A Business Forecasting Competition Approach to Modeling Artificial Neural Networks for Time Series Prediction

Sven F. Crone
Department of Management Science
Lancaster University, UK
E-Mail: crone@bis-lab.com

Abstract - **Various heuristic approaches have been proposed to limit design complexity and computing time in artificial neural network modelling, parameterisation and selection for time series prediction. However, no single approach demonstrates robust superiority on arbitrary datasets, causing additional decision problems and a trial-and-error approach to network modelling. To reflect this, we propose an extensive modelling approach exploiting available computational power to generate a multitude of models. This shifts the emphasis from evaluating different heuristic rules towards the valid and reliable selection of a single network architecture from a population of models, as common in business forecasting competitions. Experimental predictions are computed for the airline passenger data using variants of a multilayer perceptron trained with backproagation to minimize a mean squared error objective function, deriving a robust selection rule for superior prediction results.**

I. INTRODUCTION

Artificial neural networks (ANN) have found increasing consideration in forecasting theory, leading to successful applications in time series and explanatory forecasting in various domains, including business and management science [1-4]. ANNs promise attractive features to business forecasting, being a data driven learning machine, permitting universal approximation [5] of arbitrary linear or nonlinear functions from examples without a priori assumptions on the model structure, often outperforming conventional statistical approaches of ARIMA- or exponential smoothing- methods.

Despite their theoretical capabilities, NN are not an established forecasting method in business practice. Scepticism on NN persist through inconsistent research findings and pessimistic reports on their performance [6, 7], in part due to an trial-and-error modelling process [8] ANNs offer vast degrees of freedom in the modelling process, requiring a multitude of interdependent decisions on parameter-settings to assure valid and reliable performance. Since a complete enumeration of all parameter combinations induces high computation time, various heuristic modelling approaches, empirical guidelines, rules of thumb and simple tricks have been proposed since the late 1980s, suggesting alternative approaches to determine the architecture, guide the training process and select appropriate models to minimize the objective function [7, 9-12]. Unfortunately, no single heuristic has demonstrated its ability to deliver valid and reliable forecasts on arbitrary datasets as opposed to single experiments, therefore extending the decision problem of modelling ANNs instead of limiting it. Consequently, the task of modelling ANNs for a particular prediction problem is considered as much an art as a science [6, 7].

As no heuristic has demonstrated superior performance, we exploit available computational power to propose an extensive, simultaneous enumeration of the most influential modelling parameters of network size and depth, activation function, data sampling strategy, size of the data subsets, initialisation ranges of the trainable weights, learning parameters and early stopping schemes for a sufficient amount of initialisations while successively extending the input vector while analysing the network performance. The large amount and variety of models simulated shifts the emphasis from determining sound heuristics to limit modelling complexity towards a valid and reliable selection of a single superior model from a large population of competing, nonlinear autoregressive ANNs. Following, we employ a stepwise selection approach, effectively modelling a miniature forecasting competition motivated from the experience and publications in the domain of business forecasting [13, 14] to derive an adequate time series prediction applying a prototype simulator.

Following a brief introduction to the use of MLPs in time series prediction and their degrees of freedom in modelling, section 3 assesses the extensive modelling and design decisions in MLP application. This is followed by our experimental design and results for the proposed selection approach on the airline passenger data in section 4. Conclusions are given in section 5.

II. MODELING MULTILAYER PERCEPTRONS FOR TIME SERIES PREDICTION

Forecasting with non-recurrent ANNs may encompass prediction of a dependent variable $\hat{y}$ from lagged realisations of the predictor variable y_{t-n}, 1 or i explanatory variables x_i of metric, ordinal or nominal scale as well as lagged realisations thereof, $x_{i,t-n}$. Therefore, ANNs offer large degrees of freedom towards the forecasting design, permitting explanatory or causal forecasting through estimation of a functional relationship of the form

$$\hat{y} = f\left(x_1, x_2, ..., x_z\right) \quad , \qquad \mathbf{(1)}$$

as well a general transfer function models and simple time series prediction. Following, we present a brief introduction to the degrees of freedom in modelling ANNs for time series prediction; a general discussion is given in [15-20].

Forecasting time series with ANN is generally based on modelling the network in analogy to an non-linear autoregressive AR(p) model [7, 9, 21]. At a point in time t, a one-step ahead forecast $\hat{y}_{t+1}$ is computed using $p=n$ observations $y_t, y_{t-1}, \ldots, y_{t-n+1}$ from n preceding points in time $t, t\text{-}1, t\text{-}2, \ldots, t\text{-}n\text{+}1$, with n denoting the number of input units of the ANN. This models a time series prediction as of

$$\hat{y}_{t+1} = f\left(y_t, y_{t-1}, \ldots, y_{t-n+1}\right) \quad . \qquad \mathbf{(2)}$$

The architecture of a feed-forward MLP, a well researched ANN paradigm, of arbitrary topology is displayed in figure 1.

