2006 International Joint Conference on Neural Networks
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

# Forecasting with Computational Intelligence - An Evaluation of Support Vector Regression and Artificial Neural Networks for Time Series Prediction

Sven F. Crone, Stefan Lessmann and Swantje Pietsch

*Abstract*— **Recently, novel algorithms of Support Vector Regression and Neural Networks have received increasing attention in time series prediction. While they offer attractive theoretical properties, they have demonstrated only mixed results within real world application domains of particular time series structures and patterns. Commonly, time series are composed of a combination of regular patterns such as levels, trends and seasonal variations. Thus, the capability of novel methods to predict basic time series patterns is of particular relevance in evaluating their initial contribution to forecasting. This paper investigates the accuracy of competing forecasting methods of NN and SVR through an exhaustive empirical comparison of alternatively tuned candidate models on 36 artificial time series. Results obtained show that SVR and NN provide comparative accuracy and robustly outperform statistical methods on selected time series patterns.**

## I. Introduction

Support Vector regression (SVR) and artificial neural networks (NN) have found increasing consideration in forecasting theory, leading to successful applications in time series and explanatory forecasting in various application domains, including business and management science [1, 2]. Methods form computational intelligence promise attractive features to business forecasting, being data driven learning machines, permitting universal approximation of arbitrary linear or nonlinear functions from examples without a priori assumptions on the model structure, often outperforming conventional statistical approaches of ARMA-, ARIMA- or exponential smoothing-methods [3]. As a consequence, significant effort has been invested in developing forecasting methods from computational intelligence [4] to reduce forecasting error.

Despite their theoretical capabilities, NN as SVR are not an established forecasting method in business practice. Recently, substantial theoretical criticism of NN has raised questions to their ability to forecast even simple time series patterns of seasonality or trends without prior data preprocessing [5]. While all novel methods must ultimately be evaluated in an objective experiment using a number of empirical time series, adequate error measures and multiple origins of evaluation [6], the fundamental questions to their ability to approximate and generalise basic time series patterns must be evaluated beforehand. Time series can generally be characterized by the combination of basic regular patterns: level, trend, season and residual errors. For trend, a variety of linear, progressive, degressive and regressive patterns are feasible. For seasonality, an additive or multiplicative combination with level and trend further determines the shape of the empirical time series. Consequently, we evaluate SVR and NN using a consistent methodology [3] in comparison to a benchmark statistical forecasting expert system using Exponential Smoothing and ARIMA-models on a set of artificially created time series derived from previous publications. We evaluate the comparative forecasting accuracy of each method on alternative error measures to avoid evaluation biases in order to reflect their ability of learning and forecasting 12 fundamental time series patterns relevant to empirical forecasting tasks under 3 levels of increasing random noise. In total, we evaluate 500,000 NN and 2,900,000 SVR candidate models for their predictive accuracy.

This paper is organized as follows: first, we provide a brief introduction to SVR and NN in forecasting time series. Section three provides an overview of the experimental design including the artificially generated time series. This is followed by the experimental results and their discussion. Conclusions are given in section 4.

## II. Computational Intelligence for Forecasting

### A. Multilayer Perceptrons

NNs represent a class of mathematical models originally motivated by the information processing in biological neural systems [7-10]. They promise a number of attractive features of arbitrary input–output mapping from examples without a priori assumptions on the model structure, being a semi-parametric, data driven universal approximator, which make them well suited for time series prediction tasks.

Forecasting with non-recurrent NNs may encompass prediction of a dependent variable $\hat{y}$ from lagged realizations of the predictor variable $y_{t-n}$, 1 or $i$ explanatory variables $x_i$ of metric, ordinal or nominal scale as well as lagged realizations thereof, $x_{i,t-n}$. Therefore, NNs offer large degrees of freedom towards the forecasting design, permitting explanatory or causal forecasting through relationship of the form $\hat{y} = f(x_1, x_2, ..., x_z)$, as well a

Sven F. Crone (corresponding author), Department of Management Science, Lancaster University Management School, Lancaster LA1 4YX, United Kingdom (phone +44.1524.5-92991; e-mail: sven.f.crone@crone.de).

Stefan Lessmann, Swantje Pietsch, Institute of Information Systems, University of Hamburg, 20146 Hamburg, Germany (e-mail: Lessmann@econ.uni-hamburg.de; mailing@swantje-pietsch.de).

0-7803-9490-9/06/$20.00/©2006 IEEE

$\hat{y} = f(x_1, x_2, ..., x_z)$, as well a general transfer function models and simple time series prediction. Following, we present a brief introduction to modelling ANNs for time series prediction; a general discussion is given in [11, 12]. Forecasting time series with ANN is generally based on modeling the network in analogy to an non-linear autoregressive AR($p$) model [1, 13]. 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-1, t-2, ..., t-n+1*, with $n$ denoting the number of input units of the ANN. This models a time series prediction of the form

$$\hat{y}_{t+1} = f(y_t, y_{t-1}, ..., y_{t-n+1}). \quad (1)$$

In this study, a special class of NN, the well researched multilayer Perceptron (MLP) is applied. MLPs are hetero-associative, feed forward neural network which are typically composed of several layers of nodes with nonlinear signal processing [14] and trained by a derivative of the back propagation algorithm [14]. Applying a standard summation as the input unction and using an arbitrary nonlinear activation a MLP with a single layer of hidden nodes may be written as [15]

$$\hat{y}_t = f_{act}\left(w_{co} + \sum_h w_{ho} f_{act}\left(w_{ch} + \sum_i w_{ih} y_{t-j}\right)\right). \quad (2)$$

The architecture of a MLP is displayed in figure 1.

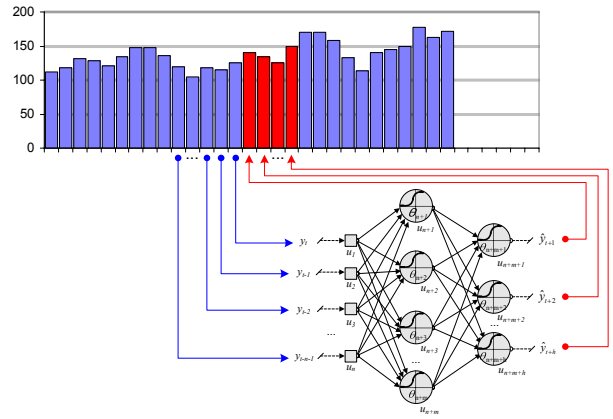


Fig. 1. Autoregressive MLP application to time series forecasting with a MLP of arbitrary topology, using $n$ input neurons for observations in *t, t-1, t-2, ..., t-n-1*, $m$ hidden units, $h$ output units for time periods *t+1, t+2, ..., t+h* and a two layers of trainable weights. The bias node is not displayed.

