A Time Series Forecasting Model based on Deep Learning with Stacked Autoencoders and SVR Integrated Algorithm for FX Prediction

Hua Shen
School of Information
Renmin University of China
Beijing 100872, China
e-mail: neyanbhbin_angel@yahoo.com

Xun Liang
School of Information
Renmin University of China,
Beijing 100872, China
e-mail: xliang@ruc.edu.cn

***Abstract*—This paper proposes an Integrated Algorithm based on Deep Learning with Stacked Autoencoders (SAE) and Support Vector Regression(SVR), it is also for the first time that applies a typical Deep Learning algorithm SAE to Foreign Exchange(FX) time series forecasting. We adopt 28currency pairs pertaining to G7 currencies and RenMinBi, and collect the real daily FX data for simulation. To implement the empirical study, we develop the program independently for SAE-SVR Integrated Algorithm, and benchmark the results with ANN and SVR model, which are considered as the best performance in Artificial Intelligence. Ultimately, the simulation results indicates that the SAE-SVR integrated algorithm performs much better over other benchmarks.**

Keywords-Deep Learning; Stacked Autoencoders; Time Series Forecasting; Foreign Exchange

I. INTRODUCTION

Since the collapse of Bretton Woods Agreement in 1973, the foreign exchange market has become the most influential market in financial world, with an average daily turnover for global foreign exchange market of $5345 billion1. Increasingly, Foreign Exchange rate plays a significant roll not only in people who engaged in the financial fields, but also in entrepreneurs and international-level macroeconomic relations and strategy measures. Therefore, it arises an ascending number of governments, economists and financial institution's interest in developing high accuracy techniques for forecasting Foreign Exchange(FX) time series[1].

Taking it by and large, the main approaches on this problem have proceeded on three fronts in the literatures. First of all, a majority of research efforts adopt the time-dependent conditional heteroskedasticity into standard models and use volatility as a key parameter. These models belong to the ARCH and GARCH approaches intitiated by Engle and Bollerslev. In addition, there are the fundamental models attempting to project the exchange rates based on rational expectations hypotheses involving major macroeconomical figures. These models are established on the foundations of supply and demand of domestic currency compared with a foreign currency. Ultimately, there are an increasing number of studies recently begin to focus on artificial intelligent approaches to forecast exchange rate. This category mostly uses time-series statistics to predict currency movements and is proven to be outperformed than the traditional approaches[2]. Optimized Algorithms of Artificial Neural Networks (ANN) are best performed and most common in Artificial Intelligence(AI) field for the moment, But ANN can still not go beyond one or two hidden layers for the problematic non-convex optimization, therefore the difficult problem of learning in deep networks for higher precision is left dormant.

However, in 2006, Geoffrey Hinton et al. rekindled interest in ANN by showing substantially better performance by a "deep" neural network that proved successful at learning their parameters[3-4]. Deep learning algorithms trained in this fashion have been shown empirically to avoid getting stuck in the kind of poor solutions one typically reaches with only random initialization[5-6]. While until now there are few people make empirical study of time series modeling with the typical deep neural network naming Stacked Autoencoder(SAE)[7-8], which consists of multiple layers of sparse autoencoders and the outputs of each layer is wired to the inputs of the successive layer[9-10].

Under this circumstance, this paper takes a novel perspective on the problem of optimizing the forecasting precision by proposing a Deep Learning with Stacked Atutoencoders(SAE) and Support Vector Regression(SVR) Integrated Algorithm to overcome the drawbacks contained in statistic models and artificial neural networks. The innovative proposed methodology could be adapt to different currencies exchange rate and even various time series.

II. THE FX TIME SERIES FORECASTING MODEL

A. The Deep Learning With SAE-SVR Integrated Model Structure

In general, the time series forecasting model we proposed based on Deep Learning with Stacked Autoencoders(SAE) and Support Vector Regression(SVR) Integrated Algorithm combines the merits of SAE to deeply learn dataset features remarkably with the advantages of SVR's superior predicting capacity for a more precise forecasting. Considering a network

1 Source: The latest statistics of BIS (Bank for International Settlements) Triennial Central Bank Survey in the size and structure of global foreign exchange and OTC derivatives markets (updated 13 September 2015).

structure of deep learning with SAE-SVR Integrated Algorithm as shown in Figure1, it consists of one Input Layer, one Output Layer and K Hidden Layers. With this greedy layer-wise training method, each hidden layer can gradually learn part-whole features of the dataset.

