Proceedings of International Joint Conference on Neural Networks, Orlando, Florida, USA, August 12-17, 2007

Forecasting Seasonal Time Series with Neural Networks: A Sensitivity Analysis of Architecture Parameters

Sven F. Crone, Rohit Dhawan

Abstract—**Neural Networks are widely applied in time series forecasting. However, no consensus exists on their capability of forecasting seasonal time series. As seasonal patterns frequently occur in empirical time series, it is imperative to establish their efficacy in forecasting seasonality. This paper seeks to evaluate the usefulness of multilayer perceptrons in forecasting time series with different forms of seasonal and trend components. Using eight synthetic time series, we systematically evaluate the impact of different combinations of hidden nodes, input nodes and activation functions on the distribution of the forecasting errors. We aim to a) establish the sensitivity of different architectural choices for neural networks in forecasting and b) analyze the relative accuracy of one or multiple neural network architectures as forecasting methods for seasonal time series. Results are presented in order to guide future selection of network parameters. We find that neural networks show sensitivity to selected architecture decisions but generally provide a robust and competitive forecasting performance on seasonal data.**

I. Introduction

SEASONAL fluctuations are commonly observed in quarterly and monthly economic time series, with multiple overlying seasonality occurring in weekly, daily and hourly data. As seasonality is a dominant feature in time series [1, 2], economists have developed methodologies to routinely deseasonalise data for modelling and forecasting. In contrast, alternative modelling approaches using neural networks (NN) frequently model seasonality directly to reflect non-deterministic [3] or dynamically changing [4] seasonal components, where static seasonal adjustments may incur additional problems [5, 6]

NN are capable of semi-parametric, non-linear regression that can approximate any arbitrary function [7] and generalise the model on unseen data. Hence in theory, NN should be able to approximate seasonal patterns directly from the underlying data generating process. Feedforward NNs are widely used in time series forecasting with the Multilayer Perceptron (MLP) being most frequently applied [8]. In forecasting with NN, even though studies using seasonal data have advocated the use of raw data [9-11] some studies have emphasised prior deseasonalisation [12-14]. However, most studies choose heuristics to determine the MLP architecture and parameters, often based on an ad-hoc trial-and-error approach with limited empirical evidence and reliability [8], often making an ex post replication of the experiments impossible. To determine all parameters in modeling a MLP, many modeling heuristics with equally limited validity and reliability were developed, often proposing conflicting rules-of-thumb on how to effectively determine the architecture of a MLP, making their successful application appear as much an art as a science. Input nodes, hidden nodes and activation functions are critical factors that can significantly affect the forecasting performance [15-17]. Consequently, the use of suboptimal architectures may impair the validity and reliability of experiments, and provide biased results in the discussion on how to forecast seasonal time series with NN.

Hence, the purpose of this study is twofold: a) to investigate the sensitivity of MLP architectural decision in forecasting seasonal time series: with respect to input nodes, hidden nodes and activation functions, and b) to determine the relative accuracy of different MLP architectures in forecasting a seasonal time series. The analysis is structured as follows: first we introduce the synthetic time series dataset and the experimental setup.

II. Experimental Design

A. Experimental data

In order to establish the sensitivity of different MLP architectures in forecasting seasonal time series we utilize a synthetic dataset, which is common practice in time series forecasting for model selection and evaluation [17, 18].

We use a data set of eight archetypical time series with medium noise derived from decomposing monthly retail sales in [19], that has been evaluated in previous experiments [20-22] and is available for download on the website www.neural-forecasting.com. Time series patterns are composed of overlaying components of a general level of the time series L, seasonality within a calendar year S, trends due to long term level shifts T and random noise E as a remaining error component. Through combination of the regular patterns of linear, exponential and degressive trends with additive or multiplicative seasonality we derive eight synthetic time series following the archetypical patterns motivated from Pegel's classification framework, later extended by Gardner to incorporate degressive trends. In particular, we create time series following an additive

Sven F. Crone, Department of Management Science, Lancaster University Management School, Lancaster LA1 4YX, United Kingdom (phone +44.1524.5-92991; e-mail: s.crone@lancaster.ac.uk)

Rohit Dhawan, Business Information Systems, Faculty of Economics and Business, University of Sydney, Australia (phone: +61.402173940; e-mail: rohit@dhawan.id.au).

seasonality without trend $L+S_A+E$ (S_A), multiplicative seasonality without trend but increasing with time $L+S_M*t+E$ (S_M), linear trend with additive seasonality $L+T_L+S_A+E$ (T_LS_A) and linear trend with multiplicative seasonality depending on the level of the time series $L+T_L*S_M+E$ (T_LS_M). Similar combinations of degressive (T_D) and progressive trend (T_P) are used with additive and multiplicative seasonality to T_DS_A, T_DS_M, T_PS_A and T_PS_M. Each time series has additive random noise $\sigma^2 = 25$ following a Gaussian distribution $N\left(0, \sigma^2\right)$.