For a time series forecasting problem, training data is provided in form of vectors of $n=p$ time lagged observations [1, 8] in form of a sliding window over the time series observations [16]. 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].

Although the network paradigm of MLP offers extensive degrees of freedom in modeling for prediction tasks, it must be noted that they do not utilize recurrent feedback of their own output or previous errors and are therefore incapable of modeling moving average processes required to approximate data generating process of seasonal ARMA or ARIMA $(p,d,q)(P,D,Q)_s$ structure. For topologies without hidden nonlinear nodes, MLPs are equivalent to a linear AR($p$) models [9]. For a detailed discussion of these issues and the ability of NN to forecast univariate time series see [1].

### B. Support Vector Regression

Recently, SVR has been applied to time series prediction. SVR represents another method from computational intelligence related to NN and methodically based upon the statistical learning theory developed by Vapnik [2, 17, 18]. In this study we consider the $\varepsilon$-SVR, which approximates a function $f(\mathbf{x})$ to provide a maximum of $\varepsilon$-deviation from all target values $y_i$ in the training dataset $((\mathbf{x}_1, y_1), ..., (\mathbf{x}_\ell, y_\ell)) \subseteq (\mathbf{X} \times Y)^\ell$ and is as flat as possible [19-21]. Unlike the NN, the training problem of the SVR is a convex optimization problem without local minima [2] For a simple linear problem this function is of the form $f(x) = \langle \mathbf{w}, \mathbf{x} \rangle + b$ with $\mathbf{w} \in \mathbf{X}, b \in \Re$ and $\langle \mathbf{w}, \mathbf{x} \rangle$ denotes the dot product in the space of the input patterns $\mathbf{x}$ [17, 19, 22]. The support vectors are those data points used to describe the searched function [23]. In removing those training patterns which are not support vectors, the solution is unchanged and hence a fast method for validation is available when the support vectors are sparse [2, 24]. As noise exists, it is useful to work with a soft margin, as known from Support Vector Machines (SVM). This is realized by slack variables $\xi_i^+, \xi_i^- \geq 0$ which extend the mathematical formulation of the convex optimization problem [2],

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{\ell}(\xi_i^+ + \xi_i^-), \quad (3)$$

which has to be minimized by $\|\mathbf{w}\|^2 = \langle \mathbf{w}, \mathbf{w} \rangle$ with the constraints $(\mathbf{w} \cdot \mathbf{x_i}) + b - y_i = \varepsilon + \xi_i^-$, $y_i - (\mathbf{w} \cdot \mathbf{x_i}) - b \leq \varepsilon + \xi_i^+$ to ensure flatness [23, 25]. The constant C determines the trade-off between flatness and the amount of outliers of the $\varepsilon$-tube, which is handled in this study with the $\varepsilon$-intensive loss function [26]

$$|\xi|_\varepsilon := \begin{cases} 0 & if \ |\xi| \leq \varepsilon \\ |\xi| - \varepsilon & otherwise \end{cases}. \quad (4)$$

For this particular cost function the Lagrange multipliers are sparse [2, 24] and only data points outside the $\varepsilon$-tube contribute to costs. To assure that the training data appear in the form of dot products between the vectors and to better handle the constraints, the problem is transformed to a Lagrangian formulation [2]:

$$L(\mathbf{w}, b, \xi_i, \xi_i^*) := \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{\ell}(\xi_i^+ + \xi_i^-) - \sum_{i=1}^{\ell}(\eta_i^+ \xi_i^+ + \eta_i^- \xi_i^-)$$
$$-\sum_{i=1}^{\ell}\alpha_i^+(\varepsilon + \xi_i^+ - y_i + (\mathbf{w} \cdot \mathbf{x_i}) + b) \quad (5)$$
$$-\sum_{i}^{\ell}\alpha_i^-(\varepsilon + \xi_i^- + y_i - (\mathbf{w} \cdot \mathbf{x_i}) - b).$$

This represents the precondition for nonlinear problems. Here $L$ is the Lagrangian function and $\eta_i^\pm$ and $\alpha_i^\pm$ are positive and the Lagrange multipliers. To receive the dual optimization problem,

$$-\frac{1}{2}\sum_{i,j=1}^{\ell}\left(\alpha_i^+-\alpha_i^-\right)\left(\alpha_i^+-\alpha_j^-\right)\left(\mathbf{x_i}\cdot\mathbf{x_j}\right) \quad , \qquad -\varepsilon\sum_{i=1}^{\ell}\left(\alpha_i^++\alpha_i^-\right)+\sum_{i=1}^{\ell}y_i\left(\alpha_i^+-\alpha_i^-\right)\rightarrow \max! \tag{6}$$

with subject to $\sum_{i=1}^{\ell}\left(\alpha_i^+-\alpha_i^-\right)=0$ and $\alpha_i^+,\alpha_i^-\in[0,C]$, the partial derivatives of L with respect to the primal variables $\mathbf{w},b$ and $\xi_i^{\pm}$ are vanished and substituted to the primal function [2]. With the condition $\partial_{\xi_i^+}L=C-\alpha_i^+-\eta_i^+=0$ and $\partial_{\xi_i^-}L=C-\alpha_i^--\eta_i^-=0$ the dual variables $\xi_i^{\pm}$ can be eliminated and thus the dual optimization problem reformulated as Support Vector (SV) expansion $f(\mathbf{x}_i)=\sum_{i=1}^{\ell}\left(\alpha_i^+-\alpha_i^-\right)\left(\mathbf{x_i}\cdot\mathbf{x}_j\right)+b$, which is a linear combination of the training patterns[27]. The coefficients $\alpha_i^{\pm}$ are the parameters to be adjusted by training and $\mathbf{x_i}$ are the training patterns. The choice of the bias b gives rise to several variants [28] In this study the Karush–Kuhn–Tucker (KKT) conditions are used [2, 24, 26]. This method base on the idea that the variables $\alpha_i^{\pm}$, for those the prediction error can be determined are uniquely. This means for the $\varepsilon$-intensive case, to select the data dots on the margin as here the exact value of the prediction error is known and calculate for the according data dot the threshold b [26]. To guarantee stability, $b$ is calculated for all dots on the margin and the average is used as threshold [26].