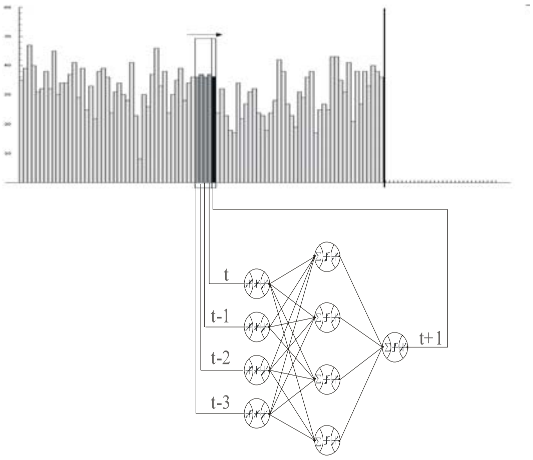

Fig. 1. Autoregressive MLP application to time series forecasting with a (4-4-1)-MLP, using *n=4* input neurons for observations in *t, t-1, t-2, t-3*, four hidden units, one output neuron for time period *t+1* and two layers of 20 trainable weights [22] The bias node is not displayed.

Data is presented to the MLP as a sliding window over the time series observations. The task of the MLP is to model the underlying generator of the data during training, so that a valid forecast is made when the trained ANN network is subsequently presented with a new input vector value [5].

The network paradigm of MLP offers extensive degrees of freedom in modeling for prediction tasks. Structuring the degrees of freedom, each expert must decide upon the selection and sampling of datasets D, the degrees of data preprocessing P, the static architectural properties A, the signal processing within nodes U and the learning algorithm L in order to achieve the design goal, characterized through the objective function or error function O, calling for decisions upon $ANN=[P, A, L, U, D, O]$.

In data preprocessing, decisions upon correction of observations C, normalization N and scaling S must be made: $P=[C, N, S]$. The architectural properties or topology of the net is primarily determined through the size of the input vector N^I corresponding to the number of input nodes and the length of the sliding window, the size N^S and depth N^L of the hidden layers through the number of layers and number of nodes in each hidden layer, and the length of the output vector determined through the number nodes N^O in the output layer. In addition, the architecture is determined through the connectivity of the weight matrix K (fully or sparsely connected, shortcut connections etc.) and the activation strategy T (feedforward or with feedback), leading to $A=[N^I, N^S, N^L, N^O, K, T]$. The signal processing within nodes, is determined by input function F^I (weighted sum, product, distance measures etc.), activation function F^A (tanh, logistic, sin etc. with offsets, limits etc.) and output function F^O (linear, winner takes all, softmax etc.), leading to $U=[F^I, F^A, F^O]$. Decisions concerning the learning algorithm encompass the choice of learning algorithm G (backpropagation, one of its derivatives, higher order methods or heuristics etc.), the complete vector of learning parameters for each individual layer L and each different phase T of the learning process, $P^{T,L}$, the procedure I^P and number of initializations for each network I^N and the choice of the stopping method for the selection of the best network solution B. In addition, the objective of the training process must be specified through the objective or function O, although often neglected in ANN theory and practice [22]. Consequently, the specification requires decisions upon $MLP=[[C, N, S],[N^I, N^S, N^L, K, T], [F^I, F^A, F^O], [G, P^T, I^P, I^N, B], D, O]$, with each parameter decision interacting with single or multiple other parameter recommendations. Consequently, we recommend an extensive modelling approach based upon an enumeration of the relevant alternatives and a model selection as common practice in business forecasting competitions. Following, we discuss the modelling setup and selection process on the airline data.

III. EXAUSTIVE MODELLING APPROACH FOR MULTILAYER PERCEPTRON APPLICATION

A. Data Analysis

We select the well known time series of monthly totals of airline passenger data, first proposed by Brown [23] and later extended by Box and Jenkins [12]. Fig. 2 gives an overview.

