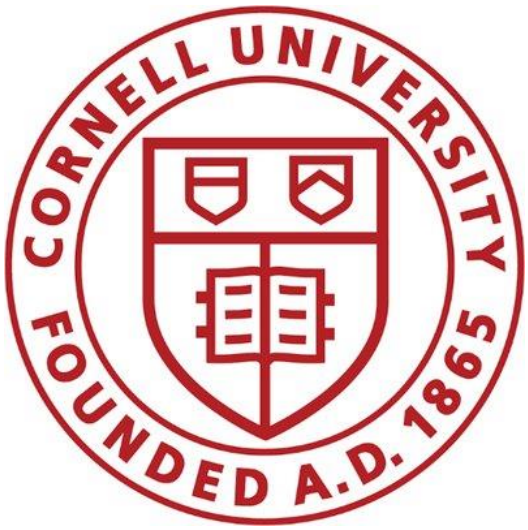

A Study of Deep-learning-based Predictive Model in Financial Engineering

A Design Project Report

Presented to the Engineering Division of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering (Electrical)

Abstract

Master of Electrical Engineering Program

Cornell University Design Project Report

Project Title: A Study of Deep-learning-based Predictive Model in Financial Engineering

Author: Chengjie Lin (cl2445@cornell.edu)

Abstract: The aim of the project is to study the stock price predictive model in terms of different data sources. For this purpose, I started with naïve Long-short Term Memory(LSTM) model with the simple numerical S&P500 dataset, and applied Transformation Under Stability-reTaining Equilibria Characterization(TRUST-TECH) to assist to LSTM with weights tuning process. Based on proof-of-concept work above, the real magic happens with performing sentiment analysis and word to vector process on the textual information crawled from multiple social media sources like twitter, Google News, and so on, then using the processed textual information as the input to the LSTM model, instead of just number S&P500 dataset.

Report Approved by Project Advisor: Hsiao-Dong Chiang

Executive Summary

Long-short term memory (LSTM) is widely used in stock price predicting scenario. The aim of the project is to study the LSTM predictive model in terms of different data sources.

For this purpose, I started with naïve Long-short Term Memory (LSTM) model with the pure numerical S&P500 dataset. I used the Adj Close field of S&P500 as the training set, and shifted this field by one row as the training label. Then I created a naïve LSTM model with only one LSTM layer and one fully connected Dense layer. The result looks good. But this is in fact not what we want. Say today's price is 100, and it may predict tomorrow's price to be 104 or 97, both of which is possible because the model makes predictions only with numerical data and nothing else meaningful in real life. which means there is no difference from the random guessing.

Instead of using only the numerical data like S&P, the use of real word textual information from different social media sources like twitter, Google News and so on, which is meaningful. For this purpose, I used Natural Language Toolkit (NLTK) for sentiment analysis and Global Vectors for Word Representation (GloVe) for word to vector conversion. Then, the processed textual information will be used as the training set of the LSTM model. The result of model turns out to be more accurate than I anticipated, out of 10000, 6243 times it predicts the trends (up or down) correctly, which means a good starting point for future improvements. Finally, the model is improved with Transformation Under Stability-reTaining Equilibria Characterization(TRUST-TECH) to assist to LSTM with weights tuning process.

There are still rooms for this project to be improved like better text processing, the better timeliness and more complicated LSTM model and so on, which means I will continue working on this project third semester.

Content

1.Introduction

Long-short Term Memory is to be used to make predictions in time series financial fields like stock price. The innovation introduced is applying the Transformation Under Stability-retaining Equilibrium Characterization (TRUST-TECH) into the deep learning neural networks training process, whose main features include its capability in identifying multiple local optimal solutions in a deterministic, systematic, and tier-by-tier manner.

After proving the TRUST-TECH on simple LSTM with S&P 500, the focus is moved to the using LSTM with real-time data crawled from different sources like tweets, Google News and so on. Sentiment Analysis and GloVe will be applied to process the news from words to vectors, as the input to our LSTM model. Using the news info from outside as the input to our machine learning model seems to be a reasonable for correct and meaning prediction.

