Swarm and Evolutionary Computation ∎ (∎∎∎∎) ∎∎∎–∎∎∎

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

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

Regular Paper

A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction

Rajashree Dash, P.K. Dash *, Ranjeeta Bisoi

Siksha 'O' Anusandhan University, Khandagiri Square, Bhubaneswar, Odisha, India

ARTICLE INFO

Article history:
Received 23 March 2014
Received in revised form
22 July 2014
Accepted 22 July 2014

Keywords:
CEFLANN
RBF
ELM
Harmony search (HS)
Differential evolution (DE)
SADHS-OELM

ABSTRACT

This paper proposes a hybrid learning framework called Self Adaptive Differential Harmony Search Based Optimized Extreme Learning Machine (SADHS-OELM) for single hidden layer feed forward neural network (SLFN). The new learning paradigm seeks to take advantage of the generalization ability of extreme learning machines (ELM) along with the global learning capability of a self adaptive differential harmony search technique in order to optimize the fitting performance of SLFNs. SADHS is a variant of harmony search technique that uses the current to best mutation scheme of DE in the pitch adjustment operation for harmony improvisation process. SADHS has been used for optimal selection of the hidden layer parameters, the bias of neurons of the hidden-layer, and the regularization factor of robust least squares, whereas ELM has been applied to obtain the output weights analytically using a robust least squares solution. The proposed learning algorithm is applied on two SLFNs i.e. RBF and a low complexity Functional link Artificial Neural Networks (CEFLANN) for prediction of closing price and volatility of five different stock indices. The proposed learning scheme is also compared with other learning schemes like ELM, DE-OELM, DE, SADHS and two other variants of harmony search algorithm. Performance comparison of CEFLANN and RBF with different learning schemes clearly reveals that CEFLANN model trained with SADHS-OELM outperforms other learning methods and also the RBF model for both stock index and volatility prediction.

1. Introduction

Accurate forecasting of future behavior of the financial time series data with respect to its tremendous sudden variation and complex non-linear dimensionality is a big challenge for most of the investors and professional analysts. Indeed financial time series is highly volatile across time and is prone to fluctuation not only for economic factors but also for non economic factors like political conditions, investor's expectations based on actual and future economic etc. However, the benefits involved in accurate prediction have motivated researchers to develop newer and advanced tools and models. Forecasting financial time series is primarily focused on estimation of future stock price index and accurate forecasting of its volatility. The models used for financial time series forecasting fall into two categories. The first category involves models based on statistical theories, e.g. autoregressive moving average (ARMA), autoregressive integrated moving average (ARIMA), autoregressive conditional heteroscedasticity (ARCH), and generalized autoregressive conditional heteroskedasticity (GARCH) models. All these models assume the linearity of previous and current variables. Generally the financial time series data, being chaotic and noisy in nature, do not necessarily follow a fixed pattern or linearity and thus the statistical approaches do not perform very well in predicting stock market indices accurately. The second category includes models based on artificial intelligence, like ANN, Fuzzy set theory, Support Vector Machine, Rough Set theory etc. Due to the inherent capabilities to identify complex nonlinear relationship present in the time series data based on historical data and to approximate any nonlinear function to a high degree of accuracy, the application of ANN in modeling economic conditions is expanding rapidly. A survey of literature indicates that among different types of ANNs, i.e. Multi Layer Perception Network (MLP) [1–9], Radial Basis Function Neural Network (RBF) [10,11] and Functional Link Artificial Neural Network (FLANN) [12,13] are the most popular ANN tool used for predictions of financial time series.

The traditional back propagation algorithm with gradient descent method is the commonly used learning technique for ANNs. But it suffers from the issues of imprecise learning rate, local minimal and slow rate of convergence. To avoid the common drawbacks of back propagation algorithm and to increase the accuracy some scholars proposed several improved measures, including additional momentum method, self-adaptive learning rate adjustment

* Corresponding author. Tel.: +91 674 2727336.
E-mail address: pkdash.india@gmail.com (P.K. Dash).

http://dx.doi.org/10.1016/j.swevo.2014.07.003
2210-6502/

method, Recursive Least square method and various search algorithms like GA, PSO DE, HS algorithms [10,14–18], in the training step of the neural network to optimize the parameters of the network like the network weights and the number of hidden units in the hidden layer etc. To increase forecasting speed and accuracy, researchers have also tried to combine and optimize different

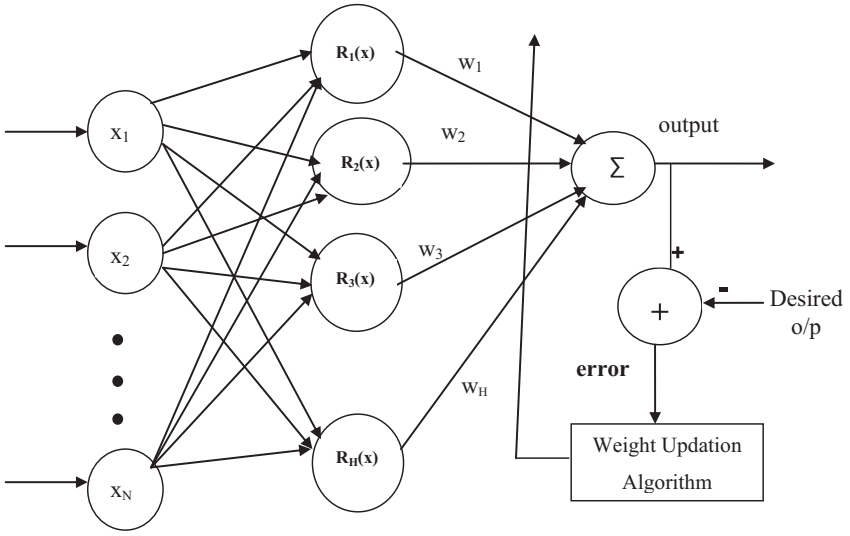

Fig. 1. Architecture of radial basis function network.

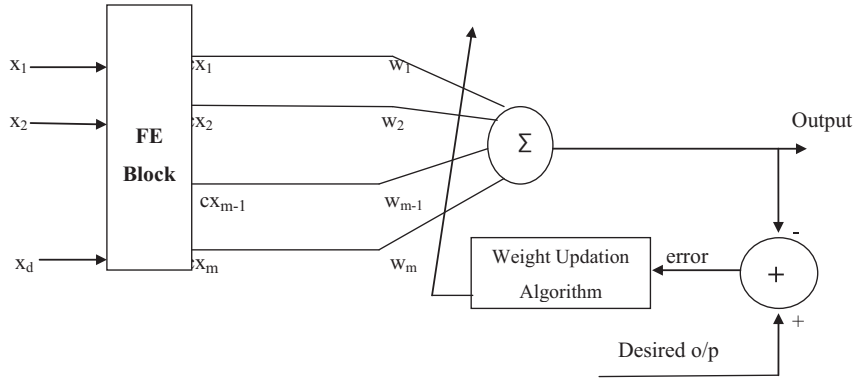

Fig. 2. Architecture of computational efficient FLANN (CEFLANN).

Table 1
Descriptive statistics of closing price.

Descriptive statistics	BSE Sensex	CNX Nifty	Nikkei 225	FTSE 100	S&P 500
Mean	1.8014e+004	5.3839e+003	9.7451e+003	5.6982e+003	1.2882e+003
Minimum	1.5175e+004	4.5442e+003	8.1600e+003	4.8058e+003	1.0226e+003
Maximum	2.1005e+004	6.3124e+003	1.3926e+004	6.5294e+003	1.5976e+003
Standard deviation	1.2352e+003	366.7439	1.0415e+003	339.0059	133.6663
Skewness	0.1411	0.2615	1.2970	0.0039	0.1491
Kurtosis	2.2326	2.4604	5.1992	2.7171	2.2666
Jarque–Bera test statistics	23.1765 ($h=1$)	17.7373 ($h=1$)	393.2247 ($h=1$)	2.7991 ($h=0$)	21.8353 ($h=1$)

Table 2
Descriptive statistics of daily return.

Descriptive Statistics	BSE Sensex	CNX Nifty	Nikkei 225	FTSE 100	S&P 500
Mean	0.0126	0.0185	0.0323	0.0213	0.0412
Minimum	−4.2129	−4.1689	−11.1534	−4.7792	−6.8958
Maximum	3.5181	3.5546	5.5223	5.0323	4.6317
Standard deviation	1.0699	1.1108	1.3243	1.0918	1.1324
Skewness	0.0293	0.0139	−0.8074	−0.1442	−0.4505
Kurtosis	3.5700	3.4826	9.5766	4.9304	6.9439
Jarque–Bera test statistics	11.3691 ($h=1$)	7.3303 ($h=1$)	1.5573e+003 ($h=1$)	133.0163 ($h=1$)	569.3927 ($h=1$)

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

algorithms, and build hybrid models. A hybridization of PSO and BP algorithm for training of ANN has been proposed in [19,20]. A hybrid evolutionary PSO and adaptive recursive least-squares (RLS) learning algorithms called RPSO has been considered together for efficiently acquiring available parameters value of the RBFNs prediction systems in [11]. In [36] a novel fuzzy time series forecasting model with entropy discretization and a Fast Fourier Transform algorithm has also been proposed for stock price forecasting. The model uses an FFT to deal with historical training data for forecasting stock prices. The approach provides a more accurate forecasting result compared to traditional ARCH, GARCH and other fuzzy time series models. Another high-order fuzzy time series model based on entropy-based partitioning and adaptive expectation model for time series forecasting has been proposed in [37]. The empirical results of the model show that entropy based discretization partitioning could be more suitable than the conventional time series model. Recently a new batch learning algorithm called Extreme Learning Machine (ELM) has been proposed in [21] for training of single hidden layer feed forward neural network. Mainly the ELM algorithm randomly initializes parameters of hidden nodes and analytically determines output weights of SLFNs. The main advantage of ELM is that the hidden layer of SLFNs need not be tuned. In fact, for the randomly chosen input weights and hidden layer biases, ELM will lead to a least squares solution of a system of linear equations for the unknown output weights having the smallest norm property. ELM has shown good generalization performances for many real applications with an extremely fast learning speed [22–26]. However, like other similar approaches based on feed forward neural networks,

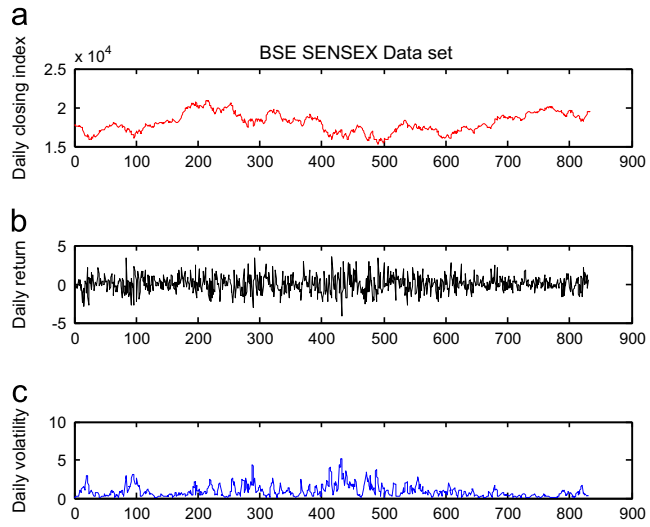

Fig. 3. BSE Sensex data set.

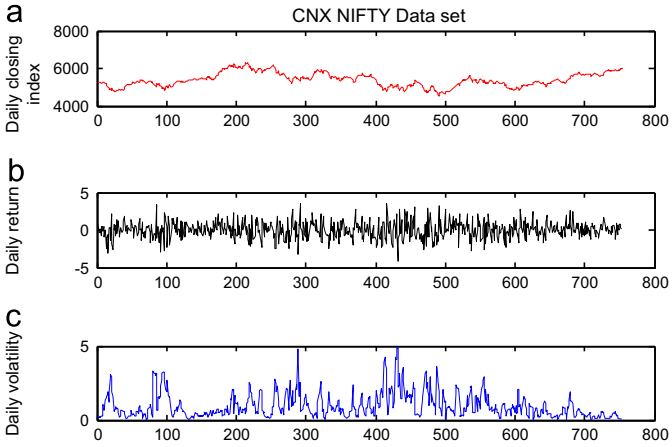

Fig. 4. CNX Nifty data set.

Table 3
Parameter space size of CEFLANN and RBF to be tuned using different learning algorithms.

Learning algorithm	Size of parameter space	
	CEFLANN	RBF
SADHS-OELM	23	50
DE-OELM	23	50
SADHS	22	49
SGHS	22	49
HS	22	49
DE	22	49
ELM	8	7
BP	22	49

Table 4
Controlling parameters of evolutionary based learning algorithms.

Learning algorithm	Population size	NI	Other controlling parameters
SADHS-OELM	20, 50	100	HMCR(min)=0.8
SADHS	20, 50	100	HMCR(max)=0.95
SGHS	20, 50	100	PADR(min)=0.2
			PADR(max)=0.5
HS	20, 50	100	BW(min)=0.15
			BW(max)=0.55
DE-OELM	20, 50	100	C_r=0.9
DE	20, 50	100	F(min)=0.3
			F(max)=0.8

Table 5
Performance comparison of RBF and CEFLANN with different learning algorithms for one day ahead stock price prediction of BSE Sensex data set (with population size=20).