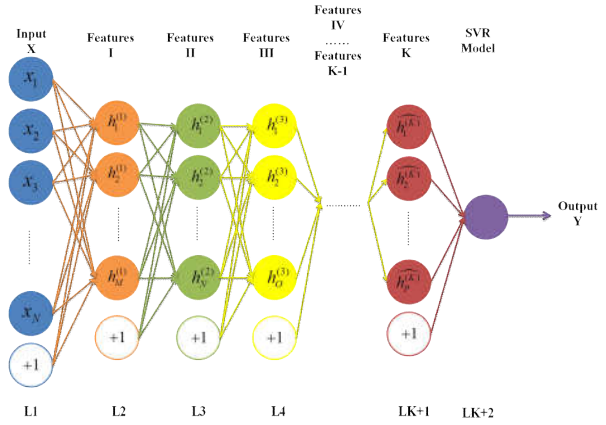

Figure 1. Deep Learning with SAE-SVR Integrated Algorithm Structure

To be specific, during the implementation process, we train the SAE-SVR Integrated Algorithm layer by layer, and each layer represents a Sparse Autoencoder, which is illustrated in Figure2 below.

For the first Sparse Autoencoder, we initiate the net-config as $(W,b)=(W^{(1)},b^{(1)},W^{(2)},b^{(2)})$, for $W_{ij}^{(1)}$ connects the jth unit in layer L with the ith unit in layer L+1, and $b_i^{(l)}$ represents the bias of ith unit in layer L+1. In addition, we define $a_i^{(l)}$ as the activation of ith unit in layer L, as well as the output of this neural unit, and $z_i^{(l)}$ as the input of this neural unit. Therefore, the functional relationship of the first Sparse Autoencoder for a single input loop is as below:

$$a_1^{(2)}=f(z_1^{(2)})=f(\sum_{i=1}^{N}W_{1i}^{(1)}x_i+b_1^{(1)})$$

$$\cdots$$

$$a_M^{(2)}=f(z_M^{(2)})=f(\sum_{i=1}^{N}W_{Mi}^{(1)}x_i+b_M^{(1)})$$

;

$$a_1^{(3)}=\hat{x}_1=f(z_1^{(3)})=f(\sum_{i=1}^{M}W_{1i}^{(2)}a_i^{(2)}+b_1^{(2)})$$

$$\cdots$$

$$a_N^{(3)}=\hat{x}_N=f(z_N^{(3)})=f(\sum_{i=1}^{M}W_{Ni}^{(2)}a_i^{(2)}+b_N^{(2)})$$

;

Where $f(\cdot):\Re\to\Re$, here we set it a Sigmoid Function as :

$$f(z)=\frac{1}{1+\exp(-z)}$$

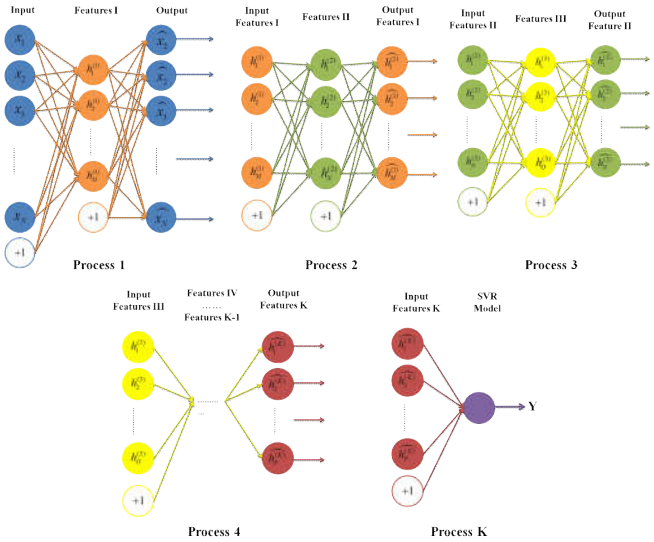

Figure 2. The Deep Learning with SAE-SVR Integrated Algorithm FeedForward Substep Structure

