Locally-Weighted Polynomial Neural Network for Daily Short-Term Peak Load Forecasting

Jungwon Yu and Sungshin Kim
Department of Electrical and Computer Engineering, Pusan National University, Busan, Korea

Abstract

Electric load forecasting is essential for effective power system planning and operation. Complex and nonlinear relationships exist between the electric loads and their exogenous factors. In addition, time-series load data has non-stationary characteristics, such as trend, seasonality and anomalous day effects, making it difficult to predict the future loads. This paper proposes a locally-weighted polynomial neural network (LWPNN), which is a combination of a polynomial neural network (PNN) and locally-weighted regression (LWR) for daily short-term peak load forecasting. Model over-fitting problems can be prevented effectively because PNN has an automatic structure identification mechanism for nonlinear system modeling. LWR applied to optimize the regression coefficients of LWPNN only uses the locally-weighted learning data points located in the neighborhood of the current query point instead of using all data points. LWPNN is very effective and suitable for predicting an electric load series with nonlinear and non-stationary characteristics. To confirm the effectiveness, the proposed LWPNN, standard PNN, support vector regression and artificial neural network are applied to a real world daily peak load dataset in Korea. The proposed LWPNN shows significantly good prediction accuracy compared to the other methods.

Keywords: Daily peak load forecasting, Locally-weighted polynomial neural network, Polynomial neural network, Locally-weighted regression

1. Introduction

Load forecasting has always been an essential part of efficient power system planning and operation [1], and is defined as the science of predicting future load demand for a specified period of time [2]. Many operational decisions in power systems, such as unit commitment, reducing spinning reserve, economic dispatch, automatic generation control, security assessment, reliability analysis, maintenance scheduling, and energy commercialization, depend on the future behavior of loads. In particular, load forecasting has become more important than ever due to increasing deregulation and free competition of the electric power industry all around the world [3]. An overestimation or underestimation of future load gives rise to an excessive supply or a failure of a smooth power supply, resulting in an increase in operational costs [4].

Over the last few decades, a wide range of techniques including statistical methods [5-8] and artificial intelligence (AI)-based methods have been proposed for electric load forecasting. Because an electric load series have striking characteristics such as nonlinearity and non-stationarity, statistical methods which assume stationary time-series have difficulty in predict-
ing future electricity demand satisfactorily. Recently, AI-based methods including artificial neural networks (ANNs) [9-11], and support vector regression (SVR) [2-4] have been applied successfully for load forecasting.

To make an accurate forecast, the following features of an electric load must be considered.

- Complex and nonlinear relationships exist between electric load of real world and its exogenous factors, e.g. historical load data, temperature, etc.
- An electric load series contains several non-stationary features such as trends, changes in level and slope, seasonality, etc. [11].
- Although electric load series are periodic time-series for a week or year, they show different demand patterns in anomalous days such as national holidays and election days from the regular days’ patterns.

This paper proposes a locally-weighted polynomial neural network (LWPNN) which is the combination of a polynomial neural network (PNN) and locally-weighted regression (LWR) for daily short-term peak load forecasting. LWPNN is a hybrid modeling approach for constructing accurate prediction models. PNN used for nonlinear system modeling has several advantages such as automatic structure identification mechanism and noise immunity [12, 13]. In LWPNN, LWR is used to optimize the regression coefficients of each neuron. In LWR, the learning data located in the vicinity of the current query point is selected using a distance function and employed to optimize the parameters of model. Each selected learning data point is weighted locally by a weighting function depending on its similarity with a current query point. By combining PNN and LWR, real-time robust parameter identification is possible and electric load series whose statistical characteristics are changed with time can be predicted effectively.

The remainder of this paper is organized as follows. Section 2 describes the structure and parameter identification of PNN. Section 3 explains LWR and optimization of bandwidth parameter and scaling factors. Section 4 summarizes the LWPNN. Section 5 shows the experimental results for real world load data, and Section 6 presents the main contributions and concluding remarks.

