Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 126 (2018) 264–272

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Knowledge Based and Intelligent Information and Engineering Systems, KES2018, 3-5 September 2018, Belgrade, Serbia

A hierarchical Deep neural network design for stock returns prediction

Oussama Lachiheb[a*], Mohamed Salah Gouider[b]

[a] *Université de Tunis, Institut supérieur de gestion, Smart Laboratory*
[b] *Université de Tunis, ESSECT, Smart Laboratory*

Abstract

We present in this paper a hierarchical Deep Neural Network for stock returns prediction. This DNN is trained in a high frequency context, we use 5 minutes returns of TUNINDEX stocks in a period of 4 years. The designed network aims to predict the next 5 minutes return of a given stock. The predictive power of our network is improved by the hierarchical design and stocks classification while the training process is simplified by dimensionality reduction techniques. Experimental study shows an accuracy up to 71% and a considerable improvement comparing to recent related works.

Keywords: Deep neural network, stock markets, finance, stock returns, deep learning;

1. Introduction

Research on stock prices predictability was the trend for many decades[6]; analysts, traders and researchers believe that movements on stock markets are predictable based on statistical methods and historical data. Identifying historical pattern of financial time series was the subject of many research works[7, 8]. In last 20 years and because of the evolution of storing and tracking systems, a huge amount of historical data is available for analysis, as a result, machine learning techniques became the main axis for new works.

[*] Corresponding author.
E-mail address: oussama.lachiheb@gmail.com

10.1016/j.procs.2018.07.260

Inspired from human brain, artificial neural networks[13] (ANN) represent a connection between multiple inputs called neurons, each neuron is weighted where weights are learned from historical data, they can learn complex patterns and extract useful information. Since early 1990's[1], ANN was being attractive for new researches on predictability of financial time series. Kimoto et al[9] presented a modular neural network with simulation on TOPIX index stocks predicting, Enke et al[10] used neural networks with information gain techniques to forecast future prices, Zhang et al[16] proposed an improved S&P500 forecasting model based on neural networks and bacterial chemotaxis optimization (IBCO), Bollen et al[14] employed fuzzy neural networks to investigate the impact of public mood on stock markets. Wang et al[11], Moghaddam et al[12], and Chen et al[15] worked on stock market index prediction with a set of machine learning techniques mainly ANN.

A deep neural network[17] (DNN) is an extension of artificial neural network where data processing is performed through multiple hidden layers which enable learning more complex pattern and representing useful results for analysis purpose. The encouraging results obtained by modelling DNN in image processing and text recognition[18,19,20] created a growing interest in applying these techniques in stock markets. DNN was employed by Yoshira et al[21] for stock market trends prediction to assist investors making decision, authors designed a recurrent neural network with a Restricted Boltzman Machine (RBM) to learn the impact of previous events on stock prices. Deep learning for stock market trends prediction was also the subject of Xiao et al[22] research work, they introduced a method of news text representation as input vector, this vector is then used as network input. Their experimental shows promising results. Directly related to our work, studies on predicting stock prices with DNN based on historical prices are limited and recent (2016 and 2017). Arevalo et al[1] proposed a high frequency DNN based on previous stock prices, they identified the impact of learning from mean price and its standard deviation on stock returns, they defined then a trading strategy using their designed network. The created portfolio achieved good results. Same method in Chong et al[4] paper, they studied the impact of data representation on network output. The concept is to select last 10 returns of the stock, encoding this selection with a representation technique (PCA, RBM, AE), the encoding result is the network input. Authors also performed a study on Korean stock market predictability with statistical methods. Despite mentioned works[1, 4] improved previous researches with performance and accuracy, they are still missing the exploitation of other stocks data. Chen et al[5] proposed DNN learned from both last stock returns and other stocks data, they identified a considerable correlation between the main stock and other stocks in terms of performance and trends. They studied also the hierarchical design and its impact on reducing information loss between DNN layers.

In this paper, we propose a high frequency hierarchical DNN for predicting stock return on next 5 minutes, this design learns from all market stocks and introduce inputs to the network in a hierarchical way. The originality in our work emerges from removing stocks with low linear correlation before the non-linear design of DNN, reducing processing complexity by connecting stocks to more general representation by holdings and business sector, and finally introducing second and third level inputs that reflect the global market status and the current context. These steps are simulated on the Tunisian stock market and its index called TUNINDEX, as proved by our experimental study, our employed techniques reduce DNN learning time while improving its predictive power. This paper is organized as follows: in section 2, we define some common elements used in the rest, in section 3, proposed method is presented. We describe the experimental framework and discuss obtained results in section 4. Section 5 is a general conclusion.