B. *The Back-Propagation Fine-Tuning of Algorithm Model*

As is known, the output of the first Sparse Autoencoder is $A^{(3)}=\{a_1^{(3)},a_2^{(3)},...,a_N^{(3)}\}\in R$, while the real value of L3 equals to input $X=\{x_1,x_2...,x_N\}\in R$ according to attribute of Stacked Autoencoders. In the following, we conduct a Back-Propagation algorithm to fine-tune the net-config during multi input loops for the first Sparse Autoencoder.

Firstly, we define a training set as $\{(x^{(1)},\hat{x}^{(1)}),...,(x^{(m)},\hat{x}^{(m)})\}$, and a Square Error Cost Function $J(W,b)$ as

$$J(W,b)=\left[\frac{1}{m}\sum_{i=1}^{m}J(W,b;x^{(i)},\hat{x}^{(i)})\right]+\frac{\lambda}{2}\sum_{l=1}^{2}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(W_{ji}^{(l)})^2+\beta\sum_{j=1}^{s_2}KL(\rho\,\|\,\hat{\rho}_j)$$

$$=\left[\frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2}\|\hat{x}^{(i)}-x^{(i)}\|^2\right)\right]+\frac{\lambda}{2}\sum_{l=1}^{2}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ji}^{(l)}\right)^2+\beta\sum_{j=1}^{s_2}KL(\rho\,\|\,\hat{\rho}_j)$$

Where $\frac{\lambda}{2}\sum_{l=1}^{2}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(W_{ji}^{(l)})^2$ is the Weight Decay Term to avoid overfitting, s_L means the number of units of layer L, and ρ means SparsityParam, $\sum_{j=1}^{s_2}KL(\rho\,\|\,\hat{\rho}_j)$ is Penalty Term based on Kullback-Leibler Divergence:

$$KL(\rho\,\|\,\hat{\rho}_j)=\rho\log\frac{\rho}{\hat{\rho}_j}+(1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_j}$$

The concrete steps of Back-Propagation algorithm for the first sparse autoencoder are as follows:

Step1: Perform a FeedForward process and obtain the activations for each unit and corresponding net-configs.

Step2: For each unit i of ouput layer L3, set error value δ as:

Identify applicable sponsor/s here. *(sponsors)*

$$\delta_i^{(3)} = \frac{\partial}{\partial z_i^{(3)}} \frac{1}{2} \left\| x_i - \hat{x}_i \right\|^2 = -(x_i - a_i^{(3)}) \bullet f'(z_i^{(3)})$$

Step3: For each unit i of hidden layer L2, set:

$$\delta_i^{(2)} = \left(\sum_{j=1}^{s3} W_{ji}^{(2)} \delta_j^{(3)} \right) f'(z_i^{(2)})$$

Step4: Calculate the desired partial derivatives $\{W_1 grad, W_2 grad, b_1 grad, b_2 grad\}$ by:

$$\frac{\partial}{\partial W_{ij}^{(l)}} J(W,b;x,\hat{x}) = a_j^{(l)} \delta_i^{(l+1)} \quad ; \quad \frac{\partial}{\partial b_i^{(l)}} J(W,b;x,\hat{x}) = \delta_i^{(l+1)}$$

Step5: Update the net-config $(W,b) = (W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)})$ by:

$$W_{ij}^{(l)} = W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W,b) \quad ; \quad b_i^{(l)} = b_i^{(l)} - \alpha \frac{\partial}{\partial b_i^{(l)}} J(W,b)$$

Above all, the first Sparse Autoencoder learns itself with multi input loops during above fine-tune procedure to obtain more precise forecasting results. However, the processes of training the remaining Sparse Autoencoders are similar to that of the first one, and the activations outputs of each layer is wired to the inputs of the successive layer, until the Kth activations obtained from process K-1 will be directly conducted as input of a SVR model.

