

# Multistage Neural Network Metalearning with Application to Foreign Exchange Rates Forecasting

Kin Keung Lai<sup>1,2</sup>, Lean Yu<sup>2,3</sup>, Wei Huang<sup>4</sup>, and Shouyang Wang<sup>1,3</sup>

<sup>1</sup> College of Business Administration, Hunan University, Changsha 410082, China

<sup>2</sup> Department of Management Sciences, City University of Hong Kong,  
Tat Chee Avenue, Kowloon, Hong Kong  
{mskk lai, msyulean}@cityu.edu.hk

<sup>3</sup> Institute of Systems Science, Academy of Mathematics and Systems Science,  
Chinese Academy of Sciences, Beijing 100080, China  
{yulean, sywang}@amss.ac.cn

<sup>4</sup> School of Management, Huazhong University of Science and Technology,  
1037 Luoyu Road, Wuhan 430074, China

**Abstract.** In this study, we propose a multistage neural network metalearning technique for financial time series predication. First of all, an interval sampling technique is used to generate different training subsets. Based on the different training subsets, the different neural network models with different training subsets are then trained to formulate different base models. Subsequently, to improve the efficiency of metalearning, the principal component analysis (PCA) technique is used as a pruning tool to generate an optimal set of base models. Finally, a neural-network-based metamodel can be produced by learning from the selected base models. For illustration, the proposed metalearning technique is applied to foreign exchange rate predication.

## 1 Introduction

Artificial neural networks (ANNs), first introduced in 1943 [1], is a system derived through neuropsychology models [2]. It attempts to emulate the biological system of the human brain in learning and identifying patterns. Moreover, ANNs can more aptly recognize poorly defined patterns. Instead of extracting explicit rules from sample data, the ANNs employed a learning algorithm to autonomously: (a) extract the functional relationship between input and output, which is embedded in a set of historical data (called training exemplars or learning samples), and (b) encode it in connection weights. Training exemplars that are readily available allow neural networks to capture a large volume of information in a rather short period of time and to continuously learn throughout their lifespan. Furthermore, neural networks have the ability to not only deal with noisy, incomplete, or previously unseen input patterns, but to also generate a reasonable response [3]. However, ANNs are far from being optimal learner. For example, the existing studies, e.g., [4] have found that the ways neural networks have of getting to the global minima vary and some networks just settle into local minima instead of global minima through the analysis of error distributions. In this case, it is hard to justify which neural network's error reaches the global minima if the error rate is not zero. Thus, it is not wise choice that only selecting a single

neural network model with the best generalization from a limited number of neural networks if the error is larger than zero. For example of financial time series forecasting, it is difficult to obtain a consistently good result by using a single neural network model due to high volatility and irregularity in financial markets. More and more researchers have realized that just selecting the neural network model that gives the best performance will result in losses of potentially valuable information contained by other neural network models with slightly weak performance relative to the best neural network model. Naturally, a different learning strategy to solving these problems that considers those discarded neural networks whose performance is less accurate as the best neural network model should be proposed. In such situations, metalearning strategy [5] based on the neural network is introduced.

Metalearning [5], which is defined as learning from learned knowledge, provides a promising solution and a novel approach to the above challenges. The idea is to use neural network learning algorithms to extract knowledge from several different data sets and then use the knowledge from these individual learning algorithms to create a unified body of knowledge that well represents the entire data. Therefore metalearning seeks to compute a metamodel that integrates in some principled fashion the separately learned models to boost overall predictive accuracy. In this study, a four-stage neural-network-based metalearning technique is proposed for financial time series forecasting. In the first stage, an interval sampling technique is used to generate different training sets. Based on the different training sets, the different neural network models with different initial conditions are then trained to formulate different base models in the second stage. In the third stage, to improve the efficiency of metalearning, the principal component analysis (PCA) technique is used as a pruning tool to generate an optimal set of base models. In the final stage, a neural-network-based metamodel can be produced by learning from the selected base models.

The rest of this study is organized as follows. Section 2 provides a neural-network-based metalearning process in detail. For verification, an exchange rate predication experiment is performed in Section 3. Finally, Section 4 concludes the article.

## 2 The Neural-Network-Based Metalearning Process

In this section, we first introduce the basic knowledge of metalearning. Based on the metalearning, a generic metamodeling process with three phases is then provided.

