Cooperative Coevolution of Feed Forward Neural Networks for Financial Time Series Problem

Shelvin Chand and Rohitash Chandra

Abstract— **Intelligent financial prediction systems guide investors in making good investments. Investors are continuously on the hunt for better financial prediction systems. Neural networks have shown good results in the area of financial prediction. Cooperative coevolution is an evolutionary computation method that decomposes the problem into subcomponents and has shown promising results for training neural networks. This paper presents a computational intelligence framework for financial prediction where cooperative coevolutionary feedforward neural networks are used for predicting closing market prices for companies listed on the NASDAQ stock exchange. Problem decomposition is an important step in cooperative coevolution that affects its performance. Synapse and Neuron level are the main problem decomposition methods in cooperative coevolution. These two methods are used for training neural networks on the given financial prediction problem. The results show that Neuron level problem decomposition gives better performance in general. A prototype of a mobile application is also given for investors that can be used on their Android devices.**

I. Introduction

Time-series prediction involves the use of past and present time series to make future predictions [1], [2]. The applications for time series prediction are wide that range from weather [3] to financial prediction [4], [5], [6], [7]. The stock markets are very unpredictable which makes investment very risky and uncertain [6]. Computational intelligence methods such as neural networks have been used to build intelligent financial prediction systems [8], [9].

Computational intelligence methods have shown good performance for financial time series problems. In the past, fuzzy evolutionary model and neuro-evolutionary methods have been used for financial time series problems where they predicted market turning points [10]. Recent work has used support vector regression for financial time series prediction [7], [4], [5]. Evolutionary radial-basis function networks for financial time series prediction for the Taiwan Stock Index [6]. Dual-factor modified fuzzy time-series model has been successfully applied to data from the NASDAQ (National Association of Securities Dealers Automated Quotations) and Taiwan Stock Index [11].

Cooperative Coevolution is a biologically inspired evolutionary computation framework which divides a large problem into subcomponents [12]. Cooperative coevolution has been applied to several different areas which include, training both feed-forward [13], [14] and recurrent neural networks [15] in solving a wide range of problems that include pattern classification, control and time series problems [16], [17], [18]. Coevolutionary methods that train neural networks is known as *cooperative neuro-evolution* [19].

Problem decomposition is a major issue in cooperative coevolution [17]. The problem decomposition method will determine how to break down a problem into subcomponents. The two main problem decomposition methods are *Synapse level* (SL) and *Neuron level* (NL) decomposition. Neuron level decomposition has been used for chaotic time series prediction in the past and has shown promising results with recurrent neural networks [18]. In this paper, Neuron and Synapse level decomposition will be used for training feed-forward neural networks with cooperative coevolution for time series prediction.

The use of smart-phones with real-world mobile applications are becoming increasingly popular as people need access to applications that are portable, connected to the Internet and updated in real-time [20]. Research has been done on the use of mobile phones in medical informatics [21] and several applications have already been developed on different platforms [22] for mobile-health. Recently, a mobile application was developed that provides short-term weather forecasts based on the user's current location by taking advantage of the latest *cloud* technology [23]. A mobile application has been developed that makes uses of machine learning algorithms to recommend interesting spots such as stores and restaurants [24]. Recently a mobile application was developed that made use of neural networks for diagnosing diabetes related problems [25]. The use of mobile technology in foreign exchange trading and its benefits has also been discussed [20]. There is a need for mobile intelligent decision support systems to warn investors about the risks and notify them of potential benefits in real-time.

The use of cooperative neuro-evolution for financial time series prediction has not been thoroughly explored. Cooperative neuro-evolution has been used to train a recurrent neural network for chaotic time series prediction on benchmark chaotic time series problems and provided promising results [18]. The implementation of intelligent prediction systems as software solutions that can be easily accessed has not been possible in the past. Advances in cloud and mobile computing can enable computational intelligence methods that require intensive computation for training. A mobile application using the intelligent prediction system can better disseminate information to the appropriate stakeholders in real-time.

This paper employs cooperative coevolution to train a

Shelvin Chand and Rohitash Chandra are with the School of Computing, Information and Mathematical Sciences, University of the South Pacific, Laucala Campus, Suva, Fiji. (email: {chand.shelvin}@gmail.com), {c.rohitash}@gmail.com).

feed-forward network for a financial time series problem. Synapse level and Neuron level problem decomposition methods are used and their performance is compared. Furthermore, a mobile application is proposed which can be used to disseminate the financial prediction results onto mobile devices. This will allow end-users to make informed financial investment decisions. We used four financial time series problems from the NASDAQ stock exchange [26] to train and test the system.

The rest of the paper is as follows. Details on the proposed system are discussed in Section 2. Section 3 presents the results of the experiment and the implementation of the mobile application. Section 4 concludes the paper with a summary of the results and discussion on future research.

