symmetry

Article

A Metaheuristics-Based Inputs Selection and Training Set Formation Method for Load Forecasting

Ioannis Panapakidis *, Michail Katsivelakis and Dimitrios Bargiotas

Department of Electrical and Computer Engineering, University of Thessaly, 38221 Volos, Greece
* Correspondence: panapakidis@uth.gr; Tel.: +30-24-2107-4821

Abstract: Load forecasting is a procedure of fundamental importance in power systems operation and planning. Many entities can benefit from accurate load forecasting such as generation companies, systems operators, retailers, prosumers, and others. A variety of models have been proposed so far in the literature. Among them, artificial neural networks are a favorable approach mainly due to their potential for capturing the relationship between load and other parameters. The forecasting performance highly depends on the number and types of inputs. The present paper presents a particle swarm optimization (PSO) two-step method for increasing the performance of short-term load forecasting (STLF). During the first step, PSO is applied to derive the optimal types of inputs for a neural network. Next, PSO is applied again so that the available training data is split into homogeneous clusters. For each cluster, a different neural network is utilized. Experimental results verify the robustness of the proposed approach in a bus load forecasting problem. Also, the proposed algorithm is checked on a load profiling problem where it outperforms the most common algorithms of the load profiling-related literature. During input selection, the weights update is held in asymmetrical duration. The weights of the training phase require more time compared with the test phase.

Keywords: clustering; load forecasting; metaheuristics; neural networks; particle swarm optimization

check for updates

Citation: Panapakidis, I.; Katsivelakis, M.; Bargiotas, D. A Metaheuristics-Based Inputs Selection and Training Set Formation Method for Load Forecasting. *Symmetry* **2022**, *14*, 1733. https://doi.org/10.3390/sym14081733

Academic Editor: Theodore E. Simos

Received: 9 May 2022
Accepted: 13 June 2022
Published: 19 August 2022

1. Introduction

Load forecasting forms the pillar that power systems operation and planning rely on [1]. In day-ahead markets, the system operator provides the official forecasts so that the generation companies will prepare the energy/price bids for the wholesale markets [2]. In the long-term horizon, demand forecasting is vital for generation capacity expansion scenarios in the energy sector [3,4]. Also, in competitive energy markets demand forecasting is vital for aggregators, retailers, and prosumers [5–7]. The importance of load forecasting is reflected in the large number of research studies, pilot programs, and relevant applications [8]. In general terms, forecasting models can be classified into time series models, computational intelligence-based models, and hybrid ones [9]. In time series models, the structure of the model, i.e., a number of time lags, auto-regressive components and other parameters should be known in advance. A special effort should be made to derive the model's structure by utilizing a set of statistical tests. Time series models include ARMA, ARIMA, GARCH, and others. Historically, they were the first to have been proposed in the literature [10–12]. On the other hand, in computational intelligence-based models like neural networks, support vector machines, and others, there are no requirements for an a priori definition of the structure [13–15]. The latter is derived from the training procedure. Hybrid models usually refer to the advancement of the previous categories, where a time series and computational-based model are combined, or a time series processing technique or other method is applied prior to the application of the main forecaster [16–18].

In the literature, there is no single model that outperforms all the rest. Therefore, in order to provide more credible conclusions, a comparison of models takes place. However, the superiority of hybrid models is reported by many researchers. In most cases, the forecaster is a feedforward neural network (FFNN). In the present research landscape, FFNNs have become the most popular model for STLF due to their high prediction efficiency and speed [19]. Also, FFNNs are available in many software packages, a fact that provides the means for further experimentations with different FFNN structures and combinations. The FFNN provides a mapping of the inputs and outputs patterns. The term pattern refers to a vector that is composed of previous load values, external parameters, and the desired load output. The training phase of the FFNN requires many patterns for better machine learning. The mapping of inputs and outputs refers to the mathematical function that describes the data of the scientific problem, whether a classification or regression task. In forecasting problems, the function is nonlinear, i.e., the load is connected in a nonlinear manner with its historical values and external parameters such as outdoor temperature. FFNNs have gathered the attention of researchers in recent years due to their potential for the simulation of complex mathematical functions [20]. However, in neural network applications, special attention should be centered on the robust selection of the types and number of inputs. Historical load data are always a candidate input. The influence of temperature and other parameters should be carefully examined. Input selection can be made based on expertise, literature, survey, trial and error, or a combination of these. A large number of inputs is not preferred since it can lead to high training time and consequently, increased requirements for computational resources. On the other hand, a low number may result in insufficient training and poor performance [21]. In the literature, some studies focus on input selection. According to [22] the input or feature selection techniques can be categorized into stepwise, filter, correlation, mutual information, and optimization algorithms. In [23] the selection is held with the mutual information criterion. The test cases include the loads of the Australian Energy Market Commission, Pennsylvania-New Jersey-Maryland, and North American electric utilities. With the feature selection method, the prediction accuracy improves and outperforms the single application of radial basis network, multi-layer perceptron, ARIMA, and others. In [24] the authors present a hybrid ant colony optimization and genetic algorithm selection method. The candidate inputs include the maximum and minimum temperature, day's rainfall, maximum and minimum humidity, indicators about the month, season, week, weekend, and holiday, and historical loads. The test set refers to a period of 300 days and the area under study is Tehran, Iran. In [25] the analysis focuses on loads of a North American utility. The test case involves several months, i.e., January, April, July, and October. The PSO is employed for input selection; the candidate inputs refer to historical loads, temperatures, and seasonal, daily, and hourly indicators. An approach to building hybrid models is to combine a clustering algorithm and a neural network or other supervised machine learning algorithm. By using a training set that includes a complete annual load time series, the FFNN is trained with a variety of load shapes and load magnitudes, i.e., working days, weekends, holidays, and other loads of special interest. By splitting the training set into sets that include more similar load patterns, the FFNN is trained with patterns of highest correlation and can more easily recognize future demand patterns, i.e., test set patterns. The peak load forecasting problem is the objective of [26]. Clustering is held with the simulated annealing algorithm and the forecasting is held with an FFNN. The patterns for clustering are vectors that include three elements, i.e., the peak load of the current day, the temperature of the test day and the difference in temperatures between the current and the test days. The FFNN includes three additional inputs, i.e., the minimum humidity value of the current day, the forecasted mean temperature, and the forecasted indicator of comfort living. The authors highlight the importance of the utilization of a low number of clusters. In [27] the authors study hourly load forecasting. The paper proposes the use of a neural gas network for clustering and an Elman neural network for the predictions. The inputs for clustering are the hourly loads of the day prior to the test day. Through the self-organizing map (SOM) the yearly load of

