Proceedings of the 36th Chinese Control Conference
July 26-28, 2017, Dalian, China

Stock Market Index Prediction Using Deep Neural Network Ensemble

Bing Yang[1], Zi-Jia Gong[1,*], Wenqi Yang[2]

1. School of Mathematics and Statistics, Shandong University, Weihai, Weihai 264209, China
E-mail: bingyang@sdu.edu.cn, zjgong@mail.sdu.edu.cn

2. School of Mathematics, Shandong University, Jinan, 250001, China
E-mail: bert1204@126.com

Abstract: In this paper, we put forward deep neural network ensemble to model and predict Chinese stock market index (including Shanghai composite index and SZSE component index), based on the input indices of recent days. A set of component networks are trained by historical data for this task, where *Backpropagation* and *Adam* algorithm are used to train each network efficiently. *Bagging* approach combines these component networks to generate ensemble, which reduces the generalization error. Indices of test examples are predicted with the model, and the trend predictions is calculated based on the predicted indices. Finally, relative errors between actual indices and predicted indices, as well as accuracy of trend predictions are computed to measure the performance of predictions. It turns out that the model proposed in this paper can partially model and predict the Chinese stock market. The accuracy of trend predictions of the daily barycenter, *high*, *low* are 71.34%, 74.15%, 74.15% respectively for Shanghai composite index and 75.95%, 73.95%, 72.34% respectively for SZSE component index. But the predictions on *close* are unsatisfactory.

Key Words: neural network, machine learning, ensemble learning, stock index, time series forecasting

1 Introduction

During the past few decades, forecasting of price fluctuations for stock markets have been investigated via various artificial intelligence methods[1–3]. According to the *efficient market hypothesis* [5–7], stock prices rapidly adjust to new information by the time the information becomes public knowledge, so that prediction of stock market movements is impossible[8].

However, amount of research is being done on the application of neural networks to stock markets and points out that prediction of stock market movements is possible because there are the future information in the historical price fluctuations for stock markets and the stock market isn't efficient[9–13].

Some of the applications include prediction of IBM daily stock prices[9], a trading system based on prediction of the daily S&P 500 index[10], short term trend prediction using dual-module networks[11], weekly index prediction[12], monthly index prediction using radial basis functions[13] etc. Qiu et al.[4] used the 3-layer feed-forward neural network to predict the returns of the Nikkei 225 indices in Japan and pointed out that the elaborate feature selection and heuristic methods (genetic algorithm, simulated annealing) for training neural networks help improve prediction accuracy. Liu and Wang [14] also used a 3-layer feed-forward neural network to predict the closing indices of Shanghai stock market in China, where the activation function of the nth hidden neuron is the n-order Legendre polynomial, and the network is trained with unequal losses on different training examples, and indicated that the elaborately designed loss function enhances the precision of prediction. Dai et al. [17] used the nonlinear independent component analysis to select features and the 3-layer feed-forward neural network to predict the daily Nikkei 225 closing cash index and Shanghai B-Share stock index closing price. Bisoi and Dash [15] used the dynamic neural network, a variation of recurrent neural network, to predict the trend of closing price of several stock markets. Xi et al. [16] proposed a kind of improved RBF neural networks to predict the closing indices of Chinese Shanghai stock market.

In this paper, we put forward the deep neural network ensemble to model and predict Chinese stock market index(including Shanghai composite index and SZSE component index). The multi-layer feedforward deep neural networks trained by historical data performs as a supervised regressor to predict the daily *high*, *low* and *close*, based on the input of past several days' indices. The conventional *sigmoid* or *tanh* activation function is replaced with the recently-proposed *Leaky Rectified Linear Unit*(*LReLU*) [18] in neural networks, which enables deep network to be trained efficiently. The backpropagation [19, 20] algorithm and Adam [21] optimization algorithm, which accelerates the training speed enormously, are applied in our neural networks.

A finite set of these deep neural networks (each neural network in the set is called a component neural network) are trained to form a neural network ensemble [22]. Each component neural network is trained with a component training set *bootstrapping* [23] from the original training set, and the average of the predictions of all component neural networks is the final output. That's called *Bagging* [24] approach, a paradigm of ensemble learning, considered to reduce the generalization error significantly.

