Neurocomputing 167 (2015) 243–253

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Forecasting exchange rate using deep belief networks and conjugate gradient method

Furao Shen*, Jing Chao, Jinxi Zhao

National Key Laboratory for Novel Software Technology, and Department of Computer Science and Technology, Nanjing University, China

ARTICLE INFO

Article history:
Received 3 April 2013
Received in revised form
3 December 2014
Accepted 27 April 2015
Communicated by M.T. Manry
Available online 5 May 2015

Keywords:
Deep belief networks
Exchange rate forecasting
Conjugate gradient
Continuous restricted Boltmann machines

ABSTRACT

Forecasting exchange rates is an important financial problem. In this paper, an improved deep belief network (DBN) is proposed for forecasting exchange rates. By using continuous restricted Boltzmann machines (CRBMs) to construct a DBN, we update the classical DBN to model continuous data. The structure of DBN is optimally determined through experiments for application in exchange rates forecasting. Also, conjugate gradient method is applied to accelerate the learning for DBN. In the experiments, three exchange rate series are tested and six evaluation criteria are adopted to evaluate the performance of the proposed method. Comparison with typical forecasting methods such as feed forward neural network (FFNN) shows that the proposed method is applicable to the prediction of foreign exchange rate and works better than traditional methods.

1. Introduction

The exchange rate market is a multivariable nonlinear system, in which the factors' mutuality is quite complex. Hence, forecasting exchange rate is an important and challenging task, and it has attracted many researchers' attention. For many years researchers have used linear techniques since it is very simple. However, it is not as helpful as we expected, because of the linear unpredictability of exchange rate.

Neural network has extensive adaptability and ability of learning, thus it has been used to model and control multivariable nonlinear systems. There are two great features which make it attractive in exchange rate forecasting. First, neural network has general nonlinear function mapping capability which can approximate any continuous function with arbitrarily desired accuracy [1,2]. Hence, it is capable of solving many complex problems. Second, neural network is a nonparametric data-driven model and it does not need restrictive assumption on the underlying process from which data are generated. Because of this feature, it is less susceptible to model misspecification problem than most parametric nonlinear methods. This is an important advantage since exchange rate does not show a specific nonlinear pattern. Given the advantages of neural network, it is not surprising that neural network is a promising tool for exchange rate prediction. Since 1990s, neural networks have been widely used in economic and financial fields.

Numerous studies have shown that neural network is one of the very effective tools in exchange rate forecasting. Weigend et al. [3,4] compared the performance of neural network with that of random walk in predicting the Deutsche mark/US dollar (DEM/USD) exchange rate. They found that neural network is better than random walk model.

Feedforward neural networks (FFNNs) are the most popular neural network paradigms in the prediction of financial time series. White [5] was the first to apply NN models in the financial domain. Kuan and Liu [6] used feedforward and recurrent neural networks to predict five different exchange rates. Their findings show that neural network can improve the predictions. Panda and Narasimhan [7] conducted a comparative study among feedforward neural networks, linear autoregressive and random walk models in forecasting the Indian rupee/US dollar (INR/USD) exchange rate. They find that forecasts generated by neural network are superior to those of linear autoregressive and random walk models. Emam [8] proved that neural networks are efficient and profitable in forecasting exchange rate in particular, with feed forward back propagation.

There are also several studies proposed to combine multiple neural networks or combine neural networks with other methods. Nag and Mitra [9] used a hybrid artificial intelligence method, based on neural network and genetic algorithm for modeling daily foreign exchange rates. The results indicate superior performance of their method as compared to traditional models.

The intrinsic defect among these above-mentioned methods is that they are easy to fall into local minima, thus, they have

* Corresponding author.
E-mail addresses: frshen@nju.edu.cn (F. Shen), yoyo72-@163.com (J. Chao), jxzhao@nju.edu.cn (J. Zhao).
URL: http://cs.nju.edu.cn/rinc (F. Shen).

http://dx.doi.org/10.1016/j.neucom.2015.04.071
0925-2312/

difficulty in achieving globally optimal situation. Therefore, we hope we can find the global minima and to better forecast the exchange rates. In this paper, we adopt deep belief network (DBN) to solve this problem.

DBN is a generative neural network model with many hidden layers along with a greedy layer-wise learning algorithm [10]. The building block of a DBN is a probabilistic model called restricted Boltzmann machine (RBM). DBNs and restricted Boltzmann machines (RBMs) have already been applied successfully to solve many problems, such as classification [11,12], traffic flow prediction [13], voice activity detection [14], dimensionality-reduction [15,16], and information retrieval [17].

Conjugate gradient algorithm is an important optimization algorithm. In this algorithm, a search is performed along conjugate directions, which produces generally faster convergence than steepest descent directions. Since the learning of a DBN costs much time, we apply conjugate gradient algorithm to accelerate it.

This paper aims to present and evaluate the DBN performance as a forecasting tool on predicting exchange rates. We improved the classical DBN model by using continuous restricted Boltzmann machines (CRBMs). This improved DBN has the ability to model continuous data. In experiments, we use British pound/US dollar (GBP/USD), Indian rupee/US dollar (INR/USD), and Brazilian real/US dollar (BRL/USD) exchange rates data sets and six evaluation criteria to evaluate the performances. We also compare our method to some traditional methods such as FFNNs. The experimental results show that the proposed method is applicable to the prediction of foreign exchange rates, and it works better than traditional methods.

2. The proposed method

2.1. Time series forecasting

Forecasting exchange rate is a univariate time series forecasting problem. The input data of the network are the past, lagged observations of exchange rate and the outputs are the future values. Each input sample is composed of a moving window of fixed length along the time series.

Suppose $x_1, x_2, \ldots, x_n$ are n time-lagged observations of the exchange rate in the training set. We use a network with p input nodes and one output node, if we need the one-step-ahead forecasts. The first training sample is composed of $x_1, x_2, \ldots, x_p$ as the inputs and x_{p+1} as the output. In the second training sample, $x_2, x_3, \ldots, x_{p+1}$ are the inputs and x_{p+2} is the target output. There are $n-p$ training samples in all. The objective of training the neural network is to find the weights in order to make some evaluation criteria optimized.

2.2. Research framework

The research framework of this study is shown in Fig. 1.

First, collecting exchange rate data and dividing it into two parts after data preprocessing. One is training set and the other is testing set.

Second, designing the architecture of DBN. Since there is no mature method to determine the architecture of a DBN, we propose an experimental method to solve this problem. In this stage, training data is used to determine the optimal numbers of input nodes, hidden nodes, and hidden layers.

Third, update the weight parameters. There are two methods to update the weight matrix in the learning algorithm of DBN, which are denoted by DBN(1) and DBN(2) separately. In this work, we conduct a comparative study to compare the effect of these two methods.

Fourth, determine other parameters. Besides, DBN model also involves a number of other parameters such as the upper and lower bounds of the function φ_j in (5), the learning rate, and some constant in the learning algorithm. The values of these parameters may also affect the results of prediction.

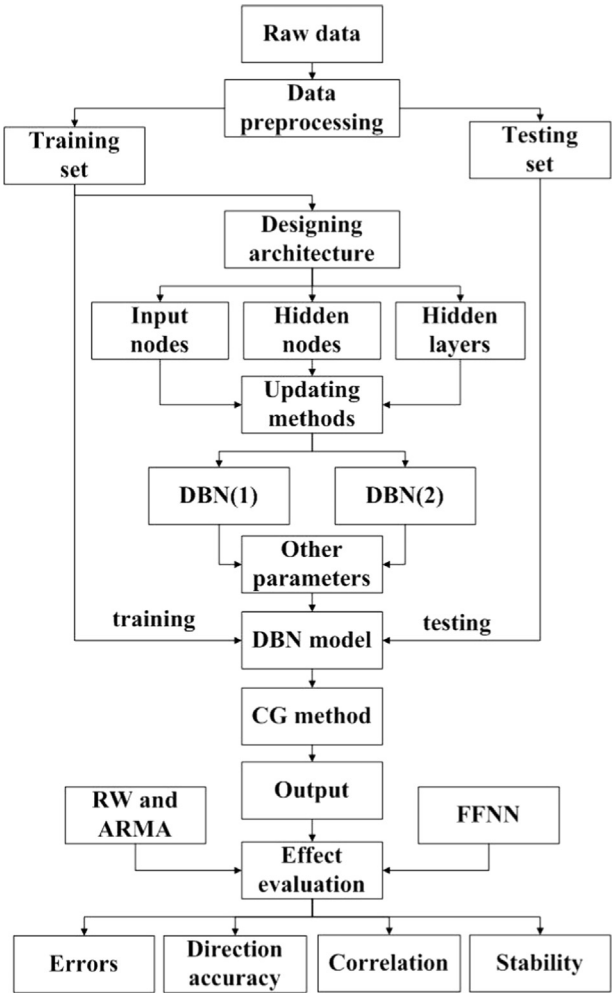

Fig. 1. The research framework of the proposed method.

