Stock Market Prediction by LSTM Model

Deqing Qu
Oregon State University
Corvallis, Oregon
qud@oregonstate.edu

Mian Xie
Oregon State University
Corvallis, Oregon
xiemia@oregonstate.edu

Mihai Dan
Oregon State University
Corvallis, Oregon
danm@oregonstate.edu

Abstract

Predicting stock prices is a difficult task due to the volatile nature of the stock market. However, it is highly sought after by investors due to its potential benefits. Currently, the LSTM model has been proven to yield the best results within the financial domain. In this paper, we introduce our LSTM network which assesses historical factors, as well as fundamental factors, to predict stock prices. We present the model along with prediction results of various companies, compared to results of state of the art approaches.

1. Introduction

The art of forecasting stock prices has been a difficult task for many of the researchers and analysts. In fact, investors are highly interested in the area of stock price prediction.

Stock market prices are highly unpredictable and volatile. However, as the technology advances, the opportunity to gain an accurate prediction of the stock market is increased. With the prediction result, investors could maximize the profit and lower the risk of investment.

Recurrent neural networks (RNN) have been proved one of the most powerful models for processing sequential data. More specifically, long Short-Term (LSTM) memory is one of the most successful RNN architectures. LSTM suits to grasp the structure of data dynamically over time with high prediction capacity.

Here, we introduce RNN with LSTM model to construct the multi-factors model to predict market price, which better reflects the situation of market changes and helps create accurate predictions. Comparing with the traditional way simply assessing the close price of stock market, we also introduce the fundamental data of companies which has a strong relationship with their stock price. Hence, the input data set includes 2 categories data.

- *Historical factors*: change, change over time, change percent, close price, open price, low price, high price, unadjusted volume, volume
- *Fundamental factors*: Price-to-Earnings Ratio, Price-to-Book Ratio

This serial data set will be input into LSTM model and be utilized to predict the price of stocks and generate portfolios with hedging strategies to achieve higher excess return and lower risk of volatility.

In addition, we build this model on TensorFlow which is a core open source library to help develop and train Deep Learning models.

2. Backgroud

In finance, Alpha measures the active return on an investment, which is the excess return of the investment relative to the return of a benchmark index. Alpha is the portion of a portfolio's return that cannot be attributed to market returns and is thus independent of them. Beta measures of the volatility or systematic risk of a portfolio in comparison to the whole market and is the portion of the return generated from a portfolio that can be attributed to overall market returns. Alpha and Beta are two key coefficients in the capital asset pricing model (CAPM) used in modern portfolio theory and are closely related to other important quantities such as standard deviation, R-squared and the Sharpe ratio.

Hedging is used to reduce the risk of adverse price movements in an asset class by taking an offsetting position in a related asset. Beta hedging involves reducing the overall beta of a portfolio by purchasing stocks with offsetting betas.

Deep learning approaches applied in investment Deep neural networks models have proven powerful for tasks as diverse as language translations, video captioning, video recognition, and time series modeling. Recurrent neural networks and long short-term memory (LSTM) networks are state-of-the-art techniques for sequence learning and are suitable for the financial time serials prediction. A number of recent papers and hedge funds consider deep learning approaches to predicting stock market performance.

3. Related Works

Alberg and Lipton [1] proposed an investment strategy that constructed portfolios of stocks today based on predicted future fundamentals. They trained deep neural networks to forecast future fundamentals based on a trailing 5-years window, then sorted the set of available stocks according to the fundamental factor and construct investment portfolios comprised of those stocks which scored highest. Simulations demonstrate that investing based on the predicted factors yields a compound annualized return (CAR) of 17.1%, vs 14.4% for a normal factor model and a Sharpe Ratio of 0.68 vs 0.55.

Sezer [8] present a new stock trading and prediction model based on a multilayer perceptron (MLP) neural network utilizing technical analysis indicator values as features. Three most commonly used technical indicators, RSI, MACD, and William % R were selected to be used. The model was trained and tested on Dow 30 stocks in order to see the evaluate the model. The results indicated that comparable results were obtained against the baseline Buy and Hold strategy even without fine tuning and optimizing the model parameters.