Each time series consists of 228 monthly observations. To evaluate the effect of NN parameters and determine the ex ante accuracy of the forecasting methods, we divide each time series sequentially into a training, validation and test data subset of 132, 48 and 48 observations respectively.

B. Experimental Setup

We seek to evaluate a number of relevant architectural decisions. A complete enumeration of all potential architectural candidate models, although theoretically and computationally feasible, is challenging and exceeds the scope of this paper. Hence we limit our sensitivity analysis to a relevant subset of architectural decisions that have proven most relevant in previous analyses.

We limit our experiments to testing the sensitivity of the forecast errors on eight different sets of input nodes using n={3, 6, 9, 12, 13, 15, 18, 21} inputs representing different lagged realizations of the dependent variable y_{t-n} in steps of 3 nodes. Considering the monthly structure of time series, the inputs lags of 1…3, 1…6, 1…9, and 1…12 etc. input nodes were chosen to capture possible autocorrelation structures over a quarter, ½ year, ¾ of a year and a full year and longer structures in quarterly steps. The input vector of length 1..13 is explicitly used with reference to previous popular studies [23-25]. Twenty sets of hidden nodes were used h={1, 2, …, 20}, always using 1 to h hidden units. Possible combinations of activation functions include the logistic (Log), the hyperbolic tangent (TanH) and the Identity (Id) functions for the hidden layer and for the output layer {Log-TanH, Log-Id, TanH-Log, TanH-TanH, TanH-Id, Id-Log, Id-TanH, Id-Id}. The number of input and hidden nodes was chosen to encompass a search space from 2 to 301 degrees of freedom in comparison to 132 training data observations in order to reflect recent findings that over-parameterized MLPs seem to provide good generalizations in electrical load forecasting [26]. The activation functions represent all commonly used functions, and are deemed representative. All predictions are calculated as one-step-ahead forecasts $\hat{y}_{t+1}$ using one output node. Each of the individual candidate architectures is initialized 10 times with random starting weights in the interval [-1, 1] to account for the local search with a standard backpropagation algorithm with a variable learning rate, starting with 0.8 and being reduced by 1% after each epoch, without momentum term. Data is sampled in random order with replacement.

Each time series is scaled to facilitate learning speed and convergence. In order to avoid saturation effects on the instationary time series we scale all input and output data to fall into the range of [-0.6, 0.6] to account for headroom, using only the minimum and maximum for training- and validation set [20]. The choice of scaling may interact with the activation function, as TanH is defined in]-1, 1[while the Logistic function is defined only in]0, 1[. For that reason we later exclude the [Log-Log] combination from our analysis. The interaction of scaling and activation function will become evident later.

An initial full factorial experiment design created 23,400 networks per time series. However, a preliminary analysis yielded only limited insight into modeling decisions due to the significant noise in the results and graphs across all architectures. Therefore we further limited the experimental complexity by first determining a set of robust architecture parameters on the seasonal time series S_A in accordance with a pre-experiment in [20]. Of all architectures we analyze the top 10% of all candidates ordered by Median Absolute Percentage Error (MdAPE) on the validation set and select the architecture most frequently in the top percentile. This yields a generic MLP architecture of 12 input nodes, 10 hidden nodes, the logistic activation function in the hidden layer and the identity function in the output layer. It should be noted that this pre-selection may bias later findings. We then evaluate the effect of varying the number of input nodes, number of hidden nodes or the type of activation functions by varying only the parameters under investigation, keeping all other parameters set to the generic architecture. As a consequence, we train 14400 neural networks for each of the time series, calculating a total of 115,200 neural networks for the analysis. The accuracy of the MLPs is evaluated using the MdAPE across the 48 t+1 step ahead forecasts, which provides a more robust error metric then the MAPE. The errors are displayed in box-plots across 10 initializations to reveal the sensitivity of each architecture towards forecasting errors.

All experiments were calculated using the software Intelligent Forecaster developed by the first author. Average computation time per MLP was below 1 second.

