Forecasting seasonal time series with computational intelligence: contribution of a combination of distinct methods.

M. Štěpnička,[1] J. Peralta,[2] P. Cortez,[3] L. Vavříčková,[1] G. Gutierrez[2]

[1]Institute for Research and Applications of Fuzzy Modeling, University of Ostrava, Ostrava, Czech Republic
[2]Computer Science Department, Carlos III University, Madrid, Spain
[3]Department of Information Systems/Algoritmi Centre, University of Minho, Guimarães, Portugal

Abstract

Accurate time series forecasting are important for displaying the manner in which the past continues to affect the future and for planning our day to day activities. In recent years, a large literature has evolved on the use of computational intelligence in many forecasting applications. In this paper, several computational intelligence techniques (genetic algorithms, neural networks, support vector machine, fuzzy rules) are combined in a distinct way to forecast a set of referenced time series. Forecasting performance is compared to the a standard and method frequently used in practice.

Keywords: Time series, Computational intelligence, Neural networks, Support vector machine, Fuzzy rules, Genetic algorithm

1. Introduction

Time Series Forecasting (TSF) predicts the behavior of a given phenomenon based solely on the past patterns of the same event. Several TSF (mainly statistical) methods were developed, e.g., Holt-Winters or Box-Jenkin's ARIMA. More recently, several Computational Intelligence (CI) techniques have been proposed for TSF [1]. Let us recall, that CI denotes an Artificial Intelligence branch that relies on heuristic algorithms that were inspired in biological and natural intelligence. Examples of CI applied to TSF include: Artificial Neural Networks (ANN) [2], evolutionary computation [3], Support Vector Machines (SVM) [4], immune systems [5], fuzzy techniques [6], or their combinations [7, 8].

In this paper, we present three recent *) CI approaches for seasonal TSF: *Automatic Design of Artificial Neural Networks* (ADANN), SVM with time lag selection based on a sensitivity analysis procedure; and *Fuzzy Linguistic Approach* to the trend-cycle analysis and forecasts. Further, we propose novel hybrid combinations of the CI methods, such that the fuzzy approach for the trend-cycle forecasts is complemented by the earlier two approaches that forecast seasonal components of the decomposition. All these CI variants are compared to the ARIMA methodology [9].

*)Separately proposed in the 2010 IEEE World Congress on Computational Intelligence (WCCI), under the special session "Computational Intelligence in Forecasting".

2. Related work and motivation

2.1. Automatic design of neural networks

ANNs provide a methodology for solving many types of nonlinear problems that are difficult to solve by traditional techniques. Often, time series processes exhibit temporal and spatial variability, and are suffered by issues of nonlinearity of physical processes, conflicting spatial and temporal scale and uncertainty in parameter estimates. ANNs are flexible models that have the capability to learn the underlying relationships between the inputs and outputs of a process, without needing the explicit knowledge of how these variables are related.

Generally, problem related to the use of ANN is its design appropriately chosen for a particular time series. Besides the decision concerning particular type of ANN that will solve the TSF task and the learning algorithm, one has to make further choices concerning the neural network architecture, initializing weights vector and the learning rate. Several works employ distinct methods to obtain an ANN design in an automatic way. We follow those that directly involve evolutionary techniques [10, 11, 12].

2.2. Support vector machines

SVM is a powerful learning tool that is based on a statistical learning theory developed in the 1990s by Vapnik et al. [13]. SVM is based on two key concepts: it transforms the input variables into a high dimensional feature space (using a kernel function) and then the algorithm finds the best hyperplane that models the data in this feature space.

The motivation of using SVM for forecasting is the same as in ANN: both are flexible models. By this we mean, that no *a priori* restriction is imposed in comparison to classical TSF methods, and presenting nonlinear learning capabilities.

Similarly to ANN, the variable (e.g. a time lag) selection is useful to discard irrelevant time lags

which leads to simpler models that are easier to interpret and that usually give better performances [14, 15] and hence, it is a critical issue. Also, SVM has further hyperparameters (kernel parameter) that need to be adjusted [16]. In this work, we address this crucial issue.

2.3. Linguistic approach

So far, a notable number of works aiming at fuzzy approach to TSF has been published [6, 7, 17]. Most of them either use Takagi-Sugeno rules or various neuro-fuzzy approaches that lie on the border between neural networks, Takagi-Sugeno models and evolving fuzzy systems.

However, the Takagi-Sugeno rules use functional consequents without any linguistic meanings and do not employ any kind of logical implication; evolving system usually well tune (Gaussian) fuzzy sets to have a center, say, at node 5.6989 and the width parameter equal to 2.8893 (see [17]). Hence, most of such fuzzy approaches, although very powerful, disregarded the importance of interpretability. Motivated by this lack, we deal with the linguistic approach introduced in [18, 19].