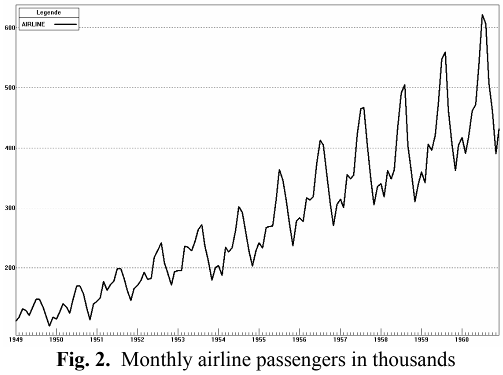

Fig. 2. Monthly airline passengers in thousands

The dataset has been repeatedly analyzed [4, 24-26] and may serve as a benchmark in ANN forecasting. The data consist of 12 years of monthly values from 1949 to 1960, leading to 132 observations. The time series requires no data cleansing regarding structural breaks, correction of outliers, pulses or level shifts due to temporal external shocks etc. As the analysis of the linear autocorrelations in the lag structure reveals a strong instationarity and seasonality in the time series, we analyze the partial linear autocorrelation coefficients of the integrated time series, as shown in fig. 3.

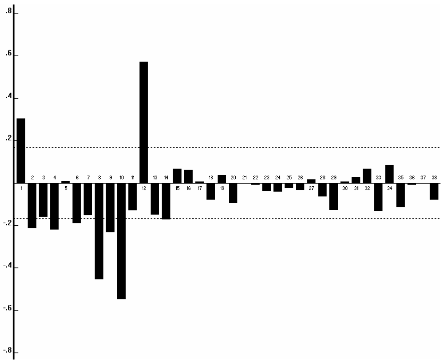

Fig. 3. Partial **a**utocorrelation function of first order integrated time series of airline passenger data for lags t-1 unto t-28

We identify significant partial linear autocorrelations at lags t-1, 2, 4, 6, 8, 9, 10, 12, 14 of the integrated time series, with the most dominant autocorrelations at t-1, 8, 10, 12. These may serve as a starting point in ANN modeling to determine non-linear lag structures, following a stepwise extension of the input vector from [t-1, 8, 10, 12] to [t-1,…, 14] for input node selection.

B. Data Preprocessing

Subsequent training, MLPs requires adequate preprocessing of data through scaling and differencing of input and output vectors. Although linear autocorrelation analysis suggests that first order differencing of the time series may assist the neural network in learning the relevant patterns [9, 21], we analyze the original un-integrated time series, despite an ongoing discussion on the necessity of integrating time series data for nonlinear NN models, especially considering the reconstruction of multiplicative, nonlinear effects such as seasonality in full scope [27, 28] versus the reduction of noise to highlight linear lag structures.

Subsequent to transformation, we normalize the original data to avoid computational problems, to meet algorithm requirements and to facilitate network learning through speeding up of the training process [29]. As various scaling methods allow linear equidistant or statistical scaling in arbitrary intervals [7, 10], we select a simple normalisation of

$$y_t' = \frac{y_t}{y_{\max} + h} \quad , \qquad \textbf{(3)}$$

to scale the data according to the natural origin of sales data [4]. In accordance with the selected activation function we scale input and output data from [114000; 622000] to the interval of [0; 1] including app. 50% headroom, $h=378000$, to avoid saturation effects of observations close to the asymptotic limits of the functions [15] and to account for trends in the test data outside the scale of training and validation data. To assure homogeneous scaling in all data sets we apply an external along channel normalization of the time series based upon the minimum and maximum values of training and validation data [7].

C. Architecture Selection and Node Processing

Architecture selection of a MLP imposes large degrees of freedom in the dataset dependent modeling process. The number of input nodes corresponds to number of lagged observations in the input vector to discover the underlying pattern in the time series for future forecasts [7]. While too few nodes leave out relevant, exploitable information of the linear and nonlinear autoregressive lag-structure of the time series, too many nodes add uncorrelated noise obscuring input patterns. In lack of a superior heuristic to determine the optimum input vector for an arbitrary dataset [30] we follow an approach by Lattermacher and Fuller [12], exploiting information derived from the preliminary analysis of linear autoregressive (AR) components to select appropriate time lags for the input vector as in ARMA and ARIMA modelling [12, 20], consequently determining the input nodes. As a result, we model nonlinear AR-ANNs, distinct from a conventional Box-Jenkins ARIMA approach due to nonlinearities in the AR-terms and omitting MA-terms [31, 32]. We commence model building with the linear AR-lags determined in the data analysis and successively extend the input vector by including less relevant autocorrelation-lags until all past observations from lags *t-1,...,t-14* are used.

