Financial Time Series Prediction Using Deep Learning

Yanjing Peng *Yau Yuk* *Lam chun hei*

Department of Computer Science & Engineering

Hong Kong University of Science and Technology

Abstract

Financial time series prediction is an important issue in academic and practical quantitative finance field. Due to the prosperity of deep learning methodology, especially succeed in image processing and natural language processing field, in this project, we try to apply deep learning approach to solve financial time series prediction problem. A deep neural network (DNN) is derived using raw financial data inputs (1-minute resolution) to predict the temporal trend of China CSI 300 Index[1] in Chinese A-Share Market. Based on the prediction of neural network, we derive a trend-following investment strategy to make profits using the soft probabilistic outputs information of the NN. The investment strategy has shown favorable performance comparing to contemporary benchmarks over two-years back-testing.

Keywords: *DNN; Financial time series prediction; Trend-following strategy; Back-testing*

1 Introduction

1.1 Project Background

Time series analysis is of major importance in a large number of research topics and many engineering issues. The main purpose of financial time series prediction is to predict the future trend and other latent pattern based on the current and previous observations. For instance, given the samples of a time series $x(n)$ $n \in Z$ and the time interval T, we want to predict the sign of $x(n+T) - x(n)$. Let $y(n)$ represents the ground true value of $sign(x(n+T) - x(n))$ and $\hat{y}(n)$ denotes the predicting value of $sign(x(n+T) - x(n))$. A loss function $L(y(n),\ \hat{y}(n))$ could be used to quantify the accuracy of the prediction model.

1 CSI 300 Index consists of the 300 largest and most liquid A-share stocks. The Index aims to reflect the overall performance of China A-share market. http://www.csindex.com.cn/en/indices/index-detail/000300

Numerous works applied statistical approaches to study time series data. The Kalman Filter [1] and Auto-Regressive (AR) Model are important statistical approaches, where ARMA [2] and ARIMA [3] (Auto-Regressive Integrated Moving Average) models are further generalization of the AR which is derived by combining the AR models with the moving-average (MA) models. Other common AR schemes is the ARCH (Autoregressive Conditional Heteroskedasticity), GARCH [4] (Generalized ARCH) and the ARIMA algorithms, and are often applied to financial series forecasting, pointed by Ariel et al. (2017) [5].

Since we notice that time series prediction problem could be transferred into a classification issue, machine learning algorithm could provide helps. Kanas [6] showed that the non-linearity of the models being used for time series forecasting is of major importance. Thus, one of the most commonly applied schemes is the K-Nearest neighbors (kNN). The kNN algorithm assumes a similarity between time series sequences that occurred in the past, to future sequences, and as such, the nearest-neighbors are used to yield the forecasts of the kNN model. Ban et al. [7] applied kNN to multi-asset data, utilizing a set of stocks sharing similar dynamics. Thus, achieving less bias and improved resiliency to temporal fluctuations, than those of a single stock, pointed by Ariel et al. (2017).

Hidden Markov Models (HMM) are also commonly applied to financial time series forecasting. HMM encodes a finite-state machine with a fixed number of non-observable states. These hidden variables are assumed to be related by a Markov process allowing HMMs to be applied to temporal pattern recognition tasks. Hassan [8] applied HMMs to forecasting stock market, by training an HMM model on a specific stock, and matching past temporal patterns to current stocks patterns. The prediction is derived by extrapolating current prices based on past events. Decision Trees [9] and SVM [10, 11] were also applied to time series prediction, pointed by Ariel et al. (2017).

However, these kinds of ML classification approaches need good features to help to separate the training data and feature extraction is a big challenge in these works. Deep neural network has the ability of extracting good features automatically from the raw inputs, so in this project, we try to apply DNN [12]to do the financial time series task and derive a trend-following investment strategy.

1.2 Project Description

This project aims to apply deep neural network to predict the trend of financial time series, to be more specific, utilizes raw financial data inputs (1-minute resolution) to predict the temporal trend of China CSI 300 Index in Chinese A-Share Market and derives a trend-following investment strategy, comparing its performance with benchmarks along two-years back-testing.

Our work mainly based on the paper[2] written by Ariel et al. (2017), but we also make some different application details comparing with the original method in that paper, mainly in two aspect, classification and trading strategy. This article documents our experimenting process and discusses our evaluation findings and applications. The sections are organized as follows.

