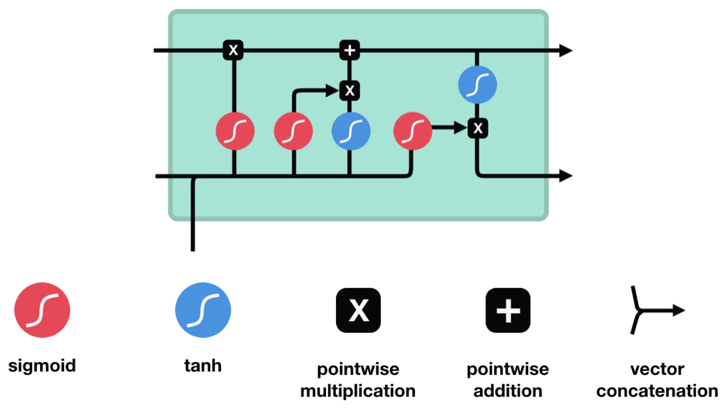

Figure 3: The inner architecture of an LSTM cell

$$\boxed{f_t} = \sigma\left(W_f \cdot [h_{t-1}, X_t] + b_f\right) \qquad \boxed{i_t} = \sigma(W_i \cdot [h_{t-1}, X_t] + b_i)$$

$$\boxed{C_t} = C_{t-1} \cdot f_t + i_t \cdot \mathbb{C}_t$$

$$h_t = O_t \cdot \tanh\,(C_t)$$

$$\boxed{\mathbb{C}_t} = tanh(W_c \cdot [h_{t-1}, X_t] + b_c)$$

$$\boxed{O_t} = \sigma(W_o \cdot [h_{t-1}, X_t] + b_o)$$

In our prediction model, three gates' activation functions are all sigmoid, which means $\sigma_1(x) = \sigma_2(x) = \sigma_3(x) = \frac{1}{1+e^{-x}}$

4.2 Neural Network Architecture

Our model consists of three layers in a neural network: the input layer, the LSTM layer (hidden layer), and the output layer (see Figure 4). Every unit in a layer is connected to all the other units in adjacent layers. There are 19 original input variables. After PCA is applied, the input layer has 13 neural units, the LSTM layer has 10 LSTM neural units, and the output layer has only 1 unit. Thus, the output layer is considered the full connection layer.

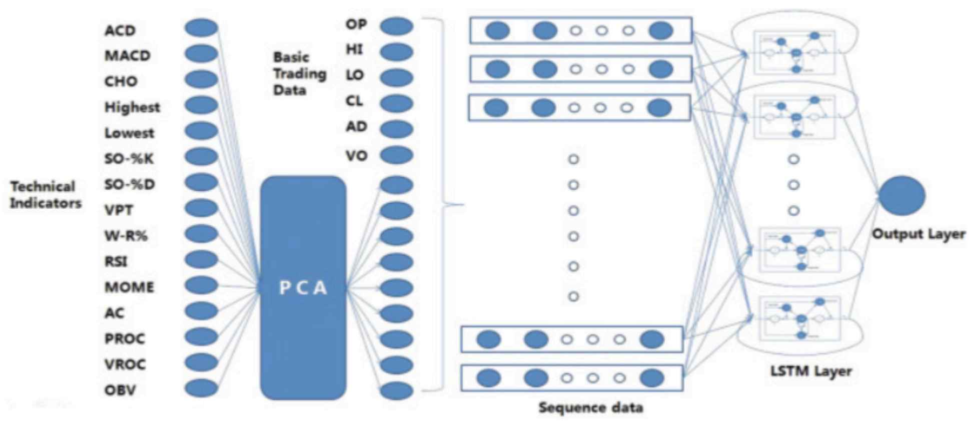

Figure 4: Architecture of the proposed neutral network

4.3 Network Training Method

We predict on APPLE's next day's closing price. We reserve three types of test data: 100 days represent the short-term prediction; 200 days denote the medium-term prediction; and 400 days represent the long-term prediction. For each type of test set, the remaining data are training data. As for training processes, they can be divided into data processing (data normalization), weights initialization and parameters optimization three parts.