Learning algorithms		RBF		CEFLANN	
		RMSE	MAPE	RMSE	MAPE
SADHS-OELM	Min	**0.0182**	**0.6418**	**0.0163**	**0.5635**
	Avg	**0.0184**	**0.6498**	**0.0165**	**0.5739**
	Std	**0.0001**	**0.0049**	**0.0001**	**0.0069**
DE-OELM	Min	0.0188	0.6532	0.0165	0.5748
	Avg	0.0190	0.6737	0.0169	0.5861
	Std	0.0005	0.0222	0.0002	0.0076
SADHS	Min	0.0268	0.9873	0.0185	0.6498
	Avg	0.0327	1.2096	0.0209	0.7418
	Std	0.0053	0.2014	0.0026	0.1035
SGHS	Min	0.0328	1.2251	0.0257	0.9058
	Avg	0.0389	1.4576	0.0273	0.9930
	Std	0.0074	0.2784	0.0051	0.2154
HS	Min	0.0368	1.3695	0.0194	0.6977
	Avg	0.0428	1.6162	0.0229	0.8214
	Std	0.0083	0.2982	0.0039	0.1524
DE	Min	0.0305	1.1256	0.0195	0.6887
	Avg	0.0350	1.2875	0.0211	0.7681
	Std	0.0076	0.2830	0.0029	0.1364
ELM	Min	0.0284	1.0424	0.0190	0.6711
	Avg	0.0287	1.0575	0.0194	0.06871
	Std	0.0008	0.0318	0.0012	0.0494
BP	Min	0.0272	1.0139	0.0191	0.6570
	Avg	0.0290	1.0785	0.0203	0.7159
	Std	0.0021	0.0759	0.0015	0.0679

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

Table 6
Performance comparison of RBF and CEFLANN with different learning algorithms for one day ahead stock price prediction of CNX NIFTY Data set (with population size=20).

Learning algorithms		RBF		CEFLANN	
		RMSE	MAPE	RMSE	MAPE
SADHS-OELM	**Min**	**0.0204**	**0.7368**	**0.0181**	**0.6407**
	Avg	**0.0205**	**0.7413**	**0.0182**	**0.6446**
	Std	**0.0001**	**0.0030**	**0.0000**	**0.0018**
DE-OELM	**Min**	0.0205	0.7412	0.0183	0.6462
	Avg	0.0211	0.7718	0.0185	0.6532
	Std	0.0006	0.0235	0.0002	0.0038
SADHS	**Min**	0.0265	1.0091	0.0186	0.6562
	Avg	0.0324	1.2461	0.0222	0.8023
	Std	0.0054	0.2070	0.0021	0.0785
SGHS	**Min**	0.0383	1.4396	0.0260	0.9495
	Avg	0.0414	1.5803	0.0289	1.0757
	Std	0.0081	0.2967	0.0040	0.1669
HS	**Min**	0.0408	1.5259	0.0209	0.7554
	Avg	0.0477	1.6804	0.0243	0.8868
	Std	0.0092	0.3045	0.0024	0.0904
DE	**Min**	0.0324	1.2378	0.0196	0.6967
	Avg	0.0387	1.4935	0.0206	0.7419
	Std	0.0054	0.2084	0.0019	0.0850
ELM	**Min**	0.0310	1.1914	0.0224	0.8411
	Avg	0.0316	1.2214	0.0236	0.8987
	Std	0.0004	0.0222	0.0021	0.0887
BP	**Min**	0.0304	1.1714	0.0203	0.7344
	Avg	0.0324	1.2350	0.0230	0.8395
	Std	0.0020	0.0736	0.0019	0.0743

Table 7
Performance comparison of RBF and CEFLANN with different learning algorithms for one day ahead volatility prediction of BSE Sensex data set (with population size=20).

Learning algorithms		RBF			CEFLANN		
		MSFE	RMSFE	MAFE	MSFE	RMSFE	MAFE
SADHS-OELM	**Min**	**0.0948**	**0.3079**	**0.2337**	**0.0887**	**0.2978**	**0.2238**
	Avg	**0.0982**	**0.3185**	**0.2417**	**0.0952**	**0.3084**	**0.2367**
	Std	**0.0037**	**0.0060**	**0.0082**	**0.0031**	**0.0050**	**0.0092**
DE-OELM	**Min**	0.1012	0.3181	0.2482	0.0974	0.3121	0.2414
	Avg	0.1153	0.3392	0.2764	0.0982	0.3133	0.2443
	Std	0.0131	0.0199	0.0216	0.0043	0.0070	0.0139
SADHS	**Min**	0.1270	0.3564	0.2769	0.1125	0.3354	0.2446
	Avg	0.1949	0.4406	0.3659	0.1566	0.3940	0.3311
	Std	0.0260	0.0292	0.0318	0.0294	0.0378	0.0443
SGHS	**Min**	0.1740	0.4171	0.3449	0.1525	0.3905	0.3044
	Avg	0.2174	0.4641	0.3918	0.2464	0.4855	0.3881
	Std	0.0449	0.0471	0.0487	0.1147	0.1090	0.1098
HS	**Min**	0.1827	0.4274	0.3611	0.1427	0.3777	0.3166
	Avg	0.2261	0.4627	0.4048	0.1614	0.3981	0.3379
	Std	0.0496	0.0482	0.0495	0.0459	0.0571	0.0554
DE	**Min**	0.1369	0.3700	0.2938	0.1405	0.3749	0.2834
	Avg	0.1720	0.4103	0.3352	0.1660	0.4051	0.3371
	Std	0.0288	0.0302	0.0329	0.0365	0.0449	0.0472
ELM	**Min**	0.1747	0.4180	0.3519	0.1568	0.3959	0.3425
	Avg	0.2417	0.4915	0.4161	0.2738	0.5103	0.4324
	Std	0.0107	0.0110	0.0124	0.1498	0.1220	0.1329
BP	**Min**	0.1752	0.4186	0.3343	0.1573	0.3966	0.3279
	Avg	0.1902	0.4359	0.3525	0.1722	0.4147	0.3415
	Std	0.0095	0.0109	0.0137	0.0132	0.0156	0.0142

some issues with the practical applications of the ELM still arise, most importantly, how to choose the optimal number of hidden nodes for a given problem which is usually done by trial and error method. The ELM tends to require more hidden neurons than conventional tuning based learning algorithms (based on error back

Table 8
Performance comparison of RBF and CEFLANN with different learning algorithms for one day ahead volatility prediction of CNX Nifty data set (with population size=20).

Learning algorithms		RBF			CEFLANN		
		MSFE	RMSFE	MAFE	MSFE	RMSFE	MAFE
SADHS-OELM	**Min**	**0.1599**	**0.3999**	**0.2860**	**0.1555**	**0.3944**	**0.2804**
	Avg	**0.1701**	**0.4102**	**0.2968**	**0.1636**	**0.4044**	**0.2890**
	Std	**0.0030**	**0.0037**	**0,0078**	**0.0018**	**0.0023**	**0.0027**
DE-OELM	**Min**	0.1727	0.4155	0.3156	0.1648	0.4059	0.2978
	Avg	0.1991	0.5124	0.3983	0.1718	0.4147	0.2922
	Std	0.0075	0.0091	0.0088	0.0038	0.0048	0.0035
SADHS	**Min**	0.2310	0.4807	0.3668	0.1755	0.4190	0.3124
	Avg	0.2830	0.5316	0.4221	0.2379	0.4866	0.3810
	Std	0.0204	0.0191	0.0221	0.0353	0.0360	0.352
SGHS	**Min**	0.2887	0.5373	0.4262	0.2616	0.5163	0.3892
	Avg	0.3474	0.5881	0.4865	0.2958	0.5407	0.4226
	Std	0.0513	0.0415	0.0545	0.0694	0.0614	0.0687
HS	**Min**	0.2766	0.5259	0.4196	0.2139	0.4625	0.3633
	Avg	0.3074	0.5551	0.4464	0.2602	0.5086	0.4023
	Std	0.0187	0.0172	0.0210	0.0424	0.0418	0.0417
DE	**Min**	0.2790	0.5282	0.3946	0.2113	0.4597	0.3592
	Avg	0.3204	0.5653	0.4494	0.2504	0.4993	0.3922
	Std	0.0365	0.0311	0.0409	0.0270	0.279	0.282
ELM	**Min**	0.2351	0.4849	0.3732	0.1991	0.4462	0.3619
	Avg	0.3490	0.5907	0.4763	0.2192	0.4679	0.3872
	Std	0.0021	0.0018	0.0011	0.0159	0.0169	0.0215
BP	**Min**	0.2509	0.5009	0.4012	0.2396	0.4895	0.3855
	Avg	0.2993	0.5466	0.4343	0.2623	0.5117	0.4053
	Std	0.0272	0.0252	0.0244	0.0237	0.0228	0.0203

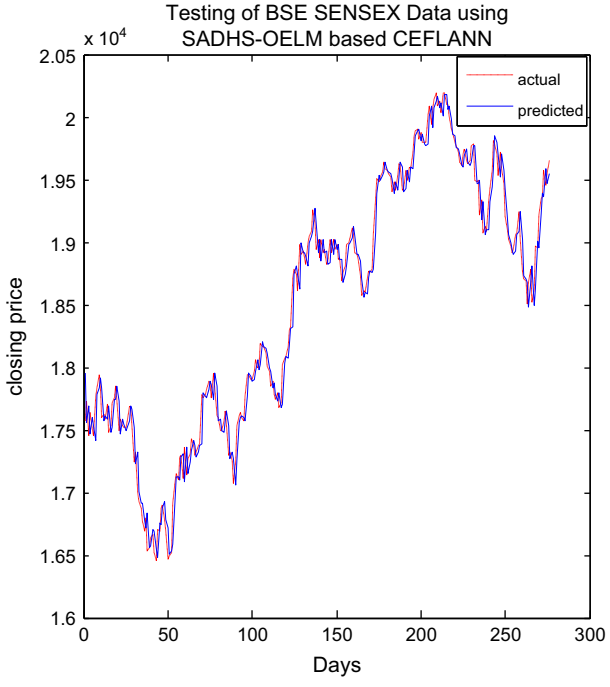

Fig. 5. One day ahead prediction of daily closing price of BSE data using SADHS-OELM based CEFLANN.

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

propagation or other learning methods where the output weights are not obtained by the least squares method) in some applications, which can negatively affect SLFN performance. Again the stochastic nature of the hidden layer output matrix may also lower the learning accuracy of ELM. Hence in order to further boost the generalization ability of extreme learning machines and to cope with the local minima problem a global optimization method called self adaptive differential harmony search (SADHS) hybridized with ELM has been proposed in this paper for learning of single-hidden layer feed forward neural networks.

The hybrid learning framework named Self Adaptive Differential Harmony Search Based Optimized Extreme Learning Machine (SADHS-OELM) uses the same concept of the ELM where the output weights are obtained using a robust least squares solution but, in order to optimize the fitting performance, the selection of

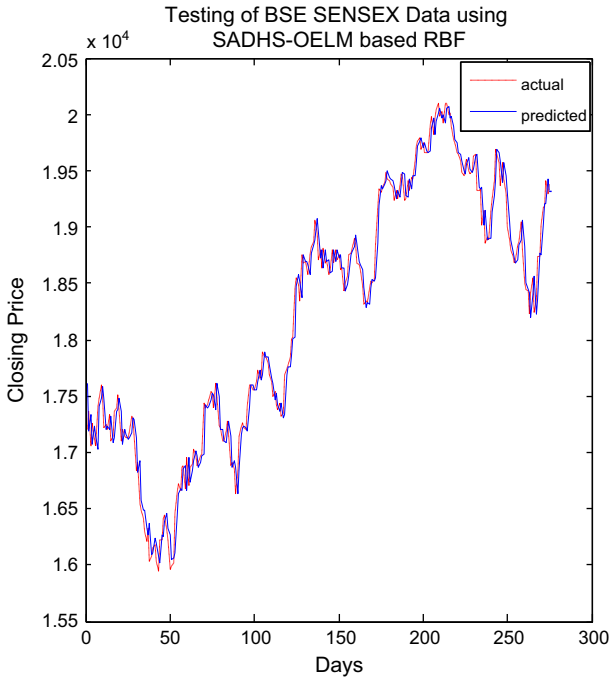

Fig. 6. One day ahead prediction of daily closing price of BSE data using SADHS-OELM based RBF.

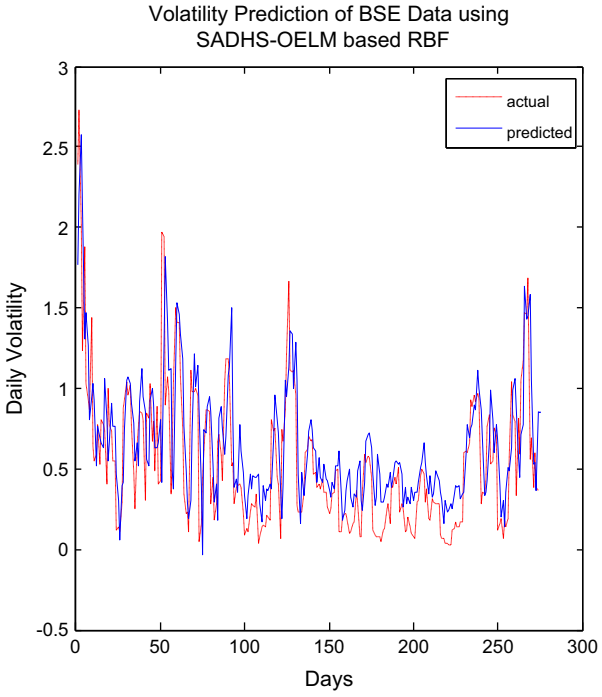

Fig. 8. One day ahead prediction of daily volatility of BSE data using SADHS-OELM based RBF.

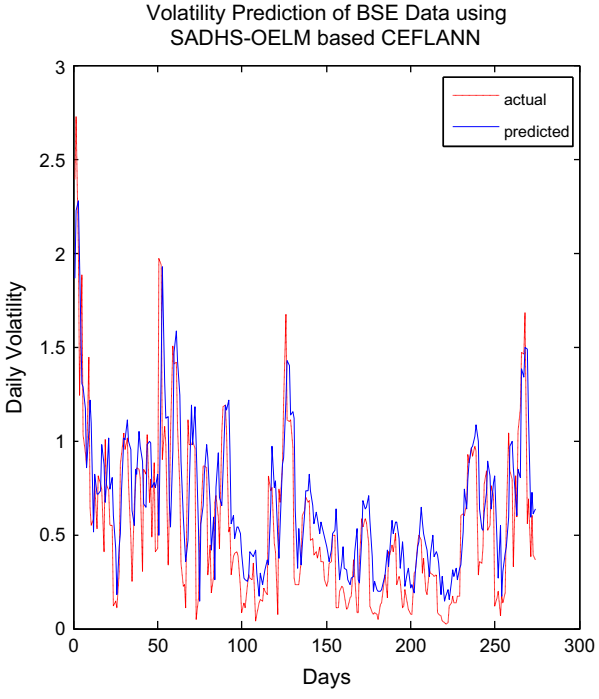

Fig. 7. One day ahead prediction of daily volatility of BSE data using SADHS-OELM based CEFLANN.