2. Definitions

2.1 ANN and DNN

The artificial neural network (ANN) was initially introduced in 1950's to model the human brain processing. ANN is based on perceptrons; each one takes several inputs and produces single output. The result is a mathematical function called activation function with weighted sum of inputs as parameters. For more complex problems modeling where output is calculated through multiple steps, neural networks with multiple hidden layers are used:

the outputs of each layer are the inputs for the next layer. We call them deep neural networks DNN, a full use case is illustrated in section 3. During this work, we use sigmoid[5] function $\varphi(x) = \frac{1+e^{-x}}{1+e^{-x}}$ as an activation function. These functions are smooth, continuous and monotonically increasing, they provide also a bounded range of output.

2.2 Log return

Log return is a technical indicator widely used in finance industry, calculated in (1), it enables comparison between two or many variables even originating from price series of unequal values.

2.2 TUNINDEX

Stock index is a measurement of the stock market globally or by sector, its value is calculated from the prices of its stocks based on a defined and public formula. TUNINDEX is the main index of the Tunisian stock market (www.bvmt.com.tn), founded in 1969 and composed from 75 stocks covering all business sectors. Our deep learning framework proposed in this paper is based on TUNINDEX and its stocks.

3. High frequency deep learning

The real time nature of stock markets requires real time analysis methods. Prediction performed on long time periods may not be the optimal tool for traders. We consider that high frequency data collected in each 5 minutes in this work has considerable contribution to the risk assessment and investment strategies.

For normalization purpose and to represent all stock information in a comparable metric, we use log returns calculated as follow:

$$R = \ln\left(\frac{p_t}{p_{t-1}}\right) = \ln(p_t) - \ln(p_{t-1}) \quad (1)$$

where p_t denotes the stock closing price at time t and p_{t-1} the stock closing price at time (t-1). During this paper, all stock returns are log return calculated in a time interval of 5 minutes.

3.1 Definition

For each stock, we need a predictor function $f(x)$ to predict the stock return at time (t+1). Deep neural networks DNN can identify complex pattern and model non-linear relationships; this relationship in one layer has the form:

$$h2 = \varphi(W1h1 + b) \quad (2)$$

where φ called activation function and $W1$ and b the model parameters, in multi-layer networks, the $y = f(x)$ predictor is obtained by applying the activation function on each layer where the input of a given layer is the output of the previous one.

3.2 Inputs dimensionality reduction

Given a stock i , we design a deep neural network to predict its return at time $(t+1)$. in the network bottom layer, we have 2 sets of inputs; $S = \{R(t,s): s \neq i\}$: the return off all TUNINDEX stocks at time t and $\{P = R(t,i), R(t-1,i), \ldots, R(t-e,i)$: the last e returns of stock i. To improve the prediction quality and the learning speed, we reduce the inputs dimension in 3 combined methods.

First, we calculate the statistical correlation factor r between each stock of S and the stock i, we remove stocks outside an interval $[-\varepsilon, \varepsilon]$ where ε is a defined threshold since they don't present any relationship with i neither in same direction nor in the opposite. Then we get a final set of related stocks called S_{final} where:

$$S_{final} = \{R(t,s\}: s \neq i \ \& \, r(s,i) \notin [-\varepsilon, \varepsilon\,] \quad (3)$$

The second point is S_{final} classification, stocks in this set are classified into n clusters, the activation function in first hidden layer is performed independently for each cluster stocks, which reduces considerably the computing complexity and enables non-linear relationships identification in a clean manner. Our observation on historical data shows a good correlation between stocks related to companies in the same group holding, we create a cluster of stocks for each group holding present in TUNINDEX, the remaining stocks are categorized by business sector, classification by business sector has shown its performance in[4], so the first hidden layer has $N = n + 1$ neurons since P returns are propagated to a single neuron.