3. Time series data and evaluation

3.1. Time series datasets

Let $\{y_t \mid t = 1, \ldots, T\} \subset \mathbb{R}$ be the past values (usually called as *in-samples*) of a given time series. Our TSF task is to built a model that analyzes the in-samples in order to forecast the so-called *out-of-samples:*

$$\{F_t = y_t - e_t \mid t = T+1, \ldots, T+h\} \subset \mathbb{R}, \quad h \geq 1$$

where e_t denotes the forecasting error that should be minimized according to an accuracy measure and h denotes the forecasting horizon. We assume that only in-sample data is used to build such TSF model. In case of $h > 1$, either the model directly outputs multi-step ahead forecasts or the out-of-samples are forecasted iteratively by using 1-ahead forecasts (and the remaining up to the $h-1$ predicted values) as inputs of the model [20].

In this work, we address seasonal data, since we believe multi-step forecasts are particularly useful for these type of series. To compare the proposed TSF methods, we selected 6 benchmark time series (Table 1). Five of them: **passengers** (numbers or airlines passengers in thousands); **pigs** (numbers of pigs slaughtered in Victoria); **cars** (car sales in Quebec); **abraham12** (gasoline demand at Ontario) and **abraham14** (U.S. houses sales data of in thousands); are monthly series from the well-known Hyndman's *Time Series Data Library* [21]. All these five datasets contain real-world data from different areas, which makes them interesting to forecast. The last series, called **mackey-glass**, is a chaotic series based on the Mackey-Glass differential equation and it is widely regarded as a benchmark for comparing the generalization ability of different methods. It has been chosen in order to extend the experimental datasets by a different kind (not monthly; not real-world; no trend, nor noise) of a benchmark.

3.2. Evaluation

The global performance of a forecasting model is evaluated by an accuracy measure, such as Mean Absolute Error (MAE), (Root) Mean Squared Error ((R)MSE), Symmetric Mean Absolute Percentage Error (SMAPE) or Mean Absolute Scaled Error (MASE).

Historically very popular (R)MSE is quite sensitive to outliers [22] and furthermore, both MSE and MAE are scale-dependent measures and cannot be used for a comparison across more time series.

Both SMAPE and MASE:

$$\text{SMAPE} = \frac{1}{h} \sum_{t=T+1}^{T+h} \frac{|e_t|}{(|y_t| + |F_t|)/2} \times 100\%, \quad (1)$$

$$\text{MASE} = \frac{1}{h} \sum_{t=T+1}^{T+h} \frac{e_t}{\frac{1}{T-1}\sum_{j=2}^{T} |y_j - y_{j-1}|}, \quad (2)$$

where $e_t = y_t - F_t$, have the advantage of being scale independent. Thus can be used to compare methods across different time series. However, there are still two SMAPE drawbacks [23]: the denominator may be close to zero, and a heavier penalty is given to under-forecasting when compared to over-forecasting. More recently proposed [23] MASE does not hold the SMAPE disadvantages. When MASE > 1, the forecasts are worse (on average) when compared with the in-sample one-step forecasts of the naïve random-walk method. In other words MASE compares the average out-of-sample e_t with the average in-sample first difference and hence, it realitivizes the prediction error with respect to time series fluctuations from the past.

Table 1: Time series seasonal period and in-sample/out-of-sample sizes. Symbols denote: K = Seasonal period; T = #in-samples; h_2 = #out-of-samples.

Series	K	T	h_2
passengers	12	120	24
pigs	12	164	24
cars	12	84	24
abraham12	12	168	24
abraham14	12	108	24
mackey-glass	30	731	60

For the comparison, we opted to compute SMAPE and MASE metrics for two distinct fore-

casting horizons: $h_1 = K$ and $h_2 = 2K$, where K is the seasonal period, see Table 1.

4. Forecasting methods

4.1. ARIMA by ForecastPro®

As a baseline comparison, we choose the popular ARIMA methodology [9] because: a) ARIMA is a common method widely used in practice; b) similary to ARIMA, all the CI approaches discussed below are of an autoregressive nature.

Note, that in order to avoid any bias from a naive implementation of ARIMA, we adopted the ForecastPro® (FP) professional forecasting software In particular, we fed the tool with the in-samples and executed the full automatic parameter selection of ARIMA to obtain the forecasts.

4.2. Automatic Design of Artificial Neural Networks (ADANN)

In order to obtain a single ANN with logistic activation functions (values within the range [0,1]), a given time series has to be normalized. After fitting the ANN, the inverse process is carried out. The time series in-samples are transformed into a pattern set with I inputs. Only one neuron was chosen at the output layer and multi-step forecasts are performed using an iterative feedback of the previous forecasts [15]. Therefore, the time series will be transformed into a patterns set, where each pattern will consists of:

$$(N_{t-I}, \ldots, N_{t-2}, N_{t-1}) \rightarrow N_t$$

where all N_t values correspond to the normalized y_t ones. This pattern set is used to train and validate each ANN generated during the GA execution. Thus, the data is split into training (with the first X% data) and validation sets (with the remaining patterns from the in-samples).