II. Proposed System

A. Cooperative Coevolution for Feed-Forward Networks

Cooperative coevolution divides a large problem into subcomponents. Problem decomposition is a major issue in cooperative neuro-evolution [19]. In Neuron level decomposition, each neuron within the hidden and output layer acts as a reference point for a particular subcomponent and also for the weights connected to it [27]. Synapse level decomposition method decomposes the network to its lowest level where the number of weights determine the number of sub-components. In this paper the two methods will be compared to see which one is more suited for financial time series prediction.

Alg. 1 Cooperative Coevolution

Step 1: Decompose the problem into k subcomponents according to the chosen encoding scheme (Neuron or Synapse level)
Step 2: Initialize and cooperatively evaluate each sub-population
for each *cycle* until termination **do**
  for each Subpopulation **do**
    for n Generations **do**
      i) Select and create new offspring
      ii) Cooperatively evaluate the new offspring
      iii) Update sub-population
    end for
  end for
end for

In Algorithm 1, the feed forward network is decomposed using the Neuron level or Synapse level encoding into *k* subcomponents. In Synapse level, the number of subcomponents (*k*) is equal to the number of weights and biases in the neural network. In Neuron level decomposition (Fig. 1), the number of subcomponents (*k*) is equal to the total number of all the hidden and output neurons [28]. The subcomponents are implemented as sub-populations. Evaluation of the fitness of each individual for a particular sub-population is done cooperatively with the fittest individuals from the other sub-populations [12].

Cooperative evaluation for an individual *j*, in a particular sub-population *i*, is done by concatenating the fittest individuals from the rest of the sub-populations. The concatenated individuals are then transformed into a neural network which is trained and tested on the given problem. The prediction error of the network is then assigned as the fitness for the individual whose fitness was being evaluated, even though it was just a small component of the overall network. All the sub-populations are evolved for a fixed number of generations where new individuals are introduced within the sub-populations. The generalized generation gap model with parent centric crossover operator (G3-PCX) evolutionary algorithm [29] is the designated evolutionary algorithm in the sub-populations as this has given good results in the past [18]. A cycle is completed when all the sub-populations are evolved.

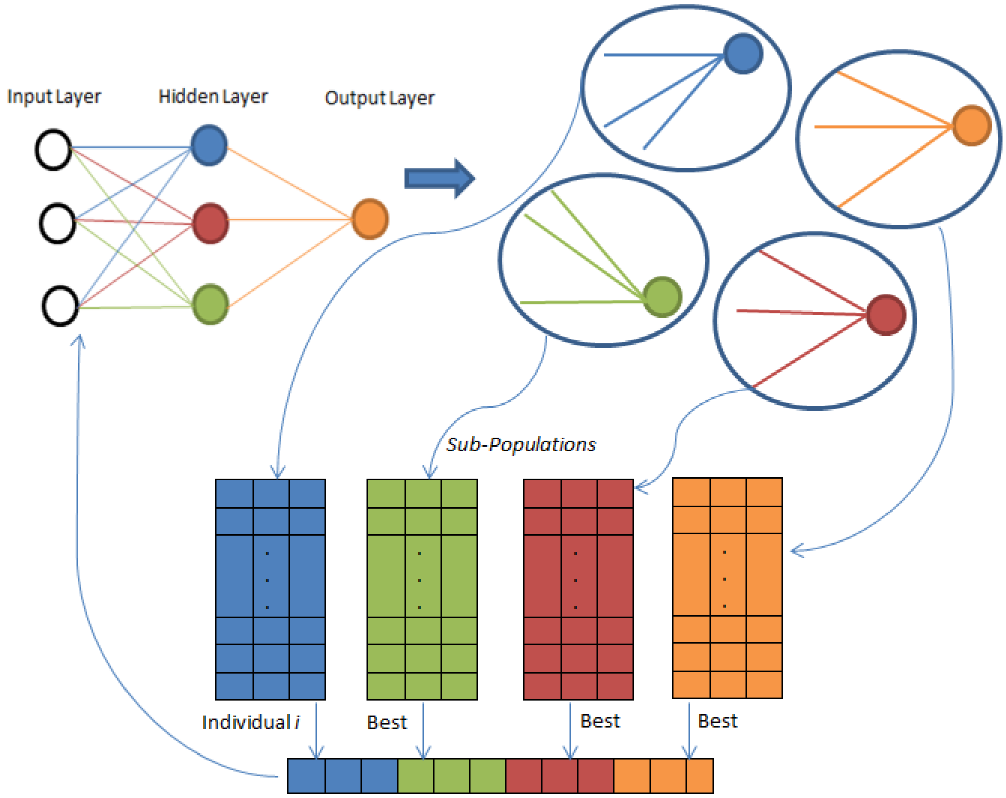
B. Mobile Application for Time Series Prediction

Mobile devices such as *tablets* and *smart phones* are not suitable for running applications with computational intelligence based algorithms that rely on heavy computation because of their limited processing power. Even with quad-core processors, running such algorithms on mobile devices will consume high battery power and produce heat. There has to be prioritization in terms of what data should be transferred or processed on such devices as having large amounts of data processing can considerably slow down such devices [30].