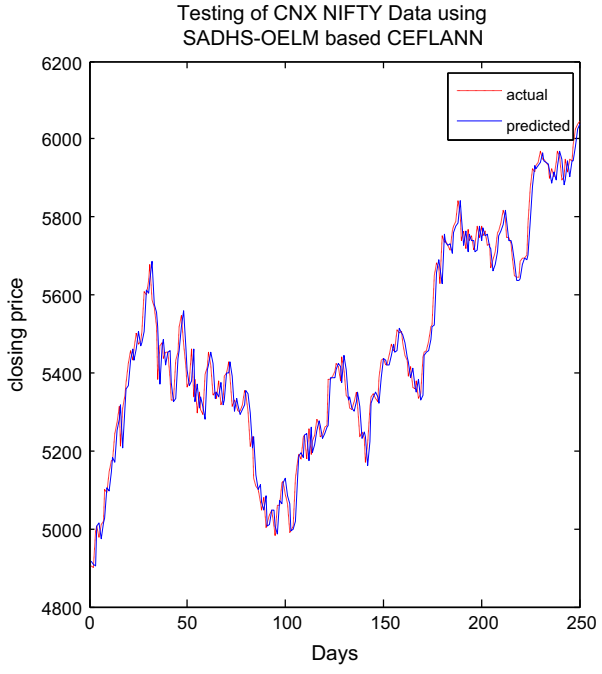

Fig. 9. One day ahead prediction of daily closing price of CNX data using SADHS-OELM based CEFLANN.

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

the weights of connections between the input layer and the hidden layer, the bias of neurons of the hidden-layer, and the regularization factor of robust least square are done using SADHS. SADHS is a variant of harmony search technique that uses the current to best mutation scheme of DE in the pitch adjustment operation for harmony improvisation process. The proposed learning algorithm has been applied on two SLFN i.e. RBF and a low complexity Functional link Artificial Neural Network (CEFLANN) for prediction of stock index and volatility of five different stock indices. The proposed learning scheme has been also compared with other learning schemes like ELM, DE-OELM, DE, SADHS and two other variants of harmony search.

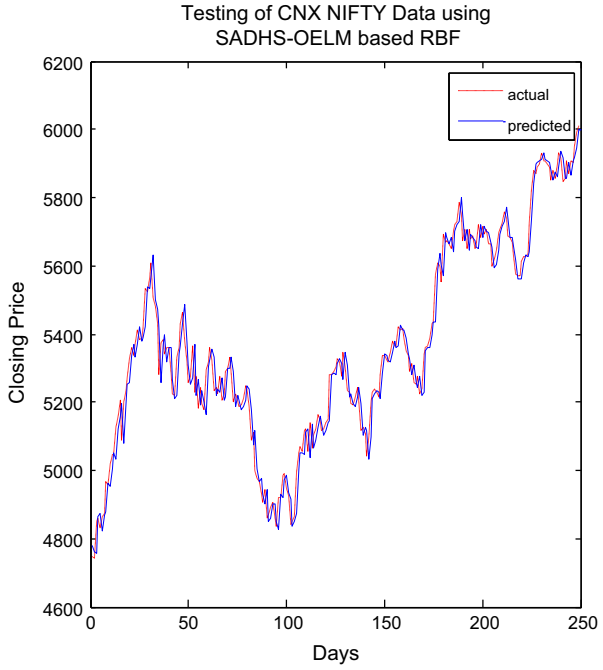

Fig. 10. One day ahead prediction of daily closing price of CNX data using SADHS-OELM based RBF.

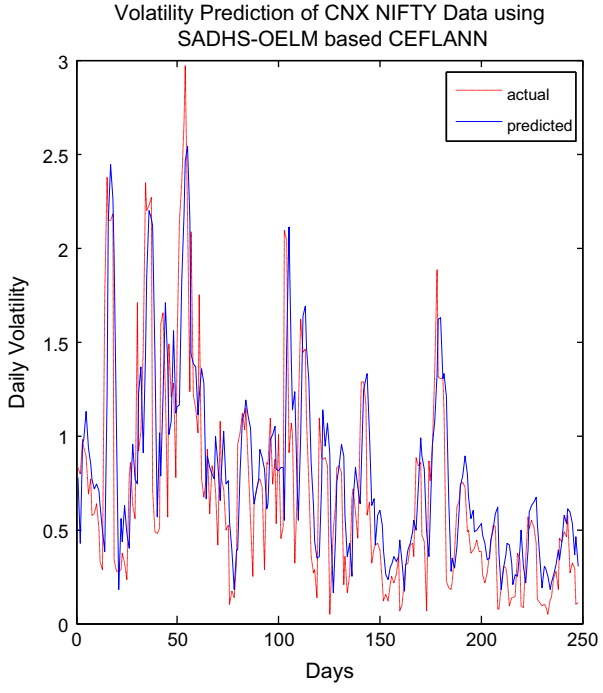

Fig. 11. One day ahead prediction of daily volatility of CNX data using SADHS-OELM based CEFLANN.

2. Single hidden layer feed forward neural network

A single hidden layer feed forward neural network consists of three types of layers. The first layer is the input layer and corresponds to the problem input variables with one node for each input variable. The second layer is the hidden layer used to capture non-linear relationships among variables. The third layer is the output layer used to provide predicted values. The number of nodes in the input and output layers is equal to the dimension of the input and output patterns. Since no processing is done in the

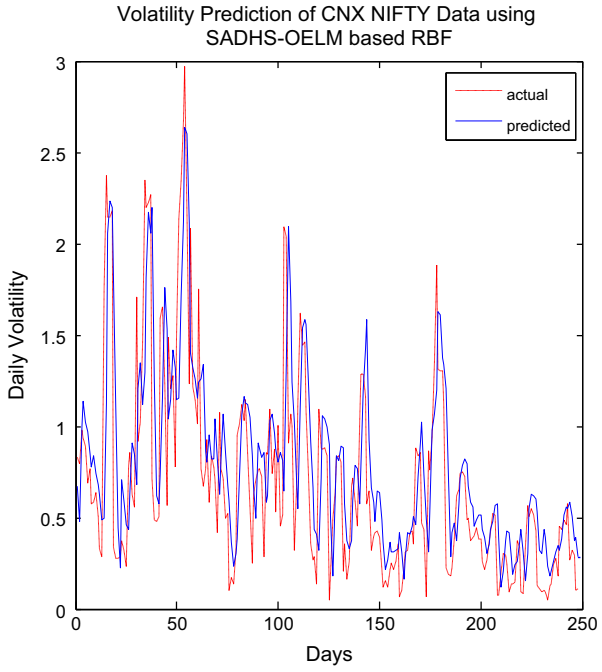

Fig. 12. One day ahead prediction of daily volatility of CNX data using SADHS-OELM based RBF.

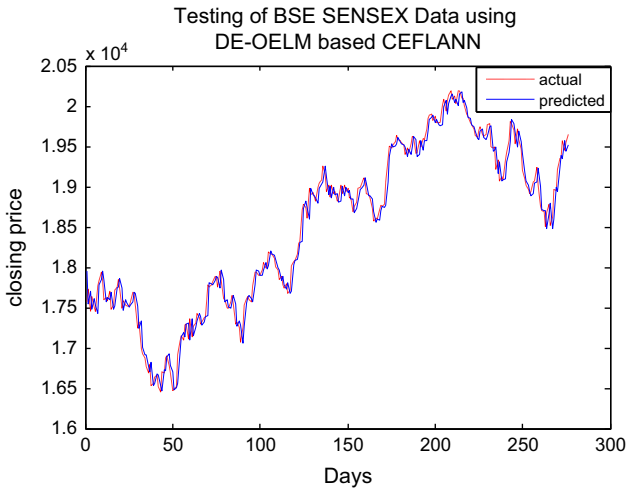

Fig. 13. One day ahead prediction of daily closing price of BSE data using DE-OELM based CEFLANN.

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

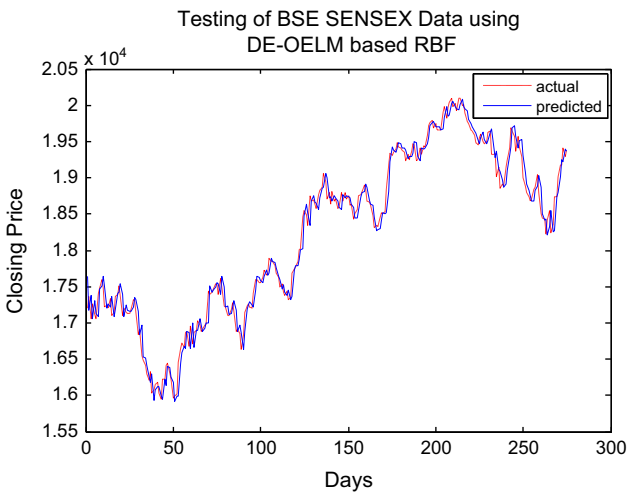

Fig. 14. One day ahead prediction of daily closing price of BSE data using DE-OELM based RBF.

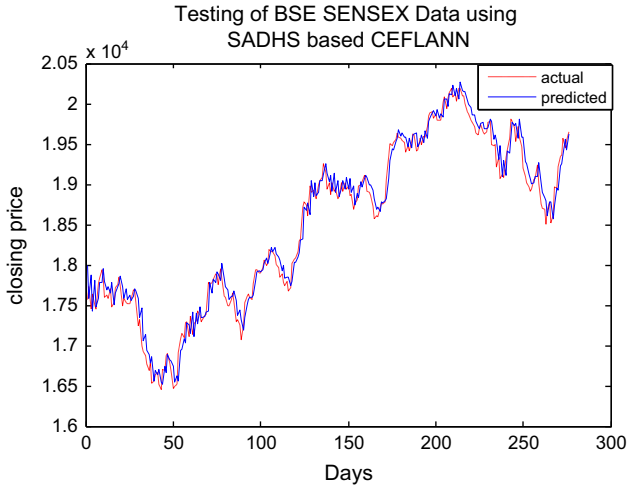

Fig. 15. One day ahead prediction of daily closing price of BSE data using SADHS based CEFLANN.

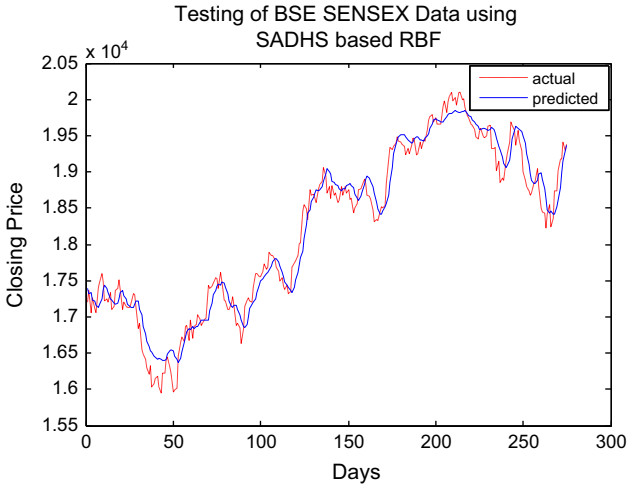

Fig. 16. One day ahead prediction of daily closing price of BSE data using SADHS based RBF.

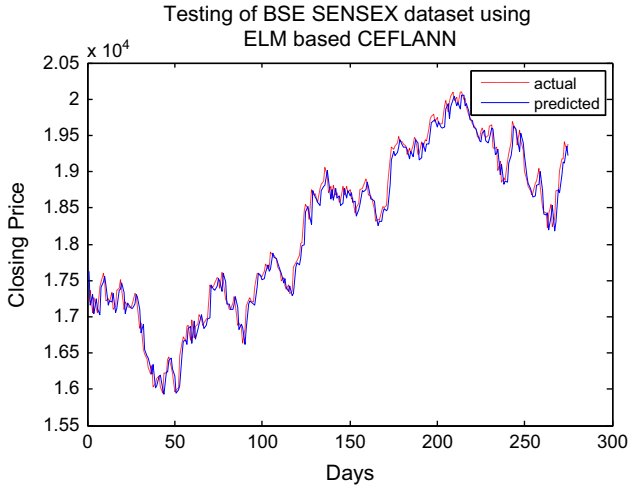

Fig. 17. One day ahead prediction of daily closing price of BSE data using ELM based CEFLANN.

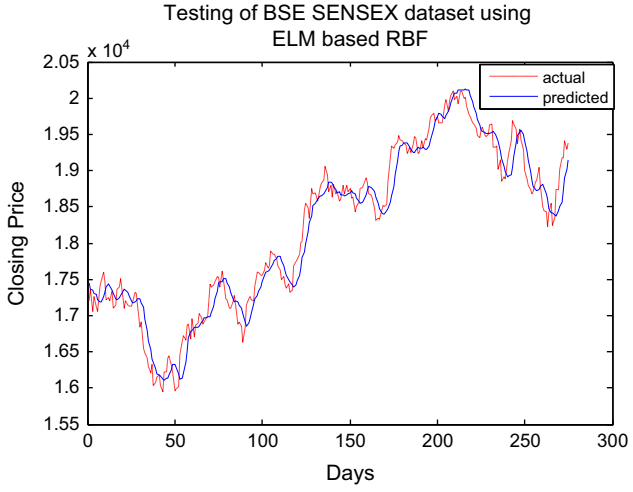

Fig. 18. One day ahead prediction of daily closing price of BSE data using ELM based RBF.

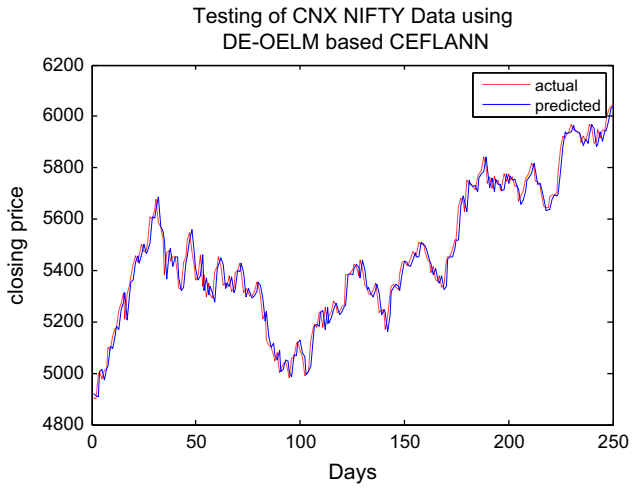

Fig. 19. One day ahead prediction of daily closing price of CNX data using DE-OELM based CEFLANN.