The search for the best ANN design can be performed by a GA [24] using exploitation and exploration.

In this work, we use a Multilayer Perceptron (MLP) neural network with one hidden layer and Backpropagation (BP) as the learning algorithm. A direct encoding schema for full connected MLP is considered. For this encoding scheme the information placed into the chromosome is: two genes (decimal digits) to codify the number of inputs nodes (I); two genes for the number of hidden nodes (H); two genes for the learning factor (α); and the last ten genes for the initialization seed (s, "long int" type) value of the connection weights. This way, the values of I, H, α and s are obtained from the chromosome as follows:

$$\begin{aligned}
\textbf{chromosome} &= g_{I_1} g_{I_2} g_{h_1} g_{h_2} g_{\alpha_1} g_{\alpha_2} g_{s_1} \cdots g_{s_{10}}, \\
s &= g_{s_1} g_{s_2} \cdots g_{s_{10}}, \\
I &= 10 g_{I_1} + g_{I_2}, \\
H &= min(2 * I, 10 g_{H_1} + g_{H_2}), \\
\alpha &= (10 g_{\alpha_1} + g_{\alpha_2})/100.
\end{aligned}$$

The search process (GA) will consist of the following steps :

1. A population (a set of chromosomes) is randomly generated.
2. The phenotypes (ANN architectures) and fitness value of each individual of the actual generation is obtained. To obtain the phenotype associated to a chromosome and its fitness value:
   (a) The phenotype of an individual of the actual generation is first obtained using the SNNS tool [25].
   (b) Then for each NN i, training and validation pattern subsets are obtained from time series data depending on the number of inputs nodes of network i.
   (c) The net is trained with BP (using SNNS). The architecture (topology and connections weights set) of the net is saved (i.e. early stopping) when the validation error is minimum during the training process. Thus, this architecture is the final phenotype of the individual.
3. Once the fitness value for whole population is available, the GA operators (Elitism, Selection, Crossover and Mutation) are applied in order to generate the population of the next generation.
4. The steps 2 and 3 are iteratively executed till a maximum number of generations is reached.

The GA parameters were set to: population size, 50; maximum number of generations, 100; percentage of the best individual that stay unchangeable to the next generation (percentage of elitism), 10%; crossover: parents are split in one point randomly selected, offspring are the mixed of each part from parents; mutation probability will be one divided between the length of the chromosome (1/lengthchrom = 1/16 = 0.07), and it will be carried out for each gene of the chromosome.

4.3. SVM - Support Vector Machine

Any regression algorithm, e.g., ANN or SVM, can be applied to TSF by adopting a sliding time window, defined by the set of time lags $\{k_1, k_2, \ldots, k_I\}$, that is used to build a forecast.

For a given time period t, the model inputs are $\mathbf{y} = (y_{t-k_I}, \ldots, y_{t-k_2}, y_{t-k_1})$ and the desired output is y_t. For example, let us consider the series $6_1, 10_2, 14_3, 18_4, 23_5$ (y_t values). If the $\{1, 3\}$ window is adopted, then two training examples can be created: $(6, 14) \rightarrow 18$ and $(10, 18) \rightarrow 23$.

In SVM regression [26], the input ($\mathbf{y}$ with domain Y) is transformed into a high m-dimensional feature space ($\Im$), by using a nonlinear mapping $\phi : Y \rightarrow \Im$ that does not need to be explicitly known but that depends on a kernel function $K(x, x') =< \phi(x), \phi(x') >$, where $< u, v >$ denotes the inner product of vectors u and v. Then, the SVM algorithm finds the best linear separating hyperplane tolerating a small error ε when fitting the data in the feature space:

$$\hat{y}_{t,t-1} = w_0 + \sum_{i=1}^{m} w_i \phi(\mathbf{y}) \quad (3)$$

where $w_i \in \Re$ are weights. The ε-insensitive loss function sets an insensitive tube around the residuals and the tiny errors within the tube are discarded.

We adopt the popular gaussian kernel: $K(x, x') = exp(-\lambda||x - x'||^2)$, $\lambda > 0$, which presents less parameters than other, e.g., polynomial, kernels [27]. The SVM performance is affected by three parameters: λ, ε and C (a trade-off between fitting the errors and the flatness of the mapping). To reduce the search space, the first two values are set using the heuristics [28]: $C = 3$ (for a standardized output) and $\varepsilon = \hat{\sigma}/\sqrt{N}$, where $\hat{\sigma} = 1.5/N \times \sum_{i=1}^{N}(y_i - \hat{y}_i)^2$ and $\hat{y}_i$ is the value predicted by a 3-nearest neighbor algorithm.

