Hybrid Ensemble Models in Time Series Forecasting

Joerg D. Wichard, *Member, IEEE*

***Abstract*— We propose hybrid ensembles for time series forecasting. A hybrid ensemble combines the forecasts of several different models in a weighted mean. The best performing models with respect to a left-out part of the time series are combined by taking the SMAPE prediction error as a weight of the single forecasts. We show the application of this approach in the ICTSF challenge.**

I. Introduction

Time series forecasting is a growing field of interest with applications in nearly any field of science. In many cases we have only the measurement of one system variable over a longer period, but no other quantitative information of what may influence the system of interest. In these cases we can only use the time series itself to build predictive models. In general, we don't know the system that produced the time series but we have to make reasonable assumptions concerning the system's nature. There are common tools for time series analysis that can give us valuable hints, like the autocorrelation function, recurrence plots [1], stationarity tests or linearity tests [2]. Depending on the outcome of the analysis, we have to decide how to forecast the time series. An early approach was introduced by Yule, who described a linear autoregressive (AR) model based on a single time series in order to predict the sunspot cycle [3]. Nowadays, the AR models belong to the classics of linear time series analysis (see for example [4]).

In general the conventional linear methods for modeling and forecasting fail if they are applied to time series originating from nonlinear systems. This seems to be evident because a nonlinear system should not be treated as a linear stochastic process [5].

In the framework of deterministic chaos, many methods for nonlinear time series modeling and prediction have been suggested. Most of them are based on state space reconstruction with time-lag variables or alternative methods like broomhead-king coordinates [6], [7], [8], [9], [10], [11].

To our knowledge, there is no single forecasting method, that could deal with all kind of time series *from scratch*, i.e. taking into account all the specialties of the time series like (non)-stationarity, (non)-linearity or (non)-periodicity. Therefore we like to introduce a simple forecasting strategy that combines several individual (in some sense specialized) models to a weighted mean of forecasts. The weighting is based on the SMAPE with respect to a left-out part of the time series that has the same length as the forecasting horizon. This leads to an adaptive forecasting schema, that gives a higher weight to those methods that performed well on a test set.

Joerg D. Wichard (email: joergwichard@web.de).

II. Forecasting Strategy and Models

In the first part of this section we give a brief description of the different forecasting models that we utilized in our approach. We further illustrate the training of these models that sometimes have free parameters to be optimized. In some cases we build ensembles of models for better model generalization. The second part of this section explains the combination strategy that combines the outcome of individual forecasting models into the final forecast. We combine the models in a weighted average, wherein the model weights are defined by the inverse forecasting errors on a test set. The error measure is the Symmetric Mean Absolute Percent Error (SMAPE) with respect to the last contiguous part of N samples of the time series. Let $\{x_t\}$ denote the time series and $\{y_t\}$ denote the forecasts of the model, then the SMAPE is defined as

$$\text{SMAPE} = \frac{100}{N}\sum_{t=1}^{N}\frac{\|x_t - y_t\|}{\frac{1}{2}\|x_t + y_t\|}. \tag{1}$$

A. State Space Reconstruction and Iterated Prediction

A common characteristic of several nonlinear models is the reconstruction of the system's state space based on the embedding theorems given by Takens [6], Sauer et al. [8] and the extension of Stark et al. [11]. From an equally sampled time series $\{x_t\}_{t=1,\ldots,N}$ we construct the d-dimensional state space vector

$$\vec{X}(t) = (x_{(t-\lambda(d-1))}, x_{(t-\lambda(d-2))}, \ldots, x_t), \tag{2}$$

wherein λ denotes the time lag. We consider a one step ahead prediction model $f(\vec{X})$ for time series prediction of the form

$$\begin{aligned} f : \mathbf{R}^d &\rightarrow \mathbf{R} \\ f(\vec{X}(t)) &= x_{t+1} =: y_t. \end{aligned} \tag{3}$$

We perform an *iterated prediction* of the time series wherein the predicted value y_t is used to construct the next state space vector $\vec{X}(t+1)$ which is used to predict the next time series sample y_{t+1} and so on.