2. Polynomial Neural Network

PNN proposed by Ivakhnenko [14] in the 1960s is a self-organizing modeling approach. PNN is an effective modeling method if there is many observation data and system input variables. PNN has been widely applied to dynamic system modeling, time-series predictions and intelligent control, etc. [14-16]. Figure 1 shows a multilayered architecture of PNN, where each neuron is connected hierarchically to build a nonlinear and complex model. Each layer except for the input layer is composed of quadratic polynomial neurons and the input layer consists of \( m \) initial input candidates. With the exception of layer 1, the output of \( m \) neurons, which are satisfied with the selection criterion among \( q (= mC_2) \) neurons of the previous layer, is used for the input for the next layer. The parameters of each neuron can be identified using least square method and the same learning data is used repeatedly at each layer and neuron.

2.1 Least Square Method

The output of an arbitrary neuron in Figure 1 is calculated as

\[
y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2,
\]

where \( \theta_0, ..., \theta_5 \) are regression coefficients optimized using least square method [17, 18]. To identify the unknown parameters, a learning dataset, \( \{(x_{1,i}, x_{2,i}; y_i), i = 1, ..., n\} \), needs to be collected; they represent the desired input-output learning data pairs of the target system. A set of \( n \) linear equations can be obtained by substituting each learning data pairs into (1) and the linear equations can be represented using concise matrix notation as follows:

\[
X \theta = y,
\]

where \( X \) is an \( n \) by 6 design matrix, \( \theta \) is an unknown parameter vector and \( y \) is an output vector. Least square estimator (LSE) is defined as

\[
\hat{\theta} = (X^T X)^{-1} X^T y.
\]
When \((3)\) is used to identify the parameter vector, there are several problems: 1) If \(X\) has dependent columns, the LSE is not unique; 2) The determinant of \(X^T X\) is often close to zero in many cases (in this case, the performance of LSE will deteriorate); and 3) \((3)\) is rather susceptible to rounding off errors [19, 20]. Singular value decomposition (SVD) can be employed to solve the above problems. Using the SVD, the optimal parameter vector can be found when \(X^T X\) is singular. Furthermore, a more robust LSE can be obtained even though the determinant of \(X^T X\) is close to zero or rounding errors occur.

Any \(m\) by \(n\) matrix \(X\) can be factored into

\[
X = USV^T = \text{(orthogonal)}(\text{diagonal})(\text{orthogonal}),
\]

where the columns of matrix \(U\) (\(m\) by \(m\)) are composed of eigenvectors of \(XX^T\) and the columns of matrix \(V\) (\(n\) by \(n\)) consist of eigenvectors of \(X^T X\). When \(X\) has rank \(r\), the \(r\) singular values on the diagonal of \(S\) (\(m\) by \(n\)) are the square roots of the non-zero eigenvalues of both \(XX^T\) and \(X^T X\) [21]. Using SVD, the minimum length least squares solution can be calculated as follows [20, 21]:

\[
\hat{\theta}^+ = X^+ y = V S^+ U^T y, \tag{4}
\]

where \(X^+ = VS^+ U^T\), which is the pseudo inverse of \(X\). The diagonal of \(S^+\) (\(n\) by \(m\)) contains the reciprocal of the \(r\) singular values in \(S\).

### 2.2 Singular Value Decomposition

When \((3)\) is used to identify the parameter vector, there are several problems: 1) If \(X\) has dependent columns, the LSE is not unique; 2) The determinant of \(X^T X\) is often close to zero in many cases (in this case, the performance of LSE will deteriorate); and 3) \((3)\) is rather susceptible to rounding off errors [19, 20]. Singular value decomposition (SVD) can be employed to solve the above problems. Using the SVD, the optimal parameter vector can be found when \(X^T X\) is singular. Furthermore, a more robust LSE can be obtained even though the determinant of \(X^T X\) is close to zero or rounding errors occur.

Any \(m\) by \(n\) matrix \(X\) can be factored into

\[
X = USV^T = \text{(orthogonal)}(\text{diagonal})(\text{orthogonal}),
\]

where the columns of matrix \(U\) (\(m\) by \(m\)) are composed of eigenvectors of \(XX^T\) and the columns of matrix \(V\) (\(n\) by \(n\)) consist of eigenvectors of \(X^T X\). When \(X\) has rank \(r\), the \(r\) singular values on the diagonal of \(S\) (\(m\) by \(n\)) are the square roots of the non-zero eigenvalues of both \(XX^T\) and \(X^T X\) [21]. Using SVD, the minimum length least squares solution can be calculated as follows [20, 21]:

\[
\hat{\theta}^+ = X^+ y = V S^+ U^T y, \tag{4}
\]

where \(X^+ = VS^+ U^T\), which is the pseudo inverse of \(X\). The diagonal of \(S^+\) (\(n\) by \(m\)) contains the reciprocal of the \(r\) singular values in \(S\).

