Short-Term Load Cross-Forecasting using Pattern-Based Neural Models

Grzegorz Dudek
Department of Electrical Engineering
Czestochowa University of Technology
42-200 Czestochowa, Al. Armii Krajowej 17, Poland
dudek@el.pcz.czest.pl

***Abstract*—In this article we present the idea of short-term load cross-forecasting. This approach combines forecasts generated by two models which learn on input data defined in different ways: as daily and weekly patterns. Pattern definitions described in this work simplify the forecasting problem by filtering out the trend and seasonal variations. The nonstationarity in mean and variance is also eliminated. Simplified relationships between predictors and output variables are modeled locally using one-neuron models. A simulation study on the sample of real data showed better performance of cross-forecasting than individual neural models.**

***Keywords*—cross-forecasting; short-term load forecasting; pattern-based forecasting; neural networks**

I. Introduction

A time series of the power system load exhibits annual, weekly and daily seasonal cycles as shown in Fig. 1. The daily curves differ in shape depending on the day type (workday, Saturday, Sunday) and season. Changes in the daily load shape and load level during the year are due to changes in weather patterns (temperature, wind speed, cloud cover, humidity, precipitation) and daylight hours. The weekly cycles are determined by workdays and holidays. The multiple seasonal cycles in the load time series as well as trend and nonstationarity in mean and variance have to be captured by a forecasting model. Load time series usually cannot be modeled directly and additional treatments such as detrending, deseasonality or decomposition are needed.

A variety of methods have been proposed for short-term load forecasting (STLF). Some of them are based on classical statistical approaches [1] such as autoregressive moving average or exponential smoothing. Others employ methods of computational intelligence and machine learning [2] such as neural networks, fuzzy logic, support vector machines, and intelligent searching methods, such as evolutionary algorithms and swarm intelligence, for optimization of the forecasting models.

In this work we propose the combined univariate forecasting approach based on two neural models which learn on time series data represented in different ways. The first model uses the daily patterns as inputs while the second one uses the weekly patterns as inputs. These two types of patterns carry different information about the forecasted variable and combining outputs of both models can improve results of the individual models.

The introduction of the appropriate definitions of the daily and weekly patterns simplify the forecasting problem which is difficult due to multiple seasonal cycles, trend and nonstationarity of the load time series. The proposed functions mapping real time series elements into patterns filter out irrelevant information and unify the input data. The simplified forecasting problem can be modeled using simpler models. We use neural models which are learned locally. This local learning approach further simplifies the problem. Our goal is to construct as simple models as possible and combine their results to get more accurate forecasts.

II. Idea of the Cross-Forecasting

One of the main problem when building a forecasting

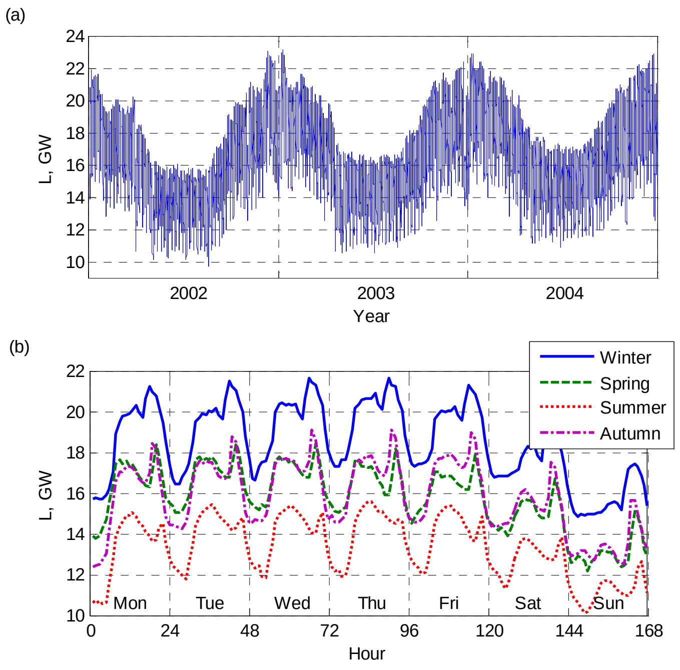

Fig. 1. The hourly load in Poland in three-year (a) and one-week (b) intervals.

model is to select predictors. In this study we construct the univariate STLF model, so we select predictors among the past power system loads. Predictors should carry as much information as possible about the forecasted variable. Usually to forecast the load for the next day the predictors are selected from the nearest past, i.e. they represent loads at successive hours of the daily period preceding the forecasted day. In this case predictors express the temporal pattern representing the shape of the daily load curve.