Last but not least, after all the K processes are accomplished, there will be a further Back-Propagation fine-tune for the whole SAE-SVR Integrated Algorithm descend the error of prediction outcomes. What should be noted different is that we define the error value of the last layer as:

$$\delta_i^{(L_n)} = \frac{\partial}{\partial z_i^{(L_n)}} \frac{1}{2} \left\| y_i - \hat{x}_i \right\|^2 = -(y_i - a_i^{(L_n)}) \cdot f'(z_i^{(L_n)})$$

In the last layer, we compare the output directly with the real forecasting value $Y = (y_1, ..., y_M) \in R$ of the dataset.

III. DATA COLLECTION AND TESTING

In this paper, we base the Foreign Exchange (FX) rate datasets on the G7 currencies (USD, GBP, EUR, JPY, AUD, CAD, CHF), and collect them on MetaTrader4 platform of FXCM. In view of the significant influences of RenMinBi, we additionally adopt CNY from SAFE (State Administration of Foreign Exchange) official website. Therefore, we use 28 currency pair datasets in total illustrated in Table 1.

TABLE1. THE 28 CURRENCY PAIR DATASETS

	USD	**EUR**	**GBP**	**CAD**	**AUD**	**JPY**	**CHF**	**CNY**
USD	—	—	—	5USDCAD	—	11USDJPY	16USDCHF	22USDCNY
EUR	1EURUSD	—	4EURGBP	6EURCAD	9EURAUD	12EURJPY	17EURCHF	23EURCNY
GBP	2GBPUSD	—	—	7GBPCAD	10GBPAUD	13GBPJPY	18GBPCHF	24GBPCNY
CAD	—	—	—	—	—	14CADJPY	19CADCHF	25CADCNY
AUD	3AUDUSD	—	—	8AUDCAD	—	15AUDJPY	20AUDCHF	26AUDCNY
JPY	—	—	—	—	—	—	—	27JPYCNY
CHF	—	—	—	—	—	21CHFJPY	—	28CHFCNY

As for the data frequency and time span, we extract the daily FX data in MetaTrader4 and SAFE from 21st Mar 2009 to 1st Feb 2016. Besides, we classify all the dataset into Training set and Testing set respectively for machine learning process, details are shown in Table 2.

TABLE2. THE DETAILS OF 28 CURRENCY PAIR DATASETS

Dataset No.	**Currency pair**	**AllData Bulk**	**Start Date**	**Expiry Date**	**TrainingSet Bulk**	**TestingSet Bulk**
1--21	See Table1	2048*21	2009-03-20	2016-02-01	1548*21	500*21
22	USDCNY	1782	2008-10-06	2016-02-01	1282	500
23	EURCNY	1782	2008-10-06	2016-02-01	1282	500
24	GBPCNY	1782	2008-10-06	2016-02-01	1282	500
25	CADCNY	1014	2011-11-28	2016-02-01	514	500
26	AUDCNY	1014	2011-11-28	2016-02-01	514	500
27	JPYCNY	1782	2008-10-06	2016-02-01	1282	500
28	CHFCNY	59	2015-11-10	2016-02-01	39	20
Sum	----	52223	----	----	38703	13520

Before we conduct the simulation test, firstly, the data should be normalized between [0, 1] scale, for each currency pair time series data $S = (s_1, s_2, ..., s_T)$, the conversion formula is:

$$z_i = \frac{s_{\max} - s_i}{s_{\max} - s_{\min}}$$

Then we get the normalized currency pair time series $Z = (z_1, z_2, ..., z_T)$, secondly, the normalized $Z = (z_1, z_2, ..., z_T)$ will be transformed into a L-lag-window multi-dimension time series vector $X = [X_1, ..., X_M] = (x_{ij})_{i,j=1}^{L,M}$, for $X_i = (z_i, ..., z_{i+L-1})' \in R^L$, $M = T - L + 1$, and the lag-window L is an integer meeting $2 \le L \le T/2$, So the new input vector is as below:

$$X = [X_1, ..., X_M] = (x_{ij})_{i,j=1}^{L,M} = \begin{bmatrix} z_1 & z_2 & z_3 & \cdots & z_M \\ z_2 & y_3 & z_4 & \cdots & z_{M+1} \\ \cdots & \cdots & \cdots & \ddots & \cdots \\ z_L & z_{L+1} & z_{L+2} & \cdots & z_T \end{bmatrix}$$