In section 1, we introduce the background and some relevant works, basic idea of this project and list our individual contributions.

In section 2, we discuss the network structure and propose some possible modifications to the original model.

In section 3, we discuss how to utilize the probability information produced by DNN to set up a trend-following investment strategy and the difference comparing to the paper's method.

In section 4, we discuss the experiment details, including data collection, data preprocessing, training parameters, optimization method, and back-testing process.

In section 5, we make a conclusion and proposed some possible future works to make optimization.

1.3 Individual Contribution

The individual contribution of the project shown as Table 1.

[2] Financial Time Series Prediction using Deep Learning (Ariel et al. 2017)

Name	Contributions
Yanjing Peng	Built trading strategy; Implemented back-testing; Wrote project report.
Lam chun hei	Built and trained DNN model.
Yau Yak	DNN model validation; Presentation; Recorded video.

Table 1: Individual contributions

2 Deep Learning Prediction of Index Trend

2.1 Classification Specification

Given $x(n)$ $n \in Z$ and the time interval T, the difference between future price and current price is $y(n) = x(n+T) - x(n)$. In the paper (Ariel et al. 2017), the author set the difference into two states, going up or going down, which means as following:

$$y(n) = \begin{cases} 1, & x(n+T) - x(n) \geq 0 \\ -1, & x(n+T) - x(n) < 0 \end{cases}$$

However, under this classification, the model has a flaw that when the absolute value of $x(n+T) - x(n)$ is small, it is impossible to make profits even when the model is completely accurate in real investment market which charges for trading fares. To overcome this flaw, in our experiments, we set the difference into three states, increasing dramatically, decreasing dramatically or others, which means as following:

$$y(n) = \begin{cases} 1, & x(n+T) - x(n) \geq \alpha \\ -1, & x(n+T) - x(n) \leq \beta \\ 0, & others \end{cases}$$

Let $y(n)$ represents the ground true state of $x(n+T) - x(n)$ and $\hat{y}(n)$ denotes the predicting state of $x(n+T) - x(n)$. We use softmax loss function $L(y(n),\ \hat{y}(n))$ to calculate the training loss and training the model by minimizing the cross-entropy loss. The probability that a sample belonging to j ($j \in \{1, \dots, K\}$) is as following:

$$p(y(n) = j) = \frac{e^{Z_j}}{\sum_{i=1}^{K} e^{Z_i}}$$

2.1 DNN Structure

In order to predict the future index price trend $\hat{y}(n)$, we apply a classification neural network trained using the raw prices data of the previous M minutes.

$$\hat{y}(n) = \hat{y}(n|x^{n-M+1}, \dots, x^{n})$$

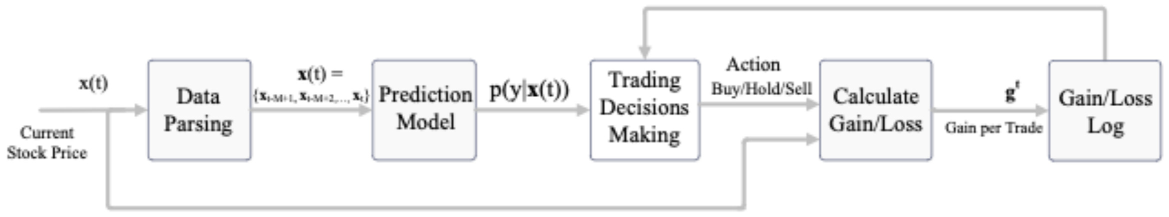


Figure 1: Schematic Illustration of our Deep-Learning based financial trends prediction system. (Ariel et al. 2017)

Since the author has tried out a large number of structures of neural networks, containing convolutional layers (however it did not yield significant accuracy improvement), we assume the structure of neural net implemented in the paper is optimal one which is shown as Fig. 2, and also Layer 5 should be changed into 20×3 FC and corresponding output is 3×1 because we set the difference as three states which is different with the paper.

