energies

Article

A New Input Selection Algorithm Using the Group Method of Data Handling and Bootstrap Method for Support Vector Regression Based Hourly Load Forecasting

Jungwon Yu, June Ho Park and Sungshin Kim *

Department of Electrical and Computer Engineering, Pusan National University, Busan 46241, Korea;
garden0312@pusan.ac.kr (J.Y.); parkjh@pusan.ac.kr (J.H.P.)
* Correspondence: sskim@pusan.ac.kr; Tel.: +82-51-510-2374

Received: 10 September 2018; Accepted: 21 October 2018; Published: 23 October 2018

Abstract: Electric load forecasting is indispensable for the effective planning and operation of power systems. Various decisions related to power systems depend on the future behavior of loads. In this paper, we propose a new input selection procedure, which combines the group method of data handling (GMDH) and bootstrap method for support vector regression based hourly load forecasting. To construct the GMDH network, a learning dataset is divided into training and test datasets by bootstrapping. After constructing GMDH networks several times, the inputs that appeared frequently in the input layers of the completed networks were selected as the significant inputs. Filter methods based on linear correlation and mutual information (MI) were employed as comparison methods, and the performance of hybrids of the filter methods and the proposed method were also confirmed. In total, five input selection methods were compared. To verify the performance of the proposed method, hourly load data from South Korea was used and the results of one-hour, one-day and one-week-ahead forecasts were investigated. The experimental results demonstrated that the proposed method has higher prediction accuracy compared with the filter methods. Among the five methods, a hybrid of an MI-based filter with the proposed method shows best prediction performance.

Keywords: hourly load forecasting; input selection; group method of data handling; bootstrap method; support vector regression

1. Introduction

Electric load forecasting (ELF) is essential for the effective and stable planning and operation of power systems [1]. Forecasting models are constructed based on historical load series and exogenous variables (e.g., weather, economic, and social factors), and the models are used to predict future loads for a specified period of time ahead. Various decisions related to power systems depend on the future behavior of loads, such as unit commitment, spinning reserve reduction, economic dispatch, automatic generation control, reliability analysis, maintenance scheduling, and energy commercialization [2,3]. ELF, especially, has major effects on deregulated electricity markets (e.g., demand response management [4–6]) and their participants because the prices on the markets are determined by the predicted future loads. Therefore, accurate and robust ELF can contribute greatly to decision making related to power systems. However, it is difficult to precisely forecast time-series loads because they exhibit a high degree of seasonality and nonlinear characteristics.

In recent decades, a wide range of methods has been proposed for ELF. In particular, artificial intelligence-based methods such as artificial neural networks (ANNs) [7–10] and support vector regression (SVR) [11–15] have been applied successfully to ELF due to their excellent learning capacity and nonlinear mapping ability without requiring any prior domain knowledge. However, ANNs based on the empirical risk minimization principle often suffer from over-fitting problems. Furthermore, when derivative-based optimization techniques are used for training ANNs, they are likely to be trapped in a local minimum. Compared to ANNs, over-fitting problems can be overcome in SVR because it is based on the structural risk minimization principle [16,17], i.e., the learning error and generalization capability are considered simultaneously.

The importance of ELF and its difficulty have motivated extensive studies, but the problem of input selection for ELF models remains an open question. In ELF, a major issue is selecting significant inputs (SIs) from many initial input candidates (IICs). Three main reasons necessitate input selection procedures [18,19]. First, properly selected inputs decrease the complexity of the model, and thus facilitate more efficient learning. Second, the forecasting performance can be improved by removing redundant inputs that are irrelevant to the outputs and dependent on other inputs. Finally, we can gain valuable insights into the fundamental features of loads and their prediction mechanism. In the following, we summarize several previous studies on input selection methods for load forecasting.

Ghofrani et al. [20] proposed a new input selection framework by combining correlation analysis and the l^2-norm for short-term load forecasting (STLF). Koprinska et al. [21] used mutual information (MI), RReliefF, and a correlation-based method for feature selection of the load forecasting model. Sheikhan and Mohammadi [22] developed a feature selection method by combining a genetic algorithm with ant colony optimization for STLF. Tikka and Hollmén [23] proposed sequential backward input selection for ELF, which is based on a linear model and a cross-validation resampling procedure. Sorjamaa et al. [24] combined a direct prediction strategy with three input selection criteria, i.e., k-nearest neighbors approximation method, MI, and nonparametric noise estimation, for ELF. Da Silva et al. [25] used filter methods based on phase-space embedding and MI and Bayesian wrapper methods for ANN-based ELF. Hu et al. [19] proposed a hybrid filter-wrapper approach with a partial MI and firefly algorithm for STLF feature selection.

In general, the methods mentioned above can be categorized as filter (model-free) or wrapper (model-based) methods [26–29]. In filter methods, after statistical analyses between the potential inputs and output, the inputs with strong relationships to the output are selected as SIs. In wrapper methods, the accuracy of learning machines is used as a selection criterion and the best input combinations are explored by sequential search procedures, such as backward elimination, forward selection, stepwise selection, and metaheuristics. Filter methods can select SIs without learning machines, but these methods do not consider the prediction accuracy, and thus they often perform worse than wrapper methods. Linear correlation (LC) analysis, frequently used in filter methods, with nonlinear predictors can degrade the forecasting performance because they cannot identify nonlinear relationships between inputs and output. In wrapper approaches, the best input combinations are determined by the sequential search procedures, so the computational complexity grows exponentially with the number of IICs. Furthermore, depending on the structure and parameter identification methods used for wrappers, biased input selection results may be obtained.

In summary, instead of only considering the one-to-one statistical correlations, input selection methods for data-driven load forecasting should be able to tackle both the many-to-one nonlinear relationships between dependent and independent variables and the accuracy of the forecasting machines.

In this paper, we propose a new input selection procedure based on the group method of data handling (GMDH) and the bootstrap method for ELF. Although there are several previous studies where GMDH is employed for ELF [30,31], this paper is the first attempt to use GMDH only for input selection. Generally, GMDH networks have been widely used for modeling the nonlinear relationships between variables. Compared with the previous approaches, this paper focuses on the fact that, based only on a given dataset, the GMDH algorithm can not only determine the network structures but also select the inputs with significant explanatory powers for predicting the outputs. In the GMDH algorithm, the GMDH networks where polynomial neurons are hierarchically connected with each other are automatically constructed using learning datasets collected from target systems. Among a number of IICs, the remaining elements in the input layer of the finally constructed GMDH network are considered as relevant inputs. The learning dataset should be divided into training and test datasets to construct the GMDH network. In this study, bootstrap method is applied for the data division. The learning dataset is divided randomly by bootstrap sampling, so the relevant inputs are different each time. Therefore, in the proposed method, after constructing GMDH networks several times using bootstrapping, the inputs that appear frequently in the input layers of the completed networks are finally selected as the SIs.

The main advantages of the proposed input selection method combining the GMDH and bootstrap method can be briefly summarized as follows. First, based on the GMDH network structures in which the polynomial neurons are hierarchically connected with each other (see Figure 1), the method can select significant inputs by taking many-to-one nonlinear relationships between explanatory and target variables into consideration. Second, since prediction accuracy is used as the input selection criterion (see Section 2.2), it is expected that the method achieves improved forecasting performance compared to filter methods based only on the statistical analyses.

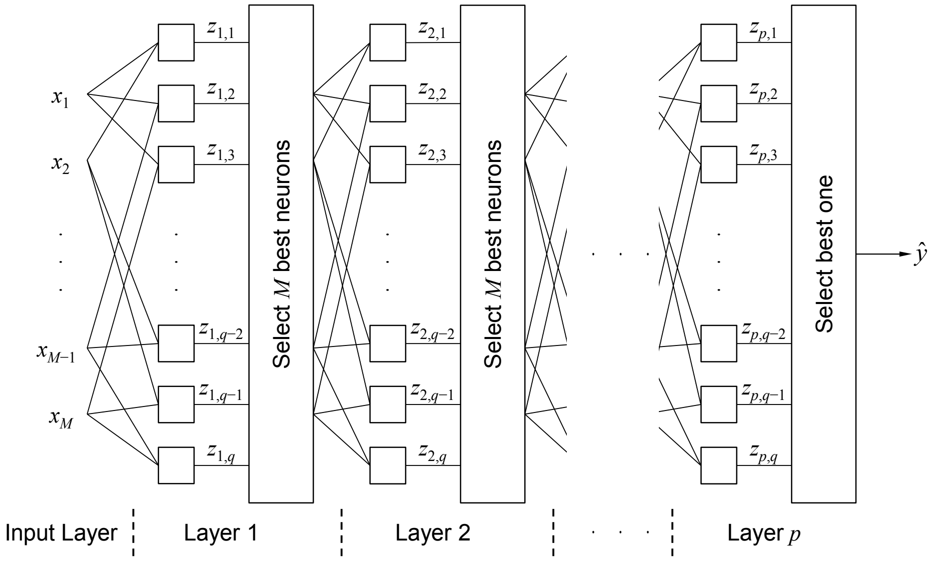


Figure 1. Group method of data handling (GMDH) network. In each layer, polynomial neurons are described by squares and they constitute a hierarchical and forward network structure. Here, the number of input variables entering each neuron is limited to 2.

To verify the performance of the proposed method, we use hourly load data from South Korea with seasonality, weekly and daily periodicity. LC- and MI-based filter methods are employed as comparison methods. In addition, hybrid approaches that combine the filter methods with the proposed method are also examined. In total, five input selection methods are investigated. Prediction models are constructed via ν-SVR, and we compare the one-hour, one-day, and one-week-ahead forecasting performance of the five input selection methods.

