# Multi-Objective Cooperative Neuro-Evolution of Recurrent Neural Networks for Time Series Prediction

Rohitash Chandra
School of Computing Information and Mathematical Sciences
University of the South Pacific, Suva, Fiji.
Email: c.rohitash@gmail.com

***Abstract*—Cooperative coevolution is an evolutionary computation method which solves a problem by decomposing it into smaller subcomponents. Multi-objective optimization deals with conflicting objectives and produces multiple optimal solutions instead of a single global optimal solution. In previous work, a multi-objective cooperative co-evolutionary method was introduced for training feedforward neural networks on time series problems. In this paper, the same method is used for training recurrent neural networks. The proposed approach is tested on time series problems in which the different time-lags represent the different objectives. Multiple pre-processed datasets distinguished by their time-lags are used for training and testing. This results in the discovery of a single neural network that can correctly give predictions for data pre-processed using different time-lags. The method is tested on several benchmark time series problems on which it gives a competitive performance in comparison to the methods in the literature.**

## I. Introduction

Multi-objective optimization involves multiple functions that have conflicting objectives [1]. Unlike a single-objective problem, in which the aim is to find a global optimal solution, a multi-objective problem gives rise to a set of optimal solutions (known as pareto optimal) in which no solution can claim to be better than any other with respect to all the objective functions [1]. Any single objective component of a solution within the Pareto optimal set can only be improved by degrading at least one of its other objective components [2].

Cooperative coevolution is an evolutionary computation method that solves a problem by dividing it into smaller subcomponents [3]. Cooperative coevolution has shown to give a more diverse set of solutions in comparison to other single-population based evolutionary algorithms [4] and has been successfully used for training neural networks for main stream problems, such as pattern classification [5] and time series prediction [6].

Multi-objective cooperative coevolution has also been explored recently. Iorio and Li developed non-dominated sorting cooperative coevolutionary genetic algorithm (NSCCGA) which was able to compare well with NSGA-II [1] on some of the benchmark functions. The authors of [7] explored multi-objective cooperative coevolution using a special niching mechanism and an extending operator to maintain diversity. Their method displayed appealing results in finding more evenly distributed non-dominated solutions. Multi-objective cooperative coevolution has also been used for large scale optimization [8].

Time series prediction involves the use of past and present data in order to make future predictions [9] [10]. A way to improve time series prediction is to explore the different features of the time series data and to choose optimal values for the associated variables that are used for preprocessing such as the values for the time lag and embedding dimension [11]. Time lag is one of the most important features of time series prediction. It defines the interval at which data points are picked in order to reconstruct the time series data [12]. In this paper, the data sets reconstructed used different time lags as the set of objectives to be optimized. The proposed multi-objective approach aims to optimize and generalize candidate recurrent neural networks across multiple time lags. This method has shown good results with feedforward neural networks [11].

Multi-objective time series prediction using computational intelligence methods have been used to improve the prediction accuracy [13]. Multi-objective evolutionary algorithms have been used to optimize radial-basis networks for time series prediction which incorporated heuristics that were able to detect and remove networks which did not contribute much to the net output while preserving those that produced good results [14]. The use of multi-objective evolutionary neural networks for time series prediction employed training and validation accuracy as the two different objectives [13]. Multiple error measures have also been used as different objectives in training evolutionary neural networks with multi-objective optimization [15].

The main contribution of this paper is to introduce a multi-objective cooperative coevolutionary framework for training recurrent neural networks for time series prediction. The aim is to observe if multi-objective optimization using datasets with different *time lags* can help improve time series prediction. This paper extends previous work [11] where a bi-objective method was proposed for training feedforward neural networks (MOCCFNN) on time series problems.

The rest of the paper is organised as follows. Background information on the related concepts is given in Section 2. The proposed method is discussed in Section 3 while Section

4 presents the experimental results. Section 5 concludes the paper with a summary of the results and discussion on future research.

## II. Background

### A. Problem Decomposition for Neuro-Evolution

Problem decomposition is a major issue in cooperative coevolution as interdependencies are present among variables [5]. Cooperative coevolution works best for separable problems [3]. A good problem decomposition method groups variables with interdependencies together [16]. In using cooperative coevolution for training neural networks, the problem decomposition method will determine the breakdown of the neural network into subcomponents. The two main problem decomposition methods for neural networks are *Synapse level* (SL) [17] and *Neuron level* (NL) [18] decomposition. In SL, the number of weights determine the number of subcomponents whereas in NL, the number of subcomponents is equal to the number of hidden neurons, plus the number of context neurons, plus the number of output neurons. Neuron level problem decomposition has shown good results in time series [6] and pattern classification problems [5].

### B. Time Series Prediction

Time series data has to be preprocessed according to a set of parameters before it could be used for training and testing. Taken's theorem was used for reconstructing the original time series into a phase space that was used by prediction models for training [12]. The *time lag* defined the interval at which the data points were picked and the *embedding dimension* specified the size of the sliding window that was used to capture points to make a reconstructed phase space [12]. These two variables were crucial as they determined the extent to which the pattern and features of the original time series was retained.

Some research has been done on the importance of time lags in time series prediction. Quantum-inspired hybrid methods have been explored in order to determine the best possible time-lag to represent the original time series, with good results on financial prediction [19]. A hybrid model that combined neural networks with a modified genetic algorithm was proposed to perform an evolutionary search for the minimum necessary time-lags for determining the phase space that generates the time series [20]. A meta-evolutionary algorithm simultaneously evolved both the neural networks and the set of lags needed to predict the time series [21]. A morphological rank linear time-lag added evolutionary forecasting method was also proposed that carries out an evolutionary search for the lowest number of relevant time-lags necessary to efficiently represent the patterns and characteristics of a complex time series [22].