III. Experimental Results

A. Input vector length

Previous studies have indicated that the choice of input variables and hence length of the input vector is crucial to the approximation and generalization performance of a MLP. The number of input nodes is determined either by heuristic trial-and-error experimentation [8] or using statistical tools through autocorrelation analysis or spectral analysis to identify seasonality and cycles [27]. Surprisingly, despite using autocorrelation or spectral analysis many authors extend the input vector of MLPs to include all lags unto the last significant lag, in contrast to the Box-Jenkins

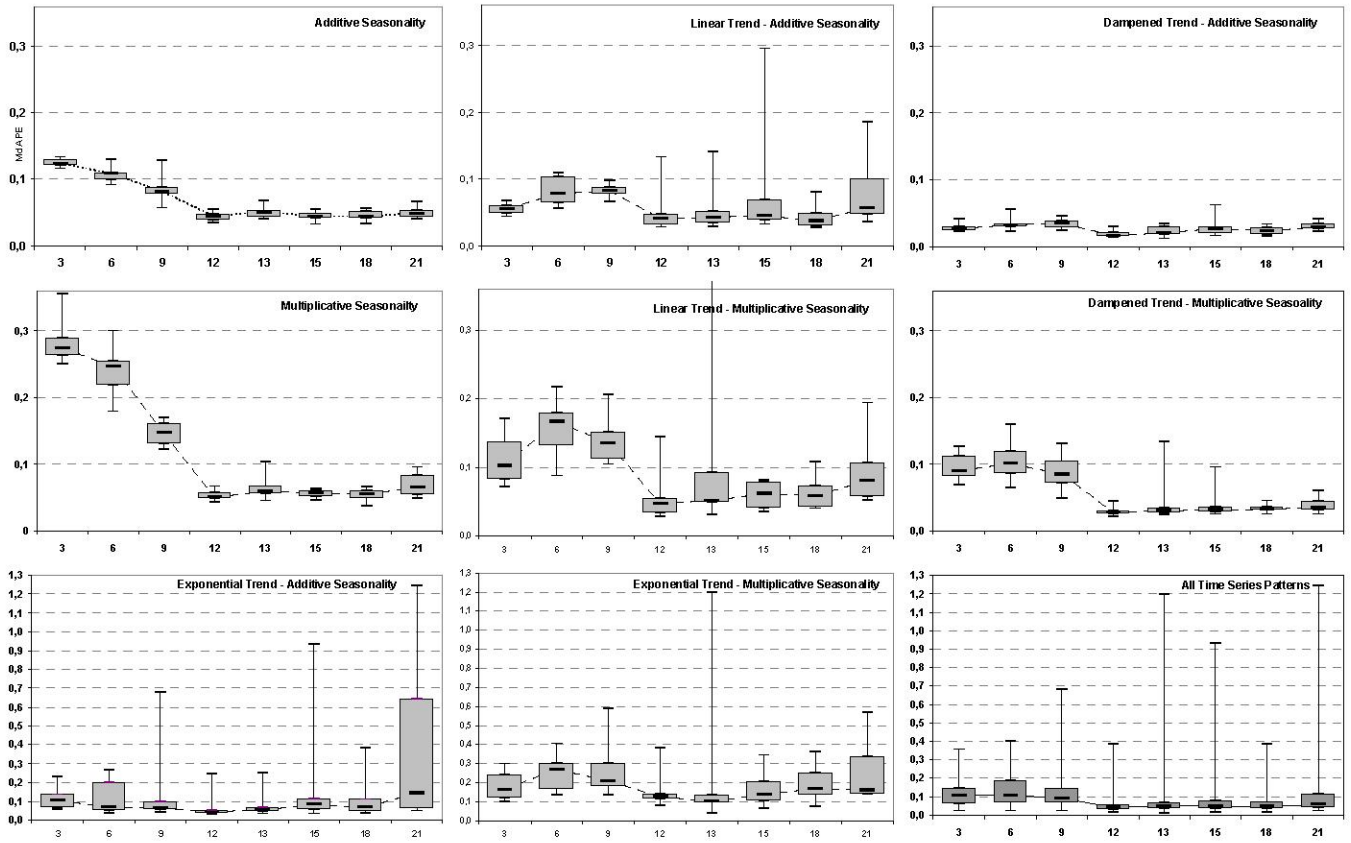

Fig 1: Box-Plots of MdAPE for different input nodes on eight time series with different seasonality and trends and across all time series

methodology of SARIMA modeling which would indicate an over-parameterized model and hence increased variance and bias [28]. Our procedure follows this simple example and reveals the impact of the input vector length in the box-plots of the MdAPE across all 10 initializations for each of the eight time series S_A, S_M, T_LS_A, T_LS_M, T_DS_A, T_DS_M, T_PS_A and T_PS_M and a combined box-plot with errors summed across all eight time series patterns. Box-plots may utilize different scale of the *y*-axis to exemplify error distributions, with horizontal lines indicating error steps of 0.1 in MdAPE.