The remainder of this paper is organized as follows. Section 2 explains the procedure for GMDH network building. In Section 3, the new input selection method based on GMDH and bootstrap sampling, LC- and MI-based filter methods, and hybrid approaches are described. Section 4 briefly summarizes ν-SVR. In Section 5, we present the experimental results, and discuss the results in detail in Section 6. Finally, we give our conclusions in Section 7.

2. Group Method of Data Handling

Ivakhnenko first proposed the GMDH algorithm in 1968, which is a self-organizing modeling technique [32–34]. In the GMDH method, as shown in Figure 1, complex and nonlinear modeling is performed by the GMDH network where polynomial neurons are hierarchically connected in a forward direction.

A quadratic two-variable polynomial is commonly used as a transfer function for each neuron and their coefficients can be estimated by the least-squares method. When the number of entering inputs and/or the order of the polynomials become larger, the number of parameters to be estimated rapidly increases. To build a GMDH network, only the learning dataset collected from target systems is needed, and relevant inputs as well as the network structure can be determined automatically.

Let $D = \{\mathbf{x}_k; y_k\}_{k=1}^{n}$ be a learning dataset for GMDH network building, where $\mathbf{x}$ = $[x_1, x_2, \ldots, x_M]^T$ is an input vector composed of IICs and M is the number of the IICs. In layer 1, (q = $M(M-1)/2$) neurons that consider all combinations of the initial inputs are generated and only the outputs of M neurons that satisfy a selection criterion are used as inputs for the next layer. In the same manner, q neurons are generated from layer 2 based on the M neurons selected in the previous layer. This layer extension process continues in a forward direction until a stopping criterion is satisfied. After stopping the extension process, the output layer and neuron are fixed and all of the elements connected with the output neuron are found sequentially by backward search.

2.1. Parameter Estimation for Polynomial Neurons

In the GMDH network, the output, z, of an arbitrary neuron is calculated as

$$z = f(x_u, x_l; \boldsymbol{\theta}) = \theta_0 + \theta_1 x_u + \theta_2 x_l + \theta_3 (x_u)^2 + \theta_4 (x_l)^2 + \theta_5 x_u x_l, \quad (1)$$

where x_u and x_l are two inputs of the neuron and $\boldsymbol{\theta}$ = $[\theta_0, \ldots, \theta_5]^T$ is composed of its polynomial coefficients. The same learning dataset D is employed repeatedly to estimate parameters for each neuron. After substituting D into (1), the n linear equations are formulated in concise matrix form, i.e., $\mathbf{X}\boldsymbol{\theta}$ = $\mathbf{z}$. Using the least-squares method, the parameter vector $\boldsymbol{\theta}$ can be optimized as

$$\hat{\boldsymbol{\theta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{z}. \quad (2)$$

In (2), if the determinant of $\mathbf{X}^T\mathbf{X}$ is close to zero, then the estimator $\hat{\boldsymbol{\theta}}$ is rather susceptible to round off errors and its performance deteriorates. In this study, the polynomial coefficients are estimated using singular value decomposition, as follows [35,36]:

$$\hat{\boldsymbol{\theta}}^{+} = \mathbf{X}^{+}\mathbf{z} = \mathbf{V}\mathbf{S}^{+}\mathbf{U}^T\mathbf{z}, \quad (3)$$

where $\mathbf{X}^{+}$ is the pseudo-inverse of $\mathbf{X}$, the columns of matrices $\mathbf{V}$ and $\mathbf{U}$ correspond to the eigenvectors of $\mathbf{X}^T\mathbf{X}$ and $\mathbf{X}\mathbf{X}^T$, respectively, and the main diagonal components of $\mathbf{S}^{+}$ are composed of the reciprocals of r singular values.

2.2. Construction of the GMDH Network

To build the GMDH network, layer extension is performed in the forward direction until the stopping criterion is satisfied. Next, the output layer and neuron are fixed and a search for all of the elements connected to the output neuron is conducted by backward tracing, i.e., from the output to input layer. After finishing the search in the backward direction, we can obtain not only the GMDH network structure, but also relevant inputs in the input layer.

In the proposed method, checking error criterion (*CEC*) [37] is employed: (1) to select the neurons whose outputs will be used as inputs for the next layer; and (2) to determine whether the layer extension process should be stopped. The number of layers in the GMDH networks is closely related to the model's complexity. The deeper the networks, the more complex their structures; in this case, many unnecessary inputs can be selected together with significant inputs. On the other hand, if the networks are too shallow, input variables with great explanatory powers can be missed. To calculate the *CEC*, the learning dataset D is separated into dataset A, $D_A = \{\mathbf{x}_k^A; y_k^A\}_{k=1}^{n_A}$, and dataset B, $D_B = \{\mathbf{x}_k^B; y_k^B\}_{k=1}^{n_B}$, using the bootstrap method [38]. In the bootstrap method, after applying n sampling with replacement to D, we obtain n data pairs. Each data pair in D has the same probability of being selected during the sampling process. Among the n sampled data pairs, only unique pairs comprise D_A for training and D_B consists of the remaining data pairs, i.e., $D_B = D \backslash D_A$ for testing. The ratio of D_A relative to D_B is approximately 6:4.

Using D_A, the parameter vector of a neuron is estimated by (3) and the outputs are calculated based on D_B, $\hat{y}_k^B = f\left((x_u)_k^B, (x_l)_k^B; (\hat{\theta}^+)^A\right)$, $k = 1, \ldots, n_B$. The *CEC* of the neuron is computed by

$$CEC = \sqrt{\frac{1}{n_B}\sum_{k=1}^{n_B}(y_k^B - \hat{y}_k^B)^2}. \tag{4}$$

After calculating the *CEC* for q neurons in the ith layer, i.e., $CEC(i, j)$, $j = 1, \ldots, q$, they are arranged in ascending order and only the M neurons with the smallest *CEC* values are selected. If an early stopping condition, $CEC(i-1) \le CEC(i)$, $i \ge 2$, (where $CEC(i) = \min_j\{CEC(i, j)\}$, $j = 1, \ldots, q$), is satisfied or the current layer is equal to the predefined maximum layer, then the layer extension is halted. Then, the output layer p^* and output neuron q^* are selected as follows:

$$p^* = \arg\min_i\{CEC(i)\}, \tag{5}$$

$$q^* = \arg\min_j\{CEC(p^*, j)\}, j = 1, \cdots, q. \tag{6}$$

After selecting the output neuron, all of the elements connected with the neuron are found by a sequential search in the backward direction. In each layer, neurons without any connections to the output neuron are removed and the remaining neurons and their connections are preserved. The GMDH network is finally constructed after finishing the backward search from the output to input layer. Algorithm 1 describes the procedure for constructing the GMDH network.

Algorithm 1. Constructing the GMDH network.

```
Input: learning dataset $D$
Divide $D$ into $D_A$ and $D_B$ using bootstrap method
$I_{\max} \leftarrow$ maximum layer
$BN \leftarrow \{x_1, \ldots, x_M\}$ //where $BN$ denotes the set of 'best neurons'.
for $i$ from 1 to $I_{\max}$
    Create ($q = M(M-1)/2$) neurons in $i$th layer based on $BN$
    for $j$ from 1 to $q$
        Estimate $(\hat{\theta}^{+})^{A}$ of $j$th neuron using $D_A$
        Calculate $CEC(i, j)$ for $j$th neuron using $D_B$
    end
    $CEC(i) = \min_j\{CEC(i, j)\}, j = 1, \ldots, q$
    if $i \geq 2$ then
        if $CEC(i-1) \leq CEC(i)$ then
            break this loop
        end
    end
    $BN \leftarrow M$ neurons in $i$th layers with smallest $CEC(i, j), j = 1, \ldots, q$
end
$p^{*} \leftarrow \text{argmin}_i\{CEC(i)\}$
$q^{*} \leftarrow \text{argmin}_j\{CEC(p^{*}, j)\}$
Remove all neurons in $p^{*}$th layer except for $q^{*}$th neuron
$p \leftarrow p^{*}$
repeat
    $p \leftarrow p - 1$
    Find the neurons in $p$th layer connected with neurons in $(p + 1)$th layer
    Remove all neurons and their connections in $p$th layer except for the founded neurons
until $p == 0$
    return GMDH network structure and a set of relevant inputs, $X = \{x_1, \cdots, x_m\} (\subset \{x_1, \cdots, x_M\})$
```

From the final constructed GMDH network, we can confirm the inputs in the input layer connected with the output neuron and they are regarded as relevant inputs.

3. Input Selection Method

In this study, to select the inputs for ELF models, we employed: (1) LC- and MI-based filter methods, (2) GMDH-based and bootstrapping-based input selection (i.e., the proposed method), and (3) hybrid methods that combine filter methods with GMDH-based input selection.

3.1. Filter Methods Based on LC and MI

In this subsection, we explain the LC- and MI-based filter methods (i.e., comparison methods). In the filter methods, we calculated the ranking function (*RF*) values, $RF(x_j, y)$, $j = 1, \ldots, M$, between all of the potential inputs and output. The inputs with *RF* values that are greater than or equal to the threshold value RF_{th} were selected as SIs. More details regarding the LC and MI used as *RF* are available elsewhere [28,29,39,40]. Among M IICs $\{x_1, \ldots, x_M\}$, m SIs $\{x_{1\text{st}}, x_{2\text{nd}}, \ldots, x_{m\text{th}}\}$ are determined as

$$\{x_{1\text{st}}, x_{2\text{nd}}, \cdots, x_{m\text{th}}\} = \{x_j | RF(x_j, y) \geq RF_{\text{th}}, j = 1, \cdots, M\}, \tag{7}$$

where the condition, $RF(x_{1\text{st}}, y) > \cdots > RF(x_{m\text{th}}, y)$, is satisfied.