As Section 1 mentioned, metalearning [5], which is defined as learning from learned knowledge, is an emerging technique recently developed to construct a metamodel that deals with the problem of computing a metamodel from multiple training data sets. Broadly speaking, learning is concerned with finding a model  $f = f_a[j]$  from a single training set  $TR_j$ , while metalearning is concerned with finding a metamodel  $f = f_a$  from several training sets  $\{TR_1, TR_2, \dots, TR_n\}$ , each of which has an associated model  $f = f_a[j]$ . The  $n$  individual models derived from the  $n$  training sets may be of the same or different types. Similarly, the metamodel may be of a different type than some or all of the single models. Also, the metamodel may use data from a meta-training set ( $MT$ ), which are distinct from the data in the individual training set  $TR_j$ .

There are two types of metalearning methods: different-training-set-based metalearning and different-model-type-based metalearning. For the first type, we are

given a large data set  $DS$  and partition it into  $n$  different training subsets  $\{TR_1, TR_2, \dots, TR_n\}$ . Assume that we build a separate model on each subset independently to produce  $n$  models  $\{f_1, f_2, \dots, f_n\}$ . Given a feature vector  $x$ , we can produce  $n$  scores  $(f_1(x), f_2(x), \dots, f_n(x))$ , one for each model. Given a new training set  $MT$ , we can build a metamodel  $f$  on  $MT$  using the data  $\{(f_1(x), f_2(x), \dots, f_n(x), y) : (x, y) \text{ in } MT\}$ .

For the second type, given a relative small training set  $\{TR\}$ , we replicate it  $n$  times to produce  $n$  training sets  $\{TR_1, TR_2, \dots, TR_n\}$  and create different models  $f_j$  on each training set  $TR_j$ , for example, by training the replicated data on  $n$  different-type models. For simplicity, assume that these models are binary classifiers so that each classifier takes a feature vector and produces a classification in  $\{0, 1\}$ . We can then produce a metamodel simply by using a majority vote of the  $n$  classifiers.

In time series forecasting, data is abundant. To improve the predication performance, we use the different-training-set-based metalearning. Generally, the generic idea of neural network metalearning is to generate a number of independent models (i.e., base models) by applying neural network learning algorithms to a collection of data sets. The independent models are then selected and combined to obtain a global model or metamodel. Fig. 1 illustrates a neural network metalearning process. From Fig. 1, the metalearning process consists of four stages, which can be described below.

**Stage 1:** For an initial data set  $DS$ , the whole data set is first divided into training set  $TR$  and testing set  $TS$ . Then the different training subsets  $TR_1, TR_2, \dots, TR_n$  are created from  $TR$  with certain sampling algorithm.

**Stage 2:** For each training subset  $TR_i$  ( $i = 1, 2, \dots, n$ ), the neural network model  $f_i$  ( $i = 1, 2, \dots, n$ ) is trained by the specific learning algorithm to formulate  $n$  different base models. After training, the testing data was applied for assessment.

**Stage 3:** For  $n$  different base models, the pruning techniques are used to generate some effective base models. After pruning,  $m$  ( $m \leq n$ ) different base models are generated for the next stage's use.

**Stage 4:** Using the whole training data set to  $m$  different base models, different neural network's results can formulate a meta-training set ( $MT$ ). Based on the meta-training set, another single neural network model is training to produce a metamodel.

From the generic neural network metalearning process, there are **four problems** to be further addressed, i.e., (a) how to create  $n$  different training subset from the original training data set  $TR$ ; (b) how to create different base models  $f_i$  with different training subsets for the same-type model; (c) how to select some effective base model from many neural network base models in the previous stage; and (d) how to formulate a metamodel with different results produced by different base models.

#### A. Data Sampling

For financial time series predication mining tasks, our aim is to improve the predication performance with historical time series data. For convenience of neural network learning, the original data set  $DS$  is first divided into two parts: training data set  $TR$  and testing data set  $TS$ . In order to capture the patterns of time series data, different data sampling is helpful for neural network learning. Here we propose an interval sampling algorithm to produce different training subsets.