Through the above process, we construct a DBN model for the exchange rate forecasting task, and the DBN is trained by using the training data.

Fifth, speed up the training. In order to accelerate the training process, conjugate gradient method is combined with DBN during the training. A DBN is trained directly and then error between the actual output and target output is calculated. This error can be expressed as a function of weight matrix W. We fine tune the weight matrix constantly using the conjugate gradient method until the error function is minimized.

At last, we turn to the forecasting period. After the DBN model is trained, testing data is used to test the effects of our model. The outputs of DBN are the predictive results of the future exchange rate.

Finally, we compare the testing results of the proposed method to those of FFNN, random walk (RW), and Auto-Regressive and Moving Average (ARMA) models. We apply six evaluation criteria from four aspects to evaluate the performance. The four aspects include prediction error, direction accuracy, correlation between actual exchange rate series and predictive ones, and the prediction stability.

2.3. Improve deep belief network to model continuous data

A DBN is a feedforward neural network with many hidden layers. An example of a DBN is shown in Fig. 2. It consists of an input layer which contains the input units (called visible units), a

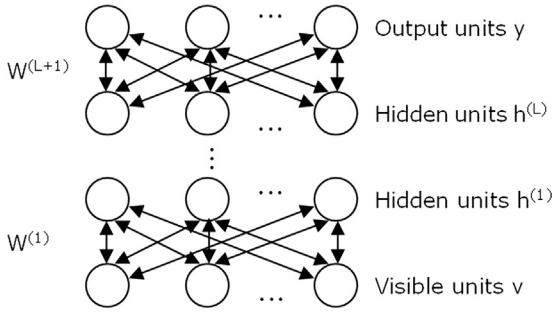

Fig. 2. DBN architecture with L hidden layers.

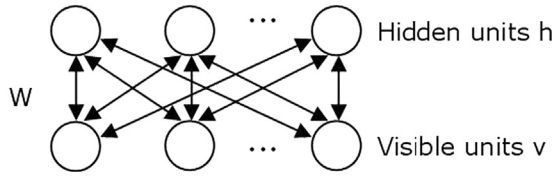

Fig. 3. RBM architecture.

number L of hidden layers and finally an output layer. w^j is the weight matrix between the units of layers $j-1$ and j, and b^j is the biases of layer j.

How to initialize the weight matrix and biases is a problem for training a DBN. Hinton [10] introduced a greedy layer-wise unsupervised learning algorithm for DBNs. This algorithm is based on the training of a sequence of RBMs. A RBM is a two layer neural network in which stochastic binary inputs are connected to stochastic binary outputs using symmetrically weighted connections. The first layer corresponds to inputs (visible units v) and the second layer to the hidden units h of the RBM. An example of a RBM is given in Fig. 3.

Let v_i and h_j represent the states of visible unit and hidden unit respectively, and $w_{ij}=w_{ji}$ are the bidirectional weights. The state probabilities of the units are

$$p_{vi}=p(v_i=1)=1\Big/\left[1+\exp\left(-\sum_j w_{ij}h_j\right)\right]. \tag{1}$$

$$p_{hj}=p(h_j=1)=1\Big/\left[1+\exp\left(-\sum_i w_{ij}v_i\right)\right]. \tag{2}$$

The RBM training process is described as follows. A training sample is first presented to the visible units to produce $\{v_i\}$. The hidden states $\{h_j\}$ are then sampled according to probabilities in (2). Repeating this process once more to update the visible and then the hidden units produce one-step 'reconstructed' states v_i' and h_j'. The update equation for w_{ij} is

$$\Delta w_{ij}=\eta(<v_jh_j>-<v_i'h_j'>) \tag{3}$$

where η is a learning rate, $<\cdot>$ refers to the mean over the training data.

If we treat the probabilities in (1) of visible units as approximations to the continuous values, then the RBM can model continuous data [18]. However, this kind of RBM tends to generate continuous data with high symmetry. Chen and Murray [19] introduce a continuous restricted Boltzmann machine (CRBM) with a simple and reliable training algorithm. With continuous-valued stochastic units, the CRBM offers improved ability with real continuous data. Since exchange rate data is continuous, we improve the classical DBN model by using continuous restricted Boltzmann machines (CRBMs) to construct a DBN. Thus the improved DBN above is able to model continuous data and to forecast exchange rate.

Let s_j be the output of neuron j, with inputs from neurons with states $\{s_i\}$:

$$s_j=\varphi_j\left(\sum_i w_{ij}s_i+\sigma\cdot N_j(0,1)\right) \tag{4}$$

with

$$\varphi_j(x_j)=\theta_L+(\theta_H-\theta_L)\cdot\frac{1}{1+e^{(-a_jx_j)}} \tag{5}$$

where $N_j(0,1)$ represents a Gaussian random variable with zero mean and unit variance. σ is a constant. $\varphi_j(x)$ is a sigmoid function with asymptotes at θ_L and θ_H. Parameter a_j is a 'noise-control' parameter. It controls the slope of the sigmoid function, and thus the nature and extent of the unit's stochastic behavior [20]. Such behavior is similar to the noisy unit in [21]. The update equations for w_{ij} and a_j are

$$\Delta w_{ij}=\eta_w(<s_is_j>-<s_i's_j'>) \tag{6}$$

$$\Delta a_j=\frac{\eta_a}{a_j^2}(<s_j^2>-<s_j'^2>) \tag{7}$$

where η_w and η_a are learning rates, s_j' denotes, as before, the one-step sampled state of unit j, and $<\cdot>$ refers to the mean over the training data. A simplified version of the same learning rule is used for the biases.

To construct a DBN we sequentially train as many CRBMs as the number of hidden layers in the DBN. These CRBMs are placed one on top of the other resulting in a DBN without the output layer.

We use a learning algorithm which is similar to that in [10]. The training of a DBN progresses on a layer-by-layer basis. First, a CRBM is trained directly on the input data. Hence, the neurons in the hidden layer of the CRBM can capture the important features of the input data. The activations of the trained features are then used as "input data" to train a second CRBM. This process of learning is continued until a prescribed number of hidden layers in the DBN have been trained.

2.4. Conjugate gradient method

Conjugate gradient (CG) algorithm is an important optimization algorithm. Based on the survey carried by Hager and Zhang [24], a CG algorithm was presented by Hestenes and Stiefel in 1952, then several researchers proposed different approaches. Since the learning of a DBN costs much time, conjugate gradient algorithm is used to speed up the learning process. During the CG training, backpropagation of error derivatives are employed to fine-tune the weights for optimal reconstruction. It is shown that the optimized two-phase training procedure enables fast convergence.

The CG training process can be realized by minimizing the mean square error (MSE) function defined by

$$MSE=E(e^Te)=E((o_d-o)^T(o_d-o)) \tag{8}$$

where $E(\cdot)$ is the mathematical expectation function, o_d-o is the error of network output. The general purpose of the CG training is to search an optimal set of connection weights in the manner that the errors defined by Eq. (8) can be minimized. During the CG training phase each weight of the network varies according to

$$w_{k+1}=w_k+\alpha_kd_k \tag{9}$$

The learning rate α_k can be determined by line search techniques in the way that $MSE(w_k+\alpha_kd_k)$ is minimized along the direction d_k, given w_k and d_k fixed. In the beginning of CG method, the initial search direction d_0 is calculated by

$$d_0=-\nabla MSE(w_0)=-g_0 \tag{10}$$

Each direction d_{k+1} is chosen to be a linear combination of the steepest descent direction $-g_{k+1}$ and the previous direction d_k.