Neural network based android applications have been developed in the past. The authors of [31] used artificial neural networks to build an Android based number plate recognition application. While the application showed great accuracy in prediction, it was not able to function well on devices with low memory and CPU power. In [32], the authors use a back-propagation neural network for recognition of handwritten digits. They were able to port their entire algorithm to an Android application and show good speed and accuracy but the authors did note that some of their methods required extra memory which can be an issue on low-end phones.

Mielke and Bruck [33] proposed an android application for automatic classification of environmental sounds. A local neural network was used for classification. The results showed that the application required high power and computation which meant that it would not be able to run across all devices. Mobile based data-mining was explored by the authors of [34] in which they proposed a Java-based framework to extend data-mining tool *'WEKA'* to Android based smart phones. The authors tested with different data-mining algorithms and found that the computing time on mobiles was greater in comparison to desktops because of the limited processing power on most smart phones.

Intelligent financial prediction systems produce useful data which can be used by investors. We propose an Android based mobile application which can notify potential investors on possible market closing prices for stocks of a particular company. Based on this, the investors can make appropriate investment decisions in real-time. The prediction is made



The visualization of Neuron Level (NL) problem decomposition for feed-forward neural networks. The fitness of an individual *i* from a given sub-population is computed by concatenating the best individuals from the other sub-populations and encoding them into a neural network.

Fig. 1. Neuron-Level Problem Decomposition

by the feedforward neural network on a cloud server. The application is developed keeping in mind the problems discussed in the literature. The application can be seen as a mobile interface for the intelligent prediction system as it is able to fetch prediction results from the intelligent system and present to the user.

In our system, we have all the computation executed on the cloud server and then have the mobile application obtain daily prediction values. The application works fine on any type of Android device regardless of its processing power as long as it supports Android 2.2 or above. The cloud server has more processing power and can produce faster results in comparison to individual mobile devices. This means that the application does not require much computation or battery consumption on the smart device.

The data transfer follows a two-tier architecture. There is a cloud server which holds the intelligent model together with the database, and an Android application on the client side for displaying the appropriate information. At the end of each market day, the closing stock prices are obtained and the neural network is trained again with new data on the cloud server.

III. Experiments and Results

This section presents the results of the experiments done on a real world financial time series data set using cooperative neuro-evolution. The behaviour and results of the model were evaluated by using different number of hidden neurons. The experimental setup and the results are given in the following sub-sections.

A. Data Set

The financial time series data set is taken from the NASDAQ stock exchange [26]. It contains daily closing prices for ACI Worldwide Inc., Intel Corporation, Staples Inc. and Seagate Technology PLC from December 2006 to February 2010. Three of the four companies, namely Intel Corporation, Staples Inc. and Seagate Technology PLC, are listed on he NASDAQ-100 which is a stock market index for the 100 largest non-financial companies listed on the NASDAQ stock exchange.

The closing stock prices were normalized between the range of 0 to 1. In all four companies, the stock prices fluctuate over the 3 year period. The dataset also overlaps with the recession that hit the US market. This allows us to evaluate how the model adapts and accounts for the different market forces. The four data sets were chosen randomly. The

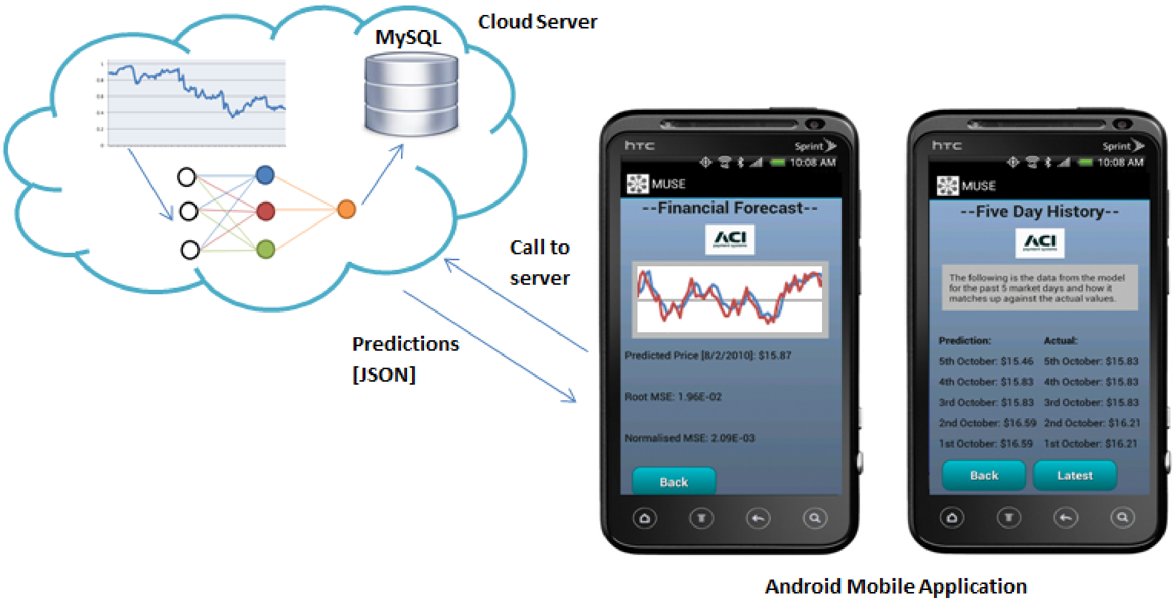


Fig. 2. The diagram shows the interaction between the cloud server(left) and the Android application (right).

financial data is first divided into training and testing sets using a 50-50 split.