Nonlinearity can be created by nonlinear mapping $\phi$ the data into a high dimensional feature space $F$ and do linear regression in this space, thus this corresponds to nonlinear regression in a low dimensional input space [2]. As mapping all data to space can easily become computationally infeasible for polynomial features of higher order and higher dimensionality [23]. To avoid this, kernel functions $k(\mathbf{x},\mathbf{x}')=\langle\phi(\mathbf{x}),\phi(\mathbf{x}')\rangle$ are used, that enable operations to be performed in the input space rather than the potentially high dimensional feature space, hence the inner product does not need to be evaluated in the feature space[29]. All kernel functions, those correspond to the inner product of some feature space, must satisfy Mercer's condition. This study uses the Gausian Radial Basis Function (RBF)

$$k(\mathbf{x},\mathbf{x}')=\exp\left(-\gamma\left|\mathbf{x}-\mathbf{x}'\right|^2\right),\gamma>0 \quad . \tag{7}$$

which represents the most commonly used kernel for regression problems [2, 26] and corresponds to minimizing the specific cost function with a regularization operator and satisfies the Mercer conditions, as any symmetric kernel function [23, 26, 28]. Finally in this study the quadratic programming problem is defined as minimize

$$\frac{1}{2}\sum_{i,j=1}^{\ell}\left(\alpha_i^+-\alpha_i^-\right)\left(\alpha_j^+-\alpha_j^-\right)k\left(\mathbf{x_i},\mathbf{x_j}\right) \quad , \qquad -\varepsilon\sum_{i=1}^{\ell}\left(\alpha_i^++\alpha_i^-\right)+\sum_{i=1}^{\ell}y_i\left(\alpha_i^+-\alpha_i^-\right) \tag{8}$$

with subject to $\sum_{i=1}^{\ell}\alpha_i^+-\alpha_i^-=0$, $\alpha_i^+,\alpha_i^-\in[0,C]$ and $i=1,\ldots,\ell$ [26]. As the RBF kernel function is used in the experiments, the output weights as well as the RBF centers and variances are adjusted by back-propagation [30].

## III. Experimental Design

### A. Experimental Data

In order to evaluate the ability of SVR and MLP to forecast a benchmark subset of common time series patterns, we develop a set of archetype time series derived from decomposing monthly retail sales in [16]. Time series patterns are composed of overlaying components of a general level *L* of the time series, seasonality *S* within a calendar year, trends *T* in the form of long term level shifts and random noise *E* as a remaining error component. Through combination of the regular patterns of linear, progressive, degressive or regressive trends with additive or multiplicative seasonality we derive 12 artificial time series following the patterns motivated from Pegel's classification framework, later extended by Gardner to incorporate degressive trends [31]. In particular, we create time series following a stationary pattern L+E denoted as (E), additive 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 L+$T_L$+E ($T_L$), linear trend with

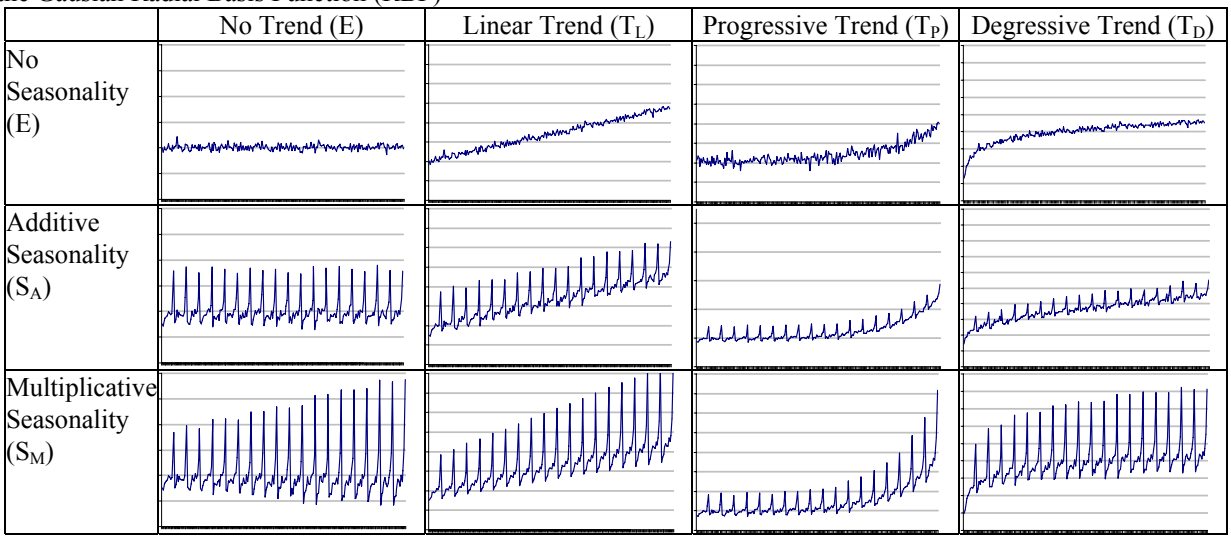


Fig. 2. Basic time series patterns of artificial time derived from the Pegels- and Gardner-classification, combining Level, Trend and Seasonality with a medium level of additive noise.

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 $L+T_L*S_M+E$ ($T_LS_M$). The functional form of these basic time series patterns is visualized in the left six quadrants of Fig. 1. In addition, we model similar combinations of degressive and progressive trend ($T_P$) with additive and multiplicative seasonality to $T_DS_A$, $T_DS_M$, $T_PS_A$ and $T_PS_M$ displayed in the six right quadrants of Fig.1.

Each time series is overlaid with tree levels of low, medium and high additive random noise $\sigma^2 = 1,\ 25,\ 100$ following a Gaussian distribution $N(0,\sigma^2)$, thereby creating a total of 36 time series of 228 monthly observations [8]. The time series may be distinguished in linear versus nonlinear patterns, with the patterns of E, $T_L$, $S_A$ and $T_L+S_A$ relating to linear model forms and all other combinations to nonlinear models. Consequently, we can subsequently analyze the experimental results of forecasting accuracy of competing methods using multiple hypotheses of varying noise and different time series structure.