While the output vector is a one-dimensional time series vector:

$$Y = (y_1, ..., y_M) \in R$$

Where $y_n = z_{n+L}$, and y_n indicates the forecasting value of $X_n = (z_n, ..., z_{n+L-1})' \in R^L$. Finally, we get the 28 time series input and output vectors after preprocessing.

The methodology we proposed for prediction with Stacked Autoencoders and SVR model is based on Non-stationary and Nonlinear time series. Therefore, this section aims to test Nonlinear and Non-stationary attributes of the dataset by using Normality and Unit Root Tests respectively. On the one hand, we adopt the well-known test Augmented Dickey-Fuller for unit root test to validate the Non-stationary attribute of the 28 currency pair time series. On the other hand, we take use of Jarque-Bera Test to implement the Normality Test of the 28 currency pair time series with help of Eviews Tool, the results could not illustrated for lack of space. However, the Unit Root Test and Normality Test results show that the 28 datasets mostly conform to the Non-stationary and Nonlinear attributes and could be used in our proposed integrated algorithm.

IV. SIMULATION AND RESULTS

In this paper, we estimate the error with MAE (Mean Absolute Error), MSE (Mean Square Error), RMSE (Root Mean Square Error) as criteria for assessing the validity of our integrated algorithm.

$$MAE = \sum_{i=1}^{n} \frac{|\hat{y}_i - y_i|}{n}; \quad MSE = \sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}; \quad RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

Where $\hat{y}_i$ is the predicted value of corresponding y_i.

The simulation environment is based on Matlab R2015a platform in 32-bit Windows7, as to the innovative SAE-SVR integrated algorithm, we refer to the UFLDL Tutorial of Deep Learning curriculum offered by Stanford University, and develop the codes independently. In addition, we benchmark our SAE-SVR integrated algorithm with another two artificial intelligence models: ANN (Artificial Neural Network), and SVR (Support Vector Regression), the ANN model is conducted with Neural Network Time Series Toolbox in Matlab R2015a, while the SVR model is implemented with LIBSVM 3.12 Toolbox.

To be more concrete, the main program implement steps for the SAE-SVR integrated algorithm come down to:

Step1: Provide the relevant parameters, involving inputSize, hiddenSizeL1,…,hidenSizeLn, sparsityParam, lambda, beta, alpha, etc.

Step2: Load normalized vectors data, including Training Set and Testing Set.

Step3: Train the first Sparse Autoencoder with training set as input vector, and get the trained net-config 'sae1Theta', then optimize 'sae1Theta' with SparseAECost function to obtain 'sae1OptTheta', and further conduct feedForwardAutoencoder function to achieve the first feature vector 'sae1Features'.

Step4: Train the second Sparse Autoencoder, set the 'sae1Features' as input vector and obtain 'sae2Theta', 'sae2OptTheta', and 'sae2Features'. 'Sae2Features' is the input vector of the next step.

Step5: By that analogy, accomplish training N layer Stacked Autoencoders, until the Nth output vector 'saeNFeatures'.

Step6: Set 'saeNFeatures' as input vector to train the SVR model, and get the output 'svmoutput'.

Step7: Fine-tune the whole SAE-SVR Algorithm: make a comparison between the model output 'svmoutput' with real forecasting value 'y', and fine-tune the model config with stackedAECost function, the updated parameters are saved in stack{}.

Step8: Predict the testing set with optimized SAE-SVR Algorithm after fine-tuning, achieve the forecasting values with stackedAEPredict function, and evaluate MSE, RMSE, MAE results.