B. Nearest Neighbor and Nearest Trajectory Model

The nearest neighbor models of our approach were developed to forecast time series originating from nonlinear (chaotic) systems [7], [9]. Several variations of this method have been suggested so far. A brief overview could be found in Kantz and Schreiber [12]. In particular, nearest neighbor models and nearest trajectory models are useful in the case of time series with a latent periodicity [13]. After embedding the time series in a reconstruction space with delay coordinates (see II-A) the model consist of a nonlinear mapping using a local approximation. A k-Nearest-Neighbor model takes

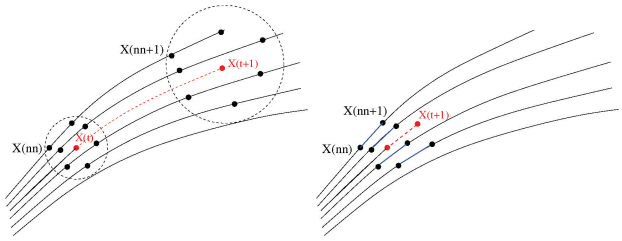

Fig. 1. Left: A nearest neighbor model takes the mean over *neighboring states* as a forecast of the next step of the time series. Right: A nearest trajectory model takes the average over the *neighboring trajectories segments* in order to forecast the next time step. Both models operate in the systems' time-lag reconstructed state space.

a weighted average of the one step ahead progression over those reconstructed states $\vec{X}_{nn}$ that are closest to the query point $\vec{X}(t)$. This leads to

$$\vec{X}(t+1) = \sum_{\vec{X}_{nn} \in N_k\{\vec{X}(t)\}} \vec{X}_{nn+1},$$

wherein $N_k\{\vec{X}(t)\}$ denotes the k-element neighborhood of $\vec{X}(t)$ defined in a given metric and $\vec{X}_{nn+1}$ is the next time step of the neighboring state $\vec{X}_{nn}$. Common choices for the metric are the L_1, L_2 and the L_∞ metric. The metric and the number of neighboring points are free parameters that we selected with cross-validation for each time series separately. The left part of Figure 1 sketches the idea of the nearest neighbor model.

The nearest trajectory model is an extension of the nearest neighbor model that was introduced by McNames [13] as the winning entry in the *K.U. Leuven time series competition* [14]. Like the nearest neighbor model it is based on the assumption that the time series stems from a dynamical system and the states can be reconstructed with a time delay embedding. A latent periodicity of the time series is sometimes a hint, that the nearest trajectory model could lead to proper forecasting results. The nearest trajectory model looks for the nearest trajectory segments in the reconstructed state space instead of the nearest neighbors (see Figure 1). The prediction is performed with a local linear model of the closest trajectory points as described in [13]. The number of neighboring trajectories is a free parameter that we select with cross-validation for each time series separately. The number of neighboring trajectories (NNTR) ranges between one and three, depending on the total length of the time series and is given in table I. The embedding dimension was $d = 70$, the time lag $\lambda = 1$ and prediction horizon $\tau = 1$ as defined in equation 3.

C. Neural Networks

We trained multilayer feed-forward neural networks with the $\tanh(\vec{x})$ as nonlinear element. This is the base model for an iterated prediction as defined in equation 3. The embedding dimension for the state space reconstruction was adopted to the different time series and the time delay was always $\lambda = 1$. For initialization, the number of hidden layers was chosen at random to be one or two and the numbers of neurons was also random (4-50 in the first layer and 3-20 in the second layer). The number of neurons and the number of hidden layers were selected by cross-validation. The training is based on the Rprop Algorithm [15], which is fast and robust. As regularization method we use the common weight decay with the penalty term

$$P(\vec{w}) = \lambda \sum_{i=1}^{N} \frac{w_i^2}{1 + w_i^2},$$

where $\vec{w}$ denotes the N-dimensional weight vector of the MLP and the regularization parameter is small $\lambda = 0.005$. Neural Networks can deal with almost all kind of time series and there are many examples of special network architectures to fulfill special tasks. However, the simple multilayer feed-forward network of this approach could be considered as a non-linear extension of an autoregressive model (see section II-F).

D. The Difference Model

We build models based on the differences $x_{diff}(t)$ of the time series

$$x_{diff}(t) = x(t) - x(t-1), \tag{4}$$

that serve as input to the three models described above: The nearest neighbor model, the nearest trajectory model and the neural network model. These models are trained in a competitive ensemble approach [16]. The final forecast of the difference model is the cumulative sum of the predicted differences, added to the last known time series sample.
Difference or return based models are a common tool in stock market analysis in order to compare different assets with respect to their relative changes [17], [18].