The predictors can be selected using autocorrelation analysis. Fig. 2 shows the autocorrelation function (ACF) for the time series of hourly loads of the Polish power system. It can be seen from this figure that the autocorrelation peeks are for lags $24n$, $n = 1, 2, ...$ This suggests using loads at hour t of the previous days as the predictors for forecasting the load at hour t of the next day. When $n = 7$ the predictors express weekly temporal pattern.

Using the daily pattern (or more precisely the contiguous sequence of time series elements) as inputs of the forecasting model is called the sequential approach (SA), whereas using the weekly pattern (time series elements of successive days of the same position in the daily cycle as the forecasted element) as inputs is called the partitioned or parallel approach (PA) [3]. In SA the load at hour t on day d+1: $L_{d+1,t}$ can be expressed as:

$$L_{d+1,t} = f(L_{d,1}, L_{d,2}, ..., L_{d,24}) + \xi\,, \qquad (1)$$

and in PA $L_{d+1,t}$ can be expressed as:

$$L_{d+1,t} = g(L_{d,t}, L_{d-1,t}, ..., L_{d-6,t}) + \xi\,, \qquad (2)$$

where f and g are some functions, and ξ is an error term.

Based on the two types of input patterns: daily and weekly ones, we build two neural models for forecasting the load at hour t for the next day. The results of both models are combined by averaging. The input data for the models are shown in Fig. 3. They form a cross, so we call this approach the cross-forecasting (C-F).

In the proposed approach we do not use raw loads as inputs and outputs of the forecasting models. Raw data exhibit multiple seasonal cycles, trend and nonstationarity. This

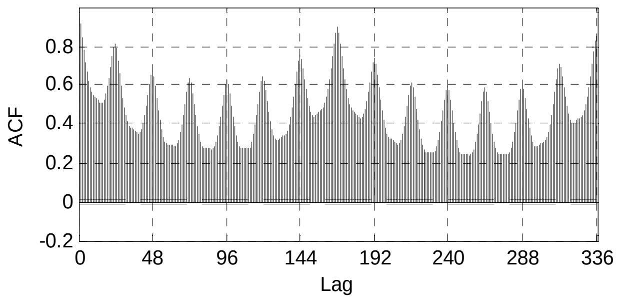

Fig. 2. The autocorrelation function of the load time series of the Polish power system.

makes it difficult to build a model. When defining daily patterns we filter out the trend, weekly and annual seasonal components. Thus the daily pattern represents the unified shape of the daily load curve. The daily input pattern representing day d is a vector $\mathbf{x}_d = [x_{d,1}\ x_{d,2}\ \dots\ x_{d,24}]^T \in X = \mathbb{R}^{24}$ with components that are functions of the actual time series elements (hourly loads). In this study we define components of $\mathbf{x}_d$ as follows:

$$x_{d,t} = \frac{L_{d,t} - \overline{L}_d}{D_d}\,, \qquad (3)$$

where $\overline{L}_d$ is the mean load in the day d and $D_d = \sqrt{\sum_{l=1}^{24} (L_{d,l} - \overline{L}_d)^2}$ is the dispersion of the hourly loads in the day d.

Equation (3) expresses normalization of the load vector $\mathbf{L}_d = [L_{d,1}\ L_{d,2}\ \dots\ L_{d,24}]^T$. After normalization the daily periods are represented by vectors $\mathbf{x}$ which have unity length, zero mean and the same variance. Note that the load time series which is nonstationary in mean and variance is represented now by x-patterns having the same mean and variance. The trend, weekly and annual cycles are filtered. X-patterns carry information only about the shapes of the daily curves.

In the similar way the output data are preprocessed. The forecasted load $L_{d+1,t}$ is encoded as:

$$y_{d+1,t} = \frac{L_{d+1,t} - \overline{L}_d}{D_d}\,. \qquad (4)$$

Note that instead using the mean load and dispersion for the day d+1 in (4), we use the mean and dispersion for the day d.