Korea in 1995 is clustered in [28]. For each cluster, an auto-regressive model is applied with the mean three-hour temperature, and its square is used as dependent variables. In [29] the combination involves SOM and a radial basis neural network for peak load forecasting of a utility in Japan. The training set is composed of the peak loads of July to September of years 1991–2000, whereas the test set refers to the same period of 2003. In [30] Fuzzy C-means (FCM) is applied in vectors where their elements include temperature values and day-type indicators. A neural network trained by the Levenberg–Marquardt is employed for the peak load forecasting of 2002 of a region in China. The same problem is investigated in [31] but the clustering algorithm is the unweighted pair group method with arithmetic mean (UPGMA). In [32], K-means is combined with a support-vector machine (SVM). The patterns include the load of the current day, the minimum and maximum temperatures of the current and test days, and a four-digit number for categorizing the days into weekday, holiday, Saturday, and Sunday. After clustering, the patterns that are selected for the SVM are the ones that are calendrically closest to the test day. The test set includes the loads of Shanghai, China in the year 2004, whereas the training test includes the loads of the period 2001–2003. In [33] the SOM clusters the training set and for each cluster there are 24 SVMs, and one is utilized for each hour. The patterns used for clustering include hourly loads, temperature and humidity values, mean wind speed, and day-type indicator; the patterns used for forecasting include only load and temperature values. The methodology is applied for two months, namely January and July of 2004. The same model is applied in [34]. Here the test refers to peak loads. In [35], the FCM is combined with SVM for the load forecasting of a city in China.

According to the literature survey, all the papers focus on an hourly load of regions or an aggregated system load. The present paper deals with the presentation of a novel hybrid model for day-ahead bus load forecasting. The bus is located in the city of Thessaloniki, Greece, and serves a mix of residential, commercial, and industrial consumers. Due to the rapid advancements of smart grids and microgrids, there is a need to develop accurate forecasting models for small loads. Bus loads exhibit more volatilities and stochasticities compared with aggregated system loads [36]. Thus, special attention should be placed to develop models that can capture and simulate the volatilities. The candidate input data set of the model considers the inputs of the FFNN that have been proposed for the Greek interconnected system [37]. Employing a binary coded PSO, a variable selection phase is manifested to select the most appropriate inputs from the initial set. Then, a real coded PSO is combined with the k-medoids to cluster the training set into smaller training sets. For each subset, a dedicated FFNN is employed. The input selection scheme is formulated as an optimization problem, where the objective function refers to the squared error between the desired output and the FFNN's output. The decision variables are the weights that connect the neurons of the hidden and output layers. The objective function includes two terms, i.e., the errors during the training and test phases. The number of desired inputs is set in advance; the optimization aims at the selection of the specific types. This fact leads to increased flexibility; based on the available execution time and other characteristics of the forecasting problem under study, the user can set the desired number of inputs and the algorithm optimally selects from a pool of available ones, the inputs that minimize the error.

2. Methodology

2.1. General Description and Inputs

The case study refers to a mix of residential, commercial, and industrial consumers supplied by a high (i.e., 150 kV) to medium (i.e., 20 kV) distribution transformer located within the city of Thessaloniki, Greece. The available load data refer to the period 1 January 2006–31 December 2010. Among them, almost 80% were used as the training set and the remaining 20% served as the test set. More specifically, the training set covers the period from 7 January 2006 to 31 December 2009 and the test set the period between 7 January 2010 to 31 December 2010. Figure 1 shows the time series of the data set. In order to examine a realistic test case, the inputs used by the neural network that provides the official forecasts of the Greek interconnected system were taken into account. The inputs were distinguished into historical loads, temperature, and day-type coding. The load data were provided by the Hellenic Independent Power Transmission Operator S.A. and the temperature data by the Hellenic National Meteorological Service [38,39].