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

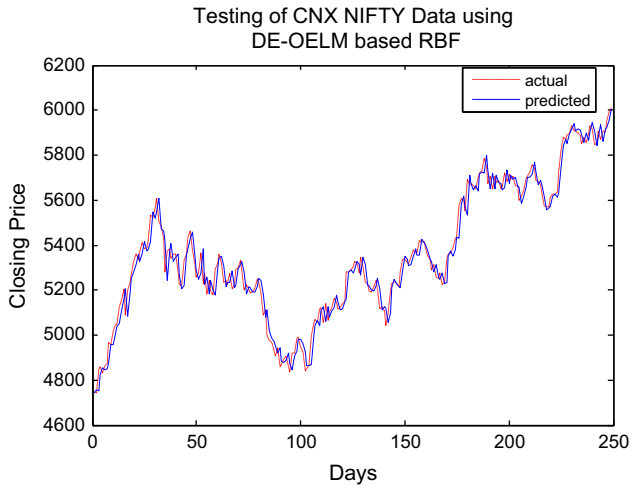

Fig. 20. One day ahead prediction of daily closing price of CNX data using DE-OELM based RBF.

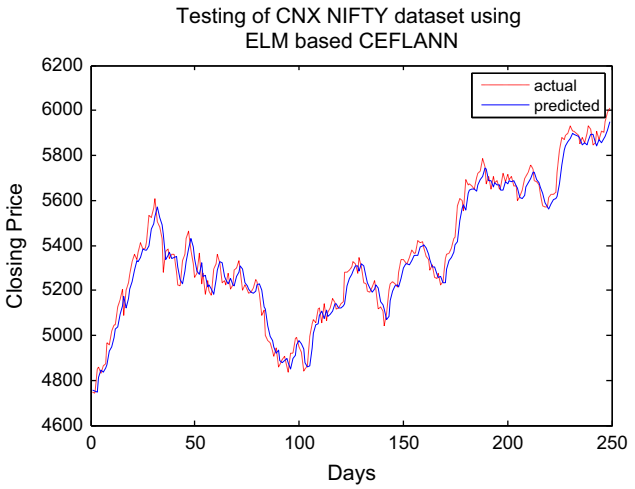

Fig. 23. One day ahead prediction of daily closing price of CNX data using ELM based CEFLANN.

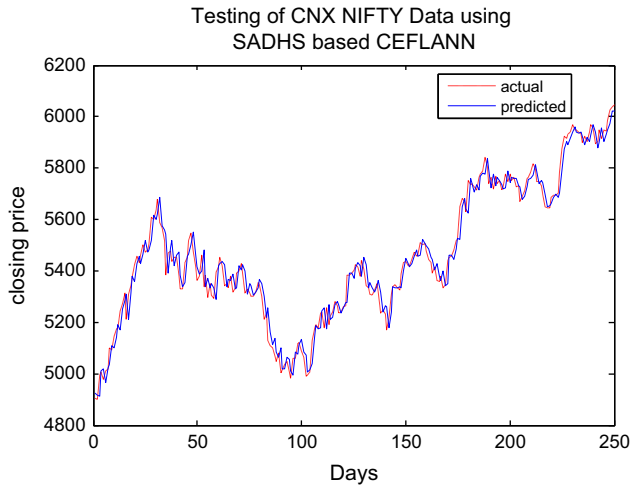

Fig. 21. One day ahead prediction of daily closing price of CNX data using SADHS based CEFLANN.

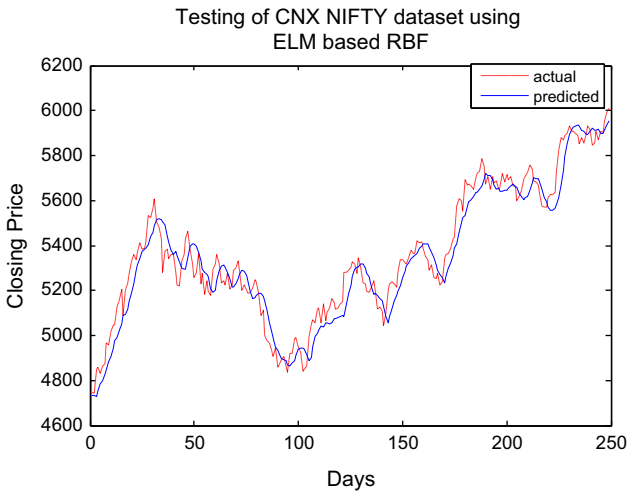

Fig. 24. One day ahead prediction of daily closing price of CNX data using ELM based RBF.

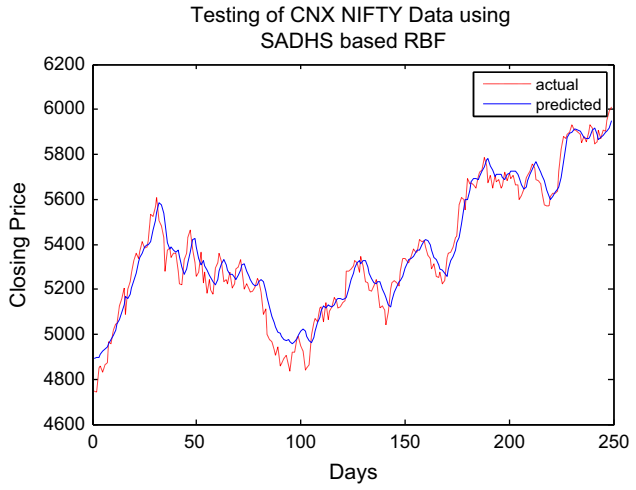

Fig. 22. One day ahead prediction of daily closing price of CNX data using SADHS based RBF.

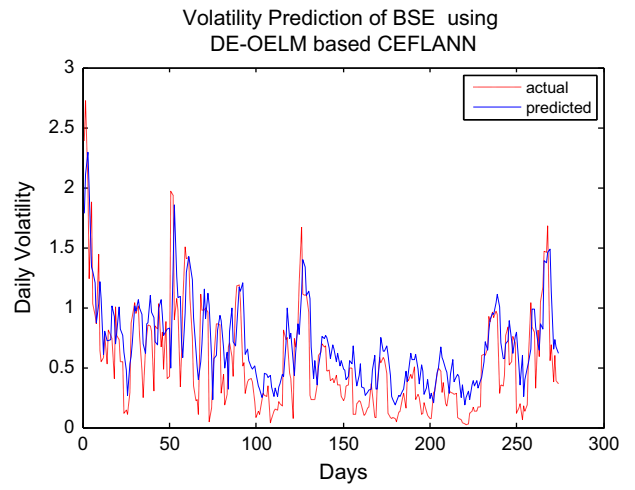

Fig. 25. One day ahead prediction of daily volatility of BSE data using DE-OELM based CEFLANN.

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

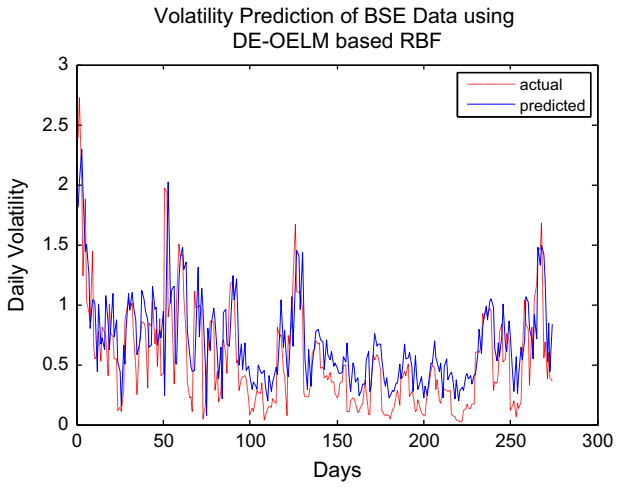

Fig. 26. One day ahead prediction of daily volatility of BSE data using DE-OELM based RBF.

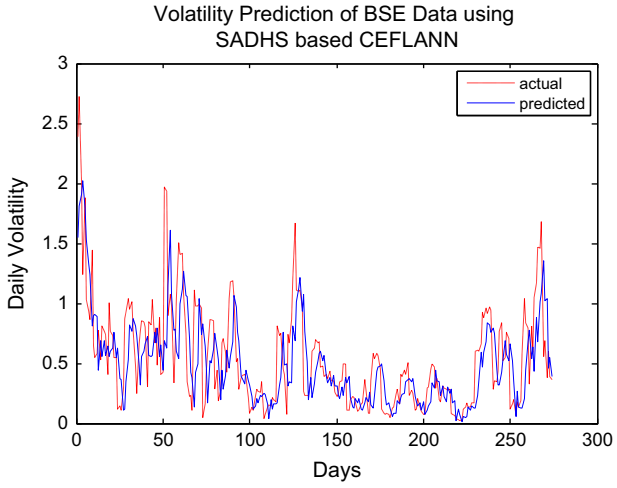

Fig. 27. One day ahead prediction of daily volatility of BSE data using SADHS based CEFLANN.

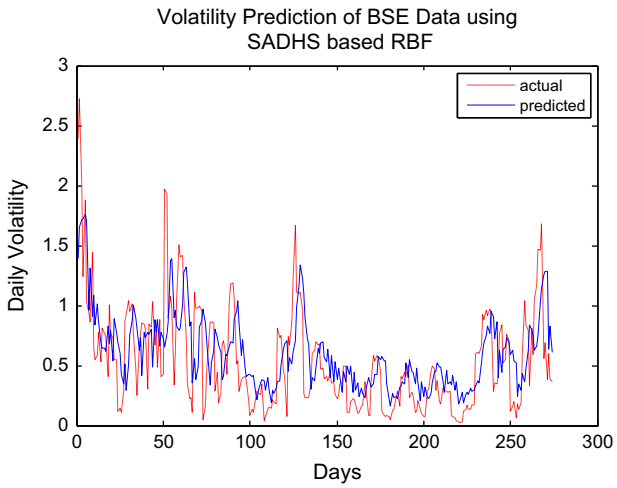

Fig. 28. One day ahead prediction of daily volatility of BSE data using SADHS based RBF.

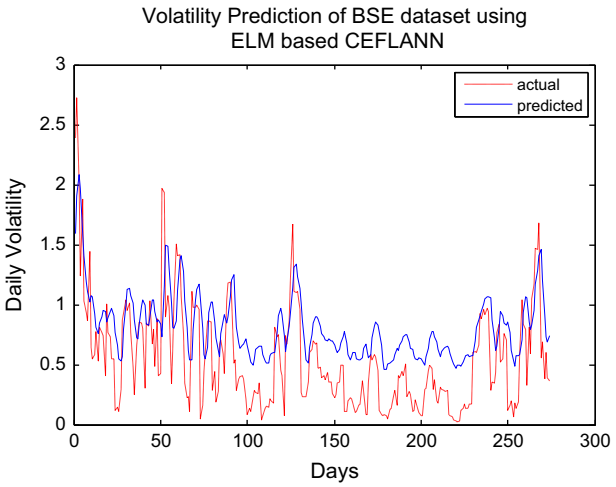

Fig. 29. One day ahead prediction of daily volatility of BSE data using ELM based CEFLANN.

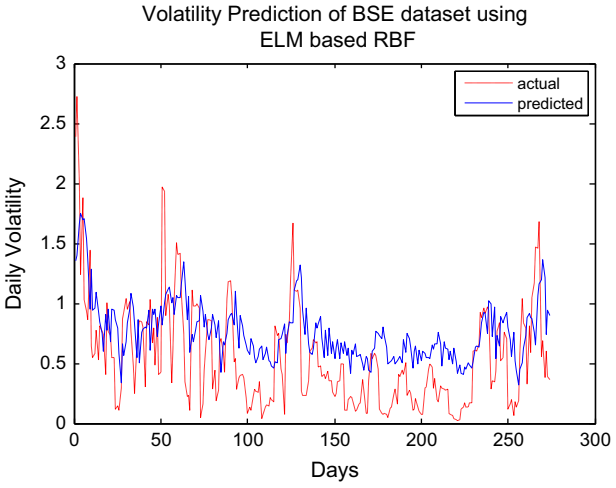

Fig. 30. One day ahead prediction of daily volatility of BSE data using ELM based RBF.

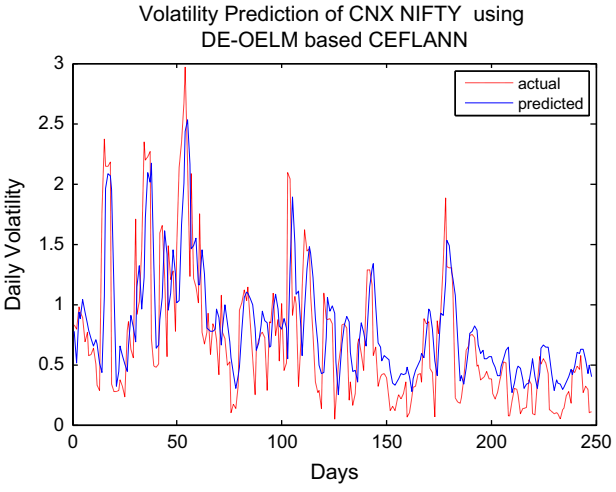

Fig. 31. One day ahead prediction of daily volatility of CNX data using DE-OELM based CEFLANN.