We write

$$d_{k+1} = -g_{k+1} + \beta_k d_k \tag{11}$$

where the scalar d_k is to be determined by the requirement that d_k and d_{k+1} must fulfill the conjugacy property. β_k is the CG update parameter and different CG methods correspond to the choice of β_k. One of them is the formula introduced by Fletcher and Reeves [25] and is given by

$$\beta_k = \frac{\| g_{k+1} \|^2}{\| g_k \|^2}. \tag{12}$$

2.5. Testing and performance measures

We apply one-step-ahead forecasting in the experiments. In one-step-ahead prediction, the predicted values are forecasted one step at a time and the actual values are then used for the next prediction. It is popular and useful in evaluating the adaptability and robustness of a forecasting model. Even if a poor prediction occurs in one step, it will not cause serious consequences. This is because actual values will be used to correct itself for future forecasts.

In this paper, we use six criteria to evaluate the performance of a DBN in forecasting exchange rate. They are root mean square error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), direction accuracy (DA), Pearson correlation coefficient (CORR), and Variance. The formulas of the five predictive accuracy measures are listed as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{T} (y_i - y_i')^2}{T}} \tag{13}$$

$$MAE = \frac{\sum_{i=1}^{T} |y_i - y_i'|}{T} \tag{14}$$

$$MAPE = \frac{1}{T} \sum_{i=1}^{T} \left| \frac{y_i - y_i'}{y_i} \right| \times 100 \tag{15}$$

$$DA = \frac{1}{T} \sum_{i=1}^{T} a_i \quad \text{where} \quad a_i = \begin{cases} 1 & \text{if } (y_{i+1} - y_i)(y_{i+1}' - y_i) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

$$CORR = \frac{E(YY') - E(Y)E(Y')}{\sqrt{E(Y^2) - E^2(Y)}\sqrt{E(Y'^2) - E^2(Y')}} \tag{17}$$

where $Y = (y_1, y_2, \ldots, y_T)^T$ is a vector of actual observations, and $Y' = (y_1', y_2', \ldots, y_T')^T$ is a vector of predicted values, and T is the number of predictions.

These criteria are frequently used performance measures in research. Furthermore, RMSE, MAE, and MAPE are three popular criteria in the literature. They are all real number greater than 0, the smaller their value the more accurate is the prediction. They are all mean based measures, and cannot show the variation through different runs. Since stability is also an important target of exchange rate prediction, we use variance of 50 runs as well as the mean based measures in performance evaluations. In other words, we train each neural network 50 times and obtain 50 values of three mean based measures. Then the variances of the three criteria are calculated to show the forecasting stability of the DBN model.

From view of investors, direction accuracy (DA) is one of the most important evaluation criteria. It represents the probability of correctly predicted direction. An investor is more enthusiastic to know the directional change in the future exchange rate rather than the exact magnitude, because they can develop the trading strategies according to the direction accuracy.

CORR is a criterion to evaluate the correlation between the predicted exchange rate series and the actual one. If the value of CORR approaches to 1, it means the correlation is strong.

3. Analysis

In Section 2, we discussed the research design of the proposed method. In this section, we will give the detail of how to preprocess the input data, design the architecture of DBN, update the weight, determine the parameters, and measure the performance.

3.1. Data preparation

We use three data sets of exchange rates for experiments.

The first one contains weekly rates of British Pound/US dollar (GBP/ USD) exchange rate from the beginning of 1976 through the end of 1993. It consists of 937 observations totally. We keep 885 observations for training and remaining 52 observations are kept for testing.

The second is weekly rates of Brazilian Real/US dollar (BRL/ USD) from 1st January 2000 to 1st January 2004. There are 167 observations in training set and 42 in testing set.

Weekly rates of Indian Rupee/US dollar (INR/USD) exchange rate for the period of 6th January 1994 to 10th July 2003 compose the last data set, for a total 496 observations. 350 observations are kept for training, and the remaining observations are used as test samples.

All the exchange rate time series above are downloaded from the database retrieval system of "Pacific Exchange Rate Service" (http://fx.sauder.ubc.ca/data.html). The weekly returns are calculated as the log differences of the levels. Let p_t be the exchange rate price for the period t. Then the exchange rate return at time t is calculated as $y_t = (\log(p_t) - \log(p_{t-1})) \times 100$. In order to reduce the round-off errors, we multiply the log difference by 100. We normalize the exchange rate return of INR/USD and BRL/USD data to the value between 0 and 1, and use raw data of GBP/USD exchange rate. The training samples are selected randomly during the training period.

3.2. Designing the architecture of DBN

In the design of a neural network, the numbers of input, hidden nodes and hidden layers are critical parameters. In exchange rate forecasting, the number of input nodes corresponds to the number of past observations related to future values. The number of hidden nodes enables neural networks to capture nonlinear patterns from the data. On one hand, neural networks with too few hidden nodes may not have enough power to model the data. On the other hand, neural networks with too many hidden nodes may lead to overfitting problems and finally result in poor forecasting performance. DBN is a generative neural network model with many hidden layers. These hidden layers allow a DBN to be more powerful in modeling the complex relationship from the data. Hence, the changeable number of hidden layers provides the designer a great deal of freedom. However, this freedom also brings problems meanwhile. Nowadays, there is no mature method in theory to determine the number of hidden layers of a DBN. Therefore, selecting the optimal DBN architecture remains to be a difficult task.

In this paper, the number of input nodes and hidden nodes of the DBN is selected by experimentation. Ten levels of the number of input nodes ranging from 1 to 10 will be used in this study. However, we only experiment with five levels of hidden nodes 4, 8, 12, 16 and 20. Because previous research [22] finds that the forecasting performance of neural networks is not as sensitive to the number of hidden nodes as to the number of input nodes.

Since we choose the one-step-ahead forecasts, one output node is enough to satisfy our needs.

How many hidden layers should we use in our experiment is another important problem. As a matter of fact, there is no simple answer. Extensive experiments by Yoshua Bengio's group [23] suggest that several hidden layers are better than one. However, there is not yet a optimal number of hidden layers in theory. DBNs give their creator a lot of freedom, and the best way to use this freedom depends on the task. In this paper, we apply an experimental method to determine the number of hidden layers. We divide the training data into two parts in our experiment. Meanwhile, 80% of the data are used for training, and the rest for validation.

3.2.1. GBD/USD

At first, we initialize the DBN with 1 hidden layer. Table 1 shows the effects of input nodes and hidden nodes on DBN validation performance of GBP/USD exchange rate forecasting. All the values of RMSE, MAE, and MAPE are listed in Table 1, at each level of input node in combination with each level of hidden node is the average of 10 runs. The results show that the best average performance across RMSE and MAPE occurs at 6 input nodes. For MAE, the best one is 0.006426 at 10 input nodes. In addition, as the number of input nodes increases, the average of RMSEs, MAEs, and MAPEs is irregularly changed, thus there is no clear hint for determining hidden nodes.

Then, we initialize the DBN with 2 hidden layers. Since the best RMSE at 6 input nodes level occurs at 16 hidden nodes, we set DBN with 6 input nodes, 16 hidden nodes in the first hidden layer, and combined with different hidden nodes in the second hidden layer. The experimental results are shown in Table 2. Obviously, the best RMSE and MAPE occur at 8 hidden nodes, which are 0.008709 and 1.028106, respectively. Therefore, the number of hidden nodes in the second hidden layer is set to 8.

The above experiment is repeated for DBN with 3 hidden layers. Fig. 4 shows the effects of the number of hidden layers on RMSE and MAE. The values of RMSE for the three levels of hidden layer cases are 0.0093367, 0.0082099 and 0.0083181. The MAE of DBN with two hidden layers is 0.0064630, which is lower than the MAE of the other two cases (0.0068202 and 0.0067298). The values of MAPE for all the three cases are shown in Fig. 5. They are 1.0863, 0.9753 and 1.0129 respectively. As shown in Figs. 4 and 5, the predictive error first decreased and then increased following the increasing of the number of hidden layers. Therefore we choose a DBN with two hidden layers in out-of-sample forecasting. Here, out-of-sample forecasting means forecasting with testing data. The optimal architecture of this DBN is 6–16–8–1. In other words, 6 input nodes, 16 hidden nodes in the first hidden layer, and 8 hidden nodes in the second hidden layer is the best combination. In out-of-sample forecasting, we use a DBN with this optimal architecture to predict the GBP/USD exchange rate.

