Modified Neuron-Synapse level problem decomposition method for Cooperative Coevolution of Feedforward Networks for Time Series Prediction

*

Ravneil Nand
FSTE
The University of the South Pacific
Suva, Fiji
ravneiln@yahoo.com

Bibhya Nand Sharma
SCIMS, FSTE
The University of the South Pacific
Suva, Fiji
bibhya.sharma@usp.ac.fj

***Abstract*—Complex problems have been solved efficiently through decomposition of a particular problem using problem decompositions. Even combination of different distinct problem decomposition methods has shown good results in time series prediction. The application of different problem decomposition methods at different stages of a network can share its strengths to solve the problem in hand better. Hybrid versions of two distinct problem decomposition methods has showed promising results in past. In this paper, a modified version of latterly introduced Neuron-Synapse level problem decomposition is proposed using feedforward neural networks for time series prediction. The results shows that the proposed modified model has got better results in more datasets when compared to its previous version. The results are better in some cases for proposed method in comparison to several other methods from the literature.**

***Index Terms*—cooperative coevolution, problem decomposition, feedforward network**

I. Introduction

Cooperation and hybridization of algorithms are becoming a popular practice to solve computationally difficult problems. It allows to reduce the difficulty level of problem or takes advantage of different algorithms to solve a problem at hand. Cooperative coevolution (CC) is one of the techniques inspired from nature which solves by disintegrating large problems into smaller subcomponents and then solves through cooperation of individual subcomponents [1], [2].

In CC while cooperating, individuals get rewarded based on their ability to cooperate better where a number of species are evolved together to achieve best solutions [3]. Applications of CC have shown inspiring results in neural networks [2], [4]. Neural networks allow to train datasets based on weights which is later used for testing the dataset.

One of the issues reported against CC in [5] is its sensitivity towards problem decomposition. According to authors of [6], the performance of cooperative coevolutionary architectures highly depend on the decomposition strategy used. Therefore, an ideal decomposition strategy needs to be identified through extensive experimentation on different time series datasets.

Lately, various problem decomposition techniques have been used in the field of cooperative neuro-evolution; two common ones are the neuron and synapse level decomposition [5] for time series prediction. The work in [2] combines the neuron and synapse decomposition strategy to form a hybrid problem decomposition method called Neuron-Synapse level (NSL) which produced better results in selected datasets [2].

In this paper, NSL is modified to have better performance in terms of training and testing on different datasets. The proposed method is called the modified Neuron-Synapse Level decomposition (mNSL) which decomposes the problem in terms of incoming and outgoing weights of an individual hidden neuron instead of different layers as seen in NSL. The mNSL is used with feedforward network for time series prediction to treat all the datasets.

The rest of the paper is arranged as follows. In Section 2, the proposed method is discussed in detail while Section 3 gives the experimental setup. In Section 4 and 5, the results and discussion are given respectively while section 6 concludes the paper with a brief discussion on the direction of future work.

II. Modified Neuron-Synapse Level decomposition

Using a combination of problem decompositions in time series prediction at different level of a network, better results can be achieved from each problem decomposition for the training and testing processes.

In this paper, a modified version of Neuron-Synapse Level decomposition (NSL) is proposed where two problem decomposition algorithms are used to solve time series problems as shown in Fig.1. It is inspiring to see results based on hybrid model where not only one but two problem decomposition can be used to enhance the performance of the model. In this research, the decomposition of a problem is not done in terms of layers as done in NSL, instead two layers are looked at one