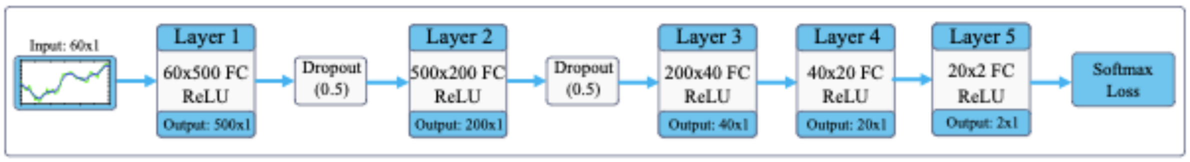


Figure 2: The neural network applied to predict the price trends. (Ariel et al. 2017)

3 Trend-Following Strategy

3.1 Benchmarks Illustration

In order to evaluate the performance of our trend-following strategy derived from DNN methodology, we develop a long and holding strategy as the benchmark to comparing the performance of both strategies over the two-years beck-testing period. Besides, I want to make an explanation about some finance terminology, long trade and short trade. While long trading means that buying a financial asset and hold it until you want to sell it (closing your position), short trading, a more complicated one, represents that firstly you borrow a financial asset from others and sell it in the market and then you must buy it back to return it to the lender.

3.2 Trade Opening Using Soft-Information

The outputs of softmax layer of the DNN represent the probabilities of future trend belonging to respective classes. In order to use soft-information $p(y|x)$ provided by the DNN model to derive trend-following investment strategy, in the paper, since the author set the future difference into two states, it opens trade in the following method. Let $O(n)\ n \in Z$ represents whether opening a new position or not at time n.

$$O(n) = \begin{cases} 1, & |p(y(n) = 2|x^{n-M+1}, \dots, x^n) - p(y(n) = 1|x^{n-M+1}, \dots, x^n)| \geq T_H \\ 0, & |p(y(n) = 2|x^{n-M+1}, \dots, x^n) - p(y(n) = 1|x^{n-M+1}, \dots, x^n)| < T_H \end{cases}$$

In our case, since we set the future difference into three states, it would open trade in a similar method.

$$O(n) = \begin{cases} long, & \hat{y}(n) = 1\ and\ p(y(n) = 1|x^{n-M+1}, \dots, x^n) \geq T_H \\ short, & \hat{y}(n) = -1\ and\ p(y(n) = -1|x^{n-M+1}, \dots, x^n) \geq T_H \\ none, & other \end{cases}$$

where T_H could help to control opening trade times.

3.3 Trade Closing Rules

Since it is an intra-day trading [3] strategy, the strategy follows these rules to close its position.

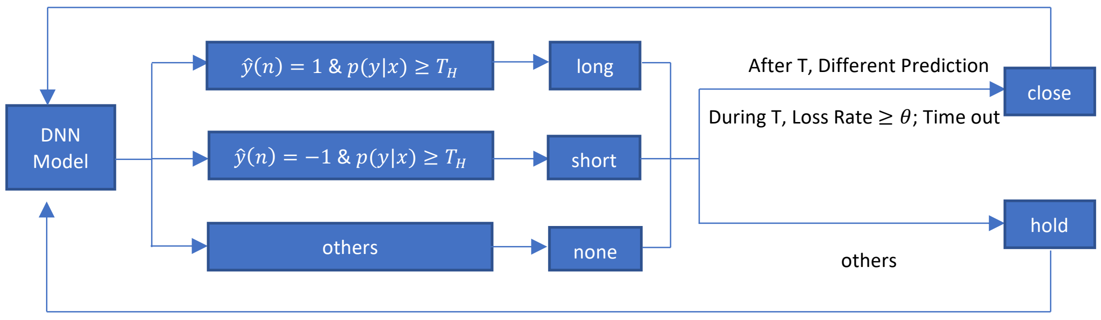
Rule1: After T minutes, if the DNN model's prediction result is different from the signal when we opened the position, we choose to close it and wait to the next signal.

Rule 2: After T minutes, if the DNN model's prediction result is same as the signal when we opened the position, we keep to hold the position.

[3] Intra-day trading means that the position should be opened and closed during one trading day.

Rule3: During T minutes, if the loss rate of the trade exceeds a certain threshold θ, we choose to close it and wait for the next signal.

Rule 4: When it is one minute before the closing time of the market, we choose to close our position since it is an intra-day trading and we do not want to take over-night risks[4].



DNN Model
$\hat{y}(n) = 1 \,\&\, p(y|x) \geq T_H$
$\hat{y}(n) = -1 \,\&\, p(y|x) \geq T_H$
others
long
short
none
After T, Different Prediction
During T, Loss Rate $\geq \theta$; Time out
close
hold
others

Figure 3: The proposed soft-threshold based trade opening and closing approaches.