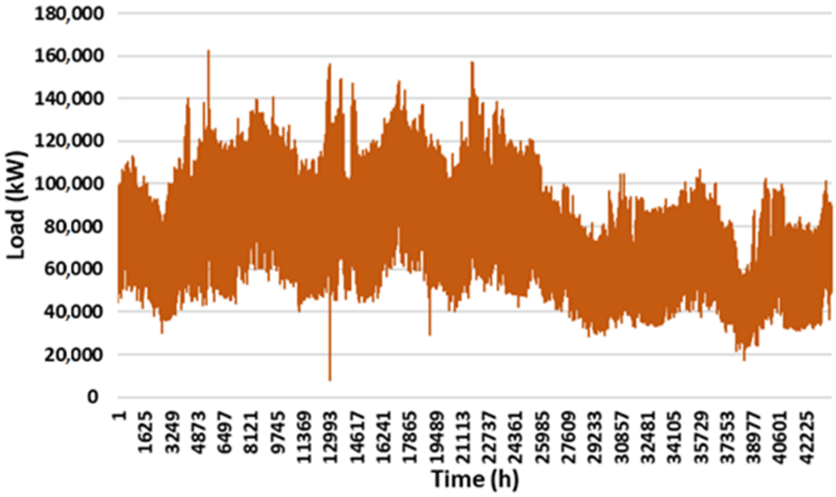

Figure 1. Load time series of the bus under study.

The first task was to examine the short-term periodicity of the load. The scope was to select the most correlated lagged load values. Let $L(h, d)$ be the load of the current hour h of day d. Utilizing the Pearson correlation index [40], the relationship between the current load value with its previous values up to 9 days in the past, i.e., h-216, were drawn. The corresponding curve is presented in Figure 2.

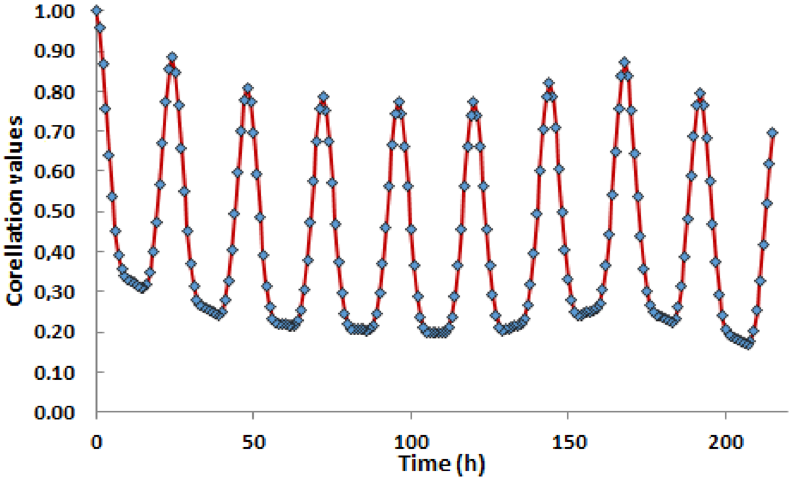

Figure 2. Correlation coefficient curve.

It can be noticed that the current hour load shows a high correlation with load values of the hours h-24 and h-168. Outdoor temperature was the most influential external variable of the demand. The inputs related to temperature contain statistical values (i.e., maximum and minimum) and temperature indices. The scope was to capture the chronological temperature variation; load variation was analogous to the temperature one. Also, day-type identification coding was needed to train the neural network to capture the difference in the load variation between working days, weekends, and holidays. The hot encoder was used for the types of days (i.e., Monday–Sunday). The seasonal effect was modeled through a pair of values obtained by periodic functions, i.e., sine and cosine.

Lastly, the holidays are denoted as "1" and the non-holidays as "0" to the neural network. Let $d,\ d-1$ and $d-7$ be the indicators that refer to the target day the day prior to it and the day one week before, respectively. The inputs of the FFNN are:

1. Inputs 1–24: hourly load of the day $d-1,\ L(h,d-1)$.
2. Inputs 25–48: hourly load of the day $d-7,\ L(h,d-7)$.
3. Inputs 49–50: maximum and minimum daily forecasted temperature of day $d,\ T_{\max}(d)$ and $T_{\min}(d)$.
4. Inputs 51–52: maximum and minimum daily temperature of day $d-1,\ T_{\max}(d-1)$ and $T_{\min}(d-1)$.
5. Input 53: the square value of the deviation of daily minimum and maximum temperature from the region of the cooling and heating threshold temperatures, $CT(d)$:

$$CT(d) = \left\{ \begin{array}{c} (T_{\max}(d) - T_{c,\min})^2,\ if\ T_{\max}(d) < T_{c,\min} \\ 0,\ if\ T_{c,\min} \le T_{\max}(d) \le T_{c,\max} \\ (T_{\max}(d) - T_{c,\max})^2,\ if\ T_{\max}(d) > T_{c,\max} \end{array} \right\} \tag{1}$$

where $T_{c,\min} = 17\ ^{\circ}\mathrm{C}$ and $T_{c,\min} = 25\ ^{\circ}\mathrm{C}$ are the cooling and heating threshold temperatures, respectively.
6. Input 54: same as input 54 for day $d-1,\ CT(d-1)$.
7. Input 55: the difference between the maximum daily temperatures of days d and $d-1,\ T_{\max}(d) - T_{\max}(d-1)$.
8. Inputs 56–57: the seasonality within the year of day d expressed as a pair of values $\{\sin(2\pi/d), \cos(2\pi/d)\}$.
9. Inputs 58–64: Encoded day-type indicator of the target day d, $DTI(d)$, i.e., 10000000 for Monday, 01000000 for Tuesday and so on.
10. Input 65: Holiday indicator, 1 for holidays and 0 for working days and weekends, $HI(d)$.