Each time series is split into training set, validation set and test set using a proportion of [60%, 20%, 20%] in accordance with [32]. As the size of the test set affects the validity of the forecasting results [33], but very long time series often do not reflect empirical data availability, a test set size of 48 observations data serves as a sufficient and acceptable trade-off. For the statistical benchmark methods, which do not require the use of a validation set, both training and validation set are used for parameterization, with an identical out-of-sample test set used for all methods

### *B. General Experimental Setup*

We determine a number of identical input variables for both NN and SVR. Each time series may be characterized by a different autoregressive lag structure and require a different number of input nodes. As a consequence, we identified suitable lag-structures for inclusion in the input vector following the approach by Lattermacher and Fuller using the linear autocorrelation function (ACF) and partial autocorrelation functions (PACF) as is common practice in ARIMA-modeling [16, 34]. In particular, we generate an input vector length using the last statistically significant PACF lag of a time series successively differenced until stationary [8, 35].

All data for NN and SVR was linearly scaled to avoid numerical difficulties and to speed up the training process [3, 8], using

$$z_t = AF_{\min} + (AF_{\max} - AF_{\min}) \cdot \frac{(x_t - x_{t\,\min})}{(x_{t\,\max} - x_{t\,\min})}, \qquad (9)$$

with $z_t$ the scaled data used for training and $x_{t\ max}$ and $x_{t\ min}$ the maximum or minimum observed value on the training and validation set of each time series [3]. In order to avoid saturation effects close to the asymptotic limits of nonlinear activation function [-1;1] through non-stationary time series with consistent trends or seasonality, we applied an additional 50% headroom $AF_{max}$=0,5 and $AF_{min}$=-0.5, effectively scaling the data into the interval [-0.5; 0.5].

As the relative performance of each forecasting method is influenced by the selection of the evaluation criteria [16, 36, 37], we evaluate the forecasting accuracy using a set of five established accuracy measures: mean absolute error (MAE), mean absolute percentage error (MAPE), median absolute percentage error (MdAPE), Root mean squared error (RMSE) and Theil's U-statistic (TU), which are discussed in detail in [16]. Although RMSE and MAPE provide a strong bias in over-penalizing large deviations or sensitivity to the scale of the errors, their information is provided to allow comparisons with alternative studies frequently applying these inefficient error statistics. The TU statistic provides a relative accuracy measure in comparison to the accuracy of a naïve forecast using the last observation as a prediction, with values TU<1 demonstrating superior performance then a naïve method and values TU> 1 indicating inferior accuracy [31]. All error measures are calculated as mean errors across an out-of-sample test set. In addition, we calculate ordinal performance metrics. Each forecasting method is ranked by each error measure, with a 1 indicating highest performance and a 3 documenting the lowest. The means of these ranks across all time series are calculated to demonstrate relative performance robust to influences from outliers. The use of ordinal error measures based upon rank-information omits information on the distance between individual methods. As a consequence, we propose an additional distance measure, with the worst error measure setting an origin of zero percent and the optimum of $e$=0 setting 100%. In relation to this, the percentage distances of the other two methods were calculated. Thus, the higher the percentage of the distance the closer the method performs to the optimum [3]. These distances may be accumulated across different error measures and time series in order to further analyze the differences in accuracy of the forecasting methods.

### *C. Setup of Forecasting Models*

The accuracy of each forecasting method is determined by its specific architecture. Both NN [2] and SVR [2] offer a large amount of degrees of freedom in customizing and parameterising models to a particular forecasting task. Thus we need to evaluate a number of candidate models to determine a suitable MLP and SVR architecture.

For the $\varepsilon$-SVR with RBF kernel function, the model accuracy depends on the parameters $\varepsilon$, C and $\gamma$ [35]. We evaluate a variety of parameter combinations through a systematic grid search with exponentially growing sequences as proposed by Hsu [28, 38]. First, a coarse grid of $C=[2^{-5}, 2^{-4.5}, \ldots, 2^{15}]$, $\gamma=[2^{-23.0}, 2^{-22.5}, \ldots, 2^{0}]$ and $\varepsilon=[2^{-12}, 2^{-11.5}, \ldots, 2^{3}]$ is used. The parameter combination with highest validation accuracy is picked and its region successively analyzed applying a refined grid using an exponential reduced sequence of step sizes from 0.5, 0.05, 0.005 unto 0.0005. As a consequence, the initial grid evaluates 59,737 parameter combinations and each successive refined grid a further 8,000 parameter combinations. Using this shrinking technique we aim to reduce the total training time in considering only a subset of

free variables [39]. Of all SVR candidates, the one with the lowest error on the validation dataset was selected.

In order to determine an effective MLP topology, a set of 70 different NN topologies using 240 different parameter combinations is evaluated, resulting in 16,800 different MLP candidate models for each time series. A maximum of 30 nodes are distributed across a maximum of 3 layers of hidden nodes, evaluating every combination of hidden layer [1,…,3] and nodes [0,…,30] in steps of 2 nodes, limiting the candidates to pyramidal topologies with the number of nodes in successive hidden layers equal or smaller to the preceding ones in order to limit the design complexity [10, 32, 40]. The maximum of 30 nodes was set to reflect the number of free parameters in relation to the training patterns. All predictions are computed as iterative one-step ahead predictions *t+1*, a single output node is used. The number of input nodes is determined ex ante through the analysis of the autocorrelation structure of each time series, resulting in a total of 70 topologies for each successive variation of model parameters. For information processing within the nodes, the established hyperbolic tangent activation function TanH is applied in all hidden nodes [4, 39] and a linear activation function in the single output node [41], using a simple summation as the input function in all nodes. To allow for randomized starting points each MLP is randomly initialized 20 times using three different initializations intervals of [-0.88;0.88], [-0.55;0.55] and [0.22;0.22]. Each MLP candidate is trained for 1000 epochs using four different initial learning rates of [0.05; 0.35; 0.65; 1] which are reduced by a cooling factor of 0.99 after each epoch of presenting all data patterns in the training set to the input nodes in random order. During training, the NN with the lowest error on the validation set is saved, applying early stopping if the MSE on the validation set has not decreased for 100 epochs. The MLP candidate showing the lowest MSE validation error is selected for forecasting. Each MLP was simulated using a NN software simulator "Intelligent Forecaster" developed for large scale empirical evaluations of NN by the authors.

To serve as a benchmark, all time series are evaluated using an established expert forecasting system ForecastPro, which evaluates ARIMA-models and various forms of Exponential Smoothing-methods using an automatic model selection technique [42], allowing robust prediction of stationary, seasonal, trended and trend-seasonal time series patterns. The superior performance of the forecasting software has been demonstrated sufficiently in outperforming other software and human experts in the M3-competition [35].

