Multiple-Model Fusion for Robust Time-Series Forecasting

Weizhong Yan

***Abstract*—This paper presents/summarizes techniques and results for the Neural Network & Computational Intelligence Forecasting Competition organized by S. Crone and supported by SAS and IIF. The competition involves forecasting a large set of time series using methods of neural networks and computational intelligence technologies. The primary goal of the competition is to identify the best strategies in developing neural network models for forecasting a large number of time series. In this paper, a multiple-model fusion scheme is examined from a point of view of forecasting a large number of time series and for the purpose of achieving more robust forecasting performance.**

I. Introduction

Despite over 15 years of research and more than 2000 publications on artificial neural networks for forecasting, neural networks have not yet been established as a valid and reliable forecasting method in time series forecasting, especially in situation where a large number of time series are involved [1]. The primary purpose of the competition is thus to identify the "best practice" methodologies of neural networks for time series forecasting by evaluating a set of consistent neural network methodologies across a representative set of time series.

Conventional design for time series forecasting models focuses on identifying the "single best" forecasting model from a collection of different models. Recent studies have demonstrated that for most real-world problems, the single model design often time cannot be designed to achieve the desired performance. And multiple models fusion, by leveraging complimentary information from different models can be more effective in achieving more robust forecasting performance. As a result, multiple models systems have gain more and more research interests in recent years. For example, neural network ensembles were used for weather forecasting [2] and for economic forecasting [3].

In the ISF05 time series competition, the author used neural network ensemble for forecasting. For the 2 time series data provided in the competition, neural network ensemble showed promising results. In this paper, we extended neural network ensemble to multiple level model fusion, where a series of generalized regression neural networks are used for time-series forecasting. In addition, we focus our effort on designing a multiple model forecasting system in an automatic fashion so that the model building process can be applied to a large number of time series.

The reminder of the paper is organized as follows. Section II describes the datasets used for the competition. Characteristics of the dataset and the challenges of forecasting are also highlighted. Our multiple-model forecasting scheme is presented in Section III. Forecasting results are given in Section IV. Section V concludes the paper.

Dr. Weizhong Yan is with GE Global Research Center, Niskayuna, NY 12309, USA. phone: 518-387-5704; fax: 518-387-6104; e-mail: yan@ crd.ge.com.

II. Problem Descriptions

A. *The datasets*

Two datasets are provided for competition. Dataset A is a complete dataset consisting of 111 monthly time series drawn from homogeneous population of empirical business time series. Dataset B, on the other hand, is a subset of Dataset A, which consists of 11 time series. Participants can submit their results for either or both of the datasets. The time series in the two datasets have different length of historical data points. Generally speaking, Dataset A has about 5 years, 60 months of data points, while Dataset B has about 10 years, 120 data points. All of the series start at January. For each of time series, we are required to forecasting 18 months of values based on the given historical data points.

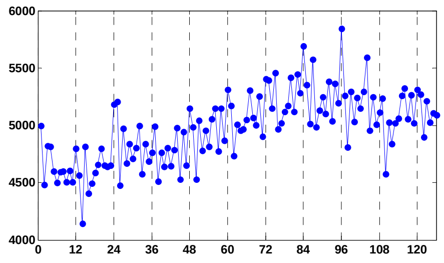

Fig. 1 – Time plot of the time series sample 1

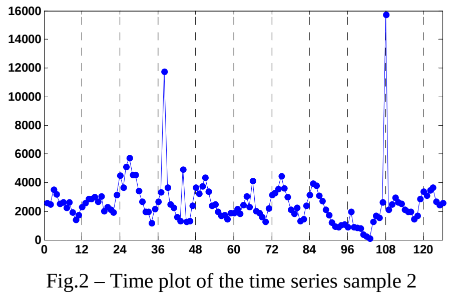

Fig.2 – Time plot of the time series sample 2

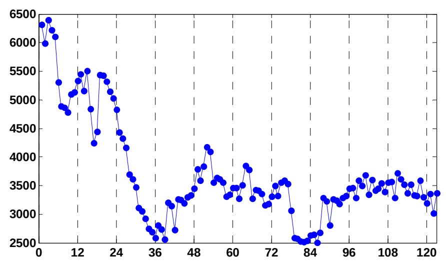
Fig.3 – Time plot of the time series sample 3