The vector that contains the inputs and outputs for the FFNN is referred to as a pattern. During training, the target outputs are known, i.e., the loads of the target day $d,\ L^{Train}(h,d)$. Let $y^{Train}(d)$ be the training vector for target day d of the training set. It has the following form:

$$\begin{array}{c} y^{Train}(d) = [L^{Train}(h,d-1), L^{Train}(h,d-7), T^{Train}_{\max}(d), T^{Train}_{\min}(d), T^{Train}_{\max}(d-1), T^{Train}_{\min}(d-1), CT^{Train}(d), \\ CT^{Train}(d-1), T^{Train}_{\max}(d) - T^{Train}_{\max}(d-1), \sin^{Train}(2\pi/d), \cos^{Train}(2\pi/d), DTI^{Train}(d), HI^{Train}(d),\ L^{Train}(h,d)]^T \end{array} \tag{2}$$

In (2) all the parameters with the *Train* indicator refer to the respective 65 inputs of the training set. The set of the training patterns is denoted as $Y^{Train} = \left\{ y^{Train}_{m^{Train}}, m^{Train} = 1, \ldots, M^{Train} \right\}$, where $m^{Train} = 1, \ldots, M^{Train}$ is the indicator denoting the number of days in the training set. In the test phase, the outputs are the loads that the model is utilized to predict, i.e., the target loads $L^{Test}(h,d)$ are not known. Accordingly, let $y^{Test}(d)$ be the training vector for target day d of the test set. It is given by:

$$\begin{array}{c} y^{Test}(d) = [L^{Test}(h,d-1), L^{Test}(h,d-7), T^{Test}_{\max}(d), T^{Test}_{\min}(d), T^{Test}_{\max}(d-1), T^{Test}_{\min}(d-1), CT^{Test}(d), \\ CT^{Test}(d-1), T^{Test}_{\max}(d) - T^{Test}_{\max}(d-1), \sin^{Test}(2\pi/d), \cos^{Test}(2\pi/d), DTI^{Test}(d), HI^{Test}(d)]^T \end{array} \tag{3}$$

where the *Test* indicator refer to the respective 65 inputs of the test set. The set of the test patterns is denoted as $Y^{Test} = \left\{ y^{Test}_{m^{Test}}, m^{Test} = 1, \ldots, M^{Test} \right\}$, where $m^{Test} = 1, \ldots, M^{Test}$ is the indicator denoting the number of days in the test set.

2.2. Input Selection Phase

Given a set of candidate inputs, the scope was to derive a reduced set of outputs in order to result in, ideally, both reductions in required training time and prediction error. A binary coded PSO was formed to indicate the optimal inputs. The flowchart of the input selection phase is presented in Figure 3. Let $L_a^{Train}\left(d^{Train},h\right)$ and $L_f^{Train}\left(d^{Train},h\right)$ be the actual and the forecasted load values of day d^{Train} of the training set, respectively, and let $L_a^{Test}\left(d^{Test},h\right)$ and $L_f^{Test}\left(d^{Test},h\right)$ be the actual and the forecasted load values of day d^{Test} of the test set, respectively. The normalized mean square errors (NMSEs) of the training and test sets are given by the following equations, respectively:

$$\text{NRMSE}^{Train} = \frac{1}{M^{Train}}\frac{1}{H}\sum_{d^{Train}=1}^{D^{Train}}\sum_{h=1}^{H}\frac{\left(L_a^{Train}\left(d^{Train},h\right)-L_f^{Train}\left(d^{Train},h\right)\right)^2}{\overline{L_a^{Train}}\;\overline{L_f^{Train}}} \tag{4}$$

$$\text{NRMSE}^{Test} = \frac{1}{M^{Test}}\frac{1}{H}\sum_{d^{Test}=1}^{D^{Test}}\sum_{h=1}^{H}\frac{\left(L_a^{Test}\left(d^{Test},h\right)-L_f^{Test}\left(d^{Test},h\right)\right)^2}{\overline{L_a^{Test}}\;\overline{L_f^{Test}}} \tag{5}$$

where M^{Train} and M^{Test} are the total number of days in the training and test sets, respectively, H is the number of hours, and $\overline{L_a^{Train}}, \overline{L_f^{Train}}, \overline{L_a^{Test}}$ and $\overline{L_f^{Test}}$ are the mean values of $L_a^{Train}\left(d^{Train},h\right)$, $L_f^{Train}\left(d^{Train},h\right)$, $L_a^{Test}\left(d^{Test},h\right)$ and $L_f^{Test}\left(d^{Test},h\right)$, respectively. The NMSE is an estimator of the overall deviations between predicted and measured values. The decision variables of the optimization project are those inputs of the FFNN that minimize the following objective/fitness function:

$$J(W) = W_{km}^{Train} \times \text{NRMSE}^{Train} + W_{km}^{Test} \times \text{NRMSE}^{Test} \tag{6}$$

The weight matrices of the FFNN of the training and the test phases, respectively:

$$W^{Train} = \left\{w_{kn}^{Train}, k = 2,4,\ldots,30, n = 1,2,\ldots,24\right\} \tag{7}$$

$$W^{Test} = \left\{w_{kn}^{Test}, k = 2,4,\ldots,30, n = 1,2,\ldots,24\right\} \tag{8}$$

where k denotes the k-th neuron in the hidden layer and n denotes the n-th neuron in the output layer. The selection of the number of neurons in the hidden layer was a subject of a sensitivity analysis. In the present study, the number of neurons in the hidden layer varied between 2 and 30, with an increasing step equal to 2. The decision variables of the optimization problem were the weight matrices during training and testing. The weight values indirectly refer to the neurons that would be selected or not to form the reduced set of inputs. Figure 3 illustrates the operation of the input selection process. Regarding Figure 3, x_i is the current position of the particle i, v_i is the current velocity of the particle i, $Xbest_i$ is the personal best solution of the particle i, t is the current iteration of the algorithm and $t_{\max}$ is the pre-defined maximum number of iterations. The PSO minimizes (5) via an iterative process. The number of desired outputs is set by the user in advance. The swarm's size equals the desired number of inputs. The output of the algorithm is a sequence of integers denoting the number of the selected inputs. Figure 3 presents the flow-chart of the input selection phase. It should be noted that the training phase weights update require more time compared with the weights update of the test phase. This relation between the two durations are asymmetrical, i.e., the durations are not proportional.