4 Experiments and Results

4.1 Data Collection and Pre-processing

The proposed schemes were experimentally verified by using the market data of the CSI index given in one-minute resolution, purchased from Tong Hua Shun [5] which is a major financial information and data supplier in China. Based on regular trading hours (9:30AM-11:30AM & 1:00PM-3:00PM), we eliminated days with partial trading hours. For the whole dataset, it starts from 4th January, 2010 and ends 19th November, 2019.

As we study intra-day trading, we divided the data into different trading days, and extracted all training instance and validation instance during one certain day. For a certain day, we could derive about 200 different data samples, to be more specific, the number of samples derived within a certain day is $60 \times 4 - M - T$, where M is the length of previous prices vector and T is the prediction time interval. We use the data after 4th January, 2018 to act as back-testing original data and the data between 4th January, 2016 and 31st December, 2017 to act as validation data and others as training data. To make the number of instances of each class to

[4] Over-night risk means the probability of asset price going down due to news of the night
[5] Its website address: www.51ifind.com

be same, we resampled the training dataset and it is not needed for validation dataset. Besides, we also standardized our inputs by subtracting mean value and divided by standard deviation.

In light of labeling, for a certain day during the training time period, we choose a continuous prices series that having length of M which make a vector $(x^{n-M+1}, \dots, x^{n})$ and calculate the difference of $x^{n+T} - x^{n}$, using the formula in section 3.2 to determine the class label.

4.2 Parameter Determination

For parameter M, the length of previous index prices, in the paper, the author implemented validation method to determine the optimal value, which was set 60 and we also use that value.

For parameter T, the prediction time interval, in the paper, the author also implemented validation method to determine the optimal value, which was set 28 and we also use that value.

For parameter α and β, we determined their value by exploring the distribution of the prices change over T minutes, as shown in Figure 4 below. It is clear that the distribution is symmetrical and in hence, we set α to be the 75% quantile value and β to be the 25% quantile value, which is 6 and -6 respectively.

For parameter T_H, controlling the times of opening a new position, given that μ and σ are the mean value and standard deviation value of probabilities provided by the DNN model on validation dataset, we set it to be $\mu + \sigma$.