## IV. Experimental results

The following tables provide the results of the forecasting performance of the SVR and NN models in comparison to the statistical methods. The time series results are separated into patterns of nonlinear trends in Table 1 versus time series of linear patterns in Table 2. For each time series, we computed a total of 16,800 MLP candidates and 91,737 SVR candidates, resulting into a total evaluation of 537,600 NN models and 2,935,584 SVR models to determine suitable candidate models for each time series pattern. Although results for all error measures are provided, information on the relative performance of each method should not be derived from the biased error metrics of RMSE or MAPE.

On non-linear time series it is evident that SVR and NN significantly outperform the statistical benchmark methods applied by ForecastPro on most performance criteria. Both methods demonstrate the ability to robustly learn and extrapolate all of the provided time series patterns, stationary or instationary, without any in data preprocessing through detrending or deseasonalisation. Their general ability to forecast is documented through a TU significantly smaller than 1, indicating higher performance than a Naïve forecast and therefore their general applicability in forecasting basic time series patterns.

TABLE 1
Out Of Sample Performance For Non linear Time Series on three Noise Levels

| | | LOW NOISE LEVEL | | | | | MEDIUM NOISE LEVEL | | | | | HIGH NOISE LEVEL | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Type | Method | MAE | MdAPE | MAPE | TU | RMSE | MAE | MdAPE | MAPE | TU | RMSE | MAE | MdAPE | MAPE | TU | RMSE |
| $T_E$ | SVR | **0.41** | **2.64** | **4.74** | **0.34** | **0.52** | **2.17** | **13.05** | 56.39 | **0.39** | **2.85** | 5.01 | **19.15** | **65.03** | 0.42 | 6.16 |
| | MLP | 0.43 | 2.67 | 5.54 | 0.36 | 0.54 | 2.61 | 19.63 | 95.61 | 0.44 | 3.24 | **4.70** | 24.31 | 125.04 | **0.40** | **5.87** |
| | Stat. M. | 2.17 | 4.92 | 7.18 | 1.58 | 2.76 | 3.95 | 16.05 | **49.18** | 0.69 | 4.92 | 5.47 | 19.32 | 71.65 | 0.45 | 6.81 |
| $T_D$ | SVR | **0.42** | **0.16** | **0.21** | **0.36** | **0.54** | 2.19 | 0.92 | 1.08 | 0.38 | **2.74** | **4.09** | **1.56** | **1.98** | **0.36** | **5.22** |
| | MLP | 0.50 | 0.20 | 0.26 | 0.42 | 0.63 | **2.17** | **0.87** | **1.07** | **0.38** | 2.76 | 4.33 | 1.70 | 2.16 | 0.37 | 5.38 |
| | Stat. M. | 1.22 | 0.54 | 0.58 | 1.02 | 1.44 | 3.30 | 1.38 | 1.56 | 0.55 | 3.96 | 4.15 | 1.60 | 2.03 | 0.37 | 5.32 |
| $T_P$ $S_A$ | SVR | **0.69** | **0.22** | **0.25** | **0.02** | **0.85** | **3.43** | **1.09** | **1.23** | **0.12** | **4.40** | **5.29** | **1.83** | **2.01** | **0.17** | **6.37** |
| | MLP | 1.17 | 0.23 | 0.35 | 0.05 | 1.79 | 3.77 | 1.18 | 1.38 | 0.13 | 4.70 | 6.31 | 1.94 | 2.32 | 0.20 | 7.88 |
| | Stat. M. | 6.18 | 1.21 | 1.52 | 0.23 | 8.28 | 10.26 | 2.45 | 2.79 | 0.35 | 12.80 | 10.81 | 2.87 | 3.13 | 0.35 | 13.67 |
| $T_P$ $S_M$ | SVR | **1.98** | **0.65** | **0.86** | **0.04** | **3.32** | **4.84** | **1.80** | **1.95** | **0.09** | **7.71** | **6.62** | **2.87** | **3.73** | **0.13** | **8.83** |
| | MLP | 2.10 | 0.71 | 0.89 | 0.05 | 3.50 | 6.01 | 2.06 | 2.42 | 0.12 | 9.77 | 7.20 | 3.44 | 3.92 | 0.14 | 9.63 |
| | Stat. M. | 6.05 | 1.55 | 1.95 | 0.10 | 8.85 | 18.62 | 3.74 | 6.25 | 0.27 | 24.63 | 11.19 | 4.23 | 4.85 | 0.19 | 14.92 |
| $T_D$ $S_A$ | SVR | 0.90 | 0.18 | 26.26 | 0.04 | 1.52 | 2.58 | **0.49** | 0.66 | 0.09 | 3.29 | 5.03 | 1.07 | 1.26 | 0.16 | 6.32 |
| | MLP | **0.88** | **0.18** | **24.81** | **0.03** | **1.13** | 2.88 | 0.61 | 0.74 | 0.10 | 3.66 | 5.24 | 1.15 | 1.33 | 0.17 | 6.54 |
| | Stat. M. | 1.01 | 0.21 | 25.55 | 0.04 | 1.27 | **2.45** | 0.57 | **0.63** | **0.08** | **2.99** | **4.96** | **0.96** | **1.24** | **0.16** | **6.29** |
| $T_D$ $S_M$ | SVR | **0.71** | **0.27** | **0.37** | **0.01** | **0.94** | 2.90 | 1.24 | 1.45 | 0.05 | 3.70 | 4.47 | 1.68 | 2.21 | 0.08 | 5.84 |
| | MLP | 1.26 | 0.57 | 0.65 | 0.02 | 1.50 | 3.29 | 1.38 | 1.68 | 0.06 | 4.12 | 4.53 | 1.92 | 2.23 | 0.08 | **5.70** |
| | Stat. M. | 0.98 | 0.36 | 0.49 | 0.02 | 1.31 | **2.38** | **0.96** | **1.22** | **0.04** | **3.03** | **4.41** | **1.65** | **2.21** | **0.08** | 5.73 |