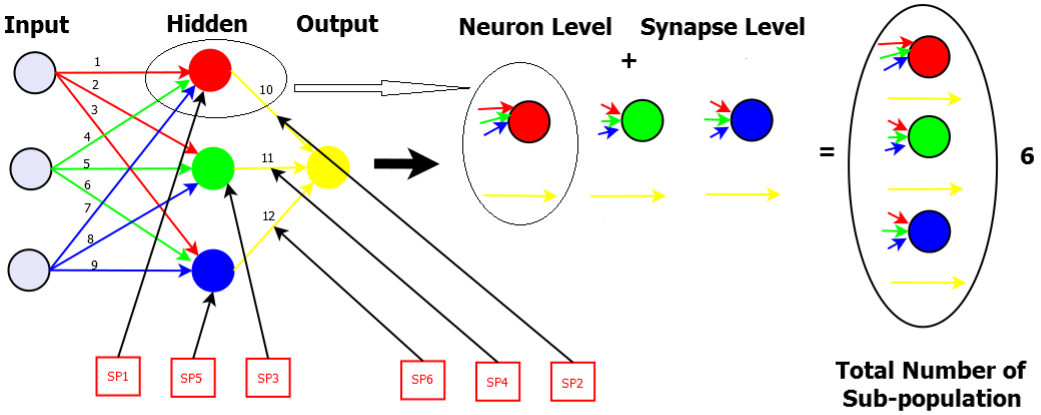

Fig. 1. modified Neuron-Synapse Level decomposition (mNSL) breaks down the neural network into separate subpopulation based on the problem decomposition method applied at a particular instance.

after the other and the process is repeated. The hybrid model will be called modified Neuron-Synapse Level decomposition (mNSL) from now on wards in this paper.

Algorithm 1 shows the mNSL which is used to train the network using both the decomposition methods at different instances and later final weights are used to test the dataset. With this approach, the best results are obtained using both of the problem decomposition working interchangeably.

In Step 1, the problem is broken down in the number of subcomponents based on the decomposition technique (mNSL) used. Firstly, incoming weights of first hidden neuron are selected as first subpopulation. Later outgoing weights are selected and then the steps are repeated for each hidden neurons. In Step 2, the encoding of problem takes place based on the number of neurons in the hidden layer and weights in hidden and output layers. A unique encoding scheme of mNSL makes the calculation or encoding of new populations order to change. The approach of finding best weights is entirely based on the decomposition technique used.

The first subcomponent formed is based on the first hidden neuron and the number of subpopulation it will have will depend on the number of incoming weights from input layer and the second subcomponent is the outgoing weights from the first hidden neuron to output layer and number of subpopulation will be one. The other subcomponents and subpopulations follow the same procedure and continue until the last hidden layer neuron. The entire subcomponent is based on number of hidden neurons and weight links in hidden and output layers as in earlier version but the order of subcomponents changes in mNSL.

Once the network has been encoded, the algorithm moves to Step 3, where firstly initialization of species takes place. After initialization, cooperative evaluation of each sub-population takes place. In evaluation process, the first population individuals for each sub-population (population vector) are selected as best for first instance and later get replaced through evolution.

The update is in terms of each hidden neuron's incoming and outgoing connections instead of layers. This allows to see the importance of updating one hidden neurons incoming and outgoing links before proceeding to other hidden neurons weight links.

Fitness for the best individuals are calculated using forward fitness pass, where, predicted output is compared with actual output and sum squared error measured. The best fitness is the individual with the minimum value for sum squared error.

Later the steps are repeated for all population for each sub-

Algorithm 1: modified Neuron-Synapse Level decomposition for training Feedforward Neural Networks

Step 1: Decompose the problem into subcomponents according to mNSL. Here problem is decomposed into input-hidden, and hidden-output layer subcomponents based on hidden neuron.
Step 2: Encode each subcomponent in a sub-population according to problem decomposition method used.
Step 3: Initialize and cooperatively evaluate each sub-population.
foreach *Cycle until termination* **do**
  foreach *Sub-population* **do**
    foreach *Depth of n Generations* **do**
      Select and create new offspring using genetic operators
      Cooperative Evaluation the new offspring
      Add new offspring's to the sub-population
    end
  end
end

population one by one until the best fitness for each sub-population is found. The fitness of a particular member of a particular species is computed by estimating how well it cooperates with other subspecies to produce good solutions. Best individuals are concatenated as one individual population.

After evaluation of the populations, evolution takes place using genetic operators for each sub-population. Using the generalized generation gap parent centric crossover (G3-PCX) genetic algorithm [7], two offsprings/kids are created based on centroid and orthogonal directions of selected parents. Later the new offspring population individuals are evaluated as done earlier for the rest of the population to determine their fitness.

After fitness is calculated, it is then tested with two random parents from the population set. Here the best fitness individuals replace the family of parents. Again all the sub-population is evolved separately, one after the other. This allows the populations to be evolved generation after generation to get better results.

All the sub-populations are evolved for a fixed number of generations. Once the network has been evolved according to the maximum fitness evaluations specified, the generalization performance is tested with training and testing dataset.

III. Experimental Setup

The experimental setup is the same as the original method NSL. A brief overview is given in this section.

