STOCK MARKET FORECASTING USING RECURRENT NEURAL NETWORK

A Thesis Presented to

the Faculty of Graduate School

at the University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Qiyuan Gao

Dr. Zhihai He, Thesis Supervisor

MAY 2016

The undersigned, appointed by the Dean of the Graduate School, have examined the thesis entitled:

STOCK MARKET FORECASTING USING RECURRENT NEURAL NETWORK

Presented by: Qiyuan Gao

A candidate for the degree of Master of Science and hereby certify that, in their opinion, it is worthy of acceptance.

Professor Zhihai He

Professor Dominic Ho

Professor Jianlin Cheng

ACKNOWLEDGEMENTS

My deepest gratitude goes to my advisor, Dr. Zhihai He. His patience and support helped me overcome a lot of difficulties. Through the rough road to finish this thesis, Dr. He's advice is like a lighthouse that always guides me to the correct direction. He taught me how to face problems with open mind and solve problems with careful thoughts.

I am grateful to Dr. Ho and Dr. Cheng. My master thesis could never be done without the knowledge they taught me. And it is a truly honor to have them serve on my committee.

I would like to thank all my lab mates: Chen Huang, Zhiqun Zhao, Yan Tang, Jianhe Yuan, Xiaolong Ke, Wenqiang Bo, Yibin Huang, Zhang Zhi and Haider Yousif. They make our lab always a great place to research and study.

Most importantly, I am deeply grateful to my parents for everything they gave me. They are the most important part in my life. Without their love and concern, none of this would have been possible.

TABLE OF CONTENTS

LIST OF ILLUSTRATION

Figure Page

LIST OF TABLES

Table Page

Abstract

In this research, we study the problem of stock market forecasting using Recurrent Neural Network(RNN) with Long Short-Term Memory (LSTM). The purpose of this research is to examine the feasibility and performance of LSTM in stock market forecasting. We optimize the LSTM model by testing different configurations, i.e., the number of neurons in hidden layers and number of samples in sequence. Instead of using daily stock price data, we collect hourly stock data from the IQFEED database in order to train our model with relatively low noise samples. Nevertheless, based on the prediction results of LSTM model, we build up a stock database with six U.S market stocks from five different industries. The average test accuracy of these six stocks is 54.83%, where the highest accuracy is at 59.5% while the lowest is at 49.75%. We then develop a trade simulator to evaluate the performance of our model by investing the portfolio within a period of 400 hours, the total profit gained by the model is $413,233.33 with $6,000,000 initial investment.

Chapter 1

Introduction

Modeling and Forecasting of the financial market has been an attractive topic to scholars and researchers from various academic fields. Financial market is an abstract concept where financial commodities such as stocks, bonds and precious metals transactions happen between buyers and sellers. In the present scenario of financial market world, especially in the stock market, forecasting the trend or the price of stocks using machine learning techniques and artificial neural networks are the most attractive issue to be investigated. As Giles *et. al* [1] explained, Financial forecasting is an instance of signal processing problem which is difficult because of high noise, small sample size, non-stationarity, and non-linearity. The noisy characteristics means the incomplete information gap between past stock trading price and volume with future price. Stock market is sensitive with political and macroeconomic environment. However, these two kinds of information are too complex and unstable to gather. The above information that cannot be included in features are considered as noise. The sample size of financial data is determined by real world transaction records. On one hand, a larger sample size refers a longer period of transaction records; on the other hand, large sample size increases the uncertainty of financial environment during the

sample period. In this research, we use hourly stock data instead of daily data in order to reduce the probability of uncertain noise, and relatively increase the sample size within a certain period of time. By non-stationarity, one means that the distribution of stock data is various during time changing. Non-linearity implies that feature correlation of different individual stocks is various [2].

Efficient Market Hypothesis was developed by Burton G. Malkiel in 1991 [3]. In Burton's hypothesis, he indicates that predicting or forecasting the financial market is unrealistic, because price changes in real world are unpredictable. All the changes in prices of financial market are based on immediate economic events or news. Investors are profit oriented, their buying or selling decisions are made according to most recent events regardless past analysis or plans. The argument about this Efficient Market Hypothesis has never been ended. So far, there is no strong proof that can verify if the efficient market hypothesis is proper or not. However, as Yaser claims [4], financial markets are predictable to a certain extent. The past experience of many price changes over a certain period of time in financial market and the undiscounted serial correlations among vital economic events affecting the future financial market, are two main evidences opposing the Efficient Market Hypothesis.

In recent years, machine learning methods have been extensively researched for their potentials in forecasting and prediction of financial market. Multi-layer feed forward neural networks, SVM, reinforcement learning, relevance vector machines, and recurrent neural networks are the hottest topics of many approaches in financial market prediction field [2, 5]. Among all the machine learning methods, neural

networks are well studied and has been successfully used for forecasting and modeling financial market. “Unlike traditional machine learning models, the network learns from the examples by constructing an input-output mapping for the problem at hand. Such an approach brings to mind the study of nonparametric statistical inference; the term “nonparametric” is used here to signify the fact that no prior assumptions are made on a statistical model for the input data”, according to Simon [5]. As Francis E.H. Tay and Lijuan Cao explained in their studies [2], Neural networks are more noise tolerant and more flexible compared with traditional statistical models. By noise tolerance, one means neural networks have the ability to be trained by incomplete and overlapped data. Flexibility refers to that neural networks have the capability to learn dynamic systems through a retraining process using new data patterns [2].

Long short-term memory is a recurrent neural network introduced by Sepp Hochreite and Jurgen Schmidhuber in 1997 [6]. LSTM is designed to forecast, predict and classify time series data even long time lags between vital events happened before. LSTMs have been applied to solve various of problems; among those, handwriting Recognition [7] and speech recognition [8] made LSTM famous. LSTM has copious advantages compared with traditional back-propagation neural networks and normal recurrent neural networks. The constant error backpropagation inside memory blocks enables in LSTM ability to overcome long time lags in case of problems similar to those discussed above; LSTM can handle noise, distributed representations and continuous values; LSTM requires no need for parameter fine tuning, it works well over a broad range of parameters such as learning rate, input gate bias and output gate bias [6].

The objective of our research can be generalized into two main parts. First, we examine the feasibility of LSTM in stock market forecasting by testing the model with various configurations. Second, according to the prediction results generated by LSTM model, we build up stock trading simulator to validate the feasibility of our model in real world stock trading activities.

Chapter 2

Related Work

1. Financial Market Prediction

In the past few decades, financial market prediction has become a new hot research topic in the machine learning field. With the aid of powerful models such as SVM (Support Vector Machine), feed forward neural network and recurrent neural network, researchers overcame numerous difficulties and achieved considerable progress. Their research results inspired our thesis. In this section, we will go through the representative ones among these studies.

1.1 Introduction to Financial Forecasting

In 1996, Yaser S. Abu-Mostafa [4] provided a brief introduction to forecasting in financial markets with emphasis on commodity futures and foreign exchange. He explained the basic of forecasting as well as the noisy nature of financial data. The methods of choosing inputs, outputs and error functions.

Financial market data is very noisy. According to Yaser [4], “consider the market as a system that takes in a lot of information (fundamentals, news events, rumors, who bought what when, etc.) and produces an output $\hat{y}$. A model e.g., a neural network,

attempts to simulate the market, but it takes an input x which is only a small subset of the information."

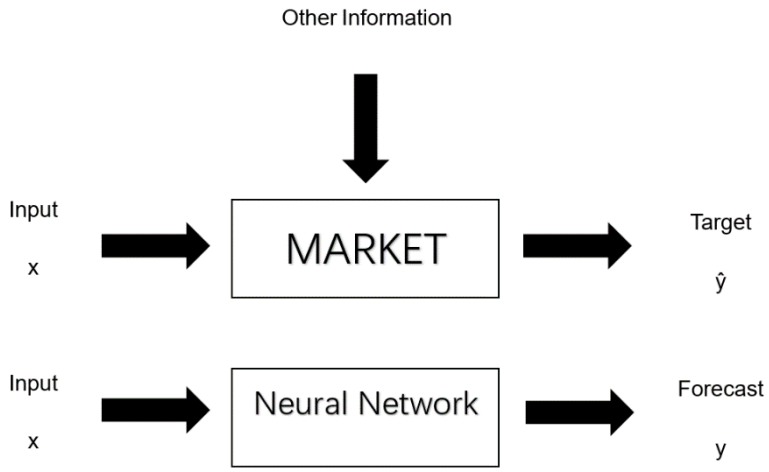

Figure 1: Illustration of the nature of noise in financial markets

Yaser suggested one possible way to minimize the noise [4] by using a very large training data set. However, because of the non-stationarity of financial data, the old data may represent a very different patterns than the new data. To overcome this difficulty, specifically, increasing the size of training data set, but limit the time length within a relatively short period of time, we use hourly stock price in our research. Thus, we gain six more sample per unit time length (one day).

