Forecasting Exchange Rate with Deep Belief Networks

Jing Chao, Furao Shen, and Jinxi Zhao

***Abstract*—Forecasting exchange rates is an important financial problem which has received much attention. Nowadays, neural network has become one of the effective tools in this research field. In this paper, we propose the use of a deep belief network (DBN) to tackle the exchange rate forecasting problem. A DBN is applied to predict both British Pound/US dollar and Indian rupee/US dollar exchange rates in our experiments. We use six evaluation criteria to evaluate its performance. We also compare our method to a feedforward neural network (FFNN), which is the state-of–the-art method for forecasting exchange rate with neural networks. Experiments indicate that deep belief networks (DBNs) are applicable to the prediction of foreign exchange rate, since they achieve better performance than feedforward neural networks (FFNNs).**

I. 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 much researchers' attention. For many years researchers have used linear techniques since it was very simple. However, it was not so 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] compare 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] is the first to apply NN models in the financial domain. Kuan and Liu [6] use 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] conduct 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] proves that neural networks are efficient and profitable in forecasting exchange rate in particular, 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] use 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.

DBN is a generative neural network model with many hidden layers, introduced by Hinton et al. [10] along with a greedy layer-wise learning algorithm. 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], dimensionality-reduction [12], and information retrieval [13].

This paper aims to present and evaluate the DBN performance as a forecasting tool on predicting exchange rates. In experiments, we use British Pound/US dollar (GBP/USD) and Indian rupee/US dollar (INR/USD) exchange rates data sets and six evaluation criteria to evaluate the performances. We also compare our method to FFNNs. The experimental results show that deep belief networks are applicable to the prediction of foreign exchange rates, since they achieve better performance than FFNNs.

II. Deep Belief Networks For Time Series Forecasting

A. Deep Belief Networks

A DBN is a feedforward neural network with many hidden layers. An example of a DBN is shown in Figure 1. It consists

J. Chao (yoyo72-@163.com), F. Shen (frshen@nju.edu.cn) and J. Zhao (jxzhao@nju.edu.cn) are with the State Key Laboratory for Novel Software Technology at Nanjing University, Nanjing, 210093, P.R. China.

978-1-4244-9637-2/11/$26.00 ©2011 IEEE

of an input layer which contains the input units (called visible units), a 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 et al. [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 Figure 2.

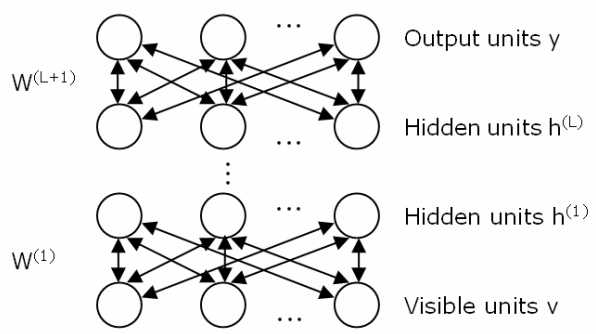

Fig. 1. DBN architecture with L hidden layers.

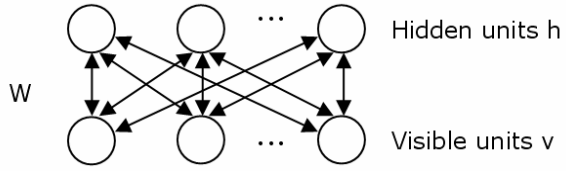

Fig. 2. RBM architecture.

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

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

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

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

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

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 [14]. However, this kind of RBM tends to generate continuous data with high symmetry. Chen and Murray [15] 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 use a CRBM instead of a RBM in our experiments.

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

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

with

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

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 [16]. Such behavior is similar to the noisy unit in [17]. The update equations for w_{ij} and a_j are

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

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

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 train sequentially as many RBMs as the number of hidden layers in the DBN. These RBMs 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.

B. 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 $x_1, x_2, \cdots, x_n$ are n time-lagged observations of the exchange rate in the training set. We can 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 $x_1, x_2, \cdots, x_p$ as the inputs and x_{p+1} as the output. In the second training sample, $x_2, x_3, \cdots, x_{p+1}$ 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.

III. EXPERIMENT

A. Data

We use two data sets of exchange rates in 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. 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 130 observations are used as test samples. The two 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.