TABLE 2
OUT OF SAMPLE PERFORMANCE FOR LINEAR TIME SERIES ON THREE NOISE LEVELS

| | | LOW NOISE LEVEL | | | | | MEDIUM NOISE LEVEL | | | | | HIGH NOISE LEVEL | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Type | Method | MAE | MdAPE | MAPE | TU | RMSE | MAE | MdAPE | MAPE | TU | RMSE | MAE | MdAPE | MAPE | TU | RMSE |
| | SVR | 0.38 | 24.28 | 690 | **0.35** | 0.48 | 1.96 | 43.67 | 487.84 | 0.35 | 2.50 | **3.80** | **47.31** | 428.05 | 0.33 | 4.87 |
| E | MLP | **0.38** | **23.91** | 830 | 0.36 | **0.48** | **1.95** | **43.37** | 449.85 | **0.34** | **2.49** | 3.83 | 47.93 | 417.17 | 0.34 | 4.94 |
| | Stat. M. | 0.39 | 25.27 | **607** | 0.36 | 0.49 | 1.96 | 44.62 | **414.20** | 0.35 | 2.51 | 3.81 | 48.18 | **154.52** | **0.33** | **4.85** |
| | SVR | 0.42 | 0.20 | 0.24 | 0.36 | 0.53 | 2.07 | 0.98 | 1.18 | 0.37 | 2.70 | 4.14 | 1.84 | 2.30 | 0.36 | 5.23 |
| $T_L$ | MLP | 0.43 | 0.21 | 0.25 | 0.37 | 0.55 | 2.17 | 1.03 | 1.24 | 0.38 | 2.76 | 4.52 | 2.08 | 2.57 | 0.39 | 5.60 |
| | Stat. M. | **0.40** | **0.18** | **0.23** | **0.34** | **0.50** | **1.98** | **0.91** | **1.12** | **0.35** | **2.54** | **3.86** | **1.62** | **2.16** | **0.34** | **4.99** |
| | SVR | **0.41** | **0.35** | **0.43** | **0.02** | **0.53** | 2.13 | 1.92 | 2.21 | 0.07 | 2.68 | **3.89** | **2.64** | **4.23** | **0.13** | **5.16** |
| $S_A$ | MLP | 0.43 | 0.36 | 0.44 | 0.02 | 0.54 | 2.17 | 1.83 | 2.26 | 0.08 | 2.75 | 4.34 | 3.49 | 4.59 | 0.14 | 5.52 |
| | Stat. M. | 0.53 | 0.46 | 0.55 | 0.02 | 0.67 | **2.02** | **1.77** | **2.09** | **0.07** | **2.59** | 4.02 | 2.90 | 4.34 | 0.14 | 5.25 |
| | SVR | 0.58 | 0.53 | 0.65 | 0.01 | 0.73 | **2.48** | 2.29 | **2.83** | **0.05** | **3.03** | **4.15** | **3.44** | **4.95** | **0.10** | **5.43** |
| $S_M$ | MLP | 0.58 | 0.50 | 0.64 | 0.01 | 0.72 | 2.52 | **2.16** | 2.86 | 0.05 | 3.16 | 4.81 | 4.18 | 5.79 | 0.10 | 5.95 |
| | Stat. M. | **0.52** | **0.42** | **0.57** | **0.01** | **0.70** | 2.81 | 2.54 | 3.05 | 0.06 | 3.69 | 5.11 | 4.05 | 6.00 | 0.11 | 6.53 |
| $T_L$ $S_A$ | SVR | 0.56 | 0.27 | 0.32 | 0.02 | 0.70 | 2.37 | 1.21 | 130.03 | 0.08 | 2.98 | 5.81 | 2.81 | 3.23 | 0.19 | 7.22 |
| | MLP | 0.57 | 0.27 | 0.32 | 0.02 | 0.70 | 2.39 | 1.08 | 132.39 | 0.08 | 2.96 | 5.48 | 2.67 | 3.05 | 0.18 | 6.83 |
| | Stat. M. | **0.43** | **0.19** | **0.24** | **0.02** | **0.54** | **2.10** | **0.97** | **117.08** | **0.07** | **2.67** | **4.04** | **1.60** | **2.27** | **0.14** | **5.27** |
| $T_L$ $S_M$ | SVR | 0.51 | 0.23 | 29.28 | 0.01 | 0.66 | 2.62 | 1.07 | 150.12 | 0.05 | 3.41 | 5.20 | 2.25 | 297.09 | 0.10 | 6.76 |
| | MLP | 0.50 | 0.23 | 28.84 | 0.01 | 0.63 | 2.61 | 1.15 | 154.00 | 0.05 | 3.36 | 5.31 | 2.42 | 301.56 | 0.10 | 6.70 |
| | Stat. M. | **0.43** | **0.20** | **24.99** | **0.01** | **0.54** | **2.16** | **0.98** | **125.95** | **0.04** | **2.76** | **4.23** | **1.63** | **242.72** | **0.08** | **5.50** |

SVR slightly outperform MLP on three of the six series for all noise levels, with statistical methods outperforming SVR and NN on two series with higher noise levels and MLPs showing only inconsistent performance. However, the differences between SVR and NN performance do not appear to be significant, with NNs always providing the second best performance across all series. Moreover, SVR and NN show robust performance regardless of time series pattern, while the statistical benchmark performs worse then naïve methods on selected time series. The results are largely consistent across error measures, with slight inconsistencies only for $T_E$ and $T_D$ $S_A$ patterns of medium noise level, showing robustness of the solution.

While we may conclude that SVR shows great promise in forecasting basic nonlinear time series patterns, their performance on linear patterns given in Table 2 is not as dominant. For linear time series patterns, NN and the statistical benchmark methods outperform SVR on all but one time series consistently across all error measures. In particular for simple linear patterns, the established statistical methods of Exponential Smoothing and ARIMA outperform both NN and SVR. Again, all methods show superior performance to the Naïve method, documenting the general ability of all three approaches to forecast all of the 12 basic time series patterns without data preprocessing except a simple scaling technique applying headroom.

In order to derive more general results, we calculate the mean out of sample errors for each method across all time series patterns on the three levels of noise in Table 3, using only the unbiased error measures of MAE, MdAPE and TU with the best performance of a method indicated in bold.

TABLE 3
MEAN OUT OF SAMPLE PERFORMANCE ACROSS ALL TIME SERIES PATTERNS

| | Low Noise | | | Medium Noise | | | High Noise | | |
|---|---|---|---|---|---|---|---|---|---|
| | MAE | MdAPE | TU | MAE | MdAPE | TU | MAE | MdAPE | TU |
| SVR | **0,66** | **2,50** | **0,13** | **2,65** | **5,81** | **0,17** | **4,79** | **7,37** | **0,21** |
| MLP | 0,77 | 2,50 | 0,14 | 2,88 | 6,36 | 0,18 | 5,05 | 8,10 | 0,22 |
| Stat.M. | 1,69 | 2,96 | 0,31 | 4,50 | 6,41 | 0,24 | 5,51 | 7,55 | 0,23 |

SVR clearly outperform statistical methods closely followed by MLP, although their enhanced performance in comparison to MLPs does not prove to be statistically different. Interestingly, the differences between forecasting methods decrease with an increasing level of noise, indicating extended complications in determining patterns.