The cooperative coevolution algorithm utilized in the proposed method is from Smart Bilo Computational Intelligence Framework [8] and it has been modified to suit our needs. Feedforward neural network (FNN) has been used to predict time series dataset. In FNN the connections moves forward in only one direction, there is no cycles or loops. Taken's embedding theorem [9] has been utilized to reconstruct the datasets. It allows reconstructions of the dataset into a state space vector using two important conditions known as *time delay (T)* and *embedding dimension (D)* [9]. It is crucial that the value of *T* and *D* be wisely selected by the vector in order to reproduce important characteristics of the original dataset [10].

Five datasets are used to train and test the proposed method mNSL. Three are benchmark datasets while the remaining two are financial datasets. The datasets are Mackey-Glass [11], Lorenz [12], Sunspot [13], ACI Worldwide Inc. [14] and Seagate [14].

The *Mackey-Glass time series* is the first benchmark dataset that is used [11]. The first 500 values are used for training while the remaining 500 values are used for testing the network. The embedding dimension is set to $D = 3$ and $T = 2$ for the reconstructing of the phase space. The range of [0,1] is used for normalizing this time series data.

Second dataset used is *Lorenz time series* [12]. The range of [-1,1] is used while the first 500 values are used for training whereas the remaining are used for testing. As in Mackay-Glass, the embedding dimension is set to $D = 3$ and $T = 2$ for reconstructing the phase space for this dataset.

The third data set that is trained and tested is *Sunspot time series* [13]. It is a real world dataset [15]. The range of [-1,1] is used while the first 500 values are used for training whereas the remaining 500 values are used for testing. The embedding dimensions is kept at $D = 5$ and $T = 2$.

The *ACI Worldwide Inc. time series* is the fourth data set that is trained and tested [14]. The range of [0,1] is used while the first 400 values are used for training whereas remaining 400 values are used for testing. As in Sunspot dataset, the embedding dimensions of $D = 5$ and $T = 2$ is used.

The last dataset used is Seagate Technology PLC [14]. To train and test the neural network, the first 400 values are used for training while the remaining 400 values are used for testing. The range of [0,1] is used in normalizing this dataset. The embedding dimensions used are $D = 5$ and $T = 2$.

The feedforward neural network employs Sigmoid units for the Mackey-Glass, Seagate, and ACI Worldwide Inc. time series while the hyperbolic tangent unit is utilized for the Lorenz and Sunspot time series. As done in [16], root mean squared error (RMSE) is used in evaluating the performance of the proposed method.

The maximum number of function evaluations was set at 50,000, which is the termination condition. The algorithm is individually run 50 times. As done in literature [16], a pool size of 2 parents and offsprings are put in the G3-PCX algorithm [7] for the purpose of the evolution .

The population size is set at 300 as done in original version. The number of generations for each sub-population was kept at 1 as done in literature. The prediction performance of the proposed method mNSL has been computed by Root Mean Squared Error (RMSE) and Normalized Mean Squared Error (NMSE).

IV. Results

The experimental results of proposed method (mNSL) based on its performance are given in this section.

The Tables I - V exhibits the results for different number of hidden neurons on the proposed method mNSL in comparison to two standalone (NL and SL) methods. The results shown in the Tables I - V are established on 95 percent confidence interval on RMSE and the best results for each method are displayed in bold. The *Training* and *Generalization* shows the train average with train error sum and test average with test error sum respectively. The *Best* value shows the best test RMSE.

Table I illustrates the evaluation of the Mackey-Glass time series problem. It was observed that mNSL has similar performance as FNN-NL method. It obtained the second best value compared with the other two methods. The proposed method mNSL showed better generalization performance and best training value with five hidden neurons.

In Table II, the results of the Lorenz time series problem are given. It was seen that the mNSL has poor performance in terms of training and generalization than FNN-NL method. The best value was better than both of the methods. It was observed that the generalization performance and training of

TABLE I
THE PREDICTION TRAINING AND GENERALIZATION PERFORMANCE (RMSE) OF NL, SL AND MNSL FOR THE MACKEY-GLASS TIME SERIES