3.2.2. INR/USD

The effects of input and hidden nodes in DBN for INR/USD are shown in Table 3. It gives similar results as GBP/USD. The best average performance across RMSE, MAE, and MAPE all occurs at 10

Table 2
Effects of hidden nodes in the second hidden layer on the validation performance (GBP/USD). Bold numbers is the minimum RMSE, MAE, and MAPE respectively.

Hidden	RMSE	MAE	MAPE
3	0.009385	0.007568	1.135931
4	0.008854	0.006981	1.048574
6	0.009479	0.007680	1.152535
8	**0.008709**	0.006844	**1.028106**
10	0.009269	**0.006587**	1.101682
12	0.009140	0.007316	1.098983
14	0.009058	0.007193	1.080773
16	0.009022	0.007132	1.071604
18	0.008764	0.006917	1.038714
20	0.009484	0.007597	1.141959

Table 1
Effects of DBN factors on the validation performance: input nodes, hidden nodes in the first hidden layer (GBP/USD).

Input	Hidden	RMSE	MAE	MAPE	Input	Hidden	RMSE	MAE	MAPE
1	4	0.009247	0.006531	1.090219	6	4	0.009274	0.006383	1.052466
1	8	0.009233	0.006486	1.079636	6	8	0.009236	0.007203	1.085414
1	12	0.009351	0.006493	1.075053	6	12	0.009148	0.006439	1.070667
1	16	0.009432	0.006482	1.067716	6	16	0.008897	0.006996	1.053810
1	20	0.009785	0.006673	1.091961	6	20	0.009129	0.006393	1.060515
	Avg	0.009410	0.006533	1.080917		Avg	0.009137	0.006683	1.064574
2	4	0.009515	0.007503	1.131746	7	4	0.009388	0.007258	1.093679
2	8	0.009657	0.007218	1.088668	7	8	0.009424	0.006562	1.087873
2	12	0.009493	0.007455	1.128953	7	12	0.009328	0.006545	1.086150
2	16	0.008647	0.006975	1.053711	7	16	0.009515	0.006600	1.088484
2	20	0.009244	0.007196	1.085657	7	20	0.009347	0.006510	1.077260
	Avg	0.009311	0.007269	1.097747		Avg	0.009400	0.006695	1.086689
3	4	0.009515	0.007503	1.131746	8	4	0.009266	0.006528	1.085534
3	8	0.009844	0.007725	1.162689	8	8	0.009292	0.006486	1.075270
3	12	0.009733	0.007640	1.153047	8	12	0.009386	0.006478	1.069954
3	16	0.009157	0.007454	1.126828	8	16	0.009360	0.006466	1.066969
3	20	0.009207	0.007252	1.094591	8	20	0.009484	0.006451	1.059118
	Avg	0.009491	0.007515	1.133780		Avg	0.009358	0.006482	1.071369
4	4	0.009734	0.007598	1.148011	9	4	0.009157	0.006474	1.077547
4	8	0.009733	0.007640	1.153047	9	8	0.009124	0.006450	1.074385
4	12	0.009236	0.007203	1.085414	9	12	0.009335	0.006482	1.073456
4	16	0.009493	0.007455	1.128953	9	16	0.009556	0.006614	1.089818
4	20	0.009002	0.006732	1.016771	9	20	0.009375	0.006447	1.063755
	Avg	0.009440	0.007326	1.106439		Avg	0.009309	0.006493	1.075792
5	4	0.009258	0.006410	1.059348	10	4	0.009226	0.006488	1.078418
5	8	0.009249	0.007195	1.083677	10	8	0.009152	0.006423	1.067404
5	12	0.009839	0.006558	1.070537	10	12	0.009038	0.006382	1.061104
5	16	0.009272	0.006540	1.088892	10	16	0.009155	0.006395	1.060783
5	20	0.009244	0.007196	1.085657	10	20	0.009117	0.006441	1.070950
	Avg	0.009372	0.006780	1.077622		Avg	0.009138	0.006426	1.067732

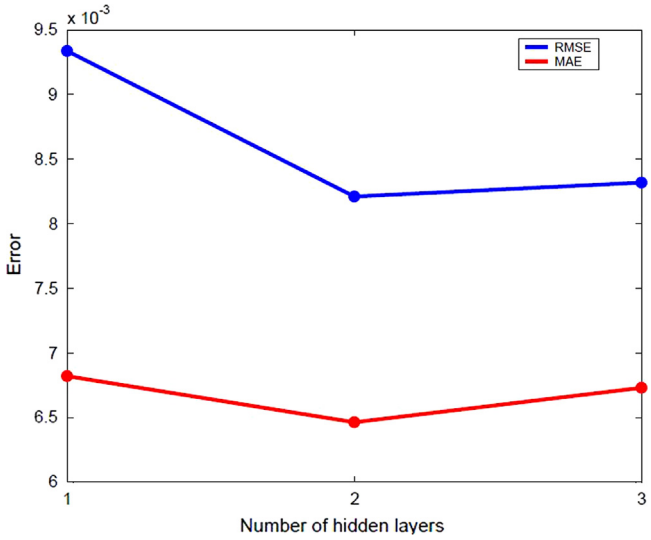

Fig. 4. Effects of the number of hidden layers on RMSE and MAE (GBP/ USD).

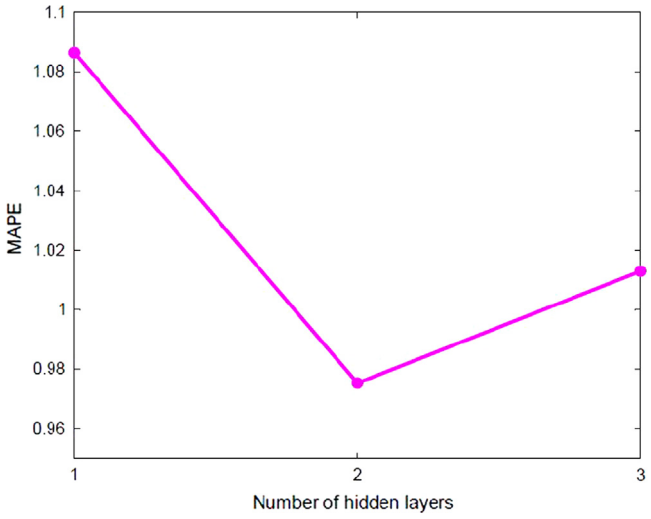

Fig. 5. Effects of the number of hidden layers on MAPE (GBP/USD).

input nodes. At 16 hidden nodes level, there are the best MAE and MAPE values. Besides, the best RMSE at 10 input nodes level occurs at 8 hidden nodes. Then we try with 10 input nodes combined with more levels of hidden nodes. We find that 3 hidden nodes have better performance than all the other levels in three measures. The values of RMSE, MAE and MAPE are 0.016310, 0.012188 and 2.923540, respectively.

Then two hidden layers are taken into consideration. After trying several combinations, we find that the optimal architecture is 10–3–3–1. That is a DBN with ten input nodes and two hidden layers with three hidden nodes in each layer. The validation performance of different hidden nodes in the second hidden layer is shown in Table 4. The results show that a DBN with 3 hidden nodes in the second hidden layer outperforms other combinations by all the evaluation criteria. We also consider DBNs with three hidden layers and compare all the three cases. The result is similar to that of GBP/USD. DBNs with two hidden layers beat the other two cases in all the three measures. Therefore a DBN with 10–3–3–1 architecture is applied in out-of-sample forecasting of INR/USD exchange rate return.

3.2.3. BRL/USD

Table 5 shows the effects of input and hidden nodes in DBN for BRL/USD. A DBN with 10 input nodes has the best average performance across RMSE, MAE, and MAPE. Since the best RMSE at 10 input nodes level occurs at 4 hidden nodes, we try to use a DBN with 10 input nodes, 4 hidden node in the first hidden layer and different hidden nodes in the second hidden layer. The experimental results are shown in Table 6. Obviously, the best RMSE, MAE and MAPE occur at 8 hidden nodes, which are 0.003854, 0.003074 and 1.055207, respectively. Hence, the number of hidden nodes in the second hidden layer is set to 8. After considering DBNs with three hidden layers and comparing all the three cases, we obtain the optimal architecture, which is 10–4–8–1. Then a DBN with 10–4–8–1 is applied in out-of-sample forecasting of BRL/USD exchange rate.