B. Performance measures

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:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{T}(y_i - y_i')^2}{T}} \tag{8}$$

$$MAE = \frac{\sum_{i=1}^{T}|y_i - y_i'|}{T} \tag{9}$$

$$MAPE = \frac{1}{T}\sum_{i=1}^{T}\left|\frac{y_i - y_i'}{y_i}\right| \times 100 \tag{10}$$

$$DA = \frac{1}{T}\sum_{i=1}^{T} a_i,$$

where $a_i = \begin{cases} 1 & \text{if } (y_{i+1} - y_i)(y_{i+1}' - y_i) > 0 \\ 0 & \text{otherwise} \end{cases}$ (11)

$$CORR = \frac{E(YY') - E(Y)E(Y')}{\sqrt{E(Y^2) - E^2(Y)}\sqrt{E(Y'^2) - E^2(Y')}} \tag{12}$$

where $Y = (y_1, y_2, \cdots, y_T)^T$ is a vector of actual observations, $Y' = (y_1', y_2', \cdots, y_T')^T$ is a vector of predicted values, and T is the number of predictions. These criteria are frequently used performance measures in research. Furthermore, we use variance of 50 runs to evaluate the performance of forecasting exchange rate with DBNs. RMSE, MAE and MAPE are three popular criteria in the literature, but they are all mean based measures. They can not 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 variance of the three criteria are calculated to show the forecasting stability of the DBN model.

C. Experimental design

The major purpose of this study is to evaluate the performance of DBNs in forecasting two exchange rates. One is a weekly GBP/USD exchange rate series, the other is weekly exchange rate return series of INR/USD. We normalize the exchange rate return of INR/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.

We will 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 the design of a neural network, the numbers of input and hidden nodes 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, 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 not a 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 are 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 [18] 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. In order to compare the forecasting performance of DBNs and FFNNs, the optimal architecture of FFNN is also selected through experimentation. 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. Since we choose the one-step-ahead forecasts, one output node is enough to satisfy our needs.

There are still some parameters to be determined in a DBN model. The learning rates η_w in equation (6), a_j in equation (7) and constant σ in equation (4) should be given by users. In this paper, we determine these values through 5-fold cross-validation. θ_L and θ_H in equation (5) are set to be the minimum and maximum of a training sample respectively. During training period, all the training samples are selected randomly to train the DBN. 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. But at the start, we try to use fixed values. We let θ_L and θ_H be the minimum and maximum of the entire training data before training the DBN. In order to compare these two methods, we apply a DBN to both GBP/USD exchange rate and INR/USD exchange rate return series forecasting. Then changeable and fixed methods are applied to train this DBN respectively by using the same time series dataset. We try ten levels of input node (from one to ten) in combination with each five hidden node (4, 8, 12, 16, 20), and calculate the average of 10 runs. Table I shows the differences between using changeable values and fixed ones in training performance of a DBN with 6 input nodes and 8 hidden nodes. All the values are average calculated

TABLE I
EFFECTS OF CHANGEABLE AND FIXED VALUES ON THE TRAINING PERFORMANCE

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 II
EFFECTS OF DBN FACTORS ON THE TRAINING PERFORMANCE (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	0.00764-	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**

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. Changeable values method is more stable and reliable in training a DBN, since DBNs with other architecture also get the similar results. 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 equation (4) and a_j in equation (5) also play a role in performance of a DBN. In order to achieve the global optima, 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 are employed to fine-tune the weights for optimal reconstruction.

IV. Results

Table II shows the effects of input nodes and hidden nodes on DBN training performance of GBP/USD exchange rate forecasting. ALL the values of three measures listed in the table, at each level of input node in combination with each five hidden node are 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, which is improved by 3.8% than that of 6 input nodes. In addition, as the number of input nodes increases, except for input nodes 3 and 7, the average of RMSEs at each level of input node decreases. But we do not find a clear hidden node effect. Since the best RMSE at 6 input nodes level occurs at 16 hidden nodes, we try to use a DBN with two hidden layers, which has 6 input nodes, 16 hidden nodes in the first hidden layer combined with different hidden nodes in the second hidden layer. The experimental results are shown by Table III. Obviously, the best RMSE and MAPE occur at 8 hidden nodes, which are 0.008709 and 1.028106. Besides, the best performance of MAE is at the 10 hidden nodes level, which is 0.006587.

TABLE III
EFFECTS OF HIDDEN NODES IN THE SECOND HIDDEN LAYER ON THE TRAINING PERFORMANCE (GBP/USD)

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 IV
EFFECTS OF DBN FACTORS ON THE TRAINING PERFORMANCE (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**

Therefore, the number of hidden nodes in the second hidden layer is set to 8.