Prob.	H	Training	Generalization	Best
FNN-NL	3	0.0107 ± 0.00131	0.0107 ± 0.00131	0.005
	5	0.0089 ± 0.00097	0.0088 ± 0.00097	**0.0038**
	7	**0.0078 ± 0.00079**	**0.0078 ± 0.00079**	0.0040
FNN-SL	3	0.0237 ± 0.0023	0.0237 ± 0.0023	0.0125
	5	0.0195 ± 0.0012	0.0195 ± 0.0012	0.0124
	7	**0.0177 ± 0.0009**	**0.0178 ± 0.0009**	**0.0121**
FNN-mNSL	3	0.0104 ± 0.0009	0.01034 ± 0.00099	0.0063
	5	**0.0089 ± 0.00084**	**0.0088 ± 0.00084**	**0.0045**
	7	0.0096 ± 0.00086	0.00956 ± 0.00086	0.0056

TABLE II
THE PREDICTION TRAINING AND GENERALIZATION PERFORMANCE (RMSE) OF NL, SL AND MNSL FOR THE LORENZ TIME SERIES

Prob.	H	Training	Generalization	Best
FNN-NL	3	**0.0170 ± 0.0031**	**0.0176 ± 0.0031**	0.0043
	5	0.0249 ± 0.0062	0.0271 ± 0.0067	**0.0021**
	7	0.0379 ± 0.0093	0.0416 ± 0.0092	0.0024
FNN-SL	3	0.0680 ± 0.0325	**0.0452** ± 0.0229	0.0153
	5	**0.0526** ± 0.0084	0.0546 ± 0.0084	0.0082
	7	0.0574 ± **0.0075**	0.0605 ± **0.0074**	**0.0079**
FNN-mNSL	3	**0.0285 ± 0.00610**	**0.0292 ± 0.0061**	0.0025
	5	0.0399 ± 0.00837	0.0429 ± 0.00833	**0.0018**
	7	0.0448 ± 0.00831	0.0484 ± 0.00826	0.0046

the mNSL decreases as the number of the hidden neuron increases. The best result for mNSL was given by three hidden neurons.

The Sunspot time series problem is evaluated in Table III. This is a real time series dataset where noise is present. The proposed method mNSL was unable to outperform FNN-NL but had similar results. The proposed method had similar best value as FNN-NL method. The best result for mNSL was given by three and five hidden neurons.

In Table IV, the ACI Worldwide Inc. time series problem results are given. The time series is real time series based on stock exchange. The proposed method mNSL had similar performance as other methods in terms of training and generalization. The best value was better than both the methods. Three hidden neurons have given the best result for the proposed method.

TABLE III
THE PREDICTION TRAINING AND GENERALIZATION PERFORMANCE (RMSE) OF NL, SL AND MNSL FOR THE SUNSPOT TIME SERIES

Prob.	H	Training	Generalization	Best
FNN-NL	3	**0.0207 ± 0.0035**	**0.0538** ± 0.0091	**0.015**
	5	0.0289 ± 0.0039	0.0645 ± 0.0093	0.017
	7	0.0353 ± 0.0048	0.0676 ± **0.0086**	0.021
FNN-SL	3	**0.5391** ± 0.0261	**0.4998** ± 0.0238	**0.210**
	5	0.5601 ± 0.0208	0.5210 ± 0.0177	0.302
	7	0.5682 ± **0.0178**	0.5250 ± **0.0132**	0.344
FNN-mNSL	3	**0.0301** ± 0.0160	0.0714 ± 0.0112	0.020
	5	0.0324 ± **0.0144**	**0.0692** ± 0.0102	**0.016**
	7	0.0338 ± 0.0171	0.0869 ± **0.0088**	0.019

TABLE IV
THE PREDICTION TRAINING AND GENERALIZATION PERFORMANCE (RMSE) OF NL, SL AND MNSL FOR THE ACI WORLDWIDE INC. TIME SERIES

Prob.	H	Training	Generalization	Best
FNN-NL	3	0.0214 ± 0.00039	0.0215 ± 0.00039	0.020
	5	0.0203 ± 0.00047	0.0212 ± 0.00041	0.019
	7	**0.0201 ± 0.00038**	**0.0208 ± 0.00033**	**0.019**
FNN-SL	3	0.4666 ± 0.0399	0.4112 ± 0.0362	0.080
	5	**0.4135** ± 0.0388	**0.3902** ± 0.0378	**0.042**
	7	0.4491 ± **0.0279**	0.4244 ± **0.0270**	0.134
FNN-mNSL	3	**0.0208** ± 0.00130	**0.0176** ± 0.00095	**0.015**
	5	0.0209 ± **0.00103**	0.0178 ± **0.00058**	0.016
	7	0.0211 ± 0.00127	0.0184 ± 0.0010	0.015