While the number of input nodes is pre-determined through variable selection and data structure, the number of output nodes N^O in the output layer is determined by the forecasting horizon of the time series forecasting problem. For a $t+n$ prediction of n steps ahead, with $n>1$, two approaches are feasible. For forecasting horizons $t+n$, with $n>1$, we may model an MLP using one output node to model iterative $t+1$ step ahead forecasts, successively using predictions as inputs for subsequent forecasting horizons as in conventional ARIMA prediction. Alternatively, we may forecast $t+1,...,t+n$ values directly using a multiple-step-ahead forecasting architecture of n output units as proposed by [22]. Following, we limit our evaluation to a t+1 one step ahead forecast due to the nature of the prediction problem.

The number of hidden layers and corresponding hidden nodes in each layer is determined using an extensive enumeration, evaluating every combination of $l=1,...,3$ hidden layers and a maximum of $n=1,...,18$ hidden neurons, applying a step size of 2 nodes and limiting the structure of multi-layered ANNs to equal sized successive layers, to limit modeling complexity with regard to the scarce data.

Information processing functions within nodes are set homogeneously based upon experience and empirical evidence. We considered the summation as the input function, a linear output function and the hyperbolic tangent (tanh) as a nonlinear activation function in all hidden and output nodes, due to advantages in error propagating behavior [33]. The bias in each node is modeled as an on neuron connected to all nodes in all hidden and output layers with trainable weights.

Generally, we consider only fully connected feed forward architectures - no recurrent, sparsely connected networks or networks with shortcut connections are evaluated.

D. Training Process

Heteroassociative training of a MLP is the task of adjusting the weights of the links w_{ij} between units j and adjusting their thresholds to minimize the error δ_j between the actual and desired system behaviour [16] using various training algorithms for supervised online-training. We apply a simple derivative of the standard backpropagation gradient descent algorithm, applying a stepwise reduction of learning rate without momentum term to assure robust minimization of the objective function and minimizing a standard objective function of mean squared error (MSE), despite its theoretical and practical limitations [34] and the importance of selecting appropriate error metrics [32], modelling the conditional distribution of the output variables similar to statistical regression problems.

Within the learning process, we evaluate various combinations of different learning rates, $\eta = \{0.85; 0.4\}$, and cooling rates for their stepwise reduction of $\varpi = \{0.95; 0.98\}$ per epoch, deriving a variety of alternative learning schemes. Each network topology is trained for up to 150000 iterations with the weight configuration causing the lowest MSE on the validation set saved for future use. To limit computation time, we apply an early stopping paradigm, evaluating the relative reduction of the network error in percent after every epoch to a 0.001% threshold for 13000 iterations. To account for random initialisation of the connection weights, we initialise each MLP 15 times prior to training. No additional heuristic pruning or growing algorithms are applied.

Due to changing input vector sizes the overall number of patterns in the dataset may vary. We divide the dataset into three distinct sub-samples of a training set to parameterize the weights, a validation to guide early stopping and prevent overfitting and a test set to evaluate generalization on a hold-out set. Various subsample ratios are evaluated, leaving the test set constant through all experiments.

E. Model Evaluation and Selection

Following the training of various architectures a single model must be selected from all generated models for the final prediction. In accordance with objective function and final evaluation criteria of MSE, we select the model with the lowest MSE on the validation data set within the model group with the optimum modelling variant over all experiments. This approach exceeds the conventional pick-the-best approach of selecting the ANN architecture with the lowest validation error regardless of training error, variance of training and validation error as a measure of robustness of parameterization and potential generalisation within different initialisations. Our motivation for this approach is exemplified and evaluated in the following empirical experiment.

IV. SIMULATION EXPERIMENT

A. Experiment Structure

We evaluate a total of 14400 ANNs as variations of selected modelling parameters. To automate this extensive experiment we apply NeuroLab, a prototype ANN simulator developed within our research group for extensive or complete enumeration in ANN time series competitions. Each of the 36 architectures is initialised 15 times to account for randomized starting weights. In combination, we evaluate 4 variants of initialisation ranges, 3 variants of data splitting between training and validation set and 2 variants of backpropagation learning schemes. The networks were trained applying early stopping in 13716 cases if no error decrease of 0.01% in 300 epochs of 96 iterations took place, leading to a median of 63960 iterations or 666 epochs, with the final network parameters saved after a median of 24960 iterations or 260 epochs.

Total computation time was 19 hours on a Pentium IV, 2400 MHz, 1GB RAM, with an average time of 5 seconds for parameterization of the network and saving all training errors, results and parameters to completely reevaluate the experiment at any time. Considering the overall computation time, saving data and regular compression of the database of 1GB paralleled the time of actual experimental calculations.