The last method consists of introducing input variables in a hierarchical way, in each layer, we put new inputs. By propagating through layers, introduced inputs category is changing from specific variables to stock i and related stocks to more general variables related to the global market and its index. This technique avoids loss of a part of correlated features by the activation functions, reduces the parameters dimension in bottom layer and improves the training performance. This hierarchical configuration is illustrated in **Fig1.**

3.3 Deep neural network design

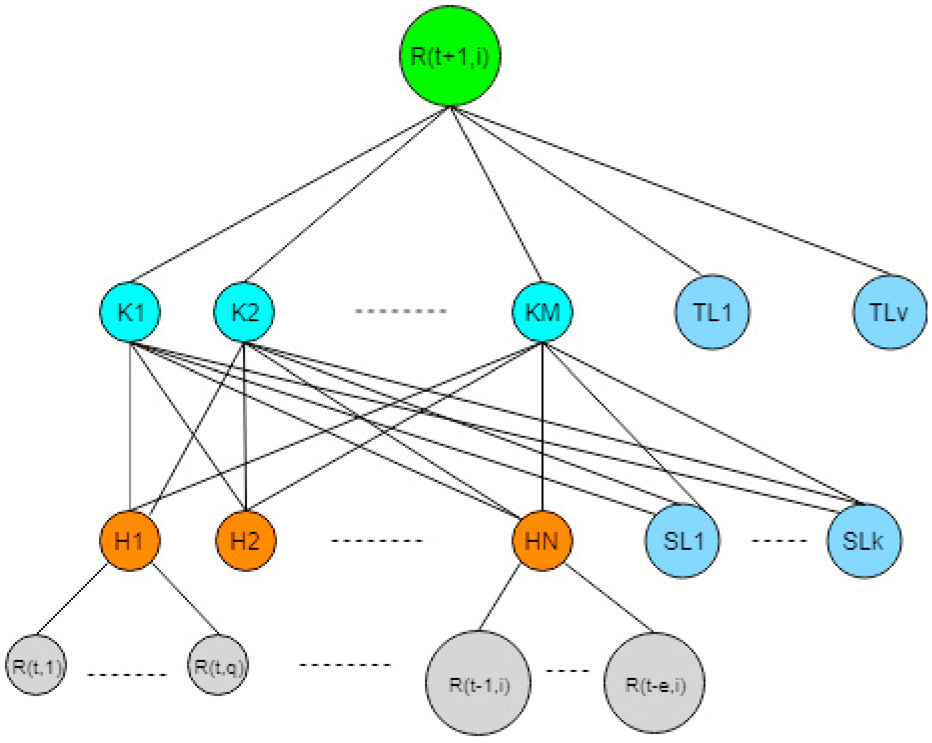

Fig1.Deep neural network design

The input layer as mentioned in **3.2** contains log returns of all TUNINDEX stocks after removing noisy ones with statistical correlation factor. We introduce also last e returns of stock i.
In the first hidden layer the H connections are calculated as follows:

$$H_j = \varphi\left(\sum_{s=0}^{m}(W_{sj}Rj + b_{sj})\right) \qquad (4)$$

We add in this layer 3 second level (SL) input neurons, these inputs present the current time in form of simple numeric values: day, hour and minute. As proved by our tests on TUNINDEX and in literature, time has an important correlation with stock prices, in fact, many trading strategies and orders are executed as scheduled jobs which impact prices in some specific periods of the trading day (see section 4).
In the second hidden layer, K connections are calculated as follows:

$$K_j = \varphi\left(\sum_{s=1}^{M}(W_{sj}Hj + b_{sj}) + \sum_{s=1}^{3}(W_{sj}SLj + b_{sj})\right) \qquad (5)$$

Traders in stock markets build their investment strategy on stock prices but also on market main indicators since they affect the market sentiment, we introduce in this layer 5 third level inputs (TL) related to TUNINDEX performance at time t. these inputs are described in table 1.

Table 1. TL inputs

Attribute	Description
Value	TUNINDEX current value
Number ups	Number of up direction stocks
Number of downs	Number of down direction stocks
Number of unchanged	Number of unchanged stocks
Total volume	Total stock value in Tunisian dinar (TND)

The output of our deep neural network $\hat{Y} = R(t+1, i)$ where R is the return of stock i at time $t+1$ is calculated in (6), note that the activation function in this case can be $\omega(x) = x$ since Y is already a continuous function.

$$\hat{Y} = \sum_{s=1}^{M}(W_s Ks + b_s) + \sum_{s=1}^{5}(W_s TLs + b_s) \qquad (6)$$

3.4 Deep neural network training

Given u the set of all inputs of the DNN, $\hat{Y} = f(u)$ is the predicted return and Y the real return of the stock, it can be called also the target. The difference between the predicted and the real return is an error function $\varepsilon(\hat{Y}, Y)$. The main target of network training is to minimize the error; this task can be done by updating weights W and bias b defined in (2). Let ϑ be the set of all network parameters.