TABLE V
THE PREDICTION TRAINING AND GENERALIZATION PERFORMANCE (RMSE) OF NL, SL AND MNSL FOR THE SEAGATE TIME SERIES

Prob.	H	Training	Generalization	Best
FNN-NL	3	**0.02530** ± 0.05582	**0.1809 ± 0.03548**	0.032
	5	0.02313 ± 0.07403	0.2261 ± 0.04811	**0.031**
	7	0.02189 ± **0.08061**	0.2408 ± 0.05306	0.053
FNN-SL	3	0.4129 ± 0.03401	**0.3492 ± 0.02977**	0.089
	5	0.3820 ± **0.03381**	0.3727 ± 0.03371	**0.088**
	7	**0.3816** ± 0.04425	0.4183 ± 0.04306	0.094
FNN-mNSL	3	0.01897 ± **0.05672**	**0.1773 ± 0.03594**	0.026
	5	0.01856 ± 0.08636	0.2629 ± 0.05358	**0.025**
	7	**0.01829** ± 0.1022	0.3336 ± 0.0529	0.034

TABLE VI
A COMPARISON OF TRAINING AND GENERALIZATION BETWEEN NSL AND MNSL ON DIFFERENT TIME SERIES DATASETS BASED ON BEST HIDDEN NEURON

Problem	Method	Training	Generalization
Mackey Glass	FNN-NSL [2]	0.0100 ± 0.00055	0.0100 ± 0.00055
	Proposed mNSL	0.0089 ± 0.00084	0.0088 ± 0.00084
Lorenz	FNN-NSL [2]	0.0350 ± 0.00914	0.0357 ± 0.0093
	Proposed mNSL	0.0285 ± 0.00610	0.0292 ± 0.0061
Sunspot	FNN-NSL [2]	0.0356 ± 0.0074	0.0842 ± 0.0098
	Proposed mNSL	0.0324 ± 0.0144	0.0692 ± 0.0102
ACI Worldwide	FNN-NSL [2]	0.0209 ± 0.00036	0.0192 ± 0.0010
	Proposed mNSL	0.0208 ± 0.00130	0.0176 ± 0.0010
Seagate	FNN-NSL [2]	0.0186 ± 0.00029	0.2562 ± 0.04221
	Proposed mNSL	0.0183 ± 0.10220	0.1773 ± 0.03594

Table V illustrates the evaluation of the Seagate time series problem. For this time series, the mNSL method outperformed the other two methods, NL, and SL. The proposed method mNSL had similar performance as other methods in terms of training and generalization. For mNSL, five hidden neurons have given the best results.

In Table VI, the proposed method mNSL is compared with original method NSL in terms of training and generalization performance. The table clearly shows that the mNSL has got

TABLE VII
A COMPARISON OF THE RESULTS FROM LITERATURE ON DIFFERENT TIME SERIES DATASETS

Problem	Prediction Method	RMSE	NMSE
Mackey Glass	AMCC-RNN [17]	7.53E-03	3.90E-04
	Locally linear neuro-fuzzy model (2006) [18]	9.61E-04	
	SL-CCRNN [16]	6.33E-03	2.79E-04
	NL-CCRNN [16]	8.28E-03	4.77E-04
	CICC-RNN [19]	3.99E-03	1.11E-04
	Neuron-Synapse Level method FNN-NSL [2]	4.86E-03	4.48E-05
	Proposed FNN-mNSL	4.47E-03	3.78E-05
Lorenz	RBF with orthogonal least squares (2006) [18]		1.41E-09
	Locally linear neuro-fuzzy model (2006) [18]		9.80E-10
	SL-CCRNN [16]	6.36E-03	7.72E-04
	NL-CCRNN [16]	8.20E-03	1.28E-03
	CICC-RNN [19]	3.55E-03	2.41E-04
	Neuron-Synapse Level method FNN-NSL [2]	2.34E-03	2.87E-05
	Proposed FNN-mNSL	1.79E-03	1.69E-05
Sunspot	RBF with orthogonal least squares (2006) [18]		4.60E-02
	Locally linear neuro-fuzzy model (2006) [18]		3.20E-02
	SL-CCRNN [16]	1.66E-02	1.47E-03
	NL-CCRNN [16]	2.60E-02	3.62E-03
	CICC-RNN [19]	1.57E-02	1.31E-03
	Neuron-Synapse Level method FNN-NSL [2]	1.33E-02	5.38E-04
	Proposed FNN-mNSL	1.62E-02	7.91E-04
ACI Worldwide	FNN-SL [20]	1.92E-02	
	FNN-NL [20]	1.91E-02	
	MO-CCFNN-T=2 [21]	1.94E-02	
	MO-CCFNN-T=3 [21]	1.47E-02	
	Neuron-Synapse Level method FNN-NSL [2]	1.51E-02	1.24E-03
	Proposed FNN-mNSL	1.49E-02	1.22E-03
Seagate	FNN-SL [20]	3.74E-02	
	FNN-NL [20]	2.24E-02	
	Neuron-Synapse Level methodFNN-NSL [2]	2.45E-02	3.56E-03
	Proposed FNN-mNSL	2.50E-02	3.68E-03