The report consists of 3 parts, (i) Simple LSTM with S&P 500 (ii) TRUST-TECH introduction (iii) Sentiment Analysis With LSTM. The highlight of the project is the third part, the sentiment analysis LSTM.

2. Initial Approach

2.1 Introduction of LSTM

Long-short Term Memory network(LSTM), as a special kind of RNN where the vanishing (exploding) gradients and long-term dependency problem is solve.

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer, while in LSTM, the inner structure is modified in a certain as below to handle the long dependency problem in RNN.[1]

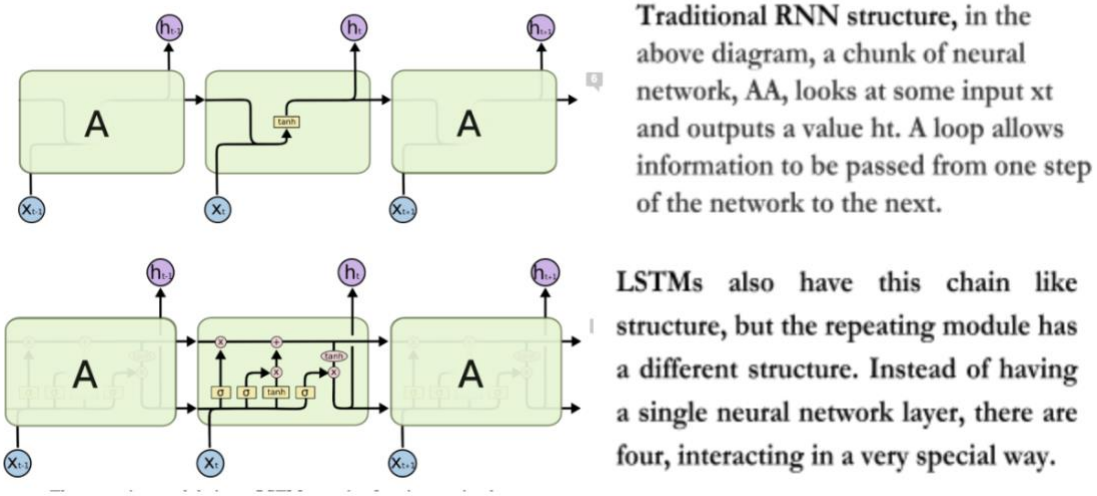

Figure 1: LSTM Introduction [1]

2.2 LSTM with S&P 500 (Point-by-Point Prediction)

The first LSTM I tried is a simple naive LSTM that simply takes the S&P 500 data into the LSTM network.

Here I first stripped out the adj close data out of the S&P500.csv file. I used the Adj Close field of S&P500 as the training set, and shifted this field

by one row as the training label data to the LSTM network

```
Epoch 50/50
2509/2509 [==============================] - 2s - loss: 6.6804 - val_loss: 3.7177
293/293 [==============================] - 0s

The mean squared error (MSE) on the test data set is 0.463 over 293 test samples.
```

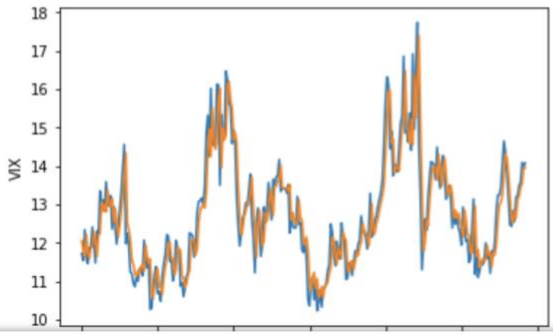

Figure 2: Naïve LSTM (Point-by-Point Prediction)

As can been seen, the point-by-point prediction is kind of accurate. But this is in fact not what we want. Here we predict the future points one by one and even when the model made a mistake on a certain future point T, the next point predicted T+1 will not be far away from the true historical data at T+1 because the previous wrong prediction at T is discarded, instead, the true historical data at T will be used for prediction of T+1. This means that the vanilla version of LSTM itself poses some inevitable design misfit into our problem if left unmodified

2.3 LSTM with S&P 500 (Initiation Window)

First, I used simple LSTM with S&P 500. The window size is set to be 50, and the first window from the testing data is set to be an initiation window. At each time step we then pop the oldest entry out of the rear of the window and append the prediction for the next time step to the front of the window, in essence shifting the window along so it slowly builds itself

with predictions, until the window is full of only predicted values. We then keep this up indefinitely, predicting the next time step on the predictions of the previous future time steps, to hopefully see an emerging trend.