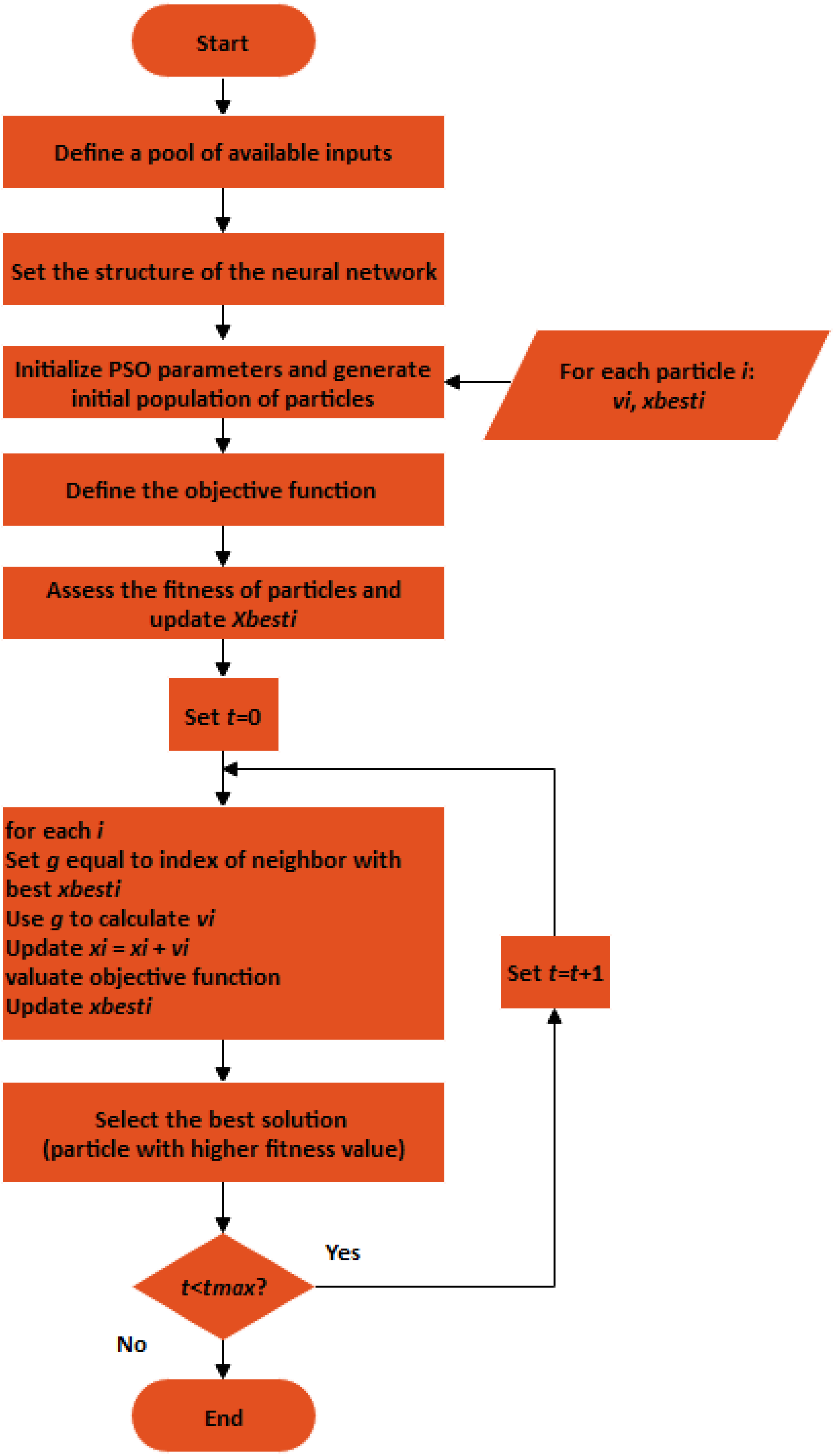
Start
Define a pool of available inputs
Set the structure of the neural network
Initialize PSO parameters and generate initial population of particles
For each particle *i*: *vi, xbesti*
Define the objective function
Assess the fitness of particles and update *Xbesti*
Set *t*=0
for each *i*
Set *g* equal to index of neighbor with best *xbesti*
Use *g* to calculate *vi*
Update *xi* = *xi* + *vi*
valuate objective function
Update *xbesti*
Set *t*=*t*+1
Select the best solution (particle with higher fitness value)
t<*tmax*?
Yes
No
End

Figure 3. Flowchart of the input selection phase.

2.3. Training Set Formation Phase

The scope was to derive a reduced set of outputs to result in, ideally, both reductions in required training time and prediction error. A binary coded PSO was formed to indicate the optimal inputs. After the selection of the inputs, the K-medoids/PSO algorithm was applied to split the training set into smaller subsets. The flowchart of the clustering phase is illustrated in Figure 4. PSO is a metaheuristics algorithm that mimics the collective behavior of swarms in search of their food [41]. The term "food" in optimization terms models the solution of the problem. K-medoids is a partitional clustering algorithm that has a similar operation to the K-means; the difference lies in the centers of gravity of the clusters. K-medoids employs the medoids of the clusters, i.e., the patterns whose average dissimilarity to all the patterns in the same cluster is minimal [42]. K-medoids and PSO algorithms are both influenced by their initialization phase. The selection of the load curves that will serve as initial centroids is crucial since they form the initial clusters and define the subsequent clustering procedure.