B. Experimental Results

We analyze the results of all 14400 experiments regarding their performance on all data sets. A selection of results ranked by validation error presented in Tab. 1.

Tab. 1. Errors on all data sets by ANN topology ranked by validation error

Rank by	Data Set Errors			
validation error	Training	Validation	Test	ANN ID
overall lowest	0,009207	0,011455	0,017760	
overall highest	0,155513	0,146016	0,398628	
1st	0,010850	0,011455	0,043413	39 (3579)
2nd	0,009732	0,012093	0,023367	10 (5873)
…	…	…	…	…
25th	0,009632	0,013650	0,025886	8 (919)
…	…	…	…	…
14400th	0,014504	0,146016	0,398628	33 (12226)

According to early stopping we should select the ANN architecture with the lowest validation error for future applications. However, already the ANN ranked 2nd shows a significantly decreased test error, questioning the validity and reliability of the selection rule of lowest overall validation error. The ambiguity of the selection criteria becomes evident in plotting training, validation and test error in Fig. 3

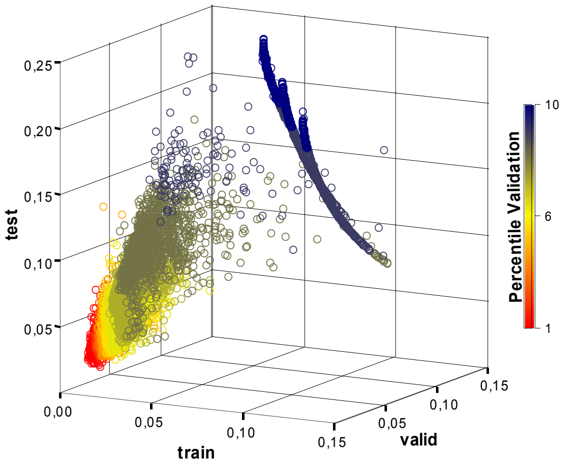

Fig. 3. Plot of training, validation and test error for all ANNs

Although displaying a general tendency of correlation between validation and test error, a large variance exists for low validation errors. This is supported by decreasing, significant correlation coefficients in an evaluation of all 14400, the top 1000 and the top 100 ANNs ranked by validation error in Tab. 2.

Tab. 2. Correlation coefficients of datasets and data ranges

	Correlation between datasets		
Data included	Train - Validate	Validate - Test	Train - Test
14400 ANNs	0,7786**	0,9750**	0,7686**
top 1000 ANNs	0,2652**	0,0917**	0,4204**
top 100 ANNs	0,2067**	0,1276**	0,4004**

The ability of generalisation through positive correlations between validation and test error within all ANNs does not apply for a stratified subsample of the top performing ANNs. For further analysis, we visualise the top percentile in an s-diagram in Fig. 4, sorting ANNs by validation error to graph the ranked training, validation and test-error to show performance, variance and correlation through symmetric development of in increasing error values in all sets. While the validation error must follow a steady upwards trend as the ranking criteria, we detect no similar pattern from the test-error, indicating limited correlation between the selection criteria of a minimum validation and test error. In addition, the graph highlights significant variance within the top percentile of ranked validation error, questioning the selection criteria of validation error further. Consequently, we have to extend our selection criteria to determine a network architecture capable of robust generalization from a single minimum validation error to limit variance in the test error through incorporation of additional selection rules in a stepwise model selection process.

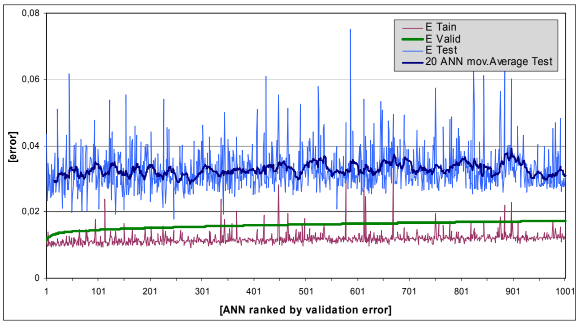

Fig. 4. S-Diagram of top percentile of ANNs ranked by validation error

C. Experimental Model Selection

In order to structure the ANN selection process to allow robust selection of a single ANN architecture, we evaluate the performance of each individual modelling variant within its group, in order to identify variants with superior performance through lower errors or lower variance of errors over all experiments. If dominant variants are identified, we combine each chosen variant and selected the ANN with the lowest validation error within the group of the dominant variant-combination. In order to analyse the sensitivity of each variant, we evaluate all groups of parameter variations for minimum, maximum and mean error as well as variance for each training, validation and test-dataset to determine a superior combination.