Further, the forecasting performance is affected by time lag and model selections while a better generalization is achieved if only relevant time lags are fed into the model [14]. The kernel parameter λ produces the highest impact in the SVM performance, in comparison to C or ε.

Sensitivity analysis [29] is a procedure that is applied after the training phase and analyzes the model responses when the inputs are changed. Let $\hat{y}_{t-k}(j)$ denote the output obtained by holding all input variables at their average values except y_{t-k}, which varies through its entire range with $j \in \{1, \ldots, L\}$ levels. If a given input variable y_{t-k} is relevant then it should produce a high variance V_k. Thus, its relative importance R_k can be given by:

$$\begin{aligned} V_k &= \textstyle\sum_{j=1}^{L} (\hat{y}_{t-k}(j) - \overline{\hat{y}_{t-k}}(j))^2/(L-1), \\ R_k &= V_k / \textstyle\sum_{i=1}^{I} V_i \times 100\ (\%). \end{aligned}$$

This is a simple procedure that only measures single input variance and not interactions of inputs. Yet, even with this limitation, this fast procedure has outperformed other more sophisticated algorithms for the input variable selection [29].

We propose a simultaneous variable and model selection procedure for multi-step ahead forecasting. The method starts with a maximum of I_{max} time lags and iteratively deletes one input until there are no time lags. The sensitivity analysis is used to select the least relevant lag to be deleted in each iteration, allowing a reduction of the computational effort by a factor of I_{max} when compared to the standard backward selection procedure. Before feeding the SVM, all variables are standardized to a zero mean and one standard deviation. After the training, the SVM outputs are post-processed with the corresponding inverse scaling function. During a given iteration, a grid search is used to find the best model hyperparameter $\gamma \in \{2^{-15}, 2^{-13}, \ldots, 2^{1}\}$. The training data is divided into training (2/3 of in-samples) and validation sets (1/3 of in-samples). After the variable and model selection phase, the final model is retrained using all in-samples. Multi-step forecasts are built iteratively by using 1-ahead predictions as inputs [20].

The SVM experiments were conducted using the **rminer** library[30] of the R tool, which adopts the Sequential Minimal Optimization algorithm to fit the model. In this work, we set $L = 6$ [29] and $I_{max} = K + 1$. The intention is to include all up to the seasonal lag plus an additional one that may be relevant for trended series.

4.4. Linguistic approach

The main idea is as follows. First, a time series is decomposed into the so called *trend-cycle* and the *seasonal component* using the *fuzzy transform*[31]. Second, the trend-cycle is described by the *linguistic description* [32] (fuzzy rule base) comprised from fuzzy rules. The fuzzy rules, as special sentences of a natural language, describe the data generating process autoregressively in an interpretable form.

Finally, a model of the seasonal components is determined and used to forecast these components. Both forecasted components are composed together to obtain the time series forecasts.

The *fuzzy transform* (abb. F-transform) [31, 33, 34] is a specific fuzzy approximation technique that transforms a given function defined on a real interval (universe) into a simpler space of n-dimensional vectors $\mathbb{R}^n$, and then transforms it back.

First, *fuzzy partition* (consisting of *basic functions*) of the universe is constructed. The basic functions are fuzzy sets A_i such that $A_i(x) > 0$ for $x \in (c_{i-1}, c_{i+1})$ and $A_i = 0$ elsewhere, where c_i are usually equidistantly distributed nodes on the universe.

Given a fuzzy partition, the *(direct) F-transform* of function f is defined as a vector of the *components of the F-transform* where each component is determined as a center of gravity of function values above (weighted by) the corresponding basic function. If the function is given only by a set of samples the integrals in the gravity formula are replaced by finite sums. This is also the case of a time series which is viewed as a discrete function $y(t)$ given at

$t = 1, \ldots, T$. Then an appropriate fuzzy partition of the interval $[1, T]$ is constructed and the fuzzy transform components determined:

$$Y_i = \frac{\sum_{t=1}^{T} y(t) A_i(t)}{\sum_{t=1}^{T} A_i(t)}, \qquad i = 1, \ldots, n. \tag{4}$$

The *inverse transform*, converts the direct F-transform vector into another continuous function that approximates the original one, that is in case of a time series:

$$y_{F,n}(t) = \sum_{i=1}^{n} Y_i A_i(t).$$

A given time series is decomposed into a trend-cycle and seasonal components. The inverse F-transform serves as a model of its trend-cycle and the seasonal component S_t that determines the de-trended time series, is given by $S_t = y_t - y_{F,n}(t)$.