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

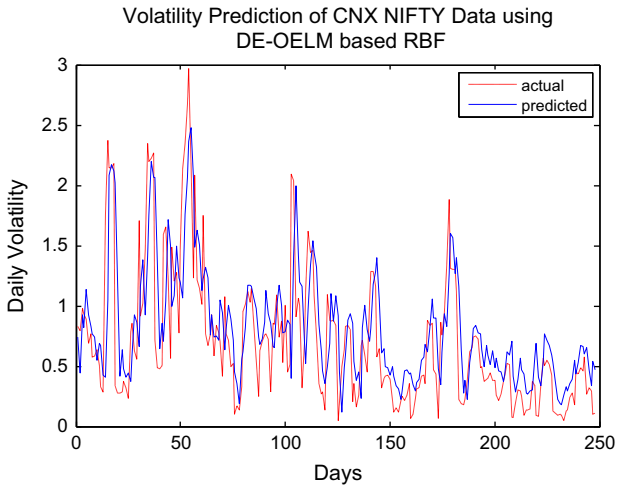

Fig. 32. One day ahead prediction of daily volatility of CNX data using DE-OELM based RBF.

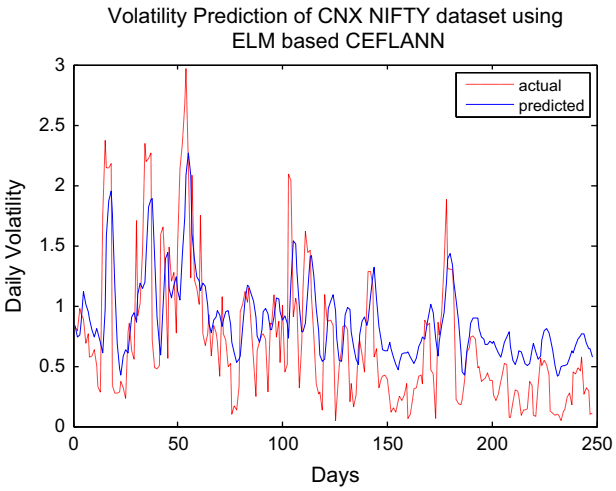

Fig. 35. One day ahead prediction of daily volatility of CNX data using ELM based CEFLANN.

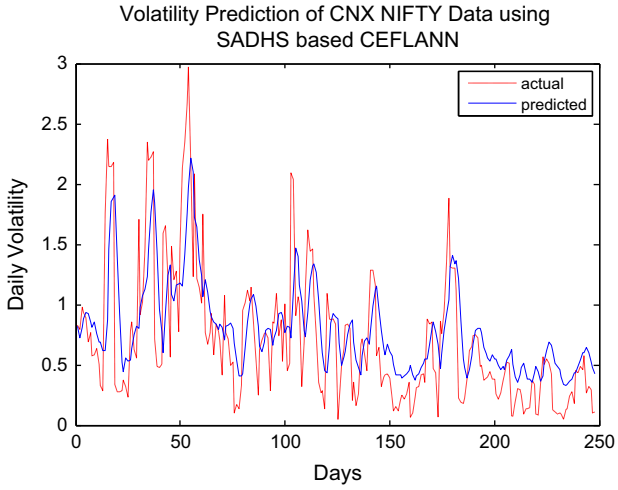

Fig. 33. One day ahead prediction of daily volatility of CNX data using SADHS based CEFLANN.

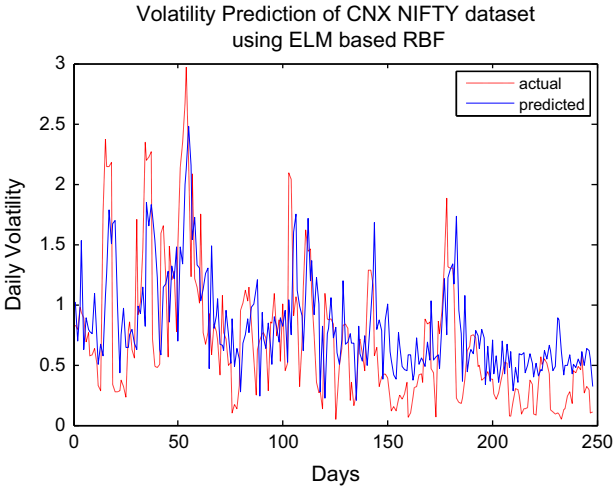

Fig. 36. One day ahead prediction of daily volatility of CNX data using ELM based RBF.

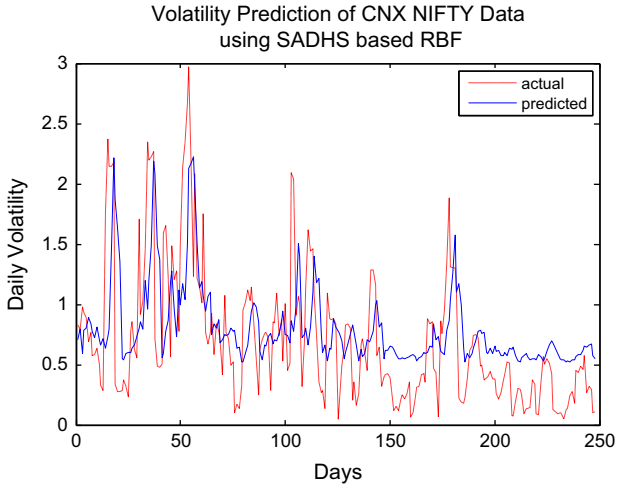

Fig. 34. One day ahead prediction of daily volatility of CNX data using SADHS based RBF.

input layer, the output of its nodes is equal to the input pattern itself. With d input variables and M number of hidden layer nodes the output of hidden layer node y_j and output of output layer node z_k are as follows:

$$oy_j = f\left(\sum_{i=1}^{d} w_{ij}x_i + \theta_j\right) \tag{1}$$

$$oz_k = g\left(\sum_{j=1}^{M} v_{jk}oy_j + \theta_k\right) \tag{2}$$

where $f(\ )$ is the activation function used for hidden layer node and $g(\ )$ is the activation function used for output layer nodes. x_i represents the input nodes, y_j represents the hidden nodes and z_k represents the output nodes, w_{ij} represents the weights between input and hidden nodes and v_{jk} represents the weights between hidden and output nodes. During training phase, an input pattern is applied to the SLFN and the node outputs of all the layers are computed. The output of the output layer nodes is compared with

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

the desired output to generate an error signal e_k. This error is used to update the weights of the network using learning algorithm.

2.1. Radial basis function network (RBFN)

Being a universal approximator, Radial basis function network (RBFN) has gained its popularity in different application fields. Because of the simple three layer architecture, RBFN has advantage of easy design, learning, strong tolerance to input noise and good generalization ability [11,16,27]. It has the same structure as a general SLFN with only difference that there are no connection weights between the input and hidden layer. Again in hidden layer a set of radial basis functions are used for calculating the output of hidden layer nodes. Gaussian and Logistic functions are the commonly used radial basis functions. Normally the hidden layer of RBFN is nonlinear where as the output layer is linear. Three types of parameters (i.e. the centers, and widths of RBFs, connection weights of hidden layers) are required to be optimized during training of the network. With a d dimensional input vector and M number of hidden layer nodes using Gaussian radial basis function the output of hidden layer node R_j and output of output layer node y_k is calculated as follows:

$$R_j(x) = \exp\left[-\frac{1}{2}\left(\frac{||x-c_j||^2}{\sigma_j^2}\right)\right] \tag{3}$$

$$y_k = \sum_{j=1}^{M} w_{jk} R_j(x) + w_{0k} \tag{4}$$

where c_j is the centre and σ_j is the deviation of the jth radial basis function, w_{jk} is the connection weights between jth hidden layer and kth output layer node. Then the error obtained by comparing the output with desired output is used to update the weights and parameters of the radial basis functions using learning algorithm Fig. 1.

2.2. Computationally efficient functional link artificial neural network (CEFLANN)

Computationally efficient FLANN is a single hidden layer ANN that uses trigonometric basis functions for functional expansion. Unlike earlier FLANNS, where each input in the input pattern is expanded through a set of nonlinear functions, here all the inputs of the input pattern passes through a few set of nonlinear functions to produce the expanded input pattern [28,29]. Fig. 2 depicts the single layer computationally efficient FLANN architecture. With the order p any d dimensional input pattern $X=[x_1, x_2,\ldots,x_d]^T$ is expanded to a m dimensional pattern CX by Trigonometric functional expansion as $CX=[cx_1, cx_2,\ldots,cx_d, cx_{d+1}, cx_{d+2},\ldots, cx_m]^T=[x_1, x_2,\ldots,x_d, cx_{d+1}, cx_{d+2},\ldots,cx_m]^T$ where $m=d+p$. For each order in p, the weighted sum of the components of the original input pattern is passed through a hyperbolic tangent (tanh( )) non linear function to produce an output o which is stored in cx_l(with $d+1 \le l \le m$). Each o_i is obtained using the following formula.

$$o_i = \tanh\left(a_{i0} + \sum_{i=1,j=1}^{i=p,j=d} a_{ij} \times x_j\right) \tag{5}$$

where a_{ij} is the associated parameter i.e. $o_1=\tanh(a_{10}+a_{11}x_1+a_{12}x_2+\cdots+a_{1d}x_d)$; $o_2=\tanh(a_{20}+a_{21}x_1+a_{22}x_2+\cdots+a_{2d}x_d)$; $o_p=\tanh(a_{p0}+a_{p1}x_1+a_{p2}x_2+\cdots+a_{pd}x_d)$.

After expansion, weight is initialized for each expanded unit and then the weighted sum of the components of the enhanced input pattern produces the output y using the following equation.

$$y_k = \sum_{j=1}^{m} w_{jk} cx_j \tag{6}$$

where w_{jk} is the connection weight between jth expanded input and kth output layer node. The error obtained by comparing the o/p with desired o/p is used to update the weights of the FLANN structure by a weight updating algorithm.

3. Extreme learning machine

Extreme learning machine is a recently introduced learning algorithm for single-hidden layer feed-forward neural networks (SLFNs) which randomly chooses the input weights (weights of connections between the input variables and neurons in the hidden layer) and the bias of neurons in the hidden layer and analytically determines the output weights instead of iterative tuning [21]. ELM not only has the capability of extremely fast learning and testing speed but also tends to achieve better generalization performance without intensive human intervention. The main advantage of ELM is that the hidden layer of SLFNs needs no tuning and it can work with a wide range of activation

Table 9
p and h values of Wilcoxon signed rank test.

Network model	Data set	SADHS-OELM/DE-OELM	SADHS-OELM/SADHS	SADHS-OELM/SGHS	SADHS-OELM/HS	SADHS-OELM/DE	SADHS-OELM/ELM	SADHS-OELM/BP
CEFLANN (stock price prediction)	BSE (p)	0.0126	2.7039e−004	1.2670e−014	5.4751e−007	1.5694e−004	3.3728e−038	1.5324e−036
	(h)	1	1	1	1	1	1	1
	CNX (p)	0.0330	0.4182	4.3090e−014	2.2285e−005	0.0056	1.0252e−035	5.4437e−034
	(h)	1	0	1	1	1	1	1
RBF (stock price prediction)	BSE (p)	0.2124	5.6444e−013	2.3836e−021	3.0078e−024	3.4190e−018	4.0447e−014	5.7053e−015
	(h)	0	1	1	1	1	1	1
	CNX (p)	0.7235	6.7775e−007	1.45e−017	3.1948e−019	1.337e−013	2.2165e−017	3.4196e−013
	(h)	0	1	1	1	1	1	1
CEFLANN (volatility prediction)	BSE (p)	2.0652e−009	0.3223	1.3774e−009	1.7990e−019	1.0826e−004	1.7735e−024	6.4302e−020
	(h)	1	0	1	1	1	1	1
	CNX (p)	3.0443e−004	0.0018	2.9429e−015	1.4639e−008	4.4992e−009	1.3080e−010	2.3602e−011
	(h)	1	1	1	1	1	1	1
RBF (volatility prediction)	BSE (p)	0.0043	0.0057	3.0787e−015	1.0568e−016	2.2895e−005	1.5154e−017	3.5964e−010
	(h)	1	1	1	1	1	1	1
	CNX (p)	0.0968	3.1173e−006	1.1695e−012	3.5054e−013	4.9923e−90	2.0647e−008	5.2006e−011
	(h)	1	1	1	1	1	1	1

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

functions including piecewise continuous functions [22,23,25]. With a given a set of N training dataset $D=(x_i, y_i)$, $i=1$ to N (where each x_i is a d-dimensional input pattern and y_i is the desired output), activation function for hidden layer nodes, M number of hidden layer nodes and a linear activation function in the output neuron, the output function of ELM for SLFN can be represented as:

$$y_i = \sum_{j=0}^{M} w_j h_j(x_i) \tag{7}$$

where $h_j(x)$ is the activation function of hidden layer and W is the weight vector connecting the hidden layer neurons to output layer neuron. If the hidden layer neurons are of the radial basis function type (RBF), the H matrix is obtained as

$$H = \begin{bmatrix} h(x_1, c_1, \sigma_1) \cdots h(x_1, c_M, \sigma_M) \\ \vdots \\ h(x_N, c_1, \sigma_1) \cdots h(x_N, c_M, \sigma_M) \end{bmatrix} \tag{8}$$

and $W=[w_1, w_2, \ldots, w_M]^T$, $Y=[y_1, y_2, \ldots, y_N]^T$, $T=$transpose of a quantity; c, σ are the center and standard deviation of the basis functions.

Eq. (7) can be written as

$$Y = HW \tag{9}$$

where H is a $N \times (M+1)$ hidden layer feature mapping matrix in which ith row specifies the hidden layer's output vector for an instance x_i.

Eq. (9) being a linear system can be solved by

$$W = H^{\Psi}T = (H^T H)^{-1} H^T \tag{10}$$

where H^{Ψ} is the Moore–Penrose generalized inverse of matrix H.

The computation of the weight vector of the ELM given in Eq. (10) is done using all the data samples at a time. However, for time series forecasting an on-line formulation for ELM known as OS-ELM has been given in [22] using each new observation or chunk of observations as

$$W(k+1) = W(k) + P(k+1)H^T(k+1)[Y(k+1) - H(k+1)W^T] \tag{11}$$

$$P(k+1) = P(k) - P(k)H^T(k+1)[I + H(k+1)P(k)H^T(k+1)]^{-1}H(k+1)P(k) \tag{12}$$

and $P(k+1) = [K(k+1)]^{-1}$, I is an unity matrix of appropriate dimension.