1.2 Financial Market Prediction using Recurrent Neural Network

Nijole, Aleksandras and Algirdas explored effects of different epochs and number of neurons on time series prediction results. In their studies [9], they aim to investigate the best parameters setting (such as number of epochs, neurons) to achieve a good prediction result of financial market.

Table 1: Dependence of test RMSE of learning from the number of epochs with data of USD/JPY and Gold

Number of epochs	RMSE of learned RNN
16	-0.008298
32	-0.007691
64	-0.007232
72	-0.006012
76	-0.002237
80	-0.002664
120	-0.002270
164	-0.001667
172	-0.001652
188	-0.001523
200	-0.001824
220	-0.001671

Table 1 indicates the RMSEs (root mean square error) under different number of epochs setting of the recurrent neural network. The obtained results of learning RMSE are taken in Table1. Results shows that a small number of epochs does not provide RNN learning. RMSE dependence on epochs shows that, after 164 epochs, the learning process becomes stable and further increasing of the epochs does not benefit any more [9].

As claimed by the authors [9], the number of neurons is a vital parameter setting in recurrent neural network learning process. Even though structure of LSTM is much complex than traditional recurrent neural networks, but it is a sub-architecture of RNN and the number of neurons principle works for it. A larger amount of neurons can generate better prediction results; however, it takes a much longer training time as well. As inspired by Nijole, Aleksandras and Algirdas's research, we performed several experiments with different numbers of neurons settings in order to find out the best configuration.

1.3 Efficient Machine Learning Techniques for Stock Market Prediction

Financial market prediction is always an important task for investors. Recently, machine learning methods take a vital role on this task. Various machine learning techniques have been carried out; "Time Series, Neural Network and its different variation (RNN, ESN, MLP, LRNN etc.) and different hybrid techniques (combination of neural network with different machine learning techniques) (ANFIS, GA/ATNN, GA/TDNN, ICA-BPN)." [10] Among this numerous of methods, neural networks generate the best results unless prediction of stock price is necessary. The experiment results of three neural network structures were compared by Zahid Iqbal *et. al* [10], "By comparing said techniques, it was observed that LRNN performs better than feed forward NN and Wsmpca-NN performs better than LRNN and NN." In their conclusion of the comparison, the authors claim that different techniques or neural network structure setting are variable, but data pre-processing and post-processing brought great effects on prediction results [10]. In our thesis, we trained our model with post-processing stock data. Instead of using raw stock data, we processed our data with several statistical methods, it will be discussed in Chapter 4.

2. Neural Networks in Business Analysis

The analysis and studies of business related problems have been dominated by traditional statistical models for decades. Recently, artificial neural networks have established their status. As a consequence, numerous of applications using artificial

neural networks to business fields have been researched, studied and produced. In this section, we review the representative applications in business related problems and its inspiration bought to our research.

2.1 Introduction to Neural Network Applications in Business Fields

Business fields have a much larger sub-category in definition than financial market fields such as: accounting, bankruptcy prediction, credit, insurance, marketing, management, etc. A large body of neural network applications have been produced to these categories. The reason why neural networks have been widely used in business related problems is because, first, neural networks have great suitability of handling incomplete, missing or noisy data; second, neural networks do not require priori assumptions about the distribution of the data, third they are capable to map complex and approximate continuous function [11]. In Vellido, Lisboa and vanghan's survey [11], they generated a detailed table of advantages and disadvantages of the application using neural networks in business related problems.

Table 2: Table of the most frequently quoted advantages and disadvantages of the application of neural networks

Advantages of neural nets	Total	Disadvantages of neural nets	Total
NNs are able to learn any complex non-linear mapping / approximate any continuous function.	31	NNs lack theoretical background concerning explanatory capabilities /NNs as "black boxes"	28
As non-parametric methods, NNs do not make *a priori* assumptions about the distribution of the data /input–output mapping function.	30	The selection of the Network topology and its parameters lacks theoretical background /It is still a "trial and error" matter.	21
NNs are very flexible with respect to incomplete, missing and noisy data /NNs are "fault tolerant"	29	Neural networks learning process can be very time-consuming	11
Neural network models can be easily updated /are suitable for dynamic environments.	15	Neural networks can overfit the training data, becoming useless in terms of generalisation.	10
NNs overcome some limitations of other statistical methods, while generalizing them.	15	There is no explicit set of rules to select a suitable NN paradigm / learning algorithm.	8
Hidden nodes, in feed-forward supervised NN models can be regarded as latent /unobservable variables.	5	NNs are too dependant on the quality /amount of data available.	6
NNs can be implemented in parallel hardware, increasing their accuracy and learning speed.	4	NNs can get stuck in local minima /narrow valleys during the training process.	5
Neural networks performance can be highly automated, minimizing human involvement.	3	NN techniques are still rapidly evolving and they are not reliable /robust enough yet.	3
Nns are specially suited to tackle problems in non-conservative domains.	3	NNs lack classical statistical properties. Confidence intervals and hypothesis testing are not available.	2

2.2 Bankruptcy Prediction for Credit Risk Using Neural Networks

In a banking system, the prediction of corporate bankruptcies is vital and widely researched problem, because successful prediction or not could directly impact on banks' decisions and profitability [12]. The traditional way for banks to perform credit risk management is using internal rating system. Several factors are accounted into this system, such as historical earnings, operating cash flow, interest coverage, and leverage

[13]. The main problem with this approach is subjective aspect of the predication, which may result in a same company gets various credit score across different banks [12].

In Amir's study, a novel set of indicators were introduced, which are extracted from the firm's stock price data [12]. By indicators, one means the inputs or features of the prediction model. In order to experiment the new novel set of indicators, Amir developed to systems: one without the stock price data, one without the stock price data. The experiment shows that the system with new indicators outperformed the old one with 89.41% on training set and 85.50% on test set; compared with 84.52% on training set and 81.46% on test set [12].

In Amir's research, the combination of indicators brought amazing effects on perdition results. Inspired by Amir's paper, we apply various features that related to the target stock as input to generated the best prediction results.

2.3 Neural Networks for Solving Prediction Problems in Business Management

Neural networks have capability new objects previously untrained by training with known examples. This feature makes neural networks can predict new outcomes from past trends [14]. Statistical or classical mathematical methods based on traditional logic are hard to modeled business management problems, since this kind of problems are very complex, uncertain, dynamic and mutable [15]. Business management problems include prediction of market demand, product prices, project cost flow, product quality, etc [15].

In Kosa's paper [15], three major steps were summarized to build, train and test the forecasting model:

- Step 1: Choosing inputs and outputs. In this step, only inputs that influence the output in order should be chose in case of overloading the network;
- Step 2: Collecting training data set. Each training data sample consists a pair of vectors, $(X^k,\ Y^k)$, $\mathrm{k} = 1,2,\dots,\mathrm{q}$, where q is the number of training samples, X is input vector, Y is output vector. Required number of train samples most depends on the complexity of interrelationships between inputs and outputs. In general, the larger the training samples, the better the results;
- Step 3: Training and testing. Before the training process, one needs to design and set up the neural networks. The number of nodes on input and output layers equal to the dimensions of input and output data. However, the number of nodes on hidden layers generally depends on the complexity of input and output interrelationships. The training process is done and the network is ready to be tested, once the error is within an assigned interval.

In Kosa's model [15], the target is to forecast the apartment price estimation in a city. The inputs are apartment area, city zone, number of bedrooms, interior apartment quality and building age. With a threshold of 5%, test accuracy is 85%. Kosa has described a procedure that using neural network solving prediction or forecasting problems. Though we use recurrent neural network model in this thesis, the general procedures of preparing data, building networks and training/testing network are coherent.

3. Related Work on LSTM

LSTM has been proved to be efficient on dealing with various problems such as speech and handwriting recognition since it has capability to learn from inputs even there are long time lags between important events.