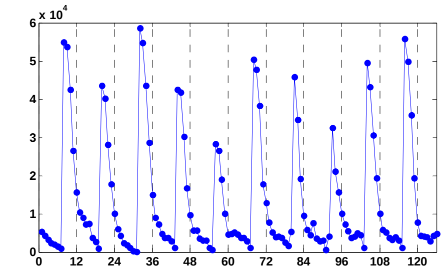
Fig.4 – Time plot of the time series sample 4

Figures 1 thru 4 show samples of the time series from Dataset B. From Figures 1 thru 4, we can make following observations about the time series in the datasets.

- Series are generally seasonal - some stronger than other (Fig.4 for example).
- Most series contain a certain level of trend (e.g., Figs. 1 and 4). The trend may not necessarily be linear.
- Series may contain outliers (Fig.2 for example).
- Data in a series may not be normally distributed. Fig. 4, for example, shows a skew data distribution.
- Almost all time series are non-stationary, i.e., with varying mean, or variance, or both.

B. *Challenges and modeling strategies*

The above-mentioned observations for the series are not particularly unusual for real-world time series. There are known methods, for example, time plots, for identifying the features of a time series. However, most of these feature identification methods are manual and ah-hoc, thus an automation of these methods is needed for the competition so that they can be efficient in dealing with a large number of time series. There are also text book approaches, e.g., the Boc-Cox transformation, for treatment of the features (trend and seasonality). However, the effectiveness of these treatment measures is highly dependent on the time series. That is, no single treatment measure works well for all time series. One of the challenges we are facing in this competition is thus on how to adopt the forecasting modeling process that has been successfully used for single or small number of time series to a situation where a large number of time series are involved. The strategies for achieving superior forecasting performance in the competition is to develop a so-called automatic forecasting method that is "adaptive" to different time series such that a robust forecasting performance is always achieved for all time series.

III. Our Approach

Geared towards developing an automated forecasting scheme that not only gives superior forecasting performance, but also works effectively for a large number of time series, in model building we focus on two equally important parts, preprocessing and modeling. These two parts are explained as follows.

A. *Preprocessing*

Preprocessing of time-series includes feature identification and feature treatment. Features of a time series refer to trend, seasonality, outliers, and discontinuities [Chatfield (2004)]. These features, especially outliers and discontinuities, complicate modeling and negatively affect forecasting performance. Correctly identifying these features so that a proper treatment measure can be taken is crucial in time series modeling and forecasting. Feature identification in our system is to automatically determine if a time series contains such features based on historical observations. If any of the afore-mentioned features is determined to be present in a series, treatment of such features is required. The treatment includes modifying outliers, imputing missing observations, and removing or modeling trend and seasonality effects. In this study, we focus on outlier removal and global trend detection. More specifically we like to have a general scheme that not only can automatically identify and treat the features (outlier and trend), but also is applicable for all time series concerned.

We define an outlier as the point whose value is great than 4 times of the median of the 3 consecutive points before and after. That is, x_i is an outlier if

$$x_i \geq 4 \cdot median(x_{i-3}, x_{i-2}, x_{i-1}) \text{ and}$$
$$x_i \geq 4 \cdot median(x_{i+1}, x_{i+2}, x_{i+3})$$

Once an outlier is found, we replace its value with the average value of the points that are immediately before and after the outlier.

In this study we are interested in identify large trends (we call them "global trends"). For example, the sample time series shown in Figures 1 & 3 are considered to have global trends while the sample time series shown in Figure 2 & 4 are not. To determine if a time series has a global trend, we split the series into a series of 12-point-window segments and calculate means and standard deviations of these segments. Let the means and the standard deviations of n segments be $\mathbf{m} = \{m_1, m_2, ... m_n\}$ and $\boldsymbol{\sigma} = \{\sigma_1, \sigma_2, ... \sigma_n\}$. We define $GTI = \frac{(\max(\mathbf{m}) - \min(\mathbf{m}))}{\frac{1}{n}\sum_{i=1}^{n}\sigma_i}$ as the global trend index. If $GTI \geq 3$, the time series has global trend. Otherwise it does not.