3.2. The Proposed Method: GMDH-Based and Bootstrapping-Based Input Selection

As explained in Section 2, the components of the input layer in the finished GMDH network are considered as relevant inputs. Since D is divided randomly into D_A and D_B by bootstrap sampling, the input selection results will vary in each experiment. Moreover, the relevant inputs obtained by constructing only a single network may yield biased results. Thus, in the proposed method, input selection was performed according to the following procedures. Firstly, the GMDH networks were constructed r times in the same experimental conditions. The set of relevant inputs from the ith experiment is denoted as X_i, $i = 1, \ldots, r$ and their union is $X = \cup_{i=1}^{r} X_i$. Then, in the union X, the total number, N_j, of each IIC, x_j, $j = 1, \ldots, M$, is counted. Finally, the inputs for which N_j is greater than or equal to a predefined threshold, N_{th}, were selected as SIs. Algorithm 2 describes the GMDH- and bootstrap-based input selection procedures.

Algorithm 2. GMDH-based and bootstrap-based input selection.

```
Input: learning dataset $D$
$r \leftarrow$ the number of network constructions
$N_{\text{th}} \leftarrow$ threshold value
$X \leftarrow \{\}$
for $i$ from 1 to $r$
    Divide $D$ into $D_A$ and $D_B$ using bootstrap method
    Select relevant input set $X_i$ using Algorithm 1
    $X \leftarrow X \cup X_i$
end
Count the number of $x_j$, $j = 1, \ldots, M$, $N_j$, in $X$
$\{x_{1\text{st}}, x_{2\text{nd}}, \ldots, x_{m\text{th}}\} = \{x_j \mid N_j \geq N_{\text{th}}, j = 1, \ldots, M\}$
        return $\{x_{1\text{st}}, x_{2\text{nd}}, \ldots, x_{m\text{th}}\}$
```

3.3. Hybrid Input Selection Method

The hybrid input selection procedures that combine filter methods and the proposed method were carried out according to two phases. First, many redundant inputs were eliminated by applying LC- or MI-based filter methods to IICs. After then, the proposed input selection procedure described in Algorithm 2 was applied to the remaining inputs.

4. ν-Support Vector Regression

In this section, we briefly summarize ν-SVR [41] as proposed by Schölkopf et al. Let $\{(\mathbf{x}_i; y_i)\}_{i=1}^{l}$ denote a collected learning dataset, where $\mathbf{x}_i \in \Re^n$ is an n-dimensional input vector and $y_i \in \Re^1$ is the target output. The basic idea of SVR is to find a linear regression function after transforming the original space into a high-dimensional feature space, which is defined as

$$f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b, \tag{8}$$

where $\phi(\cdot)$ is a nonlinear mapping function from the input to feature space, and $\mathbf{w}$ and b are parameters of f that should be estimated from the learning dataset. The constrained optimization problem of ν-SVR is defined by introducing two positive slack variables ξ_i and ξ_i^*, as follows [42]:

$$\begin{aligned} &\min_{\mathbf{w},b,\xi,\xi^*,\varepsilon} \frac{1}{2}\|\mathbf{w}\|^2 + C\left(\nu\varepsilon + \frac{1}{l}\sum_{i=1}^{l}(\xi_i + \xi_i^*)\right) \\ &\text{subject to} \begin{cases} \left(\mathbf{w}^T\phi(\mathbf{x}_i) + b\right) - y_i \leq \varepsilon + \xi_i, \\ y_i - \left(\mathbf{w}^T\phi(\mathbf{x}_i) + b\right) \leq \varepsilon + \xi_i^*, \\ \xi_i, \xi_i^* \geq 0, i = 1, \cdots, l, \varepsilon \geq 0. \end{cases} \end{aligned} \tag{9}$$

where $\frac{1}{2}\|\mathbf{w}\|^2$ is a regularization term, the parameter $\nu \in (0, 1]$ controls the number of support vectors and training errors, and C is a regularization constant that determines the tradeoff between the model's complexity and its accuracy. The training errors of regression function f are penalized by ξ_i and $\xi_i{}^*$, if they are larger than ε. The size of ε is traded off against the model's complexity and slack variables via a constant ν [41]. As described in (9), SVR avoids under-fitting and over-fitting by minimizing both the regularization term and training errors.

By introducing Lagrange multipliers α_i and $\alpha_i{}^*$, and applying Karush-Kuhn-Tucker conditions [43], the constrained optimization problem given by (9) is reformulated as follows:

$$\begin{aligned}&\min_{\boldsymbol{\alpha},\boldsymbol{\alpha}^*}\frac{1}{2}(\boldsymbol{\alpha}-\boldsymbol{\alpha}^*)^T\mathbf{Q}(\boldsymbol{\alpha}-\boldsymbol{\alpha}^*)+\mathbf{y}^T(\boldsymbol{\alpha}-\boldsymbol{\alpha}^*)\\&\text{subject to}\begin{cases}\sum\limits_{i=1}^{l}(\alpha-\alpha^*)=0,\ \sum\limits_{i=1}^{l}(\alpha-\alpha^*)=0\leq C\nu,\\ \qquad 0\leq\alpha_i,\alpha_i^*\leq C/l, i=1,\cdots,l.\end{cases}\end{aligned} \tag{10}$$

where $\mathbf{Q}$ is a kernel matrix and its entry in the ith row and jth column is $Q_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$. The kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ defined in the input space is the same as the inner product of $\mathbf{x}_i$ and $\mathbf{x}_j$ in the feature space. The nonlinear mapping and inner products can be calculated in the original input space using the kernel function. In this paper, among the widely used kernel functions, such as polynomial, radial basis function (RBF), and sigmoidal kernels, we employed the RBF kernel function defined as

$$K(\mathbf{x}_i, \mathbf{x}) = \exp\left(-\gamma\|\mathbf{x}_i - \mathbf{x}\|^2\right), \tag{11}$$

where γ controls the width of the RBF function. By solving (10), we can obtain the approximated regression function as follows:

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{l}(\alpha_i^* - \alpha_i)K(x_i, x) + b. \tag{12}$$

The prediction accuracy of ν-SVR depends on the selection of appropriate design parameters, i.e., ν, C, and γ. Trial and error, cross-validation, grid search, and global optimization have been applied widely for determining the parameter values [2]. The aim of this paper is to examine the performance of the proposed input selection methods, so the procedures for selecting the design parameters will not be described in detail. In this study, the method presented in [3] is used to determine the design parameters.

5. Experimental Results

To verify the performance of the proposed method, hourly load data from South Korea between 1 January 2012 and 31 December 2014 was used. Figure 2 shows the hourly load curves for South Korea during one year (i.e., 2013) and Figure 3 shows an example of the weekly profile for the curves during three weeks from 7 January 2013 until 27 January 2013.

As shown in Figures 2 and 3, the target loads exhibit clear seasonality with weekly and daily periodicities. The load demand is usually higher in the summer and winter seasons compared with the spring and autumn seasons. The demand for electricity is higher on weekdays (from Monday to Friday) compared with weekends and the load demand is slightly higher on Saturday than Sunday. The minimum load on Monday is lower than that on other working days. In addition, the load demand on special days (e.g., national holidays and election days) exhibit remarkably unusual behavior. In this paper, we are not concerned with hourly load forecasting on special days, i.e., we focus only on hourly load forecasting for ordinary days.

To validate the seasonal forecasting performance, we select three months as validation months, i.e., April, July and November in 2014, where the objective is to predict their hourly load demands one hour, one day, and one week ahead. For the one-day and one-week-ahead forecasts that correspond to multi-step-ahead predictions, we employed the recursive strategy [44] explained in Appendix A.

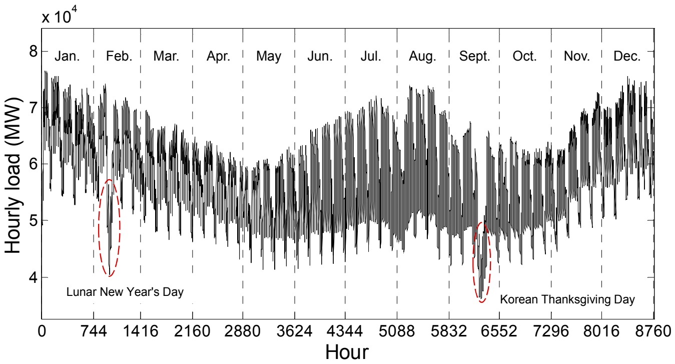


Figure 2. Hourly load curves for South Korea from 1 January 2013 to 31 December 2013. Dashed red lines indicate the main holidays in South Korea, i.e., Lunar New Year's Day and Korean Thanksgiving Day.

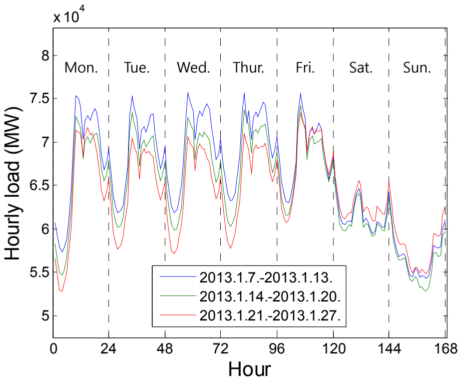


Figure 3. Example of the weekly load profile from South Korea over three weeks, i.e., from 7 January 2013 to 27 January 2013. The weekly and daily periodicities are clearly evident.