This will also be a many-to-one model since we will need to make predictions about multiple time steps in the future. As a reminder, for a many-to-one LSTM model, the input is shaped into the [samples, timesteps, features] in the Keras context. The idea is that every 50 days correspond to a datapoint, 50 is the number of timesteps in this datapoint. In this way, we can feed the historic stock price into LSTM network for training

We are using a many-to-one LSTM models here and limit our prediction sequence to 50 future time steps and then shifting the initiation window by 50 each time, in effect creating many independent sequence predictions of 50 time steps.

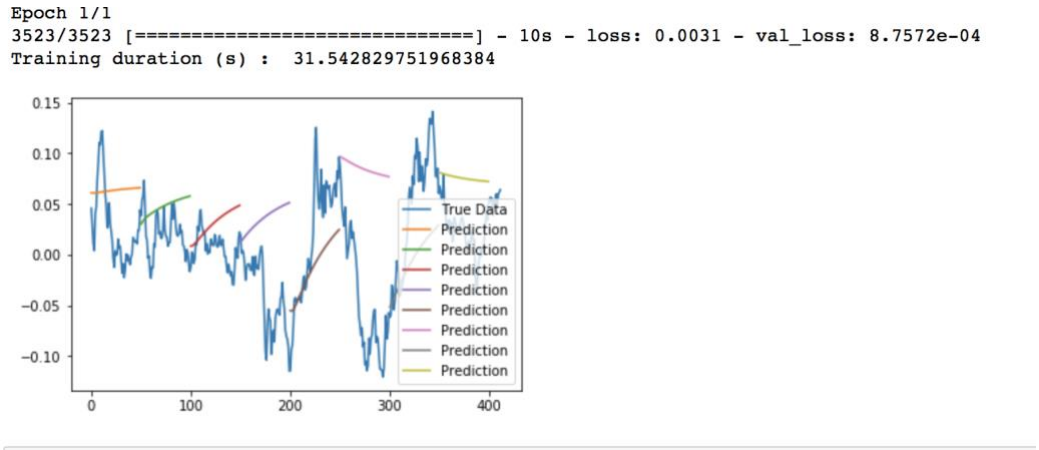

Figure 3: LSTM with S&P500 (Initiation Window)

As can be seen from above figure, the initialization window method may sound reasonable at first, but the actual result is really bad.

2.4 Problem with LSTM Using S&P 500

As can be seen, the result of point-to-point LSTM seems to be good, but the truth is, in real life, simply using the S&P 500 as the input data is meaningless. The reason is because that even if your model is relatively good, the next prediction is still based on the previous one. Say today's price is 100, and a good may predict tomorrow's price to be 104 or 97, both of which is possible because the model can be wrong sometimes no matter how much its previous prediction fits the actual prediction. There is no difference from the random guessing. The conclusion is that, using only numerical data is far from enough.

So, instead of using only the numerical data like S&P, the use of real word textual information from different sources like twitter, Google News and so on, seems promising. In this case, the sentiment analysis will be used here.

3. LSTM with Sentiment Analysis

3.1 Data Crawling

Scrapers are implemented to crawl news from Twitter, Google News and such news websites, in real-time every 10 seconds. We acquire the news in two approaches. The information from Twitter is obtained through its public search API, through which we search for tweets related to bitcoins. For the other news sources, we rely on the services provided by

News API, a live news retrieving provider. The news will be stored in local MongoDB of the server with timestamps and other metadata of their posting. As a news is scraped and stored in our database, we multicast the news to the edge server in order to make sure the availability of the latest news to frontend and also reduce the amount of text our machine learning model needs to handle. We will need to utilize the timestamps of news and bitcoin prices in order to build input vectors and labels for our machine learning model.