To forecast the trend-cycle, it is sufficient to forecast future F-transform components and to compute the inverse F-transform for $t+1, \ldots, t+h$. The F-transform component evolvement may be analyzed and described using autoregressive fuzzy rules (linguistic description) and as their first- and/or second-order differences. This leads to a linguistic description comprised of (automatically generated [35] implemented in LFLC2000 software [36]) fuzzy rules such as:

$$\begin{aligned} &\text{IF } \Delta Y_{i-1} \text{ is } \mathcal{B}_{\Delta i-1} \text{ AND } Y_i \text{ is } \mathcal{B}_i \\ &\text{THEN } \Delta Y_{i+1} \text{ is } \mathcal{C}_{\Delta i+1} \end{aligned} \tag{5}$$

describing the autoregressive nature of the trend-cycle. Expressions $\mathcal{B}, \mathcal{C}$ are *evaluative linguistic expressions*, such as *very big, extremely small* or *roughly medium*.

These expressions are built on the basic trichotomy *small, medium, big* using specific adverbs called linguistic hedges (*extremely, significantly, very, more or less, roughly, quite roughly, very roughly*) that either widen or narrow their meaning. For further details we refer to [37, 19].

Using an inference mechanism and a defuzzification method, the linguistic rules such as (5) may be successfully used to forecast future F-transform components. In case of the use of the evaluative linguistic expressions, *perception-based logical deduction* (PbLD)[38] and *defuzzification of evaluative expressions* (DEE) are the appropriate ones, see [18].

Given an input, *perception* selects the most fired rule(s) according to the degree up to which the antecedents are fulfilled. If there are more than one such antecedents (rules fired to the same degree), the most specific antecedent(s) (linguistic hedges play a crucial role in the determinancy of the specificity) is/are chosen and the respective fuzzy rule(s) is/are fired.

After the perception procedure, the Łukasiewicz fuzzy implication is applied to deduce a conclusion that is a fuzzy sets. Note, that the Łukasiewicz fuzzy implication is adopted from the original PbLD but it may be replaced by other appropriate fuzzy implications. If more rules than one are fired, their consequences are aggregated by the minimum operation. In order to forecast the future F-transform component that is a crisp number, the DEE defuzzification [37] is employed.

4.5. Combination of CI techniques

While ADANN and SVM approaches introduced above aim at modeling and forecasting entire time series, the linguistic fuzzy approach focuses only on modeling and forecasting the trend-cycle of a given time series. This means that seasonal components have to be forecasted separately and composed together with the trend-cycle forecast. Seasonal components may be predicted by arbitrary approriate TSF technique. Given the scope of this paper, we found natural to propose the use of CI approaches, i.e. the use of ADANN and SVM, leading to two novel fuzzy hybrids, termed here Fuzzy ANN (FANN) and Fuzzy SVM (FSVM).

5. Results

5.1. Accuracy

Further, we follow the idea of the combination of the linguistic approach and both CI approaches – ADANN and SVM. Let us recall that the linguistic approach is used to model and forecast trend-cycles of the given time series while the latter two approaches model the de-trended time series. Finally, both components are composed together and the forecast accuracy is compared to ARIMA implemented in ForecastPro® (FP) software serves as a comparison benchmark.

Series	h_1			h_2		
	FP	FANN	FSVM	FP	FANN	FSVM
pass.	**1.24**	1.73	1.99	1.60	**0.79**	0.83
pigs	**0.64**	0.68	0.73	**0.76**	1.04	1.15
cars	**0.54**	0.82	0.87	**0.66**	0.83	0.90
ab.12	1.12	0.71	**0.59**	1.27	**1.17**	1.19
ab.14	**1.39**	3.41	2.71	**1.19**	2.25	3.23
m.g.	1.30	0.17	**0.14**	1.48	0.48	**0.44**
Avg.	**1.04**	1.25	1.17	1.16	**1.09**	1.29

Table 2: Comparison of FANN, FSVM and FP performance measured by MASE. Best values in **bold**, "Avg." denotes the average.

Observing Tables 2 and 3, one can again see that there is a slight difference between comparisons based on MASE and SMAPE. Although the differences are not fundamental and that we may observe tendencies and potential of success in performing the TSF task for particular methods, no direct conclusion in the sense of the order of the above introduced methods can be stated.

Series	h_1			h_2		
	FP	FANN	FSVM	FP	FANN	FSVM
pass.	**6.5**	9.7	11.7	8.0	**4.0**	**4.0**
pigs	**6.1**	6.5	7.0	**7.1**	10.0	11.0
cars	**7.4**	12.8	13.3	**9.1**	12.2	13.1
ab.12	5.5	3.5	**2.9**	6.2	**5.7**	5.8
ab.14	**15.0**	29.9	26.4	**12.7**	19.1	28.6
m.g.	22.7	3.1	**2.6**	26.2	9.6	**8.7**
Avg.	**10.5**	10.9	10.7	11.6	**10.1**	11.9

Table 3: Comparison of FANN, FSVM and FP performance measured by %SMAPE'. Best values in **bold**, "Avg." denotes the average.