Recurrent neural networks are considerably suitable for processing sequential problems [16], and recurrent neural networks outperformed both Hidden Markov Models (HMMs) and discrete symbolic grammar learning algorithms(SGLAs) [17] [18]. Differ from RNNs, HMMs ignore important events if temporal delays occur. Even this problem can also be solved by introducing separate internal state for each possible delay, the model would become complex and inefficient [19]. Moreover, unlike RNNs, SGLAs are not capable with noisy inputs [20]. RNNs, instead, perform gradient descent in potentially noise-resistant algorithms, using distributed internal memories mapping input sequences to output sequences [19]. Traditional RNNs have one major inherent disadvantage that the gradient of output error is based on previous inputs vanishes when time lags between inputs and errors increases [19]. However, LSTMs can overcome this problem since it can train to bridge time lags more than 1000 discrete time steps [6].

3.1 LSTM Learn Simple Context Free and Context Sensitive Language

In Flex and Jurgen paper “LSTM Recurrent Networks Learn Simple Context Free and Context Sensitive Language” [19], the LSTM network is given one input symbol

at a time to predict next possible symbols. If the next possible symbols contain more than one elements, all symbols are expected to be predicted. According to Flex and Jurgen [19], “Symbols are encoded locally by d-dimensional binary vectors with only one non-zero components, where d equals the number of language symbols plus one for either the start symbol in the input or the “end of string” symbol in the output.” Three sets of samples are trained and tested,

CFL $a^n b^n$:

Input:	S	a	a	a	a	a	b	b	b	b	b
Target:	a/T	a/b	a/b	a/b	a/b	a/b	b	b	b	b	T

Figure 2: CFL $a^n b^n$ Dataset Illustration

CFL $a^n b^m B^m A^n$:

Input:	S	a	a	a	a	b	b	b	B	B	B	A	A	A	A
Target:	a/T	a/b	a/b	a/b	a/b	b/B	b/B	b/B	B	B	A	A	A	A	T

Figure 3: CFL $a^n b^m B^m A^n$ Dataset Illustration

CSL $a^n b^n c^n$:

Input:	S	a	a	a	a	a	b	b	b	b	b	c	c	c	c	c
Target:	a/T	a/b	a/b	a/b	a/b	a/b	b	b	b	b	c	c	c	c	c	T

Figure 4: CSL $a^n b^n c^n$ Dataset Illustration

A three-layer LSTM model with single input and output is used in this paper [19]. The model has a single LSTM memory block with a single cell, only one hidden layer is used.

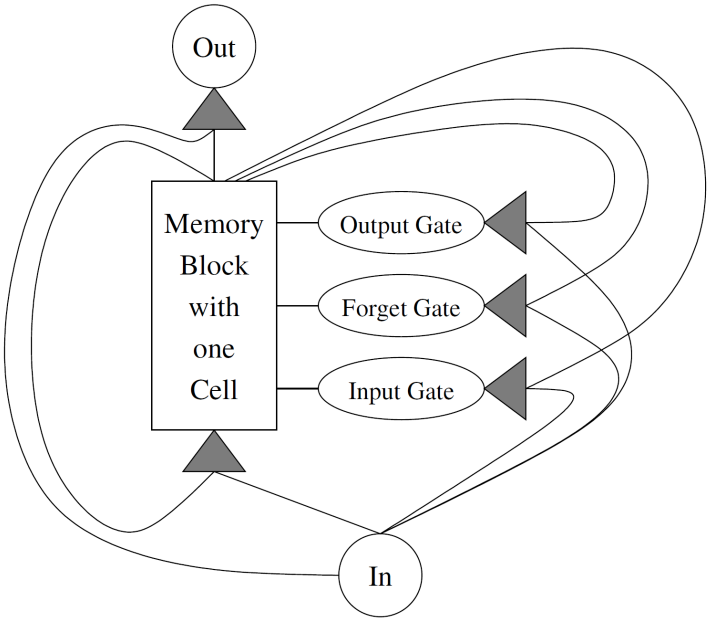

Figure 5: Three-Layer LSTM

LSTM model outperformed traditional RNNs models based on testing of the three samples [19], for all CFL $a^n b^n$, CFL $a^n b^m B^m A^n$ and CSL $a^n b^n c^n$ samples, LSTM 100% solved for all training sets. The LSTM model can even generate n=1000 sets when training sets has 10 samples for CFL $a^n b^n$.

3.2 Action Classification Using LSTM

Moez, Franck, Christian, Christophe and Atilla proposed soccer action classification using LSTM without priori information [21]. The outline of their model is shown below. Inputs are a sequence of descriptors with a set of features, each sample represents one image. The input sequences forward to LSTM network, the LSTM generates action type based on given inputs [21].

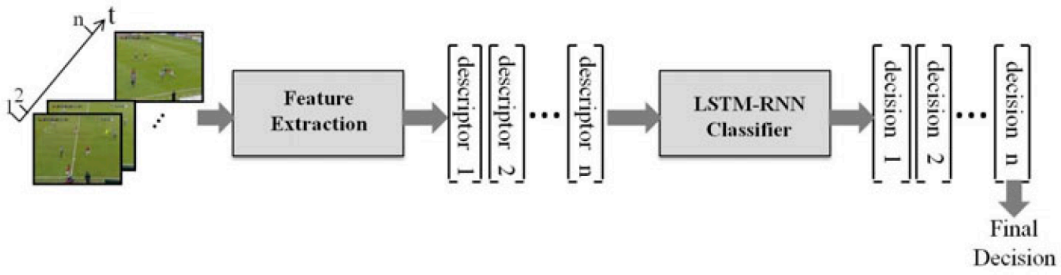

Figure 6: an action classification model

In their research [21], various configurations of network settings have been tested and they figure that a large number of LSTM memory blocks results in overfitting, while a small number of memory blocks leads divergence. 150 LSTM neuros was found to be the best configuration for the task.

	Classification rate
BoW + k-NN	52,75 %
BoW + SVM	73,25 %
BoW + LSTM-RNN	**76 %**
Dominant motion + LSTM-RNN	**77 %**
BoW + dominant motion + LSTM-RNN	**92 %**

Figure 7: action classification results

The results show that the LSTM outperforms both KNN and SVM models. With different input feature configurations, results are different. The model achieved best accuracy of 92% with both BoW and dominant motion applied. In our research, we also test our model under various network configurations in order to avoid overfitting or divergence, but generate the best results.

Chapter 3

Methods

1. Overview

The outline of proposed model is shown in the below. Raw stock data only contains open price, highest price, lowest price, close price and transaction volume of a specific time point, which are insufficient to train LSTM model. Before forwarding inputs to the network, we pre-processed the raw stock data into 369 features with competitor stocks and index information included. Then, the processed samples are forwarded into the LSTM model sequentially. The number of samples in sequence is the maximum amount of backpropagation steps to take back through time. Details on pre-processing will be discussed in Chapter 5.

Figure 8: proposed classification outline in this thesis

The aim is to predict target stock performance in next three hours based on information given in current hour. The future performance is evaluated according to increasing percentage of next three hours' highest price compare with current hour open

price. And the performance can be categorized into three classes: increasing between 0%~1%, increasing above 1%, and not increasing (less than 0%).

Recurrent neural network (RNN) with Long Short-Term Memory model is trained to classify the stock future performance into three classes. Details of out LSTM-RNN model is shown below.

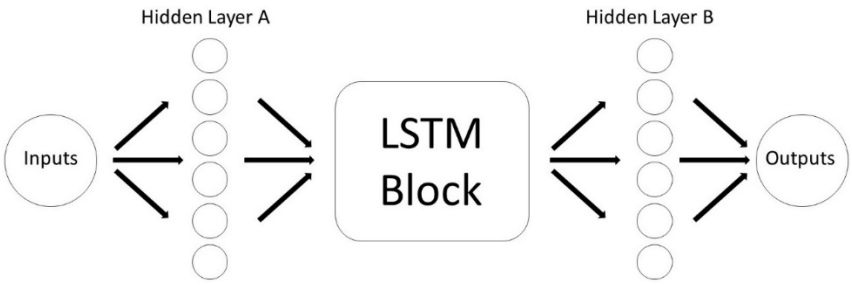

Figure 9: LSTM-RNN model structure

We apply one LSTM block in our RNN model; one hidden layer after the input layer and one hidden layer before the output layer. We also test various configurations of number of neurons in these two hidden layers to generate the best performance setting. Details about how different configuration effects experiment results will be covered in Chapter 5.