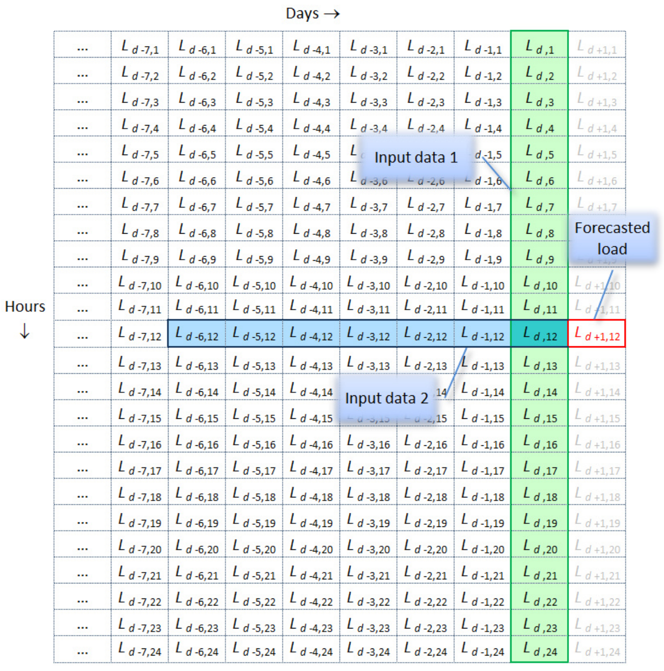

Fig. 3. Example of input data for forecasting models.

This is because $\overline{L}_{d+1}$ and D_{d+1} are not known in the moment of forecasting. Using the known values $\overline{L}_d$ and D_d enables us to calculate the forecast of $L_{d+1,t}$ when the forecast of $y_{d+1,t}$ is generated by the neural model. After rearrangement of (4) we obtain:

$$\widehat{L}_{d+1,t} = \widehat{y}_{d+1,t} D_d + \overline{L}_d \,. \quad (5)$$

In the parallel approach the weekly input patterns $\mathbf{w}_{d,t} = [w_{d,t}\ w_{d-1,t}\ \ldots\ w_{d-6,t}]^T \in W = \mathbb{R}^7$ are defined representing the weekly shapes at hour t. Their components are defined as follows:

$$w_{d,t} = \frac{L_{d,t} - \overline{L}_{d,t}}{D_{d,t}}, \quad (6)$$

where $\overline{L}_{d,t}$ is the mean load at hour t on the successive days d–6, d–5, …, d, and $D_{d,t} = \sqrt{\sum_{l=0}^{6} (L_{d-l,t} - \overline{L}_{d,t})^2}$ is the dispersion of the loads at hour t in these days.

Weekly patterns are normalized versions of the load vectors $[L_{d,t}\ L_{d-1,t}\ \ldots\ L_{d-6,t}]^T$. Similarly to x-patterns patterns $\mathbf{w}_{d,t}$ are unified: they all have zero mean, unity length and the same variance. So the trend and additional seasonal variations are filtered and there is no problem with nonstationarity. This simplify the forecasting problem and enables us using simpler neural model in PA.

In PA the forecasted load $L_{d+1,t}$ is encoded as:

$$z_{d+1,t} = \frac{L_{d+1,t} - \overline{L}_{d,t}}{D_{d,t}} \,. \quad (7)$$

Thus we use known values determined from the history, $\overline{L}_{d,t}$ and $D_{d,t}$, for encoding $L_{d+1,t}$. When the forecast of $z_{d+1,t}$ is performed by the neural model, the forecast of load is calculated from transformed (7):

$$\widehat{L}_{d+1,t} = \widehat{z}_{d+1,t} D_{d,t} + \overline{L}_{d,t} \,. \quad (8)$$

The weekly patterns can be extended or shortened: $\mathbf{w}_{d,t} = [w_{d,t}\ w_{d-1,t}\ \ldots\ w_{d-n,t}]^T$, n = 1, 2, …. In such a case in (6) the mean $\overline{L}_{d,t}$ and dispersion $D_{d,t}$ are calculated based on the load vector $[L_{d,t}\ L_{d-1,t}\ \ldots\ L_{d-n,t}]^T$. When n = 14 we get 2-week patterns, when n = 21 we get 3-week patterns. Such patterns are examined in the experimental part of this work. Examples of daily and weekly patterns are shown in Fig. 5 (Section IV).