3.2 Input Processing

Input: Normalized sentiment value calculated from news posted between timestamp t and t-10s.

For this project, we will use NLTK to get the sentiments of the aggregated news. NTLK use a Sentiment Analyzer, which is a tool to implement and facilitate Sentiment Analysis tasks using NLTK features and classifiers, especially for teaching and demonstrative purposes.

For each price data with timestamp t, the news headings posted between t and after t-10 seconds will be aggregated together to compute the sentiments value of these news in between [t, t-10] using NLTK.

	title	time	text	weight	source
0	twitter_post	1523133939	I tested a new utility token out there - Synap...	0	twitter
1	twitter_post	1523133938	RT @SDWouters: 10 months ago #Bitcoin had 7k f...	28	twitter
2	twitter_post	1523133937	bitcoin cash 32 Megabyte Block Size Increase a...	0	twitter
3	twitter_post	1523133937	What is blockchain - the technology behind Bit...	0	twitter
4	twitter_post	1523133936	RT @Viking7070: Local governments are starting...	3	twitter

Figure 4: Original News Feed

```
[[ 0.      0.     -0.1531  0.      0.      0.3818  0.4019  0.6369 -0.5719
   0.      0.6832  0.      0.      0.      0.743   0.802   0.     -0.6486
   0.      0.7003  0.2732  0.296   0.8126  0.4404  0.7579  0.      0.
0.
  -0.2732  0.768 ]
 [ 0.      0.     -0.1531  0.      0.      0.3818  0.4019  0.6369 -0.5719
   0.      0.6832  0.      0.      0.      0.743   0.802   0.     -0.6486
   0.      0.7003  0.2732  0.296   0.8126  0.4404  0.7579  0.      0.
0.
```

Figure 5: Word to Vector After Sentiment Analysis

3.3 Output Processing

Output: Price difference between timestamp t and t-10s.

	price	timestamp
0	6999.8	1523132235
1	6984.1	1523133211
2	6984.1	1523133241
3	6984.1	1523133272
4	6984.1	1523133303

Step 1

	price	timestamp
0	NaN	1523132235
1	-15.7	1523133211
2	0.0	1523133241
3	0.0	1523133272
4	0.0	1523133303

Step 2

	price	timestamp
1	-15.7	1523133211
2	0.0	1523133241
3	0.0	1523133272
4	0.0	1523133303
5	0.0	1523133334

Step3

Figure 6: Steps of price processing

For the price data, the original will be processed so that the original price will be used to calculate the price difference between the consecutive timestamp, as shown in the Step 3.

3.4 Machine Learning Model

Long-short Term Memory is used here time-series data predictions and Many-to-many LSTM model is used for the prediction here. We rearrange the input data so that every datapoint is a (sample, timesteps, features) 3dimension format for both input and output, where the timesteps is set to be 8 here, meaning that it will use the last 80 timesteps to predicts the next 8 timesteps.

Every time the edge server models make a prediction on prices of incoming 8 timesteps, it will record the starting time and ending time of the process and remove predictions in this time cost from the predictions since they are meaningless now, which will eventually lead to valid predictions on coming 8-(start_time-end_time)/10 timesteps, which translated into 1 minute in the future, by the time user actually sees the predictions to assure the time-effectiveness of the model. The predicted results will be stored into the MongoDB. Every time the user clicks the prediction button, it will fetch the pre-stored predicted results from MongoDB and return it to the