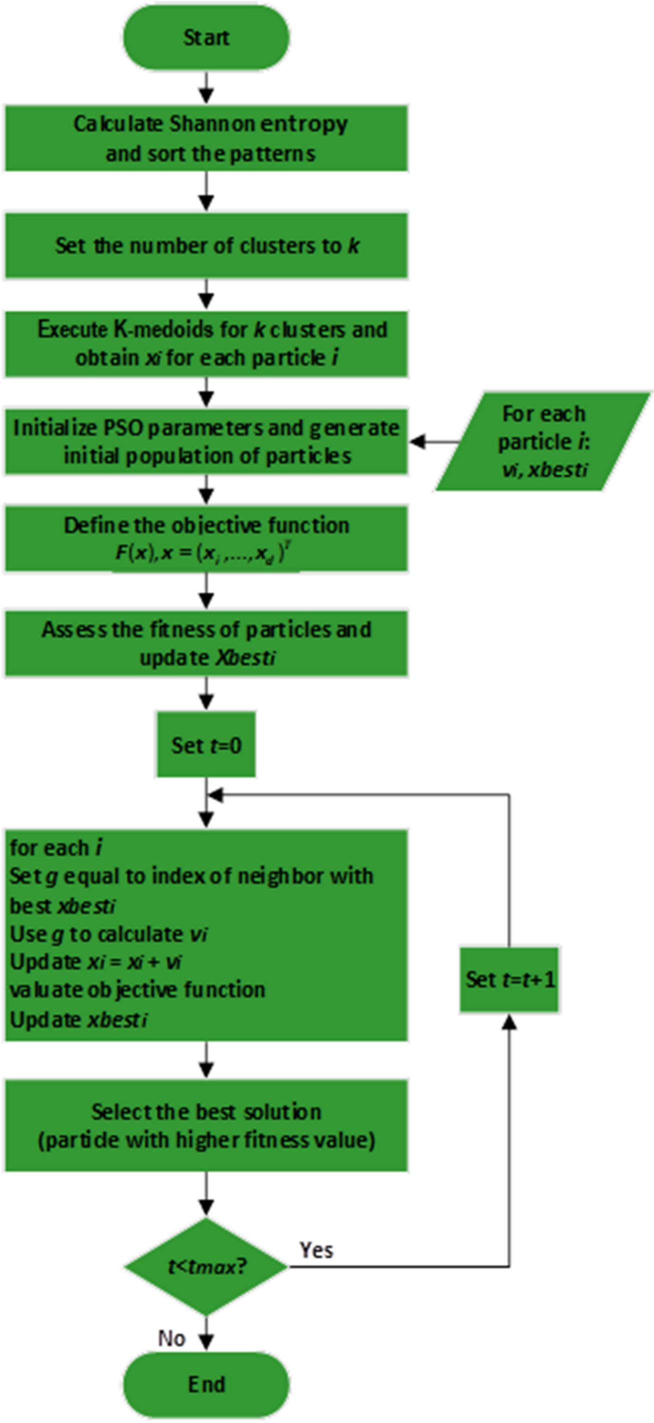
Start
Calculate Shannon entropy and sort the patterns
Set the number of clusters to *k*
Execute K-medoids for *k* clusters and obtain *xi* for each particle *i*
Initialize PSO parameters and generate initial population of particles
For each particle *i*: *vi*, *xbesti*
Define the objective function $F(x), x = (x_i, \ldots, x_d)^T$
Assess the fitness of particles and update *Xbesti*
Set *t*=0
for each *i*
Set *g* equal to index of neighbor with best *xbesti*
Use *g* to calculate *vi*
Update *xi* = *xi* + *vi*
valuate objective function
Update *xbesti*
Set *t*=*t*+1
Select the best solution (particle with higher fitness value)
t<*tmax*?
Yes
No
End

Figure 4. Flowchart of the clustering phase.

The default forms of the K-medoids and PSO are based on a random selection. Different executions of the algorithms lead to different clustering outcomes. Thus, it is not always guaranteed that the algorithm will reach the optimal solution every time, i.e., the algorithm will converge in local optima. To deal with this situation, a new approach was proposed. For each daily load curve, the Shannon entropy was calculated. The aim was to initialize K-medoids with centroids that are highly dissimilar to one another so that a better partitioning of the load data would be manifested. Shannon entropy is a measure of the randomness of the data. In time series modeling, it is a measure of the volatilities of the curves. Thus, load curves that present large differences in the Shannon entropy values correspond to highly dissimilar shapes and can serve as different centroids. Let j be a value within a time series and p_j the probability that this value will appear in the series. The Shannon entropy H is expressed as [43]:

$$H = -\sum_{J} p_j \log p_j \tag{9}$$

Low entropy values refer to less volatile series. A time series with high entropy contains hourly load values that are repeated, i.e., they are present in more than one instance within the period that a time series refers to. Therefore, the volatility is relatively low.