### 2.3 Performance Criterion for PNN Structure Identification

When using PNN, it is important to consider the following two important questions:

1. How can the best neurons be selected at each layer?
2. How can the maximum number of layer be decided?

This study uses performance criterion (PC) for structure identification of PNN. First of all, prepared learning data is divided into two groups: dataset A, \(\{(x_i^A, y_i^A)\}_{i=1}^{n_A}\), and dataset B, \(\{(x_i^B, y_i^B)\}_{i=1}^{n_B}\), where \(x = [x_1, x_2, \ldots, x_m]^T\) is the \(m\) dimensional input vector, \(y\) is the desired output, and \(n_A\) and \(n_B\) are the number of data pairs in dataset A and B, respectively.

Next, model \(f_A\) is constructed by dataset A. The root mean square error \(e_A\) and \(e_B\) for dataset A and B are calculated using model \(f_A\). Finally, the PC is calculated by

\[
e_A^2 = \frac{1}{n_A} \sum_{i=1}^{n_A} (y_i^A - f_A(x_i^A))^2, \quad e_B^2 = \frac{1}{n_B} \sum_{i=1}^{n_B} (y_i^B - f_A(x_i^B))^2,
\]

\[
PC = e_A^2 + e_B^2 + \eta |e_A^2 - e_B^2|, \tag{5}
\]

where \(y_i^A = f_A(x_i^A)\) and \(y_i^B = f_A(x_i^B)\) are the model output of \(i\)-th data pair in dataset A and B, respectively, and \(\eta\) is a weighting factor on the term, \(|e_A^2 - e_B^2|\). The use of the PC aims to minimize the errors and at the same time, prevent over-fitting problems of PNN.

### 2.4 Example of PNN Structure Identification Procedure

This subsection gives an example of PNN structure identification procedure using the PC. Figure 2 shows an example of the PNN structure identification procedures when the initial input vector is \(x = [x_1, x_2, x_3, x_4]^T\). First, \(6 (= 4C_2)\) neurons are generated from 4 initial input candidates and \(PC(i) = \min_j\{PC(i,j)\} (i = 1; j = 1, \ldots, 6)\) is calculated, where \(PC(i,j)\) is the PC value of \(j\)-th neuron of \(i\)-th layer. Next, 4 neurons of layer 1 are selected for the next layer’s inputs. In common with layer 1, at layer 2, \(PC(i)\) is calculated and \(PC(1)\) and \(PC(2)\) are compared. If \(PC(1) > PC(2)\), the extension to layer 3 is conducted. Until \(PC(i)\) increases, an extension of the layer is performed continuously. At layer 4, because \(PC(3) < PC(4)\), an extension of the layer is stopped and the neuron with a minimum PC at layer 3 is selected as the output neuron. Finally, from layer 2 to the input layer, the elements that are connected to the output neuron are selected.
successively. In this example, the stop layer is layer 3 and the relevant inputs selected among the initial input candidates are \( x_1, x_3 \) and \( x_4 \). In other words, the stop layer, best neurons and relevant inputs can be automatically selected using the PC.

3. Locally Weighted Regression

LWR is a memory-based learning method that involves storing the learning data in memory and finding the relevant data in the database to answer a current query [22]. All learning data points must be conserved in memory until the query points are processed entirely.

In LWR, the validation data points waiting in a queue are treated sequentially. As a first step, a similarity measure is performed between the current query point and learning data stored in memory using distance function. The learning data points with high similarity are selected to answer a current query. Selected learning data points are given different weights depending on their similarities to the current query point.