4.3.1 Data Processing

In this part, data processing just refers to data normalization, it's more like a "scaling-down" transformation of the attributes. In our model, min-max normalization is used to achieve normalization. The equation for data normalization is given by:

$$v_i' = \frac{v_i - v_{min}}{v_{max} - v_{min}}(new_{max} - new_{\mathrm{min}}) + new_{min}$$

where v = (v1,v2,...,vn), v_i' is the ith normalized data, $\mathrm{new}_\mathrm{max}$= 1, $\mathrm{new}_\mathrm{min}$= 0

We use zero-mean for further data processing so each feature has a mean value of zero. For practical reasons, it is advantageous to center the data. We give its formula as follows:

$$v_i'' = v_i' - v_{mean}'$$

where v_i'' is the zero mean data.

At the last step of preprocessing, we shuffle the training data randomly after each epoch of training. This shuffling can break the strong correlations between consecutive training data.

4.3.2 Network weights initialization

In our prediction model, two approaches are used for weight initialization. We use Glorot uniform as the initialization method of the input matrices and orthogonal initialization for the recurrent matrices.

Glorot uniform initialization generates random weights and biases by sampling from a uniform distribution function (Glorot & Bengio, 2010). In this model, the feedforward weights are initialized by sampling from the uniform distribution the following initialization procedure to approximately satisfy our objectives of maintaining activation variances and back- propagated gradient variance as one moves up or down the network:

$$W \sim U[-\frac{\sqrt{6}}{\sqrt{numUnits1 + numUnits2}}, \frac{\sqrt{6}}{\sqrt{numUnits1 + numUnits2}}]$$

where numUnits1 = the number of neural units in the low layer; numUnits2 = the number of neural units in the high layer.

In the internal element of the LSTM layer, the initialization uses the orthogonal matrix method, which generates a random orthogonal matrix. This method has been shown to have good performances in RNNs (Le, Jaitly, & Hinton, 2015). Orthogonal matrices have many interesting properties, but the most important one for us is that all its eigenvalues have an absolute value of one. This means that no matter how many times we per- form repeated matrix multiplication, the resulting matrix neither explodes nor vanishes. This allows gradients to backpropagate more effectively.

4.3.3 Optimization Algorithm for Parameters Update

We utilize Adam to perform optimization. This method is computationally efficient, has few memory requirements, and is invariant to diagonal rescaling of the gradients (Kingma & Ba, 2014). The core of Adam is estimating the first and second moments of gradients to do the update.

5 Result Evaluation

The test results were measured majorly from the perspective of its forecasting performance. The first measure is the mean absolute error (MAE), which is a common approach in the forecasting domain (Hyndman & Koehler, 2006). The mean absolute percentage error (MAPE) computes the percentage of error between the actual and predicted values, and we include it as our second measure. In our report, we will use both measure to evaluate the forecasting performance of our model.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} | \tilde{\varphi}_i - \varphi_i | \qquad MAPE = 100 * \frac{1}{n}\sum_{i=1}^{n} | \frac{\tilde{\varphi}_i - \varphi_i}{\varphi_i} |$$

Figure 5 to Figure 7 plots predicting stock price under different model inputs; Figure 8 and Figure 9 separately calculates MAE and MPAE under different models with various testing samples.