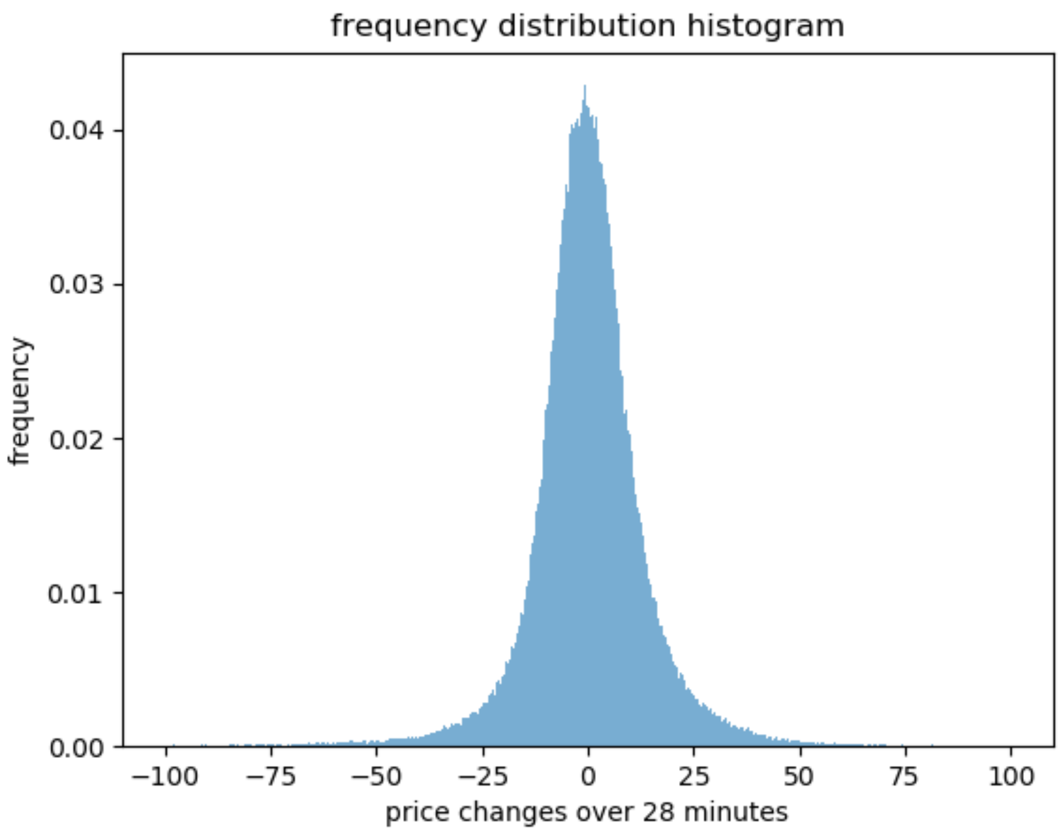


Figure 4: Price changes over T (set to be 28) minutes distribution.

4.3 Training DNN Model Process and Results

The neural network depicted in Figure 2 has 5 fully-connected layers with 500, 200, 40, 20, and 3 ReLU activation units. Dropout follows the first and second layers to avoid overfitting problem. The input is a 60×1 adjacent historical raw prices values. It is built using Keras [6] package and we choose Adam algorithm to minimize the model loss, where mini-batches of size is 100, the learning rate is 0.0001, beta_1 is 0.9 and beta_2 is 0.99. The training accuracy and validation accuracy is shown in Figure 5.

[6] Keras is a python package building deep learning models and its website address is www.keras.io

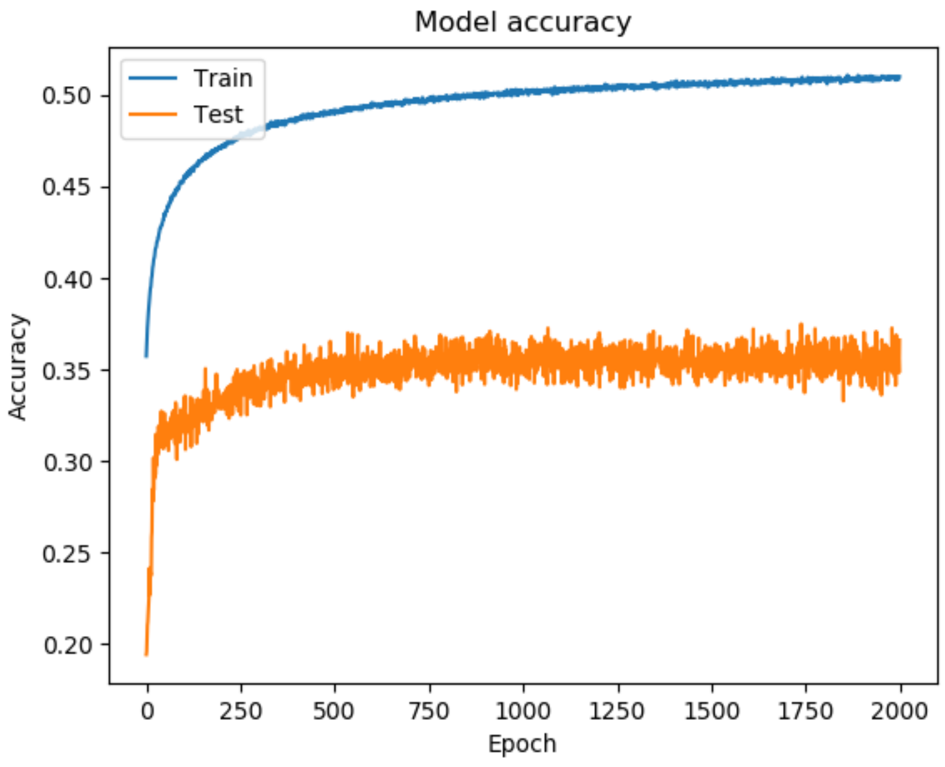


Figure 5: The training and test accuracy versus epochs

4.4 Back-Testing Process and Results

We implemented back-testing process based on the trade opening rules and closing rules in section 3 over the time period of 4th January, 2018 to 19th November, 2019. The cumulative profits of our trend-following strategy which achieved over 12% return rate based on DNN model is shown in Figure 6.