$$\vartheta = \{W1 \dots WM, b1 \dots bM\} \qquad (7)$$

The gradient decent[23,24] is an iterative optimizer algorithm, its main goal is to minimize a defined function. In our work, for a parameter Wl from ϑ and a dataset$\{\hat{Y}, Y\}^N$, the objective function is:

$$O = \frac{1}{N}\sum_{n=1}^{N}(Y - \widehat{Y})^2 + \lambda \sum_{m=1}^{M} Wm^2 \qquad (8)$$

Where λ is a training parameter used to avoid overfitting. As explained in Algorithm 1, gradient descent output regularized parameters set where O is minimized to a defined error rate.

Algorithm 1: Gradient Descent Algorithm

```
Input: $\vartheta_0$ initial random set of all DNN parameters
Output: $\vartheta_{final}$ regularized parameters set
Calculate $O(\vartheta_0)$
While (($Diff > error\_rate$) $And$ ($index < iterationNumber$))
  For each $Wm$ in $\vartheta_0$
      Update $Wm$ based on learning rate
  End For
   $\vartheta_{n+1}$ =The new parameters set updated
   Calculate $O(\vartheta_{n+1})$

   $Diff = \left|\frac{O - O(\vartheta_{n+1})}{O}\right|$

End While
```

4. Experimental study

In this section, we perform an experimental study of the designed prediction framework, we focus on improvements provided by our dimensionality reduction methods and on comparison with literature standard networks.

4.1 Dataset description

TUNINDEX is the Tunisian stock exchange main index and composed from 75 stocks. From Jan 1st 2013 to December 31st 2017, we collected prices of 45 stocks every five minutes; a log return is then calculated and stored. We have chosen these stocks based on their volume and data availability during our dataset period. For each return, we store the current TUNINDEX global value and main indicators, we store also the current time. Our data covers 4 fiscal years including 1165 trading days and 81212 five minutes returns, as detailed in Fig2, we use the first 80% for DNN training and the remaining data for validation and tests.

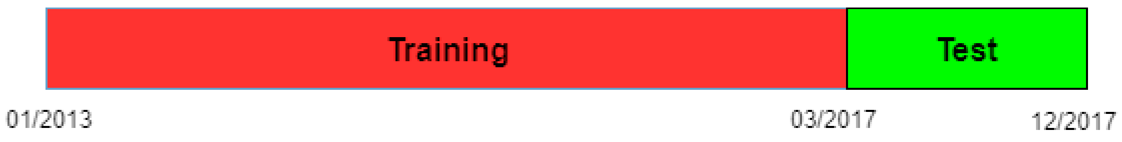

Fig. 2. Dataset repartition

4.2 DNN performance evaluation

The mean squared error MSE is a statistical indicator that measures the difference between a real value and a predicted value. It's calculated as follows:

$$MSE = \frac{1}{N}\sum_{n=1}^{N}\left(Y - \hat{Y}\right)^2 \quad (9)$$

Table 2. TUNINDEX experimental stocks

ID	Code	Name	Sector
1	ADW	ADWIYA	Health
2	ASD	ASSAD	Electronics
3	TJL	ATTIJARI LEASING	Finance
4	BNA	BANQUE NATIONAL AGRICOLE	Finance
5	CITY	CITY CARS	Commerce
6	LNDOR	LANDOR	Industry
7	STAR	SOCIETE TUNISIENNE D'ASSURANCE ET DE REASSURANCE	Insurance
8	XABYT	HEXABYTE	Services
9	TAIR	TUNISAIR	Transports
10	MAG	MAGASIN GENERALE	Distribution

From our available 45 stocks, we build 10 DNN for 10 TUNINDEX stocks, these stocks were selected randomly with diversifying business sectors, and their detailed description is available on table 2.

Each DNN is trained by minimizing the objective function defined in (8) with 2000 learning epochs (iteration) and with training parameter $\lambda = 10^{-3}$, $error\_rate = 10^{-6}$. For comparative studies, we trained for each stock three other networks: the artificial neural networks with all inputs (ANN), DNN with all inputs in first layer[4] called DNN1, DNN with hierarchical design with attributes and segmentation described in[5] called DNN2, these networks are compared to our proposed DNN with dimensionality reduction, input selection and correlation improvements called DNN_new.