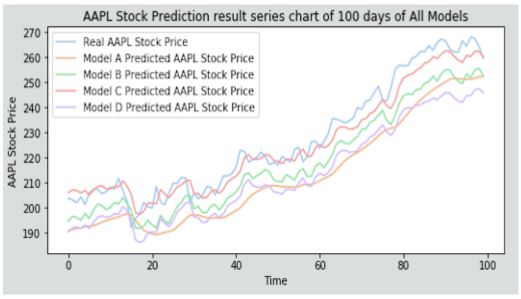

Figure 5 Testing Sample=100

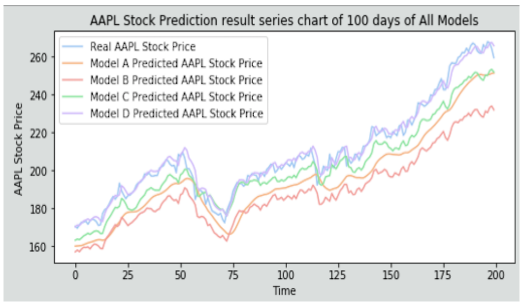

Figure 6 Testing Sample=200

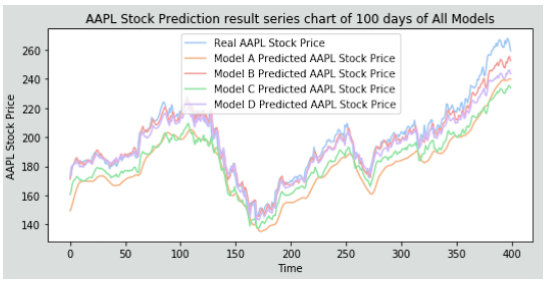

Figure 7 Testing Sample=400

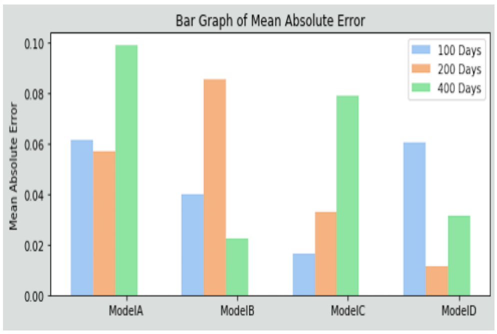

Figure 8 MAE under various models

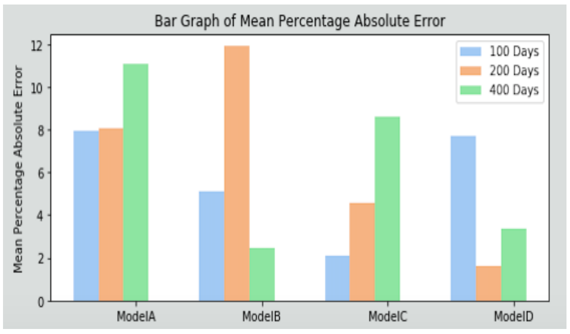

Figure 9 MPAE under various models

Table 3 and Table 4 provides MAE and MPAE results under different models with various testing days(where N stands for number of testing samples).

Table 3: MAE with different Models under various testing samples

	Model A	Model B	Model C	Model D
N=100	0.02115	0.05630	0.11235	0.01582
N=200	0.03514	0.01326	0.03405	0.02378
N=400	0.02784	0.04610	0.04399	0.07395

Table 4: MPAE with different Models under various testing samples

	Model A	Model B	Model C	Model D
N=100	2.71952	7.30391	14.47475	2.04399
N=200	4.82203	1.86439	4.81546	3.22805
N=400	3.21479	4.99365	4.68181	8.07936

The results above reflect our target model (which is model D) does illustrate better and stronger capability of predicting the future stock price as our predicting trend of stock price is very consistent with the actual one, and our model D's MAE and MPAE is relatively lower comparing to other models.

But when we run our models multiple times, we found results vary, see Figure 10 to Figure 11.