better training and generalization performance in all the cases when compared to NSL.

The best results from Table I - V with some of the related methods in literature are given in Table VII. The RMSE best run together with NMSE are given for comparison purposes. The proposed mNSL method has shown good performance in nearly all the datasets when compared to other methods in the literature.

The best result of the Mackey-Glass time series problem is compared to works in literature in Table VII under problem Mackey-Glass. The proposed modified hybrid method was unable to beat CICC-RNN in terms of RMSE. mNSL outperformed its earlier version NSL in terms of RMSE and NMSE.

In Table VII under problem Lorenz, it shows the best result of Lorenz time series problem being compared to works in the literature. It has been seen that mNSL outperformed all the methods expect for CICC-RNN in terms of RMSE. The proposed method outperformed its earlier version NSL in terms of RMSE and NMSE.

The best result of the Sunspot time series problem is compared to works in literature in Table VII under problem Sunspot. The proposed method mNSL has outperformed the rest of the methods expect for CICC-RNN and NSL. The method was unable to outperform its earlier version in terms of RMSE and NMSE.

In Table VII, the best result of the ACI Worldwide Inc. time series problem is compared with results in the literature under problem ACI Worldwide Inc.. The proposed method has outperformed all the methods expect for MO-CCFNN-T=3 method. Good and stable performance has been achieved by the mNSL in terms of NSL.

The best result of the Seagate time series problem is compared to works in literature in Table VII under problem Seagate. The proposed method has outperformed all the methods expect for FNN-NL in terms of RMSE and FNN-NSL method in terms of both RMSE and NMSE. The method was unable to outperform its earlier version in terms of RMSE and NMSE.

Figures (2- 3) show predicted results of FNN-mNSL method with original results on Sunspot and ACI Worldwide Inc. dataset. The error graph is also given which indicates that better methods will be needed to cater for the chaotic nature of the time series problems at certain time intervals.

V. DISCUSSION

The results obtained for the proposed method is competitive when compared to works from literature involving five different data sets. The application of different decomposition

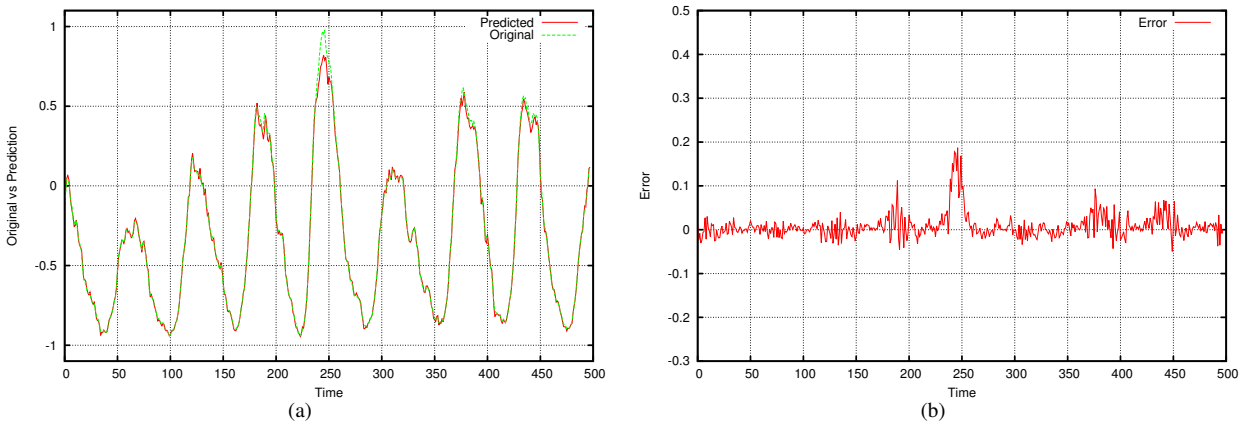

Fig. 2. Random picked dataset showing prediction by mNSL for Sunspot dataset. (a) Performance given by mNSL on the testing set. (b) Error on the test dataset given by mNSL.