First, we evaluate four different initialisation ranges of network weights {[-0.01; 0.01], [-0.33, 0.33], [-0.66, 0.66], [-1, 1]} of 3510 ANNs each for their impact on performance. Fig. 5 reveals the superiority of the Init3 [-0.66, 0.66] initialisation through complete dominance of all other initialisations through the lowest minimum, mean, maximum and variance of error within training and validation set. This appears consistent with the results on the test set, although the minimum values in all datasets originate from different network experiments.

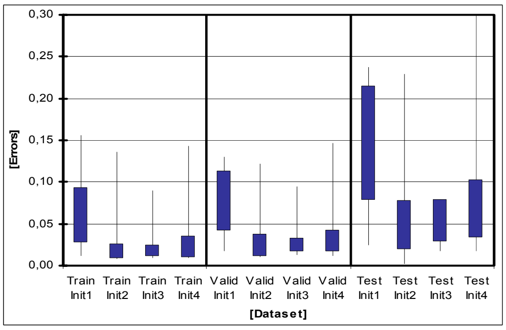

Fig. 5. Box plot of minimum, maximum and mean plus and minus one standard deviation of errors for variants of initialisation range

Following, we analyse the impact of different data samplings, assigning different partitions of data from to training and validation set, while leaving the test set size constant at 20%, equalling two years of observations. We evaluate three variants {[40%, 40%, 20%], [50%, 30%, 20%], [60%, 20%, 20%]], with the results of non-normalized errors permitting only an analysis of the test error in Fig. 6. The test errors highlight the dominance of a [60%- 20%] distribution of training to validation data in dataset sample 3, offering the most information for parameterisation through a larger training set. Finally, we analyse the impact of different learning schemes, demonstrating limited impact on the results of mean error and variance, but with lower maximum values in dominant learning scheme 3, also displayed in Fig. 6.

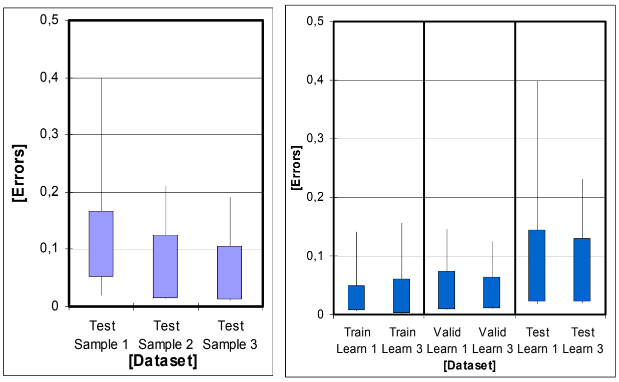

Fig. 6. Box plot of minimum, maximum and mean plus and minus one standard deviation of errors for variants of data sampling (left) and learning parameters for backpropagation (right)

Finally, we analyse the topologies of 10 single, 18 double and 8 three-hidden-layered network architectures regarding their mean errors on training, validation and test set over all 360 experiments each to identify superior architectures for the dataset. The results in Fig. 7 show a superior group of single layer architectures with 14 (ANN8), 16 (ANN9) and 18 (ANN10) hidden nodes, displaying low validation and test errors and correlation. Generally, one-layered ANNs outperformed multi-layered architectures in or experiment.

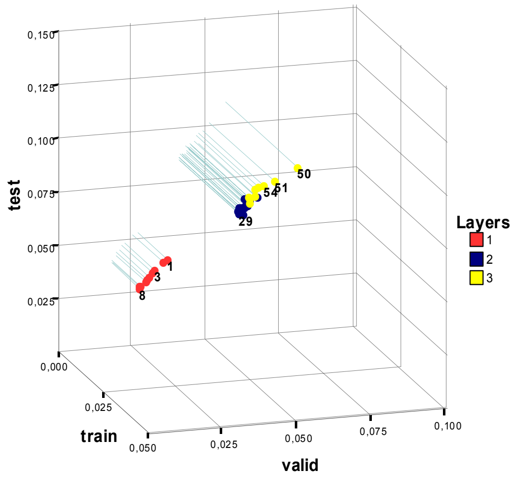

Fig. 7. Mean training, validation and test errors by ANN topology over all 360 variants each

Combining all evaluated modelling variants, we select the ANN with the lowest validation error within the combined group. The ranked errors of the selected subgroup for all 15 initializations are displayed in Fig. 8.