The historical data of Shanghai composite index and SZSE component index of the past 26 years is collected and preprocessed. Each dataset is divided into training period and test period. The training period is used to train the neural network ensembles and the test period is used to test the performance of the predictions. On test period, the relative error of predictions are computed, as well as the accuracy of trend predictions of index change, which are used to measure the performance of predictions.

The rest of this paper is organized as follows. In section

*Corresponding author.

2, the collecting and preprocessing of data is introduced. In section 3, model of deep neural network ensemble is presented in detail. In section 4, the implementation and result of our empirical study is reported. Finally, section 5 concludes this paper.

2 Data and Preprocessing

The historical datasets of Shanghai composite index and SZSE component index are used in this study, each dataset is a series of daily open, *high*, *low*, *close* and trading volume. The dataset of Shanghai composite index covers the horizon from December 19, 1990 to October 14, 2016, and the dataset of SZSE component index covers from April 3, 1991 to October 14, 2016.

Each dataset is divided into two periods. For dataset of Shanghai composite index, the first period runs from December 19, 1990 to September 22, 2014 and the second period runs from September 23, 2014 to October 14, 2016. And for the dataset of SZSE component index, the first period runs from April 3, 1991 to September 22, 2014 and the second period runs from September 23, 2014 to October 14, 2016. The first period of each dataset, the training period, is used for training neural networks. The second period, the testing period, containing 500 days, is used for test the performance of predictions.

The indices of open, *high*, *low* and *close* (denoted as O, H, L and C respectively) remain as the raw form without normalization. The trading volumes are taken logarithm (natural logarithm) because of the vast differences of their orders of magnitude. V denotes the logarithmic trading volumes.

3 Model of Deep Neural Network Ensemble

3.1 Model

This study aims to model and predict the next day's H, L and C based on the indices and trading volumes of recent days. The input of the model is O, H, L , C , V and $H - L$ of the recent N days, where N is a hyperparameter denotes the horizon of the input window. The output is the next day's H, L and C. For example, suppose $N = 20$, if we input the O, H, L , C , V and $H - L$ of first 20 days to our model, then we expect the output to be the predictions of H, L and C of the 21st day, as Table 1 shows.

Table 1: input window and output of the model

date	serial number	O	H	L	C	$H-L$	V
1990/12/19	1	O_1	H_1	L_1	C_1	H_1-L_1	V_1
1990/12/20	2	O_2	H_2	L_2	C_2	H_2-L_2	V_2
1990/12/21	3	O_3	H_3	L_3	C_3	H_3-L_3	V_3
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1991/1/16	20	O_{20}	H_{20}	L_{20}	C_{20}	$H_{20}-L_{20}$	V_{20}
1991/1/17	21	O_{21}	H_{21}	L_{21}	C_{21}	$H_{21}-L_{21}$	V_{21}
1991/1/18	22	O_{22}	H_{22}	L_{22}	C_{22}	$H_{22}-L_{22}$	V_{22}
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
2014/9/22	d	O_d	H_d	L_d	C_d	H_d-L_d	V_d

Table 1 shows the data of training period of the dataset of Shanghai composite index. In the table, O_i, H_i, L_i, C_i and V_i denote the open, *high*, *low*, *close* and logarithmic trading volume of the ith day respectively. d denotes the number of days in the training period, and $d = 5814$. Suppose $N = 20$, then the box area denotes the input window of the model and the shade area denotes the output.

Three deep neural network ensembles are trained separately for this task. They receive the same input window as Table 1 shows, but the first predicts H, the second predicts L and the third predicts C. After input window in Table 1 as well as corresponding output slides down from the top of table of training period to the bottom, each time for one day, all the training examples are obtained. For convenient, we reshape each input window into a column vector. The input of ith training example is denoted $x^{(i)}$. Suppose the number of training examples is m, Then three training sets for three neural network ensembles can be denoted as $\{(x^{(i)}, H_{i+N})_{i=1}^m\}$, $\{(x^{(i)}, L_{i+N})_{i=1}^m\}$ and $\{(x^{(i)}, C_{i+N})_{i=1}^m\}$. The three test sets can be generated with the test period in the same way.

In this paper, models to predict Shanghai composite index and SZSE component index are trained separately with two datasets.

3.2 Deep Neural Networks

Deep neural network is a machine-learning paradigm for modeling complex nonlinear mappings between input and output, in which the internal parameters are updated iteratively to make the given inputs fit with target outputs. Unlike most conventional machine-learning paradigms, it can take the raw data as input directly and learn multiple levels of data representations automatically with multiple layers, requiring less feature-extracting work by hand and taking advantage of large data and computational resources [25].