3.1 Distance Function

In this research, we use diagonally scaled Euclidean distance function defined as [22]

\[
d(x, q) = \sqrt{\sum_{i=1}^{n} (m_i(x_i - q_i))^2}
\]

\[= \sqrt{(x - q)^T M^T M(x - q)}, \quad (6)
\]

where \( x \) and \( q \) are the \( n \)-dimensional vectors, \( x_i \) and \( q_i \) are the \( i \)-th dimensional component of \( x \) and \( q \), respectively, \( m_i \) is a scaling factor of \( i \)-th dimension and \( M = \text{diag}[m_1, m_2, ..., m_n] \) is diagonal distance function matrix whose diagonal components are composed of the scaling factors of each dimension. The input space can be scaled into a radially symmetric axis parallel ellipse using the distance function matrix. More accurate regression performance can be achieved if the input space is properly distorted or warped by \( M \). If certain scaling factors are set to zero, those dimensions are ignored in similarity measure. But a zero scaling factor does not mean that the regression model disregards the zero scaled input.

3.2 Weighting Function

In this study, commonly used Gaussian weighting function is employed, which is defined as

\[
K(d) = \begin{cases} 
\exp(-d^2) & \text{if } |d| < c \\
0 & \text{otherwise.}
\end{cases} \quad (7)
\]

The Gaussian weighting function has the maximum value at \( d = 0 \) and decreases smoothly as \( d \) increases. Because \( K(d) \) is defined at \(-\infty < d < \infty\), the condition that \( K(d) \) is non-zero at \( |d| < c \), where \( c \) is the threshold value, can reduce the algorithm learning time.

3.3 Bandwidth Parameter

In LWR, the weight of the \( j \)-th selected learning data point by the weighting function is calculated as

\[
K \left( \frac{d(x_k, q)}{h} \right) = \exp \left[ - \left( \frac{d(x_k, q)}{h} \right)^2 \right], \quad (8)
\]

where \( \{x_k\}, k = 1, ..., l \) is the selected dataset by the similarity measure between the current query point \( q \) and learning data points stored in memory, and \( h \) is a bandwidth parameter. In [8], the parameter, \( h \), related with the range of the weighting function plays an important role in the regression performance [23]. Local modeling gradually becomes global as \( h \) approaches infinity. If \( h \) is too wide, the regression performance may deteriorate. On the other hand, the risk of over-fitting increases if \( h \) is too narrow [2]. Therefore, it is essential to find the optimal bandwidth for accurate LWR.
In this study, particle swarm optimization (PSO) [24], a global optimization method, is used to optimize the bandwidth parameter, \( h \), and scaling factors \( m_i \). In the optimization of bandwidth parameter \( h \) and scaling factors \( m_i, i = 1, \ldots, n \), the position vector of a particle \( x_i \) is encoded as

\[
x_i = [h_i, m_{1,i}, m_{2,i}, \ldots, m_{n,i}]^T \tag{9}
\]

The same learning data used in structure identification of PNN are employed for the optimization. The dataset A is stored in memory and the dataset B is waiting in the queue. The objective is to find the optimal \( h \) and \( m_i \) that minimize the model error for the dataset B.

4. Locally-Weighted Polynomial Neural Network

In LWPNN, the parameter vector \( \theta \) of each neuron is optimized using the weighted least square method and its objective function is defined as

\[
E_W(\theta) = (y - X\theta)^T W (y - X\theta), \tag{10}
\]

where \( W \) is a weighting matrix whose diagonal components are composed of weights for selected learning data points and defined as

\[
W = \text{diag}\left( K \left( \frac{d(x_1, q_i)}{h} \right), \ldots, K \left( \frac{d(x_m, q_i)}{h} \right) \right) \tag{11}
\]

The weighted LSE \( \hat{\theta}_W \) which minimizes \( (10) \) can be obtained as

\[
\hat{\theta}_W = (X^TWX)^{-1} X^T Wy \tag{12}
\]

As mentioned in Subsection 2.2, there are several difficulties in \( (12) \). Therefore, SVD is employed to find the optimal LSE \( \hat{\theta}_W \) defined as

\[
\hat{\theta}_W^+ = (\sqrt{W}X)^+ \sqrt{W}y \tag{13}
\]

Figure 3 presents a flowchart of the LWPNN algorithm. Firstly, as mentioned in Subsection 2.4, structure identification of PNN for finding the stop layer, the best neurons at each layer and the relevant inputs is conducted. And then, optimization of the bandwidth parameter \( h \) and scaling factors \( m_i \) is carried out. Finally, prediction for validation data is sequentially performed as follows:

Step 1: Among learning data \( x_j (j = 1, \ldots, M) \), select the data points satisfied with the condition \( d(x_j, q_i) \leq c \) for a current query point \( q_i \), where \( d(\cdot) \) is the distance function and \( c \) is the threshold value.