Time series data needs to be preprocessed and reconstructed into a state space vector [12]. Given an observed time series $x(t)$, an embedded phase space $Y(t) = [(x(t), x(t-T), ..., x(t(D-1)T)]$ can be generated, where, $T$ is the time delay, $D$ is the embedding dimension, $t = 0, 1, 2, ..., N - DT - 1$ and $N$ is the length of the original time series [12]. Taken's theorem expresses that the vector series reproduces many important characteristics of the original time series.

### C. Recurrent Neural Networks

A recurrent neural network has feedback loops from its outputs to its inputs which greatly improves its learning capability [23]. It is composed of an input layer, a context layer, hidden layers and an output layer, each of which consist of one or more neurons [6]. The context layer is used to preserve state information. Recurrent neural networks have been successfully applied to a wide range of problems such as time series prediction [6] and grammatical inference problems [18]. The dynamics of the change of hidden state neuron activation's in Elman style recurrent networks is given by the Equation (1) as shown in Figure 1.

$$y_i(t) = f\left(\sum_{k=1}^{K} v_{ik}\, y_k(t-1) + \sum_{j=1}^{J} w_{ij}\, x_j(t-1)\right) \tag{1}$$

where $y_k(t)$ and $x_j(t)$ represent the output of the context state neuron and input neurons respectively. $v_{ik}$ and $w_{ij}$ represent their corresponding weights. $f(.)$ is a sigmoid transfer function.

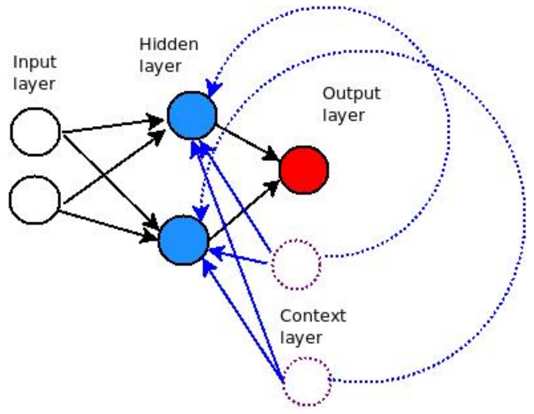


Fig. 1. Elman recurrent neural network used for time series prediction. Although two neurons are showing an input layer, note that only one neuron is used in the input and output layer for the time series prediction problems. The number of hidden neurons vary as per application. The network unfolds in time according to the size of the dimension (D).

### D. Cooperative Neuro-Evolution for Time Series Prediction

Initial work on the use of cooperative coevolution for recurrent neural networks on time series prediction focused on the effects of different problem decomposition methods and showed better performance than some of the established methods from the literature [6]. An adaptive modularity cooperative coevolution (AMCC) framework for training recurrent neural networks was proposed [24] and applied later for time series prediction [25]. The method utilized three different problem decomposition methods that adapted or changed amongst each other during evolutionary process. AMCC gave competitive results in comparison to the methods in the literature when tested on benchmark problems. An application to financial time series prediction was done using cooperative coevolution of feedforward networks [26]. The method achieved good results on real world datasets from the NASDAQ stock exchange. A competitive cooperative coevolutionary method was proposed for training recurrent networks on time series problems [27], [28]. This method showed considerable improvement in comparison to standalone cooperative coevolution.

Multi-objective evolutionary algorithms have been used to optimize radial-basis networks for time series prediction [14]. The use of multi-objective evolutionary neural networks for time series prediction employed training and validation accuracy as the two different objectives [13]. Multiple error measures have also been used as the different objectives in training evolutionary neural networks with multi-objective optimization [15]. Hybrid fuzzy model has been proposed for predicting non-linear time series data in which their two objectives were to improve prediction accuracy and minimize the number of required fuzzy rules [29]. A knee-point strategy multi-objective approach has shown promising results for evolving feedforward neural networks when compared to established multi-objective evolutionary algorithms [30]. Hybrid multi-objective evolutionary method has been used for evolution of recurrent neural network weights and structure with ensembles where a set of Pareto solutions are obtained [31] and has shown promising results.

## III. Multi-Objective Cooperative Neuro-Evolution

**Alg. 1** Multi-Objective Cooperative Neuro-Evolution

```
Step 1: Decompose the problem into $k$ subcomponents using Neuron level decomposition
Step 2: Initialize and cooperatively evaluate each sub-population for each objective
Step 3: Rank the sub-population and assign pareto front
for each cycle until termination do
    for each Sub-population do
        for $n$ Generations do
            i) Select and create new offspring using Parent-Centric Crossover
            ii) Cooperatively evaluate the new offspring for each objective
            iii) Update sub-population with the best individuals
            iv) Rank the sub-population and identify non-dominated individuals
            v) Assign Pareto Front
        end for
    end for
end for
```

As mentioned in the previous section, cooperative coevolution has shown good results with training neural networks on time series problems. However, there is still room for improvement in terms of quality and accuracy of results. A multi-objective approach can help improve results further. The datasets with different time lags are used as different objectives to be optimized. Different time-lag values reproduce the original time series in different ways, which in effect means dealing with different datasets. Training with different time-lags can allow the neural network to generalize better and explore different aspects and patterns within the time series. Noise within the dataset is also a major issue and the time-lag determines how much of noise to be present.

In traditional cooperative coevolution, the fitness of an individual in a particular sub-population is computed by concatenating it with the best individuals from the rest of the sub-populations. The complete solution is then evaluated and a fitness value is assigned to the individual whose fitness was being evaluated. This despite the fact that the particular individual only formed part of the complete solution [3].

The given multi-objective method builds on this particular framework. In Algorithm 1, the recurrent neural network is decomposed using neuron level decomposition method into $k$ subcomponents where $k$ is equal to the sum of the number of hidden neurons, plus the number of context neurons, plus the number of output neurons [18]. Each sub-population is randomly initialized and the individuals within the sub-population are evaluated for fitness. Each individual is evaluated for two given objectives which are represented by the different time-lags.