In this paper, multi-layer feedforward neural networks are used. In a feedforward network, each neuron of each layer is connected with all neurons of previous layer. Each layer in this neural network receives the output of previous layer, performs a linear mapping and optionally follows it with a non-linear activation function. Because the outputs are all real values, the neural network performs as a regressor, so the linear activation function $g(z) = z$ is used in the output layer. The input and output of the neural network are denoted as x and $y^*(x)$ respectively, and the number of neurons of the first layer up to the Lth layer are denoted as $s_1, s_2, ..., s_L$, then the feed-forward procedure is

$$a^{(1)} = g\left(W^{(1)}x + b^{(1)}\right)$$

$$a^{(l)} = g\left(W^{(l)}a^{(l-1)} + b^{(l)}\right), \quad for\ l = 2, ..., L-1$$

$$y^*(x) = z^{(L)} = W^{(L)}a^{(L-1)} + b^{(L-1)}, \tag{1}$$

where $a^{(l)}$ denotes the activations of lth layer, and all $a^{(l)}$ $for\ l = 2, ..., L-1$ are column vectors, $a^{(l)} \in \mathbb{R}^{s_l}$; $W^{(l)}$ and $b^{(l)}$ denote the weight and bias of neurons in lth layer, $W^{(l)} \in \mathbb{R}^{s_l \times s_{l-1}}$, $b^{(l)} \in \mathbb{R}^{s_l}$, and all $W^{(l)}$ and $b^{(l)}$ are learnable parameters of the neural network; g denotes the activation function, $g : \mathbb{R} \to \mathbb{R}$, which operates on every component of the argument vector.

In this paper, dimensions of input x is the size of input window in Table 1, so $x \in \mathbb{R}^{6N}$. The output $y^* \in \mathbb{R}$, i.e $s_L = 1$. The number of neurons in each hidden layer is 11. The *Leaky Rectified Linear Unit* (*LReLu*) is used as activa-

tion function,

$$g(z) = \begin{cases} z & z > 0, \\ 0.05z & \text{otherwise}, \end{cases} \quad (2)$$

which is frequently-used in deep neural networks, leads to faster learning speed and less vanishing gradient than traditional activation functions such as *sigmoid* and *tanh* [18].

For every input training example $x^{(i)}$, we expect the output of the neural network $y^*(x^{(i)})$ to be as close to supervised output $y^{(i)}$ as possible. Then the least-square loss function for training example $(x^{(i)}, y^{(i)})$ is

$$\mathcal{L}(W, b; x^{(i)}, y^{(i)}) = \frac{1}{2}\left(y^*(x^{(i)}) - y^{(i)}\right)^2, \quad (3)$$

where $W = \{W^{(1)}, ..., W^{(L)}\}$ and $b = \{b^{(1)}, ..., b^{(L)}\}$, then (W, b) denotes all parameters of the network.

Then, considering the entire training set $D = \{(x^{(i)}, y^{(i)})_{i=1}^m\}$, the loss function of the network is

$$\mathcal{L}(W, b; D) = \frac{1}{m}\sum_{i=1}^{m} L(W, b; x^{(i)}, y^{(i)}) + \frac{\lambda}{2}R(W) \quad (4)$$

where m is the number of training examples in the training set, $R(W)$ is the regularization term, λ is the weight of the regularization term. λ is a hyperparameter.

The regularization term tends to alleviate overfitting problem, the $L2$ penalty of all weight W is used in this study,

$$R(W) = \sum_{l=1}^{L} \|W^{(l)}\|_F^2 = \sum_{l=1}^{L}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l-1}}\left(W_{ij}^{(l)}\right)^2 \quad (5)$$

where s_0 denotes the dimension of input layer, i.e. $6N$.

3.3 Training

The training process of the network, is also the process to optimize the loss function of the network, i.e searching the optimal $(\hat{W}, \hat{b})$ that minimize the loss function

$$(\hat{W}, \hat{b}) = \arg\min_{W,b} \mathcal{L}(W, b; D) \quad (6)$$

The most widely-used methods to train networks are gradient-based methods [25], which require the derivatives of the loss function with respect to all parameters (W, b). *Backpropagation* algorithm is used to calculate these derivatives $\nabla_{(W,b)}\mathcal{L}(W, b; D)$ analytically with chain rule. *Backpropagation* was proposed by several researchers independently in 1970s and 1980s [20, 25], which accelerates the training speed enormously, considered to be a milestone in the evolution of neural network. Details of *Backpropagation* algorithm can be found in [20].