We take different strategies for time series with and without global trends in model building. For time series without global trend, we directly use the raw values as model inputs. On the other hand, we take *first difference* for time series with global trend as the simplest way to remove the trend.

It is worthy pointing out that even though we only take actions towards outliers and trend, our preprocess step is fully automated and can be applied to all 111 time series without human intervention.

We don't take particular treatment for seasonality in preprocessing step. We take care of the seasonality effort in modeling step.

B. *Modeling*

We propose a multiple-model fusion based forecasting system. In our forecasting system, there are 2 levels of multiple-model fusions. In the first level, we use 18 models, each of which is to perform one of 18 out-of-sample predictions. That is, the i^{th} model performs *i*-step-ahead forecasting, where *i* =1,2, … 18. Figure 5 illustrates how the 18 models work.

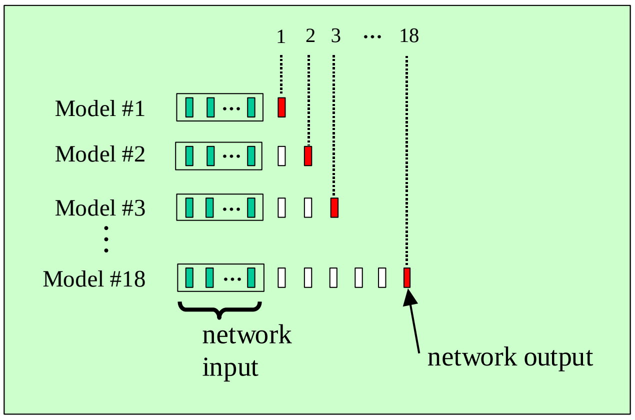

Figure 5: The first level of multiple models (18 models for 1-to 18-step ahead prediction)

We use the generalized regression neural network (GRNN) for our base model. GRNN is a universal approximator that can approximate a continuous function to an arbitrary accuracy, given a sufficient number of neurons. As shown in Figure 6, a typical GRNN has two layers of artificial neurons. While the first layer consists of radial basis neurons the second layer consists of neurons with a linear transfer function. First layer weights are simply the transpose of input vectors from the training set. The weighted input for the first layer neurons is the distance between the input vector and its weight vector. The neuron's output is the radial basis function of the input scaled by the spread factor. Therefore, if a neuron weight is equal to the input vector, the distance between the two is zero, which gives an output of 1. This type of neuron gives an output characterizing the closeness between input vectors and weight vectors (training inputs).

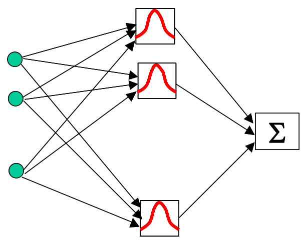

Figure 6: A typical GRNN architecture

A GRNN has one tunable parameter, *spread factor*. When the spread factor is small the radial basis function is steep so that the neuron with the weight vector closest to the input will have a much larger output that other neurons. The network tends to respond with the target vector associated with the nearest design input vector. As the spread factor increases, the radial basis function's slope gets smoother and several neurons may respond to an input vector. The network then acts like it is taking a weighted average between target vectors whose design input vectors are closest to the new input vector. As spread factor gets larger and larger, more and more neurons contribute to the average with the result that the network function becomes smoother.

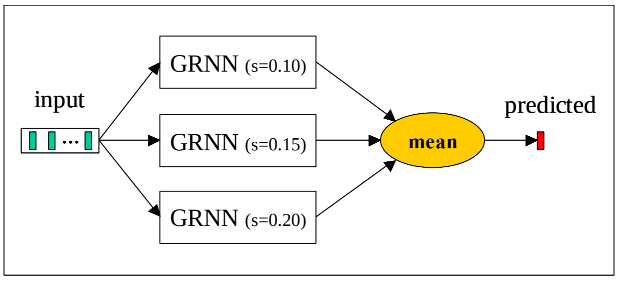

Figure 7: The second level of multiple model fusion -For each of the 18 models in the first level, three individual GRNNs with different spread factors are used and the predictions of these individual GRNNs are fused