In a multi-objective environment, there are multiple optimal solutions in which no solution can claim to be better than any other with respect to all the objective functions [1]. For this reason, during fitness evaluation, the representative of each sub-population is chosen randomly from the set of non-dominated individuals for that particular sub-population. The fitness of an individual in a particular sub-population is computed by combining it with randomly chosen non-dominated individuals from the rest of the sub-populations to form a complete solution. The complete solution is then encoded into a neural network and evaluated on the given objectives. The resulting network error is assigned as the fitness of the individual which was being evaluated, even though it is just a small component of the overall solution. During the initialization phase, when the non-dominated individuals are unknown, an individual is evaluated by selecting and concatenating random individuals from the other sub-populations.

Once the fitness is evaluated, all sub-populations are ranked and the non-dominated individuals are identified. An individual is given a rank according to the number of individuals within the sub-population that dominate that particular individual. An individual will have a rank of 0 if it is non-dominated. Based on their rank, the individuals are assigned to the different set of pareto fronts. Subsequently, individuals with a rank of 1 belong to the second pareto front and so on until the entire population is classified. This classification allows us to easily keep track of non-dominated individuals and determine how the rest of the individuals are performing in comparison to the optimal solutions.

The generalized generation gap parent-centric crossover genetic algorithm [32] is used to evolve the sub-populations. All sub-populations are evolved using the parent centric crossover operator. To generate an offspring, $n$ parents are chosen. One of them is randomly chosen from the non-dominated front while the others are randomly chosen from the entire sub-population. The crossover operator is applied to the chosen parents and the offspring are generated. Once the offspring are generated, they are evaluated for fitness and compared with a new set of $x$ parents which are randomly chosen from the sub-population. The $x$ best individuals from this pool are picked and replaced in the main sub-population.

The algorithm is terminated once the maximum number of function evaluations are reached. A final solution is then generated and evaluated with the testing datasets (one per objective). Each sub-population has one representing individual in the final solution. The representing individual is randomly chosen from the set of non-dominated individuals for that particular sub-population. As mentioned earlier, in a multi-objective environment, multiple optimal solutions exist where none can claim to be better than the others. For this reason, a random selection from the non-dominated front is the best option. Therefore, one neural network is trained to give predictions for multiple time-lags.

## IV. Experiments & Results

### A. Experimental Setup

The proposed method is tested on four time series problems. Three of these are benchmark data sets while the remaining one is a financial data set taken from the NASDAQ stock exchange. Mackey-Glass, Lorenz and the Sunspot datasets represent the benchmark problems [33]. A total of 1000 data points are used from the Mackey-Glass and Lorenz time series while for the sunspot time series, a larger data set of 2000 points was used. In the financial data set, the closing stock prices of ACI Worldwide Inc., between the period of December 2006 and February 2010 is used [34]. This is equivalent to around 800 data points. Each dataset is divided into training and testing set using a 50-50 split as this will allow for proper training and good test for generalization. The embedding dimension for the real world problems (Sunspot and ACI) is set to $d$=5 and for the simulated problems (Mackey-Glass and Lorenz), $d$=4 is used.

Root mean squared error (RMSE) is used to evaluate the performance of the proposed method. The *depth of search* is kept at 1 for all the sub-populations as this number has achieved good results previously [6]. Neuron level problem decomposition method is used for dividing the neural network into subcomponents. The maximum number of function evaluations is set to 50,000 and the population size is kept at 300. These parameters are kept the same in comparison to an earlier method in which feedforward networks are used [11].

The G3-PCX evolutionary algorithm is used to create and evolve all the sub-populations. The G3-PCX genetic algorithm uses the *generation gap model* [32] for selection. A pool size of 2 parents and 2 offspring was used as in previous works [6]. Each individual within the population maintains multiple fitness values for the different objectives. The two objectives for this experiment are the different *time lag (T)* values ($T$=2 & $T$=3).

### B. Results

This section reports the performance of the proposed method for training recurrent neural networks for the time series prediction. The results for 50 experimental runs with 95% confidence interval are given in Tables I - IV with the best results highlighted in bold.

In the Mackey-Glass time series problem, the multi-objective method recorded the best training and generalization performance with 7 hidden neurons. The overall performance improved as the number of hidden neurons increased up-till an optimal number of 7 hidden neurons. There was no significant difference in the performance of the proposed method across different time-lags. This shows that the method was able to generalize well across the different time-lags. Another important observation was that there was no significant difference in the training and generalization performance.

Just like the Mackey-Glass time series, the Lorenz time series is also a simulated problem. For this problem, the proposed method recorded the best training and generalization performance with 9 hidden neurons. The performance kept improving as the number of hidden neurons increased. Once again, the method was able to generalize well across the different time-lags with a uniform performance. Just like with the Mackey-Glass problem, the training and generalization performance were similar across the different hidden neurons.

The real test for the proposed method were the Sunspot and ACI Worldwide Inc. time series problems. These are real world time series which contain noise that makes prediction difficult. In the Sunspot time series problem, the proposed method recorded the best training and generalization performance with 5 hidden neurons. As the number of hidden neurons increased (beyond 5 neurons), the performance significantly deteriorated. As with the two previous problems, the proposed method was able to generalize well across the different time-lags with stable performance.

The ACI Worldwide Inc. time series has recorded the best training and generalization performance with 5 hidden neurons. It gave a better performance for the data set reconstructed using a time-lag of 3. The method was able to generalize well across the different hidden neurons but did not manage to give a uniform performance across the different time-lags. This can be partially attributed to the nature of the problem as stock market time series tend to be highly chaotic and noisy. As a result, different time-lags inherit different amounts of this noise.