III. Neural Models with Local Learning

To forecast the load at hour t on day d+1 we build two neural models and then combine their results. The first model learns locally on the daily patterns. The local learning [4] consists in learning the neural model on a subset of the training sample, which contains patterns from the neighborhood of the current input pattern for which we want to get the forecast (we call this pattern a query pattern **x***). In the simplest case by the neighborhood of **x*** we mean the set of its k nearest neighbors defined as the k x-patterns from the history which are closest to **x*** (in terms of Euclidean distance) and which represent the same day type $\delta \in$ {Monday, …, Sunday} as **x***. In the neighborhood of **x*** the function mapping x-patterns to y-values is less complex than in the entire range of **x** variation. So we can use simple model to approximate this function. The tests reported in [5] showed that for small k using only one sigmoid neuron brings not worse results than networks with several neurons in the hidden layer. Thus in this work we use one sigmoid neuron as an optimal neural model.

The second neural model learns locally on the weekly patterns. The training set consists of k pairs $\langle \mathbf{w}_{d,t}, z_{d+1,t} \rangle$, where $\mathbf{w}_{d,t}$ is one of the k nearest neighbors of the query pattern **w***. We can expect that the relationship between w-patterns and z-values in the neighborhood of **w*** is simple for small k and we can use the simple model for approximate it. In the experimental part of this work we use neural model composed of one sigmoid neuron.

The advantage of the one-neuron model is the limited number of parameters (weights) to learn. In our case the number of weights is 25 in SA and 8, 15 or 22 in PA when we use 1-week, 2-week or 3-week patterns, respectively. In such simple models the overfitting is limited. To further improve generalization, the neural models are learned using Levenberg-Marquardt algorithm with Bayesian regularization, which minimizes a combination of squared errors and weights. The models are learned independently for each forecasting task, i.e. the forecast of the power system load at hour t on day d+1.

The idea of short-term load cross-forecasting using pattern-based neural models in Fig. 4 is summarized.

IV. Simulation Study

In this section the proposed C-F approach is examined in the STLF problem. We use the time series of the hourly electrical load of the Polish power system from the period 2002–2004. This series is shown in Fig. 1. (This data can be downloaded from the website http://gdudek.el.pcz.pl/varia/stlf-data.) The goal is to forecast the hourly loads for the next day in July 2004. The number of the forecasting tasks (i.e. forecast the load at hour t for the next day) is 31·24 = 744. For each task we learn two neural models (sequential and parallel) using historical data as described in Section III. The number of nearest neighbors for local learning is set to k = pattern dimension + 1, i.e. 25 for daily patterns, 8 for 1-week patterns, 15 for 2-week patterns, and 22 for 3-week patterns.

In Fig. 5 x- and w-patterns are shown for one of the forecasting task. The query patterns are drawn with dashed lines and their nearest neighbors, used for learning neural models, with continuous lines. Note that patterns express unified daily, 1-week, 2-week and 3-week shapes. The minima