B. Reconstructing the Data Set

Taken's embedding theorem [35] was used to reconstruct the data before it was used by the model. It allows for chaotic time series data to be reconstructed into a state space vector with the two conditions of *time delay (T)* and *embedding dimension (D)* [35]. The value of D and T must be carefully chosen in order for the vector to be able to reproduce important characteristics of the original data set [36]. In this paper, we used D=5 and T=2. These values were carefully chosen after carrying out a set of trial experiments.

A time lapse of *2* allowed the training and test dataset to retain the pattern and characteristics of the original data set. A greater time lapse could have meant missing out on some of the key information on the market behaviour. D=5 allowed the algorithm to learn better, having a good set of historical points in each prediction, which increased the probability of getting an accurate prediction.

C. Experimental Setup

The feedforward neural network employs sigmoid units in the hidden layer and the output layer. Root mean squared error (RMSE) is used to evaluate the performance of the proposed method [18].

The maximum number of function evaluations was set at 50 000 which was used as a termination condition for the algorithm. The G3-PCX algorithm uses the *generation gap model* [29] for selection in which we put a pool size of 2 parents and 2 offsprings as done in literature [18]. The G3-PCX evolutionary algorithm is used to evolve all the sup-populations as it has shown good results with cooperative neuro-evolution in the past [19]. We ran trial experiments and found that population size of 300 gave the best results and hence we used it in our experiments. Neuron level and Synapse level problem decomposition methods were used for dividing the problem into subcomponents.

The *depth of search* which represents the number of generations for each subpopulation is kept at 1 as this number has achieved optimal results in our past work [18]. This indicates that all the sub-populations in cooperative coevolution are evolved in a round-robin fashion for a single generation. The algorithm terminates once the maximum number of function evaluations has been reached.

D. Results: Financial Time Series Prediction

This section reports on the performance evaluation for training a feedforward neural network on the financial time series problem with Synapse (SL) and Neuron level (NL) decomposition methods. The results for 50 experimental runs with 95% confidence are given in Tables I, II, III and IV with the best results highlighted in bold.

The results for ACI Worldwide Inc. given in Table I are better in comparison to the results for the other three companies in terms of generalization error. The Mean RMSE for Neuron level was lower compared to Synapse level for all cases given the different number of hidden neurons. The general trend for both NL and SL was that as the number of hidden neurons increased, the error rate decreased. In this data set, NL decomposition was able to give a more stable performance in comparison to SL. It should also be noted that ACI Worldwide is the only company in our list which does not belong to the NASDAQ-100 Index, and therefore, its prices are expected to be more stable.

The results for Intel Corporation (Table II) were similar to that of ACI Worldwide, as NL outperformed SL. Neuron level decomposition produced better results in comparison to Synapse level decomposition with lower RMSE. NL gave the best performance with three hidden neurons and as the number of hidden neurons increased, its performance deteriorated. The performance of SL decomposition improved as the number of hidden neurons increased up-till an optimal number of seven hidden neurons. Intel ranks the highest of the three companies that were chosen from the NASDAQ-100 Index.

In the case of Staples Inc. (Table III) data set, Synapse level decomposition outperformed Neuron level decomposition. SL gave better overall generalization in comparison to NL for the different number of hidden neurons. In NL, as the number of hidden neurons increased, the generalization performance deteriorated while for SL, at first, the performance improved as the number of hidden neurons increased but after reaching an optimal level of seven hidden neurons, the performance deteriorated. We observed that this problem was more challenging for our algorithm in comparison to both the previous problems.

We observed that the Seagate Technology PLC. (Table IV) data set was the most difficult to predict. Similar to the first two companies, NL again outperformed SL in terms of generalization. Both decomposition methods had difficulty in predicting this data set as the generalization performance for both is not as good when compared to the results of ACI Worldwide or Intel. The performance for NL and SL fluctuated with the different number of hidden neurons giving the most optimal result with five hidden neurons.