In other words, one cannot say that one of the methods clearly outperforms the others or is significantly worse that the others. As we may see, ARIMA seems to be performing the best in most of the cases. This hypothetical conclusion could be supported even on average, taking into account the shorter horizon h_1. However, the performance of FANN on h_2 is better on average than the one provided by FP. Finally, one cannot even conclude that FSVM is the worst one since it performs better than FANN on the shorter horizon h_1. Furthermore, the differences between the average performances of particular methods are negligible.

It may be noted that **abraham14** series is most responsible for the higher errors in case of FANNN and FSVM while **mackey-glass** does such an "unwanted job" for FP which supports the well-known no free lunch theorem. So, we may conclude that both combinations are at least comparable to the standard ARIMA and seasonal ARIMA methods that are widely used in practice and that all method may be complementary and appropriate for a joint use in ensemble techniques.

5.2. Interpretability

Interpretability is generally assumed to be a key feature and advantage of fuzzy models. However, this aspect of fuzzy models is sometimes overused and has thus become a kind of cliche. Hence, this feature deserves a bit deeper discussion.

Undoubtedly, there is a significant difference between rather numerically oriented fuzzy models such as the T-S rules and fuzzy rules with fuzzy sets that interpret antecedents as well as consequents. But even in the latter models, there are fundamental differences, e.g., a misleading interpretation of fuzzy rules, such as claiming Mamdani-Assilian rules as fuzzy IF-THEN rules, although their meaning is rather different [40, 41]. Even if the interpretation is correct, some treatments with distinct parameters and labels may lead to something that is very far from anything that may be called "linguistic".

By this we mean well-tuned fuzzy models constructed with help of various tuning strategies leading to black-box functions that disregard the importance of interpretability, as stated in. Let us recall the crucial idea from [42]: "*one may argue that proper input-output behavior is the central goal of automatic tuning. To some extent, this is true; however, this is not the primary mission of fuzzy systems.*"

There is no doubt that the precision of forecasts is the key issue. Nevertheless, we have to keep in mind the motivation behind using a fuzzy models. The goal is an interpretable model that does not necessarily "*leads to a painful loss of accuracy*" [42].

To underline interpretability and the linguistic nature of evaluative expressions and the used fuzzy IF/THEN rules, we present two rules from one of the generated models. Let us consider the **cars** time series:

$$\begin{aligned}&\text{IF } Y_i \text{ is ML Sm AND } \Delta Y_i \text{ is -ML Me}\\&\text{THEN } \Delta Y_{i+1} \text{ is } -\text{Me},\end{aligned} \tag{6}$$

$$\begin{aligned}&\text{IF } Y_i \text{ is Si Bi AND } \Delta Y_i \text{ is Me}\\&\text{THEN } \Delta Y_{i+1}\text{-VR Bi}.\end{aligned} \tag{7}$$

As we may see, the rules are purely linguistically built and they do not contain artificial fuzzy sets related to anonymous expression denoted as $\mathcal{A}_{ij}$. Thus, every fuzzy rule can indeed be taken as a sentence in natural language. For instance, fuzzy rule (6) may be read as follows:

If the number of cars sold in the current year is more or less small and the biannual sales increment is negative with more or less medium strength, then the upcoming biannual increment will be negative and have medium strength.

It may be understood as: given more or less small sales and a decreasing trend of a more or less medium strength from the last biannual observation, the decrease will not finish but will continue at a medium strength. Similarly, fuzzy rule (7) may be read as follows:

If the number of cars sold in the current year is significantly large and the biannual increment shows medium strength, then the next biannual increment will be negative and very roughly large.

This may be understood as: given significantly large sales with an increasing, medium-strength trend, signalizes a sort of saturation that will be followed by a very roughly large decrease in sales.

We claim, that such readable information is an additional value that might be very helpful for further decision-making and management processes.

6. Conclusions

We have introduced two hybrid CI methodologies where both of them use the linguistic approach to forecasting the trend-cycle. The de-trended time series are then forecasted either by Automatic Design of Neural Networks (with the use of genetic algorithms) or by Support Vector Machines with the

sensitivity analysis. So, the discussed approaches combine distinct computational intelligence subfields. The goal was to provide readers with a kind of tasting of distinct methods that may serve as an alternative to standard statistical methods. As a comparison baseline, we have chosen the standard (seasonal) ARIMA implemented in the professional software package ForecastPro®. Forecast accuracy was measured by two well-established and strongly motivated accuracy measures on two distinct horizons.