A similar picture is derived in averaging the performance metrics further across all time series and noise levels in Table 4.

TABLE 4
MEAN PERFORMANCE ACROSS ALL TIME SERIES AND NOISE LEVELS

| | MAE | MdAPE | TU |
|---|---|---|---|
| SVR | **2,70** | **5,23** | **0,17** |
| MLP | 2,90 | 5,66 | 0,18 |
| Statistical Methods. | 3,90 | 5,64 | 0,26 |

Again, the results indicate the preeminent accuracy of SVR against statistical methods as well as MLPs. To extend this analysis, we compute a distance based accuracy to evaluate relative method performance for different noise levels and linear versus nonlinear patterns in Fig. 3.

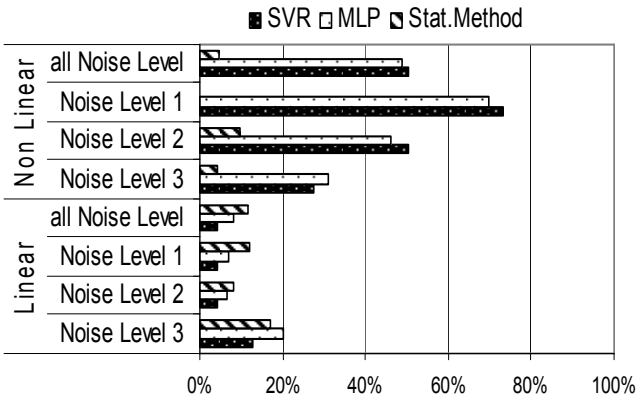


Fig. 3. The distance measure visualizes the difference of the accuracy between the single forecasting methods, with noise level 1 denoting low noise, level 2 medium noise and level 3 high noise.

The distance measures again indicate that SVR performs best on all non linear time series closely followed by NN and at a significant distance form the statistical methods. For non-linear time series on a low noise level the statistical benchmark does not appear since its forecasts were always the worst on each evaluation criterion. For linear time series the statistical methods perform best, again closely followed by the NN. It must be noted, that the level of differences in performance between the methods on linear time series is much smaller then on the nonlinear time series, in particular for noise level one and two. Therefore, the ordinal rank based accuracy measures provided in table 3 may suggest a slight bias in the evaluation of methods.

TABLE 3
ACCUMULATED RELATIVE ORDINAL MEAN RANKS

| Mean Ranks | All Noise Levels | Noise Level 1 | Noise Level 2 | Noise Level 3 |
|---|---|---|---|---|
| Non Linear Time Series | | | | |
| SVR | **1.42** | **1.07** | **1.40** | **1.67** |
| MLP | 1.92 | 1.93 | 2.27 | 1.87 |
| Stat. Method | 2.66 | 3.00 | 2.33 | 2.47 |
| Linear Time Series | | | | |
| SVR | 2.40 | 2.23 | 2.50 | 2.47 |
| MLP | 1.87 | 2.03 | **1.70** | 1.87 |
| Stat. Method | **1.73** | **1.73** | 1.80 | **1.67** |
| All Time Series | | | | |
| SVR | 1.91 | **1.65** | **1.95** | 2,07 |
| MLP | **1.89** | 1.98 | 1.98 | **1,87** |
| Stat .Method | 2.19 | 2.36 | 2.06 | 2,07 |

The ranking illustrate more clearly the ability of SVR to predict non linear time series, while their performance deteriorates for linear time series. Statistical methods perform best on linear time series and worst on nonlinear patterns. NN perform second best on both types of time series, always coming in close second place also with regard to other non-ordinal error measures. In summarizing over all time series, NN show the best performance, allowing valid and reliable forecasting of linear as well as nonlinear time series patterns.

## V. CONCLUSIONS

We analyze the performance of competing forecasting methods of SVR and MLP from computational intelligence versus established benchmarks of univariate statistical forecasting methods. In order to derive the general ability of SVR and MLP to predict the most common time series patterns, we combine various forms of seasonal and trended time series patterns to create a benchmark dataset of 36 time series, consisting of 12 basic patterns overlaid with three levels of noise. In order to facilitate future comparisons, all time series are published at the website www.neural-forecasting.com.

The results are evaluated using five established error measures on out-of-sample accuracy. He experiments clearly indicate the ability of SVR as well as MLP to robustly forecast various forms of stationary, trended, seasonal and trend-seasonal time series without prior detrending or deseasonalisation of the data. SVR and MLPs demonstrate preeminent accuracy in comparison to statistical methods on non linear time series patterns. While statistical methods outperform SVR and NN on basic linear patterns, the differences in accuracy are not substantial. Therefore, SVR show a generally superior forecasting performance closely followed by MLP on mean accuracy measures, and MLP showing a robust forecasting accuracy using an evaluation on rank based accuracy measures.

Similar to other empirical studies, we do not attempt to demonstrate a general superiority of a particular time series method for all potential forecasting applications and time series. However, in the light of recent criticism that NN are incapable of forecasting even basic time series patterns, we provide a strong indication that MLP as well as SVR may indeed be applied successfully for time series prediction of various trend-seasonal time series without prior data analysis and iterative data preprocessing. Moreover, both SVR and MLPs validate their semi-parametric ability to learn an adequate model form and the corresponding parameters directly from the presented data, avoiding issues of conventional model selection of statistical forecasting methods. However, this evaluation has certain limitations. Even if a substantial variety of SVR parameters and NN architectures was evaluated, the evaluation took only a single methodology into consideration, which was based upon a refined simple grid search and a linearly motivated estimation of adequate lag structures. Also, not all potential NN architectures, activation functions or SVR kernel functions were evaluated. In particular, recurrent NN which are theoretically capable of approximating nonlinear AR($p$) as well as nonlinear ARIMA($p,d,q$)-processes should be evaluated. It may be possible, that alternative SVR and NN models with better forecasting accuracy or robustness exit even for the time series in question. In particular, the reduced SVR performance on the linear time series may be

attributed to the use of an RBF kernel function, which would further support the need for extended experimentation.