As expected, the errors for basic time series patterns without trend S_A, S_M decrease significantly for input lags over 12 nodes, as the seasonal pattern requires an autoregressive lag of y_{t-11} and corresponding 12 input nodes. However, extending the input vector beyond 12 inputs for series S_A, S_M leads to only insignificantly deteriorated performance of the MLPs over all its initializations, as can be seen in a constant variance, minimum, maximum and median. This indicates a robust minimization instead of an expected over-specification. For selected time series patterns (e.g. series with linear trend T_LS_A, T_LS_M), the increase of the input vector leads to increased variance, despite limited increase of mean errors.

A similar pattern becomes evident in the graphs across all series. Different series can be approximated better across all initializations regardless of input vector length, e.g. S_A and the series with dampened trends T_DS_A, T_DS_M, due to the sigmoid form of the activation functions in internal information processing. Time series with exponential trend T_PS_A and T_PS_M lead to significantly higher errors, again with a decrease in median error and variance to 12 inputs and an increase in variance if the vector is further extended. The overall pattern is also confirmed in the distribution of the errors over all eight time series patterns, showing reduced median errors and variance for adequate specification of the input vector length. However, the majority of all initialisations appear surprisingly robust against input vector misspecification.

To summarize, MLP appear to be rather robust against using too many input nodes and irrelevant input vector information, once the relevant information is contained in the input vector. However, if there is a constraint on the number of data points available, a parsimonious modeling approach would indicate using as few as possible inputs, hence 12. If only a single MLP architecture was to be used, it should use 12 input nodes, as identified in the generic architecture.

B. Number of hidden nodes

In most NN studies, the number of hidden nodes is determined by ad-hoc experimentation or undisclosed rules of the thumb [29, 30], questioning the validity and reliability of the experiments and limiting their replicability. In contrast, some authors have noted that the number of hidden nodes has only limited impact on forecasting accuracy of a