3.3. Methods of updating weight matrix

The training of a DBN progresses on a layer-by-layer basis. During the training, we apply two different methods to update the weight matrix W of DBN.

The first one is, training the first CRBM and updating its weight matrix until it is stable. Then turn to train the next CRBM, with all the weight matrixes of former CRBM fixed. This method is denoted by DBN(1).

The other method is updating all the weight matrixes of CRBMs at the same time, which is denoted by DBN(2). We conduct a comparative experiment between two methods above by using the same time series data.

Table 7 shows the results, in which DBN(1) denotes the first method and DBN(2) is the other. In the series of GBP/USD, DBN (2) obtains the best performance. All the five criteria are improved, especially the direction accuracy, which is equal to 0.6362, is improved by 10.5%. For INR/USD data, the errors of DBN(1) are lower than those of DBN(2), and DBN(1) obtains the better direction accuracy. For CORR, the value of DBN(2) is 0.3753, which is better than DBN(1). In BRL/USD series, it is not clear which method has the best performance. Although all the errors of DBN (1) is lower than those of DBN(2), the results of DA and CORR are just the opposite.

3.4. Other parameters of DBN

There are still some parameters to be determined in a DBN model. The learning rates η_w in (6), a_j in (7) and constant σ in (4) should be given by users. In this paper, we determine these values through 5-fold cross-validation conducted on training sets.

θ_L and θ_H in (5) are set to be the minimum and maximum of a training sample respectively. There are two methods to determine the parameters. The first one is to use fixed values, i.e., set θ_L and θ_H as the minimum and maximum of the entire training data before training the DBN. For the second one, since during the training period, all the training samples are selected randomly. Therefore, when a new training sample is selected, the values of θ_L and θ_H are changed to be minimum and maximum of this sample. That means the values of these two parameters are changeable according to the selected sample.

In order to compare these two methods, we apply a DBN to both GBP/ USD and INR/USD with changeable and fixed methods. Table 8 shows the differences between changeable and fixed methods with 6 input nodes and 8 hidden nodes. All the values are average calculated through 10 runs. Obviously, changeable values beat the fixed ones by all the three mean based measures. Moreover, the variance of using changeable values method is superior to that of fixed values method, i.e., changeable values

Table 3
Effects of DBN factors on the validation performance: input nodes, hidden nodes in the first hidden layer (INR/USD).

Input	Hidden	RMSE	MAE	MAPE	Input	Hidden	RMSE	MAE	MAPE
1	4	0.018847	0.015220	3.692316	6	4	0.017489	0.013836	3.343832
1	8	0.016753	0.012914	3.115281	6	8	0.016906	0.012936	3.125710
1	12	0.016477	0.011959	2.879695	6	12	0.016690	0.012542	3.025899
1	16	0.017578	0.013436	3.248335	6	16	0.015980	0.011509	2.768321
1	20	0.017960	0.014000	3.387795	6	20	0.016910	0.012928	3.121430
	Avg	0.017523	0.013506	3.264684		Avg	0.016795	0.012750	3.077038
2	4	0.017650	0.013749	3.320943	7	4	0.017705	0.013518	3.269499
2	8	0.017475	0.013488	3.256673	7	8	0.015925	0.010743	2.555650
2	12	0.017226	0.013072	3.153819	7	12	0.017885	0.013831	3.347238
2	16	0.017320	0.013207	3.187277	7	16	0.017672	0.013550	3.277142
2	20	0.016501	0.012039	2.895303	7	20	0.017719	0.013629	3.297093
	Avg	0.017234	0.013111	3.162803		Avg	0.017381	0.013054	3.149324
3	4	0.017215	0.013100	3.160465	8	4	0.016579	0.012278	2.963428
3	8	0.017228	0.013119	3.165104	8	8	0.017361	0.013402	3.241503
3	12	0.017271	0.013174	3.178757	8	12	0.016891	0.012411	2.995415
3	16	0.017313	0.013263	3.200497	8	16	0.017792	0.013631	3.299707
3	20	0.017081	0.012908	3.112482	8	20	0.017862	0.013724	3.322671
	Avg	0.017222	0.013113	3.163461		Avg	0.017297	0.013089	3.164545
4	4	0.017434	0.013713	3.316882	9	4	0.017937	0.013972	3.233582
4	8	0.018161	0.014684	3.555119	9	8	0.017743	0.013714	3.321035
4	12	0.016653	0.012469	3.009571	9	12	0.018209	0.014467	3.504322
4	16	0.016655	0.012486	3.013569	9	16	0.018200	0.014281	3.459824
4	20	0.017003	0.012949	3.128789	9	20	0.018021	0.014290	3.360399
	Avg	0.017181	0.013260	3.204786		Avg	0.018022	0.014145	3.375832
5	4	0.018068	0.014563	3.523504	10	4	0.016743	0.012330	2.961865
5	8	0.018539	0.015106	3.659409	10	8	0.016739	0.012491	3.001156
5	12	0.018850	0.015457	3.746940	10	12	0.016988	0.013120	3.156640
5	16	0.016988	0.012881	3.112370	10	16	0.016825	0.012241	2.939337
5	20	0.017080	0.012961	3.132876	10	20	0.016629	0.012413	2.981169
	Avg	0.017905	0.014194	3.435040		Avg	0.016785	0.012519	3.008033

Table 4
Effects of hidden nodes in the second hidden layer on the validation performance (INR/USD). Bold numbers is the minimum RMSE, MAE, and MAPE respectively.

Hidden	RMSE	MAE	MAPE
3	**0.016075**	**0.012078**	**2.896550**
4	0.016450	0.012244	2.938897
6	0.016758	0.012672	3.043515
8	0.017817	0.013790	3.323594
10	0.017192	0.013099	3.155563
12	0.017580	0.013494	3.252126
14	0.018660	0.014743	3.560877
16	0.017580	0.013579	3.275304
18	0.016636	0.012348	2.963746
20	0.017024	0.012710	3.058764

method is more stable and reliable. This result shows that using changeable values can improve the predictive accuracy, as well as the stability, which is better than using fixed values all the way.

The initial values of w_{ij} in (4) and a_j in (5) also play a role in performance of a DBN. According to the principle of simulated annealing, a number of randomly generated initial values are employed in our experiments. We train each DBN 50 times by using 50 sets of different initial values. The average solution in 50 runs is used as the training solution of a DBN. Then backpropagation of error derivatives is employed to fine-tune the weights for optimal reconstruction. During this phase, we apply CG method to accelerate the training.

Simulated annealing is able to find the global optima. The training of DBN is based on simulated annealing, and if we do enough iteration for DBN, we can get global optima. Even simulated annealing in principle can reach the global optima, since this algorithm is a probability method, it is not sure that we will obtain enough iteration times during the training phase. Therefore, we cannot guarantee the final training result must be the global optima.

The advantage of DBN is contributed by the combination of the above two steps. The pre-training step in DBN provides a decent initial value in training phase, but this step usually cannot reach the minimum point directly. Providing a situation that if we could obtain decent initial value, we can adopt the CG fine-tuning step to converge on the minimum point quickly.

4. Comparison experiments

The major purpose of this study is to evaluate the performance of DBNs in forecasting exchange rates. In this section, we do out-of-sample analysis to examine the forecasting power of DBNs. We conduct a comparative study among DBNs, FFNNs, and two traditional methods (RW, ARMA) in exchange rate forecasting by using the same time series data. In our experiment, the comparison-used FFNNs adopts the same structure (which means maintaining the same layers and the same node numbers with DBN) to compare with DBN results. The reason for us to make this decision is due to the fact that we find the experimental results not to be sensitive to different structures of FFNNs in our experiment. To stress the comparison with DBN experiment, we adopt the same structures in FFNNs. The FFNNs used in comparison have four layers. Logistic activation functions are employed in the hidden layer and the linear activation function is utilized in the output layer. Besides, the cost function is mean square error and backpropagation algorithm is employed to train the FFNN. In order to simplify the comparison with DBNs, the parameter settings of the autoregressive part and the moving average part in each ARMA orders are set to same numbers with the input neuron numbers in the corresponding DBN.