The initial values are obtained using the Moore–Penrose generalized inverse as follows:

$$W(0) = [H^T(0)H(0)]^{-1}H^T(0)Y(0)$$
$$K(0) = H^T(0)H(0) \tag{13}$$

Since in ELM the output weights are computed based on the random input weights and bias of the hidden nodes, there may be inclusion of a set of non-optimal or unnecessary input weights and bias of the hidden nodes. The random initialization of hidden node parameters may affect the performances of ELM, where, human choices, like the number of nodes or the type of activation functions, also have a great impact on the network's usefulness. Furthermore, the ELM tends to require more hidden neurons than conventional tuning based learning algorithms in some applications, which can negatively affect SLFN performance in unknown testing data. Under practical working or testing conditions, its performance is also sensitive to noisy input which may give rise to unsatisfying accuracy [24,26]. The other issue in ELM is in choosing the optimal number of hidden nodes for a given problem which is usually done by trial and error method. The next section describes the Harmony search technique for choosing optimum weights and biases of the hidden neurons in order to provide a more robust learning of the ELM.

4. Harmony search

Harmony search (HS) is a recently developed meta-heuristic algorithm that mimics the improvisation process of searching for a perfect state of harmony by the music players [30]. Due to its simplicity, few parameters, and easy implementation, the HS algorithm has captured much attention and has been successfully applied to a wide range of real-world problems [31,32]. Compared to earlier meta-heuristic optimization algorithms, the HS algorithm imposes

Table 10
Performance comparison of RBF and CEFLANN with different evolutionary learning algorithms for one day ahead stock price prediction of BSE Sensex and CNX Nifty data set (with population size=50).

Learning algorithms		BSE Sensex				CNX Nifty			
		RBF		CEFLANN		RBF		CEFLANN	
		RMSE	MAPE	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE
SADHS-OELM	**Min**	**0.0182**	**0.9052**	**0.0163**	**0.5633**	**0.0203**	**0.7356**	**0.0181**	**0.6409**
	Avg	**0.0183**	**0.9117**	**0.0165**	**0.5738**	**0.0205**	**0.7400**	**0.0182**	**0.6458**
	Std	**0.0002**	**0.0083**	**0.0001**	**0.0065**	**0.0002**	**0.0075**	**0.0001**	**0.0039**
DE-OELM	**Min**	0.0186	0.6584	0.0164	0.5645	0.0205	0.7398	0.0182	0.6452
	Avg	0.0189	0.6717	0.0168	0.5849	0.0208	0.7534	0.0183	0.6492
	Std	0.0003	0.0137	0.0002	0.0074	0.0004	0.0163	0.0001	0.00450
SADHS	**Min**	0.0259	0.9627	0.0166	0.5784	0.0261	1.0029	0.0190	0.6638
	Avg	0.0305	0.1203	0.0182	0.6349	0.0314	1.2035	0.0204	0.7309
	Std	0.0040	0.1398	0.0014	0.0543	0.0032	0.1236	0.0012	0.0519
SGHS	**Min**	0.0271	1.0122	0.0179	0.6189	0.0325	1.2319	0.0199	0.7138
	Avg	0.0356	1.3142	0.0187	0.6564	0.0377	1.4467	0.0240	0.8771
	Std	0.0058	0.1994	0.0007	0.0278	0.0045	0.1858	0.0044	0.1879
HS	**Min**	0.0273	0.9967	0.0179	0.6196	0.0265	1.0151	0.0184	0.6541
	Avg	0.0323	1.1869	0.0191	0.6728	0.0299	1.1376	0.0192	0.6860
	Std	0.0043	0.1605	0.0017	0.0677	0.0027	0.1099	0.0006	0.0235
DE	**Min**	0.0257	0.9487	0.0174	0.6008	0.0290	1.1251	0.0194	0.6940
	Avg	0.0287	1.0564	0.0181	0.6290	0.0347	1.3269	0.0230	0.8351
	Std	0.0024	0.0807	0.0007	0.0303	0.0033	0.1151	0.0031	0.1189

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

limited mathematical requirements and is not sensitive to the initial value settings. The key concepts of HS algorithm are musicians, notes, harmonies, and harmony memory. In most optimization problems solvable by HS, the musicians are the decision variables of the function being optimized. The notes played by the musicians are the values each decision variable can take. The harmony contains the notes played by all musicians, namely a solution vector containing one value per variable. Harmony memory contains harmonies played by the musicians, or it can be viewed as the storage place for solution vectors. Original HS uses five parameters, including three core parameters such as the size of harmony memory (HMS), the harmony memory consideration rate (HMCR), and the maximum number of iterations (NI) and two optional ones such as the pitch adjustment rate (PAR) and the adjusting bandwidth (BW). The number of musicians is defined by the problem itself and is equal to the number of variables in the optimization function. First, an initial population of harmony vectors are randomly generated and sorted in a harmony memory (HM). Then a new candidate harmony is generated from the HM by using a memory consideration, a pitch adjustment, and a random selection. Finally, the HM is updated by comparing the new candidate harmony with the existing worst harmony vector in the current HM. The above process is repeated until the termination criterion is satisfied. The parameter HMCR controls the balance between exploration and exploitation and takes value between 0 and 1. PAR determines whether further adjustment is required according to BW parameter and can be visualized as local search. To improve the performance of original harmony search algorithm a no of variants of harmony search has been discussed in literature [18,33–35].

4.1. Standard harmony search algorithm

Step 1: Initialize the parameters (HMS, HMCR, PAR, BW, NI).
Step 2: Initialize the HM randomly and calculate fitness function value of each harmony vector.
Step 3: Improvise a new harmony (X_{new}) from HM as follows:
Step 3.1: For ($j=1$ to n) do
Step 3.2: If (rand (0, 1) < HMCR) then $X_{new}(j)=X_a(j)$ where $a \in (1,2,\ldots,HMS)$

Step 3.3: If (rand (0, 1) < PAR) then

$$X_{new}(j) = X_{new}(j) \pm r3 \times BW \quad (14)$$

Step 3.4: Else

$$X_{new}(j) = LB(j) + r \times (UB(j) - LB(j)) \quad (15)$$

Step 4: If $f(X_{new})$ is better than the f(worst) update the HM as $X_{worst}=X_{new}$
Step 5: Repeat steps 3 and 4 until NI is reached. Best harmony vector X_{best} in the HM is the solution to the problem.

5. Proposed self adaptive differential harmony search based optimized ELM

For SLFN ELM is a much faster learning machine with better generalization performance than other learning algorithms, but the stochastic nature of the hidden layer output matrix may lower its learning accuracy. Since in ELM the output weights are computed based on the random input weights and bias of the hidden nodes, there may be inclusion of a set of non optimal or unnecessary input weights and bias of the hidden nodes. The random initialization of hidden node parameters may affect the performances of ELM, where, human choices, like the number of nodes or the type of activation functions, also have a great impact on the network's usefulness. Under practical working or testing conditions, its performance is also sensitive to noisy input which may give rise to unsatisfying accuracy. Hence in order to improve the generalization ability of ELM in presence of irrelevant and noisy input variables, a hybrid learning framework named Self Adaptive Differential Harmony Search Based Optimized Extreme Learning Machine (SADHS-OELM) has been proposed in this paper. The proposed approach uses the same concept of the ELM where the output weights are obtained analytically using a robust least squares solution including a regularization parameter but, in order to optimize the fitting performance, the selection of the hidden layer parameters, and the regularization factor of robust least square are done using SADHS. The use of least squares method with regularization parameter will help in improving the performance of ELM in

Table 11
Performance comparison of RBF and CEFLANN with different evolutionary learning algorithms for one day ahead volatility prediction of BSE Sensex data set (with population size=50).

Learning algorithms		RBF			CEFLANN		
		MSFE	RMSFE	MAFE	MSFE	RMSFE	MAFE
SADHS-OELM	**Min**	**0.0927**	**0.3045**	**0.2401**	**0.0888**	**0.2979**	**0.2215**
	Avg	**0.0934**	**0.3056**	**0.2348**	**0.0925**	**0.3041**	**0.2279**
	Std	**0.0038**	**0.0062**	**0.0080**	**0.0019**	**0.0031**	**0.0033**
DE-OELM	**Min**	0.0988	0.3143	0.2450	0.0911	0.3019	0.2294
	Avg	0.1036	0.3219	0.2604	0.0959	0.3096	0.2382
	Std	0.0065	0.0104	0.0119	0.0032	0.0051	0.0063
SADHS	**Min**	0.1581	0.3976	0.3193	0.1044	0.3231	0.2499
	Avg	0.2151	0.4621	0.3925	0.1315	0.3614	0.2969
	Std	0.0405	0.0446	0.0496	0.0230	0.0320	0.0364
SGHS	**Min**	0.1693	0.4115	0.3375	0.1328	0.3644	0.2787
	Avg	0.2722	0.5163	0.4460	0.2074	0.4486	0.3677
	Std	0.0883	0.0844	0.0873	0.0832	0.0878	0.1014
HS	**Min**	0.2016	0.4490	0.3826	0.1191	0.3452	0.2803
	Avg	0.2373	0.4863	0.4177	0.1542	0.3915	0.3261
	Std	0.0317	0.0327	0.0325	0.0256	0.0332	0.0345
DE	**Min**	0.1480	0.3752	0.3075	0.1155	0.3398	0.2604
	Avg	0.1842	0.4279	0.3555	0.1417	0.3750	0.3106
	Std	0.0301	0.0359	0.0347	0.0291	0.0375	0.0459

Table 12
Performance comparison of RBF and CEFLANN with different evolutionary learning algorithms for one day ahead volatility prediction of CNX Nifty Data set (with population size=50).

Learning algorithms		RBF			CEFLANN		
		MSFE	RMSFE	MAFE	MSFE	RMSFE	MAFE
SADHS-OELM	**Min**	**0.1605**	**0.4006**	**0.2823**	**0.1572**	**0.3964**	**0.2815**
	Avg	**0.1630**	**0.4037**	**0.2918**	**0.1628**	**0.4035**	**0.2865**
	Std	**0.0029**	**0.0036**	**0.0064**	**0.0037**	**0.0046**	**0.0029**
DE-OELM	**Min**	0.1665	0.4081	0.2952	0.1578	0.3972	0.2871
	Avg	0.1706	0.4130	0.3044	0.1637	0.4045	0.2940
	Std	0.0027	0.0033	0.0065	0.0041	0.0051	0.0039
SADHS	**Min**	0.2218	0.4710	0.3655	0.1908	0.4368	0.3268
	Avg	0.2728	0.5210	0.4113	0.2042	0.4518	0.3468
	Std	0.0440	0.0416	0.0475	0.0083	0.0093	0.0135
SGHS	**Min**	0.2865	0.5352	0.4168	0.2092	0.4574	0.3383
	Avg	0.3164	0.5622	0.4614	0.3214	0.5596	0.4543
	Std	0.0216	0.0192	0.0369	0.1177	0.1019	0.1160
HS	**Min**	0.2794	0.5286	0.4124	0.1998	0.4470	0.3543
	Avg	0.3153	0.5610	0.4569	0.2182	0.4667	0.3774
	Std	0.0301	0.0269	0.0349	0.0206	0.0218	0.0247
DE	**Min**	0.2349	0.4847	0.3810	0.1939	0.4403	0.3439
	Avg	0.2959	0.5431	0.4387	0.2167	0.4652	0.3655
	Std	0.0374	0.0354	0.0400	0.0184	0.0198	0.0211

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

presence of noisy data. Again SADHS is a variant of harmony search technique that uses the current to best mutation scheme of DE in the pitch adjustment operation for harmony improvisation process [34]. The steps of SADHS-OELM for learning RBF and CEFLANN are given as follows:

Step 1: Initialize the parameters (HMS, HMCR, PAR, BW, NI).
Step 2: Initialize the harmony memory HM randomly specifying the hidden layer parameters (i.e. center, spread of radial basis functions used in hidden nodes of RBF or the associated parameters needed for functional expansion in CEFLANN).
Step 3: Set $t=0$.
Step 4: For each harmony vector in HM repeat steps 4.1 to 4.3.
Step 4.1: Calculate the hidden layer output matrix W.
Step 4.2: Find the output weight matrix W using robust least square solution as follows:

$$W = H^{\psi}T = (H^T W + \alpha I)^{-1} H^T \quad (16)$$

where $\alpha > 0$ is the regularization parameter and I is the $M \times M$ identity matrix.
Step 4.3: Calculate RMSE error of desired output and output obtained using the harmony vector and corresponding output weight vector as the fitness function value of each harmony vector.
Step 5: Improvise a new harmony (X_{new}) from HM as follows:
Step 5.1: For ($j=1$ to n) do.
Step 5.2: If (rand (0, 1) < HMCR) then $X_{new}(j)=X_a(j)$ where $a \in (1,2,\ldots,HMS)$.
Step 5.3: If (rand (0, 1) < PAR) then

$$X_{new}(j) = X_{new}(j) + BW \times (X_{best}(j) - X_{new}(j)) + BW \times (X_b(j) - X_c(j)) \quad (17)$$

where $b,c \in (1,2,\ldots,HMS)$.
Step 5.4: Else $X_{new}(j)=LB(j)+\text{rand}(0,1)\times(UB(j)-LB(j))$.
Step 6: If $f(X_{new})$ is better than the $f(worst)$ update the HM as $X_{worst}=X_{new}$.

Table 13
MAPE Error during testing of SADHS-OELM based CEFLANN with different input size and prediction horizon for stock price prediction.

Data set	Input size	Prediction horizon		
		1	5	10
BSE Sensex	6	0.5739	1.3478	1.8162
	11	0.5618	1.3618	1.8063
	16	0.5572	1.3699	1.7977
CNX Nifty	6	0.6446	1.5489	2.0907
	11	0.6498	1.5529	2.0751
	16	0.6438	1.4962	2.0270

Table 14
MSFE error during testing of SADHS-OELM based CEFLANN with different input size and prediction horizon for volatility prediction.