5.1. Data Preparation

To prepare the learning dataset, we considered historical time-series load data $\{\varphi_t, t = 1, \ldots, N\}$. For example, to predict the hourly load demand from 1 to 30 April in 2014, the load data from 1 January 2012 to 31 March 2014 is employed to prepare the learning data. After organizing the learning data matrix described in (13) using N historical data, the max-min normalization process is applied to each column as presented in (14).

$$\mathbf{\Phi} = \begin{bmatrix} \varphi_d & \varphi_{d-1} & \cdots & \varphi_1 & \varphi_{d+1} \\ \varphi_{d+1} & \varphi_d & \cdots & \varphi_2 & \varphi_{d+2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \varphi_{N-1} & \varphi_{N-2} & \cdots & \varphi_{N-d} & \varphi_N \end{bmatrix}. \tag{13}$$

$$\Phi'_{i,j} = \frac{\Phi_{i,j} - \min\limits_i\{\Phi_{i,j}\}}{\max\limits_i\{\Phi_{i,j}\} - \min\limits_i\{\Phi_{i,j}\}}. \tag{14}$$

where $\Phi_{i,j}$ is the original component in the ith row and jth column of $\mathbf{\Phi}$ and $\Phi'_{i,j}$ is its normalized value. The learning data matrix is an $(N - d) \times (d + 1)$ matrix, where d is the window size, i.e., the number of IICs. In this paper, to capture the daily and weekly periodicities of the target load, we chose a one-week window size, i.e., $d = 168$. The last column in $\mathbf{\Phi}$ is composed of the desired outputs and the remaining columns consist of 168 IICs. Each row vector in $\mathbf{\Phi}$ corresponds to an input-output learning data pair.

To take account of the seasonality, we reformulated a new matrix $\mathbf{\Phi}_{\text{new}}$ composed of the row vectors of $\mathbf{\Phi}$, where the last column's components (i.e., desired outputs) only correspond to the load values of a validation month and the previous month. For example, to forecast the hourly load during April in 2014, $\mathbf{\Phi}_{\text{new}}$ is made up of row vectors where the desired outputs correspond to the hourly load during March and April in 2012 and 2013 and March in 2014.

Finally, the row vectors that contain hourly load values on special days are removed from $\mathbf{\Phi}_{\text{new}}$. In other words, if only one component of the row vectors corresponds to the hourly load for special days, the row vectors are discarded. This is necessary because the hourly load curves for special days have different shapes from those for normal days. If the prediction models for normal days are trained by learning data with load values on special days, biased forecasting results could be obtained.

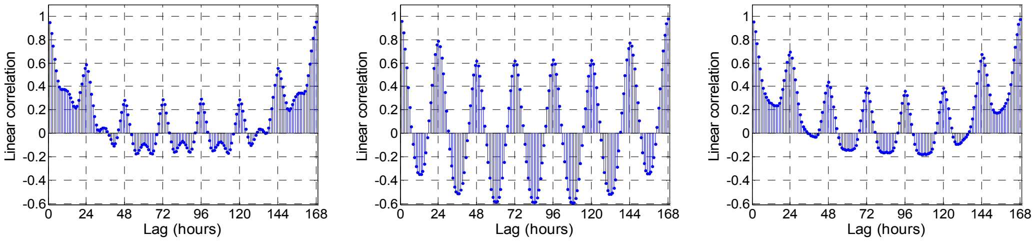
The input selection methods explained in Section 3 were applied to the learning dataset prepared according to the procedures described above.

5.2. Input Selection Results

In this subsection, we present the results of applying the input selection methods to the prepared learning dataset. As explained in Section 3, we employed five input selection methods, which are abbreviated as follows.

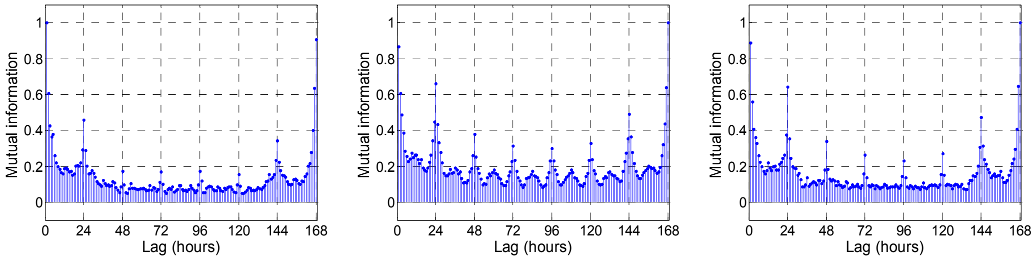
(1) 'LC': LC-based filter method.
(2) 'MI': MI-based filter method.
(3) 'GMDH': the proposed method described in Algorithm 2.
(4) 'LC + GMDH': hybrid input selection method combining 'LC' and 'GMDH'.
(5) 'MI + GMDH': hybrid input selection method combining 'MI' and 'GMDH'. (Note: The acronyms, LC and MI, denote the RFs, but the abbreviations, 'LC' and 'MI', refer to the LC- and MI-based filter methods.)

Figures 4 and 5 show the LC and MI values calculated using the prepared learning dataset for predicting the loads in April, July and November in 2014.



(a) (b) (c)

Figure 4. Linear correlation (LC) values calculated using the prepared learning dataset for predicting the loads in: (**a**) April 2014; (**b**) July 2014; (**c**) November 2014.



(a) (b) (c)

Figure 5. Normalized mutual information (MI) values calculated using the prepared learning dataset for predicting the loads in: (**a**) April 2014; (**b**) July 2014; (**c**) November 2014.

In Figure 5, the MI values are normalized in a range of [0, 1]. Figures 4 and 5 demonstrate that the shapes of LC and MI differ from the validation months because the target loads exhibit distinct seasonal behaviors. Daily and weekly periodicities can also be observed in these figures. In the filter methods based on LC and MI, after rearranging the calculated *RF* values in descending order, *m* inputs with *RF* values greater than or equal to the predefined RF_{th} are finally selected as SIs. Figures 6 and 7 show the input selection results using 'LC' and 'MI'.

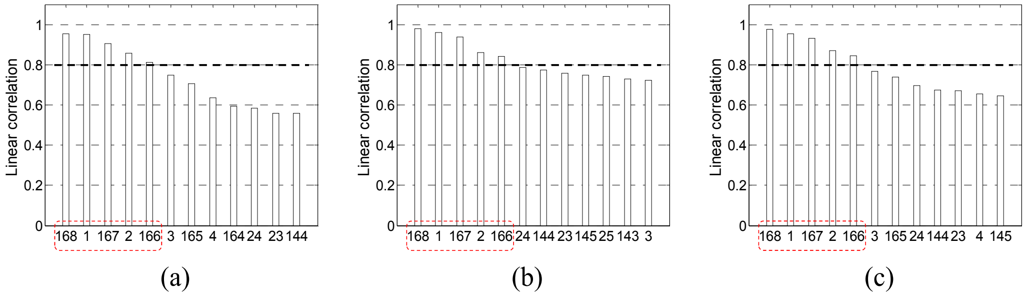


Figure 6. Results of LC-based filter method ('LC') for predicting the loads in: (**a**) April 2014; (**b**) July 2014; (**c**) November 2014. The threshold values and selected inputs are indicated by dashed thick solid black lines and dashed red lines, respectively.

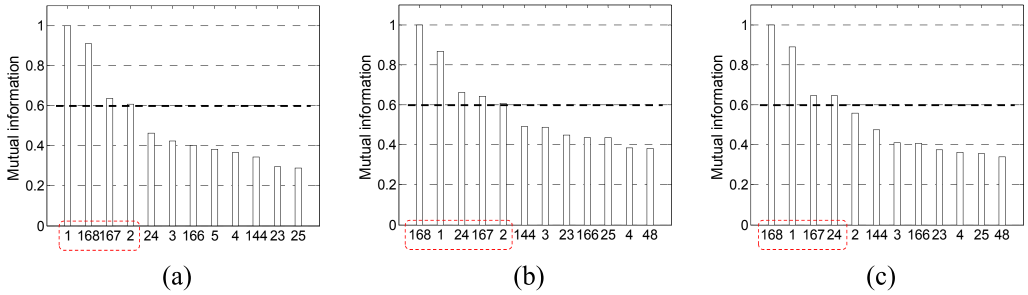


Figure 7. Results of MI-based filter method ('MI') for predicting the loads in: (**a**) April 2014; (**b**) July 2014; (**c**) November 2014.

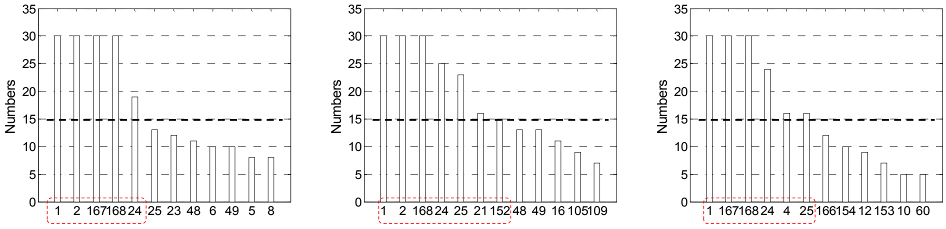
As shown in Figures 6 and 7, the threshold values for LC and MI are set as 0.8 and 0.6, respectively, and they are indicated by dashed thick solid black lines. The selected inputs are indicated by dashed red lines in the figures.