Edet [4] predicted the movements of the S&P 500 index using variations of the recurrent neural network. In predicting the S&P 500 index, they considered 14 economic variables, 4 levels of hidden neurons of the networks and 5 levels of epochs. In applying these networks (i.e. Simple RNN, LSTM, GRU) to forecast the movement of S&P 500 index, they used the concept of experimental design to choose the features that were most appropriate for prediction. In each case of the three neural networks to be used for prediction, they performed 20 experiments to determine which of the experiments gave the best accuracy score. The three selected experiments for these models were able to predict the movement of the S&P 500 index with an accuracy of 75%, 74% and 74% respectively.

Fischer and Krauss [5] applied LSTM networks to a large-scale financial market prediction task on the S&P 500, from December 1992 to October 2015. They framed a proper prediction task, derive sensible features in the form of 240-day return sequences, standardize the features during preprocessing to facilitate model training, discuss a suitable LSTM architecture and training algorithm, and derive a trading strategy based on the predictions, in line with the existing literature. With daily returns of 0.46 percent and a Sharpe Ratio of 5.8 prior to transaction costs, they found LSTM networks to outperform memory-free classification methods, i.e., a random forest (RAF), a deep neural network (DNN), and a logistic regression classifier (LOG).

Khaidem [6] applied random forest to generate a model to predict the direction of stock market prices. They proposed a new method to minimize the risk of stock market investment by using a powerful machine learning algorithms known as ensemble learning. The learning model is an ensemble of multiple decision trees and the predictive output of the model can be used to support the decisions of those who invest in the stock market. However, there are some drawbacks in the random forest: the heuristic learning rule does not effectively minimize the global training loss; the model size is usually too large for many real applications [2, 3]. Ren [7] proposed a global refinement approach to address these issues. The global refinement approach relearns the leaf nodes of all trees under a global objective function in order to use the complementary information between multiple trees.

4. Prediction Model

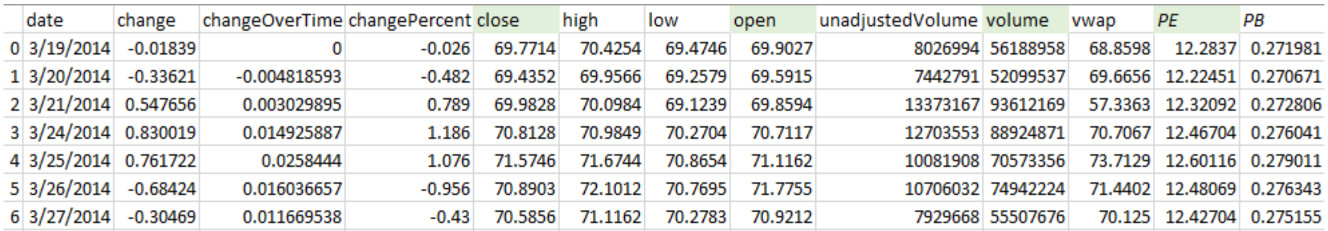
4.1. Data Details

Our model uses historical and fundamental data when training and creating predictions. Historical data refers to trading patterns of the stock itself, such as price, volume, etc. and fundamental data involves any data, besides historical data, that is expected to impact the price or perceived value of a stock.

The historical data was queried from The Investors Exchange (IEX) API, which provides various data based on company ticker and time frame. More specifically, we queried the API for `change`, `changeOverTime`, `changePercent`, `close`, `high`, `low`, `open`, `unadjustedVolume`, `volume`, and `vwap` over the most recent five years. This data provides insight on the change over time in stock value. The volume-weighted average price, or `vwap`, is the ratio of the value traded to total volume traded over the span of a day.