E. The Trend Cycle Model

The analysis of the autocorrelation function indicates, that several time series show a strong periodicity. We fitted a linear regression to the time series and added the cycle. The periodicity of the cycle was taken from the autocorrelation function of the time series under investigation. If there was no detectable periodicity, the period was set to ∞ and only the linear extrapolation was used. For the further process we define the cycle model $c_{cycle}(t)$ as the mean over all cycles plus the linear trend. The values of the periods are given in table I.

F. The Autoregressive Model

Several time series showed almost no periodicity (see table I). We considered them as outcome of a random process and included therefor a common autoregressive (AR) model. The AR model is a linear prediction that attempts to predict the output of a system based on the previous outputs [3]. For the sake of simplicity, the dimension of the AR model was the same as the dimension d of the nonlinear model as reported in table I.

III. Hybrid Ensemble Models

It is well known, that the generalization abilities of predictive models could be improved by ensemble building (also known as model averageing or query by commitee). In the neural network community, this approach was developed and improved by Hansen and Salamon [19], Sueng et al. [20], Geman et al. [21], Perrone and Cooper [22] and Krogh and Vedelsby [23]. The ensemble approach is not restricted to neural networks but works also for several other model classes [24], [16]. An ensemble is the average output of several different models $f_i(\mathbf{x})$

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{K} \phi_i f_i(\mathbf{x}), \tag{5}$$

wherein we assume that the model weights ϕ_i sum to one $\sum_{i=1}^{K} \phi_i = 1$. The generalization error of an ensemble is in general lower than the mean of the generalization error of the single ensemble members [23] and this holds in general, independent of the model class under investigation. The generalization error of an ensemble model could be improved if the single models on which averaging is done *disagree* in the sense that the model output is uncorrelated [25], [26]. There are several ways to introduce uncorrelated output of the individual ensemble members. A general approach is to train various models on selected subsets of the training data [23], [27] or to use different parameter settings for initialization. The models for the ensemble are selected in a cross-validation scheme (see [28] for a detailed discussion of the method). The ensembles for the difference model, the neural networks, the nearest neighbor model and the nearest trajectory model were rather small with $K = 15$ members. They were simply averaged according to equation 5 with equal weights $\phi_i = 1/K$.

IV. Data Analysis, Preprocessing and Parameters

This initial step in forecasting is usually a fundamental analysis and visual inspection of the data, in order to get an idea what the data looks like and to estimate the required model parameters. The fundamental analysis includes to:

- Plot the time series
- Find and remove missing values and eliminate non-numerical values like Inf and NaN
- Caculate mean, variance, autocorrelation function (acf), spectral density (see [12] for definitions) and difference values (defined in equation 4)
- Perform a non-linearity test (see [2] for details)

The outcome of this analysis influenced the choice of the modelling parameters as reported in table I. In the case of periodic time series, we identify the intrinsic frequency of the cycle model by the first non-trivial maximum of the acf. The dimension d for the state space reconstruction was a multiple of the intrinsic frequency or set to be the length of the forecasting horizon. The time lag was set to $\lambda = 1$ for all time series. The number of nearest neighbors ranges from 1 to 3 and was adapted to the length of the time series.

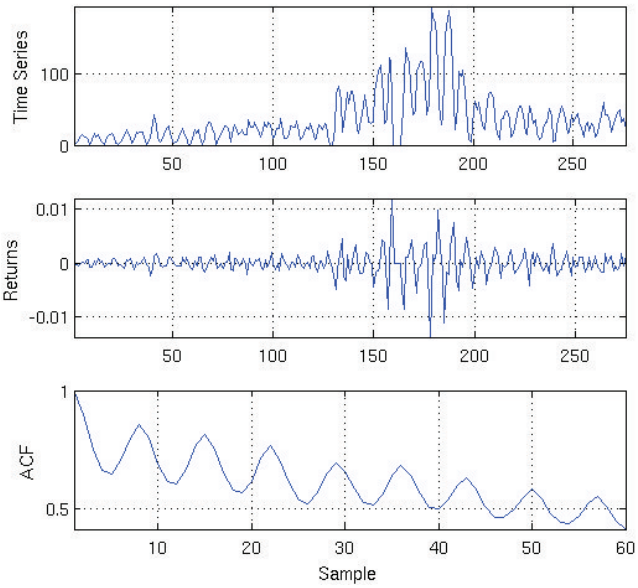

Fig. 2. The figure shows the 4th time series of the challenge, the returns and the autocorrelation function. The acf indicates an intrinsic frequency of 7 samples.