However, the SAE-SVR Integrated Algorithm flow chart is illustrated below in Figure3:

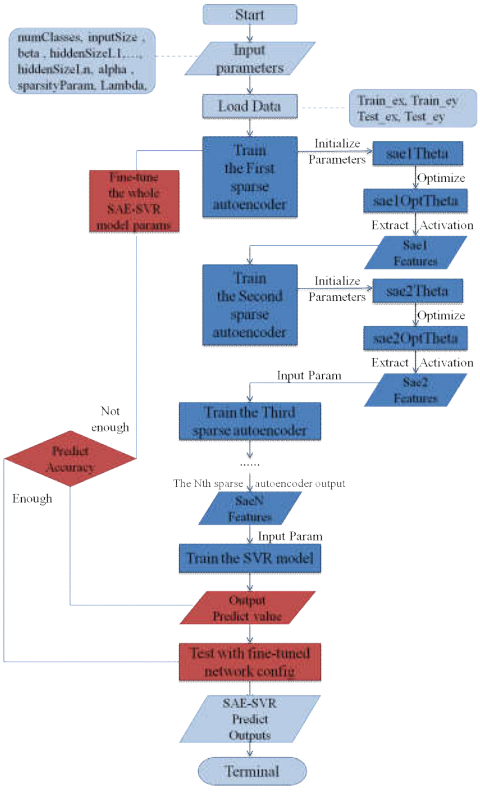

Figure 3. The SAE-SVR Integrated Algorithm Flow Chart

Eventually, the simulation outcomes of the SAE-SVR Integrated Algorithm are summarized in Table3 below:

TABLE3. THE SIMULATION OUTCOMES OF SAE-SVR INTEGRATED ALGORITHM

DataSet	MSE	MAE	RMSE	DataSet	MSE	MAE	RMSE
1EURUSD	2.88E-04	0.02234	0.016981343	8AUDCAD	1.57E-04	0.01089	0.012513113
2GBPUSD	6.09E-04	0.01169	0.024677135	9EURAUD	4.12E-05	0.00513	0.006416206
3AUDUSD	1.95E-04	0.01012	0.01397165	10GBPAUD	2.77E-04	0.01465	0.016632949
4EURGBP	2.51E-04	0.01149	0.015852855	11USDJPY	2.12E-04	0.01577	0.014573435
5USDCAD	1.82E-04	0.00922	0.013475793	12EURJPY	7.06E-05	0.00655	0.008399655
6EURCAD	1.44E-04	0.00984	0.011981319	13GBPJPY	9.14E-05	0.0082	0.009560063
7GBPCAD	2.40E-04	0.01808	0.015485768	14CADJPY	1.70E-04	0.00939	0.013043121
DataSet	**MSE**	**MAE**	**RMSE**	**DataSet**	**MSE**	**MAE**	**RMSE**
15AUDJPY	5.06E-05	0.0089	0.007110471	22USDCNY	3.11E-04	0.0099	0.017630116
16USDCHF	3.42E-05	0.00788	0.005845439	23EURCNY	2.94E-04	0.01578	0.017139049
17EURCHF	7.01E-05	0.0049	0.008370759	24GBPCNY	1.68E-04	0.01114	0.012952683
18GBPCHF	9.99E-05	0.00541	0.009996749	25CADCNY	3.20E-04	0.01675	0.01789905
19CADCHF	4.82E-05	0.00978	0.006940519	26AUDCNY	3.63E-04	0.01772	0.019063997
20AUDCHF	9.80E-05	0.01181	0.009900909	27JPYCNY	1.32E-04	0.00086	0.011479983
21CHFJPY	4.14E-05	0.00835	0.006437608	28CHFCNY	1.92E-05	0.00438	0.004377614

Comparing with the ANN and SVR model, the 28datesets simulation results are aggregated in Table4 below, from which we can tell, ANN model performs much better than SVR model in predicting foreign exchange rate, so we contrast SAE-SVR model directly to ANN model with calculation formula:

$$\Pr omoted = \frac{|(SAESVR)MSE - (ANN)MSE|}{(ANN)MSE}$$

$$Sumup = \sum_{i=1}^{N=28} \Pr omoted\_value(i)$$

The promoted column reveals that in 28 entire datasets, although the SAE-SVR model performs not better than ANN in 9 currency pairs involving 1EURUSD, 5USDCAD, 7GBPCAD, 10GBPAUD, 11USDJPY, 22USDCNY, 23EURCNY, 25CADCNY, 26AUDCNY, the other 21 currency pairs datasets all indicate a better performance than ANN and SVR model. Further we calculate the Sum Up of the SAE-SVR's proposed performance, it shows that the SAE-SVR is more than 6 times better than ANN model in MSE criteria, and more than 2 times better than ANN in MAE criteria, which come to the conclusion that the SAE-SVR Integrated Algorithm we proposed is attained with distinction.