2. LSTM

The initial version of LSTM block was introduced by Sepp and Jurgen in 1997 [6]. Structure of the LSTM is shown below.

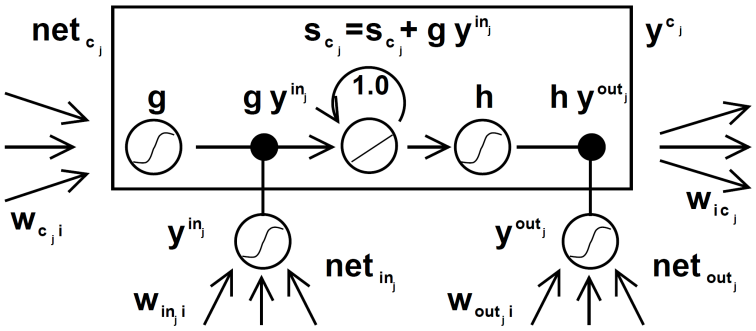

Figure 10: architecture of initial LSTM model introduced in 1997

Memory cells and gate units were introduced to RNN network by Sepp and Jurgen. Constant Error Carrousel (CEC) is the key feature of LSTM. This unit enables constant error forward through unique, self-connected units without the disadvantages of the naïve approach [6]. Input gate can control whenever to keep or overwrite information in memory cell and output gate can decide whenever to access memory in memory cell or not [6].

In this research, we use LSTM with forget gates introduced by Felix, Jurgen and Fred [22]. Structure details of the extended LSTM structure is shown below.

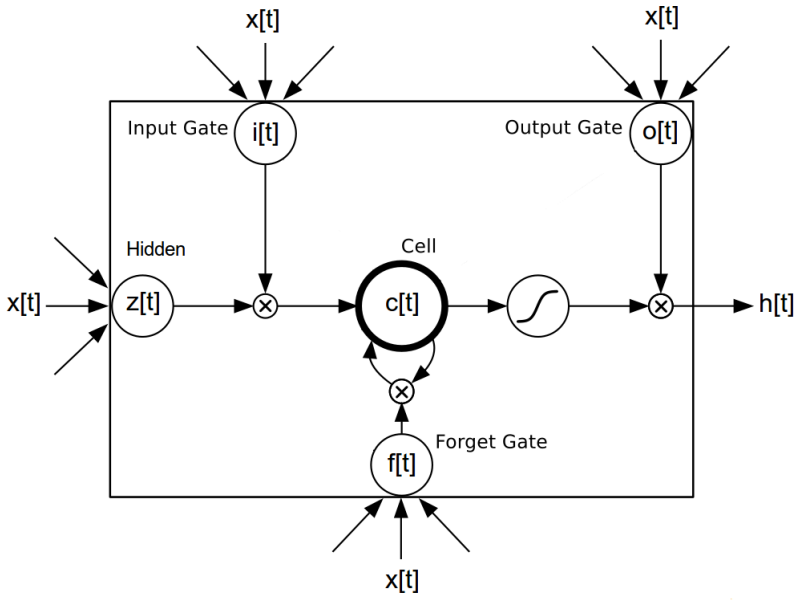

Figure 11: structure of LSTM with forget gate

Comparing to initial LSTM, LSTM with forget gate has one additional structure: Forget Gate. Forget Gate enables LSTM to reset its own state [23]. The actual implementation corresponds to the following algorithm adapted from [23]:

$$\textit{Block input unit}: \ z^t = g(W_z x^t + R_z y^{t-1} + b_z);$$

Where g is logistic sigmoid, W is weight matrix from input to z; x is current step input; R is weight matrix from output of previous step to z and b is bias vector.

$$\textit{Input gate unit}: \quad i^t = \sigma(W_i x^t + R_i y^{t-1} + b_i);$$

Where σ is logistic sigmoid, W is weight matrix from input to i; x is current step input; R is weight matrix from output of previous step to i; and b is bias vector.

$$\textit{Forget gate unit}: \quad f^t = \sigma(W_f x^t + R_f y^{t-1} + b_f);$$

Where σ is logistic sigmoid, W is weight matrix from input to f; x is current step input; R is weight matrix from output of previous step to f; and b is bias vector.

$$\textit{Cell state unit}: \quad c^t = i^t \odot z^t + f^t \odot c^{t-1};$$

Where i is input gate parameters; ⊙ is point-wise multiplication of two vectors; f is forget gate parameters and c^{t-1} is cell state from previous step.

$$\textit{Output gate unit}: \quad o^t = \sigma(W_o x^t + R_o y^{t-1} + b_o);$$

Where σ is logistic sigmoid, W is weight matrix from input to o; x is current step input; R is weight matrix from output of previous step to o; and b is bias vector.

$$\textit{Block output unit}: \quad y^t = o^t \odot h(c^t);$$

Where o is output gate parameter; ⊙ is point-wise multiplication of two vectors; h is logistic sigmoid and c is cell state parameter.

3. Forward Pass

We pass pre-processed samples through LSTM model in sequence.

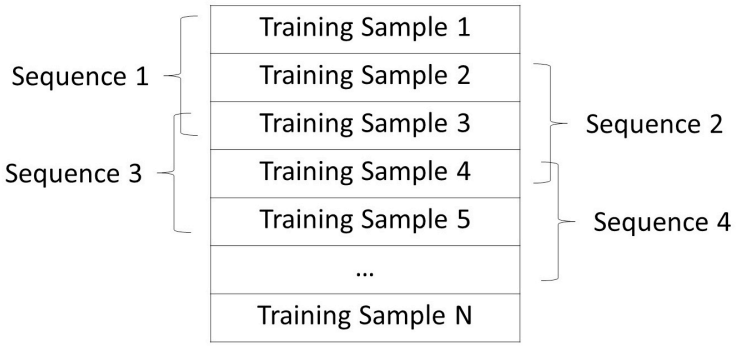

Figure 12: illustration of input sequence

The sequence size is the maximum number of steps taking back through time during backpropagation; nevertheless, the size decides the maximum amount of previous steps

stored in LSTM block memory. Figure 12 shows the illustration of input sequence with size of three. Different input sequence size brings vital influence on test results, we will discuss it in details in Chapter 5. Referring to Klaus *et. al* [23], weights of LSTM can be divided into three categories:

- Input weights: $W_z, W_i, W_f, W_o \in R^{NxM}$
- Recurrent weights: $R_z, R_i, R_f, R_o \in R^{NxN}$
- Bias weights: $b_z, b_i, b_f, b_o \in R^N$

Where z is block input, i is input gate, f is forget gate, o is output gate, N is number of LSTM block (1 in our case), and M is number of inputs. To help understand of forward pass process, formulas of each function unit except cell and block output can be split into two stage: before activity and after activity [23]:

- *Block input unit before activity*: $\bar{z}^t = g(W_z x^t + R_z y^{t-1} + b_z)$;
- *Block input unit after activity*: $z^t = g(\bar{z})$;
- *Input gate unit before activity*: $\bar{\imath}^t = W_i x^t + R_i y^{t-1} + b_i$;
- *Input gate unit after activity*: $i^t = \sigma(\bar{\imath})$;
- *Forget gate unit before activity*: $\bar{f}^t = W_f x^t + R_f y^{t-1} + b_f$;
- *Forget gate unit after activity*: $f^t = \sigma(\bar{f})$;
- *Cell state unit*: $c^t = i^t \odot z^t + f^t \odot c^{t-1}$;
- *Output gate unit before activity*: $\bar{o}^t = W_o x^t + R_o y^{t-1} + b_o$;
- *Output gate unit*: $o^t = \sigma(\bar{o}^t)$;
- *Block output unit*: $y^t = o^t \odot h(c^t)$;

According to the formulas above, in forward passing process, system unit's parameters, except block output unit, depend on memories of previous time stage, as shown in Figure 13:

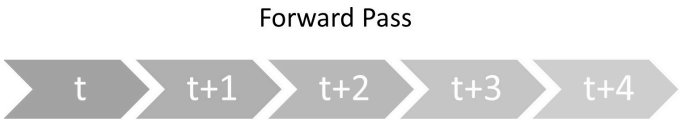

Figure 13: forward pass

4. Backpropagation Through Time

Backpropagation is the most popular and widely used algorithm in artificial neural network field. Yet Backpropagation Through Time, known as BPTT, is a gradient-based training techniques for recurrent neural networks. BPTT is a widely used tool, and it has numerous applications such as, pattern recognition, dynamic modeling, sensitivity analysis, and the control of system over time [24]. In certain applications such as, speech recognition, handwriting recognition or predictions as in our research; classification or prediction results at time t can be more accurate if one can account for what has been passed through the system earlier times [24]. BPTT is the most efficient and widely used method to train LSTM. According to Klaus *et. al* [23], BPTT of LSTM can be divided into two stages, First stage, calculating the deltas inside LSTM bock:

- Block output: $\delta y^t = \Delta^t + R_z^T \delta z^{t+1} + R_i^T \delta i^{t+1} + R_f^T \delta f^{t+1} + R_o^T \delta o^{t+1}$;
- Output gate: $\delta o^t = \delta y^t \odot h(c^t) \odot \sigma'(\bar{o}^t)$;
- Cell state: $\delta c^t = \delta y^t \odot o^t \odot h'(c^t) + \delta c^{t+1} \odot f^{t+1}$;

- Forget gate: $\delta f^t = \delta c^t \odot c^{t-1} \odot \sigma'(\bar{f}^t)$;
- Input gate: $\delta i^t = \delta c^t \odot z^t \odot \sigma'(\bar{\imath}^t)$;
- Block input: $\delta z^t = \delta c^t \odot i^t \odot g'(\bar{z}^t)$;

Δ^t is delta passed down from previous layer, which is the delta of hidden layer B in our case. Referring to [25], $\delta_j \stackrel{\text{def}}{=} \frac{\partial \mathcal{L}}{\partial a_j}$, where j is any unit in the network, $\mathcal{L}$ is loss function, and a is network unit input.

- $\delta x^t = W_z^T \delta z^t + W_i^T \delta i^t + W_f^T \delta f^t + W_o^T \delta o^t$;

Formula above is used for train the layer after LSTM block (backward pass), which is hidden layer A in our case.

In the second stage, calculate the deltas of weights:

- $\delta W_* = \sum_{t=0}^{T} < \delta *^t, x^t >$;
- $\delta R_* = \sum_{t=0}^{T-1} < \delta *^{t+1}, y^t >$;
- $\delta b_* = \sum_{t=0}^{T} \delta *^t$;

* can be any of $\{z, i, f, o\}$, $<*,*>$ is the outer product of two vectors and T is total steps to take back in time [23]. For the output layer, we use LogSoftMax activation function, and we choose negative log likelihood as the loss function, which can be expressed as, $\mathcal{L}(x, class) = -\log\left(\frac{e^{x[class]}}{sum(e^{x_j})}\right)$.

According to formulas above, in BPTT, the backpropagation takes inverse order of time, which is shown in Figure 14:

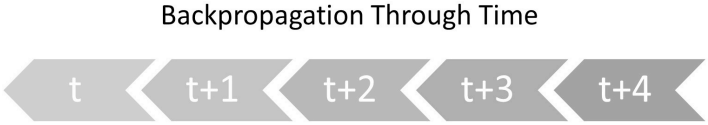

Figure 14: backward pass illustration

Chapter 4

Implementation

1. Torch7

In our research, we use Torch7 framework to build and train our model. Torch7 is a numeric computing framework and machine learning library based on Lua [26]. The reason why we chose Torch7 is it has three considerable advantages: fast, easy to build and train neural networks and capability to be extended.

2. GPU and CUDA

CUDA is a parallel computing platform and application programming interface (API) model invented by NVIDIA. It enhances computing performance by using CUDA enabled graphics processing unit (GPU). Over the last decade, GPUs have accelerated applications in various fields (e.g. Bioinformatics [27], Graph [28] and Tree Traversal [29]). Training a Recurrent Neural Network is time consuming since its complex internal structure. Even though we use relatively small training sample set (3,000 samples), traditional CPU computing cost too much time than we expect. In our research, with implementation of CUDA computing, the training and tuning process have been significantly accelerated.

3. Dataset

"Financial forecasting is an example of signal processing problem which is challenging due to high noise, small sample size, non-stationarity, and non-linearity" [1]. To overcome these difficulties, we optimize the input data under two principles: increasing the sample size within a short real time range; augment number of features.

We randomly choose six stocks from different industry: Black Rock, Inc.(BLK), Alphabet, Inc.(GOOGL), QUALCOMM, Inc.(QCOM), Exxon Mobil Corp.(XOM), International Business Machines Corp.(IBM), and JPMorgan Chase & Co. (JPM). Among those, BLK belongs to asset management industry; GOOGL and IBM are from information technology industry; QCOM is in communication equipment industry; XOM belongs to energy industry; and JPM is from banking industry. We use hourly historical stock data from 07/08/2013 to 05/11/2015 (9:30 a.m. to 16:00 p.m. Eastern Time) from IQFeed.net. The data size of each is stock is 3,703, and the first 300 samples and last 3 samples are used as past stock information as features and prediction target respectively. The size of training set is 3,000 and size of testing set is 400.

Initial stock data fragment of BLK is shown in Table 3 below:

Table 3: BLK Initial Stock Data Fragment

20130708	9	259.06	260.59	257.30	258.93	106288
20130708	10	259.23	260.55	256.54	256.89	163455
20130708	11	256.91	257.35	255.87	257.17	76463
20130708	12	257.00	257.60	256.73	257.24	79765
20130708	13	257.26	258.11	256.58	257.99	53033
20130708	14	258.05	258.13	257.36	257.98	63655
20130708	15	257.89	257.96	256.85	257.14	146390
20130708	16	257.14	257.16	256.85	256.85	57213
20130709	9	259.40	259.65	257.29	257.73	63097
20130709	10	257.52	257.91	255.24	256.95	160938
20130709	11	256.80	258.40	256.72	257.91	97113
20130709	12	257.79	258.64	257.47	258.30	81779
20130709	13	258.48	260.54	258.03	259.90	115252
20130709	14	260.02	262.69	260.02	262.58	109576
20130709	15	262.58	263.77	262.17	263.39	209498
20130709	16	263.33	263.58	263.21	263.38	39881

Each row represents one hour in one market day, columns from left to right are date, time, open price at the beginning of this hour, highest price within one hour, lowest

price within one hour, close price at the end of this hour and transaction volume within one hour.

To optimize input data, we create 369 features with target stock data, stock index data and competitor stock data. Stock index data is the value of a category of the stock market, it is calculated by the weighted average of selected stock's price. In our research, we use NASDAQ and S&P 500 index. Two competitors are chosen for each target stock according to finance.yahoo.com. Details of competitor stocks of each target stock are shown in Table 4:

Table 4: Target Stocks Competitor Stocks Details

Target Stock	**Competitor A**	**Competitor B**
BLK	State Street Corp.(STT)	Legg Mason Inc.(LM)
GOOGL	Apple Inc.(AAPL)	Facebook Inc.(FB)
QCOM	Microsoft Corp.(MSFT)	Texas Instruments Inc.(TXN)
XOM	Devon Energy Corp.(DVN)	Cabot Oil & Gas Corp.(COG)
IBM	Cisco Systems Inc. (CSCO)	HP Inc. (HPQ)
JPM	American Express (AXP)	Citigroup Inc. (C)

MATLAB is applied to process the data, Table 5 shows the combination of the features,

Table 5: Illustration of Features

Features:	Description:
1	Target stock current hour open price
2	NASDAQ current hour open index
3	SP500 current hour open index
4	Competitor A current hour open price
5	Competitor B current hour open price
6~65	Target stock previous 60 hours close price
66~125	Target stock previous 60 hours transaction volume
126~185	NASDAQ previous 60 hours close index
186~245	S&P 500 previous 60 hours close index
246~305	Competitor A previous 60 hours close price
306~365	Competitor B previous 60 hours close price
366	Target stock previous 200 hours average close price
367	Target stock previous 200 hours average high price
368	Target stock previous 200 hours average low price
369	Standard deviation of target stock previous 200 hours close price

Chapter 5

Experiments and Results

1. Overview

We optimize LSTM model by comparing the accuracy of various parameter settings (number of hidden layer units and number of samples in sequence). In order to test the universality of our LSTM model, QCOM is randomly selected as target stock during the tuning process. Then the optimized model is applied to all selected stocks to simulate real world stock trading.