As an example of load curves with different entropy values, Figure 5 shows three daily load curves of the test set. The load curve with $H = 4.58$ displays high volatility in comparison with the rest. The load curves of Figure 5 can serve as the initial centroids for the K-medoids if the clustering was applied only to the load data set. The K-medoids/PSO is applied in the set Y^{Train}. The outputs of clustering are the cluster labels, i.e., integer values that are assigned to each training pattern. Let $C^{Train} = \left\{ c^{Train}_{k^{Train}}, k^{Train} = 1, \ldots, K^{Train} \right\}$ be the centroids of the training set, where $k^{Train} = 1, \ldots, K^{Train}$ is the indicator denoting the number of clusters in the training set. The fitness function of the PSO J^{Train} refers to the sum of squared errors between patterns $y^{Train}_{m^{Train}}$ and centroids $c^{Train}_{k^{Train}}$:

$$J^{Train} = \sum_{\substack{m^{Train} \\ y^{Train}_{m^{Train}} \in S^{Train}}}^{M^{Train}} d^2_{Eucl}\left(y^{Train}_{m^{Train}}, c^{Train}_{k^{Train}} \right) \tag{10}$$

where S^{Train} is the subset of C^{Train} that includes the population of the k-th cluster and d^2_{Eucl} is the square of the Euclidean distance.

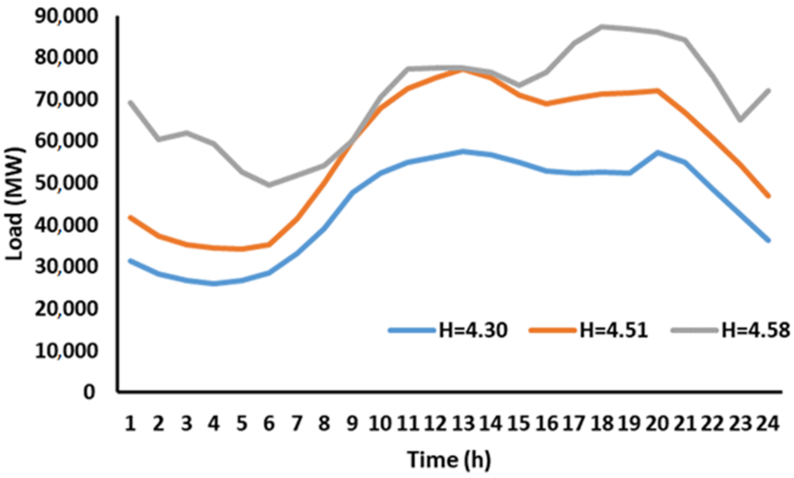

Figure 5. Example of three load curves with different entropy values.

2.4. Training Phase

The clustering split the initial training set into training subsets/clusters. Then, for each cluster, separate FFNNs were trained. After a series of experimentations, one hidden layer had been selected for the FFNN. The training algorithm was the Levenberg–Marquardt [44]. For both the neurons of the hidden and the output layers, the hyperbolic tangent sigmoid activation function had been chosen [45]. The maximum number of iterations for training was set to 500. A training epoch or iteration refers to a full cycle of presenting all patterns to the FFNN. The training process ended either when the FFNN reaches the predefined number of epochs or when the improvement of the training error between two subsequent iterations was below a predefined threshold. The latter condition corresponded to the convergence of the training process.

2.5. Test Phase

After training, the actual application of forecasting on the target set was held. In order to select the appropriate neural network for a specific test pattern, the following procedure was followed: The last 24 values of the centroids of the clusters were removed. Next, using the Euclidean distance metric, the distances among the patterns of the set Y^{Test} and the centroids of the set C^{Train} were calculated. For each pattern, the selected neural network corresponded to the cluster where its centroid had the closest distance to the pattern. The removal of the last 24 values was done so that the calculation of the Euclidean distance would be feasible, i.e., the vectors (test patterns and centroids) should be of equal length. The number of neural networks is equal to the number of clusters. Thus, the test set pattern was fed to the specific FFNN that had been trained with similar patterns to the test set pattern.

2.6. Summary

The proposed method included two distinct phases, namely the input selection and the formation of the training subsets. In summary, it consisted of the following steps:

Step#1: Formation of the initial data set. The formation can be taken into account: (i) Expertise knowledge, (ii) literature, and (iii) expertise knowledge and literature. Employing expert knowledge refers to a problem-specific approach. The initial data set is held via trial and error. In the present paper, the literature approach was followed. The inputs that have been proposed in the previous study were regarded [37]. The specific study refers to the aggregated system of Greece. In the present paper, the scope was to regard the same inputs with the model that had been proposed for the aggregated system to a bus load forecasting problem.

Step#2: Execution of the PSO. Based on a pre-defined desired number of reduced inputs, the PSO was applied to an initial neural network training. The optimal inputs were drawn based on the minimization of the error function (Equation (6)). If the results were not satisfactory, the PSO was re-executed by considering different parameters.

Step#3: Formation of training set clusters with the K-medoids/PSO algorithm. Both the conventional forms of the K-medoids and PSO refer to random initialization of initial populations. This fact may lead to poor clustering results. To overcome the K-medoids limitation, a new approach was proposed to select the initial centroids based on their randomness as quantified by the Shannon entropy index. The initial centroids should differ in terms of randomness as much as possible. Next, the K-medoids were executed and provided an initial clustering. The obtained centroids were used as the initial ones for the PSO. The latter provided the final centroids. It should be noted that the patterns that were used for clustering represent the vectors that contain the reduced set of inputs.

Step#4: Neural network training. For each cluster, a separate neural network was trained.

Step#5: Neural network application. A specific neural network was selected for each target day of the test day. The STLF is a day-ahead forecasting process. The aim was to forecast the daily load of a complete year.