TABLE4. THE AGGREGATED 28 DATASETS SIMULATION RESULTS OF ANN, SVR, SAE-SVR MDOELS

DataSet Name	ANN MSE	SVR MSE	SAE-SVR MSE	Promoted	ANN MAE	SVR MAE	SAE-SVR MAE	Promoted
1EURUSD	2.7307e-04	0.0026098	2.88E-04	-5.60%	0.0122	0.0399	0.02234	-83.11%
2GBPUSD	6.8046e-04	0.0011528	6.09E-04	10.51%	0.0193	0.0272	0.01169	39.43%
3AUDUSD	2.3025e-04	0.0003145	1.95E-04	15.22%	0.0113	0.0143	0.01012	10.44%
4EURGBP	2.6657e-04	0.0015129	2.51E-04	5.72%	0.0122	0.0307	0.01149	5.82%
5USDCAD	1.3537e-04	0.0022363	1.82E-04	-34.15%	0.0085	0.0332	0.00922	-8.47%
6EURCAD	2.9867e-04	0.0005757	1.44E-04	51.94%	0.0126	0.0189	0.00984	21.90%
7GBPCAD	2.2838e-04	0.0007986	2.40E-04	-5.00%	0.0167	0.0215	0.01808	-8.26%
8AUDCAD	4.8400e-04	0.00068021	1.57E-04	67.65%	0.0167	0.0198	0.01089	34.79%
9EURAUD	1.3568e-04	0.00024962	4.12E-05	69.66%	0.0084	0.0114	0.00513	38.93%
10GBPAUD	2.1326e-04	0.00062255	2.77E-04	-29.73%	0.0109	0.0191	0.01465	-34.40%
11USDJPY	1.4181e-04	0.00465398	2.12E-04	-49.77%	0.0088	0.0567	0.01577	-79.20%
12EURJPY	2.7442e-04	0.00034466	7.06E-05	74.29%	0.0120	0.0142	0.00655	45.42%
13GBPJPY	1.9389e-04	0.00141236	9.14E-05	52.86%	0.1299	0.0299	0.0082	93.69%
14CADJPY	4.2757e-04	0.00055366	1.70E-04	60.21%	0.0149	0.0177	0.00939	36.98%
15AUDJPY	3.6397e-04	0.0002919	5.06E-05	86.11%	0.0136	0.0129	0.0089	34.56%
16USDCHF	2.8011e-04	0.0009669	3.42E-05	87.80%	0.0101	0.0154	0.00788	21.98%
17EURCHF	1.9228e-04	0.0022557	7.01E-05	63.56%	0.0061	0.0304	0.0049	19.67%
18GBPCHF	3.0714e-04	0.0010526	9.99E-05	67.46%	0.0103	0.0155	0.00541	47.48%
19CADCHF	2.9759e-04	0.00111476	4.82E-05	83.81%	0.0107	0.0208	0.00978	8.60%

20AUDCHF	4.1407e-04	0.0015979	9.80E-05	76.33%	0.0129	0.0268	0.01181	8.45%
21CHFJPY	2.2217e-04	0.0014423	4.14E-05	81.35%	0.0086	0.0224	0.00835	2.91%
22USDCNY	8.7996e-05	0.00103769	3.11E-04	-253.22%	0.0061	0.0135	0.0099	-62.30%
23EURCNY	2.2732e-04	0.0017301	2.94E-04	-29.22%	0.0110	0.0318	0.01578	-43.45%
24GBPCNY	4.5092e-04	0.00052641	1.68E-04	62.79%	0.0149	0.0163	0.01114	25.23%
25CADCNY	1.7722e-04	0.0045405	3.20E-04	-80.78%	0.0097	0.0536	0.01675	-72.68%
26AUDCNY	2.3929e-04	0.00255696	3.63E-04	-51.88%	0.0118	0.0403	0.01772	-50.17%
27JPYCNY	1.9699e-04	0.00179411	1.32E-04	33.10%	0.0099	0.0349	0.00086	91.31%
28CHFCNY	0.0187	0.0316333	1.92E-05	99.90%	0.0886	0.1728	0.00438	95.06%
Sum Up	---	---	---	610.92%	---	---	---	240.61%