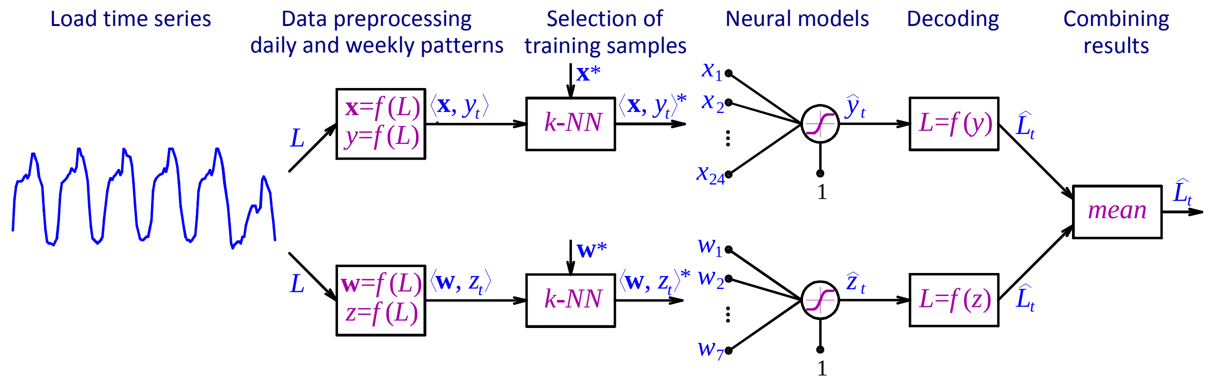

Fig. 4. The idea of the short-term load cross-forecasting using one-neuron models learned locally on the daily and weekly patterns ($\langle \mathbf{x}, y_t \rangle^*$ are k training samples nearest to $\mathbf{x}^*$).

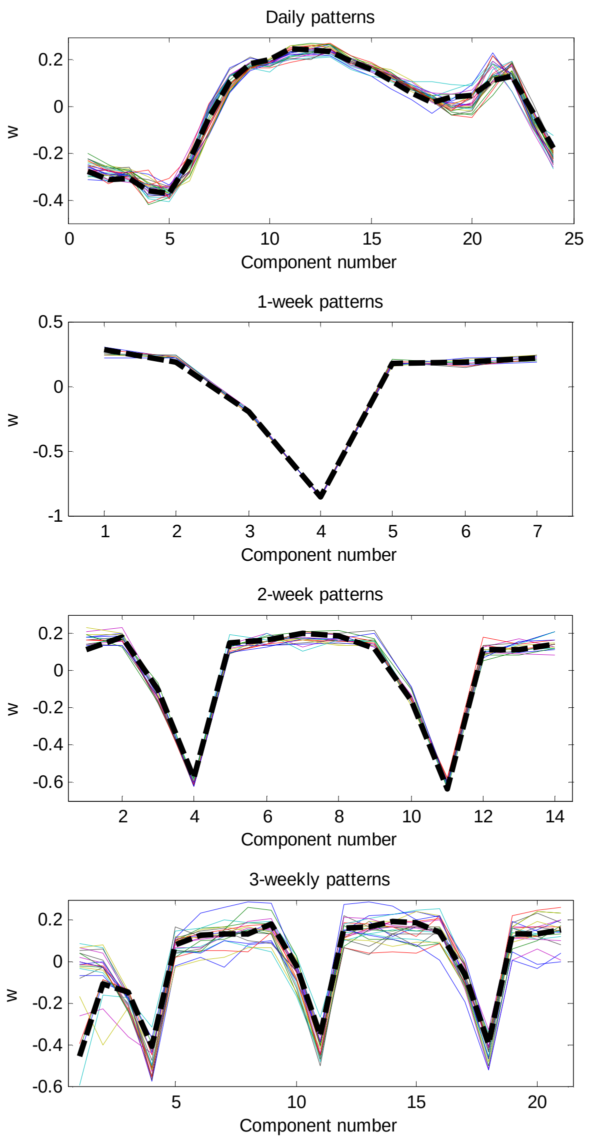

Fig. 5. Daily and weekly patterns for learning neural models which forecast the load at hour 12 on Thuesday, 1 July, 2004 (dashed lines - the query patterns).

of weekly patterns are for Sundays. The first component of the query pattern for 3-weekly representation is lower than expected. This is because it represent the public holiday.

To evaluate the forecasting methods we use MAPE as an error measure, which is traditionally used in STLF, and its interquartile range (IQR) as a measure of error dispersion. In Table 1 results of forecasting are presented, where in PA 3-week patterns were used. (When using 1-week patterns in PA MAPE was 1.21, whereas when using 2-week patterns MAPE was 1.03.) As we can see from this table the best results was achieved for C-F. The Wilcoxon rank sum test with 5% significance level indicates the statistically significant difference between errors for C-F and SA, and for C-F and PA (there is no difference between SA and PA).

Regression plot shown in Fig. 6 displays the proposed C-F model outputs (forecasted loads) with respect to target loads. It can be seen from this plot that the fit is very good with R value of above 0.99.