As explained in Section 3.2, in the 'GMDH', GMDH networks were constructed independently *r* times on the same conditions. In this paper, *r* is set to 30. In Table 1, and the maximum layer, $l_{\max}$, is set to 5 based on several trial-and-errors. Note that $l_{\max}$ corresponds to the maximum acceptable depth of the GMDH networks. Through several prior experiments, we confirmed that the forecasting performance may be degraded when $l_{\max}$ is too large (e.g., $l_{\max} \geq 7$) or too small (e.g., $l_{\max} \leq 3$). If $l_{\max}$ is set to be excessively large values, and consequently, too many inputs are selected, then it is difficult to intuitively understand the predictive principles of target data. Table 1 lists the results of applying Algorithm 1 for 30 times to the prepared learning dataset for predicting the loads in April 2014.

Table 1. Example illustrating the application of Algorithm 1 30 times independently to the prepared learning dataset for predicting the loads in April 2014. The IICs and output are $\{\varphi_{t-d+1}, d = 1, 2, \ldots, 168\}$ and φ_{t+1}, respectively.

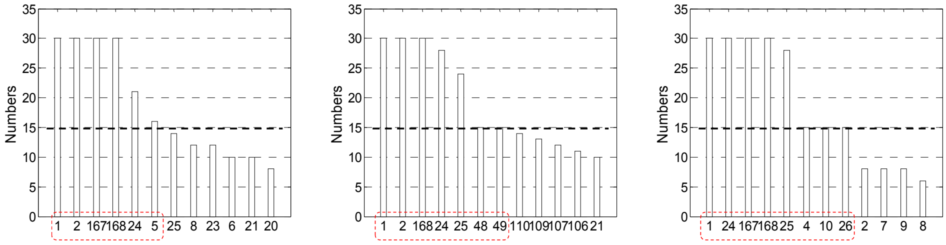
Experiment, *i*	Set of Relevant Inputs, X_i
1	$X_1 = \{\varphi_{t-d+1} \mid d = 1, 2, 5, 7, 8, 22, 23, 135, 167, 168\}$
2	$X_2 = \{\varphi_{t-d+1} \mid d = 1, 2, 6, 7, 24, 25, 167, 168\}$
3	$X_3 = \{\varphi_{t-d+1} \mid d = 1, 2, 4, 7, 21, 24, 48, 49, 167, 168\}$
⋮	⋮
30	$X_{30} = \{\varphi_{t-d+1} \mid d = 1, 2, 6, 7, 21, 23, 24, 96, 167, 168\}$

After the relevant input set X_i is joined into the union $(X = X_1 \cup \cdots \cup X_{30})$, the frequency of each element in X, N_j (j = 1, . . . , 168), is counted. Next, N_j are arranged in descending order and the m inputs that satisfy a condition, i.e., $N_j \geq N_{th}$, are finally selected, where N_{th} is the threshold value of N_j. In the hybrid methods, after removing many redundant inputs using 'LC' or 'MI', 'GMDH' is applied to the remaining inputs. In this study, two-thirds of the IICs were filtered by 'LC' or 'MI'. Figures 8–10 show the input selection results using 'GMDH', 'LC + GMDH' and 'MI + GMDH', respectively.



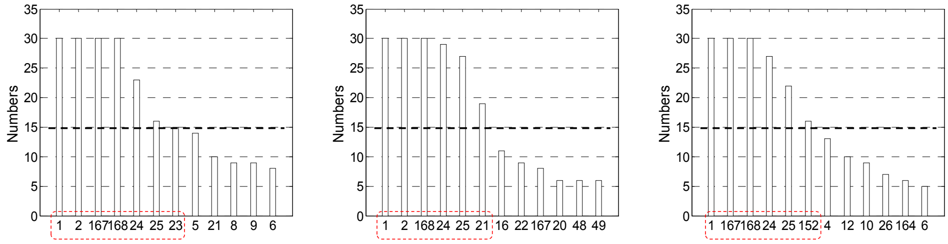
(a) (b) (c)

Figure 8. Results of the proposed method ('GMDH') for predicting the loads in: (**a**) April 2014; (**b**) July 2014; (**c**) November 2014. The threshold values, N_{th}, and selected inputs are indicated by dashed thick solid black lines and dashed red lines, respectively.



(a) (b) (c)

Figure 9. Results of 'LC + GMDH' for predicting the loads in: (**a**) April 2014; (**b**) July 2014; (**c**) November 2014.



(a) (b) (c)

Figure 10. Results of 'MI + GMDH' for predicting the loads in: (**a**) April 2014; (**b**) July 2014; (**c**) November 2014.

As shown in Figures 8–10, the threshold value, N_{th}, is set as 15 and the values are marked by dashed thick solid black lines in the figures. The inputs for which N_j is greater than or equal to N_{th}, i.e., $N_j \geq N_{th}$, are enclosed within dashed red lines. Table 2 summarizes the SIs selected by the five methods for each validation month (i.e., April, July, and November in 2014). As listed in Table 2, the loads in the same hour during the past several days (i.e., φ_{t-23} and φ_{t-167}) as well as several recent loads (i.e., φ_t and φ_{t-1}) are useful for predicting future loads.

Table 2. Significant inputs (SIs) selected for each validation month using the five methods: 'LC', 'MI', 'GMDH', 'LC + GMDH' and 'MI + GMDH'. The IICs and output are $\{\varphi_{t-d+1}, d = 1, 2, \ldots, 168\}$ and φ_{t+1}, respectively.

Validation Month	Method	Set of Selected SIs	No.
April 2014	'LC'	$\{\varphi_{t-d+1} \mid d = 1, 2, 166, 167, 168\}$	5
	'MI'	$\{\varphi_{t-d+1} \mid d = 1, 2, 167, 168\}$	4
	'GMDH'	$\{\varphi_{t-d+1} \mid d = 1, 2, 24, 167, 168\}$	5
	'LC + GMDH'	$\{\varphi_{t-d+1} \mid d = 1, 2, 5, 24, 167, 168\}$	6
	'MI + GMDH'	$\{\varphi_{t-d+1} \mid d = 1, 2, 23, 24, 25, 167, 168\}$	7
July 2014	'LC'	$\{\varphi_{t-d+1} \mid d = 1, 2, 166, 167, 168\}$	5
	'MI'	$\{\varphi_{t-d+1} \mid d = 1, 2, 24, 167, 168\}$	5
	'GMDH'	$\{\varphi_{t-d+1} \mid d = 1, 2, 21, 24, 25, 152, 168\}$	7
	'LC + GMDH'	$\{\varphi_{t-d+1} \mid d = 1, 2, 24, 25, 48, 49, 168\}$	7
	'MI + GMDH'	$\{\varphi_{t-d+1} \mid d = 1, 2, 21, 24, 25, 168\}$	6
November 2014	'LC'	$\{\varphi_{t-d+1} \mid d = 1, 2, 166, 167, 168\}$	5
	'MI'	$\{\varphi_{t-d+1} \mid d = 1, 24, 167, 168\}$	4
	'GMDH'	$\{\varphi_{t-d+1} \mid d = 1, 4, 24, 25, 167, 168\}$	6
	'LC + GMDH'	$\{\varphi_{t-d+1} \mid d = 1, 4, 10, 24, 25, 26, 167, 168\}$	8
	'MI + GMDH'	$\{\varphi_{t-d+1} \mid d = 1, 24, 25, 152, 167, 168\}$	6

In addition to the selected SIs listed in Table 2, we also employed binary-valued vectors used in [45], i.e., $\mathbf{w} \in \{0,1\}^7$ and $\mathbf{h} \in \{0,1\}^{24}$, for ELF models to keep track of the daily and weekly periodicities, where **w** and **h** are vectors of zero with a 1 in the position of the day of the week and the hour of the day for the load under consideration, respectively.

5.3. Forecasting Results

After conducting the input selection procedures, ν-SVR learning was carried out with the SIs and binary-valued vectors, **w** and **h**. For example, when the load demand during April 2014 was predicted with 'LC', the input vector for ν-SVR corresponds to $\mathbf{x} = [\varphi_t, \varphi_{t-1}, \varphi_{t-165}, \varphi_{t-166}, \varphi_{t-167}, \mathbf{w}_{t+1}, \mathbf{h}_{t+1}]^T$. In this paper, we used LIBSVM [46] to implement ν-SVR. The accuracy of the ELF was measured using six performance indices, which were also employed in [2,3], i.e., mean absolute percentage error (MAPE), symmetric mean absolute percentage error (SMAPE), mean absolute error (MAE), normalized mean squared error (NMSE), relative error percentage (REP) and magnitude of maximum error (MME). In the following, we only present the results of the performance comparisons using MAPE, which is defined as

$$\mathrm{MAPE} = \frac{1}{N}\sum_{i=1}^{N} \frac{|A_i - F_i|}{A_i} \times 100, \tag{15}$$

where A_i and F_i are the actual and predicted values for the ith validation dataset, respectively, and N is the number of validation dataset. The performance comparisons using the other indices are presented in Appendix B. Table 3 and Figure 11 illustrate the MAPE values of each validation month and the overall values for five input selection methods.

In Table 3, the best entries among 'LC', 'MI' and 'GMDH' are indicated by the superscript plus sign and the best entries among the five input selection methods are highlighted by the superscript asterisk. When $H = 1$, i.e., in the case of one-hour ahead forecasting, MAPE values of every validation month are similar to each other; but when H = 24 and 168, they are quite different. This can be attributed to the fact that load variations in July and November are higher than those in April because of seasonal effects; it can be observed that MAPE values for multi-step ahead forecasting in July and November are larger than those in April.