Then, all the parameters are initialized as random numbers and the Adam algorithm is used to update the parameters.

Every weight parameter $W_{ij}^{(l)}$ is randomly sampled from the Gaussian distuibution $\mathcal{N}(0, \frac{2}{s_{l-1}})$, and all bias $b^{(l)}$ are initialized as zero vectors. This initialization method was proposed by He el al. [26].

The Adam algorithm was proposed by Kingma and Ba [21] recently. It's a variation of stochastic gradient decent, where momentum is also considered, and the step size could be adjusted automatically as the training process goes deeper. The Adam is widely-used in the training of deep neural networks.

A simplified version of Adam algorithm is used in this paper, as shown in Algorithm 1, where $\circ$ denotes the Hadamard

Algorithm 1 Adam algorithm

input: number of epochs n_e, batch size b_s, initialized parameter vector (W, b)
initialize $M = \mathbf{0}$ and $V = \mathbf{0}$

repeat
  repeat
    randomly sample a subset D' from D satisfying $|D'| = b_s$
    compute $\nabla_{(W,b)}\mathcal{L}(W, b; D')$ using backpropagation algorithm
    $M := \beta_1 M + (1 - \beta_1)\nabla_{(W,b)}\mathcal{L}(W, b; D')$
    $V := \beta_2 V + (1 - \beta_2)\left(\nabla_{(W,b)}\mathcal{L}(W, b; D')\right)^{\circ 2}$
    $(W, b) := (W, b) - \alpha M \circ \left(V^{\circ \frac{1}{2}} + \epsilon\right)^{\circ -1}$
  until for $\left[\frac{|D|}{b_s}\right]$ times
until for n_e epochs

operations, which performs element-wise operations on the vector; $[\cdot]$ denotes the round operation to the nearest integer; α is the step size, a hyperparameter; β_1, β_2 and ϵ are all hyperparameters, the setting of them is $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$ in this study. And we set $\alpha = 10^{-4}$, $b = 1000$ via the results of some shallow attempts.

A large n_e is used in this study, i.e. $n_e = 2 \times 10^5$. A validation set D_v is used for selecting best parameters (W, b) during training, which is a subset of training set D. The network is trained with set $D - D_v$, and the loss on validation set $\mathcal{L}(W, b; D_v)$ is computed after each epoch, and the the minimum $\mathcal{L}(W, b; D_v)$ and corresponding parameters (W, b) are tracked. The final parameters are (W, b) that minimize $\mathcal{L}(W, b; D_v)$ during all Adam epochs.

3.4 Neural Networks Ensemble

Neural network ensemble is an ensemble-learning paradigm where a finite set of neural networks are trained and then the predictions of these neural networks are combined to produce the final output. It's proposed by Hansen and Salamon [22] in 1993 firstly, which improves the generalization ability of the neural network significantly.

Bagging approach is used in this study, where several component training sets from the original training set are generated with *bootstrapping*, and then each component neural network is trained on each component training set, and the average of the predictions of all the component neural networks is the final output. Bagging was proposed by Breiman [24] in 1990s.

The *bootstrapping* [23] is a method to randomly sample a new training set from the original one. If we want to sample a set D^* from D, then the bootstrapping works as follows: randomly choose a example from D, copy it into D^*, and then put it back to D; and this step repeated for $|D|$ times. Then we can get the set D^* that have some duplicated examples.

In this paper, not only the training sets of these component neural networks are different, but also the regu-

Table 2: average relative error on test set

	Shanghai composite index	SZSE component index
$\overline{E_H}$	0.8483%	0.9511%
$\overline{E_L}$	1.018%	1.162%
$\overline{E_C}$	1.427%	1.667%

larization coefficient λ and the number of layers L. For each component neural network, L is randomly chosen in $\{1, 2, 3, 4, 5, 6\}$; λ is sampled from a logspace-uniform distribution of $\left[10^{-6}, 1\right]$, i.e. sampling a random number c from uniform distribution $U(-6, 0)$ and then let $\lambda = 10^c$.