In Fig. 7 MAPEs for each hour of a day and for each day of the week are presented. As we can see C-F gives the lowest errors for 17 hours, while SA for 3 and PA for 4 hours. For Thursday, Friday and Saturday SA is better than PA, whereas for the remaining days of the week PA outperforms SA. C-F allows to achieve the lowest errors for all days of the week except Monday, where PA is a little better.

V. Conclusions

This paper has examined the idea of short-term load cross-forecasting as a combination of sequential and parallel approaches. The proposed methods is a type of forecasting model ensemble with only two members, which are neural models. They learn on distinct pattern representations: daily

TABLE I. FORECASTING RESULTS

	Sequential approach	**Parallel approach**	**Cross-forecasting**
MAPE	1.01	0.98	0.85
IQR	0.87	0.78	0.71

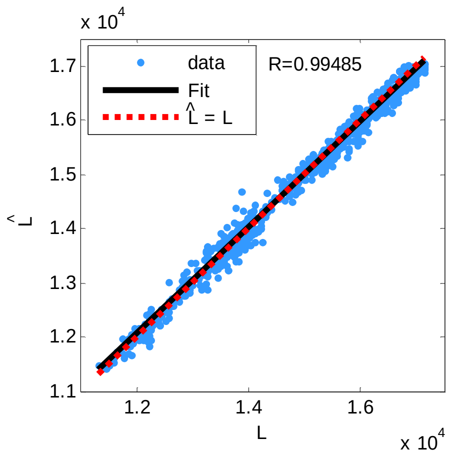

Fig. 6. Regression plot for C-F.

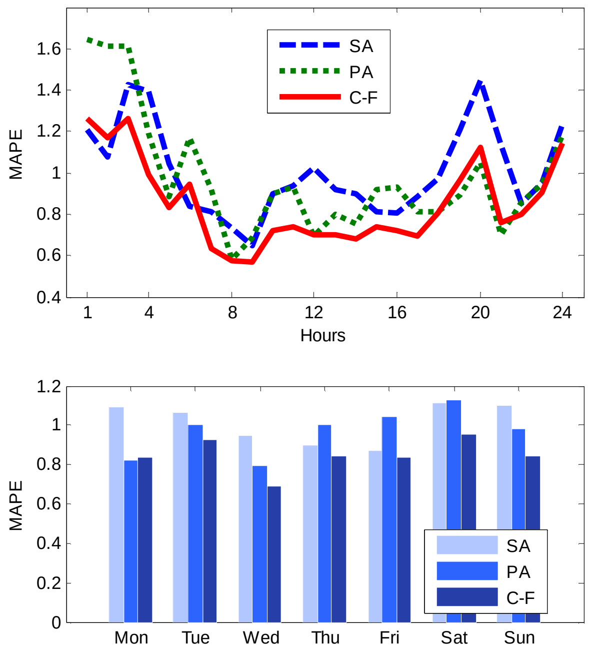

Fig. 7. Errors for each hour of a day and for each day of the week.

and weekly ones. Ensemble learning is primarily used to improve the performance of individual models. This leads to more certain, precise and accurate results.

Representation of the multiple seasonal time series of load using daily and weekly patterns allows the forecasting problem to be simplified by filtering out the trend and seasonal variations. The relationship between predictors and output variables is approximated locally in the neighborhood of the query pattern using one-neuron models. The simulation study shows that using cross-forecasting based on different representations leads to a reduction in the forecast error. However, the more general conclusions need more experimentation.

References

[1] R. Weron, Modeling and Forecasting Electricity Loads and Prices. Wiley, 2006.

[2] K. Metaxiotis, A. Kagiannas, D. Askounis, J. Psarras, "Artificial intelligence in short term electric load forecasting: A state-of-the-art survey for the researcher," Energy Conversion and Management, vol. 44, pp. 1525–1534, 2003.

[3] D. Faya, J. V. Ringwoodb, M. Condona and M. Kelly, "24-h electrical load data: a sequential or partitioned time series," Neurocomputing, vol. 55, pp. 469–498, 2003.

[4] L. Battou, V. Vapnik, "Local learning algorithms," Neural Computation, vol. 4, pp. 888-900, 1992.

[5] G. Dudek, "Forecasting time series with multiple seasonal cycles using neural networks with local learning," Proc. Artificial Intelligence and Soft Computing ICAISC 2013, LNCS 7894, Springer, pp. 52-63, 2013.