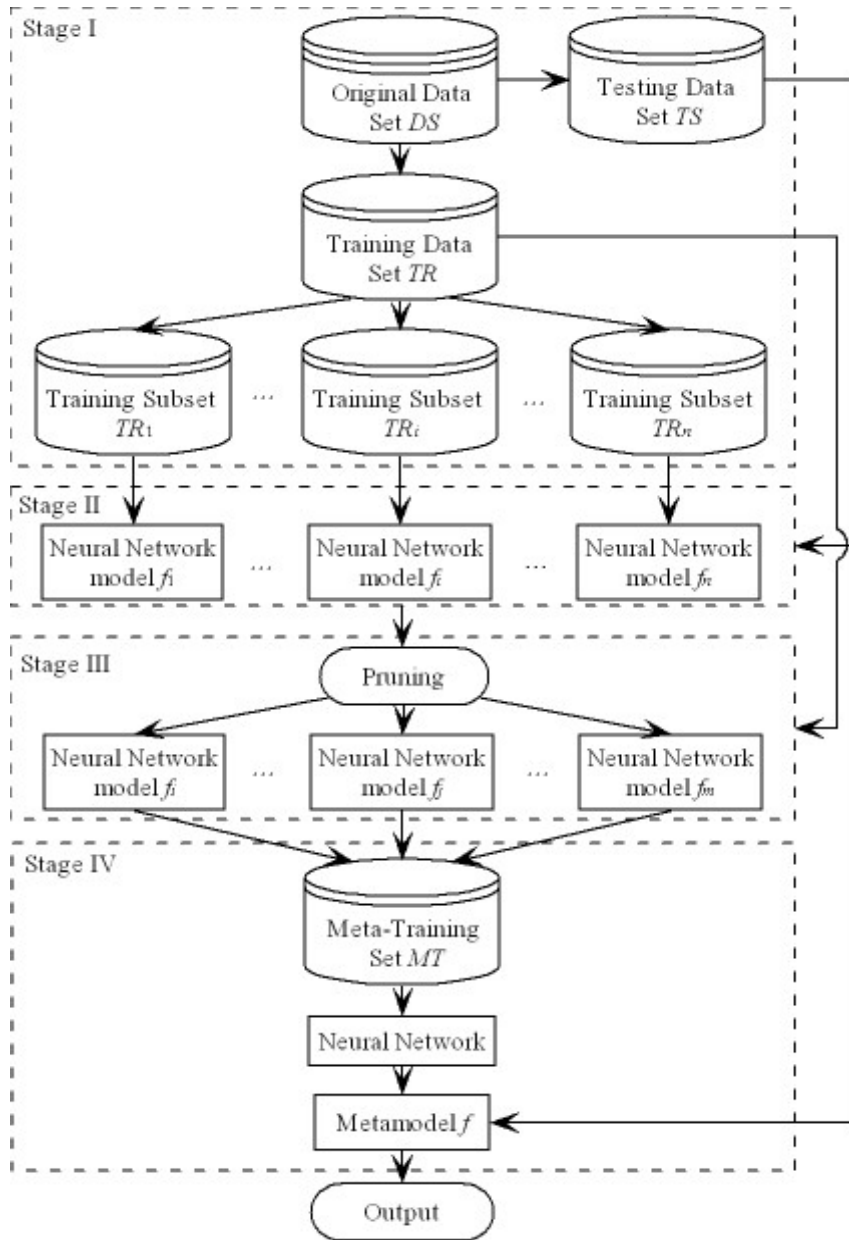
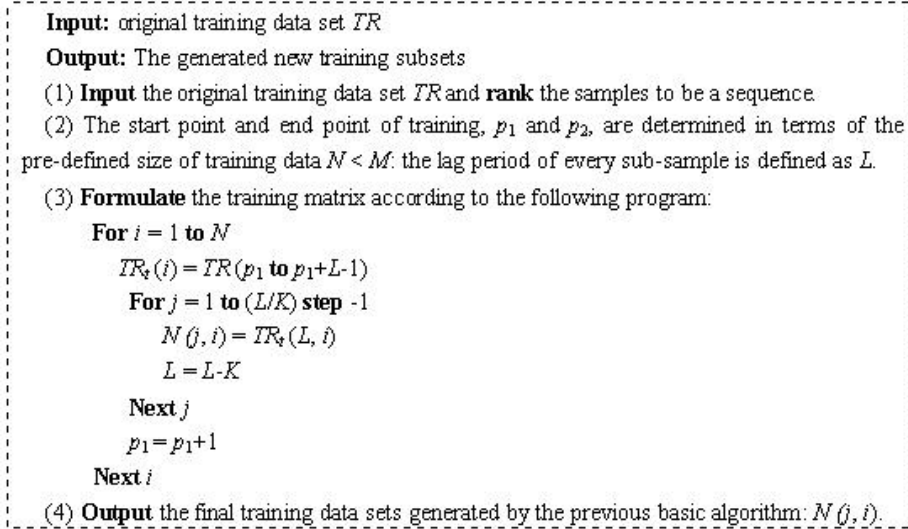