Then, the model of deep neural network ensemble, which contains T component networks(we choose $T = 10$ in this paper), are trained as Algorithm 2.

Algorithm 2 Training of deep neural network ensemble

input: training set D, number of component networks T

1: **for** $i = 1, 2, ..., T$ **do**
2: bootstrapping a component training set D^* from D
3: randomly choose a L_i from $\{1, 2, 3, 4, 5, 6\}$
4: randomly sample a λ_i from logspace-uniform distribution of $\left[10^{-6}, 1\right]$
5: train a L_i-layer component neural network $y^{(*,i)}$ with D^* and use $D - D^*$ as validation set, λ_i as regularization coefficient
6: **end for**
output: the model of deep neural network ensemble $Y(x) = \frac{1}{T}\sum_{i=1}^{T} y^{(*,i)}(x)$

4 Result

The models to predict Shanghai composite index and SZSE component index are trained separately with 2 corresponding datasets. For each dataset, three deep neural network ensembles is trained separately to predict H, L and C respectively, then they predict 500 test examples in the test sets. The trend predictions can be derived by the predicted indices. Two statistics are used to measure the performance of predictions: relative error and accuracy of trend predictions.

4.1 Relative Error

The relative error, denoted as E, is computed as follows,

$$E = \frac{|y^* - y|}{y} \times 100\% \tag{7}$$

where y is the actual index, and the y^* is the output of neural network ensemble (i.e. the predicted index), y could be one of H, L or C.

The average relative errors on H, L and C are denoted as $\overline{E}_H$, $\overline{E}_L$ and $\overline{E}_C$ respectively. The result is shown in Table 2, and the histogram of relative errors on all test examples of Shanghai composite index is shown in Fig. 1.

As shown in Fig. 1, the tails of the histograms indicate that there are less test examples of which the relative errors of predictions are high. In order to observe that where the predictions with high relative errors occur, we plot all predicted H, L, C and actual H, L, C of Shanghai composite

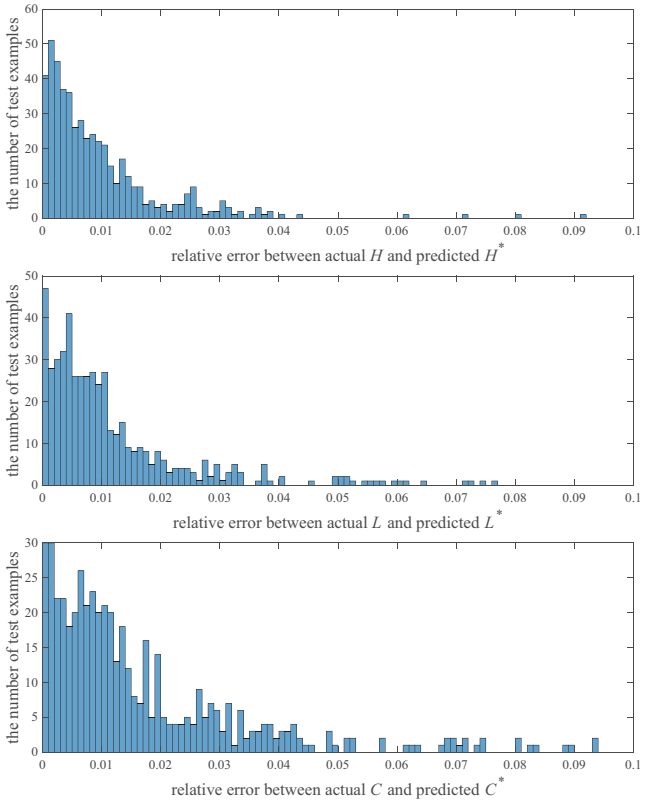

Fig. 1: the distributions of relative errors of predictions on the test set of Shanghai composite index

Table 3: average relative error on test set and on subsampled examples with top 10% with highest rate of daily change

	Shanghai composite index	SZSE component index
$\overline{E_H}$	0.8483%	0.9511%
$\overline{Es_H}$	2.39%	2.62%
$\overline{E_L}$	1.018%	1.162%
$\overline{Es_L}$	3.37%	3.56%
$\overline{E_C}$	1.427%	1.667%
$\overline{Es_C}$	4.96%	5.52%

index, along with the relative errors between them, as Fig. 2 shows.