TABLE I

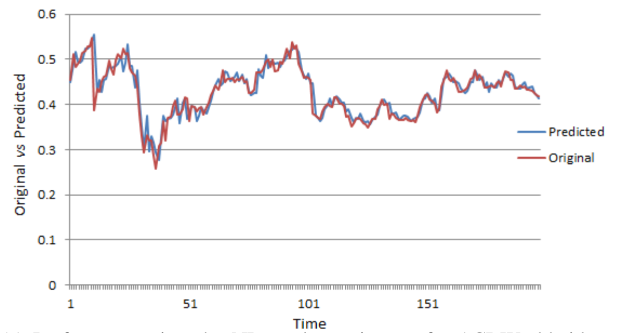
THE PREDICTION TRAINING AND GENERALIZATION PERFORMANCE (RMSE) OF NL AND SL FOR THE ACI WORLDWIDE FINANCIAL DATA.

Prob.	H	Training	Generalization	Best
FNN-NL	3	0.0208 ± 0.0004	0.0213 ± 0.0004	**0.0191**
	5	0.0202 ± 0.0004	0.0209 ± 0.0003	0.0194
	7	0.0198 ± 0.0004	0.0208 ± 0.0003	0.0195
	9	**0.0197 ± 0.0003**	**0.0207 ± 0.0002**	0.0196
FNN-SL	3	0.0474 ± 0.0023	0.0483 ± 0.0042	0.0299
	5	0.0297 ± 0.0016	0.0258 ± 0.0017	**0.0192**
	7	0.0268 ± 0.0008	**0.0246 ± 0.0011**	0.0200
	9	**0.0266 ± 0.0007**	0.0247 ± 0.0009	0.0195

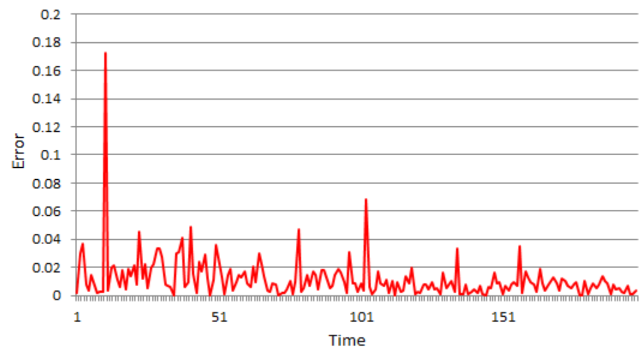
TABLE II

THE PREDICTION TRAINING AND GENERALIZATION PERFORMANCE (RMSE) OF NL AND SL FOR THE INTEL FINANCIAL DATA.

Prob.	H	Training	Generalization	Best
FNN-NL	3	0.0186 ± 0.0170	**0.0633 ± 0.0116**	0.0178
	5	0.0183 ± 0.0182	0.0637 ± 0.0131	**0.0180**
	7	**0.0181 ± 0.0229**	0.0792 ± 0.0156	0.0182
	9	0.0182 ± 0.0287	0.0989 ± 0.0179	0.0192
FNN-SL	3	0.0340 ± 0.0023	0.1465 ± 0.0107	0.0516
	5	0.0262 ± 0.0010	0.0991 ± 0.0136	0.0245
	7	**0.0243 ± 0.0007**	**0.0824 ± 0.0130**	0.0217
	9	**0.0243 ± 0.0007**	0.0891 ± 0.0156	**0.0204**

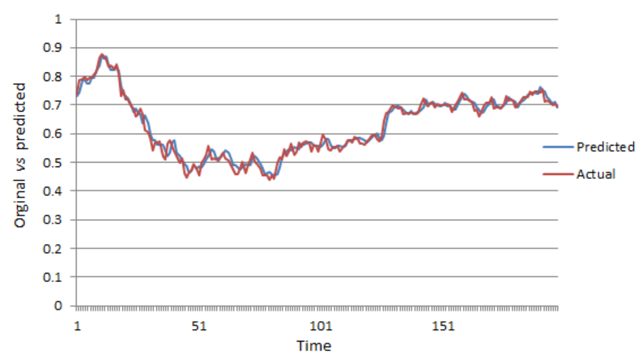


(a) Performance given by NL on the testing set for ACI Worldwide.

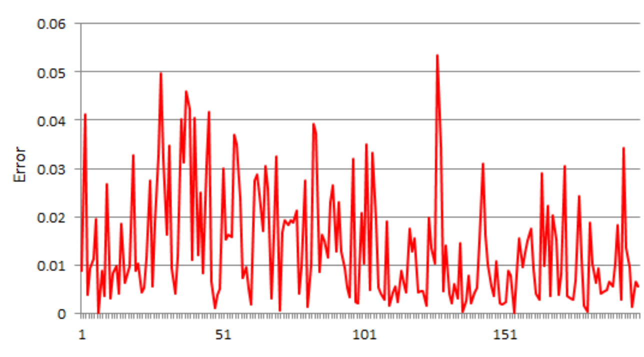


(b) Error on the test data set given by NL for the ACI Worldwide time series.

Fig. 3. Typical prediction given by NL for ACI Worldwide time series.



(a) Performance given by NL on the testing set for Intel Corp.



(b) Error on the test data set given by NL for the Intel Corp. time series.