For future evaluations we also seek to extend the experiments to linear and polynomial kernel functions and to analyze the resulting forecasting accuracy with regard to the increased complexity of the modeling process. In addition, we need to extend our evaluation towards multiple time origins trough different sampling proportions, multiple step ahead forecasts and different forecasting horizons as well as empirical time series of a given application domain, in order to assure a valid and reliable evaluation on the ability of SVR and MLP to enhance future forecasting research and practice.

## REFERENCES


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

[2] A. J. Smola; and B. Schölkopf, "A Tutorial on Support Vector Regression," Australian National University / Max-Planck-Institut für biologische Kypernetik, Canberra / Thübingen 2003.

[3] K.-P. Liao; and R. Fildes, "The accuracy of a procedural approach to specifying feedforward neural networks for forecasting," *Computers & Operations Research*, pp. 2121-2169, 2005.

[4] G. Zhang, "Linear and Nonlinear Time Series Forecasting with Artifical Neural Networks," vol. Doctor of Philosophy: Kent State Graduate School of Management, 1998, pp. 152.

[5] 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.

[6] L. J. Tashman, "Out-of-sample tests of forecasting accuracy: an analysis and review," *International Journal of Forecasting*, vol. 16, pp. 437-450, 2000.

[7] A. Zell, *Simulation neuronaler Netze*, vol. 1. Aufl. Bonn: Addison - Wesley Verlag, 1994.

[8] S. F. Crone, "Stepwise Selection of Artificial Neural Networks Models for Time Series Prediction," University of Lancaster, Lancaster (UK) 2004.

[9] V. N. Vapnik, "An Overview of Statistical Learning Theory," *IEE Transactions on Neural Networks*, pp. 988-999, 1999.

[10] R. Callan, *Neuronale Netze im Klartext*. München: Pearson Studium, 2003.

[11] C. M. Bishop, *Neural networks for pattern recognition*. Oxford New York: Clarendon Press;Oxford University Press, 1995.

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

[13] A. Lapedes and R. Farber, "How neural nets work," in *Neural Information Processing Systems*, D. Z. Anderson, Ed. New York: American Institute of Physics, 1988, pp. 442-456.

[14] S. D. Balkin; and J. K. Ord, "Automatic neural network modelling for univariate time series," *International Journal of Forecasting*, pp. 509-515, 2000.

[15] J. V. Hansen; and R. D. Nelson, "Neural Networks and Traditional Time Series Methods: A Synergistic Combination in State Economic Forecasts," *IEEE TRANSACTIONS ON NEURAL NETWORKS*, vol. 8, 1997.

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

[17] N. Cristianini; and J. Shawe-Taylor, *An introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge: Cambridge University Press, 2000.

[18] M. Anthony; and N. Biggs, *Computational Learning*. Cambridge: Cambridge University Press, 1992.

[19] R. Stahlbock; and S. Lessmann, "Potential von Support Vektor Maschinen im analytischen Customer Relationship Management," Universität Hamburg, Hamburg, Arbeitspapier 2004.

[20] M. Welling, "Support Vector Regression," Department of Computer Science, University of Toronto, Toronto (Kanada) 2004.

[21] J. Bi; and K. P. Bennett, "A Geometric Approach to Support Vector Regression," Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180, New York 2003.

[22] B. Schölkopf, *Support Vektor Learning*. Berlin: GMD - Forschungszentrum Informationstechnik, 1997.

[23] S. R. Gunn, "Support Vector Machines for Classification and Regression," Faculty of Engineering, Science and Mathematics, School of Electronics and Computer Science, University of Southampton 1998.

[24] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," in *Data Mining and Knowledge Discovery*, U. Fayyad, Ed. Boston: Kluwer Academic Publishers, 1998, pp. 121–167.

[25] A. Smola, "Regression Estimation with Support Vector Learning Machines," Technische Universität München, 1996.

[26] K.-R. Müller, A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik, "Predicting Time Series with Support Vector Machines," in *Advances in Kernel Methods — Support Vector Learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge: MIT Press, 1999, pp. 243-254.

[27] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," presented at Annual Conference on Computational Learning Theory, Pittsburgh (U.S.A.), 1992.

[28] C.-C. Chang; and C.-J. Lin, "LIBSVM: a Library for Support Vector Machines," National Science Council of Taiwan, Taiwan 2005.

[29] C.-H. Wu, C.-C. Wei, D.-C. Su, M.-H. Chang, and J.-M. Ho, "Travel Time Prediction with Support Vector Regression," Institute of Information Science, Academia Sinica, Taipei, Taiwan 2003.

[30] G. P. Zhang; and M. Qi, "Computing, Artificial Intelligence and Information Technology - Neural network forecasting for seasonal and trend time series," *European Journal of Operation Research*, pp. 501 - 514, 2003.

[31] S. Pietsch, "Computational Intelligence zur Absatzprognose - Eine Evaluation von Künstlichen Neuronalen Netzen und Support Vector Regression zur Zeitreihenprognose," in *Institut für Wirtschaftsinformatik*, vol. Diplomarbeit. Hamburg: Universität Hamburg, 2006.

[32] 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*, pp. 381-396, 2001.

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

[34] R. Schlittgen; and B. H. J. Streitberg, *Zeitreihenanalyse*, 8. Auflage ed. München; Wien: Oldenburg: Oldenburg Verlag, 1999.

[35] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A Practical Guide to Support Vector Classification," Department of Computer Science and Information Engineering - National Tawain University, Taipei (Taiwan) 2003.

[36] J. S. Armstrong; and F. Collopy, "Error Measures For Generalizing About Forecasting Methods: Empirical Comparisons," *International Journal of Forecasting*, pp. 69-80, 1992.

[37] K.-W. Hansmann, *Kurzlehrbuch Prognoseverfahren*. Wiesbaden: Gabler, 1983.

[38] C.-W. Hsu; and C.-J. Lin, "A Comparison of Methods for Multiclass Support Vector Machines," presented at IEEE Transactions on Neural Networks, 2002.

[39] S. F. Crone, H. Kausch, and D. Preßmar, "Prediction of the CATS benchmark using a Business Forecasting Approach to Multilayer Perceptron Modelling," presented at IJCNN'04, Budapest (Hungary), 2004.

[40] J. Faraway; and C. Chatfield, "Time Series Forecasting with Neural Networks: A Case Study," University of Bath, Research Report 1995.

[41] R. L. Goodrich, "The Forecast Pro methodology," *International Journal of Forecasting*, vol. 16, pp. 533-535, 2000.

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