Step 2: Find the weights for the selected data points using \( (11) \).

Step 3: Train the PNN using \( (13) \).

Step 4: Calculate the prediction value \( \hat{y}_i \) of a current query point \( q_i \).

Step 5: Repeat the procedures from Steps 1 to 4 until all validation data is processed completely.

Figure 4 shows the block diagram of LWPNN.

5. Experimental Results

In this section, real world peak load data, collected from the Electric Power Statistics Information System in Korea, from January 1, 2010 to December 31, 2012 is employed to verify the performance of the proposed LWPNN. The objective of the proposed model is to predict the daily peak load for the 366 days in 2012 using (i) the given historical peak load data from 2010 to 2011, (ii) meteorological elements, such as temperature, amount of clouds, rainfall, relative humidity, and wind speed offered from the Korea Meteorological Administration, and...
Figure 6. Correlation between the daily peak load and daily average temperature.

(iii) annual holiday information to consider the anomalous day effects.

5.1 Data Analysis

To construct accurate prediction models, the main characteristics of the daily peak load are analyzed in this Subsection. Figure 5 shows the daily peak load and daily average temperature in Korea from January 1, 2010 to December 31, 2011. As indicated in Figure 5, the daily peak load data is affected greatly by seasonal variations and the periodic time-series for a year: high demand in the summer (from Jun. to Aug.) and winter (from Dec. to Feb.), whereas low demand in spring (from Mar. to May) and autumn (from Sept. to Nov.). Figure 6 shows the correlation between the daily peak load and the daily average temperature from 2010 to 2011. As shown in Figure 6, there is a nonlinear relationship between the daily peak load and daily average temperature. A positive correlation (correlation coefficient is about 0.4143) is confirmed when the average temperature is higher than approximately 18°C (see the dotted line in Figure 6). On the other hand, a negative correlation (correlation coefficient is about -0.6621) is observed when the mean temperature is lower than approximately 18°C. The correlation between the daily peak load and daily maximum or minimum temperature is analogous to the case of the average temperature. There are no correlations between the daily peak load and other meteorological factors except for temperature.

In this paper, historical load values and average temperature are only considered as initial input candidates.

Another characteristic of the daily peak load is that there is periodicity every week. Figure 7 shows the daily peak load for February 2010 and February 2011.

February 2010 and February 2011. The daily peak load is a periodic time-series in the interval except for the anomalous days, such as national holidays and election days. The load demand on weekdays (Monday to Friday) is higher than that on weekends. In addition, the load demand on Sunday is slightly lower than that on Saturday. From the two year historical peak load data, the peak load demand on anomalous days is similar to or lower than that on weekends. The type of day indicator which will be explained in Subsection 5.2 is employed to consider the effects of anomalous days on the daily peak load.

5.2 Data Preparation

Daily peak load data in Korea from 2010 to 2012 are reconstituted as follows:

\[ \{C(t), T(t), L(t), t = 1, ..., 1096\}, \]

where \(T(t)\) is the daily average temperature, \(L(t)\) is the daily peak load and \(C(t)\) is the decimal-valued type of day indicator to keep track of the periodicity in every week and anomalous day effects. For example, Monday corresponds to \(C(t) = 1\), Tuesday corresponds to \(C(t) = 2\), ..., Sunday corresponds to \(C(t) = 7\) and anomalous days correspond to \(C(t) = 8\).

The initial inputs and output of the learning and validation data are formed as \([C(t+1), T(t+1), L(t-6), L(t-5), L(t-4), L(t-3), L(t-2), L(t-1), L(t); L(t+1)]\). Figure 8 shows the learning period for identifying the model and validation period for checking the identified model to verify the proposed LWPNN. On the first day of each month, the structure of PNN, bandwidth parameter and scaling factors are newly updated. All input and output variables are normalized at the range of \([0, 1]\).
as
\[ x_{\text{norm}} = \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}}, \]
(14)
where \(x\) refers to the original value of input or output variable, \(x_{\text{min}}\) and \(x_{\text{max}}\) are the minimum and maximum values of the original value, respectively and \(x_{\text{norm}}\) corresponds to the normalized value.