The results for out-of-sample forecasting performance on weekly GBP/USD series are presented in Table 9. All the values listed in the table are mean values through 50 runs. In Table 9, we can see that DBN gives better predictive accuracy than FFNN in all the five criteria.

Table 5
Effects of DBN factors on the validation performance: input nodes, hidden nodes in the first hidden layer (BRL/USD).

Input	Hidden	RMSE	MAE	MAPE	Input	Hidden	RMSE	MAE	MAPE
1	4	0.004993	0.003755	1.252816	6	4	0.005200	0.004348	1.494126
1	8	0.004988	0.003743	1.248423	6	8	0.005113	0.004284	1.471872
1	12	0.004991	0.003749	1.250619	6	12	0.005304	0.004474	1.537906
1	16	0.004991	0.003749	1.250619	6	16	0.005213	0.004375	1.50347
1	20	0.00499	0.003747	1.249887	6	20	0.005047	0.004197	1.442278
	Avg	0.004990	0.003749	1.250473		Avg	0.005175	0.004336	1.489930
2	4	0.006585	0.005515	1.862919	7	4	0.005461	0.004621	1.588872
2	8	0.006436	0.005267	1.777107	7	8	0.005462	0.004650	1.599367
2	12	0.006526	0.005198	1.723072	7	12	0.005267	0.004468	1.536687
2	16	0.006241	0.004788	1.609481	7	16	0.005906	0.005109	1.757867
2	20	0.005924	0.004408	1.478460	7	20	0.004749	0.003984	1.369764
	Avg	0.006343	0.005035	1.690208		Avg	0.005369	0.004566	1.570512
3	4	0.005707	0.004833	1.643715	8	4	0.005091	0.004334	1.489604
3	8	0.005798	0.004877	1.658476	8	8	0.005382	0.004611	1.586055
3	12	0.005956	0.004915	1.669765	8	12	0.005191	0.004424	1.521660
3	16	0.005871	0.004683	1.588712	8	16	0.005241	0.004487	1.543434
3	20	0.005816	0.00445	1.506655	8	20	0.004986	0.004238	1.457675
	Avg	0.005830	0.004752	1.613464		Avg	0.005178	0.004419	1.519686
4	4	0.00556	0.004750	1.623006	9	4	0.004678	0.003994	1.371409
4	8	0.005544	0.004708	1.608794	9	8	0.004463	0.003771	1.295051
4	12	0.005219	0.004364	1.490759	9	12	0.004718	0.003986	1.369299
4	16	0.005351	0.004439	1.516218	9	16	0.004759	0.004021	1.381664
4	20	0.005031	0.004093	1.396897	9	20	0.004743	0.004039	1.388157
	Avg	0.005341	0.004471	1.527135		Avg	0.004672	0.003962	1.361116
5	4	0.005243	0.004493	1.540349	10	4	0.004039	0.003377	1.160798
5	8	0.005459	0.004686	1.606807	10	8	0.004217	0.003520	1.210410
5	12	0.005126	0.004332	1.485048	10	12	0.004313	0.003604	1.239331
5	16	0.005298	0.004493	1.540559	10	16	0.004498	0.003716	1.277909
5	20	0.00556	0.00475	1.623006	10	20	0.004262	0.003456	1.187966
	Avg	0.005213	0.004428	1.518153		Avg	0.004266	0.003535	1.215283

Table 6
Effects of hidden nodes in the second hidden layer on the validation performance (BRL/USD). Bold numbers is the minimum RMSE, MAE, and MAPE respectively.

Hidden	RMSE	MAE	MAPE
2	0.004357	0.003600	1.237284
4	0.004121	0.003358	1.153245
6	0.004302	0.003520	1.209119
8	**0.003854**	**0.003074**	**1.055207**
10	0.004524	0.003778	1.298458
12	0.005277	0.004463	1.536039
14	0.004260	0.003473	1.192101
16	0.006805	0.005931	2.047795
18	0.006265	0.005441	1.875787
20	0.003957	0.003197	1.098099

Table 7
Effects of different weight updating methods. Bold numbers is the minimum RMSE, MAE, and MAPE respectively.

Time series	GBP/USD		INR/USD		BRL/USD	
	DBN(1)	DBN(2)	DBN(1)	DBN(2)	DBN(1)	DBN(2)
RMSE	8.21E−3	**7.69E−3**	**1.65E−2**	1.70E−2	**3.85E−3**	4.09E−3
MAE	6.46E−3	**6.01E−3**	**1.24E−2**	1.31E−2	**3.07E−3**	3.30E−3
MAPE	0.9753	**0.9028**	**2.9687**	3.1486	**1.0552**	1.1336
DA	0.5755	**0.6362**	**0.5873**	0.5670	0.4375	0.4509
CORR	0.8733	**0.8817**	0.3677	**0.3753**	0.7101	**0.7157**

The correlation coefficient for DBN is 0.8733, which is better than that of FFNN (0.8459). Hence the correlation of predictive series generated by DBN and actual one is higher than that of FFNN.

We use DA to measure the accuracy of predicting the trend of exchange rate. From investor's point of view, DA is one of the most important evaluation criteria. Because an investor is more concerned about the directional change in future exchange rate than the exact values of it. Table 9 shows that DBN has significantly higher direction accuracy, which is 0.5755, than FFNN model (0.4174). It is obvious that the out-of-sample forecasts of DBN are more accurate than FFNN forecasts.

Table 8
Effects of changeable and fixed parameters on the validation performance. Bold numbers is the minimum RMSE, MAE, and MAPE respectively.

Time series	GBP/USD		INR/USD	
	Changeable	Fixed	Changeable	Fixed
RMSE	**0.008958**	0.009031	**0.016447**	0.018782
MAE	**0.006348**	0.006437	**0.012290**	0.015344
MAPE	**1.057375**	1.075311	**2.959554**	3.715404
Variance (RMSE)	**6.163E−10**	3.028E−8	**8.439E−7**	5.417E−6
Variance (MAE)	**5.813E−10**	1.996E−8	**2.064E−6**	7.651E−6
Variance (MAPE)	**2.128E−5**	7.310E−4	**0.130077**	0.460467

Table 9
Out-of-sample results on weekly GBP/USD series. Bold numbers is the minimum RMSE, MAE, and MAPE respectively.

Method	RMSE	MAE	MAPE	DA	CORR
RW	1.4430E−2	1.0685E−2	1.6034	0.5000	0.6522
ARMA	1.3820E−2	1.1814E−2	1.7635	0.3782	0.6933
DBN(1)	8.2099E−3	6.4630E−3	0.9753	0.5755	0.8733
DBN(2)	**7.6900E−3**	**6.0100E−3**	**0.9028**	**0.6362**	**0.8817**
FFNN	9.4094E−3	7.3788E−3	1.1132	0.4174	0.8459

In order to test the stability of these two models, we calculate the variance of RMSE, MAE and MAPE in 50 runs. Table 10 shows the variance of three measures in forecasting weekly GBP/USD series. All the three variance of DBNs are significantly lower than those of FFNNs, which means that the errors of exchange rate prediction with DBNs are more stable than FFNN. That is, DBNs are superior in stability and reliability to FFNNs in exchange rate forecasting.

The out-of-sample forecasting performance of weekly INR/USD exchange rate return prediction is presented in Table 11. All the values are mean values through 50 runs. It can be seen that DBN outperforms FFNN model by all the five evaluation criteria. The RMSE of DBN (1.6505E−2) is lower than the RMSE of FFNN (1.8899E−2). DBN has also got smaller values for MAE and MAPE as compared to the values of FFNN model. The DBN fitted values have significantly higher correlation, which is 0.3677, with the actual series as compared to the values of FFNN (0.1481). In Table 11, we can see that DBN gives better direction accuracy, which is 0.5873, than FFNN model (0.5407). Table 12 shows the variance of errors in forecasting weekly INR/USD exchange rate return series. Similar to the results of GBP/USD exchange rate forecasting, all the variances of DBN outperform those of FFNN model. The variance of RMSE, MAE and MAPE is 5.5911E−8, 2.5202E−7 and 1.5053E−2 respectively. The corresponding values for FFNN model are 5.2420E−7, 7.7398E−4 and 4.8386E−2, respectively. That is to say, DBNs also have high stability in forecasting exchange return series.