Data set	Input size	Prediction horizon		
		1	5	10
BSE Sensex	6	0.0952	0.2939	0.3257
	11	0.1391	0.2665	0.3130
	16	0.1801	0.2425	0.2813
CNX Nifty	6	0.1636	0.3792	0.4172
	11	0.2442	0.3664	0.3816
	16	0.3054	0.3517	0.3721

Table 15
Performance comparison of SADHS-OELM based RBF and SADHS-OELM based CEFLANN for one day ahead stock price prediction of Nikkei225, FTSE100 and S&P500 dataset.

Data set	RBF		CEFLANN	
	RMSE	MAPE	RMSE	MAPE
Nikkei 225				
Min	0.0570	1.7782	0.0144	0.8718
Avg	0.0762	2.2727	0.0155	0.9015
Std	0.0219	0.4872	0.0010	0.0305
FTSE100				
Min	0.0509	1.3689	0.0189	0.5977
Avg	0.0589	1.5300	0.0202	0.6421
Std	0.0080	0.1631	0.0013	0.0400
S&P500				
Min	0.0188	0.8198	0.0130	0.5502
Avg	0.0679	2.3116	0.0149	0.6325
Std	0.0308	0.9438	0.0022	0.0925

Table 16
Performance comparison of SADHS-OELM based RBF and SADHS-OELM based CEFLANN for one day ahead volatility prediction of Nikkei225, FTSE100 and S&P500 dataset.

Data set	RBF			CEFLANN		
	MSFE	RMSFE	MAFE	MSFE	RMSFE	MAFE
Nikkei 225						
Min	0.3649	0.6040	0.4315	0.3693	0.6077	0.4261
Avg	0.4338	0.6576	0.4629	0.4144	0.6435	0.4497
Std	0.0519	0.0391	0.0299	0.0226	0.0177	0.0175
FTSE100						
Min	0.1165	0.3412	0.2523	0.1228	0.3505	0.2520
Avg	0.1259	0.3548	0.2649	0.1293	0.3595	0.2581
Std	0.0050	0.0071	0.0078	0.0070	0.0096	0.0061
S&P500						
Min	0.1138	0.3374	0.2396	0.1219	0.3492	0.2332
Avg	0.1235	0.3512	0.2495	0.1383	0.3716	0.2555
Std	0.0093	0.0130	0.127	0.0117	0.0157	0.0105

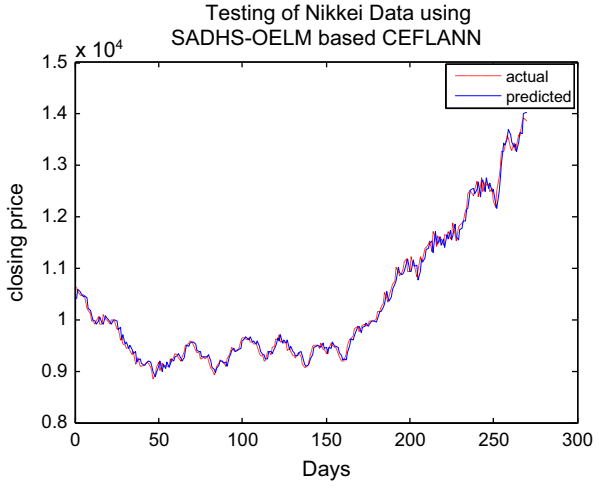

Fig. 37. One day ahead prediction of daily closing price of Nikkei data using SADHS_OELM based CEFLANN.

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

Step 7: The parameters HMCR, PAR, BW at iteration t are adapted as follows:

$$\mathrm{HMCR}(t) = \mathrm{HMCR}_L + (\mathrm{HMCR}_U - \mathrm{HMCR}_L) \times t/\mathrm{NI}$$
$$\mathrm{PAR}(t) = \mathrm{PAR}_L + (\mathrm{PAR}_U - \mathrm{PAR}_L) \times t/\mathrm{NI}$$
$$\mathrm{BW}(t) = \mathrm{BW}_L + (\mathrm{BW}_U - \mathrm{BW}_L) \times t/\mathrm{NI} \quad (18)$$

Step 8: Set $t=t+1$.
Step 9: Repeat steps 4 to 8 until $t=\mathrm{NI}$ is reached.
Step 10: Save the Best harmony vector X_{best} in the HM to represent the hidden layer parameters and its corresponding weight vector W obtained using ELM for testing of the model.

6. Experimental result analysis

In this study the performance of two single hidden layers feed forward neural networks i.e. RBF and CEFLANN has been compared with a new learning algorithm called SADHS-OELM. Again the performance of both the models has been compared with other learning algorithms like DE-OELM, self adaptive differential harmony search (SADHS), SGHS [18], HS, DE, ELM and back propagation (BP) with gradient descent approach for one day ahead prediction of both closing price and volatility for the same data set.

Simulations are carried on 5 different stock indices data set. Two of them i.e. BSE Sensex and CNX Nifty index belong to Indian stock market, Nikkei 225 belongs to Tokyo stock exchange, FTSE 100 belongs to London stock exchange and S&P500 belongs to US stock market. The total number of samples used in experiment for BSE, Nikkei, FTSE, and SP500 is 832 from 4th January 2010 to 30th April 2013 and for CNX are 754 from 4th January 2010 to 4th January 2013. The basic statistical characteristics of the closing price and daily return series of all the data set has been specified in Tables 1 and 2. The positive skewness value of the closing price implies that all the data are spread out more toward right. The kurtosis analysis implies that stock price data of Nikkei is more outlier prone where as all other data set are less outlier prone. Again from the Jarque–Bera test statistics, it has been observed that the closing price of all data set except FTSE100 are non-normal distributed. Similarly from the statistics of return series it has been observed that the return series of BSE, CNX and S&P500 are spread out more toward right whereas the return series of Nikkei and FTSE are spread more toward left. The kurtosis analysis of return series implies that all the data set are more outlier prone than the normal distribution. The Jarque–Bera test statistics further confirms that the daily returns is non-normal distributed. These preliminary analyses of the data encourage the adoption of more sophisticated model rather using traditional models.

In this paper the direct method has been used for prediction of both closing price and volatility. The sample set has been passed through a windowing process with a chosen window size and prediction horizon. To measure the generalization ability of the model initially data sets are divided into training and testing sets. The training part has been used to train the model, while the test part has been used to compute the predictions. Initially all the learning algorithms have been compared on the CEFLANN and RBF network by taking the BSE and CNX data set to perform a short term prediction. As in short term prediction few days or weeks of data are taken as input, so initially with a smaller input size both the network has been compared.

For predicting the one day a head closing price using CEFLANN and RBF, the number of input layer node has been set to 6 to express the closing index of 5 days ago and the simple moving average of it and the number of output node to 1 for expressing the closing index of 6th day. For predicting one day a head volatility using CEFLANN and RBF, the number of input layer node has been set to 6 to express the 2 days previous daily return, 3 days previous daily volatility and the simple moving average of 5 days previous daily volatility and the number of output node to 1 for expressing the daily volatility of 6th day. Before prediction of volatility, initially the daily stock return series were generated by taking the natural logarithm difference of the daily stock index closing price and the previous day's stock index closing price and multiplied by 100. Then the realized volatility on day t is calculated as

$$\sigma_t^2 = \frac{1}{n}\sum_{i=t-1}^{t-n}\left(r_i - \sum_{i=t-1}^{t-n} r_i/n\right)^2 \quad (19)$$

where $r_t = 100 \times (\ln P_t - \ln P_{t-1})$ denote the continuously compounded rate of stock returns from time $t-1$ to t, where P_t is the daily closing stock price at time t. n is the number of days before the nearest expiry option. The daily closing price, stock returns and daily volatility for BSE and CNX data sets are shown in Figs. 3 and 4. To improve the performance initially all the inputs

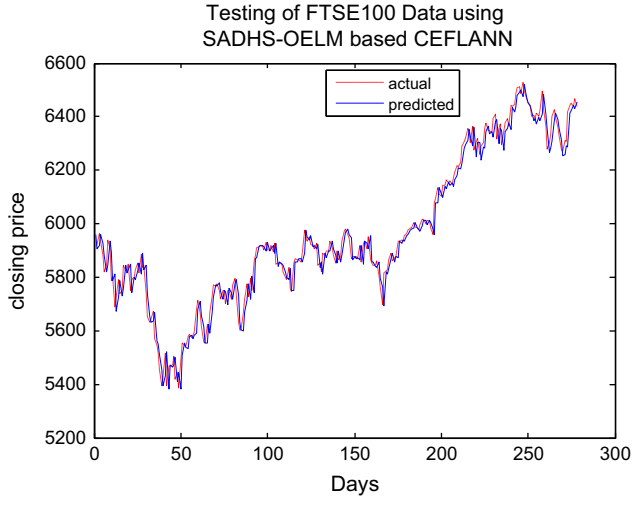

Fig. 38. One day ahead prediction of daily closing price of FTSE data using SADHS_OELM based CEFLANN.

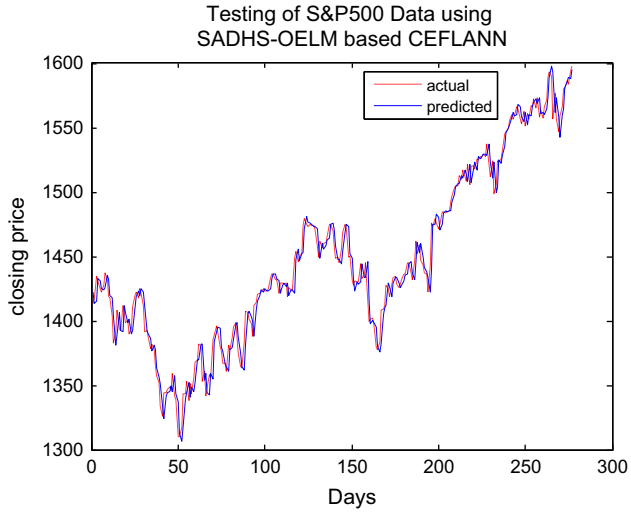

Fig. 39. One day ahead prediction of daily closing price of S&P500 data using SADHS_OELM based CEFLANN.

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

are scaled between 0 and 1 using the min max normalization as follows:

$$y = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{20}$$

where y is the normalized value; x is the value to be normalized; $x_{\min}$ is the minimum value of the series to be normalized; $x_{\max}$ is the maximum value of the series to be normalized.

To perform a number of simulations the order of the CEFLANN is set to 2 and the number of hidden layer nodes of RBF is set to 6. The size of parameter space need to be tuned using different learning algorithm for both the network is given in Table 3. For CEFLANN with order 2 and input size 6, the number of associated parameters used in functional expansion is 14 and number of weights between expanded pattern and output neuron is 8. Hence the total number of unknown parameters needed to be tuned by a learning algorithm is 22. But as ELM is used only to find the output weights, the parameter space size for ELM learning is 8. Again in SADHS-OELM and DE-OELM along with the 22 unknown parameters there is an unknown regularization factor value of ELM that needs to be tuned using the evolutionary algorithm, making the total parameter space size to 23. Out of these 23 parameters 15 are set by evolutionary algorithm and 8 by ELM. Similarly for RBF with 6 number of hidden layer nodes, input size 6, the number of center and deviation parameters used in the radial basis functions is 42 and number of weights between expanded pattern and output neuron is 7. Hence the total number of unknown parameters needed to be tuned by a learning algorithm is 49. But as ELM is used only to find the output weights, the parameter space size for training RBF through ELM is 7. Again in SADHS-OELM and DE-OELM along with the 49 unknown parameters and one unknown regularization factor value of ELM, the total parameter space size is 50. Out of these 50 parameters 43 are set by evolutionary algorithm and 7 by ELM. However, this parameter space size is greatly dependent on the input size and the order of functional expansion of CEFLANN or the number of radial basis functions used in RBF.

The controlling parameters of any evolutionary algorithm are normally application oriented. Thus no fixed value is assigned to these parameters. Therefore, we have derived the controlling parameters of the evolutionary algorithms initially through a number of simulations and then the same controlling parameters are set for training of the RBF and CEFLANN through each evolutionary learning cycle using DE-OELM and SADHS-ELM approaches. With a suitable population size, the position of each individual i.e. the unknown parameters of the model are initialized randomly according to the specified dimension of parameter space. The RMSE error is taken as the fitness function for all the above learning algorithms. The same population with maximum 100 iterations is used for all the evolutionary learning based methods. Table 4 specifies the controlling parameters used for different evolutionary based learning paradigms.

The Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) is used to compare the performance of the models for predicting the closing price of the BSE Sensex and CNX Nifty indices one day in advance with different learning algorithms. The RMSE and MAPE are defined as:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(y_k - \hat{y}_k)^2} \tag{21}$$

$$\text{MAPE} = \frac{1}{N}\sum_{k=1}^{N}\left|\frac{y_k - \hat{y}_k}{y_k}\right| \times 100 \tag{22}$$

where y_k is the actual closing price on kth day; $\hat{y}_k$ is the predicted closing price on kth day; N is the number of test data.

Comparison of volatility forecasts is conducted for one-step ahead horizon in terms of mean squared forecast error (MSFE), root mean squared forecast error (RMSFE) and mean absolute forecast error (MAFE) defined as follows:

$$\text{MSFE} = \frac{1}{N}\sum_{i=1}^{N}(\sigma_i^2 - \hat{\sigma_i^2})^2 \tag{23}$$

$$\text{RMSFE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\sigma_i^2 - \hat{\sigma_i^2})^2} \tag{24}$$

$$\text{MAFE} = \frac{1}{N}\sum_{i=1}^{N}|\sigma_i^2 - \hat{\sigma_i^2}| \tag{25}$$

where σ_i^2 is the realized volatility and $\hat{\sigma_i^2}$ is the predicted volatility.