The results approved that CI subfields may compete with standard methods and in case of their combination, one may benefit from different advantages inherited from the original methods: flexibility, adaptability, generality or interpretability.

This study is supposed to be an introduction to the above given field. For example, using the linguistic approach to model the trend-cycle if a given time series have no trend is obviously rather unsupported, and information that no trend is present in the time series would be fully sufficient. Similarly using FANN or FSVM for **abraham 14** rather than ARIMA or vice-versa, using ARIMA for **mackey-glass** rather than FANN or FSVM would lead to worse results. This is a direct corollary of the well-known "no-free lunch" theorem on distinct time series methods.

This is usually solved by constructing (combining) ensembles of methods [44, 45]. This means that a set of methods is used to forecast a given time series and the particular forecasts (or some of them) are combined in a distinct way to eliminate the possibility of choosing only one – the wrong one, see [46]. The fact that the introduced approached were not generally worse than perhaps the most-widely used standard method opens the gate for their wide use in ensemble techniques. This is in our opinion the next necessarily upcoming step for the future investigation.

Acknowledgment

We gratefully acknowledge support of the project DAR 1M0572 of the MŠMT ČR.

References

[1] Ajoy K. Palit and Dobrivoje Popovic. *Computational Intelligence in Time Series Forecasting: Theory and Engineering Applications (Advances in Industrial Control)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[2] G. Zhang, B. Eddy Patuwo, et al. Forecasting with artificial neural networks::: The state of the art. *International journal of forecasting*, 14(1):35–62, 1998.

[3] P. Cortez, M. Rocha, and J. Neves. Evolving Time Series Forecasting ARMA Models. *Journal of Heuristics*, 10(4):415–429, 2004.

[4] K. Miiller, A. Smola, G. Ratsch, B. Scholkopf, J. Kohlmorgen, and V. Vapnik. Predicting time series with support vector machines. In *Proceedings of the 7th International Conference on Artificial Neural Networks*, pages 999–1004. Springer, 1997.

[5] Ian Nunn and Tony White. The application of antigenic search techniques to time series forecasting. In *GECCO*, pages 353–360, 2005.

[6] J. Aznarte, J. Benítez, and J. Castro. Smooth transition autoregressive models and fuzzy rule-based systems: Functional equivalence and consequences. *Fuzzy Sets and Systems*, 158:2734–2745, 2007.

[7] N. Kasabov and Q. Song. Denfis: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction. *IEEE Transactions on Fuzzy Systems*, 10:144–154, 2002.

[8] German Gutierrez Juan Peralta, Xiaodong Li and Araceli Sanchis. Time series forecasting by evolving artificial neural networks using genetic algorithms and differential evolution. In *IJCNN*, 2010.

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

[10] Xin Yao, S. M. Ieee, and Yong Liu. A new evolutionary system for evolving artificial neural networks. *IEEE Transactions on Neural Networks*, 8:694–713, 1996.

[11] Hiroaki Kitano. Designing neural networks using genetic algorithms with graph generation system. *Complex Systems Journal*, 4:461–476, 1990.

[12] Xin Yao. A review of evolutionary artificial neural networks. *International Journal of Intelligent Systems*, 4:539–567, 1993.

[13] C. Cortes and V. Vapnik. Support Vector Networks. *Machine Learning*, 20(3):273–297, 1995.

[14] W. He, Z. Wang, and H. Jiang. Model optimizing and feature selecting for support vector regression in time series forecasting. *Neurocomputing*, 72(1-3):600–611, 2008.

[15] P. Cortez, M. Rocha, and J. Neves. *Time Series Forecasting by Evolutionary Neural Networks*, chapter III, pages 47–70. Idea Group Publishing, USA, 2006.

[16] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, NY, USA, 2nd edition, 2008.

[17] G. Leng, T. McGinnity, and G. Prasad. An approach for on-line extraction of fuzzy rules using a self-organising fuzzy neural network. *Fuzzy Sets and Systems*, 150:211–243, 2005.

[18] V. Novák, M. Štěpnička, A. Dvořák, I. Perfilieva, V. Pavliska, and L. Vavříčková. Analysis of seasonal time series using fuzzy approach. *International Journal of General Sys-*

tems, 39:305–328, 2010.

[19] M. Štěpnička, A. Dvořák, V. Pavliska, and L. Vavříčková. Linguistic approach to time series modeling with the help of F-transform. *Fuzzy Sets and Systems*, to appear.

[20] P. Cortez, M. Rio, M. Rocha, and P. Sousa. Internet Traffic Forecasting using Neural Networks. In *Proceedings of the 2006 International Joint Conference on Neural Networks (IJCNN 2006)*, pages 4942–4949, Vancouver, Canada, July 2006. IEEE.

[21] R. J. Hyndman. Time series data library. *http://robjhyndman.com/TSDL/*.