V. Combining Forecasts

We combined the output of different models in order to improve the quality and stability of the forecast. The final forecasts that entered the ICTSF challenge were generated as follows:

- The last N samples of the time series were kept out from model training and were used as test set, wherein N was the prediction horizon
- The forecasts of the nearest trajectory model, the difference model, the neural network ensemble, the trend cycle and the auto regressive model were compared with the test set and the forecasting error E_j was computed following equation 1
- The model weights $\omega_j = 1/E_j$ were chosen inversely proportional to the forecasting error with the constraint $\sum_j \omega_j = 1$
- In the final step, we trained each model on the full data set (adding the former test set to the training data) and built the combined forecast by taking the sample-wise mean over all individual models

$$\hat{y}_t = \sum_{j=1}^{5} \omega_j y_t^j. \tag{6}$$

The weights of the individual models are reported in Table I. The weighted average increases the robustness of the approach and gives the more appropriate models an increased vote to the final forecast.

VI. Main Results

We described a framework for forecasting time series with hybrid ensembles. We used several forecasting methods that were trained separately on a training set and combined with

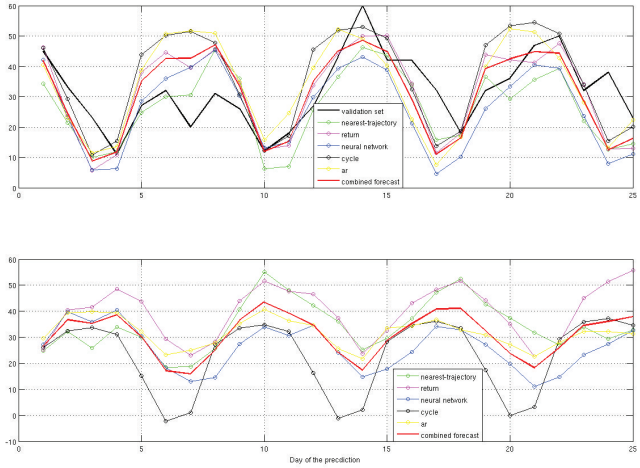

Fig. 3. The figure on top shows the outcome of the individual models (nearest trajectory model, difference model, neural network, trend cycle and auto regressive model) for the left-out part of the time series (number 4 in the challenge). The SMAPE of these models defines the weight for the combined forecast. The final outcome of this approach that entered the ICTSF challenge is shown on the bottom (solid red line).

TABLE I

THE MODEL PARAMETERS FOR THE TIME SERIES 1-8.

TS	1	2	3	4	5	6	7	8
Length	893	133	27	276	1031	609	73	133
Horizon	150	15	7	25	200	50	20	15
Lag λ	1	1	1	1	1	1	1	1
Dim d	21	15	7	25	145	21	20	15
Period	7	∞	∞	7	24	7	∞	∞
NNTR	3	1	1	2	3	3	1	1
ω_{traj}	0.15	0.1	0.1	0.2	0.26	0.18	0.19	0.13
ω_{net}	0.15	0.1	0.39	0.17	0.28	0.16	0.19	0.05
ω_{diff}	0.16	0.22	0.12	0.22	0.13	0.29	0.27	0.35
ω_{cycle}	0.36	0.49	0.21	0.22	0.19	0.23	0.21	0.40
ω_{ar}	0.18	0.1	0.2	0.19	0.14	0.14	0.14	0.07

respect to their performance on a test set. The parameters for different time series are given in table I. We investigated the autocorrelation function of the time series in order to detect the intrinsic periodicity and the the embedding dimension d. The final hybrid ensemble model consists of several individual models: A nearest neighbor/trajectory ensemble model, a feed foreward neural network ensemble, a trend cycle model, an autoregressive model and an ensemble model based on the differences of the time series.

REFERENCES

[1] J.-P. Eckmann, S. O. Kamphorst, and D. Ruelle, "Recurrence plots of dynamical systems," *EPL (Europhysics Letters)*, vol. 4, no. 9, p. 973, 1987.

[2] J. Theiler, S. Eubank, A. Longtin, B. Galdrikian, and J. Farmer, "Testing for nonlinearity in time series: the method of surrogate data," *Physica D*, vol. 58, pp. 77–94, 1992.

[3] U. Yule, "On a method of investigating periodicities in disturbed series with special reference to wolfer's sunspot numbers," *Philos. Trans. Roy. Soc. Series A*, vol. 226, pp. 267–298, 1927.