Fig. 4. Typical prediction given by NL for Intel Corp. time series.

TABLE III

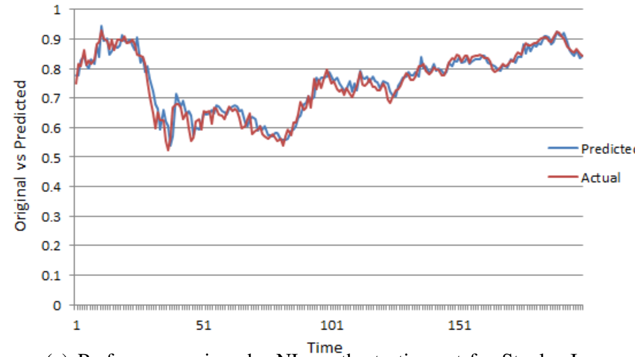
THE PREDICTION TRAINING AND GENERALIZATION PERFORMANCE (RMSE) OF NL AND SL FOR THE STAPLES INC. FINANCIAL DATA.

Prob.	H	Training	Generalization	Best
FNN-NL	3	0.0172 ± 0.0202	**0.0802** ± **0.0102**	0.0301
	5	0.0171 ± 0.0226	0.0883 ± 0.0110	0.0353
	7	**0.0167** ± **0.0257**	0.0977 ± 0.0124	0.0276
	9	0.0168 ± 0.0285	0.1082 ± 0.0129	**0.0271**
FNN-SL	3	0.0341 ± 0.0274	0.1264 ± 0.0099	0.0444
	5	0.0275 ± 0.0226	0.0983 ± 0.0113	0.0260
	7	0.0254 ± 0.0176	**0.0757** ± **0.0107**	**0.0249**
	9	**0.0237** ± **0.0213**	0.0892 ± 0.0111	0.0259

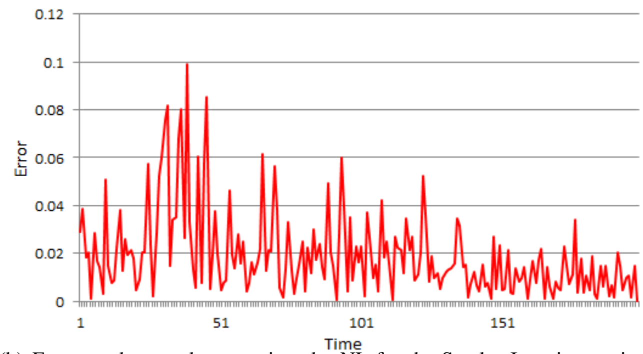
TABLE IV

THE PREDICTION TRAINING AND GENERALIZATION PERFORMANCE (RMSE) OF NL AND SL FOR THE SEAGATE FINANCIAL DATA.

Prob.	H	Training	Generalization	Best
FNN-NL	3	0.0188 ± 0.0620	0.2015 ± 0.0359	0.0373
	5	0.0186 ± 0.0483	**0.1609** ± **0.0278**	0.0458
	7	0.0185 ± 0.0610	0.1948 ± 0.0365	0.0274
	9	**0.0184** ± **0.0606**	0.1944 ± 0.0359	**0.0224**
FNN-SL	3	0.0365 ± 0.0512	0.2179 ± 0.0097	0.1197
	5	0.0284 ± 0.0543	**0.1956** ± **0.0283**	**0.0374**
	7	0.0257 ± 0.0743	0.2316 ± 0.0476	0.0439
	9	**0.0237** ± **0.0973**	0.3229 ± 0.0509	0.0593



(a) Performance given by NL on the testing set for Staples Inc.



(b) Error on the test data set given by NL for the Staples Inc. time series.

Fig. 5. Typical prediction given by NL for Staples Inc. time series.

E. Prototype for Mobile Application

The intelligent financial prediction system is made up of two main components which include the intelligent prediction system and the mobile application. The cooperative neuro-evolutionary method was programmed using C++ and executed on the cloud server. Each day it is trained with new data after which it outputs new predictions which are updated in a MySQL database that runs on the same server. Crons jobs are used to update the new market data and to automatically train the model at a specific time each day. Any regular cloud service with appropriate processing capability should be sufficient in this case.

The Android application synchronizes the prediction data from the server through a HTTP server request using a PHP (Hypertext Preprocessor) script which queries the database. The database is able to easily maintain a history of all the data generated from the model at different times. The application does not query the database each time it is loaded. It, however, does check with the cloud server if any changes have been made or any new data has been added and then appropriately queries the database if need be.

The Android application displays the previous days prediction and actual price together with the prediction for the present day. It also displays the accuracy at which the prediction was made so that users are aware of the risk factor. This will be updated each day depending on the results from the intelligent prediction system. The user can also view the prediction data from the past week in comparison to the actual prices on a separate view in the Android application to see how accurate the system was in the past week. The objective is to prioritize in terms of what information to send to the mobile device so that it is not overloaded with large amounts of data.