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 [19] suggest that several hidden layers is 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. Since we have chosen the number of hidden nodes in the second hidden layer, we turn to consider a DBN with three hidden layers. It has 6 input nodes, 16 and 8 hidden nodes in first two hidden layers respectively combined with different hidden nodes in the third one. Then the average predictive error of all the combinations is calculated and compared to the cases of one and two hidden layers.

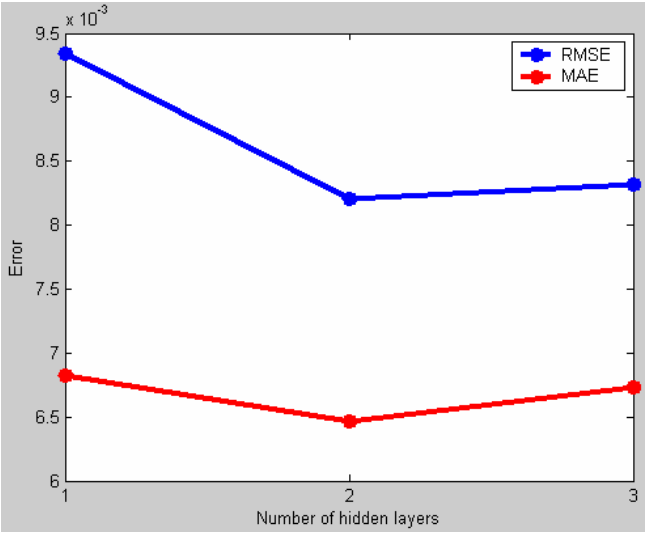

Fig. 3. Effects of the number of hidden layers on RMSE and MAE.

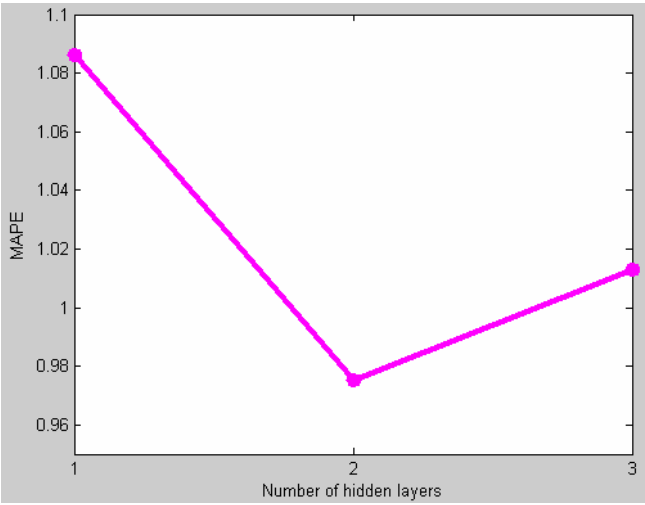

Fig. 4. Effects of the number of hidden layers on MAPE.

Figure 3. 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 by Figure 4., which are 1.0863, 0.9753 and 1.0129 respectively. These two figures show that the predictive error first decreased and then increased as the number of hidden layer increases. Therefore we choose a DBN with two hidden layers in out-of-sample forecasting. 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.

TABLE V
EFFECTS OF HIDDEN NODES IN THE SECOND HIDDEN LAYER ON THE TRAINING PERFORMANCE (INR/USD)

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

The effects of input and hidden nodes on DBN training performance of INR/USD exchange return forecasting are shown in Table IV. It gives similar results as forecasting weekly exchange rate of GBP/USD. Except for input nodes 5, 7 and 9, the average of RMSEs at each level of input node decreases as the number of input nodes increases. Moreover, the best average performance across RMSE, MAE and MAPE all occurs at 10 input nodes. But in Table IV there is not a clear hidden node effect either. 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 3 hidden nodes has better performance than all the other levels in three measures. The values of RMSE, MAE and MAPE respectively are 0.016310, 0.012188 and 2.923540. Then two hidden layers is taken into consideration. After trying several combinations, we find 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 training performance of different hidden nodes in the second hidden layer is shown by Table V. 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 forecasting weekly

exchange rate 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.

Next, we turn to out-of-sample analysis to examine the forecasting power of DBNs. We conduct a comparative study between DBNs and FFNNs in exchange rate forecasting by using the same time series data.