### C. Discussion

The goal of multi-objective time series prediction is to improve the prediction accuracy [15]. The proposed method produced good results on all the benchmark problems and also outperformed majority of the methods in the literature. As shown in Tables V - VIII, it gave better results in comparison to some of the recent and related methods such as the cooperative coevolutionary recurrent neural network (CCRNN) [6], the Type-2 fuzzy neural network [35] and the adaptive modularity cooperative co-evolutionary recurrent neural network (AMCC-RNN) [25]. The proposed method also outperformed MOCCFNN [11] on both the real world time series problems (Sunspot & ACI Worldwide Inc.). In addition, the recurrent neural network based method (MOCCRNN) records lower computation time in comparison to feedforward neural networks (MOCCFNN). The proposed method used recurrent neural network that contains feedback connections that seem to be more appropriate than feedforward networks [11]. This can also be the reason of better generalization performance in comparison to other methods in the literature

The proposed method enabled to utilise different frequencies defined by time-lag within the time series data, which can be optimised in different problems. It allows for a more in-depth utilisation of information that depend on this parameter used for feature extraction and embedding. There needs to be more in-depth analysis of the time-lag parameter in order to take full advantage of embedding. With the proposed method, a single neural network can be generalized for several different time-lags and hence it would be useful in datasets with noise and also missing data variables.

## V. Conclusion

This paper presented a multi-objective cooperative coevolutionary method for training recurrent neural networks. The method was applied to time series prediction where different time-lags defined different sets of features and were used as competing objectives. The multi-objective method gave good results on different benchmark problems and also outperformed majority of the methods from the literature. The proposed method enabled recurrent neural networks perform better than feedforward networks on real world datasets. The recurrent neural network based multi-objective method also records lower computation time in comparison to feedforward networks.

In future research, different combinations of the embedding dimension and time-lag can be used to extend the proposed method. In the current framework, the embedding dimension was fixed, but this can vary and could help to improve the results further in future. The proposed method can be applied to real world problems such as climate change problems for cyclone track and wind intensity prediction.

## References

[1] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 182–197, 2002.

[2] C. K. Goh, K. C. Tan, D. S. Liu, and S. C. Chiam, "A competitive and cooperative co-evolutionary approach to multi-objective particle swarm optimization algorithm design," *European Journal of Operational Research*, vol. 202, pp. 42–54, 2010.

[3] 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.

[4] M. A. Potter and K. A. De Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evol. Comput.*, vol. 8, pp. 1–29, 2000.

[5] R. Chandra, M. Frean, and M. Zhang, "On the issue of separability for problem decomposition in cooperative neuro-evolution," *Neurocomputing*, vol. 87, pp. 33–40, 2012.

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

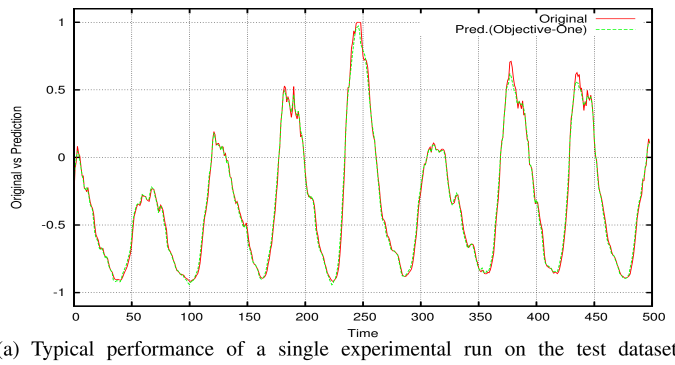
[7] K. C. Tan, T. H. Lee, Y. J. Yang, and D. S. Liu, "A cooperative coevolutionary algorithm for multiobjective optimization," *2004 IEEE International Conference on Systems, Man and Cybernetics*, pp. 1926—1931, 2004.

[8] L. M. Antonio and C. A. Coello Coello, "Use of cooperative coevolution for solving large scale multiobjective optimization problems," *2013 IEEE Congress on Evolutionary Computation*, pp. 2758–2765, 2013.

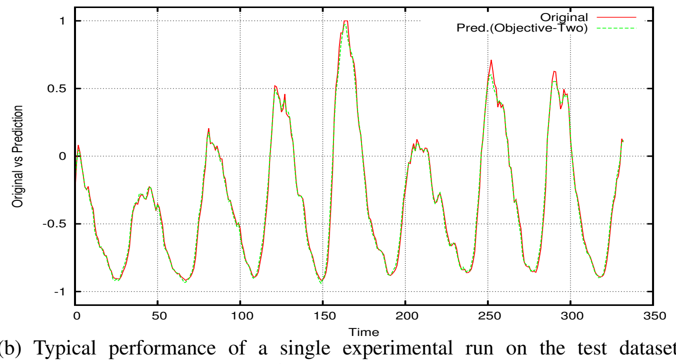
[9] E. Lorenz, "Deterministic non-periodic flows," *Journal of Atmospheric Science*, vol. 20, pp. 267 – 285, 1963.

[10] H. K. Stephen, *In the Wake of Chaos: Unpredictable Order in Dynamical Systems*. University of Chicago Press, 1993.

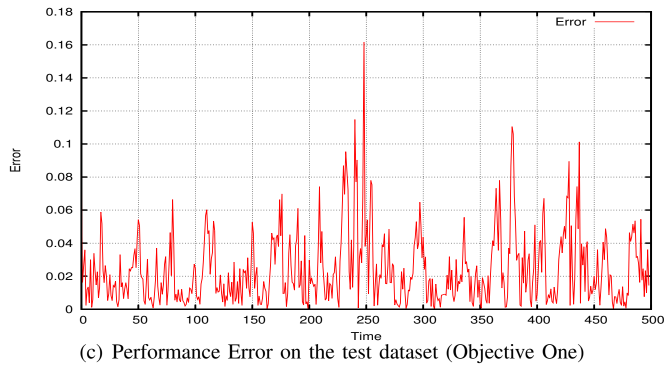
[11] S. Chand and R. Chandra, "Multi-objective cooperative coevolution of neural networks for time series prediction," in *International Joint Conference on Neural Networks (IJCNN)*, Beijing, China, July 2014, pp. 190–197.