Fig. 2 shows that the predictions with high relative errors are likely to occur when the indices of stock market fluctuate fiercely. To verify this observation, the daily rate of change is used to measure fluctuation level of indices:

$$Rc_i = \frac{|y_i - y_{i-1}|}{y_{i-1}} \times 100\% \tag{8}$$

where Rc_i denotes the rate of change of the ith day in test period of data, y_i is the actual index of the ith day. We sort all the test examples by their rate of change, and subsample the top 10% of test examples with highest rate of change. The subsampled examples are also emphasized in Fig. 2 with red line. And the average relative error on subsampled samples (denoted as $\overline{Es}$) and that on the entire test set are compared, as shown in Table 3, where the statistics of SZSE component index are calculated following the same procedure.

Table 3 indicates that the average relative errors on subsampled samples with top 10% highest rate of daily change are much higher than that of the entire test set, which

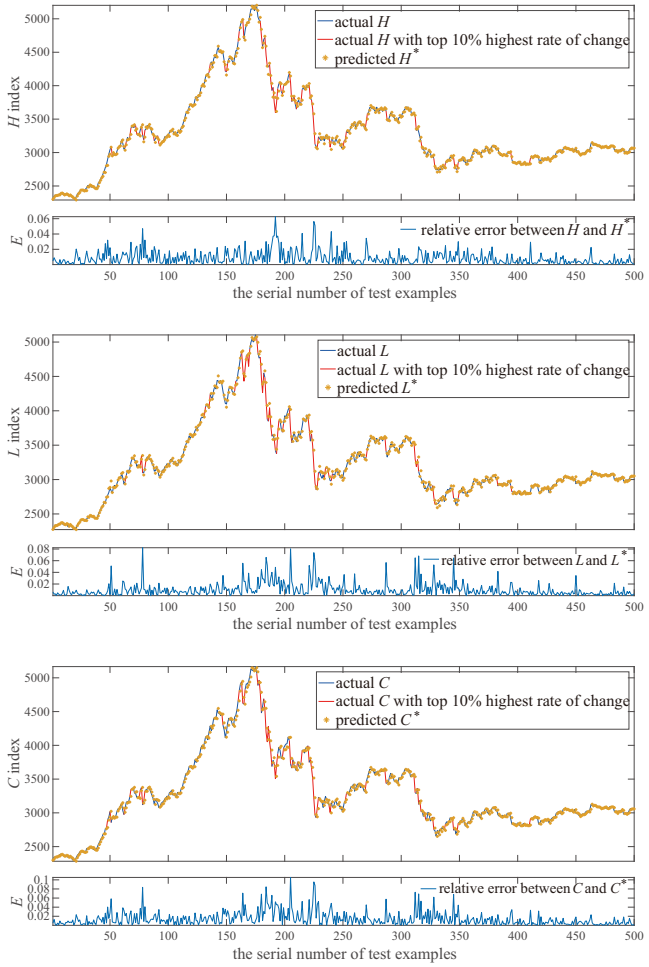

Fig. 2: The predicted H, L, C, actual H, L, C, and their relative errors of test examples of Shanghai composite index

confirms the observation that predictions with high relative errors occur when the indices of stock market fluctuate fiercely.

4.2 Accuracy of Trend Predictions

If we input ith test example to the model, then we can get the predicted H^*_{i+N}, L^*_{i+N} and C^*_{i+N}. The predicted trend of index H is $sign(H^*_{i+N} - H_{i+N-1})$, where 1 means increasing and -1 means decreasing. The actual trend of index H is $sign(H_{i+N} - H_{i+N-1})$. Then we can calculate the accuracy of trend predictions of H based on the predicted H^*_{i+N},

$$A_H = \frac{1}{m'} \sum_{i=1}^{m'} \mathbb{I}\left\{sign(H_{i+N} - H_{i+N-1}) = sign(H^*_{i+N} - H_{i+N-1})\right\} \tag{9}$$

where A_H denotes the accuracy of trend predictions of H, m' is the number of test examples. The accuracy of trend predictions of L and C can be calculated in the same way, denoted as A_L and A_C respectively.

The barycenter of daily index, defined as $(H + L)/2$, of which the accuracy of trend predictions is also computed, denoted as A_B.

$$A_B = \frac{1}{m'} \sum_{i=1}^{m'} \mathbb{I}\left\{sign(B_{i+N} - B_{i+N-1}) = sign(B^*_{i+N} - B_{i+N-1})\right\} \tag{10}$$

where $B^*_i = (H^*_i + L^*_i)/2$ and $B_i = (H_i + L_i)/2$.