The fundamental data was queried from the Morning Star API, which provides similar functionality as the IEX API, but includes more fundamental information such as key ratios. Based on previous investing experience, we decided to use the Price-to-Earning (P/E) Ratio and Price-to-Book (P/B) Ratio to include in our initial design of the model. The P/E ratio is used as a measure showing whether a company's stock price is overvalued or undervalued, as well as how the stock's valuation compares to that of its industry group. The P/E ratio also helps determine the market value of a stock compared to the company's earnings. The P/B ratio is used to compare a company's net assets available to common shareholders relative to the sale price of its stock. Similarly to the P/E ratio, the P/B ratio can be used to determine whether a company is undervalued or overvalued. Both P/E and P/B ratios are good measures of a company's financial health, and have the potential to yield better results if used in a deep learning model.

Historically, the P/E and P/B ratios are calculated using the previous quarter's balance sheet. Since we only have access to yearly balance sheets, we calculated the P/E and

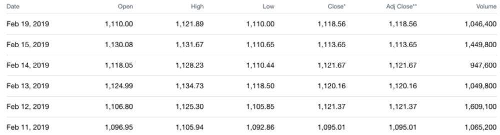


	date	change	changeOverTime	changePercent	close	high	low	open	unadjustedVolume	volume	vwap	PE	PB
0	3/19/2014	-0.01839	0	-0.026	69.7714	70.4254	69.4746	69.9027	8026994	56188958	68.8598	12.2837	0.271981
1	3/20/2014	-0.33621	-0.004818593	-0.482	69.4352	69.9566	69.2579	69.5915	7442791	52099537	69.6656	12.22451	0.270671
2	3/21/2014	0.547656	0.003029895	0.789	69.9828	70.0984	69.1239	69.8594	13373167	93612169	57.3363	12.32092	0.272806
3	3/24/2014	0.830019	0.014925887	1.186	70.8128	70.9849	70.2704	70.7117	12703553	88924871	70.7067	12.46704	0.276041
4	3/25/2014	0.761722	0.0258444	1.076	71.5746	71.6744	70.8654	71.1162	10081908	70573356	73.7129	12.60116	0.279011
5	3/26/2014	-0.68424	0.016036657	-0.956	70.8903	72.1012	70.7695	71.7755	10706032	74942224	71.4402	12.48069	0.276343
6	3/27/2014	-0.30469	0.011669538	-0.43	70.5856	71.1162	70.2783	70.9212	7929668	55507676	70.125	12.42704	0.275155

Figure 1. Sample of the AAPL data set

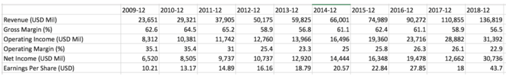
P/B ratios using the `close` price and the previous year's balance sheet for each example in the data set.

Currently, our model uses `close`, `open`, `volume`, and the P/E ratio to create predictions. These columns are highlighted in Figure 1, which depicts the first few inputs for the ticker *AAPL*. For future work, we would like to see if including more historical or fundamental data will improve performance, or introduce unnecessary noise to the data.



Date	Open	High	Low	Close*	Adj Close**	Volume
Feb 19, 2019	1,110.00	1,121.89	1,110.00	1,118.56	1,118.56	1,046,400
Feb 15, 2019	1,130.08	1,131.67	1,110.65	1,113.65	1,113.65	1,449,800
Feb 14, 2019	1,118.05	1,128.23	1,110.44	1,121.67	1,121.67	947,600
Feb 13, 2019	1,124.99	1,134.73	1,118.50	1,120.16	1,120.16	1,049,800
Feb 12, 2019	1,106.80	1,125.30	1,105.85	1,121.37	1,121.37	1,609,100
Feb 11, 2019	1,096.95	1,105.94	1,092.86	1,095.01	1,095.01	1,065,200

Figure 2. Stock Statistical Data



	2009-12	2010-12	2011-12	2012-12	2013-12	2014-12	2015-12	2016-12	2017-12	2018-12
Revenue (USD Mil)	23,651	29,321	37,905	50,175	59,825	66,001	74,989	90,272	110,855	136,819
Gross Margin (%)	62.6	64.5	65.2	58.9	56.8	61.1	62.4	61.1	58.9	56.5
Operating Income (USD Mil)	8,312	10,381	11,742	12,760	13,966	16,496	19,360	23,716	28,882	31,392
Operating Margin (%)	35.1	35.4	31	25.4	23.3	25	25.8	26.3	26.1	22.9
Net Income (USD Mil)	6,520	8,505	9,737	10,737	12,920	14,444	16,348	19,478	12,662	30,736
Earnings Per Share (USD)	10.21	13.17	14.89	16.16	18.79	20.57	22.84	27.85	18	43.7