Fig. 1. The generic neural network metalearning process

Given that the size of the original training set  $TR$  is  $M$ , the size of new training set is  $N$ , and sampling interval is  $K$ , the interval sampling algorithm is illustrated in Fig. 2.



**Fig. 2.** The bagging algorithm

With the above interval sampling algorithm, we can obtain different training subsets only through varying the sampling interval or starting point of each time series.

### ***B. Individual neural network base model creation***

With the work about bias-variance trade-off [4], a metamodel consisting of diverse models with much disagreement is more likely to have a good performance. Therefore, how to create the diverse base model is the key path to the creation of an effective metamodel. For neural network model, there are several methods for generating diverse models.

- (1) Initializing different starting weights for each neural network models for different training subsets.
- (2) Varying the architecture of neural network, e.g., changing the different numbers of layers or different numbers of nodes in each layer.
- (3) Using different training algorithms, such as the back-propagation [6-7], radial-basis function [8], and Bayesian regression [9] algorithms.

In this study, the individual neural network models with different training subsets are therefore used as base models  $f_1, f_2, \dots, f_m$ , as illustrated in Fig. 1.

When a large number of neural network base models are generated, it is necessary to select the appropriate number of component models for improving the efficiency of neural network metalearning system. It is well known to us that not all circumstances are satisfied with the rule of “the more, the better” [11]. That is, some individual base models produced by this phase may be redundant, wasting resources and reducing system performance. Thus, it is necessary to prune some inappropriate individual base models for metamodel construction.

### C. Neural network base model pruning

In order to select the appropriate number of neural network base model, we utilize the principal component analysis (PCA) to arrive at this goal.

The PCA technique [10], an effective feature extraction method, is widely used in signal processing, statistics and neural computing. The basic idea in PCA is to find the components  $(s_1, s_2, \dots, s_p)$  that can explain the maximum amount of variance possible by  $p$  linearly transformed components from data vector with  $q$  dimensions. The mathematical technique used in PCA is called eigen analysis. In addition, the basic goal in PCA is to reduce the dimension of the data (Here the PCA is used to reduce the number of individual models). Thus, one usually chooses  $p \leq q$ . Indeed, it can be proven that the representation given by PCA is an optimal linear dimension reduction technique in the mean-square sense [10]. Such a reduction in dimension has important benefits. First, the computation of the subsequent processing is reduced. Second, noise may be reduced and the meaningful underlying information identified. The following presents the PCA process for individual model selection [11].

Assuming that there are  $n$  individual data mining models and that every model contains  $m$  forecasting or classification results, then result matrix ( $Y$ ) is represented as

$$Y = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1m} \\ y_{21} & y_{22} & \cdots & y_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nm} \end{bmatrix} \quad (1)$$

where  $y_{ij}$  is the  $j$ th predication or classification result with the  $i$ th data mining model.

Next, we deal with the result matrix using the PCA technique. First, eigenvalues  $(\lambda_1, \lambda_2, \dots, \lambda_n)$  and corresponding eigenvectors  $A=(a_1, a_2, \dots, a_n)$  can be solved from the above matrix. Then the new principal components are calculated as

$$Z_i = a_i^T Y \quad (i=1, 2, \dots, n) \quad (2)$$

Subsequently, we choose  $m$  ( $m \leq n$ ) principal components from existing  $n$  components. If this is the case, the saved information content is judged by

$$\theta = (\lambda_1 + \lambda_2 + \cdots + \lambda_m) / (\lambda_1 + \lambda_2 + \cdots + \lambda_n) \quad (3)$$

If  $\theta$  is larger than a specified threshold, enough information has been saved after the feature extraction process. Thus, some redundant base models can be pruned. Through applying the PCA technique, we can obtain the appropriate numbers (e.g.,  $m$  in this case) of base models for metamodel generation.