Table 3. Mean absolute percentage error (MAPE) values of each validation month and the overall values. The last three rows indicate the overall MAPE values for three validation months and the second column represents the prediction horizons, i.e., H = 1, H = 24, and H = 168, which correspond to one-hour, one-day and one-week-ahead forecasting, respectively.

Validation Month	H	'LC'	'MI'	'GMDH'	'LC + GMDH'	'MI + GMDH'
April 2014	1	0.559	0.545	0.526 $^{+}$	0.531	0.496 *
	24	1.361	1.321	1.252 $^{+}$	1.265	1.162 *
	168	1.675	1.656	1.604 $^{+}$	1.604	1.559 *
July 2014	1	0.459	0.474	0.457 $^{+}$	0.478	0.438 *
	24	1.759	1.576	1.514 $^{+}$	1.678	1.434 *
	168	2.108	1.900 $^{+}$	1.925	2.025	1.834 *
November 2014	1	0.582	0.556	0.544 $^{+}$	0.548	0.534 *
	24	2.047	1.666 $^{+,*}$	1.725	1.728	1.676
	168	2.485	2.301 $^{+}$	2.371	2.393	2.293 *
Overall	1	0.532	0.524	0.508 $^{+}$	0.519	0.489 *
	24	1.723	1.522	1.497 $^{+}$	1.558	1.424 *
	168	2.090	1.952 $^{+}$	1.966	2.008	1.894 *

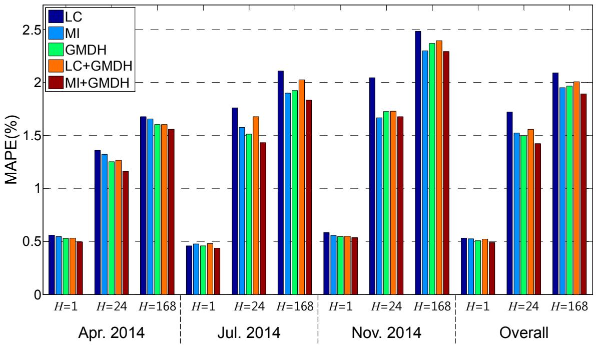


Figure 11. Bar chart showing the mean absolute percentage error (MAPE) values for each validation month and the overall values. H = 1, H = 24, and H = 168 correspond to one-hour, one-day and one-week-ahead forecasting, respectively.

As illustrated in Table 3 and Figure 11, 'GMDH' shows the best prediction performance among 'LC', 'MI', and 'GMDH' in terms of the overall MAPE, except for one-week-ahead forecasting. Among the five methods, 'MI + GMDH' shows the best forecasting performance, except for the one-day-ahead forecast for November 2014. Let us look at how much 'MI + GMDH' improves the overall MAPE compared with the other methods. For one-hour-ahead forecasting, the percentage improvements with 'MI + GMDH' are 8.19%, 6.79%, 3.80%, and 5.72% compared with the four methods, respectively. For one-day and one-week-ahead forecasting, 'MI + GMDH' improves the overall MAPE compared with the other methods by 17.35%, 6.45%, 4.91%, and 8.64%, and by 9.34%, 2.93%, 3.64%, and 5.65%, respectively. Figure 12 shows box plots of the overall absolute percentage errors. As shown in Figure 12, 'MI + GMDH' exhibits superior performance compared with the other methods. Figure 13 shows examples of the actual and predicted load curves obtained using 'MI + GMDH' and ν-SVR. Due to space constraints, the load curves for the other validation months and periods are not presented. In Figure 13, the load values predicted by 'MI + GMDH' and ν-SVR are very similar to the actual values. Figure 14 shows a histogram of illustrating the one-hour-ahead prediction errors by 'MI + GMDH' and ν-SVR for July 2014.

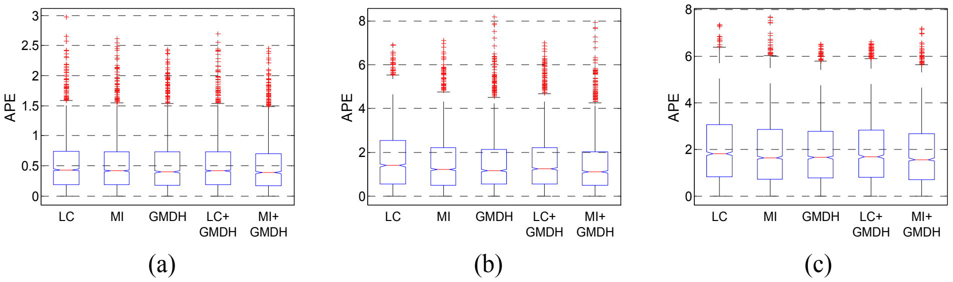


Figure 12. Overall absolute percentage errors for the three validation months based on comparisons between the actual and predicted loads: (**a**) $H = 1$; (**b**) $H = 24$; (**c**) $H = 168$.

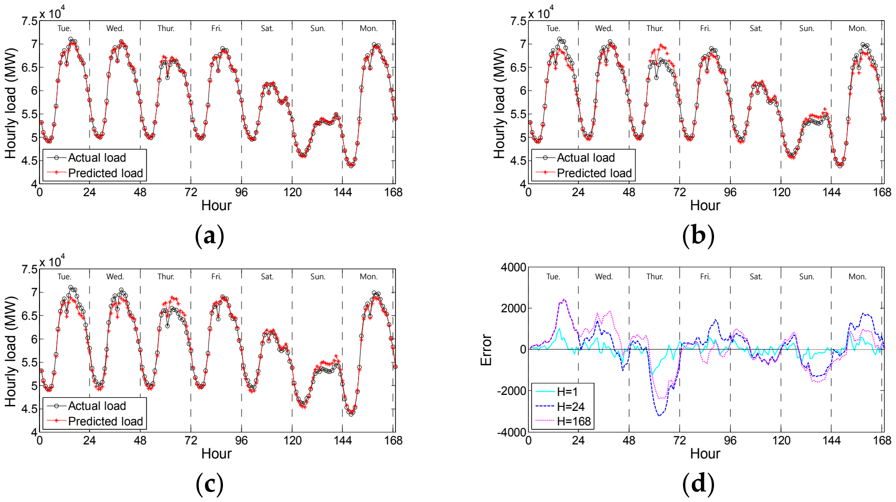


Figure 13. Examples of the actual and predicted hourly load curves obtained using 'MI + GMDH', and their errors from 1 to 7 July 2014: (**a**) $H = 1$; (**b**) $H = 24$; (**c**) $H = 168$; (**d**) Errors.

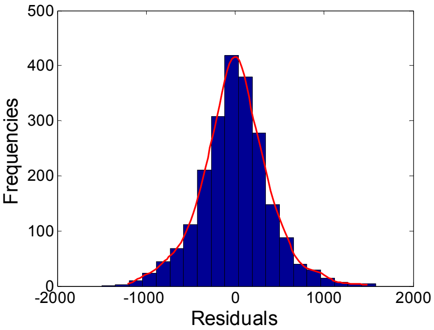
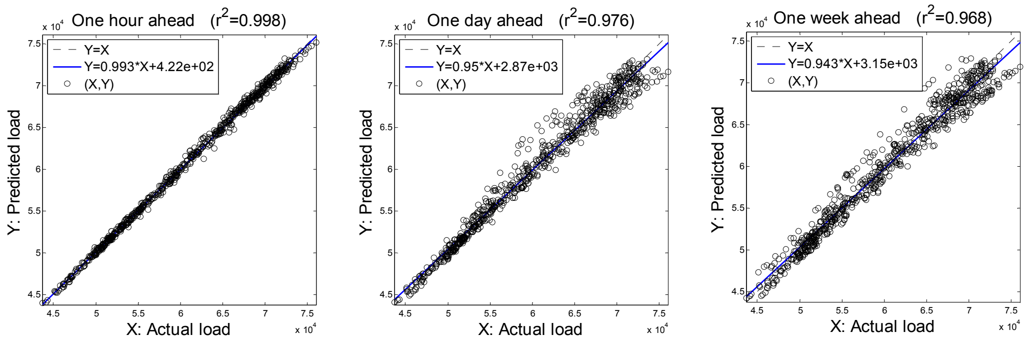


Figure 14. Histogram showing the one-hour-ahead prediction errors (=actual loads − predicted loads) by 'MI + GMDH' and ν-SVR for July 2014. The solid red line is estimated by the kernel method.

As shown in Figure 14, the errors are centered symmetrically on zero and there are no severe outliers. Figure 15 illustrates the results of linear regression analysis between the actual and predicted load values by 'MI + GMDH' and ν-SVR for July 2014.

The results for the other validation months are not presented due to space limitations. In Figure 15, the X and Y axes correspond to the actual and predicted load values, respectively, the black circles are scatter plots of the data points, i.e., (X, Y), and the dashed black lines and thick solid blue lines indicate the straight lines, $Y = X$, and the best linear regression lines, respectively. The r^2 values presented above the figures represent the relationships between X and Y. A value close to 1 indicates that X and Y have a strong linear relationship. As illustrated in Figure 15, we can confirm that there are strong linear relationships between the actual and predicted load values.



(a) (b) (c)

Figure 15. Linear regression analysis between the actual and predicted load values by 'MI + GMDH' and ν-SVR for July 2014: (**a**) H = 1; (**b**) H = 24; (**c**) H = 168.