Figure 3. Fundamental Data

4.2. LSTM Model Design

4.2.1 Data Preparation

(1) Sliding Window[9]

Suppose the stock prices is a time series of length N, defined as $p_0, p_1, ..., p_{N1}$ in which p_i is the close price on day i, $0 \leq i < N$. Imagine that we have a sliding window of a fixed size w and every time we move the window to the right by size w, so that there is no overlap between data in all the sliding windows(As shown in Figure 4).

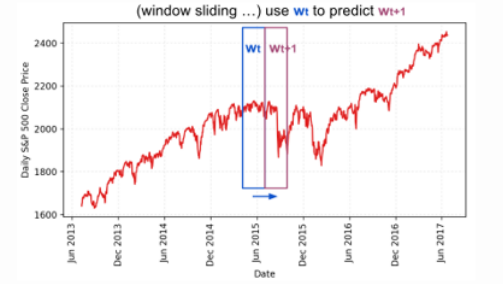


Figure 4. Input data with sliding windows

(2) Train/Test data split

We set $test\_ratio$ as 0.05 which means using 95% of data to train LSTM model and using 5% as the test data to validate how accurate the model is.

(3) Normalization

The increasing index brings the problem that most values in the test set are out of the scale and thus the model has to handle with some values never seen before. To solve the out-of-scale issue and make prediction process more quickly, normalization is necessary. We normalize all the data(including price, volume, PE, etc) in each sliding window. The task becomes predicting the relative change rates instead of the absolute values.

4.2.2 Model Structure

Our RNN model consists of LSTM cells as basic hidden units[9]. We use values from the very beginning in the first sliding window W_0 to the window W_t at time t,

$W_0 = (p_0, p_1, ..., p_{w1})$

$W_1 = (p_w, p_{w+1}, ..., p_{2w-1})$

...

$W_t = (p_{tw}, p_{tw+1}, ..., p_{(t+1)w-1})$

to predict the prices in the following window w_{t+1}:

$W_{t+1} = (p_{(t+1)w}, p_{(t+1)w+1}, ..., p_{(t+2)w-1})$

Considering how back propagation through time works, we usually train LSTM in a unrolled version so that we do not have to do propagation computation too far back and save the training complication.

It is common practice to create an unrolled version of the network, which contains a fixed number (num_steps) of LSTM inputs. The model is then trained on this finite approximation of the LSTM. This can be implemented by inputing inputs of length num_steps at a time and performing a backward pass after each prediction.

Thus, each sliding window contains $input\_size$ values and is considered as one single input unit. Then any num_steps consecutive input units are grouped into one training input data set, forming an unrolled version of LSTM for training(As shown in Figure 5). The corresponding label is the data unit right after them.

Therefore, we need to define 3 parameters: input_size, num_steps and batch_size to predict the stock price during a period of time.

- *input_size*: size of sliding window/ one training data point. It is the basic unit of the input and output. For example, if define this parameter as 10, means we will predict prices for 10 days.
- *num_steps*: the number of sliding windows used as the input data to predict price.
- *batch_size*: number of data to use in one mini-batch;

Thus, we define inputs and output/target as shown below. The placeholder in TensorFlow define the dimensions of input and output.

- inputs = tf.placeholder(tf.float32, [None, num_steps, input_size])
- targets = tf.placeholder(tf.float32, [None,input_size])

Obviously, the input is a 3 dimensions data set (the first *none* indicates the batch_size). The output is a 2 dimensional data set, where input_size indicates the number of days we can predict.