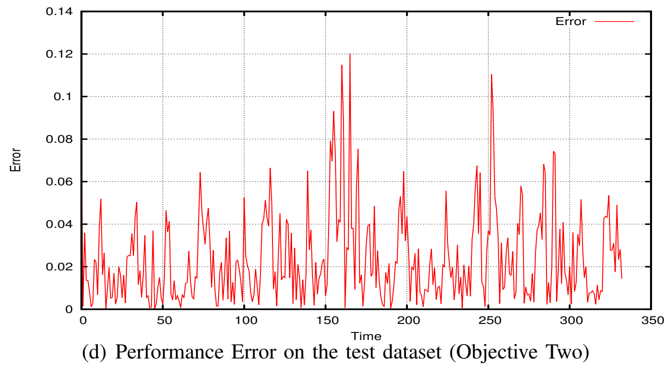
(a) Typical performance of a single experimental run on the test dataset (Objective One) (RMSE: 0.0323)



(b) Typical performance of a single experimental run on the test dataset (Objective Two) (RMSE: 0.0315)



(c) Performance Error on the test dataset (Objective One)



(d) Performance Error on the test dataset (Objective Two)

Fig. 2. Typical prediction performance given by by MOCC-RNN for Sunspot time series.

TABLE I. The prediction training and generalization performance (RMSE) for Mackey Glass data set.

| Prob. | Hidden | Training | Generalization | Best |
|---|---|---|---|---|
| *T*=2 | 3 | 1.13E-02 ± 7.39E-04 | 1.13E-02 ± 7.43E-04 | 6.52E-03 |
| | 5 | 1.07E-02 ± 6.28E-04 | 1.07E-02 ± 6.34E-04 | 5.79E-03 |
| | 7 | **1.01E-02 ± 6.79E-04** | **1.01E-02 ± 6.83E-04** | 6.29E-03 |
| | 9 | 1.11E-02 ± 6.57E-04 | 1.11E-02 ± 6.58E-04 | **5.11E-03** |
| *T*=3 | 3 | 1.13E-02 ± 7.36E-04 | 1.12E-02 ± 7.44E-04 | 6.34E-03 |
| | 5 | 1.07E-02 ± 6.24E-04 | 1.07E-02 ± 6.36E-04 | 5.78E-03 |
| | 7 | **1.01E-02 ± 6.75E-04** | **1.00E-02 ± 6.87E-04** | 6.24E-03 |
| | 9 | 1.11E-02 ± 6.54E-04 | 1.10E-02 ± 6.61E-04 | **4.99E-03** |

TABLE II. The prediction training and generalization performance (RMSE) for Lorenz data set.

| Prob. | Hidden | Training | Generalization | Best |
|---|---|---|---|---|
| *T*=2 | 3 | 1.73E-02 ± 1.54E-03 | 1.76E-02 ± 1.55E-03 | 7.45E-03 |
| | 5 | 1.39E-02 ± 1.32E-03 | 1.41E-02 ± 1.37E-03 | 5.93E-03 |
| | 7 | 1.49E-02 ± 1.09E-03 | 1.53E-02 ± 1.15E-03 | 6.65E-03 |
| | 9 | **1.36E-02 ± 1.39E-03** | **1.37E-02 ± 1.40E-03** | **5.26E-03** |
| *T*=3 | 3 | 1.74E-02 ± 1.54E-03 | 1.75E-02 ± 1.55E-03 | 7.43E-03 |
| | 5 | 1.40E-02 ± 1.32E-03 | 1.41E-02 ± 1.36E-03 | 5.92E-03 |
| | 7 | 1.50E-02 ± 1.11E-03 | 1.52E-02 ± 1.15E-03 | 6.63E-03 |
| | 9 | **1.36E-02 ± 1.38E-03** | **1.37E-02 ± 1.39E-03** | **5.24E-03** |

TABLE III. The prediction training and generalization performance (RMSE) for Sunspot data set.

| Prob. | Hidden | Training | Generalization | Best |
|---|---|---|---|---|
| *T*=2 | 3 | 1.96E-02 ± 1.61E-03 | 4.88E-02 ± 7.76E-03 | 1.91E-02 |
| | 5 | **1.72E-02 ± 1.22E-03** | **4.50E-02 ± 9.06E-03** | 1.64E-02 |
| | 7 | 1.73E-02 ± 1.15E-03 | 5.86E-02 ± 1.82E-02 | 1.72E-02 |
| | 9 | 1.87E-02 ± 1.71E-03 | 7.60E-02 ± 2.10E-02 | **1.50E-02** |
| *T*=3 | 3 | 1.98E-02 ± 1.56E-03 | 4.88E-02 ± 7.83E-03 | 1.82E-02 |
| | 5 | **1.73E-02 ± 1.17E-03** | **4.49E-02 ± 9.04E-03** | 1.61E-02 |
| | 7 | 1.75E-02 ± 1.17E-03 | 5.86E-02 ± 1.82E-02 | 1.71E-02 |
| | 9 | 1.90E-02 ± 1.67E-03 | 7.60E-02 ± 2.10E-02 | **1.53E-02** |

TABLE IV. The prediction training and generalization performance (RMSE) for ACI Worldwide Inc. data set.