[22] J.S. Armstrong and F. Collopy. Error measures for generalizing about forecasting methods: Empirical comparisons. *International Journal of Forecasting*, 8:69–80, 1992.

[23] R.J. Hyndman and A.B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688, 2006.

[24] D. B. Fogel. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. IEEE Press Series on Computational Intelligence. Wiley-IEEE Press, third edition, December 2005.

[25] Andreas Zell, Günter Mamier, R. Hübner, N. Schmalzl, Tilman Sommer, and Michael Vogt. Snns: An efficient simulator for neural nets. In *MASCOTS '93: Proceedings of the International Workshop on Modeling, Analysis, and Simulation On Computer and Telecommunication Systems*, pages 343–346, San Diego, CA, USA, 1993. Society for Computer Simulation International.

[26] A. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14:199–222, 2004.

[27] W. Wang, Z. Xu, W. Lu, and X. Zhang. Determination of the spread parameter in the Gaussian kernel for classification and regression. *Neurocomputing*, 55(3):643–663, 2003.

[28] V. Cherkassy and Y. Ma. Practical Selection of SVM Parameters and Noise Estimation for SVM Regression. *Neural Networks*, 17(1):113–126, 2004.

[29] R. Kewley, M. Embrechts, and C. Breneman. Data Strip Mining for the Virtual Design of Pharmaceuticals with Neural Networks. *IEEE Trans Neural Networks*, 11(3):668–679, May 2000.

[30] P. Cortez. Data Mining with Neural Networks and Support Vector Machines using the R/rminer Tool. In P. Perner, editor, *Advances in Data Mining – Applications and Theoretical Aspects, 10th Industrial Conference on Data Mining*, pages 572–583, Berlin, Germany, July 2010. LNAI 6171, Springer.

[31] I. Perfilieva. Fuzzy transforms: theory and applications. *Fuzzy Sets and Systems*, 157:993–1023, 2006.

[32] V. Novák. Linguistically oriented fuzzy logic controller and its design. *Internat. J. Approx. Reason.*, 12(3–4):263–277, 1995.

[33] I. Perfilieva and R. Valášek. Fuzzy transforms in removing noise. In B. Reusch, editor, *Computational Intelligence, Theory and Applications*, Advances in Soft Computing, pages 221–230, Berlin, 2005. Springer.

[34] M. Štěpnička and O. Polakovič. A neural network approach to the fuzzy transform. *Fuzzy sets and Systems*, 160:1037–1047, 2009.

[35] R. Bělohlávek and V. Novák. Learning rule base of the linguistic expert systems. *Soft Computing*, 7:79–88, 2002.

[36] A. Dvořák, H. Habiballa, V. Novák, and V. Pavliska. The software package LFLC 2000 - its specificity, recent and perspective applications. *Computers in Industry*, 51:269–280, 2003.

[37] V. Novák. A comprehensive theory of trichotomous evaluative linguistic expressions. *Fuzzy Sets and Systems*, 159(22):2939ï¿½–2969, 2008.

[38] V. Novák. Perception-based logical deduction. In B. Reusch, editor, *Computational Intelligence, Theory and Applications*, Advances in Soft Computing, pages 237–250, Berlin, 2005. Springer.

[39] J. Casillas, O. Cordón, F. Herrera Triguero, and L. Magdalena, editors. *Interpretability Issues in Fuzzy Modeling (Studies in Fuzziness and Soft Computing Vol. 128)*. Springer, Heidelberg, 2003.

[40] D. Dubois and H. Prade. What are fuzzy rules and how to use them. *Fuzzy Sets and Systems*, 84:169–185, 1996.

[41] V. Novák and S. Lehmke. Logical structure of fuzzy IF-THEN rules. *Fuzzy Sets and Systems*, 157(15):2003–2029, 2006.

[42] U. Bodenhofer and P. Bauer. Interpretability of linguistic variables: a formal account. *Kybernetika*, 2:227–248, 2005.

[43] F. M. Pouzols, A. Lendasse, and A. Barriga Barros. Autoregressive time series prediction by means of fuzzy inference systems using nonparametric residual variance estimation. *Fuzzy Sets and Systems*, 161:471–497, 2010.

[44] C. Lemke and B. Gabrys. Meta-learning for time series forecasting in the nn gc1 competition. In *Proc. 16th IEEE Int. Conf. on Fuzzy Systems*, page in press, Barcelona, 2010.

[45] M. Constantini and C. Pappalardo. A hierarchical procedure for the combination of forecasts. *International Journal of Forecasting*, 26:725–743, 2010.

[46] D. K. Barrow, S. Crone, and N. Kourentzes. An evaluation of neural network ensembles and model selection for time series prediction. In *Proc. 16th IEEE Int. Conf. on Neural Networks*, page in press, Barcelona, 2010.