### D. Neural-network-based metamodel generation

Once the appropriate numbers of base models are selected, applying the whole data set to these selected base models can produce a set of neural network output. These neural network outputs can formulate a new training set called as ‘‘meta-training set ( $MT$ )’’. In order to reflect the principle of metalearning, we utilize another neural network model to generate a metamodel by learning from the meta-training set ( $MT$ ). That is, we use another neural network model to learn the relationship between base models by taking the outputs of the selected base models as input, combined with

their targets or expected values. That is, the neural-network-based metamodel can be viewed as a nonlinear information processing system that can be represented as

$$\hat{f} = g(\hat{f}_1, \hat{f}_2, \dots, \hat{f}_m) \quad (4)$$

where  $(\hat{f}_1, \hat{f}_2, \dots, \hat{f}_m)$  is the output of individual neural network predictors,  $\hat{f}$  is the aggregated output,  $g(\cdot)$  is nonlinear function determined by neural network. In this sense, neural network learning algorithm is used as a meta-learner (*ML*) shown in Fig. 1 for metamodel generation. Of course, other learning algorithm, e.g., support vector machine regression, can also be used as a meta-learner, as proposed by Lai et al. [12].

In summary, the proposed metamodel can actually be seen as an embedded neural network system with two-layer neural network models, as illustrated in Fig. 1. Suppose that there is an original data set *DS* which is divided into two parts: training set (*TR*) and testing set (*TS*). The training set is usually preprocessed by various sampling methods (e.g., interval sampling here) in order to generate diverse training subsets  $\{TR_1, TR_2, \dots, TR_n\}$  before they are applied to the first layer's neural network learners:  $L_1, L_2, \dots, L_n$ . After training, the diverse neural network models (i.e., base models),  $f_1, f_2, \dots, f_n$  are generated. Through PCA-based pruning, a set of selected base model are obtained. Afterwards the whole training set *TR* was applied and the corresponding results  $(\hat{f}_1, \hat{f}_2, \dots, \hat{f}_m)$  of each selected base model in the first layer were used as inputs of the second layer neural network model. This neural network model in the second layer can be seen as a meta-learner (*ML*). By training, the neural-network-based metamodel can be generated. Using the testing set *TS*, the performance of the neural-network-based metamodel can be assessed.

### 3 Experimental Analysis

#### 3.1 Research Data and Experiment Design

The research data used in this study is euro against dollar (EUR/USD) exchange rate. This data are daily and are obtained from Pacific Exchange Rate Service (<http://fx.sauder.ubc.ca/>), provided by Professor Werner Antweiler, University of British Columbia, Vancouver, Canada. The entire data set covers the period from January 1 1993 to December 31 2004 with a total of 3016 observations. The data sets are divided into two periods: the first period covers from January 4 1993 to December 31 2002 while the second period is from January 1 2003 to December 31 2004. The first period, which is assigned to in-sample estimation, is used to network learning and training. The second period is reserved for out-of-sample evaluation, which is for the testing purposes. In addition, the training data covered from January 1 1993 to December 31 2002 are divided into ten training subsets by varying the sampling interval ( $K = 1, 2, \dots, 10$ ) for this experiment. Using these different training subsets, different neural network base models with different initial weights are presented. For neural network base models, a three-layer back-propagation neural network with 10 TANSIG neurons in the hidden layer and one PURELIN neuron in the output layer is used. The network training function is the TRAINLM. For the neural-network-based

metamodel, a similar three-layer back-propagation neural network (BPNN) with 10 inputs neurons, 8 TANSIG neural in the second layer and one PURELIN neuron in the final layer is adopted for metamodel generation. Besides, the learning rate and momentum rate is set to 0.1 and 0.15. The accepted average squared error is 0.05 and the training epochs are 1800. The above parameters are obtained by trial and error.