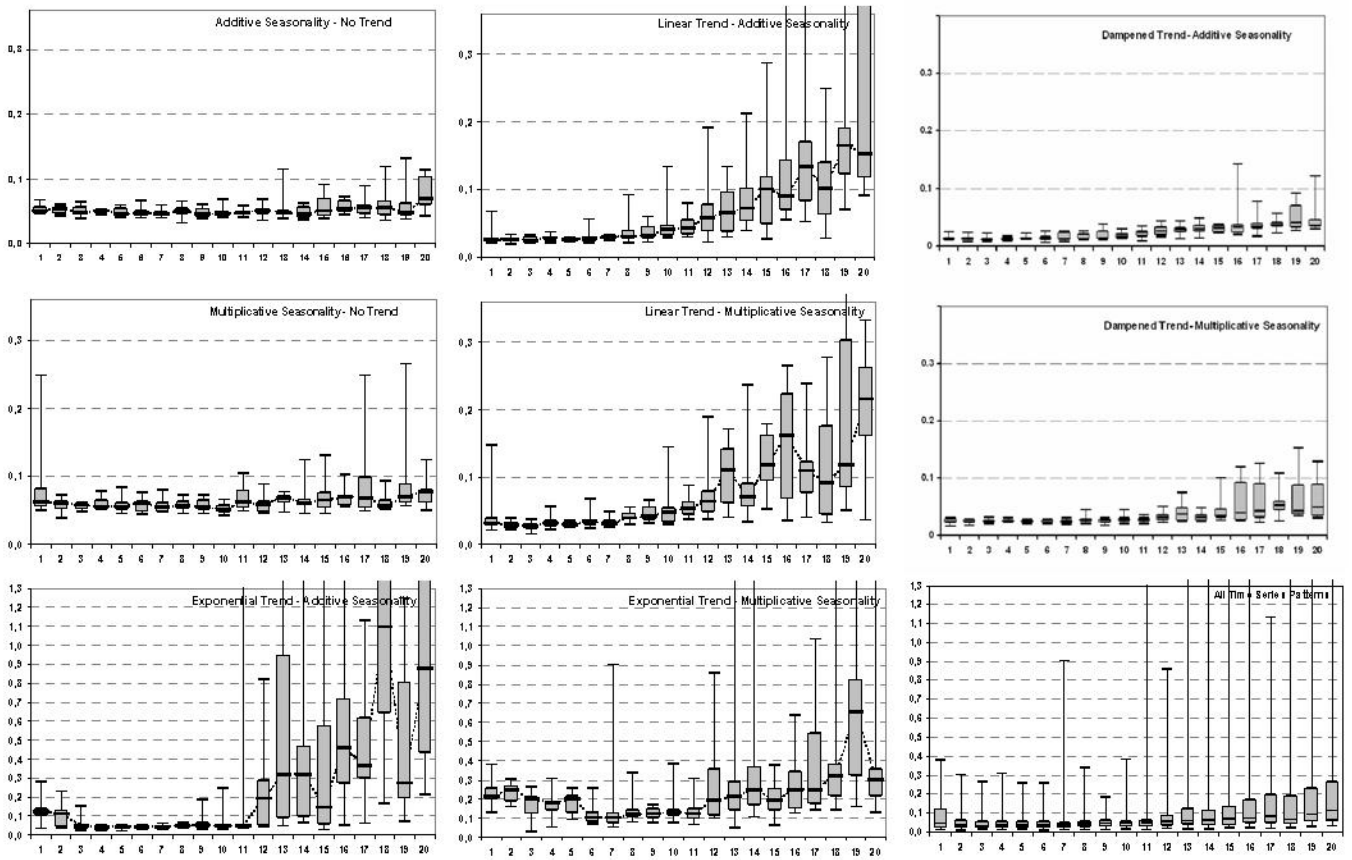

Fig 2: Box-Plots of MdAPE for different hidden nodes on eight time series with different seasonality and trends and across all time series

MLP. The impact of hidden nodes on the time series is displayed in Fig. 2. The box plots have been scaled to the same domain as in Fig. 1, occasionally cutting of maximum values and upper quartiles of the distribution. As the main objective is to allow a comparison of variance introduced by different modeling selections, only the lower tail of the distribution is of interest. Therefore we accept this tradeoff.

A number of distinct patterns become evident, depending on the structure of the time series. First, it is noticeable that the median error and error variance is different for different time series, indicating that some series may not only be predicted more accurately than others, but are also more robust to model misspecification, e.g. T_DS_A and T_DS_M. All time series show a significant increase in error magnitude and error variance with increasing numbers of hidden nodes, possibly indicating problems of training and convergence. Hence the importance of parsimonious model building can be confirmed, supporting the use of the Akaike (AIC) or Bayesian Information Criterion (BIS) for model selection.

Many time series show best performance for a low number of nodes, indicated by a continuously increasing median error and increasing error variance with additional hidden nodes, e.g. S_A, T_DS_A and T_DS_M. The other series show a pattern of continuously decreasing, almost stable and then increasing median errors and error variance, e.g. S_M, T_LS_A, T_LS_M, T_PS_A and T_PS_M. This demonstrates the need to find an adequate number of hidden nodes for each time series separately. Across the series, all architectures using more than 12 hidden nodes with the generic 12 input nodes represent the well-known problem of over-parameterized models with more degrees of freedom than observations in the training set. For the given set, an analysis of the average errors over all series shows that a good and robust performance can be achieved using 3, 5, 6 or 9 hidden nodes, and not for overparameterised MLPs as suggested by earlier research.

C. Choice of activation function

The activation functions in the hidden and output layer determine the form of the linear or nonlinear processing capabilities in the NN. The impact of all combinations is displayed in Fig.3.

The activation function combinations of [Id-Id] were eliminated across all graphs as they showed significantly inferior results, the minimum error of [Id-Id] always exceeding the maximum error of the [Log-Id] and most other activation function candidates. The inferior performance appears interesting, as MLPs with only linear activation functions represent a simple linear model of an AR(p)-process and many time series can be approximated as linear autoregressive patterns. It must be assumed, that the iterative computation of the learning algorithm impairs the ability of learning with these architectures, which requires further investigation. Also, the combination of [TanH-Id]