5.3 Model Construction

As explained in the Subsection 2.4, the structure identification method based on the PC is employed to select the relevant input variables, stop layer and the best neurons at each layer. Figure 9 shows the MSE and PC of the neuron, which has the lowest value of the PC at each layer while constructing the model for January 2012. The PC is calculated using (5) and \(\eta\) is set as 2. As illustrated in Figure 9(b), the stop layer is determined to be 5 because the PC increases when the number of layers is expanded from 5 to 6. After determining the stop layer, the best neurons at each layer and relevant inputs can be selected, as described in Subsection 2.4. For example, to predict the daily peak load of Jan. 2012, stop layer is decided as 6 and the selected inputs are as follows: \(\{C(t), T(t), L(t-6), L(t-2), L(t)\}\).

In the case of Jan. 2012, because the number of selected inputs is 5, the diagonal distance function matrix \(M\) is 5 by 5. Therefore, the total number of parameters that should be optimized by PSO is 6. In PSO, after a few trials, \(c_1\) and \(c_2\) are set to 0.3 and 0.1, respectively, the population size and maximum iteration number are set to 30 and 5000, respectively, and \(w(0)\) and \(w(t_{\text{max}})\) are set to 0.9 and 0.45, respectively. Using PSO, the optimized bandwidth parameter and diagonal distance function matrix are \(h = 0.2912\) and \(M = \text{diag}[2.1361, 0.5629, 0.4543, 0.6831, 0.1841]\).

5.4 Forecasting Results

The objective of the proposed LWPNN is to predict the daily peak load in Korea for the 366 days of 2012. As mentioned above, the structure of PNN, bandwidth parameter and diagonal distance function matrix are newly updated on the first day of each month. The mean absolute percentage error (MAPE) is used to compare the proposed method with other methods. The MAPE is defined as

\[ \text{MAPE} = \frac{1}{N} \sum_{i=1}^{N} \left| \frac{L_i - \hat{L}_i}{L_i} \right| \times 100, \]
(15)
where \(L_i\) and \(\hat{L}_i\) are the actual and predicted daily peak load on \(i\)-th day of each month and \(N\) is the number of validation data at each month. Table 1 lists the MAPE, from January 2012 to December 2012, of the proposed LWPNN and other comparison methods, such as ANN and SVR. SVR and ANN are implemented using the LIBSVM [25] and the MATLAB Neural Network Toolbox, respectively. In SVR, the radial basis function (RBF) kernel is used and design parameters, such as \(\varepsilon, C\) and \(\gamma\), are selected by trial and error. In ANN, the 11–6–1 and 11–9–1 multilayer network architectures are used and trained using Levenberg-Marquardt algorithm. All experiments in ANN are run 10 times and the MAPE value is the mean of 10 runs. Table 2 lists the
Table 1. Monthly forecasts from January 2012 to December 2012 (MAPE)

| Method      | MAPE   | Type          |
|-------------|--------|---------------|
| Standard PNN | 2.3605%| Comparison method |
| SVR         | 1.7560%| Comparison method |
| ANN(11-6-1) | 1.7961%| Comparison method |
| ANN(11-9-1) | 2.0316%| Comparison method |
| LWPNN       | 1.6389%| Proposed method |

Table 2. Comparison of the proposed LWPNN and other methods for the 366 days of 2012

MAPE for the 366 days of 2012 of proposed LWPNN and other comparison methods. As listed in Table 2, proposed LWPNN outperforms the comparison methods, SVR and ANN. LWPNN improves the MAPE over SVR and ANN (11-6-1) by 0.12% and 0.16%, respectively. These slight improvements have a highly positive impact on power system operations. As a result, the superiority of the proposed method over other methods is demonstrated. Figure 10 shows the prediction results of the daily peak load for the 366 days of 2012 using the proposed LWPNN. As shown in Figure 10, the predicted peak loads using LWPNN are very close to the actual load.

6. Conclusion

This paper proposed the LWPNN that combines PNN and LWR for daily short-term peak load forecasting. The main contributions of this paper can be summarized as follows:

- PC-based structure identification for PNN was introduced to overcome the over-fitting problems and minimize the PNN’s error.

- A robust LSE using SVD was developed to optimize the regression coefficients.