| Prob. | Hidden | Training | Generalization | Best |
|---|---|---|---|---|
| *T*=2 | 3 | 2.15E-02 ± 3.29E-04 | 2.15E-02 ± 7.11E-04 | 1.91E-02 |
| | 5 | **2.13E-02 ± 2.08E-04** | **2.08E-02 ± 3.39E-04** | **1.90E-02** |
| | 7 | 2.22E-02 ± 6.85E-04 | 2.21E-02 ± 1.05E-03 | 1.92E-02 |
| | 9 | 2.43E-02 ± 2.18E-03 | 2.56E-02 ± 4.15E-03 | 1.91E-02 |
| *T*=3 | 3 | 2.24E-02 ± 3.34E-04 | 1.76E-02 ± 8.16E-04 | 1.44E-02 |
| | 5 | **2.21E-02 ± 2.40E-04** | **1.67E-02 ± 4.88E-04** | **1.43E-02** |
| | 7 | 2.30E-02 ± 6.49E-04 | 1.83E-02 ± 1.29E-03 | **1.43E-02** |
| | 9 | 2.51E-02 ± 2.16E-03 | 2.20E-02 ± 4.30E-03 | 1.45E-02 |

TABLE V. A comparison with the results from literature on the Mackey-Glass time series

| Prediction Method | RMSE | NMSE |
|---|---|---|
| Auto regressive moving average with neural network (ARMA-ANN)(2008) [36] | 2.50E-03 | |
| Radial basis network with orthogonal least squares (RBF-OLS)(2006) [37] | 1.02E-03 | |
| Locally linear neuro-fuzzy model - Locally linear model tree (LLNF-LoLiMot) (2006) [37] | 9.61E-04 | |
| Boosted recurrent neural networks (2006) [38] | | 1.60E-04 |
| Neuro-fuzzy system with time delay coordinates (2008) [39] | | 1.26E-03 |
| Particle swarm optimisation (CCPSO) (2009) [40] | 8.42E-03 | |
| Neural fuzzy network and particle swarm optimisation (PS0) (2009) [40] | 2.10E-02 | |
| Neural fuzzy network and cooperative particle swarm optimisation (CPS0) (2009) [40] | 1.76E-02 | |
| Neural fuzzy network and differential evolution (DE) (2009) [40] | 1.62E-02 | |
| Neural fuzzy network and genetic algorithm (GA ) (2009)[40] | 1.63E-02 | |
| Hybrid NARX-Elman RNN with Residual Analysis (2010) [41] | 3.72E-05 | 2.70E-08 |
| Backpropagation neural network and genetic algorithms with residual analysis (2011) [42] | 1.30E-03 | |
| Synapse Level-CCRNN (2012) [6] | 6.33E-03 | 2.79E-04 |
| Neuron Level-CCRNN (2012) [6] | 8.28E-03 | 4.77E-04 |
| HMM-Fuzzy with EA (2012) [29] | 4.80E-03 | |
| AMCC-RNN (2013) [25] | 7.53E-03 | 3.90E-04 |
| Type-2 Fuzzy Neural Networks (2014) [35] | 3.90E-02 | |
| MOCCFNN with 2 objectives (T=2)[11] | 3.84E-03 | 2.80E-05 |
| MOCCFNN with 2 objectives (T=3)[11] | 3.77E-03 | 2.70E-05 |
| **Proposed MOCCRNN with 2-objectives (T=2)** | **5.11E-03** | **1.82E-04** |
| **Proposed MOCCRNN with 2-objectives (T=3)** | **4.99E-03** | **1.73E-04** |

TABLE VI. A COMPARISON WITH THE RESULTS FROM LITERATURE ON THE LORENZ TIME SERIES

| Prediction Method | RMSE | NMSE |
|---|---|---|
| Boosted recurrent neural networks (2006) [38] | | 3.77E-03 |
| Evolutionary RNN (2007) [43] | 8.79E-06 | 9.90E-10 |
| Auto regressive moving average with neural network (ARMA-ANN)(2008) [36] | 8.76E-02 | |
| Backpropagation-through-time (BPTT-RNN) (2010) [44] | | 1.85E-03 |
| Real time recurrent learning (RTRL-RNN) (2010) [44] | | 1.72E-03 |
| Recursive Bayesian LevenbergMarquardt (RBLM-RNN) (2010) [44] | | 9.0E-04 |
| Hybrid NARX-Elman RNN with Residual Analysis (2010) [41] | 1.08E-04 | 1.98E-10 |
| Backpropagation neural network and genetic algorithms with residual analysis (2011) [42] | 2.96E-02 | |
| Synapse Level-CCRNN (2012) [6] | 6.36E-03 | 7.72E-04 |
| Neuron Level-CCRNN (2012) [6] | 8.20E-03 | 1.28E-03 |
| AMCC-RNN (2013) [25] | 5.06E-03 | 4.88E-04 |
| MOCCFNN with 2 objectives (T=2)[11] | 2.19E-03 | 2.53E-05 |
| MOCCFNN with 2 objectives (T=3)[11] | 2.18E-03 | 2.54E-05 |
| **Proposed MOCCRNN with 2-objectives (T=2)** | **5.26E-03** | **5.36E-04** |
| **Proposed MOCCRNN with 2-objectives (T=3)** | **5.24E-03** | **5.33E-04** |

TABLE VII. A COMPARISON WITH THE RESULTS FROM LITERATURE ON THE SUNSPOT TIME SERIES

| Prediction Method | RMSE | NMSE |
|---|---|---|
| Radial basis network with orthogonal least squares (RBF-OLS)(2006) [37] | | 4.60E-02 |
| Locally linear neuro-fuzzy model - Locally linear model tree (LLNF-LoLiMot) (2006) [37] | | 3.20E-02 |
| Hybrid NARX-Elman RNN with Residual Analysis (2010) [41] | 1.19E-02 | 5.90E-04 |
| Synapse Level-CCRNN (2012) [6] | 1.66E-02 | 1.47E-03 |
| Neuron Level-CCRNN (2012) [6] | 2.60E-02 | 3.62E-03 |
| AMCC-RNN (2013) [25] | 2.41E-02 | 3.11E-03 |
| MOCCFNN with 2 objectives (T=2)[11] | 1.84E-02 | 1.02E-03 |
| MOCCFNN with 2 objectives (T=3)[11] | 1.81E-02 | 9.98E-04 |
| **Proposed MOCCRNN with 2-objectives (T=2)** | **1.50E-02** | **1.52E-03** |
| **Proposed MOCCRNN with 2-objectives (T=3)** | **1.53E-02** | **1.57E-03** |