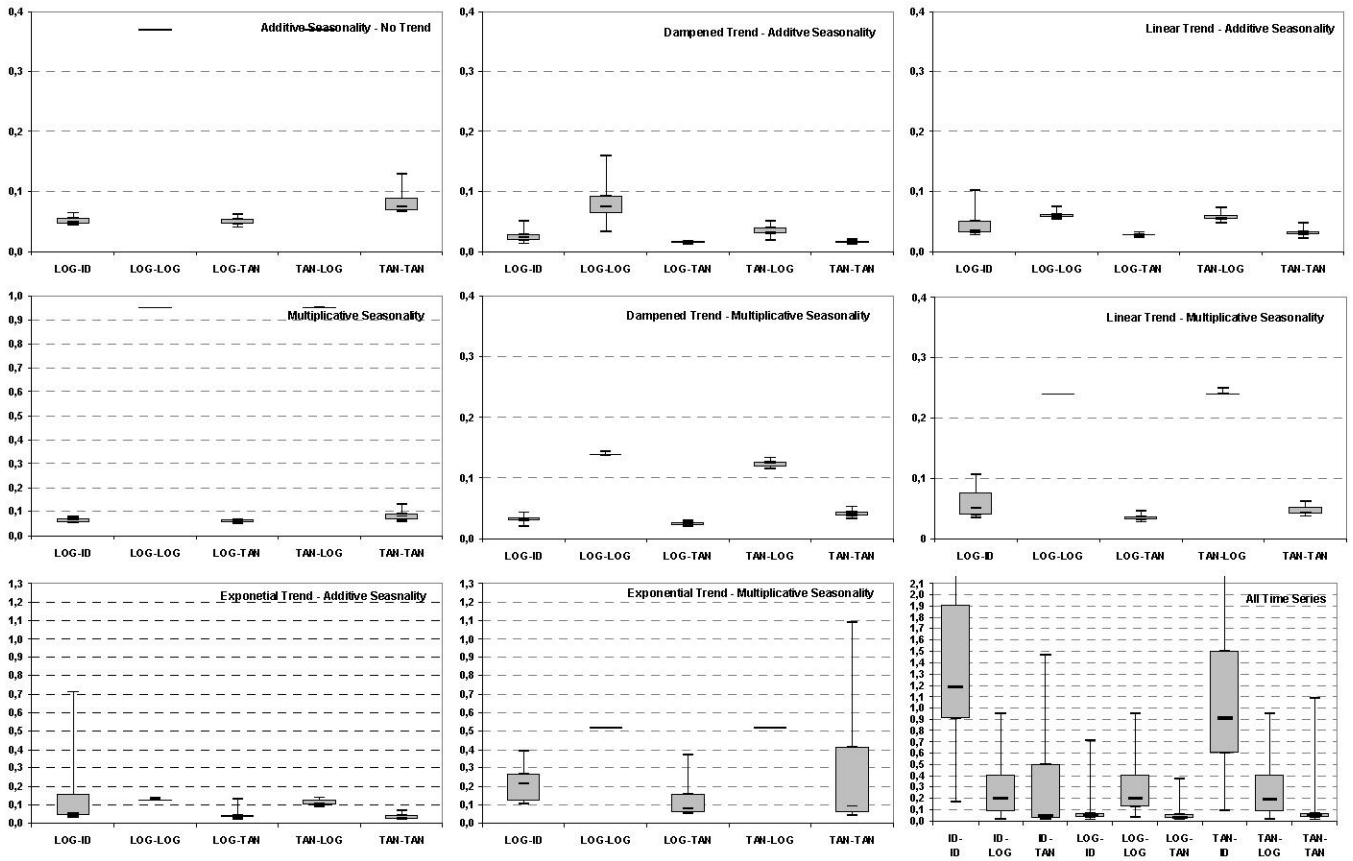

Fig 3: Box-Plots of MdAPE for different activation functions on eight time series with different seasonality and trends and across all time series

surprisingly showed significantly inferior performance, with the minimum of all initializations exceeding the error of the maximum of [Log-Id] and most other combinations, distorting the box-plots on all time series. This is surprising, as no other modeling selection, such as scaling into [-0.6, 0.6] should impair its performance, and considering the enthusiastic support from many authors to use only TanH activation functions due to the increase convergence speed [31]. Although showing competitive performance on some time series we also excluded [Id-Log] and [Id-TanH] in the detailed graphs of each series as a similar effect may be achieved with a MLP without hidden layers and a single output node, questioning the need for multiple layers.

The inferior performance of all combinations with a logistic activation function in the output layer can be directly attributed to the scaling of input- and output-data into the interval of [-0.6, 0.6] which exceeds the output range of a logistic function of [0, 1]. This becomes particularly evident for the time series S_M without trend, where most network outputs fall in this region. While this signifies and supports the importance of matching scaling to architecture selection [20], it does not indicate in inferiority of the Log activation function in the output layer. Further experiments using scaling into an adequate range, e.g. [0.25, 0.75] would be required to determine superiority of Log or TanH functions and to assess the efficacy of [Log-Log].