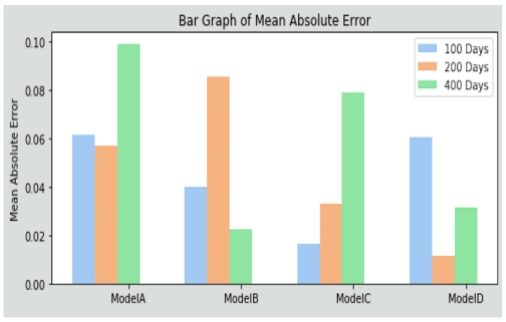

Figure 11 First time MAE

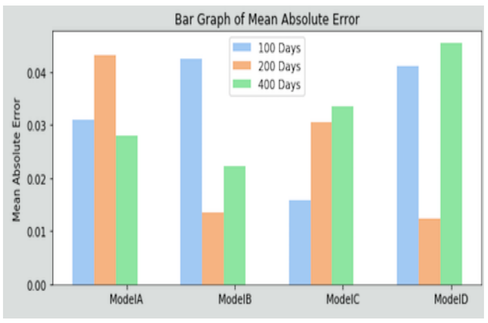

Figure 12 Second time MAE

We consider this problem comes from data processing, as every time we do weights initialization, it will bring new inputs into our model. Therefore we try to explore if there is a pattern hidden behind these varying results. We automatically run model many times, and calculate the average MAE and MPAE to see if it reflects some pattern (Figure 13)

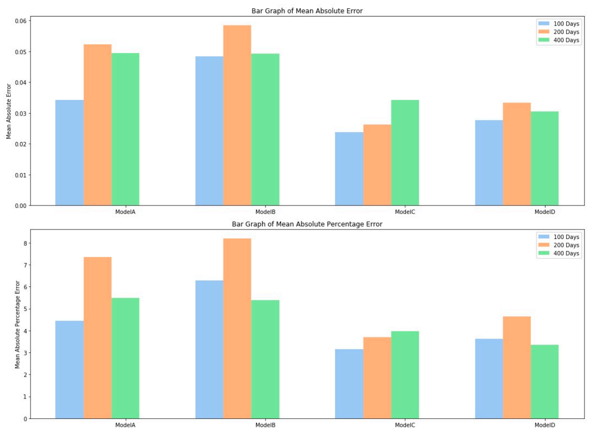

Figure 13 Average MAE and MPAE

Results above shows model D does a better job in predicting (with lower MAE and MPAE)

6 Conclusion & Reflection

LSTM+PCA model does show us its efficiency in predicting the stock price, but its disadvantage is obvious as well. If we want to get a pattern of price trending, we may have to run the model a lot of times, as it can be very time-consuming. Therefore we reflect to fix the Tensorflow seed and try to optimize parameters to see if it brings better result.

First, we fix the Tensorflow seed, so that every time when we randomly initialize weights, we will have same input into the model. Then we try to find relationship between different Units and Batch_size choice individually to the final predicting result. Figure 14 and Figure 15 shows our result.

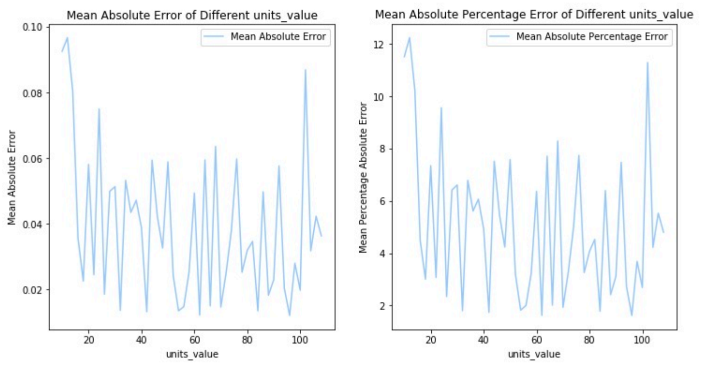

Figure 14 Effects of Different Units Selection to Forecasting Performance

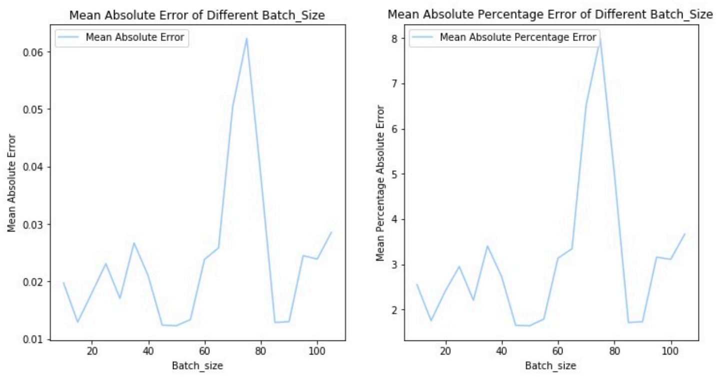

Figure 15 Effects of Different Batch_size Selection to Forecasting Performance

On the basis of hyperparameter selection above, we choose the number of units and batch_size that bring the best predicting performance to construct our model. Then we want to test if this model can really help us earn some money in practice. Therefore we design four strategies using this model.