- Using LWR, real-time robust parameter identification is possible.

- By combining PNN and LWR, the load series, which has complex, nonlinear and non-stationary characteristics, can be predicted effectively.

Real world peak load data in Korea from January 1, 2010 to December 31, 2012 was used to verify the proposed LWPNN. As described in Subsection 5.4, LWPNN showed the best load forecasting performance, and the predicted load values were very close to the actual values.
Figure 10. Actual and predicted daily peak load in 2012: (a) From January to April. (b) From May to August. (c) From September to December.

Conflict of Interest

No potential conflict of interest relevant to this article was reported.

Acknowledgements

This work was supported by the Energy Efficiency & Resources Core Technology Program of the Korea Institute of Energy Technology Evaluation and Planning (KETEP) granted financial resource from the Ministry of Trade, Industry & Energy, Republic of Korea (No. 20151110200040).

References

[1] T. Senjyu, H. Takara, K. Uezato, and T. Funabashi, “One-hour-ahead load forecasting using neural network,” *IEEE Transactions on Power Systems*, vol. 17, no. 1, pp. 113-118, 2002. [http://dx.doi.org/10.1109/59.982201]

[2] E. E. Elattar, J. Goulermas, and Q. H. Wu, “Electric load forecasting based on locally weighted support vector regression,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 4, pp. 438-447, 2010. [http://dx.doi.org/10.1109/TSMCC.2010.2040176]

[3] S. Fan and L. Chen, “Short-term load forecasting based on an adaptive hybrid method,” *IEEE Transactions on Power Systems*, vol. 21, no. 1, pp. 392-401, 2006. [http://dx.doi.org/10.1109/TPWRS.2005.860944]

[4] J. Nagi, K. S. Yap, F. Nagi, S. K. Tiong, and S. K. Ahmed, “A computational intelligence scheme for the prediction of the daily peak load,” *Applied Soft Computing*, vol. 11, no. 8, pp. 4773-4788, 2011. [http://dx.doi.org/10.1016/j.asoc.2011.07.005]

[5] B. Wang, N. L. Tai, H. Q. Zhai, J. Ye, J. D. Zhu, and L. B. Qi, “A new ARMAX model based on evolutionary algorithm and particle swarm optimization for short-term load forecasting,” *Electric Power Systems Research*, vol. 78, no. 10, pp. 1679-1685, 2008. [http://dx.doi.org/10.1016/j.epsr.2008.02.009]

[6] D. J. Trudnowski, W. L. McReynolds, and J. M. Johnson, “Real-time very short-term load prediction for power-system automatic generation control,” *IEEE Transactions on Control Systems Technology*, vol. 9, no. 2, pp. 254-260, 2001. [http://dx.doi.org/10.1109/87.911377]

[7] S. J. Huang and K. R. Shih, “Short-term load forecasting via ARMA model identification including non-Gaussian process considerations,” *IEEE Transactions on Power Systems*, vol. 18, no. 2, pp. 673-679, 2003. [http://dx.doi.org/10.1109/TPWRS.2003.811010]

[8] J. F. Chen, W. M. Wang, and C. M. Huang, “Analysis of an adaptive time-series autoregressive moving-average (ARMA) model for short-term load forecasting,” *Electric Power Systems Research*, vol. 34, no. 3, pp. 187-196, 1995. [http://dx.doi.org/10.1016/0378-7796(95)00977-1]

[9] N. Kandil, R. Wamkeue, M. Saad, and S. Georges, “An efficient approach for short term load forecasting using artificial neural networks,” *International Journal of Electrical Power & Energy Systems*, vol. 28, no. 8, pp. 525-530, 2006. [http://dx.doi.org/10.1016/j.ijepes.2006.02.014]
[10] T. S. Dillon, S. Sestito, and S. Leung, “Short term load forecasting using an adaptive neural network,” *International Journal of Electrical Power & Energy Systems*, vol. 13, no. 4, pp. 186-192, 1991. [http://dx.doi.org/10.1016/0142-0615(91)90021-M](http://dx.doi.org/10.1016/0142-0615(91)90021-M)