TABLE VIII. A COMPARISON WITH THE RESULTS FROM LITERATURE ON THE ACI-WORLDWIDE INC. TIME SERIES

| Prediction Method | RMSE | NMSE |
|---|---|---|
| CCFNN [26] | 1.91E-02 | - |
| MOCCFNN with 2 objectives (T=2)[11] | 1.90E-02 | - |
| MOCCFNN with 2 objectives (T=3)[11] | 1.55E-02 | - |
| **Proposed MOCCRNN with 2-objectives (T=2)** | **1.90E-02** | **4.39E-03** |
| **Proposed MOCCRNN with 2-objectives (T=3)** | **1.43E-02** | **2.66E-03** |

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

[13] S. Chiam, K. Tan, and A. Mamun, "Multiobjective evolutionary neural networks for time series forecasting," in *Evolutionary Multi-Criterion Optimization*, ser. Lecture Notes in Computer Science, S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, Eds. Springer Berlin Heidelberg, 2007, vol. 4403, pp. 346–360.

[14] J. Gonzlez, I. Rojas, H. Pomares, and J. Ortega, "RBF neural networks, multiobjective optimization and time series forecasting," in *Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence*, ser. Lecture Notes in Computer Science, J. Mira and A. Prieto, Eds. Springer Berlin Heidelberg, 2001, vol. 2084, pp. 498–505.

[15] J. Fieldsend and S. Singh, "Pareto evolutionary neural networks," *Neural Networks, IEEE Transactions on*, vol. 16, no. 2, pp. 338–354, Mar. 2005.

[16] R. Salomon, "Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. a survey of some theoretical and practical aspects of genetic algorithms," *Biosystems*, vol. 39, no. 3, pp. 263 – 278, 1996.

[17] F. Gomez, J. Schmidhuber, and R. Miikkulainen, "Accelerated neural evolution through cooperatively coevolved synapses," *J. Mach. Learn. Res.*, vol. 9, pp. 937–965, 2008.

[18] R. Chandra, M. Frean, M. Zhang, and C. W. Omlin, "Encoding subcomponents in cooperative co-evolutionary recurrent neural networks," *Neurocomputing*, vol. 74, no. 17, pp. 3223 – 3234, 2011.

[19] R. de A Araujo, A. de Oliveira, and S. Soares, "A quantum-inspired hybrid methodology for financial time series prediction," in *Neural Networks (IJCNN), The 2010 International Joint Conference on*, Barcelona, Spain, Jul. 2010, pp. 1–8.

[20] T. Ferreira, G. Vasconcelos, and P. Adeodato, "A new evolutionary approach for time series forecasting," in *Computational Intelligence and Data Mining, Proceedings of the IEEE Symposium on*, Honolulu, Hawaii, USA, Mar. 2007, pp. 616–623.

[21] E. Parras-Gutierrez and V. Rivas, in *Time series forecasting: Automatic determination of lags and radial basis neural networks for a changing horizon environment*, Barcelona, Spain, Jul. 2010, pp. 1–7.

[22] R. de A.Araujo, R. Aranildo, and T. Ferreira, in *Morphological-Rank-Linear Time-lag Added Evolutionary Forecasting method for financial time series forecasting*, Hong Kong, China, Jun. 2008, pp. 1340–1347.

[23] M. Negnevitsky, *Artificial Intelligence A Guide to Intelligent Systems : 2nd Edition*. Addison Wesley, 2005.

[24] R. Chandra, M. Frean, and M. Zhang, "Adapting modularity during learning in cooperative co-evolutionary recurrent neural networks," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 16, no. 6, pp. 1009–1020, 2012.

[25] 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, August 2013, pp. 1–8.

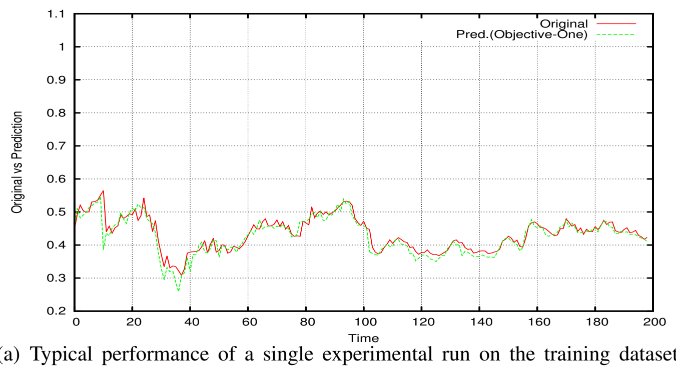
[26] S. Chand and R. Chandra, "Cooperative coevolution of feed forward neural networks for financial time series problem," in *International Joint Conference on Neural Networks (IJCNN)*, Beijing, China, July 2014, pp. 202–209.

[27] 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, July 2014, pp. 565–572.

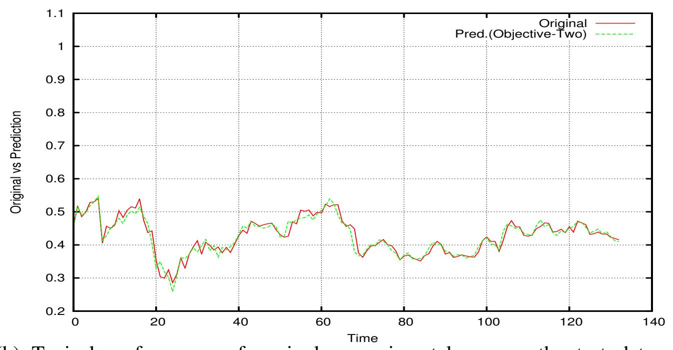
[28] ——, "Competition and collaboration in cooperative coevolution of Elman recurrent neural networks for time-series prediction," *Neural Networks and Learning Systems, IEEE Transactions on*, p. In Press, 2015.