user.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 5, 32)             4352
_________________________________________________________________
time_distributed_1 (TimeDist (None, 5, 1)              33
=================================================================
Total params: 4,385
Trainable params: 4,385
Non-trainable params: 0
_________________________________________________________________
```

Figure 7: Summary of LSTM Model

3.5 Result

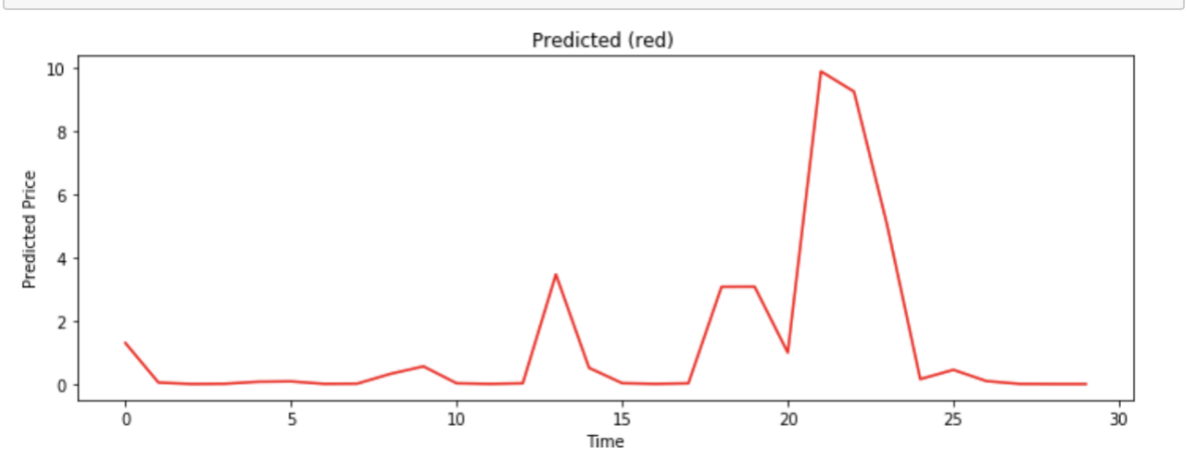

Figure 8: Result of LSTM with Sentiment Analysis

The result will reflect the price floating in next coming 30 second, where the transaction will be made based on the machine learning model.

Every half an hour the model on central server re-trains itself based on recently crawled data. The old model will be saved as .json file and .h5

file and be used in the re-train process, every time it re-trains, it will load the old model and keeps on training.

The result of model turns out to be more accurate than I anticipated, out of 10000, 6243 times it predicts the trends (up or down) correctly, which means it is a good starting point for future improvements.

4. TRUST-TECH Tuning

4.1 Why TRUST-TECH

In terms of nonlinear system optimization problems, one of the most difficult thing is to find global optimized solution. There are lots of local method to solve for local solution. But once we locate one of the local solution, it would be really hard for us to move to another local solution. There do exist some global solver methods, but all of them have really complex and miscellaneous computation steps. And there is not a systematical way on solving generic problem when using those methods. And locating global optimal solutions is almost intractable. TRUST-TECH can help us with this problem. The TRUST-TECH method with neural network details can be found at [2]

4.2 TRUST-TECH in LSTM Weight Tuning

The goal here is assist the LSTM with weight tuning using TRUST-TECH.

First Form the gradient like system $-\nabla f(x,y)$. Then choose random initial point and find the corresponding local optimal solution.

Move along Eigen vector and directions we choose, until objective function's value suddenly decrease, then we find the exit point. If after move a long time, we still cannot find one exit point, we assume in that direction, the stability boundary is infinity.

Once find exit point, move a little bit forward, then we use this as the starting point for the LSTM model and train the LSTM model again.

After search through all the directions, check new solution and repeat again for better solutions. In this way, the TRUST-TECH can help the LSTM model exploit its best potential with best weights.

5. Conclusion

LSTM is rather reasonable model used in stock price prediction. However, the main problem lies in the source of the data set. As the first two naïve LSTM model shows, the pure numerical S&P 500 data is far from sufficient and poses no solid meaning in terms of real life decision making. That is why the LSTM should be used with real time news and social media data like tweets from twitter. However, the model itself is universal and we can definitely use naïve LSTM as the testbed for such purpose. Apart from that, method like TRUST-TECH can also help the training process.

6. Future Work

For the next semester, the TRUST-TECH will be used in the sentiment analysis machine learning model. An online version of predictive model will be hosted on servers from AWS Beanstalk and TRUST-TECH algorithm will go online with the model.

Reference

[1] Christopher Olah. Understanding LSTM Networks. August 27, 2015. Retrieved from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[2] Wang, B. D., Hsiao-Dong Chiang. 2011."ELITE: Ensemble of Optimal, Input-Pruned Neural Networks Using TRUST-TECH."IEEE Transactions on Neural Networks22(1): 96-109.