2. Configuration of Number of Neurons

Optimizing the number of neurons in hidden layers for different tasks is still an unsolved problem in ANN research area. Too few hidden units cause high errors since under-fitting, while too many hidden units could also lead high errors because of over-fitting [30]. Rules of thumbs were proposed by several researchers for determining the optimized number of hidden units for all ANN tasks. For example: The number of hidden units should never be more than twice than units in input layer [31]; size of hidden units should be somewhere between input units and output units [32]; the amount of neurons should capture 70~90% of the variance of the input data set [33].

We test different configurations of hidden layer settings by using five samples in one sequence, and iteration based on training error (Δ_{error} less than 10^{-4}). Since the outputs and inputs of LSTM block are identical, number of neurons in hidden layer A and B should be the same. Test results are shown in Figure 15:

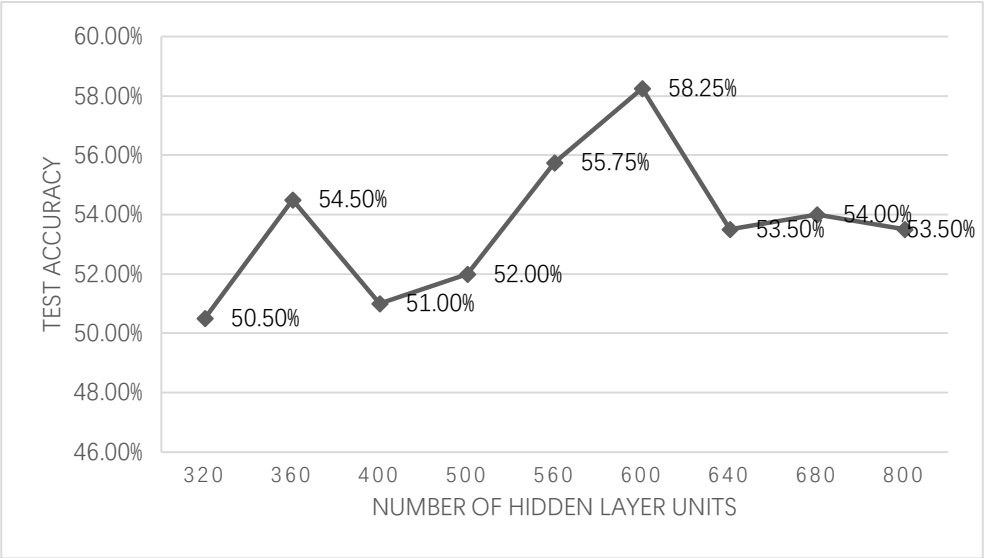

Figure 15: QCOM Test Accuracy with Various Number of Hidden Layer Units

Total number of neurons in both hidden layer ranges from 320 to 800. The obtained results show test accuracy increases along with number of neurons when it less than 600 units. The accuracy reaches peak point when number of neurons equals 600, then starts dropping.

Figure 16 shows the average training time for each iteration (all sequences pass forward then backward through the model counts as one iteration)

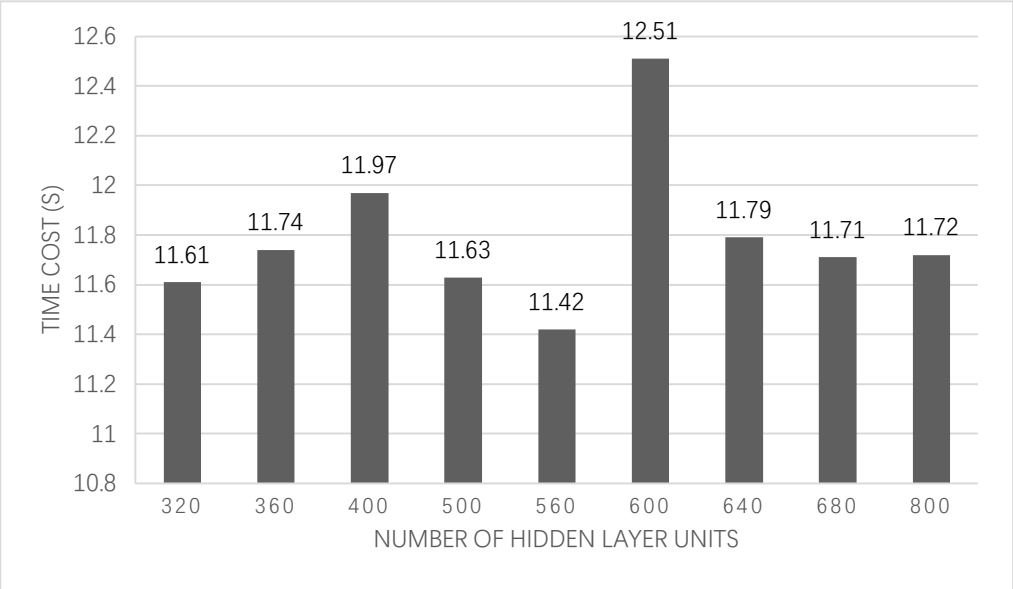

Figure 16: QCOM Average training time of one iteration with Various Number of Hidden Layer Units

The average time cost for one iteration does not increase or decrease monotonically, but fluctuate around 12 seconds per iteration. One possible explanation of the phenomenon is that training time of LSTM model does not change along with number of neurons in the system when the number of hidden layers is constant.

3. Configuration of Sequence Size

Sequence size is the maximum number of steps taking back through time during backpropagation; moreover, the size decides the maximum amount of previous steps stored in LSTM block. We test various sequence size settings from 2 to 70 to optimize our LSTM model with QCOM.

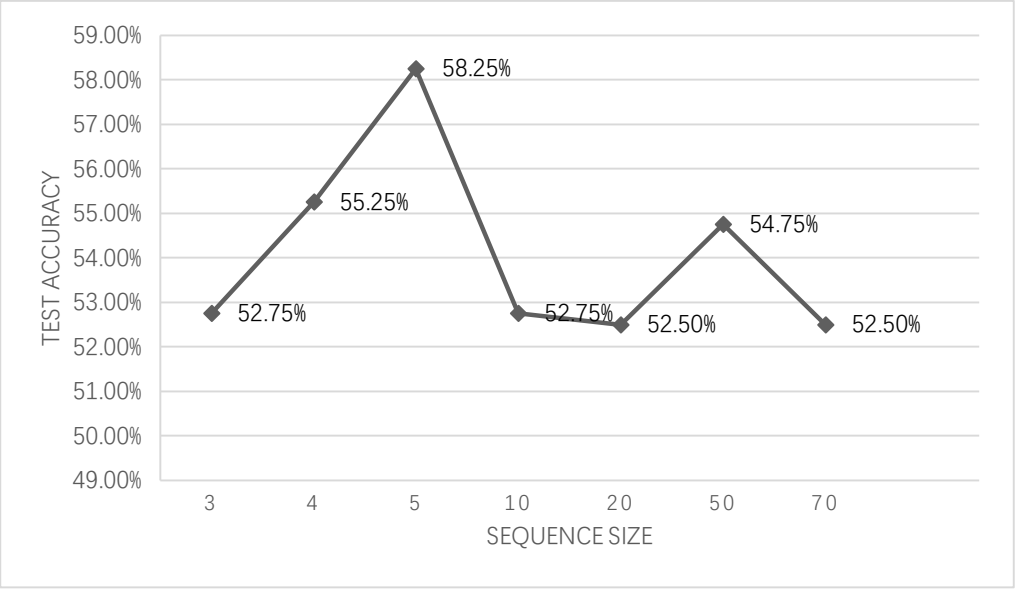

Figure 17: QCOM Test Accuracy with Various Number of Samples in Sequence

The obtained results show test accuracy increases dramatically with sequence size less than five, and decreases when sequence size larger than five. Theoretically, LSTMs can train to bridge time lags excess of 1000 discrete time steps [6]. But in our case, financial market data is non-stationarity and non-linearity [1]; feature correlation can be totally different time to time. With a long range of time steps, the increased uncertainty brings error.