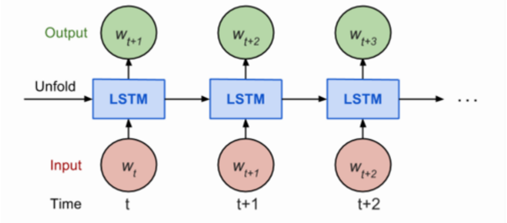


Figure 5. LSTM model with unfold version

5. Experiments

We first study the impact of important parameters (Section 5.1). We then compare LSTM model to other machine learning methods (Random Forest and Refined Random Forest) in stock price prediction (Section 5.2). At last we show the portfolios generated by our model (Section 5.3).

We choose three data sets for study, the same as those used in [6], i.e. *AAPL*, *MSFT*, *AMZN*. The time span of stock data ranges from 03/18/2014 to 03/18/2019. The ground truth and prediction prices of our model are shown in Fig. 6, 7, 8. Although there are some errors between the real price and prediction price (especially for *AMZN*), the predicted price has the same trend as the real price.

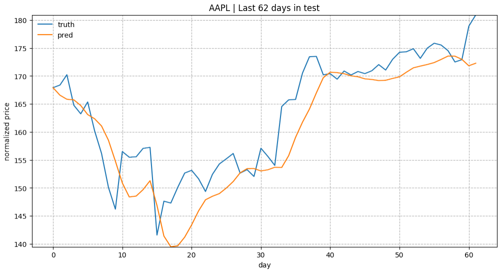


Figure 6. Results for AAPL stock obtained with LSTM model

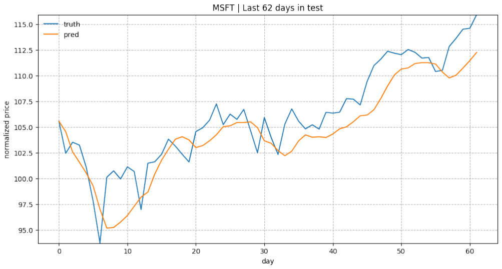


Figure 7. Results for MSFT stock obtained with LSTM model

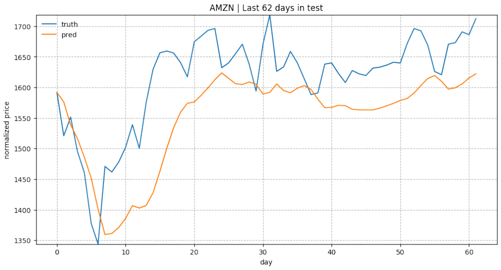


Figure 8. Results for AMZN stock obtained with LSTM model

5.1. Impact of important parameters

In this section, we investigate one important parameter that affect the prediction results.

Number of units in LSTM cell. In TensorFlow, the number of units in LSTM cell can be specified by the 'num_units' parameter for tf.nn.rnn_cell.LSTMCell. The 'num_units' can be interpreted as the analogy of hidden layer from the feed forward neural network (as shown in Figure 9). The number of nodes in hidden layer of a feed forward neural network is equivalent to the number of LSTM units in a LSTM cell at every time step of the network. Although it will take longer and increase the risk of over-fitting, using more units makes it more likely to perfectly memorize the complete training set. The results are shown in Table 1 and the mean squared error (MSE) decreases when the unit numbers increase.

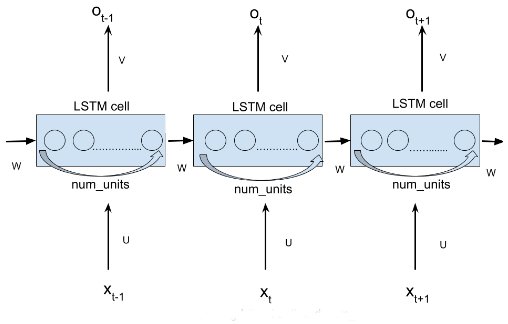


Figure 9. `num_units` in LSTM cells

Unit Num / MSE	AAPL	MSFT	AMZN
128	36.54	7.38	2347.55
256	28.15	5.92	1630.24
512	21.07	4.25	1262.46

Table 1. MSE using different number of units in LSTM cell

5.2. Comparison to other Methods

Khaidem etc [6] applied Random Forest to predict the stock prices. Ren etc. [7] proposed global refined random forest which could effectively minimize the global training loss of random forest. In this report, we compared the prediction accuracy of stock price using the LSTM model and the other two machine learning methods.