[29] M. R. Hassan, B. Nath, M. Kirley, and J. Kamruzzaman, "A hybrid of multiobjective evolutionary algorithm and hmm-fuzzy model for time series prediction," *Neurocomputing*, vol. 81, pp. 1–11, 2012.

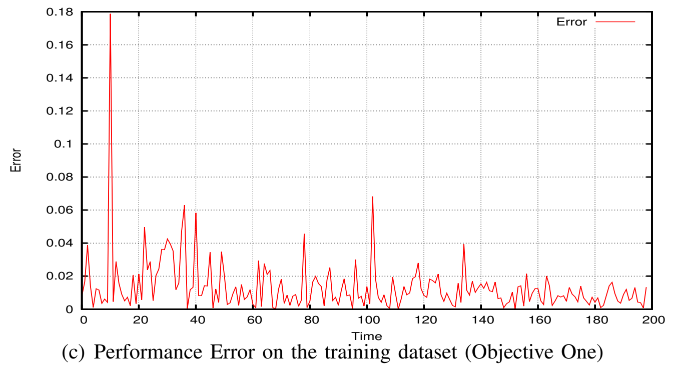
[30] W. Du, S. Y. S. Leung, and C. K. Kwong, "Time series forecasting by neural networks: A knee point-based multiobjective evolutionary



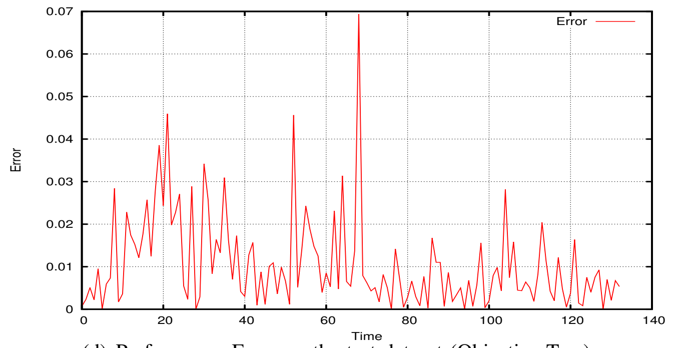
(a) Typical performance of a single experimental run on the training dataset (Objective One) (RMSE: 0.0215)



(b) Typical performance of a single experimental run on the test dataset (Objective Two) (RMSE: 0.0151)



(c) Performance Error on the training dataset (Objective One)



(d) Performance Error on the test dataset (Objective Two)

Fig. 3. Typical prediction performance given by by MOCC-RNN for Finance-ACI time series.

algorithm approach," *Expert Systems with Applications*, vol. 41, no. 18, pp. 8049 – 8061, 2014.

[31] C. Smith and Y. Jin, "Evolutionary multi-objective generation of recurrent neural network ensembles for time series prediction," *Neurocomputing*, vol. 143, pp. 302 – 311, 2014.

[32] 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.

[33] 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/

[34] "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

[35] F. Gaxiola, P. Melin, F. Valdez, and O. Castillo, "Interval type-2 fuzzy weight adjustment for backpropagation neural networks with application in time series prediction," *Information Sciences*, vol. 260, pp. 1 – 14, 2014.

[36] I. Rojas, O. Valenzuela, F. Rojas, A. Guillen, L. Herrera, H. Pomares, L. Marquez, and M. Pasadas, "Soft-computing techniques and arma model for time series prediction," *Neurocomputing*, vol. 71, no. 4-6, pp. 519 – 537, 2008.

[37] 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.

[38] M. Assaad, R. Bon, and H. Cardot, "Predicting chaotic time series by boosted recurrent neural networks," in *Neural Information Processing*, ser. Lecture Notes in Computer Science, I. King, J. Wang, L.-W. Chan, and D. Wang, Eds. Springer Berlin / Heidelberg, 2006, vol. 4233, pp. 831–840.

[39] J. Zhang, H. Shu-Hung Chung, and W.-L. Lo, "Chaotic time series prediction using a neuro-fuzzy system with time-delay coordinates," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 20, no. 7, pp. 956 –964, july 2008.

[40] C.-J. Lin, C.-H. Chen, and C.-T. Lin, "A hybrid of cooperative particle swarm optimization and cultural algorithm for neural fuzzy networks and its prediction applications," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 39, no. 1, pp. 55–68, Jan. 2009.

[41] M. Ardalani-Farsa and S. Zolfaghari, "Chaotic time series prediction with residual analysis method using hybrid Elman-NARX neural networks," *Neurocomputing*, vol. 73, no. 13-15, pp. 2540 – 2553, 2010.

[42] ——, "Residual analysis and combination of embedding theorem and artificial intelligence in chaotic time series forecasting," *Appl. Artif. Intell.*, vol. 25, pp. 45–73, 2011.

[43] Q.-L. Ma, Q.-L. Zheng, H. Peng, T.-W. Zhong, and L.-Q. Xu, "Chaotic time series prediction based on evolving recurrent neural networks," in *Proceedings of the International Conference on Machine Learning and Cybernetics*, Hong Kong, China, Aug. 2007, pp. 3496 –3500.

[44] D. Mirikitani and N. Nikolaev, "Recursive bayesian recurrent neural networks for time-series modeling," *Neural Networks, IEEE Transactions on*, vol. 21, no. 2, pp. 262 –274, Feb. 2010.