Table 13 shows the out-of-sample forecasting performance on weekly BRL/USD series. We can see that DBN gives better predictive accuracy than FFNN in all the five criteria. The correlation coefficient for DBN(2) is 0.7157, it is better than that of FFNN (0.6810). Hence the correlation of predictive series generated by DBN and actual one is higher than that of FFNN. Besides, DBN has higher direction accuracy, which is 0.4509, than FFNN model, which is 0.4366. Therefore, the out-of-sample forecasts of DBN are more accurate.

As shown Tables 9–13, the most important result is that nonlinear models (FFNN, DBN) performed better than linear models (RW, ARMA) in all series. Moreover, it should be emphasized that the forecast evaluation based on the errors (RMSE, MAE,MAPE) was not useful, in some cases, to distinguish between the best and the worst models. In the case of the BRL/USD series, all the errors of the DBN (2) model were greater than those of DBN(1). However, the direction accuracy of DBN(2) is the optimal through all the models listed in Table 13.

Fig. 6 shows the results of the tests on the weekly exchange rate for 46 weeks. It is obvious that the predicted exchange rate curve obtained during the simulation is very near to the actual one. For the exchange rate forecasting, the historical data are used to train the forecasting models. Also, historical data are adopted to predict the future exchange rate. It means that there is a certain relevance between the predicted values and the historical values. This is the reason why the predicted exchange rate seems to lag the actual exchange rate in Fig. 6. Furthermore, as shown in Fig. 7, the outputs of a DBN are nearer to the real exchange rate return values than those of FFNN model in a long period. Compared with the outputs of a DBN, the forecasts generated by a FFNN do not always follow the fluctuation. The error between predictive value and actual one is very high at some points. For example, at 42nd week, the actual value of INR/USD exchange return is 0.4174. But the predicted value of FFNN is 0.3725, and the predicted value of DBN is 0.4132. Fig. 8 shows the results of weekly predictions of BRL/USD. Compared to the curve generated by FFNN, the curve of the DBN is closer to the real exchange rate curve. Besides, at some point, the predictive values of FFNN seriously deviate from the real exchange rate value. For example, at the fifth week, the real exchange rate is 2.8713, while the predictive value of the DBN is 2.8971, the error is 0.0258. However, the predictive value of FFNN is 3.0508, and the error increases to 0.1795. In a word, the DBN

Table 10
Variance of RMSE, MAE, and MAPE on weekly GBP/USD series.

Method	Variance(RMSE)	Variance(MAE)	Variance(MAPE)
DBN	3.8886E−8	2.6525E−8	6.4391E−4
FFNN	8.5852E−7	6.1171E−7	1.3831E−2

Table 11
Out-of-sample results on weekly INR/USD exchange rate return series. Bold numbers is the minimum RMSE, MAE, and MAPE respectively.

Method	RMSE	MAE	MAPE	DA	CORR
RW	9.8488E−2	1.3244	2.9854	0.4044	0.3575
ARMA	8.7135E−2	1.4389	3.0250	0.4578	0.3536
DBN(1)	**1.6505E−2**	**1.2387E−2**	**2.9687**	**0.5873**	0.3677
DBN(2)	1.7000E−2	1.3100E−2	3.1486	0 .567	**0.3753**
FFNN	1.8899E−2	1.4250E−2	3.4292	0.5407	0.1481

Table 12
Variance of RMSE, MAE and MAPE on weekly INR/USD exchange rate return series.

Method	Variance (RMSE)	Variance (MAE)	Variance (MAPE)
DBN	5.5911E−8	2.5202E−7	1.5053E−2
FFNN	5.2420E−7	7.7398E−4	4.8386E−2

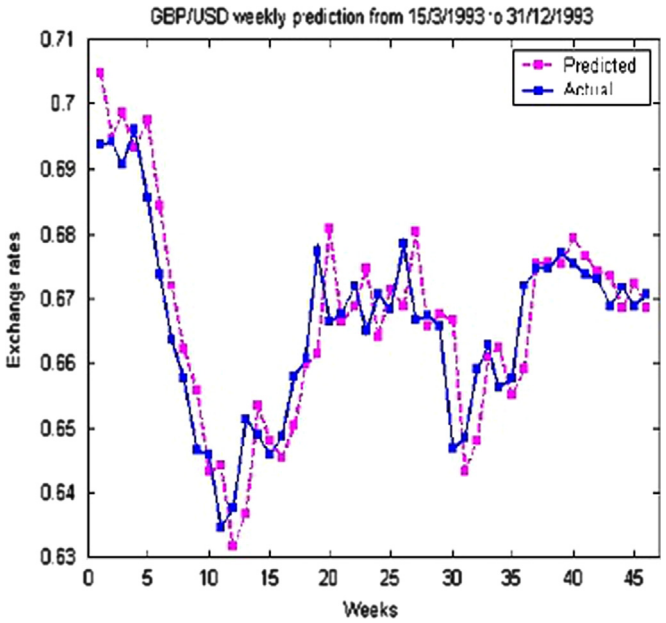

Fig. 6. GBP/USD weekly predictions with DBN.

Table 13
Out-of-sample results on weekly BRL/USD series. Bold numbers is the minimum RMSE, MAE, and MAPE respectively.

Method	RMSE	MAE	MAPE	DA	CORR
RW	3.2846E−2	2.7280E−2	1.9383	0.3438	0.5835
ARMA	3.2069E−2	2.7205E−2	1.93572	0.375	0.5997
DBN(1)	**3.8540E−3**	**3.0740E−3**	**1.0552**	0.4375	0.7101
DBN(2)	4.0880E−3	3.3003E−3	1.1336	**0.4509**	**0.7157**
FFNN	1.1128E−2	9.6440E−3	1.3216	0.4366	0.6810

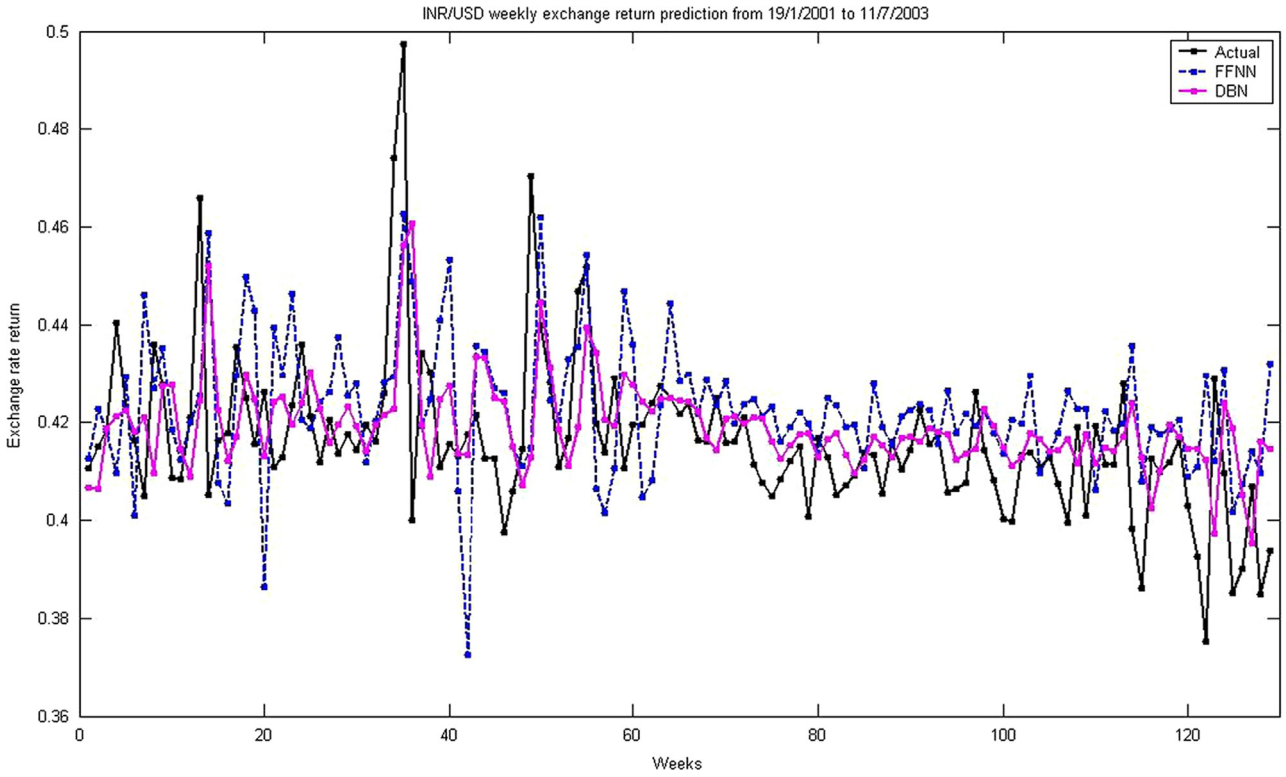

Fig. 7. The predicted output of the DBN and FFNN models against the actual weekly exchange rates return series of INR/USD.