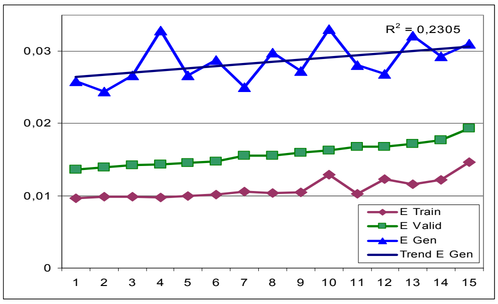

Fig. 8. S-diagram of training, validation and test error over 15 initializations of selected ANN8 architecture ranked by validation error

The ANN architectures show a consistent error development within the subgroup, supported by positive correlations of 0,8778 between training and validation, 0,4475 between validation and test and 0,4762 between training and test set. Consequently, the selection of the ANN with the lowest validation error may be considered robust, although not optimal.

Of all evaluated architectures, we selected ANN 919, a 14-14-1 architecture of variant 8 for it lowest validation error. The MLP was trained for 150000 iterations, without reaching the early stopping criteria. As determined by the selected variants, the ANN was trained using a starting learning rate of 0.85 and decreasing by 0.98 every epoch for 50000 iterations, followed by a starting learning rate of 0.4 and decreasing by 0.99 for 100.000 iterations, an initialization range of [-0.66, 0.66] and a split of the dataset of [60%, 20%, 20%], computing errors of [0.009632; 0.001365; 0.025886] on training, validation and test dataset. The network's t+1 output are shown in Fig. 9.

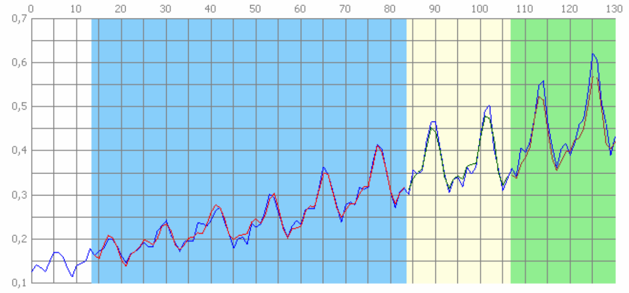

Fig. 9. Time series and predictions of ANN 919 on training, validation and test data set

However, the ANN selected does not represent the ANN with lowest test error used as the routine measure of generalization ability, as seen in Tab. 1. In fact, the ANN is ranked 9th on training, 25th on validation and 102nd on generalization error, and with single realizations of the same architecture in different variants achieving lower ranks. Therefore the selection strategy does achieve a "pick-best"-solution but aims to assure a robust selection process to derive valid and reliable results.

V. CONCLUSION

We proposed a complete or extensive enumeration of ANNs, avoiding problems in selecting acceptable heuristics and trial-and-error modelling approaches. However, the selection of a single ANN architecture based on naïve selection rules such as the lowest error on the validation dataset may lead to inconsistent, invalid and unreliable results. Consequently, we derive a more robust selection approach, evaluating a variety of modelling variants and limiting the ANN selection to the combined group of dominant variants.

Due to software limitations in the prototype simulator, we had to initial limit our experimental design, achieving only an extensive enumeration of the most relevant parameters. Further extensions of the software will incorporate additional modelling degrees of freedom and heuristics to derive robust results for a simultaneous enumeration of all possible variants over a variety of time series and benchmark datasets.

REFERENCES

[1] F. M. Thiesing, *Analyse und Prognose von Zeitreihen mit neuronalen Netzen*, Als Ms. gedr ed. Aachen: Shaker, 1998.

[2] F. M. Thiesing and O. Vornberger, "Forecasting Sales Using Neural Networks Computational Intelligence," presented at Proceedings International Conference, 5th Fuzzy Days, Dortmund, Germany, 1997.

[3] F. M. Thiesing and O. Vornberger, "Sales forecasting using neural networks," presented at Neural Networks,1997., International Conference on, 1997.

[4] Z. Tang and P. A. Fishwick, "Feed-forward Neural Networks as Models for Time Series Forecasting," *ORSA Journal on Computing*, vol. 5, pp. 374-386, 1993.

[5] K. Hornik, M. Stinchcombe, and H. White, "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks*, vol. 2, pp. 359 - 366, 1989.

[6] C. Chatfield, "Neural Networks - Forecasting Breakthrough or Passing Fad," *International Journal of Forecasting*, vol. 9, pp. 1-3, 1993.

[7] G. Zhang, B. E. 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.

[8] M. Adya and F. Collopy, "How effective are neural networks at forecasting and prediction? A review and evaluation," *Journal of Forecasting*, vol. 17, pp. 481-495, 1998.