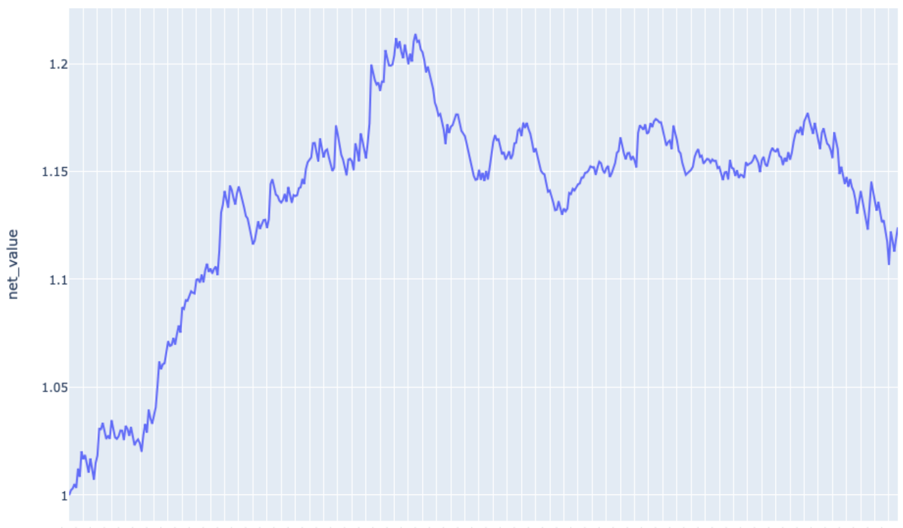


Figure 6: Cumulative profits of our trend-following strategy based on DNN model.

5 Conclusions

5.1 Conclusions

During this project, we proposed two main improvements based on the works of Ariel et al. (2017), setting the trend into three states instead of two and adding stop loss method to the trend-following intra-day trading strategy. We also completely built and trained a DNN model using Keras package, developed a trend-following investment strategy using soft-information provided by the DNN model, and implemented a back-testing process to evaluate a trading strategy quantitatively, which outperformed to the benchmarks.

5.2 Future Works

As pointed by Ariel et al. (2017), since the market's characteristics changes period by period, in practical world, it is needed to dynamically train the deep neural networks to adapt to the new market's pattern, otherwise, the profit of the strategy derived by the DNN model would be continuable. Besides, when we built and trained the DNN model, we found the accuracy of the model is not high enough. We guess the probable reasons may be the trained epochs are not enough or the DNN structure needs to be modified. Due to the limit of time and computing power, we did not implement these works and we want to focus on these issues in the future.

References

[1] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. Journal of basic Engineering, 82(1):35–45, 1960.

[2] James Durbin. Efficient estimation of parameters in moving-average models. Biometrika, 46(3/4):306–316, 1959.

[3] George EP Box and David A Pierce. Distribution of residual autocorrelations in autoregressive- integrated moving average time series models. Journal of the American statistical Association, 65(332):1509–1526, 1970.

[4] Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. Journal of economet- rics, 31(3):307–327, 1986.

[5] A. Navon and Y. Keller, “Financial time series prediction using deep learning,” 11 2017.

[6] Angelos Kanas and Andreas Yannopoulos. Comparing linear and nonlinear forecasts for stock returns. International Review of Economics & Finance, 10(4):383–398, 2001.

[7] Tao Ban, Ruibin Zhang, Shaoning Pang, Abdolhossein Sarrafzadeh, and Daisuke Inoue. Refer- ential knn regression for financial time series forecasting. In International Conference on Neural Information Processing, pages 601–608. Springer, 2013.

[8] Md Rafiul Hassan and Baikunth Nath. Stock market forecasting using hidden markov model: a new approach. In Intelligent Systems Design and Applications, 2005. ISDA’05. Proceedings. 5th International Conference on, pages 192–196. IEEE, 2005.

[9] Robert K Lai, Chin-Yuan Fan, Wei-Hsiu Huang, and Pei-Chann Chang. Evolving and clustering fuzzy decision tree for financial time series data forecasting. Expert Systems with Applications, 36(2):3761–3773, 2009.

[10] Francis EH Tay and Lijuan Cao. Application of support vector machines in financial time series forecasting. Omega, 29(4):309–317, 2001.

[11] Kyoung-jae Kim. Financial time series forecasting using support vector machines. Neurocomput- ing, 55(1):307–319, 2003.

[12] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. Nature, 521(7553):436–444, 2015.