As can be seen from above, spread factor of GRNN is one of the important factors affecting the prediction performance of GRNN. Unfortunately there is no unique spread factor that works for all problems. Moreover, there is no analytical way to accurately determine the spread factor. For a given problem, experimental determination of spread factor is an important part of design efforts of GRNN. For time series forecasting concerned in this study, since we are dealing with a large number of time series, we can't afford to do experiments on determining the best spread factor for each individual time series. We could include an optimizer (e.g., GA) as a wrapper to determine the optimal spread factor for each individual time series. The objective function of the optimization could be the performance measure, i.e., the Symmetric mean absolute percentage error (SMAPE) of the testing samples. However, due to time limitation, in this submission, we do not adopt the optimization scheme for determining the spread factor. Instead, for each of the 18 models we use three GRNNs, each of which uses one of the three pre-defined spread factors, 0.10, 0.15, and 0.20. The prediction is then the average of the outputs of the three networks. Figure 7 illustrates the concept. Our thinking here is that leveraging different spread factors would be our best strategy in achieving overall good performance for all 111 time series concerned without identifying specific spread factor for each individual time series. Certainly the leveraging strategy we adopt here would give us a less superior prediction performance comparing to using optimization scheme. In future work, we would like to explore the optimization schemes to see how much improvement we can gain in terms of prediction performance.

Inputs to all GRNNs are kept the same, that is, the past consecutive 12 points, x_{t-1}, x_{t-2},…, x_{t-12}, for all 111 time series. Depending on weather or not the time series contains global trend, xs can be raw values or the first difference (see III-A for details).

For each time series, we reserve last 18 points for testing and the remaining points for training set. We calculate the Symmetric mean absolute percentage error (SMAPE) for out-of-sample prediction of the 18 testing points. And we use the SMAPE to gauge the model performance for predicting the unknown 18 points.

IV. Results

Figures 8 thru 11 show some forecasting results for sampled time series.

V. Conclusion

In this paper, we presents/summarizes techniques and results for the Neural Network & Computational Intelligence Forecasting Competition organized by S. Crone and supported by SAS and IIF. Geared towards developing an automated forecasting scheme that not only gives superior forecasting performance, but also works effectively for a large number of time series, we propose a multi-level model fusion scheme for time-series forecasting.

References

[1] http://www.neural-forecasting-competition.com/motivation.htm

[2] B. Zhu and J. Lin, "Principal component analysis and neural network ensemble based economic forecasting", Proceedings of Chinese Control Conference, Harbin, China, Aug., 2006, pp1769-72,

[3] G.P. Zhang and V.L. Berardi, "Time series forecasting with neural network ensembles: an application for exchange rate prediction", Journal of the Operational Research Society, Vo. 32, No.6, June 2001, pp652-664.

[4] M. Mohandes, "Support vector machine for short-term electrical load forecasting", International Journal of Energy Research, Vol. 26, No. 4, pp335-345, 2002