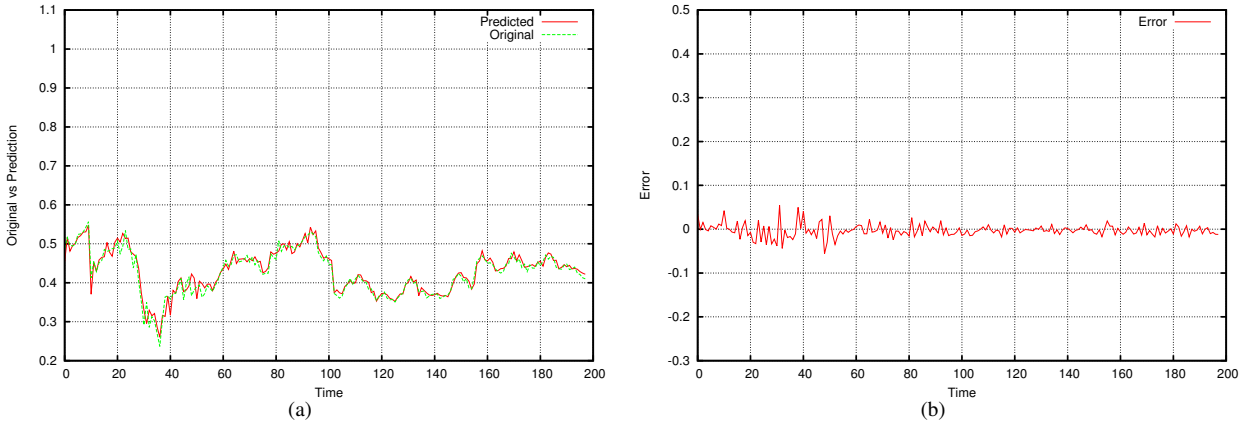

Fig. 3. Random Picked test dataset showing prediction by mNSL for ACI Worldwide Inc. dataset. (a) Performance given by mNSL on the testing set. (b) Error on the test dataset given by mNSL.

method at different stages of network helps in prediction and the order of update matters.

The proposed modified model has given better performances in nearly all of the datasets used when compared to its original method NSL. The results have shown improvements in terms of training and generalization. The results for mNSL method on Lorenz and financial datasets are better in terms of RMSE. Even when compared to CICC in majority of cases, the proposed modified hybrid model was able to outperform in terms of NSME. The proposed method under performed in terms of RMSE due to the fact that CICC was based on recurrent network.

In all of the cases, mNSL gave better performance than standalone methods based on cooperative coevolution (CCRNN-SL and CCRNN-NL) when compared to those in the literature. The proposed method mNSL has also given better performance in comparison to adaptive modularity cooperative coevolution (AMCC), where the problem decomposition method changed with given time.

One of the advantages of mNSL is the use of two different problem decomposition. The modification helped the model to generalize better and give competitive results. The cases where the method was unable to perform are due to either over training or over fitting.

VI. Conclusion

This paper has proposed a modification of a hybrid model and the proposed model is called the modified Neuron-Synapse Level decomposition (mNSL) which focuses on hidden layer neurons rather than different layers for update and cooperative coevolution. mNSL was used with feedforward neural network for time series prediction.

The research started with the testing of the proposed modified hybrid method with three benchmark datasets and later on two financial datasets. The application of different decomposition methods at different layers or stages of the

neural network helps in decision making and assist in diversity of search. By having subcomponent update based on hidden neurons rather then layers makes the model perform better in terms of generalization and testing.

The proposed methods performance in terms of best RMSE was competitive when compared to works from literature involving the five different datasets. In general, mNSL has shown better training and generalization when compared to its original method NSL. The best values are better in three out of five cases.

In future work, the proposed method can be applied to recurrent network to see its performance on time series prediction. Some other applications could be on pattern classification problems and global optimization problems. Also, the method can be used for multi-step ahead and multi-variant time series prediction as well.