To evaluate the performance of the proposed neural-network-based metamodel, several typical financial time series predication models, the autoregressive integrated moving average (ARIMA), individual BPNN with optimal learning rates [16-17] and support vector machine (SVM), are selected as benchmarks. In the individual BPNN model, a three-layer back-propagation neural network with 5 input nodes, 8 hidden nodes and 1 output nodes is used. The hidden nodes use sigmoid transfer function and the output node uses the linear transfer function. In the SVM, the kernel function is Gaussian function with regularization parameter  $C = 50$  and  $\sigma^2 = 5$ . Similarly, the above parameters are obtained by trial and error.

For further comparison of the performance of neural network metamodel, two hybrid model proposed by [11] and [13], and three metalearning approaches, i.e., simple averaging based metamodel [14-15], weighted averaging based metamodel [14-15] and support vector machine regression (SVMR) based metamodel [12] are also used for foreign exchange rates predication. Actually, these two metalearning approaches are two neural network ensemble methods. For simple averaging approach, the final metamodel can be obtained by averaging the sum of each output of the neural network base models. Weighted averaging is where the final metamodel is calculated based on individual base model's performances and a weight attached to each base model's output. The gross weight is 1 and each base model is entitled to a portion of this gross weight according to their performance or diversity. For more details about these two metalearning techniques, please refer to [12, 14-15]. Finally, the root mean square error (*RMSE*) and direction change statistics ( $D_{stat}$ ) [11] of financial time series are used as performance evaluation criteria in terms of testing set.

### 3.2 Experiment Results

According to the experiment design, different predication models with different parameters can be built. For comparison, the ARIMA model, individual BPNN, individual SVM, simple averaging based metamodel and weighted averaging based metalearning approach, are also performed. The results are reported in Table 1.

As can be seen from Table 1, we can find the following conclusions from the general view. First of all, all metamodels listed in this study perform better than individual models and hybrid models in terms of both *RMSE* and  $D_{stat}$ , indicating that the metamodel is a promising solution to the foreign exchange rates predication. The possible reason is that the metamodel can get more information from different base models and thus increase predication accuracy. Second, of the four metamodels, the SVMR based metamodel is the best in terms of *RMSE*. The possible reason is that SVMR can overcome some shortcomings of neural networks, such as local minima and overfitting, due to structural risk minimization principle of SVM. However, the neural network based metamodel perform the best from the  $D_{stat}$  perspective. The reason is still unknown and is worth exploring further. Third, the performance of two hybrid models seems to be better than those of three individual models. The main

reason is their complementarities between the hybrid models. Fourth, the ARIMA model is the worst of the nine models for both  $RMSE$  and  $D_{stat}$  in this case. The inherent reason is that the ARIMA model is difficult to capture the nonlinear patterns of financial time series since ARIMA is a class of linear model and the financial time series contain much nonlinearity and irregularity. Finally, the results of  $D_{stat}$  are different from those of  $RMSE$  because two criteria are different. The former is a level estimation criterion, while the latter is a directional evaluation measurement.

**Table 1.** The prediction performance comparison with different approaches

Model	Detail	$RMSE$	Rank	$D_{stat}(\%)$	Rank
Individual model	ARIMA	0.2242	9	57.86	9
	BPNN	0.1256	8	72.78	8
	SVM	0.1124	6	74.75	7
Hybrid model	ANN+GLAR [11]	0.1237	7	75.87	5
	ANN+ES [13]	0.1185	5	75.24	6
Metamodel	Simple averaging	0.1058	4	78.35	3
	Weighted averaging	0.0986	3	77.56	4
	Neural network	0.0813	2	87.44	1
	SVMR	0.0778	1	85.61	2

Focusing on the  $RMSE$  indicator, it is difficult to find that the SVMR based metamodel is the best, followed by neural network based metamodel, weighted averaging based metamodel and simple averaging based metamodel, the ARIMA model performs the worst. To summarize, the metamodels outperform the individual model.