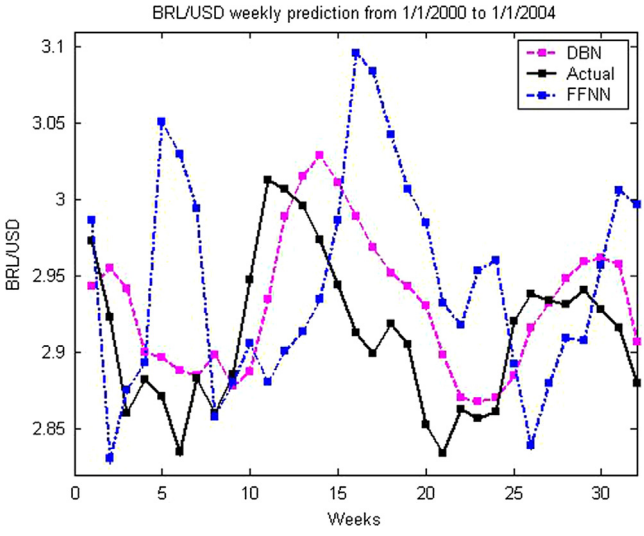

Fig. 8. BRL/USD weekly predictions with DBN and FFNN models.

learns well from the exchange rate data and has the ability to simulate the trend of exchange rate well.

5. Conclusion

There has been an increasing interest in modeling and forecasting foreign exchange rate movements. In this paper, we improved the classical deep belief network (DBN) model by using continuous restricted Boltzmann machines (CRBMs) to construct a DBN. Consequently, this improved DBN is able to model continuous data. Then we propose the use of a DBN to tackle the exchange rate forecasting problem. Weekly GBP/USD, BRL/USD exchange rate series and weekly INR/USD exchange rate return series are used in the study. We apply a DBN combined with conjugate gradient method as alternative forecasting technique to the FFNN model in forecasting these three time series. The empirical results clearly suggest that DBN model outperforms FFNN model by all the five measures. Moreover, we also pay close attention to the stability of DBNs and FFNNs during exchange rate forecasting, by using variance to evaluate their performance. In out-of-sample forecasting, the DBN has not only superior predictive accuracy but also higher stability than FFNN.

DBN is an efficient method for exchange rate series forecasting. Using only weekly exchange rate data is the limitation of this study. Therefore, future work should attempt to evaluate the performance of a DBN in daily exchange rate forecasting.

Acknowledgements

This work is supported in part by the National Science Foundation of China under Grant nos. (61375064), (61373001), (61321491), and Jiangsu NSF Grant (BK20131279). Thanks to the reviewers and Ren Zhang for their helpful comments during the revising process.

References

[1] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals 2 (1989) 303–314.
[2] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2 (1989) 359–366.
[3] A.S. Weigend, B.A. Huberman, D.E. Rumelhart, Predicting sunspots and exchange rates with connectionist networks, Nonlinear Model. Forecast. (1992) 395–432.
[4] A.S. Weigend, D.E. Rumelhart, B.A. Huberman, Generalization by weight-elimination with application to forecasting, Adv. Neural Inf. Process. Syst. 3 (1991) 875–882.
[5] H. White. Economic prediction using neural networks: the case of IBM daily stock returns, in: Proceedings of the 2nd Annual IEEE Conference on Neural Networks, 1988, pp. 451–459.
[6] C.M. Kuan, T. Liu, Forecasting exchange rates using feedforward and recurrent neural networks, J. Appl. Econom. 10 (1995) 347–364.

[7] C. Panda, V. Narasimhan, Forecasting exchange rate better with artificial neural network, J. Policy Model. 29 (2007) 227–236.
[8] A. Emam. Optimal artificial neural network topology for foreign exchange forecasting, in: Proceedings of the 46th Annual Southeast Regional Conference on XX, New York, 2008, pp. 63–68.
[9] A.K. Nag, A. Mitra, Forecasting daily foreign exchange rates using genetically optimized neural networks, J. Forecast. 21 (2002) 501–511.
[10] G.E. Hinton, A fast learning algorithm for deep belief nets, Neural Comput. 18 (2006) 1527–1554.
[11] H. Lee, R. Grosse, R. Ranganath and A.Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations, in: Proceedings of the 26th International Conference on Machine Learning, 2009, pp. 609–616.
[12] M.M.R. Sazal, S.K. Biswas, M.F. Amin and K. Murase. Bangla handwritten character recognition using deep belief network, in: Proceedings of the 2013 International Conference on Electrical Information and Communication Technology (EICT), 2014, pp. 1–5.
[13] Guojie. Wenhao Huang, Haikun. Song, Kunqing Xie. Hong, Deep architecture for traffic flow prediction: deep belief networks with multitask learning, IEEE Trans. Intell. Transp. Syst. 15 (2014) 2191–2201.
[14] Ji Wu. Xiao-Lei Zhang, Deep belief networks based voice activity detection, IEEE Trans. Audio, Speech, Lang. Process. 21 (2013) 697–710.
[15] G. Hinton, R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (2006) 504–507.
[16] Jipeng Xie, Yan Yang, Tianrui Li, Weidong Jin, Learning features from high speed train vibration signals with deep belief networks, in: 2014 International Joint Conference on Neural Networks (IJCNN), 2014, pp. 2205–2210.
[17] M. Welling, M. Rosen-Zvi, G. Hinton, Exponential family harmoniums with an application to information retrieval, Adv. Neural Inf. Process. Syst. 17 (2005) 1481–1488.
[18] A.F. Murray, Novelty detection using products of simple experts-a potential architecture for embedded systems, Neural Netw. 14 (2001) 1257–1264.
[19] H. Chen, A.F. Murray, Continuous restricted Boltzmann machine with an implementable training algorithm, IEE Proc.-Vis. Image Signal Process. 120 (2003) 153–158.
[20] H. Chen, A.F. Murray, A continuous restricted Boltzmann machine with hardware-amenable learning algorithm, in: Proceedings of the 12th International Conference on Artificial neural networks (ICANN2002), 2002, pp. 358–363.
[21] B.J. Frey, Continuous sigmoidal belief networks trained using slice sampling, Adv. Neural Inf. Process. Syst. 9 (1997) 452–458.
[22] Z. Gioqinang, M.Y. Hu, Neural network forecasting of the British pound/US dollar exchange rate, Int. J. Manag. Sci. 26 (1998) 495–506.
[23] N. le Roux, Y. Bengio, Representational power of restricted Boltzmann machines and deep belief networks, Neural Comput. 20 (2008) 1631–1649.
[24] W. Hager, H. Zhang, A survey on nonlinear conjugate gradient methods, Pac. J. Optim. 2 (2006) 35–58.
[25] R. Fletcher, C. Reeves, Function minimization by conjugate gradients, Comput. J. 7 (1964) 149–154.

Furao Shen received the Engineering degree from Tokyo Institute of Technology, Tokyo, Japan, in 2006. Currently he is a professor at Nanjing University, China. His research interests include neural computing and robotic intelligence.

Jing Chao received the Master degree in Computer Science in 2012 from Nanjing University, China. Her research interests include neural computing and pattern recognition.

Jinxi Zhao received the Ph.D. from Nanjing University, China. Currently he is a professor at Nanjing University, China. His research interests include numerical analysis and neural computing.