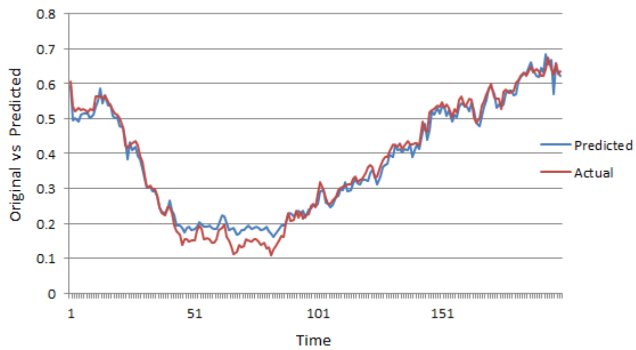
The intelligent system is independent of the Android application architecture. The Android application is written using Java and the front-end is designed using responsive Extensible Markup Language (XML) format so that the application can easily work on any device including *tablets* and *smart phones*. The application was developed for Android 2.2 and above which covers a larger number of devices supporting different versions of Android, both new and old.

The application was tested using an Android Simulator and later an actual Android based mobile device. It was tested on speed and functionality using a standard 3G mobile network. The application did not show any lag and was able to retrieve the required information quickly and efficiently.

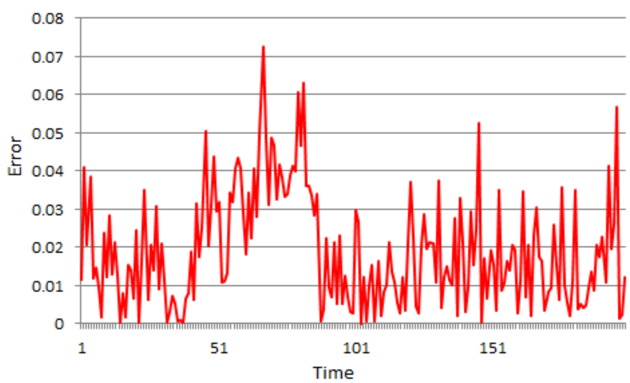
F. Discussion

The cooperative neuro-evolutionary method was earlier applied to bench mark data sets (Mackey-Glass, Lorenz and Sunspot) where in general, Neuron level decomposition out performed Synapse level decomposition [18]. The financial data is a real world data set which contains noise elements and despite this the accuracy level was high in terms of generalization. NL method out-performed the SL method with the financial time series problem.

The results show that Neuron level decomposition is more suited to a financial time series problem in comparison to



(a) Performance given by NL on the testing set for Seagate Technology PLC.



(b) Error on the test data set given by NL for the Seagate Technology PLC. time series.

Fig. 6. Typical prediction given by NL for Seagate Technology PLC. time series.

Synapse level decomposition. Neuron level decomposition is able to efficiently decompose the problem by grouping interacting variables into separate sub-components which reduces interaction between the sub-components during evolution [17]. This allows Neuron level to give a better training performance in comparison to Synapse level decomposition. The neural network training performance affects generalization performance.

The mobile application is able to instantly get the latest financial prediction information as soon as the intelligent system makes an update to the database. It is also suitable for low bandwidth scenarios as the data generated by the cooperative neuro-evolutionary method is summarized before being transfered over the network. The application does not require much processing power on the client side as the main model is executed on the cloud server. This application can guide investors in the right direction while making important financial decisions. The framework can also be used for any similar intelligent method.

An alternative approach for the overall system would be to train the neural network on the cloud server and then transfer the weights for the trained neural network to the mobile device. The mobile application will receive the weights and apply it to the local neural networks built within the application. As a result, each day the application will fetch the latest market prices from the cloud server and feed the prices into its local neural network which will output predictions.

In the future, the same system framework can be employed for *smart-phones* and *tablets* that make use of the Apple platform (iOS). Using push notification to inform the users about the latest prediction data can also be explored where the server will initiate the request for an update.

IV. Conclusion

This paper presented an application of cooperative cov-evolution with feed-forward neural networks to financial time series prediction. The results for Neuron level and Synapse level problem decomposition methods were compared. Neuron level out-performed Synapse level in terms of generalization performance for 3 of the 4 data sets. This indicates that Neuron level decomposition is able to better decompose the problem into subcomponents. A mobile application was also proposed which can be used for updating important information from the intelligent prediction system to the end-users. This can allow investors to make informed financial decisions while on the move.

In future research, the cooperative neuro-evolutionary model can be applied to different financial data sets dealing with foreign exchange rates, interest rates, dividend rates and evaluate the performance. Moreover, it would be interesting to know how other neural network architectures would perform on these data sets, such as recurrent neural networks.

References

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

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

[3] E. Lorenz, *The Essence of Chaos*. University of Washington Press, 1993.

[4] H. Jiang and W. He, "Grey relational grade in local support vector regression for financial time series prediction," *Expert Systems with Applications*, vol. 83, pp. 136–145, 2012.