[9] A. Lapedes, R. Farber, and Los Alamos National Laboratory., "Nonlinear signal processing using neural networks prediction and system modelling." Los Alamos, NM: Los Alamos National Laboratory, 1987.

[10] A. S. Weigend, "Predicting Sunspots and Exchange Rates with connectionist Networks," in *Nonlinear modeling and forecasting*, M. Casdagli and S. Eubank, Eds. New York, 1992, pp. 395-432.

[11] C. de Groot and D. Wurtz, "Analysis of univariate time series with connectionist nets: A case study of two classical examples," *Neurocomputing*, vol. 3, pp. 177-192, 1991.

[12] G. E. P. Box and G. M. Jenkins, *Time series analysis; forecasting and control*. San Francisco,: Holden-Day, 1970.

[13] R. Fildes, M. Hibon, S. Makridakis, and N. Meade, "Generalising about univariate forecasting methods: further empirical evidence," *International Journal of Forecasting*, vol. 14, pp. 339-358, 1998.

[14] S. Makridakis and M. Hibon, "The M3-Competition: results, conclusions and implications," *International Journal of Forecasting*, vol. 16, pp. 451-476, 2000.

[15] A. Zell, *Simulation Neuronaler Netze*. Munich: R. Oldenbourg Verlag, 2000.

[16] R. D. Reed and R. J. Marks, *Neural smithing : supervised learning in feedforward artificial neural networks*. Cambridge, Mass.: The MIT Press, 1999.

[17] S. S. Haykin, *Neural networks : a comprehensive foundation*, 2nd ed. Upper Saddle River, N.J.: Prentice Hall, 1999.

[18] C. M. Bishop, *Neural networks for pattern recognition*. Oxford: Clarendon Press, 1995.

[19] I. Kaastra and M. Boyd, "Designing a neural network for forecasting financial and economic time series," *Neurocomputing*, vol. 10, pp. 215-236, 1996.

[20] S. G. Makridakis, S. C. Wheelwright, and R. J. Hyndman, *Forecasting : methods and applications*. New York, 1998.

[21] D. Srinivasan, A. C. Liew, and C. S. Chang, "A Neural-Network Short-Term Load Forecaster," *Electric Power Systems Research*, vol. 28, pp. 227-234, 1994.

[22] S. F. Crone, "Training Artificial Neural Networks using Asymmetric Cost Functions," in *Computational Intelligence for the E-Age*, L. Wang, J. C.Rajapakse, K.Fukushima, and X. Y. S.-Y.Lee, Eds. Singapore: IEEE, 2002, pp. pp. 2374-2380.

[23] R. G. Brown, *Statistical forecasting for inventory control*. New York [u.a.]: McGraw-Hill, 1959.

[24] J. Faraway and C. Chatfield, "Time series forecasting with neural networks: A case study.," in *Research report 95-06 of the statistics group*. University of Bath, 1995, pp. 1-21.

[25] J. Faraway and C. Chatfield, "Time series forecasting with neural networks: A comparative study using the airline data," *Journal of the Royal Statistical Society Series C-Applied Statistics*, vol. 47, pp. 231-250, 1998.

[26] J. Faraway and C. Chatfietd, "Time series forecasting with neural networks: A comparative study using the airline data," *Applied statistics*, vol. 47, pp. 20, 1998.

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

[28] W. Remus and M. O'Connor, "Neural networks for time-series forecasting," in *Principles of forecasting: a handbook for researchers and practitioners*, J. S. Armstrong, Ed. Boston ; London: Kluwer Academic, 2001, pp. 245-258.

[29] E. M. Azoff, *Neural network time series forecasting of financial markets*. Chichester ; New York: Wiley, 1994.

[30] G. Lachtermacher and J. D. Fuller, "Backpropagation in time series forecasting," *Journal of Forecasting*, pp. 381-393, 1987.

[31] A. Zellner, "A tale of forecasting 1001 series : The Bayesian knight strikes again," *International Journal of Forecasting*, vol. 2, pp. 491-494, 1986.

[32] R. Fildes and S. Makridakis, "Forecasting and loss functions," *International Journal of Forecasting*, vol. 4, pp. 545, 1988.

[33] D. E. Rumelhart, J. L. McClelland, and University of California San Diego. PDP Research Group., *Parallel distributed processing : explorations in the microstructure of cognition*. Cambridge, Mass.: MIT Press, 1986.

[34] J. S. Armstrong, "Principles of forecasting: a handbook for researchers and practitioners," in *International series in operations research & management science ; 30*. Boston ; London: Kluwer Academic, 2001, pp. xii, 849.