[11] A. J. R. Reis and A. P. A. da Silva, “Feature extraction via multiresolution analysis for short-term load forecasting,” *IEEE Transactions on Power Systems*, vol. 20, no. 1, pp. 189-198, 2005. [http://dx.doi.org/10.1109/TPWRS.2004.840380](http://dx.doi.org/10.1109/TPWRS.2004.840380)

[12] R. K. Mehra, “Group method of data handling (GMDH): review and experience,” in *Proceedings of 1977 IEEE Conference on Decision and Control including the 16th Symposium on Adaptive Processes and A Special Symposium on Fuzzy Set Theory and Applications*, New Orleans, LA, 1977, pp. 29-34. [http://dx.doi.org/10.1109/CDC.1977.271540](http://dx.doi.org/10.1109/CDC.1977.271540)

[13] B. Zhu, C. Z. He, P. Liatisis, and X. Y. Li, “A GMDH-based fuzzy modeling approach for constructing TS model,” *Fuzzy Sets and Systems*, vol. 189, no. 1, pp. 19-29, 2012. [http://dx.doi.org/10.1016/j.fss.2011.08.004](http://dx.doi.org/10.1016/j.fss.2011.08.004)

[14] A. G. Ivakhnenko, “Polynomial theory of complex systems,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 1, no. 4, pp. 364-378, 1971. [http://dx.doi.org/10.1109/TSMC.1971.4308320](http://dx.doi.org/10.1109/TSMC.1971.4308320)

[15] J. A. Muller and F. Lemke, *Self-organising Data Mining: Extracting Knowledge from Data*. Hamburg: Books on Demand, 2000.

[16] H. R. Madala and A. G. Ivakhnenko, *Inductive Learning Algorithms for Complex Systems Modeling*. Boca Raton, FL: CRC Press, 1994.

[17] T. C. Hsia, *System Identification: Least-Squares Methods*. Lexington, MA: Lexington Books, 1977.

[18] J. S. R. Jang, C. T. Sun, and E. Mizutani, *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Upper Saddle River, NJ: Prentice Hall, 1997.

[19] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed. New York, NY: Press Syndicate of the University of Cambridge, 1992.

[20] R. L. Burden and J. D. Faires, *Numerical Analysis*, 5th ed. Boston: Cengage Learning, 1993.

[21] G. Strang, *Linear Algebra and Its Applications*. Belmont, CA: Thomson-Brooks/Cole, 2005.

[22] C. G. Atkeson, A. W. Moore, and S. Schaal, “Locally weighted learning,” *Artificial Intelligence Review*, vol. 11, no. 1, pp. 11-73, 1997. [http://dx.doi.org/10.1023/A:1006559212014](http://dx.doi.org/10.1023/A:1006559212014)

[23] H. Leung, Y. Huang, and C. Cao, “Locally weighted regression for desulphurisation intelligent decision system modeling,” *Simulation Modelling Practice and Theory*, vol. 12, no. 6, pp. 413-423, 2004. [http://dx.doi.org/10.1016/j.simpat.2004.06.002](http://dx.doi.org/10.1016/j.simpat.2004.06.002)

[24] A. P. Engelbrecht, *Computational Intelligence: An Introduction*, 2nd ed. Chichester: John Wiley & Sons, 2007.

[25] C. C. Chang and C. J. Lin, “LIBSVM: a library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, article no. 27, 2011. [http://dx.doi.org/10.1145/1961189.1961199](http://dx.doi.org/10.1145/1961189.1961199)

Jungwon Yu was born in Busan, Korea in 1986. He received the B.S. and M.S. degrees from the Department of Electrical and Computer Engineering from Pusan National University (PNU), Busan, Korea, in 2012 and 2014, respectively, and is currently pursuing the Ph.D. degree in the Department of Electrical and Computer engineering at PNU. His research interests include time-series analysis, data mining and fault detection and diagnosis, etc.

E-mail: garden0312@pusan.ac.kr

Sungshin Kim received his B.S. and M.S. degrees in Electrical Engineering from Yonsei University, Korea, in 1984 and 1986, respectively, and his Ph.D. degree in Electrical Engineering from the Georgia Institute of Technology, USA, in 1996. He is currently a professor at the Electrical Engineering Department, Pusan National University. His research interests include fuzzy logic controls, neuro fuzzy systems, neural networks, robotics, signal analysis, and intelligent systems.

E-mail: sskim@pusan.ac.kr