[4] P. Brockwell and R. Davis, *Time Series : Theory and Methods*, 2nd ed. New York, Inc.: Springer-Verlag, 1991.

[5] M. Priestley, *Non-linear and Non-stationary Time Series Analysis*. London: Academic Press, 1988.

[6] F. Takens, "Detecting strange attractors in turbulence," in *Dynamical Systems and Turbulence*, ser. Lect. Notes Math. Berlin: Sringer-Verlag, 1981.

[7] J. Farmer and J. Sidorowich, "Predicting chaotic time series," *Phys. Rev. Lett.*, vol. 59(8), pp. 845 – 848, 1987.

[8] T. Sauer, J. Yorke, and M. Casdagli, "Embedology," *J.Stat.Phys.*, vol. 65, pp. 579–618, 1991.

[9] M. Casdagli, "Nonlinear prediction of chaotic time series," *Physica D*, vol. 35, pp. 335–356, 1989.

[10] K. Stokbro and D. Umberger, "Forecasting with weighted maps," in *Nonlinear Modeling and Forcasting*, M. Casdagli and S. Eubank, Eds. Addison Wesley, 1990.

[11] J. Stark, D. Broomhead, M. Davies, and J. Huke, "Takens embedding theorems for forced and stochastic systems," *Nonlinear Analysis*, vol. 30, pp. 5303–5314, 1997.

[12] H. Kantz and T. Schreiber, *Nonlinear time series analysis*. Cambridge UK: Cambridge University Press, 1997.

[13] J. McNames, "A nearest trajectory strategy for time series prediction," in *Proceedings of the International Workshop on Advanced Black-Box Techniques for Nonlinear Modeling*. K. U. Leuven Belgium, 1998.

[14] J. Suykens and J. Vandewalle, "The K.U.Leuven competition data: A challenge for advanced neural network techniques," in *Proc. of the European Symposium on Artificial Neural Networks (ESANN'2000)*, Bruges, Belgium, 2000, pp. 299–304.

[15] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," in *Proc. of the IEEE Intl. Conf. on Neural Networks*, San Francisco, CA, 1993, pp. 586–591.

[16] J. Wichard and M. Ogorzałek, "Time series prediction with ensemble models applied to the CATS benchmark," *Neurocomputing*, vol. 70, no. 13-15, pp. 2371–2378, August 2007.

[17] R. Mantegna and H. Stanley, *An Introduction to Econophysics: Correlations and Complexity in Finance*, 1st ed. Cambridge, England: Cambridge University Press, 2000.

[18] J. Wichard, M. Ogorzałek, and C. Merkwirth, "Detecting correlation in stock markets," *Physica A*, vol. 344, pp. 308–311, 2004.

[19] L. Hansen and P. Salamon, "Neural network ensembles," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[20] H. S. Seung, M. Opper, and H. Sompolinsky, "Query by committee," in *COLT '92: Proceedings of the fifth annual workshop on Computational learning theory*. New York, NY, USA: ACM, 1992, pp. 287–294.

[21] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural Computation*, vol. 4, pp. 1–58, 1992.

[22] M. P. Perrone and L. N. Cooper, "When Networks Disagree: Ensemble Methods for Hybrid Neural Networks," in *Neural Networks for Speech and Image Processing*, R. J. Mammone, Ed. Chapman-Hall, 1993, pp. 126–142.

[23] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation, and active learning," in *Advances in Neural Information Processing Systems*, G. Tesauro, D. Touretzky, and T. Leen, Eds., vol. 7. The MIT Press, 1995, pp. 231–238.

[24] G. Valentini and T. Dietterich, "Bias-variance analysis and ensembles of svm," in *Third International Workshop on Multiple Classifier Systems*, ser. Lecture Notes in Computer Science, J. Kittler and F. Roli, Eds. New York: Springer Verlag, 2002, vol. 2364, pp. 222–231.

[25] A. Krogh and P. Sollich, "Statistical mechanics of ensemble learning," *Physical Review E*, vol. 55, no. 1, pp. 811–825, 1997.

[26] U. Naftaly, N. Intrator, and D. Horn, "Optimal ensemble averaging of neural networks," *Network, Comp. Neural Sys.*, vol. 8, pp. 283–296, 1997.

[27] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[28] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. Springer-Verlag, 2001.