Strategy 1: Long only. If we predict the stock price next day is going to be higher than price today, we long and hold the stock.

Strategy 2: Short only. If we predict the stock price next day is going to be lower than price today, we short the stock.

Strategy 3: Long only for trend. If today we predict the price tomorrow is going to be higher than the predicted price we made yesterday about today's price, we long and hold the stock.

Strategy 4: Short only for trend. If today we predict the price tomorrow is going to be lower than the predicted price we made yesterday about today's price, we short the stock.

Without considering the transaction cost, we set our holding period for 1 day and 2 days, using strategies mentioned above and see how their P&L perform. Resutls in Figure 16 and Figure 17 display if we hold the position for only one day, the results vary a lot. The best strategies earn you 5 percent but the worst one get you 30 percent loss. But if we hold 2 days, things change, all the strategies go fairly well, which bring our P&L all in positive rate.

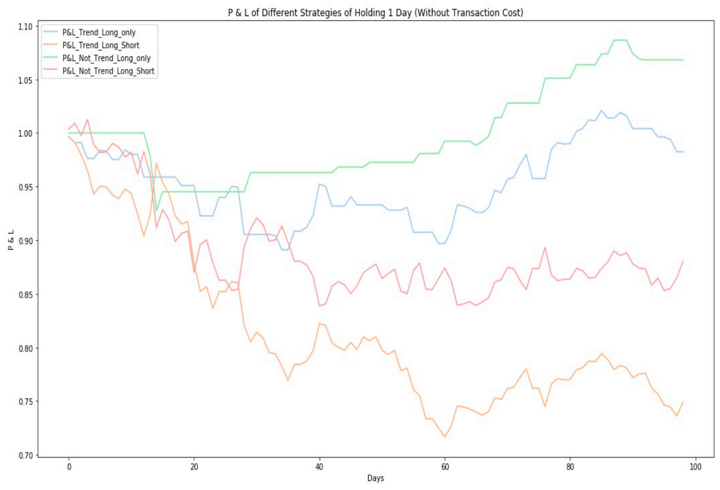

Figure 16 Holding period=1 day (Initial position all start at 1)

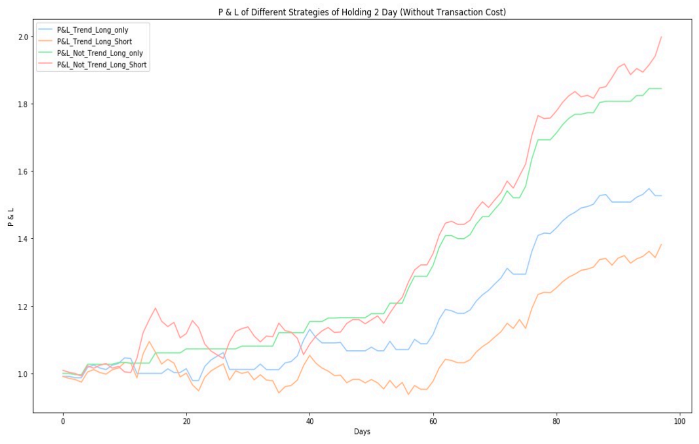

Figure 17 Holding period=2 day (Initial position all start at 1)

Apart from putting the model into practice and checking its profitability, we come up with the idea about adding more new factors into our features and get them involved in the process of PCA. In this case, we tried on Fama-French 5 factors, and made a comparison between the original model and the model we add new factors.