6. Discussion

Based on the experimental results presented in Section 5, we can highlight several key findings. Let us begin with the comparisons between the filter methods (i.e., 'LC' and 'MI') and 'GMDH'. At one-hour and one-day ahead forecasting, 'GMDH' performed better than the filter methods in terms of the overall MAPE. There are two main reasons for the improved performance of 'GMDH': (1) 'GMDH' can capture many-to-one nonlinear relationships between inputs and output via hierarchical network structure, whereas 'LC' or 'MI' can only capture one-to-one linear or nonlinear relationships; and (2) unlike filter methods that select SIs based on statistical *RF*, the prediction accuracy (i.e., *CEC*) is employed as an input selection criterion in 'GMDH'.

Second, we consider the performance comparisons between 'GMDH' and the hybrid methods. In all cases, 'MI + GMDH' obtained the best forecasts in terms of the overall MAPE. The load series has nonlinear characteristics and ν-SVR with a nonlinear kernel (i.e., RBF kernel) was used for the prediction models, so the hybrid method that combines 'MI' with 'GMDH' performed better. The prediction results of 'LC + GMDH' were poor compared with 'GMDH' alone because LC-based filtering procedure may remove the inputs with weak linear but strong nonlinear relationships with the output.

Finally, let us discuss the performance of the five methods in terms of their computational time requirements. Table 4 lists the computational time required by each input selection method for April 2014.

Table 4. Computational time required by the five input selection methods for April 2014.

Input Selection Methods	Time
'LC'	9.863 s
'MI'	3402.265 s
'GMDH'	1483.911 s
'LC + GMDH'	189.622 s
'MI + GMDH'	3572.328 s

The learning dataset is composed of 2904 input-output pairs and 168 IICs comprise the input part for each pair. The computer used to measure the computational time had 8 GB of RAM and a 2.8 GHz quad-core CPU. The result shows clearly that the computational efficiency of 'GMDH' is higher than that of 'MI'. The computational time required by 'LC + GMDH' is much less than that of 'GMDH' alone because the number of inputs handled by 'GMDH' is reduced to one-third of the IICs by the LC-based filtering procedure.

7. Conclusions

In this study, we proposed a new input selection procedure, which combines GMDH with a bootstrap method for SVR-based short-term hourly load forecasting. After constructing GMDH networks many times under the same experimental conditions, the inputs that remain frequently in the input layers were finally selected as SIs. The networks were constructed several times because each relevant input is different due to the random division of the learning dataset. Indeed, only constructing a single network could yield biased input selection results. In experimental assessments, we employed LC- and MI-based filter methods for comparison, and also verified the performance of two hybrid methods. In total, five input selection methods were examined in this study. To illustrate the performance of the proposed method, an hourly load dataset from South Korea was used and the one-hour, one-day and one-week-ahead forecasting performances were compared.

The experimental results showed that the proposed method can select SIs in an effective manner. The forecasting performance of the proposed method (i.e., 'GMDH') was better than that of the LC- and MI-based filter methods. To be specific, in one-hour, one-day, and one-week ahead forecasts, 'GMDH' improves the overall MAPE values by 0.024, 0.226, and 0.124, respectively, over 'LC'. Although the overall MAPE with H = 168 worsens by 0.014, when H = 1 and 24, 'GMDH' improves the overall MAPE values by 0.016 and 0.025, respectively, compared with 'MI'. In addition, the computational efficiency of the proposed method was higher than that of the MI-based filter method. Among the five methods, 'MI + GMDH' achieved the best prediction accuracy. When H = 1, 24, and 168, 'MI + GMDH' outperforms 'GMDH', with 0.019, 0.073, and 0.072, respectively, improvements in overall MAPE values.

Let us summarize the main reasons for the improved performance of the proposed method. First, the proposed method can perform input selection by capturing many-to-one nonlinear relationships between potential inputs and output via the hierarchical structure of the GMDH network. Second, compared with the filter methods, the prediction accuracy (i.e., *CEC*) is employed as a selection criterion, which improves the prediction performance. Finally, SIs are selected by constructing GMDH networks many times, thereby facilitating more robust input selections.

In future research, the proposed method will be applied to hourly load forecasting on special days and daily peak load forecasting. Moreover, the proposed method can be used in various real-world applications such as financial time-series analysis, process monitoring, and nonlinear function approximations.

Author Contributions: J.Y. analyzed the data and wrote the paper. The analysis results and the paper were supervised by J.H.P. and S.K.

Acknowledgments: This work was supported by the Human Resources Program in Energy Technology of the Korea Institute of Energy Technology Evaluation and Planning (KETEP), granted financial resources from the Ministry of Trade, Industry & Energy, Republic of Korea. (No. 20174030201770).

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Recursive Strategy for Multi-Step-Ahead Prediction

In H-step-ahead forecasting, the objective is to predict H future time-series data, $\{\varphi_t, t = N + 1, \ldots, N + H\}$, using N historical data, $\{\varphi_t, t = 1, \ldots, N\}$, where $H \geq 1$ is called the prediction horizon, i.e., the values of H for one-hour, one-day, and one-week-ahead hourly load forecasting correspond to 1, 24, and 168, respectively.

In the recursive strategy [44], after constructing one-step-ahead prediction models, multi-step predictions are performed by feeding back the predicted values recursively as inputs. The following describes a one-step-ahead time-series predictor:

$$\varphi_{t+1} = f(\varphi_t, \cdots, \varphi_{t-d+1}) + w, \tag{A1}$$

where only the selected inputs are used for the predictor f, w is the zero-mean error term, and d is the window size, i.e., the number of IICs. After building the predictor using the historical dataset, H-step-ahead predictions are carried out as follows:

$$\hat{\varphi}_{N+h} = \begin{cases} \hat{f}(\varphi_N, \cdots, \varphi_{N-d+1}) & \text{if } h = 1 \\ \hat{f}(\hat{\varphi}_{N+h-1}, \cdots, \hat{\varphi}_{N+1}, \varphi_N, \cdots, \varphi_{N-d+h}) & \text{if } h \in \{2, \cdots, d\} \\ \hat{f}(\hat{\varphi}_{N+h-1}, \cdots, \hat{\varphi}_{N-d+h}) & \text{if } h \in \{d+1, \cdots, H\} \end{cases} \tag{A2}$$

Appendix B. Performance Evaluation Using Five Indices Excluding MAPE

Excluding MAPE, the performance indices used in this study, i.e., SMAPE, MAE, NMSE, REP, and MME, are defined as follows:

$$\text{SMAPE} = \frac{1}{N}\sum_{i=1}^{N} \frac{|A_i - F_i|}{(|A_i| + |F_i|)/2}, \tag{A3}$$

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N} |A_i - F_i|, \tag{A4}$$

$$\text{NMSE} = \frac{1}{\Delta^2 N}\sum_{i=1}^{N} (A_i - F_i)^2, \text{ where } \Delta^2 = \frac{1}{N-1}\sum_{i=1}^{N} (A_i - \overline{A})^2, \tag{A5}$$

$$\text{REP} = \sqrt{\frac{\sum_{i=1}^{N} (A_i - F_i)^2}{\sum_{i=1}^{N} A_i^2}} \times 100, \tag{A6}$$

$$\text{MME} = \max_i (|A_i - F_i|),\ i = 1,\ \cdots,\ N, \tag{A7}$$

where A_i and F_i are the actual and predicted values of the ith validation dataset, respectively, $\overline{A}$ is the mean of the actual values, and N is the number of validation datasets. Table A1 lists the results of performance comparisons of the five input selection methods using the five indices.

Table A1. Performance comparisons of the five input selection methods using the five indices.

Indices	H	'LC'	'MI'	'GMDH'	'LC + GMDH'	'MI + GMDH'
SMAPE	1	0.0053	0.0052	0.0051 $^+$	0.0052	0.0049 *
	24	0.0173	0.0152	0.0150 $^+$	0.0156	0.0142 *
	168	0.0209	0.0195 $^+$	0.0197	0.0201	0.0190 *
MAE	1	314.60	310.02	300.51 $^+$	306.45	289.27 *
	24	1046.18	918.81	903.30 $^+$	943.22	860.06 *
	168	1245.11	1162.16 $^+$	1171.44	1199.36	1128.97 *
NMSE	1	0.0050	0.0048	0.0045 $^+$	0.0046	0.0041 *
	24	0.0508	0.0401	0.0390 $^+$	0.0409	0.0358 *
	168	0.0655	0.0600 $^+$	0.0609	0.0629	0.0580 *
REP	1	0.706	0.697	0.678 $^+$	0.690	0.651 *
	24	2.335	2.081	2.048 $^+$	2.119	1.961 *
	168	2.635	2.511 $^+$	2.518	2.579	2.456 *
MME	1	1710.36	1561.01	1489.93 $^+$	1548.48	1420.95 *
	24	4549.24	4162.12 $^+$	4217.88	4152.85	4061.22 *
	168	4610.24	4484.98	4389.05 $^+$	4502.89	4354.41 *

Due to space limitations, only the overall indices of the three validation months (i.e., April, July, and November in 2014) are presented in Table A1. In Table A1, the best entries among 'LC', 'MI' and 'GMDH' are highlighted by the superscript plus sign, and the best entries among the five input selection methods are accentuated by the superscript asterisk. As listed in Table A1, among the 'LC', 'MI' and 'GMDH', the proposed method improves the four indices except for MME in one-hour and one-day-ahead forecasting, and improves MME in one-hour and one-week-ahead forecasting compared with the filter methods. Among the five methods, the 'MI + GMDH' method obtains the best predictions with respect to all the indices for one-hour, one-day and one-week-ahead forecasting.