The whole framework (network design and gradient descent algorithm) was implemented with Java 8 in an 8 Go RAM computer with Intel i5 3.2 GHZ processor running on windows 7. We compare the mean squared error MSE for all networks to measure their performance. Results are listed on Table3.

Table3. MSE Performance evaluation

Stock ID	ANN	DNN1	DNN2	DNN_new
1	0.2645	0.1275	0.1675	0.1311
2	0.2602	0.1120	0.1584	0.1230
3	0.2524	0.2092	0.1531	0.1140
4	0.2031	0.1153	0.1239	0.1625

5	0.2022	0.1984	0.1134	0.1478
6	0.2158	0.1456	0.1493	0.1388
7	0.2520	0.2617	0.2435	0.2519
8	0.2891	0.2389	0.1793	0.1408
9	0.2529	0.2109	0.1573	0.1188
10	0.2489	0.1856	0.1476	0.1054

By Analysing obtained results in Table3, we can identify three main conclusions:

- Traditional ANN performs worst in most cases which confirms that correlation between stock prices is a complex pattern and can be better identified by multiple hidden layers.
- The hierarchical design in DNN2 and DNN_new reduce the information loss in different layers and make prediction more accurate.
- Removing inputs with low correlation factor, segmentation by stocks group holding and the choice of new inputs on different layers made our proposed design the best performer for 60% of test stocks.

4.3 Up/down prediction accuracy

After evaluating our design performance by mean squared error, we perform in this section an up/down prediction accuracy experience based on the first 100 log return of each stock (table2) in our test dataset. If network output is positive then we predict an up direction else a down direction, we compare the correct predicted directions on Fig3.The proposed DNN realized a prediction accuracy up to 73% and it was the best network for 7 stocks.

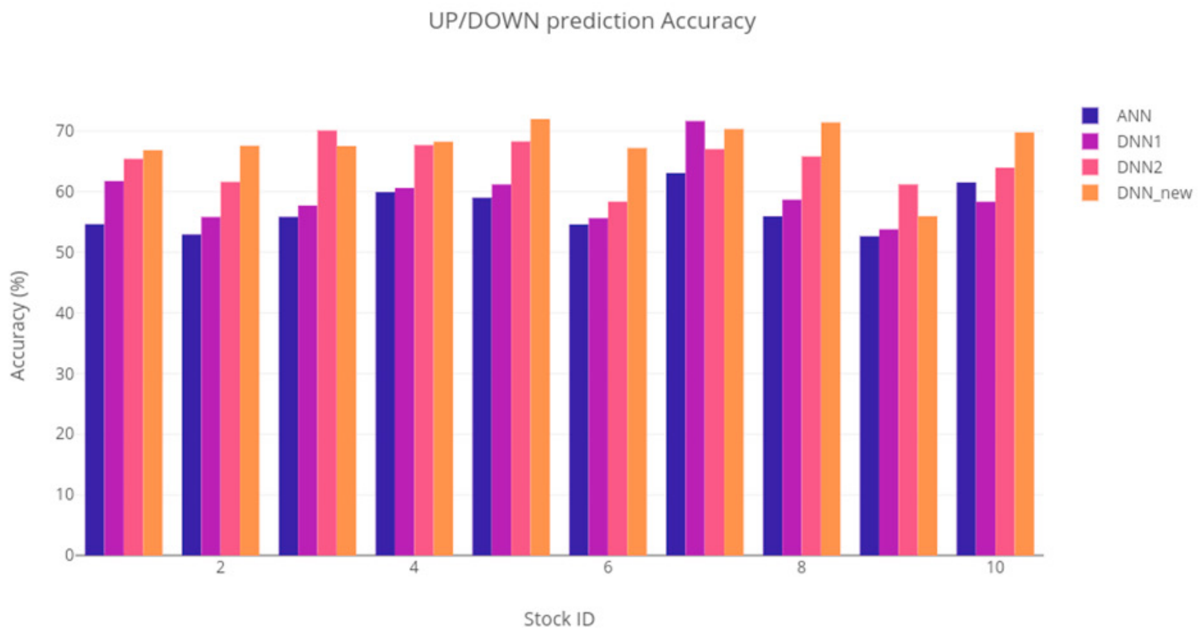

Fig. 3. UP/DOWN prediction accuracy

5. Conclusion

In this paper, we designed a deep neural network to predict the next return of a given stock in a high frequency context. This design is an improvement to previous works in terms of inputs selection and information loss, the proposed network learns from both stock past returns and other stocks returns. We evaluated the performance by mean squared error and up/down accuracy; our experimental studies show that the new DNN performs best in most cases compared to artificial neural networks and recent DNN designs for stock return prediction. In future works, we focus on learning from financial news, reports and moods since they have high correlation with stock prices and market status.