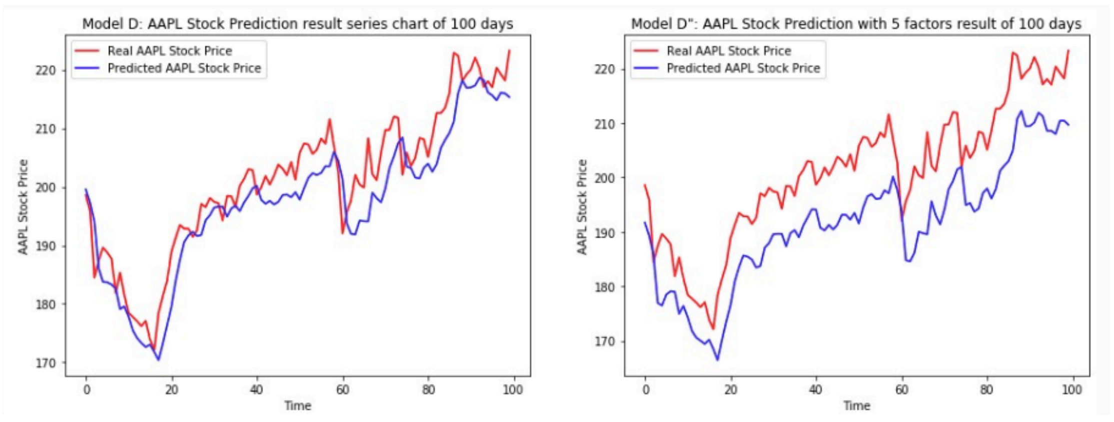

Figure 18 Prediction Comparison by including FF5factors

From Figure 18 You can see that the prediction gets worse. That may be due to the fact that the FF5-factors are not only for AAPL but for all the market.

Follow this logic, we continue to substitute our AAPL data with S&P 500 Index data and run the model again. You can see from the Figure 19, to the Left, the original model predicts well, which means our model can be generalized to predict the price of either stock or index. However, things go wrong again if we use FF-5factors. So perhaps the best way is still to include the technical indicators.

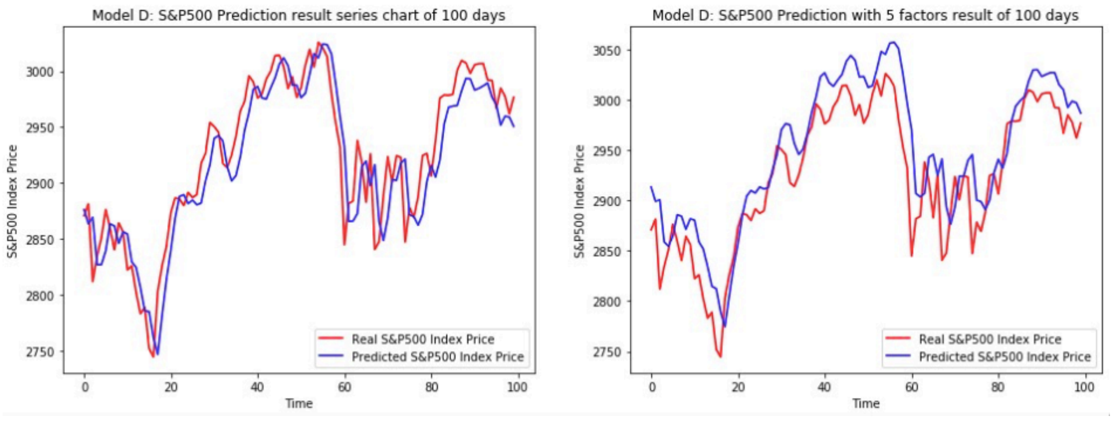

Figure 18 Price Prediction Based on S&P 500 Index

Reference

Forecasting East Asian Indices Futures via a Novel Hybrid of Wavelet-PCA Denoising and Artificial Neural Network Models, Jacinta Chan Phooi M'ng & Mohammadali Mehralizadeh, May 2016

Improving Stock Closing Price Prediction Using Recurrent Neural Network and Technical Indicators, Tingwei Gao & Yueting Chai, 2018