Figure 18 shows the average time cost of each iteration under different sequence size settings:

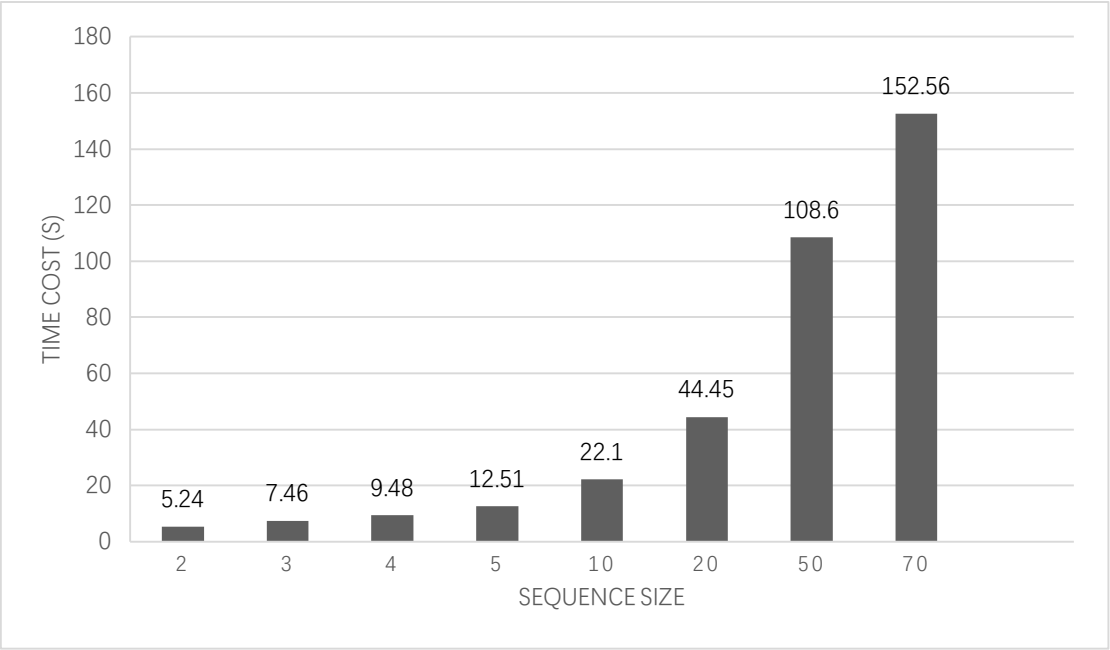

Figure 18: QCOM Average training time of one iteration with Various Sequence Size

Average training time of one iteration increases dramatically when sequence size goes up. Even though GPU acceleration is applied in our model, when sequence size exceeds 50, training time for one iteration needs over 100 seconds; moreover, total training time of 70 sequence size costs 16,661.36 seconds (4.63 hours).

By optimizing the LSTM model configuration (600 hidden units, 5 samples in one sequence), test accuracy for all six stocks is shown in Table 6:

Table 6: Test Accuracy of All Six Stocks with Optimized Configuration

Stock Name	**Test Accuracy**
BLK	59.50%
GOOGL	49.75%
QCOM	58.25%
XOM	55.50%
IBM	54.50%
JPM	51.50%

4. Stock Trading Simulator

We build a stock trading simulator based on prediction results made by the LSTM model to test if the LSTM model makes profit in real world stock trading activity. Recall three target classes are: class 1 as increasing between 0%~1%, class 2 as increasing above 1% and class 3 as not increasing (less than 0%) within next three hours.

For each stock we set initial investment principle as $ 1 million, and at each time t, transaction amount (money used to buy stocks) is no more than $200,000. Moreover, the simulator buys stock at the beginning of time t, and sells stock at the end of time t; however, in real world, selling transaction can be taken place during any time when "trigger" activated. "trigger" refers to a certain condition that activates the selling transaction. For instance, for class 2, the "trigger" is set as 1%, that is, when stock price exceed more than 1% of the price we bought it, then sell the stock, otherwise sell it at the end of 3rd hour with its closing price. The "trigger" is difficult to set for class 1 in

our simulator, since class 1 refers to 0%~1% increment. For conservative results, we set “trigger” equals to 0.1% for class 1. In real world, “trigger” for class 1 can be set as any price larger than buying price.

The simulator includes two pools: money pool and stock pool. Money pool refers to cash balance available for investment, and stock pool refers to stock balance. The process of our trading simulator can be divided into three stages:

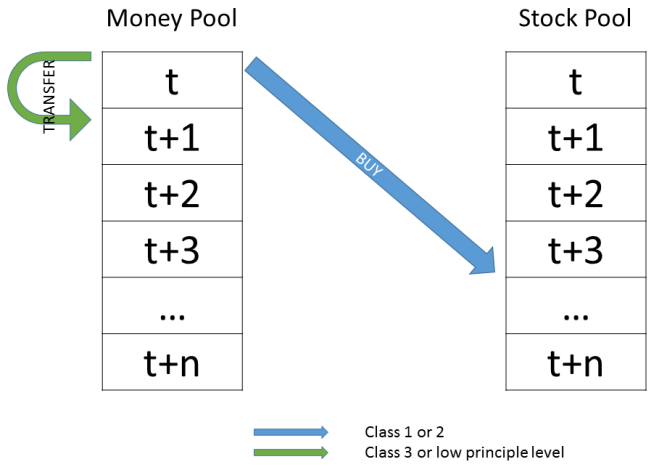

Figure 19: Illustration of Stage 1 of Stock Trading Simulator

Stage 1 is shown in Figure 19. At the beginning of each time step t, if the prediction result is class 1 or class 2, and the amount of cash is more than $200,000, simulator buys stocks at its current open price. Once the buying transaction completed, cash is taken away from money pool, and stocks are stored into stock pool at t+3. However, if the prediction result is 3, or the amount of cash is less than $200,000, simulator transfers all the cash to next time step t+1.

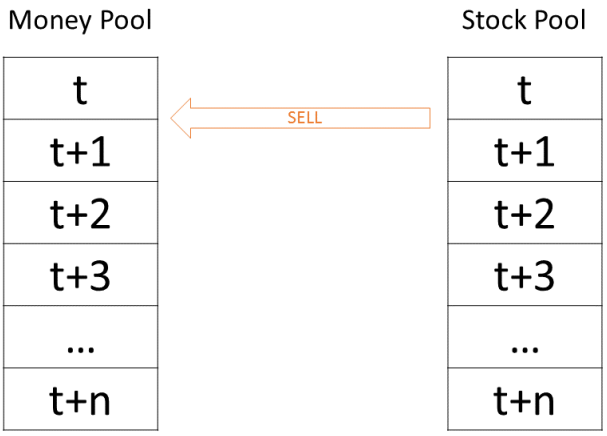

Figure 20: Illustration of Stage 2 of Stock Trading Simulator

Stage 2 is shown in Figure 20. At the end of each time step t, stocks that previously bought in time step t-3 are sold according to "trigger" rule. For example, on one hand, if prediction result is 1 and real result is 1 or 2, "trigger" will be activated and stocks will be sold at (1+0.1%)*buying price (profit gained); on the other hand, if prediction result is 1 and real result is 3, "trigger" will not be activated and stocks will be sold at the closing price of time t+3 (loss profit). Nevertheless, if prediction result is 2 and real result is 2 as well, "trigger" will be activated and stocks will be sold at (1+1%)*buying price (profit gained); on the other hand, if prediction result is 1 and real result is 1 or 3, "trigger" will not be activated and stocks will be sold at the closing price of time t+3 (loss profit).

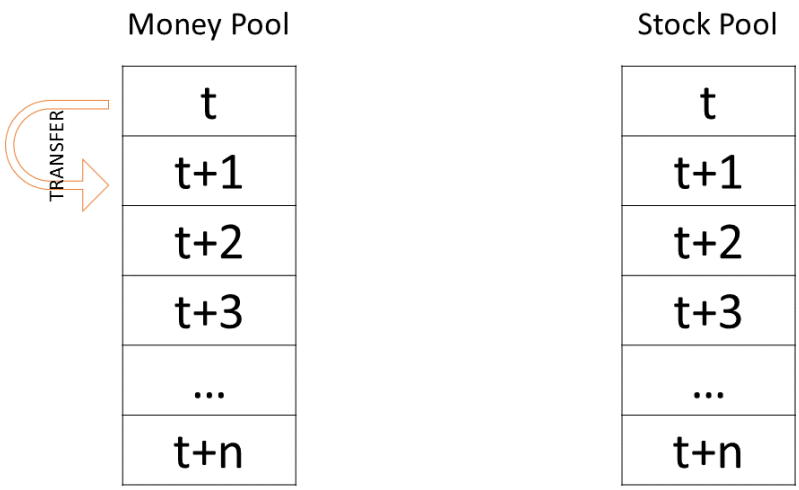

Figure 21: Illustration of Stage 3 of Stock Trading Simulator

Stage 3 is shown in Figure 21. Once stage 1 and 2 are completed, all the cash balance at time step t is transferred to next time t+1.