Originally the dataset has been divided in to a single train and test set. So performance metrics has been obtained by evaluating 10 different executions of the algorithm on the same training and testing sets. The best, average and standard deviations of these 10 runs are reported for all the data sets. As the parameter space size for evolutionary learning techniques varies from 20 to 50, two different population sizes i.e. 20 and 50 are considered for experimental study. Tables 5 and 6 list the various forecast statistics of CEFLANN and RBF models trained using different learning algorithms with a population size of 20 for one day ahead prediction of BSE Sensex and CNX Nifty stock indices. In a similar way Tables 7 and 8 depict one day ahead different volatility forecast statistics of CEFLANN and RBF models using the same stock indices. The one day ahead closing price prediction and volatility forecasting of BSE and CNX data set using CEFLANN and RBF models trained using different learning algorithms are presented in Figs. 5–36. Wilcoxon Signed Rank test has been performed to show whether the difference between the performance metrics of different learning algorithms are statistically meaningful or not and the significance level in the test is taken as 0.05. The p and h values of the statistics are specified in Table 9. The h value 0 indicates that it is not statistically possible to prove that the outputs of two compared algorithms are different from each other. Again the entire experiment is done with a population size of 50, and the corresponding stock price and volatility prediction outputs are presented in Tables 10–12. With the two different population sizes both the networks trained using SADHS_OELM algorithm are providing closer results for both stock price and volatility prediction, but the experimental time is increased

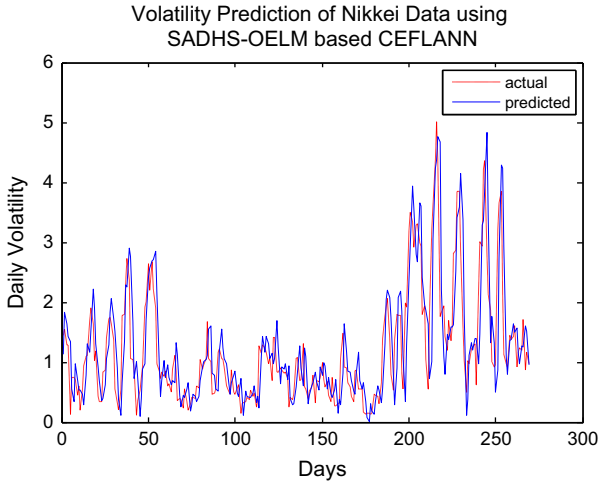

Fig. 40. One day ahead prediction of daily volatility of Nikkei data using SADHS_OELM based CEFLANN.

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

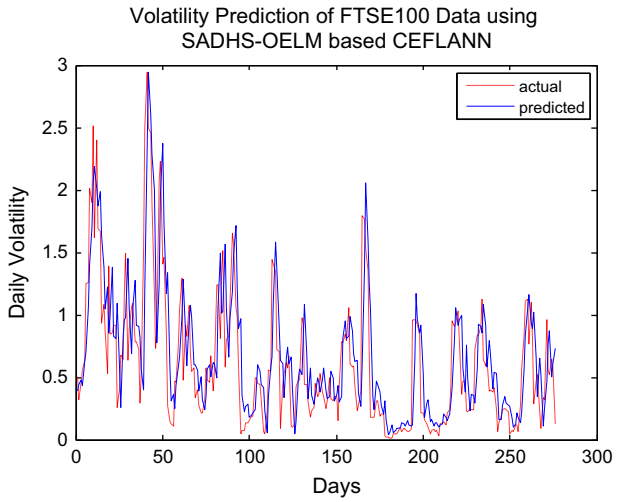

Fig. 41. One day ahead prediction of daily volatility of FTSE data using SADHS_OELM based CEFLANN.

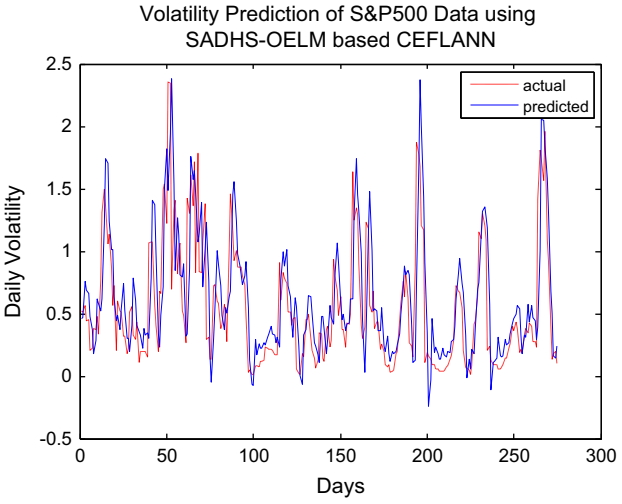

Fig. 42. One day ahead prediction of daily volatility of S&P500 data using SADHS_OELM based CEFLANN.

slightly with the increased population size. Further the performance of CEFLANN trained using SADHS-OELM is tested with different input size and prediction horizon as shown in Tables 13 and 14, respectively.

Performance comparison of SADHS-OELM based RBF and SADHS-OELM based CEFLANN for one day ahead stock price and volatility prediction of three international stock indices i.e. Nikkei225, FTSE100 and S&P500 are listed in Tables 15 and 16, respectively. The one day ahead closing price and volatility prediction of these three international stocks using SADHS-OELM based CEFLANN are exhibited in Figs. 37–42.

Summarizing the results, the following inference is drawn:

- A comparison of different learning approaches over CEFLANN and RBF models shows clearly the superior forecasting performance of SADHS-OELM approach in comparison to other learning algorithms.
- The global search capability of harmony search is improved by including the mutation operation in pitch adjustment of each harmony.
- The generalization ability of ELM is further improved by using the self adaptive differential harmony search technique in optimizing the parameters of hidden layer nodes of a SLFNN.
- Unlike BP, the stochastic learning algorithms do not get trapped in local minima. Hence the degree of prediction accuracy of the models trained using stochastic learning and hybrid stochastic learning is more than that of BP.
- Again from the experimental result analysis it is clearly apparent that the CEFLANN provides superior forecasting performance when trained with SADHS_OELM algorithm in comparison to RBF in forecasting both stock closing price and volatility.

7. Conclusion

This paper proposes a hybrid learning framework called Self Adaptive Differential Harmony Search Based Optimized Extreme Learning Machine (SADHS-OELM) for two single hidden layer feed forward neural networks i.e. RBFNN and CEFLANN. The new learning framework is based on ELM, where the output weights are obtained using a robust least squares solution. However, to optimize the fitting performance, the selection of the weights of connections between the input layer and the hidden layer, the bias of neurons of the hidden-layer, and the regularization factor of robust least squares, a global optimization technique called SADHS is used in this paper. SADHS is a variant of harmony search technique that uses the current to best mutation scheme of DE in the pitch adjustment operation for harmony improvisation process. Computational experiments illustrating the effectiveness of the RBF and CEFLANN models learned using SADHS-OELM are provided by forecasting the one day ahead closing price and volatility of a few benchmark stock indices in comparison with some other learning schemes like DE-OELM, DE, SADHS, SGHS, HS, ELM and BP, etc. The computational results demonstrate that the CEFLANN model trained using SADHS-OELM offers significant improvements in forecasting performance compared to all other learning schemes including the RBF model.

Future research may include applications of the proposed model to handle different issues of financial time series forecasting, like financial time series classification, trading, portfolio management etc. Another area of research will focus on systematic approaches for selecting optimal network size for both RBF and FLANN models. Finally the research can be extended to incorporate the advantage of fuzzy logic to handle uncertainties and outliers in the financial data.

References

[1] F. A. Oliveira, C. N. Nobre, The use of artificial neural networks in the analysis and prediction of stock prices, in: IEEE International Conference on Systems, Man and Cybernetics (SMC), 2011, pp. 2151–2155.
[2] M. P. Naeini, H. Taremian, H. B. Hashemi, Stock market value prediction using neural networks, in: International Conference on Computer Information Systems and Industrial Management Applications (CISIM), 2010, pp. 132–136.
[3] B. Kozarzewski, A neural network based time series forecasting system, in: Third International Conference on Human System Interaction, 2010, pp. 59–62.
[4] L. Q.Yu, F. ShaoRong, Stock market forecasting research based on neural network and pattern matching, in: International Conference on E-Business and E-Government, 2010, pp. 1940–1943.
[5] H. Tahersima, M. H. Tahersima, M. Fesharaki, Forecasting stock exchange movements using neural networks: a case study, in: International Conference on Future Computer Sciences and Application, 2011, pp. 123–126.
[6] E. Hajizadeh, A. Seifi, M.H. Fazel Zarandi, I.B. Turksen, A hybrid modeling approach for forecasting the volatility of S&P 500 index return, Expert Syst. Appl. 39 (2012) 431–436.
[7] T.H. Roh, Forecasting the volatility of stock price index, Expert Syst. Appl. 33 (2007) 916–922.

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003

[8] C.P. Wanga, S.H. Lin, H.H. Huang, P.C. Wu, Using neural network for forecasting TXO price under different volatility models, Expert Syst. Appl. 39 (2012) 5025–5032.
[9] L. Xia, H. Muzhoub, M.H. Leec, J. Lic, D. Weic, H. Haic, Y. Wu, A new constructive neural network method for noise processing and its application on stock market prediction, Appl. Soft Comput. 15 (2014) 57–66.
[10] W. Shen, X.G. C. Wub, D. Wu, Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm, Knowl. Based Syst. 24 (2011) 378–385.
[11] H. Feng, H. Chou, Evolutional RBFNs prediction systems generation in the applications of financial time series data, Expert Syst. Appl. 38 (2011) 8285–8292.
[12] J. C. Patra, C. N. Thanh, P. K. Meher, Computationally efficient FLANN-based intelligent stock price prediction system, in: Proceedings of International Joint Conference on Neural Networks, 2009, pp. 2431–2438.
[13] B. Majhi, S. Hasan, F. Mowafak, FLANN based forecasting of S&P 500 index, Inf. Technol. J. 4 (3) (2005) 289–292.
[14] H. H. Ning, Short term forecasting of stock price based on genetic neural network, in: Sixth International Conference on Natural Computation, 2010, pp. 1838–1841.
[15] M. E. Abdual-Salam, H. M. Abdul-Kader, W. F. Abdel-Wahed, Comparative study between differential evolution and particle swarm optimization algorithms in training of feed-forward neural network for stock price prediction, in: Seventh International Conference on Informatics and Systems(INFOS), 2010 pp. 1–8.
[16] S. Oh, W. Kim, W. Pedrycz, S. Joo, Design of *K*-means clustering-based polynomial radial basis function neural networks (PRBF NNs) realized with the aid of particle swarm optimization and differential evolution, Neuro Comput. 78 (2012) 121–132.
[17] S. Chakravarty, P.K. Dash, A PSO based integrated functional link net and interval type-2 fuzzy logic system for predicting stock market indices, Appl. Soft Comput. 12 (2012) 931–941.
[18] S. Kulluk, L. Ozbakir, A. Baykasoglu, Training neural networks with harmony search algorithms for classification problems, Eng. Appl. Artif. Intell. 25 (2012) 11–19.
[19] B. Lu, A Stock Prediction method based on PSO and BP hybrid algorithm, in: International Conference on E-Business and E-Government, 2011, pp. 1–4.
[20] S. Dehuri, B. Cho Sung, A comprehensive survey on functional link neural networks and an adaptive PSO-BP learning for CFLNN, Neural Comput. Appl. 19 (2) (2010) 187–205.
[21] G. Huang, Q. Zhu, C. Siew, Extreme learning machine: theory and applications, Neuro Comput. 70 (2006) 489–501.
[22] G.B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multiclass classification, IEEE Trans. Syst. Man Cybern. Part B Cybern 42 (2) (2012) 513–529.
[23] A.B. Crespo, P.J. G. Laencina, J.L. S. Gomezc, Neural architecture design based on extreme learning machine, Neural Netw. 48 (2013) 19–24.
[24] K. Javed, R. Gouriveau, N. Zerhouni, SW-ELM: a summation wavelet extreme learning machine algorithm with a priori parameter initialization, Neurocomputing 123 (2014) 299–307.
[25] S.O. Olatunji, A. Selamat, A. Abdulraheem, A hybrid model through the fusion of type-2 fuzzy logic systems and extreme learning machines for modeling permeability prediction, Inf. Fusion 16 (2014) 29–45.
[26] M. Luo, K. Zhang, A hybrid approach combining extreme learning machine and sparse representation for image classification, Eng. Appl. Artif. Intell. 27 (2014) 228–235.
[27] H. Yu, T. Xie, S. Paszczyñski, M.W. Bogdan, Advantages of radial basis function networks for dynamic system design, IEEE Trans. Ind. Electron. 58 (12) (2011) 5438–5450.
[28] E.A. El-Sebakhy, O. Asparouhov, A.A. Abdulraheem, A.A. AlMajed, D. Wu, K. Latinski, I. Raharja, Functional networks as a new data mining predictive paradigm to predict permeability in a carbonate reservoir, Expert Syst. Appl. 39 (2012) 10359–10375.
[29] R. Bisoi, P.K. Dash, V. Padhee, M.H. Naeem, Mining electricity prices in energy markets using a computationally efficient neural network, in: IEEE International Conference on Energy, Automation and Signal (ICEAS), 2011, pp. 1–5.
[30] K.S. Lee, Z.W. Geem, A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice, Comput. Methods Appl. Mech. Eng. 194 (2005) 3902–3933.
[31] R. Diao, Q. Shen, Feature selection with harmony search, IEEE Trans. Syst. Man Cybern. Part B Cybern. 42 (6) (2012) 1509–1523.
[32] B. Wu, C. Qian, W. Ni, S. Fan, Hybrid harmony search and artificial bee colony algorithm for global optimization problems, Comput. Math. Appl. 64 (2012) 2621–2634.
[33] Y. Yuan, H. Xu, J. Yang, A hybrid harmony search algorithm for the flexible job shop scheduling problem, Appl. Soft Comput. 13 (2013) 3259–3272.
[34] N. Poursalehi, A. Zolfaghari, A. Minuchehr, Differential harmony search algorithm to optimize PWRs loading pattern, Nucl. Eng. Des. 257 (2013) 161–174.
[35] S. Lee, S. Mun, Improving a model for the dynamic modulus of asphalt using the modified harmony search algorithm, Expert Syst. Appl. 41 (2014) 3856–3860.
[36] M.Y. Chen, B.T. Chen, Online fuzzy time series analysis based on entropy discretization and a fast Fourier transform, Appl. Soft Comput. 14 (2014) 156–166.
[37] M.Y. Chen, A high-order fuzzy time series forecasting model for internet stock trading, Future Gener. Comput. Syst. 37 (2014) 461–467.

Please cite this article as: R. Dash, et al., A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm and Evolutionary Computation (2014), http://dx.doi.org/10.1016/j.swevo.2014.07.003