An analysis of the other popular activation function combinations indicate that a logistic function in the hidden layer and an identity function in the output layer leads to low and robust errors across different time series patterns, and hence the best performance of the frequently used activations functions in forecasting [32]. Interestingly, the previously unused combination of [Log-TanH] outperforms the [Log-Id] architectures robustly over all series. This effect requires further investigation, but may be attributed to the decreasing error contributions of the derivatives in earlier layers, hence providing the motivation for combining TanH with a robust hidden layer activation function.

IV. CONCLUSION AND FUTURE DIRECTION

We conducted a sensitivity analysis in order to investigate the conditions under which MLPs are capable of forecasting seasonal time series and how sensitive the forecasting accuracy is to parameter variations. The results show a significant impact of input vector length, number of hidden nodes and the choice of activation function. While the input vector length and the number hidden nodes must be determined individually for each time series, using the logistic activation function in the hidden layer and the identity function in the output layer provides robust results, although other functions show similar performance. The use of TanH in the hidden layer and Identity in the output layer shows discouraging performance and increased error variance across all time series, hence requiring careful

considerations should it be applied.

To compare across the equally scaled box plots, we can determine a comparable sensitivity to varying the individual parameters. The level of error and error variance introduced by misspecifying the activation function to [TanH-Id] or mismatching the activation function and scaling of input variables is followed by the avoidance of larger number of hidden nodes and the correct choice of a minimum input vector length to include all relevant information, followed by the selection of an adequate number of hidden nodes. To ensure valid and reliable results, all aspects must be considered with care, building parsimonious models with individual input vectors and number of hidden nodes for each time series. Once all these parameters are determined experimentally, MLPs are capable of forecasting seasonal time series without preprocessing.

The objective of our analysis was not to establish superiority of a particular modeling decision, but to investigate the conditions under which NN perform well and are robust. The results indicate substantial room for model mis-specification through the selection of suboptimal choices and again raise the issue of a robust neural network modeling methodology. The results of the sensitivity analysis may serve only as guidance for future modeling of seasonal time series. In order to establish NNs as a promising alternative to statistical methods in time series forecasting, we seek to extend the experiments to a full factorial design using statistical significance tests of ANOVA on multiple performance metrics including conventional yet non-robust measures of RMSE. The use of synthetic data provides better control of experimental design, but does not reflect the problems of real-world, short time series including outliers, level shifts and structural breaks. Hence, the experiments need to be extended to empirical datasets, such as the M-competition data, using multiple-step-head predictions, more than one hidden layer and multiple forecasting horizons to make fair generalizations with statistical benchmark methods. For future analysis, the evaluation of the naïve use of 1 to n time lags as inputs should be extended towards individual lags, e.g. 1, 12 and 13 from autocorrelation- and spectral analysis.

REFERENCES

[1] P. H. Franses, *Periodicity and Stochastic Trends in Economic Time Series*: Oxford University Press, 1996.

[2] A. Miron, *The Economics of Seasonal Cycles*: MIT Press, 1996.

[3] C. W. J. Granger, "Seasonality: causation, interpretation, and implications," in *Essays in econometrics: Spectral analysis, seasonality, nonlinearity, methodology, and forecasting*: Cambridge University Press, 2001, pp. 121-146.

[4] E. Ghysels, "A study towards a dynamic theory of seasonality for economic time series," *Journal of the American Statistical Association*, vol. 83, pp. 168–172, 1988.

[5] W. R. Bell and S. C. Hillmer, "Issues Involved with the Seasonal Adjustment of Economic Time Series," *Journal of Business & Economic Statistics*, vol. 2, pp. 291-320, 1984.

[6] E. J. Hannan, R. D. Terrell, and N. E. Tuckwell, "The Seasonal Adjustment of Economic Time Series," *International Economic Review*, vol. 11, pp. 24-52, 1970.

[7] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359-366, 1989.

[8] G. Zhang, B. Eddy Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks: The state of the art," *International Journal of Forecasting*, vol. 14, pp. 35-62, 1998.

[9] P. H. Franses and G. Draisma, "Recognizing changing seasonal patterns using artificial neural networks," *Journal of Econometrics*, vol. 81, pp. 273-280, 1997.