References

1. Senjyu, T.; Takara, H.; Uezato, K.; Funabashi, T. One-hour-ahead load forecasting using neural network. *IEEE Trans. Power Syst.* **2002**, *17*, 113–118. [CrossRef]
2. Nagi, J.; Yap, K.S.; Nagi, F.; Tiong, S.K.; Ahmed, S.K. A computational intelligence scheme for the prediction of the daily peak load. *Appl. Soft Comput.* **2011**, *11*, 4773–4788. [CrossRef]
3. Elattar, E.E.; Goulermas, J.; Wu, Q.H. Electric load forecasting based on locally weighted support vector regression. *IEEE Trans. Syst. Man Cybern. C Appl. Rev.* **2010**, *40*, 438–447. [CrossRef]
4. Apostolopoulos, P.A.; Tsiropoulou, E.E.; Papavassiliou, S. Demand Response Management in Smart Grid Networks: A Two-Stage Game-Theoretic Learning-Based Approach. *Mob. Netw. Appl.* **2018**, 1–14. [CrossRef]
5. Shi, W.; Li, N.; Xie, X.; Chu, C.C.; Gadh, R. Optimal residential demand response in distribution networks. *IEEE J. Sel. Areas Commun.* **2014**, *32*, 1441–1450. [CrossRef]
6. Maharjan, S.; Zhu, Q.; Zhang, Y.; Gjessing, S.; Basar, T. Dependable demand response management in the smart grid: A Stackelberg game approach. *IEEE Trans. Smart Grid* **2013**, *4*, 120–132. [CrossRef]
7. Hippert, H.S.; Pedreira, C.E.; Souza, R.C. Neural networks for short-term load forecasting: A review and evaluation. *IEEE Trans. Power Syst.* **2001**, *16*, 44–55. [CrossRef]
8. Taylor, J.W.; Buizza, R. Neural network load forecasting with weather ensemble predictions. *IEEE Trans. Power Syst.* **2002**, *17*, 626–632. [CrossRef]
9. Chen, Y.; Luh, P.B.; Guan, C.; Zhao, Y.; Michel, L.D.; Coolbeth, M.A.; Friedland, P.B.; Rourke, S.J. Short-term load forecasting: Similar day-based wavelet neural networks. *IEEE Trans. Power Syst.* **2009**, *25*, 322–330. [CrossRef]
10. Felice, M.D.; Yao, X. Short-Term load forecasting with neural network ensembles: A comparative study. *IEEE Comput. Intell. Mag.* **2011**, *6*, 47–56. [CrossRef]
11. Chen, B.J.; Chang, M.W.; Lin, C.J. Load forecasting using support vector machines: A study on EUNITE competition 2001. *IEEE Trans. Power Syst.* **2004**, *19*, 1821–1830. [CrossRef]
12. Ceperic, E.; Ceperic, V.; Baric, A. A strategy for short-term load forecasting by support vector regression machines. *IEEE Trans. Power Syst.* **2013**, *28*, 4356–4364. [CrossRef]
13. Fan, G.F.; Peng, L.L.; Hong, W.C.; Sun, F. Electric load forecasting by the SVR model with differential empirical mode decomposition and auto regression. *Neurocomputing* **2016**, *173*, 958–970. [CrossRef]
14. Che, J.; Wang, J. Short-term load forecasting using a kernel-based support vector regression combination model. *Appl. Energy* **2014**, *132*, 602–609. [CrossRef]
15. Ghelardoni, L.; Ghio, A.; Anguita, D. Energy load forecasting using empirical mode decomposition and support vector regression. *IEEE Trans. Smart Grid* **2013**, *4*, 549–556. [CrossRef]
16. Vapnik, V. *The Nature of Statistical Learning Theory*; Springer: Berlin, Germany, 1995; ISBN 9780387987804.
17. Kim, J.S. Vessel Target Prediction Method and Dead Reckoning Position Based on SVR Seaway Model. *Int. J. Fuzzy Logic Intell. Syst.* **2017**, *17*, 279–288. [CrossRef]
18. Sindelar, R.; Babuska, R. Input selection for nonlinear regression models. *IEEE Trans. Fuzzy Syst.* **2004**, *12*, 688–696. [CrossRef]
19. Hu, Z.; Bao, Y.; Xiong, T.; Chiong, R. Hybrid filter–wrapper feature selection for short-term load forecasting. *Eng. Appl. Artif. Intell.* **2015**, *40*, 17–27. [CrossRef]
20. Ghofrani, M.; Ghayekhloo, M.; Arabali, A.; Ghayekhloo, A. A hybrid short-term load forecasting with a new input selection framework. *Energy* **2015**, *81*, 777–786. [CrossRef]
21. Koprinska, I.; Rana, M.; Agelidis, V.G. Correlation and instance based feature selection for electricity load forecasting. *Knowl.-Based Syst.* **2015**, *82*, 29–40. [CrossRef]
22. Sheikhan, M.; Mohammadi, N. Neural-based electricity load forecasting using hybrid of GA and ACO for feature selection. *Neural Comput. Appl.* **2012**, *21*, 1961–1970. [CrossRef]

23. Tikka, J.; Hollmén, J. Sequential input selection algorithm for long-term prediction of time series. *Neurocomputing* **2008**, *71*, 2604–2615. [CrossRef]
24. Sorjamaa, A.; Hao, J.; Reyhani, N.; Ji, Y.; Lendasse, A. Methodology for long-term prediction of time series. *Neurocomputing* **2007**, *70*, 2861–2869. [CrossRef]
25. Da Silva, A.P.A.; Ferreira, V.H.; Velasquez, R.M.G. Input space to neural network based load forecasters. *Int. J. Forecast.* **2008**, *24*, 616–629. [CrossRef]
26. Tran, H.D.; Muttil, N.; Perera, B.J.C. Selection of significant input variables for time series forecasting. *Environ. Model. Softw.* **2015**, *64*, 156–163. [CrossRef]
27. Crone, S.F.; Kourentzes, N. Feature selection for time series prediction–A combined filter and wrapper approach for neural networks. *Neurocomputing* **2010**, *73*, 1923–1936. [CrossRef]
28. Guyon, I.; Elisseeff, A. An introduction to variable and feature selection. *J. Mach. Learn. Res.* **2003**, *3*, 1157–1182.
29. May, R.; Dandy, G.; Maier, H. Review of input variable selection methods for artificial neural networks. *Intech Open* **2011**. [CrossRef]
30. Abdel-Aal, R.E. Short-term hourly load forecasting using abductive networks. *IEEE Trans. Power Syst.* **2004**, *19*, 164–173. [CrossRef]
31. Elattar, E.E.; Goulermas, J.Y.; Wu, Q.H. Generalized locally weighted GMDH for short term load forecasting. *IEEE Trans. Syst. Man Cybern. C Appl. Rev.* **2011**, *42*, 345–356. [CrossRef]
32. Madala, H.R.; Ivakhnenko, A.G. *Inductive Learning Algorithms for Complex Systems Modeling*; CRC Press: Boca Raton, FL, USA, 1994; ISBN 0-8493-4438-7.
33. Mueller, J.A.; Lemke, F. *Self-Organising Data Mining: An Intelligent Approach to Extract Knowledge from Data*; Libri: Hamburg, Germany, 2000; ISBN 9783898118613.
34. Yu, J.; Kim, S. Locally-weighted polynomial neural network for daily short-term peak load forecasting. *Int. J. Fuzzy Logic Intell. Syst.* **2016**, *16*, 163–172. [CrossRef]
35. Burden, R.L.; Faires, J.D. *Numerical Analysis*; Brooks/Cole, Cengage Learning: Boston, MA, USA, 2011; ISBN 9780538733519.
36. Strang, G. *Linear Algebra and Its Applications*; Thomson, Brooks/Cole: Belmont, CA, USA, 2005; ISBN 9780534422004.
37. Chiu, S.L. Selecting input variables for fuzzy models. *J. Intell. Fuzzy Syst.* **1996**, *4*, 243–256. [CrossRef]
38. Han, J.; Kamber, M.; Pei, J. *Data Mining: CONCEPTS and Techniques*; Elsevier: Amsterdam, The Netherlands, 2011; ISBN 9780123814791.
39. Rossi, F.; Lendasse, A.; François, D.; Wertz, V.; Verleysen, M. Mutual information for the selection of relevant variables in spectrometric nonlinear modelling. *Chemom. Intell. Lab. Syst.* **2006**, *80*, 215–226. [CrossRef]
40. Kraskov, A.; Stögbauer, H.; Grassberger, P. Estimating mutual information. *Phys. Rev. E* **2004**, *69*. [CrossRef] [PubMed]
41. Schölkopf, B.; Smola, A.J.; Williamson, R.C.; Bartlett, P.L. New support vector algorithms. *Neural Comput.* **2000**, *12*, 1207–1245. [CrossRef] [PubMed]
42. Chang, C.C.; Lin, C.J. Training ν-support vector regression: Theory and algorithms. *Neural Comput.* **2002**, *14*, 1959–1977. [CrossRef] [PubMed]
43. Meng, Q.; Ma, X.; Zhou, Y. Forecasting of coal seam gas content by using support vector regression based on particle swarm optimization. *J. Nat. Gas Sci. Eng.* **2014**, *21*, 71–78. [CrossRef]
44. Taieb, S.B.; Sorjamaa, A.; Bontempi, G. Multiple-output modeling for multi-step-ahead time series forecasting. *Neurocomputing* **2010**, *73*, 1950–1957. [CrossRef]
45. Espinoza, M.; Suykens, J.A.K.; Belmans, R.; De Moor, B. Electric load forecasting. *IEEE Control Syst.* **2007**, *27*, 43–57. [CrossRef]
46. Chang, C.C.; Lin, C.J. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.* **2011**, *2*. [CrossRef]