TABLE VI
OUT-OF-SAMPLE RESULTS ON WEEKLY GBP/USD SERIES

Method	RMSE	MAE	MAPE	DA	CORR
DBN	8.2099E-3	6.4630E-3	0.9753	0.5755	0.8733
FFNN	9.4094E-3	7.3788E-3	1.1132	0.4174	0.8459

The results for out-of-sample performance on weekly GBP/USD series are presented in Table VI. All the values listed in the table are mean values through 50 runs. In Table VI, we can see that DBN gives better predictive accuracy than FFNN in all the five criteria. The correlation coefficient for DBN is 0.8733, compared with that of FFNN, which is 0.8459. Hence the correlation of predictive series generated by DBN and actual one is higher than the corresponding figure 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 VI 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 VII
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

In order to test the stability of these two models, we calculate the variance of RMSE, MAE and MAPE in 50 runs. Table VII 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 the errors of exchange rate prediction with DBNs are more stable through 50 runs. That is, DBNs are more superior in stability and reliability to FFNNs in exchange rate forecasting.

TABLE VIII
OUT-OF-SAMPLE RESULTS ON WEEKLY INR/USD EXCHANGE RATE RETURN SERIES

Method	RMSE	MAE	MAPE	DA	CORR
DBN	1.6505E-2	1.2387E-2	2.9687	0.5873	0.3677
FFNN	1.8899E-2	1.4250E-2	3.4292	0.5407	0.1481

The out-of-sample performance of weekly INR/USD exchange rate return prediction is presented in Table VIII. All the values are mean values through 50 runs as before. 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 equal to 0.3677, with the actual series as compared to the values of FFNN (0.1481). In Table VIII, we can see that DBN gives better direction accuracy, which is 0.5873, than FFNN model (0.5407). Table IX 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 variance of DBN precede those of FFNN model. The variance of RMSE, MAE and MAPE are 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 IX
VARIANCE OF RMSE AND MAE 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

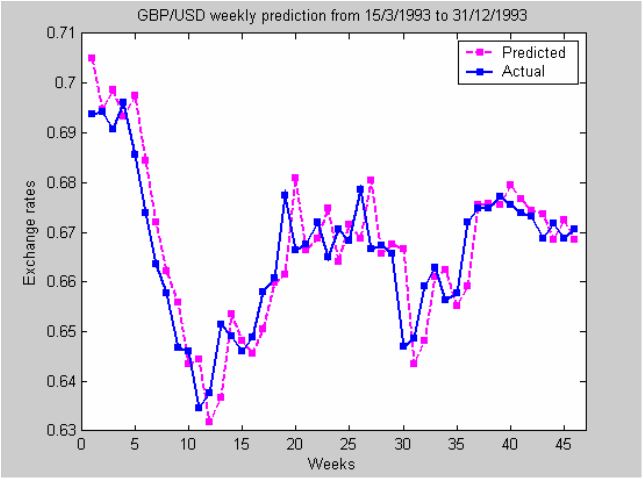

Fig. 5. GBP/USD weekly predictions with DBNs.

Figure 5. 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. Furthermore, as shown in Figure 6., 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, compared with 0.4132 predicted by DBN. In a word, the DBN learns well from the exchange rate data and has the ability to simulate the trend of exchange rate well.

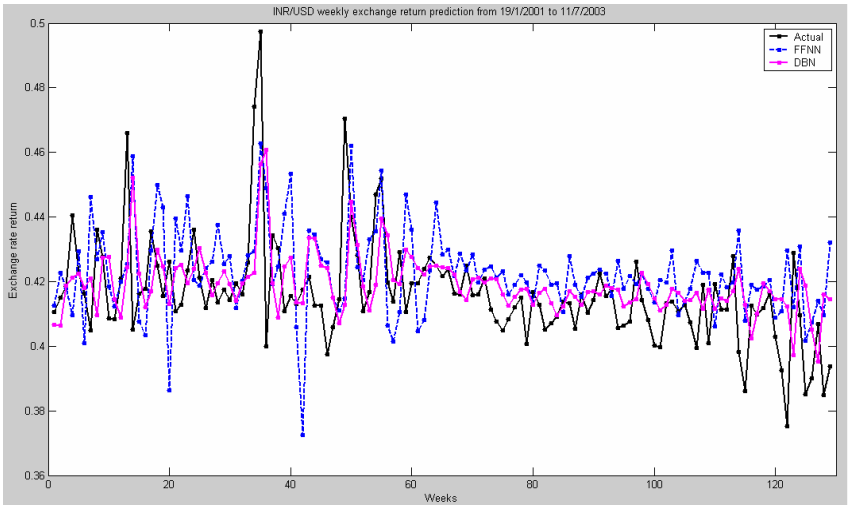

Fig. 6. The predicted output of the DBN and FFNN models against the actual weekly exchange rates return series of INR/USD.