However, the low  $RMSE$  does not necessarily mean that there is a high hit ratio of forecasting direction for foreign exchange movement direction prediction. Thus, the  $D_{stat}$  comparison is necessary. Focusing on  $D_{stat}$  of Table 1, we find the neural network based metamodel also performs much better than the other models according to the rank. Furthermore, from the business practitioners' point of view,  $D_{stat}$  is more important than  $RMSE$  because the former is an important decision criterion. From Table 1, the differences among different models are very significant. In this case, the  $D_{stat}$  for the single ARIMA model is 57.86%, for the two hybrid models, the  $D_{stat}$ s are 75.87% and 75.24%, respectively, and for the SVMR based metamodel, the  $D_{stat}$  is only 85.61%, while for the neural network based metamodel,  $D_{stat}$  reaches 87.44%.

## 4 Conclusions

In this study, a neural-network-based metalearning technique is proposed for exchange rates prediction. In terms of the empirical results, we find that across different forecasting models for the test case of EUR/USD, the proposed neural network based metamodel performs the best, implying that the neural network metalearning technique can be used as a viable solution for foreign exchange rate prediction.

## Acknowledgements

This work is partially supported by National Natural Science Foundation of China; Chinese Academy of Sciences; Key Research Institute of Humanities and Social Sciences in Hubei Province-Research Center of Modern Information Management and Strategic Research Grant of City University of Hong Kong (SRG No. 7001677).

## References

1. McCulloch, W.S., Pitts, W.: A Logical Calculus of the Ideas Imminent in Nervous Activity. *Bulletin and Mathematical Biophysics* 5 (1943) 115-133
2. Hertz, J., Krogh, A., Palmer, R.G.: *Introduction to the Theory of Neural Computation*, Addison-Wesley, Reading, MA, 1989
3. Tsaih, R., Hsu, Y., Lai, C.C.: Forecasting S&P 500 Stock Index Futures with a Hybrid AI System. *Decision Support Systems* 23 (1998) 161-174
4. Yu, L., Lai, K. K., Wang S.Y., Huang, W.: A Bias-Variance-Complexity Trade-off Framework for Complex System Modeling. *Lecture Notes in Computer Science* 3980 (2006) 518-527
5. Chan, P., Stolfo, S.: Meta-learning for Multistrategy and Parallel Learning. *Proceedings of the Second International Workshop on Multistrategy Learning* (1993) 150-165
6. White, H.: Connectionist Nonparametric Regression: Multilayer Feedforward Networks Can Learn Arbitrary Mappings. *Neural Networks* 3 (1990) 535-549
7. Hornik, K., Stinchcombe, M., White, H.: Multilayer Feedforward Networks are Universal Approximators. *Neural Networks* 2 (1989) 359-366
8. Broomhead, D.S., Lowe, D.: Multivariable Functional Interpolation and Adaptive Networks. *Complex Systems* 2 (1988) 321-355
9. Mackay, D.J.C.: The Evidence Framework Applied to Classification Problems. *Natural Computation* 4 (1992) 720-736
10. Jolliffe, I.T.: *Principal Component Analysis*. Springer-Verlag, 1986
11. Yu, L., Wang, S.Y., Lai, K.K.: A Novel Nonlinear Ensemble Forecasting Model Incorporating GLAR and ANN for Foreign Exchange Rates. *Computers & Operations Research* 32 (2005) 2523-2541
12. Lai, K.K., Yu, L., Wang, S.Y., Huang, W.: A Novel Nonlinear Neural Network Ensemble Model for Financial Time Series Forecasting. *Lecture Notes in Computer Science* 3991 (2006) 790-793.
13. Lai, K.K., Yu, L., Wang, S.Y., Huang, W.: Hybridizing Exponential Smoothing and Neural Network for Financial Time Series Prediction. *Lecture Notes in Computer Science* 3994 (2006) 493-500.
14. Hansen, L.K., Salamon, P.: Neural Network Ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12 (1990) 993-1001
15. Benediktsson, J.A. Sveinsson, J.R. Ersoy, O.K. Swain, P.H.: Parallel Consensual Neural Networks. *IEEE Transactions on Neural Networks* 8 (1997) 54-64
16. Yu, L., Wang, S.Y., Lai, K.K.: A Novel Adaptive Learning Algorithm for Stock Market Prediction. *Lecture Notes in Computer Science* 3827 (2006) 443-452
17. Yu, L., Wang, S.Y., Lai, K.K.: An Adaptive BP Algorithm with Optimal Learning Rates and Directional Error Correction for Foreign Exchange Market Trend Prediction. *Lecture Notes in Computer Science* 3973 (2006) 498-503