We utilized the MSE to evaluate the prediction results on the same data sets using Random Forest, Refined Random Forest, and LSTM model. As the results shown in Table 2, the LSTM model outperforms the Random Forest and the Refined Random Forest. We believe this is due to that the LSTM is more powerful for retaining a long-term memory and more suitable to predict time series data, such as stock price.

Method / MSE	AAPL	MSFT	AMZN
Random Forest	620.99	312.79	77554.75
Refined RF	509.35	302.10	56021.69
LSTM Model	21.07	4.25	1262.46

Table 2. MSE using Random Forest, Refined Random Forest and LSTM Model

5.3. Portfolios

For more intuitively inspection of the prediction results, an automated portfolio generator with a hedging strategy is used to simulate the real trading process. Based on the prediction price, the portfolio generator will automatically pick the top 5 best performing stock from the stock pool. To reduce the risk of macroeconomic factors, the portfolio is hedged with S&P 500 index. Though using such a hedging strategy will reduce profit in a bull market, the investment risk control is more important, especially in a bear market. As shown in Figure 10, the ROI (return on investment) of the portfolio achieves 5.06% in 62 trading days.

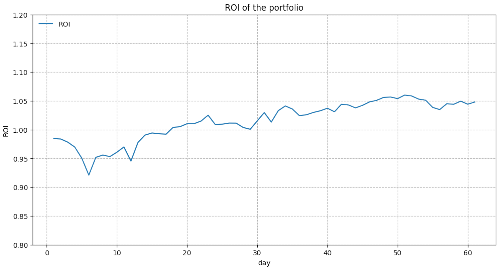


Figure 10. ROI of the portfolio

6. Conclusion and Future Work

In this work, we introduced the LSTM model for stock price prediction. Both the historical statistical data (i.e. stock price, trading volume) and the fundamental data (i.e. P/E, P/B, EPS) are utilized for network training and creating predictions. In the experiments, we find that using more units in LSTM cell makes the network more likely to perfectly memorize the whole the training set and improve the prediction accuracy. Moreover, the LSTM model outperforms the Random Forest and Refined Random Forest, because the LSTM model is more suitable to train and predict the time series data set.

In the future, more features could be applied for network training and stock price prediction. In the meantime, more research can be conducted on how to decide which features are more useful for price prediction.

References

[1] J. Alberg and Z. C. Lipton. Improving factor-based quantitative investing by forecasting company fundamentals. *31st*

Conf. Neural Inf. Process. Syst. NIPS 2017 Long Beach CA USA, Nov 2017.

[2] L. Breiman. Random forests. *Machine Learning, University of California Berkeley,*, 45(1):5–32, Oct 2001.

[3] T. Bylander. Estimating generalization error on two-class datasets using out-of-bag estimates. *Machine Learning, University of California Berkeley,*, 48(1):287–297, Jul 2002.

[4] S. Edet. Recurrent neural networks in forecasting s&p 500 index. *Social Science Research Network, Rochester, NY, SSRN Scholarly Paper*, Jul 2017.

[5] T. Fischer and C. Krauss. Deep learning with long short-term memory networks for financial market predictions. *Eur. J. Oper. Res.*, 20:654–669, Oct 2017.

[6] L. Khaidem, S. Saha, and S. R. Dey. Predicting the direction of stock market prices using random forest. *CoRR*, abs/1605.00003, 2016.

[7] S. Ren, X. Cao, Y. Wei, and J. Sun. Global refinement of random forest. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 723–730, 2015.

[8] O. B. Sezer, A. M. Ozbayoglu, and E. Dogdu. An artificial neural network-based stock trading system using technical analysis and big data framework. *CoRR*, 2017.

[9] L. Wang. Predict stock prices using rnn. `https://lilianweng.github.io/lil-log/2017/07/08/predict-stock-prices-using-RNN-part-1.html`, 2017.