V. Conclusion

There has been an increasing interest in modeling and forecasting foreign exchange rate movements. In this paper, we propose the use of a deep belief network (DBN) to tackle the exchange rate forecasting problem. Both weekly GBP/USD exchange rate series and weekly INR/USD exchange rate return series are used in the study. We apply a DBN as alternative forecasting technique to the FFNN model in forecasting these two 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 limitation of this study. Therefore, future work should attempt to evaluate the performance of a DBN in daily exchange rate forecasting. And several experiments will be conducted for different currencies.

Acknowledgments

This work was supported in part by China NSF grant (#60975047, #60723003, #60721002), Jiangsu NSF grant (#BK2009080), and 973 Program (2010CB327903).

References

[1] G. Cybenko, "Approximation by superposfitions of a sigmoidal function," *Mathematical Control Signals*, vol. 2, pp. 303–314, 1989.

[2] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, 1989.

[3] A. S. Weigend, B. A. Huberman, and D. E. Rumelhart, "Predicting sunspots and exchange rates with connectionist networks," in *Nonlinear Modeling and Forecasting*, M. Casdagli and S. Eubank. AddisonWelsey, Ed. Redwood City, 1992, pp. 395-432.

[4] A. S. Weigend, D. E. Rumelhart, and B. A. Huberman, "Generalization by weight-elimination with application to forecasting," *Advances in Neural Information Processing Systems*, vol. 3, pp. 875-882, 1991.

[5] H. White, "Economic prediction using neural networks: The case of IBM daily stock returns," in *Proc. 2nd Annu. IEEE Conf. on Neural Networks*, San Diego, 1988, pp. 451-458.

[6] C. M. Kuan and T. Liu, "Forecasting exchange rates using feedforward and recurrent neural networks," *Journal of Applied Econometrics*, vol. 10, pp. 347-364, 1995.

[7] C. Panda and V. Narasimhan, "Forecasting exchange rate better with artificial neural network," *Journal of Policy Modeling*, vol. 29, pp. 227-236, 2007.

[8] A. Emam, "Optimal artificial neural network topology for foreign exchange forecasting," in *Proc. 46th Annu. Southeast Regional Conf. on XX*, New York, 2008, pp. 63-68.

[9] A. K. Nag and A. Mitra, "Forecasting daily foreign exchange rates using genetically optimized neural networks," *Journal of Forecasting*, vol. 21, pp. 501-511, 2002.

[10] G. E. Hinton, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, pp.1527-1554, 2006.

[11] H. Lee, R. Grosse, R. Ranganath and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proc. 26th Int. Conf. on Machine Learning*, Montreal, 2009, pp. 609-616.

[12] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, pp. 504-507, 2006.

[13] M. Welling, M. Rosen-Zvi and G. Hinton, "Exponential family harmoniums with an application to information retrieval," Advances in Neural Information Processing Systems, vol. 17, pp. 1481-1488, 2005.

[14] A. F. Murray, "Novelty detection using products of simple experts—a potential architecture for embedded systems," *Neural Networks*, vol. 14, pp. 1257-1264, 2001.

[15] H. Chen and A.F. Murray, "Continuous restricted Boltzmann machine with an implementable training algorithm," *IEE Proc-Vis. Image Signal Process*., vol. 120, pp. 153-158, 2003.

[16] H. Chen, and A. F. Murray, "A continuous restricted Boltzmann machine with hardware-amenable learning algorithm," in Proc. 12th Int. Conf. on Artificial neural networks (ICANN2002), Madrid, 2002, pp. 358-363.

[17] B. J. Frey, "Continuous sigmoidal belief networks trained using slice sampling," *Advances in Neural Information Processing Systems*, vol. 9, pp. 452-458, 1997.

[18] Z. Gioqinang and M. Y. Hu, "Neural network forecasting of the British pound/US dollar exchange rate," International Journal of Management Sxience, vol. 26, pp. 495-506, 1998.

[19] N. Le Roux and Y. Bengio, "Representational power of restricted Boltzmann machines and deep belief networks," *Neural Computation*, vol. 20, pp. 1631–1649, 2008.