REFERENCES

[1] M. Potter and K. De Jong, "A cooperative coevolutionary approach to function optimization," in *Parallel Problem Solving from Nature PPSN III*, ser. Lecture Notes in Computer Science, Y. Davidor, H.-P. Schwefel, and R. Mnner, Eds. Springer Berlin Heidelberg, 1994, vol. 866, pp. 249–257.

[2] R. Nand and R. Chandra, "Neuron-synapse level problem decomposition method for cooperative neuro-evolution of feedforward networks for time series prediction," in *Neural Information Processing*. Springer, 2015, pp. 90–100.

[3] N. García-Pedrajas, E. Sanz-Tapia, D. Ortiz-Boyer, and C. Hervás-Martínez, "Introducing multi-objective optimization in cooperative coevolution of neural networks," in *Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence*. Springer, 2001, pp. 645–652.

[4] R. Chandra and M. Zhang, "Cooperative coevolution of elman recurrent neural networks for chaotic time series prediction," *Neurocomputing*, vol. 86, pp. 116–123, 2012.

[5] R. Chandra, "Problem decomposition and adaptation in cooperative neuro-evolution," 2012.

[6] K. K. Bali, R. Chandra, and M. N. Omidvar, "Competitive island-based cooperative coevolution for efficient optimization of large-scale fully-separable continuous functions," in *International Conference on Neural Information Processing*. Springer, 2015, pp. 137–147.

[7] K. Deb, A. Anand, and D. Joshi, "A computationally efficient evolutionary algorithm for real-parameter optimization," *Evol. Comput.*, vol. 10, no. 4, pp. 371–395, 2002.

[8] Smart bilo: An open source computational intelligence framework. 2015. [Online]. Available: http://smartbilo.aicrg.softwarefoundationfiji.org/

[9] F. Takens, "Detecting strange attractors in turbulence," in *Dynamical Systems and Turbulence, Warwick 1980*, ser. Lecture Notes in Mathematics, 1981, pp. 366–381.

[10] C. Frazier and K. Kockelman, "Chaos theory and transportation systems: Instructive example," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 20, pp. 9–17, 2004.

[11] M. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," *Science*, vol. 197, no. 4300, pp. 287–289, 1977.

[12] E. Lorenz, "Deterministic non-periodic flows," *Journal of Atmospheric Science*, vol. 20, pp. 267 – 285, 1963.

[13] S. S., "Solar cycle forecasting: A nonlinear dynamics approach," *Astronomy and Astrophysics*, vol. 377, pp. 312–320, 2001.

[14] "NASDAQ Exchange Daily: 1970-2010 Open, Close, High, Low and Volume," accessed: 02-02-2015. [Online]. Available: http://www.nasdaq.com/symbol/aciw/stock-chart

[15] SILSO World Data Center, "The International Sunspot Number (1834-2001), International Sunspot Number Monthly Bulletin and Online Catalogue," Royal Observatory of Belgium, Avenue Circulaire 3, 1180 Brussels, Belgium, accessed: 02-02-2015. [Online]. Available: http://www.sidc.be/silso/

[16] R. Chandra and M. Zhang, "Cooperative coevolution of Elman recurrent neural networks for chaotic time series prediction," *Neurocomputing*, vol. 186, pp. 116 – 123, 2012.

[17] R. Chandra, "Adaptive problem decomposition in cooperative coevolution of recurrent networks for time series prediction," in *International Joint Conference on Neural Networks (IJCNN)*, Dallas, TX, USA, Aug. 2013, pp. 1–8.

[18] A. Gholipour, B. N. Araabi, and C. Lucas, "Predicting chaotic time series using neural and neurofuzzy models: A comparative study," *Neural Process. Lett.*, vol. 24, pp. 217–239, 2006.

[19] R. Chandra, "Competitive two-island cooperative coevolution for training Elman recurrent networks for time series prediction," in *International Joint Conference on Neural Networks (IJCNN)*, Beijing, China, Jul. 2014, pp. 565 – 572.

[20] S. Chand and R. Chandra, "Cooperative coevolution of feed forward neural networks for financial time series problem," in *International Joint Conference on Neural Networks (IJCNN)*, Beijing, China, July 2014, pp. 202–209.

[21] ——, "Multi-objective cooperative coevolution of neural networks for time series prediction," in *2014 International Joint Conference on Neural Networks (IJCNN)*, July 2014, pp. 190–197.