References

1. Arévalo, Andrés, Jaime Niño, German Hernández, and Javier Sandoval. "High-frequency trading strategy based on deep neural networks." In International conference on intelligent computing, pp. 424-436. Springer, Cham, 2016.
2. Patel, Jigar, Sahil Shah, Priyank Thakkar, and K. Kotecha. "Predicting stock market index using fusion of machine learning techniques." Expert Systems with Applications 42, no. 4 (2015): 2162-2172.
3. Ding, Xiao, Yue Zhang, Ting Liu, and Junwen Duan. "Deep learning for event-driven stock prediction." In Ijcai, pp. 2327-2333. 2015.
4. Chong, Eunsuk, Chulwoo Han, and Frank C. Park. "Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies." Expert Systems with Applications 83 (2017): 187-205.
5. Chen, Hao, Keli Xiao, Jinwen Sun, and Song Wu. "A double-layer neural network framework for high-frequency forecasting." ACM Transactions on Management Information Systems (TMIS) 7, no. 4 (2017): 11.
6. Granger, Clive Wiliam John, and Oskar Morgenstern. Predictability of stock market prices. Heath Lexington Books, 1970.
7. Marszałek, A., and T. Burczyński. "Modeling and forecasting financial time series with ordered fuzzy candlesticks." Information sciences 273 (2014): 144-155.
8. Mills, Terence C., and Raphael N. Markellos. The econometric modelling of financial time series. Cambridge University Press, 2008.
9. Kimoto, Takashi, Kazuo Asakawa, Morio Yoda, and Masakazu Takeoka. "Stock market prediction system with modular neural networks." In Neural Networks, 1990., 1990 IJCNN International Joint Conference on, pp. 1-6. IEEE, 1990.
10. Enke, David, and Suraphan Thawornwong. "The use of data mining and neural networks for forecasting stock market returns." Expert Systems with applications 29, no. 4 (2005): 927-940.
11. Wang, Jie, and Jun Wang. "Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks." Neurocomputing 156 (2015): 68-78.
12. Moghaddam, Amin Hedayati, Moein Hedayati Moghaddam, and Morteza Esfandyari. "Stock market index prediction using artificial neural network." Journal of Economics, Finance and Administrative Science 21, no. 41 (2016): 89-93.
13. Schalkoff, Robert J. Artificial neural networks. Vol. 1. New York: McGraw-Hill, 1997.
14. Bollen, Johan, Huina Mao, and Xiaojun Zeng. "Twitter mood predicts the stock market." Journal of computational science 2, no. 1 (2011): 1-8.
15. Chen, An-Sing, Mark T. Leung, and Hazem Daouk. "Application of neural networks to an emerging financial market: forecasting and trading the Taiwan Stock Index." Computers & Operations Research 30, no. 6 (2003): 901-923.
16. Zhang, Yudong, and Lenan Wu. "Stock market prediction of S&P 500 via combination of improved BCO approach and BP neural network." Expert systems with applications 36, no. 5 (2009): 8849-8854.
17. LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." nature 521, no. 7553 (2015): 436.
18. Sun, Yi, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. "Deep learning face representation by joint identification-verification." In Advances in neural information processing systems, pp. 1988-1996. 2014.
19. Liu, Ziwei, Ping Luo, Xiaogang Wang, and Xiaoou Tang. "Deep learning face attributes in the wild." In Proceedings of the IEEE International Conference on Computer Vision, pp. 3730-3738. 2015.
20. Parkhi, Omkar M., Andrea Vedaldi, and Andrew Zisserman. "Deep Face Recognition." In BMVC, vol. 1, no. 3, p. 6. 2015.
21. Yoshihara, Akira, Kazuki Fujikawa, Kazuhiro Seki, and Kuniaki Uehara. "Predicting stock market trends by recurrent deep neural networks." In Pacific rim international conference on artificial intelligence, pp. 759-769. Springer, Cham, 2014.
22. Ding, Xiao, Yue Zhang, Ting Liu, and Junwen Duan. "Deep learning for event-driven stock prediction." In Ijcai, pp. 2327-2333. 2015.
23. Glorot, Xavier, and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks." In Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp. 249-256. 2010.
24. Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." arXiv preprint arXiv:1502.03167 (2015).