To prevent the interference of class imbalance(it turns out that two class is relatively balance), the *precision* and *recall* of predictions on trend are also computed, denoted as Pr and Re respectively,

$$Pr_H = \frac{\sum_{i=1}^{m'} \mathbb{I}\left\{sign(H_{i+N} - H_{i+N-1}) = 1, sign(H^*_{i+N} - H_{i+N-1}) = 1\right\}}{\sum_{i=1}^{m'} \mathbb{I}\left\{sign(H^*_{i+N} - H_{i+N-1}) = 1\right\}} \tag{11}$$

$$Re_H = \frac{\sum_{i=1}^{m'} \mathbb{I}\left\{sign(H_{i+N} - H_{i+N-1}) = 1, sign(H^*_{i+N} - H_{i+N-1}) = 1\right\}}{\sum_{i=1}^{m'} \mathbb{I}\{sign(H_{i+N} - H_{i+N-1}) = 1\}} \tag{12}$$

where the precision and recall of predictions on trend of H are computed, and that of L, C and B can be computed in the same way.

The accuracy of trend predictions is shown as Table 3. It turns out that the the model can predict the trend of the daily barycenter B, H and L accurately in most cases. But the accuracy of trend predictions on C is only about 49.9% for Shanghai composite index and 51.1% SZSE component index, which random guess may also achieve. The index C is instantaneous, apt to be manipulated, which may give rise to this result.

Table 4: accuracy of trend predictions

	Shanghai composite index	SZSE component index
A_H	74.15%	73.95%
Pr_H	78.17%	78.67%
Re_H	76.82%	76.53%
A_L	74.15%	72.34%
Pr_L	77.43%	81.47%
Re_L	73.70%	66.55%
A_C	51.50%	53.31%
Pr_C	59.72%	59.47%
Re_C	45.42%	48.91%
A_B	71.34%	75.95%
Pr_B	75.56%	81.10%
Re_B	72.60%	74.10%

Table 5: details of all component networks in model of Shanghai composite index

	E_H	E_L	E_C
net1	0.88%	1.08%	1.46%
net2	0.90%	1.05%	1.45%
net3	0.90%	1.02%	1.47%
net4	0.89%	1.09%	1.49%
net5	0.87%	1.06%	1.44%
net6	0.88%	1.06%	1.47%
net7	0.93%	1.11%	1.51%
net8	0.91%	1.03%	1.45%
net9	0.92%	1.06%	1.47%
net10	0.87%	1.04%	1.48%

4.3 The Effect of Ensemble

The details of all these component networks of the model of Shanghai composite index is showed in Table 5, including relative errors on predictions of H, L and C in test set. It turns out that the relative error of the ensemble shown in Table 2 is lower than that of all its component networks. That means the ensemble reduces the generalization error.

5 Conclusions

In this study, deep neural network ensemble is used to predict Chinese stock market index. The trend predictions is

computed based on the predicted indices. The relative errors of predicted indices and actual indices, as well as accuracy of trend predictions, are calculated to measure the performance of predictions. From the result, the following conclusions are derived:

- We conclude that Chinese stock market is partially predictable based on the result of this empirical study. The performance of predictions on Shanghai composite index and SZSE component index are similar. For trend predictions, the accuracy of trend predictions of the daily barycenter, *high*, *low* are 71.34%, 74.15%, 74.15% for Shanghai composite index and 75.95%, 73.95%, 72.34% for SZSE component index, and even higher *precision* and *recall* are achieved. These high-accuracy trend predictions are owed to high-accuracy index predictions. For index predictions, the relative error of *high*, *low* are 0.8483%, 1.018% for Shanghai composite index and 0.9511%, 1.162% for SZSE component index.
- The predictions on *close* indices are unsatisfactory. The accuracy of trend predictions on close are 49.9% for Shanghai composite index and 51.1% for SZSE component index, which random prediction may also achieve. The relative errors are 1.427% for Shanghai composite index and 1.667% for SZSE component index, which are much higher than that of *high* and *low*.
- The relative errors of predictions on *high*, *low* and *close* are higher when the market index fluctuate fiercely.
- The relative error of an ensemble is lower than that of its all component networks. That means the ensemble reduces the generalization error.