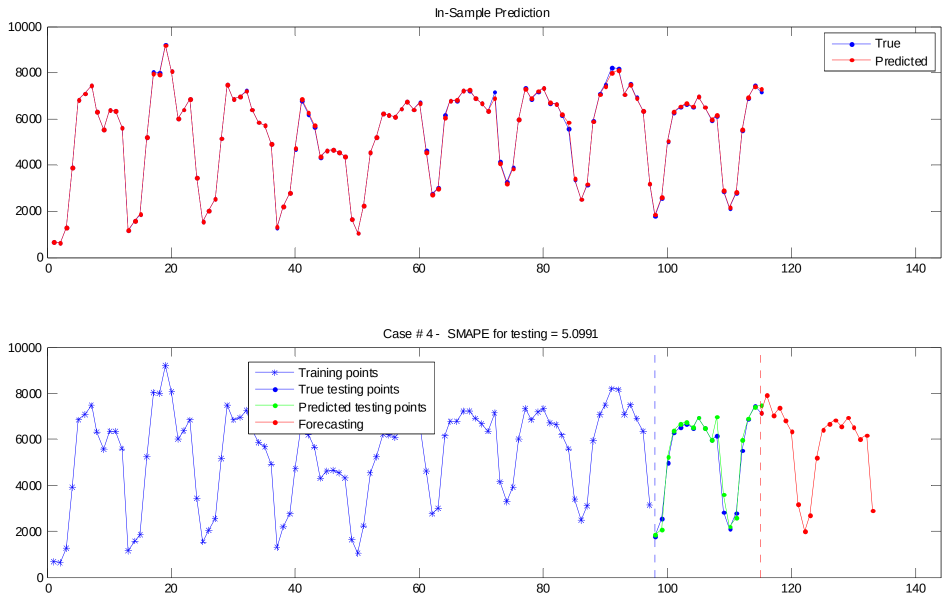

Figure 8: Sample prediction result – case # 4

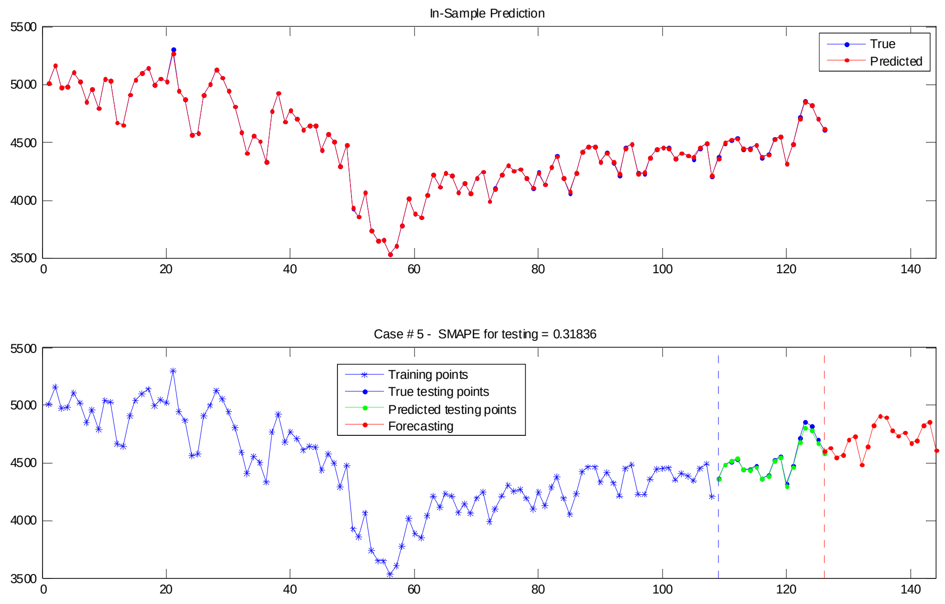

Figure 9 : Sample prediction result – case # 5

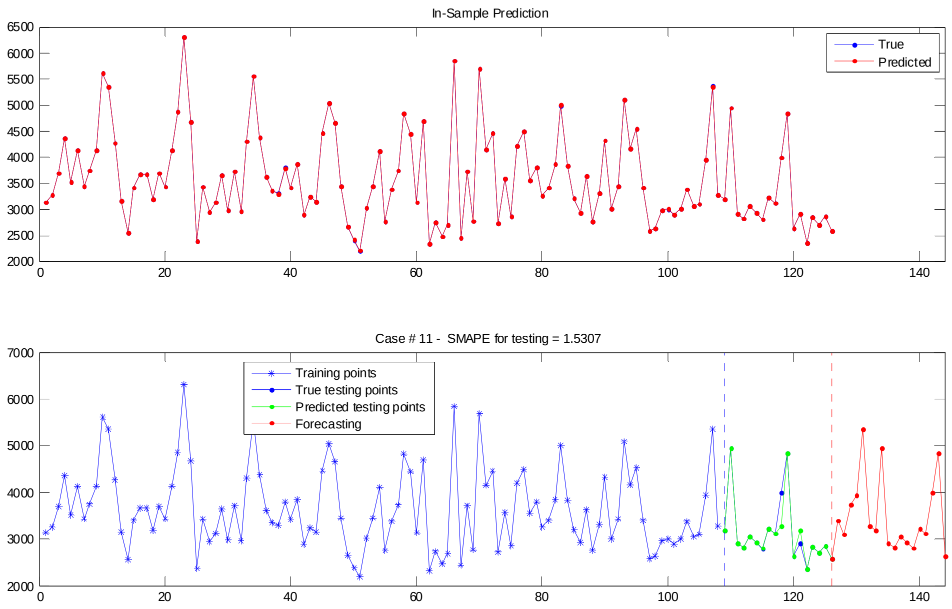

Figure 10 : Sample prediction result – case # 11