V. CONCLUSIONS

With the rapid variation in FX market, it brings an ascending number of attentions to make more precise forecasting for Foreign Exchange Rate. In this paper, we propose an innovative Integrated Algorithm based on Deep Learning with Stacked Autoencoders and SVR, we take a novel perspective to extract the high-dimensional abstract features from K layers Sparse Autoencoders and send the output activations into the SVR model for prediction. For the sake of verifying the integrated algorithm, we take advantage of FX real data pertain to G7 and RenMinBi currency pairs in MetaTrader4 platform as well as SAFE website, and normalize and test all the datasets before simulation. To implement the simulation, we develop the program independently referring to UFLDL Tutorial by Stanford University, and benchmark our SAE-SVR integrated algorithm with the best performed ANN and SVR model. Ultimately, the aggregated comparison indicates that the SAE-SVR integrated algorithm outperformed than ANN and SVR, which verifies the outperformance of this innovative algorithm.

Moreover, the proposed algorithm has the potential ability to deal with massive a more complicated datasets, and more advanced optimization algorithm for parameter selection will be developed to enhance the forecasting accuracy.

ACKNOWLEDGMENT

This work is supported by the Fundamental Research Funds for the NSF of China under grant number 71271211, the Beijing NSF under grant number 4132067, and the Central Universities and the Research Funds of RenMin University of China. (10XNI029).

REFERENCES

[1] Liao, G. C. and Tsao, T. P., Application of a fuzzy neural network combined with a chaos genetic algorithm and simulated annealing to short term load forecasting, IEEE Transactions on Evolutionary Computation, 10, 330340 (2006).

[2] Van Gestel, T., Suykens, K. J., Baestaens, D., Lambrechts, A., Lanckriet, G., Vandaele, B., De Moor, B. and Vandewalle, J., Financial time series prediction using least squares support vector machines within the evidence framework, IEEE Transactions on Neural Networks, 12, 809-821 (2001).

[3] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In Bernhard Scholkopf, John Platt, and Thomas Hoffman, editors, ¨ Advances in Neural Information Processing Systems 19 (NIPS'06), pages 153–160. MIT Press, 2007.

[4] G.E. Hinton, S. Osindero, and Y.W. Teh. A fast learning algorithm for deep belief nets. Neural Computation, 18:1527–1554, 2006.

[5] Qiu X, Zhang L, Ren Y, et al. Ensemble deep learning for regression and time series forecasting[C]// Computational Intelligence in Ensemble Learning (CIEL), 2014 IEEE Symposium on. IEEE, 2014:1-6.

[6] Fakhr M W. Online Nonstationary Time Series Prediction using Sparse Coding with Dictionary Update[C]// Information and Communication Technology Research (ICTRC), 2015 International Conference on. IEEE, 2015.

[7] Zhang R, Shen F, Zhao J. A model with Fuzzy Granulation and Deep Belief Networks for exchange rate forecasting[C]// International Joint Conference on Neural Networks. IEEE, 2014:366-373.

[8] Shen F, Chao J, Zhao J. Forecasting exchange rate using deep belief networks and conjugate gradient method[J]. Neurocomputing, 2015, 167(C):243-253.

[9] H. Lee, C. Ekanadham, and A. Ng. Sparse deep belief net model for visual area V2. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, Advances in Neural Information Processing Systems 20 (NIPS'07), pages 873–880, Cambridge, MA, 2008. MIT Press.

[10] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," The Journal of Machine Learning Research, vol. 11, pp. 3371–3408, 2010.