References

[1] De Gooijer, J. G., & Hyndman, R. J. (2006). 25 years of time series forecasting. *International journal of forecasting*, 22(3), 443-473.

[2] Chan, M. C., Wong, C. C., & Lam, C. C. (2000). Financial time series forecasting by neural network using conjugate gradient learning algorithm and multiple linear regression weight initialization. In *Computing in Economics and Finance*(Vol. 61).

[3] Kulkarni, A. S. (1996). Application of neural networks to stock market prediction. *Technical Report*.

[4] Qiu, M., Song, Y., & Akagi, F. (2016). Application of artificial neural network for the prediction of stock market returns: The case of the Japanese stock market. *Chaos, Solitons & Fractals*, 85, 1-7.

[5] HellstrSom, T., & HolmstrSom, K. (1998). Predicting the stock market.

[6] Magdon-Ismail, M., Nicholson, A., & Abu-Mostafa, Y. S. (1998). Financial markets: very noisy information processing. *Proceedings of the IEEE*, 86(11), 2184-2195.

[7] Malkiel, B. G., & Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The journal of Finance*, 25(2), 383-417.

[8] Azoff, E. M. (1994). Neural network time series forecasting of financial markets. *International Journal of Forecasting*(4), 601-602.

[9] White, H. (1988). Economic prediction using neural networks: the case of IBM daily stock returns. IEEE International Conference on Neural Networks (Vol.2, pp.451-458 vol.2).

[10] Chenowethl, T., Obradovic, Z., & Lee, S. (1995). Technical trading rules as a prior knowledge to a neural networks prediction system for the S&P 500 index. *Northcon 95. I Eee Technical Applications Conference and Workshops Northcon* (Vol.82, pp.4668-4669).

[11] Jang, G. S., Lai, F., Jiang, B. W., Pan, C. C., & Chien, L. H. (1991). An intelligent stock portfolio management system based on short-term trend prediction using dual-module neural networks. In *Proc. of the 1991 International Conference on Artificial Neural Networks* (Vol. 1, pp. 447-52).

[12] Freisleben, B. (1992, June). Stock market prediction with backpropagation networks. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems* (pp. 451-460). Springer Berlin Heidelberg.

[13] Komo, D., Chang, C. I., & Ko, H. (1994, March). Stock market index prediction using neural networks. In *SPIE's International Symposium on Optical Engineering and Photonics in Aerospace Sensing* (pp. 516-526). International Society for Optics and Photonics.

[14] Liu, F., & Wang, J. (2012). Fluctuation prediction of stock market index by Legendre neural network with random time strength function. *Neurocomputing*, 83, 12-21.

[15] Bisoi, R., & Dash, P. K. (2014). A hybrid evolutionary dynamic neural network for stock market trend analysis and prediction using unscented Kalman filter. *Applied Soft Computing*, 19, 41-56.

[16] Xi, L., Muzhou, H., Lee, M. H., Li, J., Wei, D., Hai, H., & Wu, Y. (2014). A new constructive neural network method for noise processing and its application on stock market prediction. *Applied Soft Computing*, 15, 57-66.

[17] Dai, W., Wu, J. Y., & Lu, C. J. (2012). Combining nonlinear independent component analysis and neural network for the prediction of Asian stock market indexes. *Expert systems with applications*, 39(4), 4444-4452.

[18] Chen, K., Zhou, Y., & Dai, F. (2015). A LSTM-based method for stock returns prediction: A case study of China stock market. *IEEE International Conference on Big Data* (pp.2823-2824). IEEE.

[19] Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013, June). Rectifier nonlinearities improve neural network acoustic models. In *Proc. ICML* (Vol. 30, No. 1).

[20] Lecun, Y., Bottou, L., Orr, G. B., & Mller, K. R. (2000). Efficient backprop. *Neural Networks Tricks of the Trade*, 1524(1), 9-50.

[21] Williams, D. R. G. H. R., & Hinton, G. E. (1986). Learning representations by back-propagating errors. *Nature*, 323, 533-536.

[22] Kingma, D., & Ba, J. (2015). Adam: a method for stochastic optimization. *Computer Science*.

[23] Hansen, L. K., & Salamon, P. (1990). Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12, 993-1001.

[24] Efron, B., & Tibshirani, R. J. (1994). *An introduction to the bootstrap*. CRC press.

[25] Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2), 123-140.

[26] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436-444.

[27] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1026-1034).