Moreover, if buying transactions take place at the last three time stages, stock pool will remain occupied at t+n+1, t+n+2 and t+n+3. Under this circumstance, at these three time steps, only stage 2 and stage 3 will be applied.

Trading simulation results of all six stocks is shown in Table 7:

Table 7: Six Stocks Trading Simulation Results

Stock Name	**Profit (in USD)**	**Rate of Return**
BLK	$71,651.31	7.17%
GOOGL	$90,357.99	9.04%
QCOM	$62,141.20	6.21%
XOM	$53,408.52	5.34%
IBM	$67,877.91	6.79%
JPM	$67,796.40	6.78%

The obtained results show the simulator made profit on all six stocks by using LSTM model prediction results. GOOGL gained the most profit of $90.357.99 while XOM gained the least at $53,408.52. Average rate of return is 6.89% (during 400 hours' period).

Chapter 6

Conclusions and Future Work

In this work, we proposed forecasting stock market using LSTM, a sub-class of Recurrent Neural Network. In order to test the universality of the LSTM model, one stock was randomly selected as tuning stock to test the optimized configuration of the Model. Then the LSTM model was applied to other five randomly selected stocks. As results, the LSTM model achieved 59.50%, 49.75%, 58.25%, 55.50%, 54.50% and 51.50% prediction accuracy of all six stocks over three target classes.

A stock trading simulator was developed to verify feasibility of our model in real world stock trading activities. During 400 hours testing period, 6.89% average rate of return was gained by the six stocks, GOOGL achieved the highest rate of 9.04% while XOM gained the least rate of 5.34%.

Some future works can be studied. One can apply after-market stock data to inputs. After-market stock data is difficult to get but applying the data can significantly enlarge training sample size and reduce noise. More, we only test LSTM model in our thesis, other Recurrent Neural Network structures can also be applied to stock market prediction such as, Echo State Network, Neural Turing Machines and Continuous-time RNN.

Bibliography

[1] C. Giles, S. Lawrence and A. C. Tsoi, "Noisy Time Series Prediction using a Recurrent Neural Network and Grammatical Inference," *Machine Learning,* pp. 161-183, July/August 2001.

[2] F. E. Tay and L. Cao, "Application of support vector machines in fnancial time series forecasting," *Omega,* pp. 309-317, August 2001.

[3] B. G. Malkiel, "Efficient Market Hypothesis," in *The world of Economics*, Palgrave Macmillan UK, 1991, pp. 211-218.

[4] Y. S. Abu-Mostafa and A. F. Atiya, "Introduction to Financial Forecasting," *Applied Intelligence,* pp. 205-213, 1996.

[5] S. Haykin, Neural Networks, A Comprehensive Foundation 2nd Edition, Simon & Schuster/ A Viacom Company, 1999.

[6] S. Hochreiter and J. Schmidhuber, " Long short-term memory," *Neural Computation ,* vol. 9, no. 8, pp. 1735-1780, 1997.

[7] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke and J. Schmidhuber, "A Novel Connectionist System for Improved Unconstrained Handwriting Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 31, no. 5, pp. 855-868, 2009.

[8] A. Graves, A.-r. Mohamed and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 2013.

[9] N. maknickiene, A. V. Rutkauskas and A. maknicksas, "Investigation of financial market prediction by recurrent neural network," *Innovative Infotechnologies for Science, Business and Education,* vol. 2, no. 11, pp. 3-8, 2011.

[10] Z. Iqbal, R. Ilyas, W. Shahzad, Z. Mahmood and J. Anjum, "Efficient Machine Learning Techniques for Stock Market Prediction," *Zahid Iqbal et al Int. Journal of Engineering Research and Applications,* vol. 3, no. 6, pp. 855-867, 2013.

[11] A. Vellidoa, P. Lisboaa and J. Vaughanb, "Neural networks in business: a survey of applications (1992–1998)," *Decision Support Systems,* vol. 19, no. 4, pp. 301-320, 1997.

[12] A. F. Atiya, "Bankruptcy Prediction for Credit Risk Using Neural Networks: A Survey and New Results," *IEEE TRANSACTIONS ON NEURAL NETWORKS,* vol. 12, no. 4, pp. 929-935, 2001.

[13] W. Treacy and M. Carey, "Credit risk rating at large US banks," *J.Banking & Finance,* vol. 24, pp. 167-201, 2000.

[14] A. Maithili, R. V. Kumari and S. Rajamanickam, "NEURAL NETWORK TOWARDS BUSINESS FORECASTING," *IOSR Journal of Engineering,* vol. 2, no. 4, pp. 831-836, 2012.

[15] K. Golic, "Application of a Neural Network Model for Solving Prediction Problems in Business Management," *Journal of Economics,* vol. 1, no. 1, pp. 146-149, 2013.

[16] H. T. Siegelmann and E. Sontag, "Turing computability with neural nets," *Applied Mathematics Letters,* vol. 4, no. 6, pp. 77-80, 1991.

[17] L. Lee, "Learning of Context-Free Languages: A Survey of the Literature (TR-12-96)," Harvard University, 1996.

[18] Y. Sakakibara, "Recent advances of grammatical inference," *Theoretical Computer Science,* vol. 185, no. 1, pp. 15-45, 1997.

[19] E. Schmidhuber and F. A. G. Jurgen, "LSTM Recurrent Networks Learn Simple Context Free and Context Sensitive Languages," *IEEE Transactions on Neural Networks,* vol. 12, no. 6, p. 1333–1340, 2001.

[20] M. Osborne and T. Briscoe, "Learning Stochastic Categorial Grammars," CoNLL, 1997.

[21] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia and A. Baskurt, "Action Classification in Soccer Videos with Long Short-Term Memory Recurrent Neural Networks," *Artificial Neural Networks – ICANN 2010,* vol. 6353, pp. 154-159, 2010.

[22] F. A. Gers, J. Schmidhuber and F. Cummins, " Learning to forget: continual prediction with LSTM," in *Artificial Neural Networks, 1999. ICANN 99. Ninth International Conference*, Edinburgh, 1999.

[23] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink and J. Schmidhuber, "LSTM: A Search Space Odyssey," arXiv preprint arXiv:1503.04069, 2015.

[24] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE,* vol. 78, no. 10, pp. 1550 - 1560, 2002.

[25] A. Graves, Supervised Sequence Labelling with Recurrent Neural Networks, Springer Berlin Heidelberg, 2012.

[26] R. Collobert, K. Kavukcuoglu and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, Sierra Nevada, Spain, 2011.

[27] D. Li and M. Becchi, "Multiple Pairwise Sequence Alignments with the Needleman-Wunsch Algorithm on GPU," in *High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion*, Salt Lake City, 2012.

[28] D. Li and M. Becchi, "Deploying Graph Algorithms on GPUs: An Adaptive Solution," in *Parallel & Distributed Processing (IPDPS), 2013 IEEE 27th International Symposium on*, Boston, 2013.

[29] D. Li, H. Wu and M. Becchi, "Nested Parallelism on GPU: Exploring Parallelization Templates for Irregular Loops and Recursive Computations," in *Parallel Processing (ICPP), 2015 44th International Conference on*, Beijing, 2015.

[30] S. Xu and L. Chen, "A Novel Approach for Determining the Optimal Number of Hidden Layer Neurons for FNN's and Its Application in Data Mining," in *5th International Conference on Information Technology and Applications (ICITA 2008)*, Cairns, Australia, 2008.

[31] M. J. A. Berry and G. S. Linoff, Data Mining Techniques: For Marketing, Sales, and Customer Support, New York: John Wiley & Sons, 1997.

[32] A. Blum, Neural Networks in C++, New York: Wiley, 1992.

[33] Z. Boger and H. Guterman, "Knowledge extraction from artificial neural network models," in *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference*, Orlando, FL, 1997.