[10] S. Kang, "An Investigation of the Use of Feedforward Neural Networks for Forecasting," vol. Ph.D.: Kent State University, 1991.

[11] R. Sharda and R. B. Patil, "Connectionist approach to time series prediction: an empirical test," *Journal of Intelligent Manufacturing*, vol. 3, pp. 317-323, 1992.

[12] T. Hill, M. O'Connor, and W. Remus, "Neural network models for time series forecasts," *Management Science*, vol. 42, pp. 1082-1092, 1996.

[13] M. Nelson, T. Hill, W. Remus, and M. O'Connor, "Time series forecasting using neural networks: should the data be deseasonalized first?" *Journal of Forecasting*, vol. 18, pp. 359-367, 1999.

[14] G. P. Zhang and M. Qi, "Neural network forecasting for seasonal and trend time series," *European Journal of Operational Research*, vol. 160, pp. 501-514, 2005.

[15] K.-P. Liao and R. Fildes, "The accuracy of a procedural approach to specifying feedforward neural networks for forecasting," *Computers & Operations Research*, vol. 32, pp. 2151-2169, 2005.

[16] G. P. Zhang, "An investigation of neural networks for linear time-series forecasting," *Computers & OR*, vol. 28, pp. 1183-1202, 2001.

[17] G. P. Zhang, B. E. Patuwo, and M. Y. Hu, "A simulation study of artificial neural networks for nonlinear time-series forecasting," *Computers & Operations Research*, vol. 28, pp. 381-396, 2001.

[18] J.-L. Lin and C. W. J. Granger, "Forecasting from non-linear models in practice," *Journal of Forecasting*, vol. 13, pp. 1, 1994.

[19] S. Makridakis, S. C. Wheelwright, and R. J. Hyndman, *Forecasting Methods and Applications*, 3rd Edition ed. New York: Wiley, 1998.

[20] S. F. Crone, J. Guajardo, and R. Weber, "The impact of Data Preprocessing on Support Vector Regression and Artificial Neural Networks in Time Series Forecasting," presented at World Congress in Computational Intelligence, WCCI'06, Vancouver, Canada, 2006.

[21] S. F. Crone, S. Lessmann, and S. Pietsch, "An empirical Evaluation of Support Vector Regression versus Artificial Neural Networks to Forecast basic Time Series Patterns," presented at World Congress in Computational Intelligence, WCCI'06, Vancouver, Canada, 2006.

[22] S. F. Crone, S. Lessmann, S. Pietsch, "Parameter Sensitivity of Support Vector Regression and Neural Networks for Forecasting," International Conference on Data Mining, DMIN'06, Las Vegas, USA, 2006.

[23] J. Faraway and C. Chatfield, "Time series forecasting with neural networks: A comparative study using the airline data," *Applied statistics*, vol. 47, pp. 231-250, 1988.

[24] Z. Tang, C. de Almeida, and P. A. Fishwick, "Time series forecasting using neural networks vs. Box-Jenkins methodology," *Simulation*, vol. 57, pp. 303-310, 1991.

[25] Z. Tang and P. A. Fishwick, "Feedforward neural nets as models for time series forecasting," *ORSA*, vol. 5, pp. 374-385, 1993.

[26] H. S. Hippert, C. E. Pedreira, and R. C. Souza, "Neural networks for short-term load forecasting: A review and evaluation," *Ieee Transactions On Power Systems*, vol. 16, pp. 44-55, 2001.

[27] R. J. Frank, N. Davey, and S. P. Hunt, "Time Series Prediction and Neural Networks," *Journal of Intelligent and Robotic Systems*, vol. 31, pp. 91-103, 2001.

[28] G. Box and G. Jenkins, *Time Series Analysis: Forecasting and Control*. San Francisco: Holden Day, 1976.

[29] A. Blum, *Neural Networks in C++*. NY: Wiley, 1992.

[30] K. Swingler, "Financial prediction: Some pointers, pitfalls and common errors," *Neural Computing & Applications*, vol. 4, pp. 192-197, 1996.

[31] R. Neuneier and H.-G. Zimmermann, "How to Train Neural Networks," in *Neural networks: tricks of the trade*, G. Orr and K.-R. Müller, Eds. Berlin; New York: Springer, 1998, pp. 373-423.

[32] S. F. Crone and D. B. Pressmar, "An extended evaluation framework for neural network publications in sales forecasting," presented at Proceedings of the 24th IASTED international conference on Artificial intelligence and applications, Innsbruck, Austria, 2006.