[5] B. Wang, H. Huang, and X. Wang, "A novel text mining approach to financial time series forecasting," *Neurocomputing*, vol. 83, pp. 136–145, 2012.

[6] H. Feng and H. Chou, "Evolutional rbfns prediction systems generation in the applications of financial time series data," *Expert Systems with Applications*, vol. 38, pp. 8285–8292, 2011.

[7] X. Liang, R. Chen, Y. He, and Y. Chen, "Associating stock prices with web financial information time series based on support vector regression," *Neurocomputing*, vol. 115, pp. 142–149, 2013.

[8] C. Chen, "Neural networks for financial market prediction," in *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on*, vol. 2, 1994, pp. 1199–1202.

[9] H. Lin and A. Chen, "Application of dynamic financial time-series prediction on the interval artificial neural network approach with value-at-risk model," in *Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, 1994, pp. 3918–3925.

[10] A. Azzini, C. Pereira, and A. Tettamanzi, "Predicting turning points in financial markets with fuzzy-evolutionary and neuro-evolutionary modeling," *Applications of Evolutionary Computing*, pp. 213–222, 2009.

[11] H. Jiang and W. He, "Fuzzy dual-factor time-series for stock index forecasting," *Expert Systems with Applications*, vol. 36, pp. 165–171", 2009.

[12] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *PPSN III: Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature*. London, UK: Springer-Verlag, 1994, pp. 249–257.

[13] ——, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evol. Comput.*, vol. 8, pp. 1–29, 2000.

[14] N. García-Pedrajas and D. Ortiz-Boyer, "A cooperative constructive method for neural networks for pattern recognition," *Pattern Recogn.*, vol. 40, no. 1, pp. 80–98, 2007.

[15] F. Gomez and R. Mikkulainen, "Incremental evolution of complex general behavior," *Adapt. Behav.*, vol. 5, no. 3-4, pp. 317–342, 1997.

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

[17] 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.

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

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

[20] C. Srensen and A. Al-Taitoon, "Organisational usability of mobile computingvolatility and control in mobile foreign exchange trading," *International Journal of Human-Computer Studies*, vol. 66, pp. 916–929, 2008.

[21] P. Klasnja and W. Pratt, "Healthcare in the pocket: Mapping the space of mobile-phone health interventions," *Journal of Biomedical Informatics*, vol. 45, pp. 184–198, 2012.

[22] C. Liu, Q. Zhu, K. Holroyd, and E. Seng, "Status and trends of mobile-health applications for ios devices: A developers perspective," *The Journal of Systems and Software*, vol. 84, pp. 2022–2033, 2011.

[23] D. Krishnappa, D. Irwin, E. Lyons, and M. Zink, "Cloudcast: Cloud computing for short-term mobile weather forecasts," *2012 IEEE 31st International Performance Computing and Communications Conference (IPCCC)*, pp. 61–70, 2012.

[24] Y. Omori, Y. Nonaka, and M. Hasegawa, "Design and implementation of a context-aware guide application for mobile users based on machine learning," *Lecture Notes in Computer Science*, vol. 6279, pp. 271–279, 2010.

[25] O. Karan, C. Bayraktar, H. Gmskaya, and B. Karlik, "Diagnosing diabetes using neural networks on small mobile devices," *Expert Systems with Applications*, vol. 39, pp. 54–60, 2012.

[26] Inforchimps. Nasdaq exchange daily 1970-2010 open, close, high, low and volume. [Online]. Available: http://www.inforchimps.com/datasets/nasdaq-exchange-daily-1970-2010-open-close-high-low-and-volume

[27] 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.

[28] R. Chandra, M. Frean, and M. Zhang, "An encoding scheme for cooperative coevolutionary feedforward neural networks," in *AI 2010: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2010, vol. 6464, pp. 253–262.

[29] 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.

[30] C. Wang, W. Duan, J. Ma, and C. Wang, "The research of android system architecture and application programming," *2011 International Conference on Computer Science and Network Technology (ICCSNT)*, vol. 2, pp. 785–790, 2011.

[31] A. Mutholib, T. S. Gunawan, and M. Kartiwi, "Design and implementation of automatic number plate recognition on android platform," *International Conference on Computer and Communication Engineering (ICCCE 2012)*, pp. 540–543, July 2012.

[32] Z. Dan and C. Xu, "The recognition of handwritten digits based on bp neural network and the implementation on android," *2013 Third International Conference on Intelligent System Design and Engineering Applications*, pp. 1498–1501, 2013.

[33] M. Mielke and R. Brck, "Smartphone application for automatic classification of environmental sound," *MIXDES 2013, 20th International Conference on Mixed Design of Integrated Circuits and Systems*, pp. 512–515, 2013.

[34] P. Liu, Y. Chen, W. Tang, and Q. Yue, "Mobile weka as data mining tool on android," *Advances in Electrical Engineering and Automation*, vol. 139, pp. 75–80, 2012.

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

[36] 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.