3. Results

3.1. Input Selection

The input selection phase allows the user to enter the desired number of inputs. As two test cases, two different numbers of inputs were set, namely 42 and 48. Recall that the initial number was 65. 45s 1 and 2 present the selected inputs for numbers 48 and 42, respectively. The scope is to compare the performance of the FFNN of the original 65 inputs with the ones that use a reduced set.

The comparison refers to the forecasting accuracy and the required training time. From Tables 1 and 2 it can be observed that inputs from all three types have been selected. Most inputs correspond to historical loads, i.e., inputs 1–48 of the original set. The input selection process does not include all the inputs related to temperature and day-type indicators. In both cases, the parameters $T_{\max}(d)$ and $T_{\min}(d-1)$ are selected by the PSO. In addition, the deviations from the thermal comfort are selected, i.e., $CT(d)$ and $CT(d-1)$. These two parameters express the temperature ranges from two thresholds that correspond to high electricity consumption, i.e., low and high temperatures. Also, among the two periodic trigonometric functions only $\sin(2\pi d/365)$ is selected, leading to the conclusion that one trigonometric function is enough to express the periodicity of the day within the year. Regarding the day-type indicator, five digits are selected instead of the original seven. The holidays indicator, i.e., input 65 is considered significant and it is selected in both cases.

Table 1. Selected 42 inputs.

Input	Variable	Input	Variable
2	$L(2,d-1)$	37	$L(13,d-7)$
3	$L(3,d-1)$	38	$L(14,d-7)$
5	$L(5,d-1)$	40	$L(16,d-7)$
6	$L(7,d-1)$	41	$L(17,d-7)$
9	$L(9,d-1)$	42	$L(18,d-7)$
12	$L(12,d-1)$	44	$L(20,d-7)$
13	$L(13,d-1)$	46	$L(22,d-7)$
15	$L(15,d-1)$	47	$L(23,d-7)$
16	$L(16,d-1)$	48	$L(24,d-7)$
18	$L(18,d-1)$	49	$T_{\max}(d)$
19	$L(19,d-1)$	52	$T_{\min}(d-1)$
20	$L(20,d-1)$	53	$CT(d)$
23	$L(23,d-1)$	54	$CT(d-1)$
24	$L(24,d-1)$	55	$T_{\max}(d)-T_{\max}(d-1)$
26	$L(2,d-7)$	56	$\sin(2\pi d/365)$
27	$L(3,d-7)$	59	day-type indicator
29	$L(5,d-7)$	61	day-type indicator
30	$L(6,d-7)$	62	day-type indicator
31	$L(7,d-7)$	63	day-type indicator
33	$L(9,d-7)$	64	day-type indicator
36	$L(12,d-7)$	65	$HI(d)$

Table 2. Selected 48 inputs.

Input	Variable	Input	Variable
1	$L(1,d-1)$	33	$L(9,d-7)$
2	$L(2,d-1)$	35	$L(11,d-7)$
3	$L(3,d-1)$	37	$L(13,d-7)$
5	$L(5,d-1)$	38	$L(14,d-7)$
6	$L(6,d-1)$	39	$L(15,d-7)$
7	$L(7,d-1)$	40	$L(16,d-7)$
8	$L(8,d-1)$	43	$L(19,d-7)$
9	$L(9,d-1)$	44	$L(20,d-7)$

Table 2. *Cont.*

Input	Variable	Input	Variable
10	$L(10, d-1)$	46	$L(22, d-7)$
11	$L(11, d-1)$	47	$L(23, d-7)$
12	$L(12, d-1)$	48	$L(24, d-7)$
14	$L(14, d-1)$	49	$T_{\max}(d)$
15	$L(15, d-1)$	51	$T_{\max}(d-1)$
17	$L(17, d-1)$	52	$T_{\min}(d-1)$
18	$L(18, d-1)$	53	$CT(d)$
19	$L(19, d-1)$	54	$CT(d-1)$
21	$L(21, d-1)$	55	$T_{\max}(d) - T_{\max}(d-1)$
22	$L(22, d-1)$	56	$\sin(2\pi d/365)$
23	$L(23, d-1)$	58	day-type indicator
24	$L(24, d-1)$	59	day-type indicator

3.2. Clustering

To verify the superiority of the K-medoids/PSO algorithm over other clustering algorithms, a load profiling experiment was formulated, i.e., the clustering of the daily load curves of the test set. The proposed algorithm was compared with the K-means [32], FCM [30,35], SOM [28,29,33,34] and UPGMA [31]. The scope is to partition the load data set of the year 2018 into a number of clusters and extract the load profiles. A daily load curve is expressed as a 24-element vector:

$$x^{(m)} = \left[x_1^{(m)}, \ldots, x_{24}^{(m)}\right]^T \tag{11}$$

The load set of data is denoted as $X = \left\{x^{(p)},\ p = 1, \ldots, 365\right\}$ where $p = 1, 2, \ldots, 365$ denotes the day of the year. The elements of the vector refer to hourly load values normalized in [0, 1] range. This is necessary for clustering since it focuses on the similarities of the load curve shapes and not on the load magnitude. The base value for the normalization is the maximum value of the set X, i.e., the peak load of the bus. The centroid of a cluster is the average of all load curves that belong to it:

$$c^{(k)} = \left\{c_j^{(k)}, \ldots, c_D^{(k)}\right\} = \frac{1}{M_K} \sum\nolimits^{M}_{\substack{m\,=\,1\\ x^{(m)}\,\in\, C_k}} x^{(m)} \tag{12}$$

where M_K is the number of daily load curves that belong to the cluster C_K. The set of the clusters is denoted as $C = \left\{c^{(k)}, k = 1, \ldots, K\right\}$. The centroids are actually artificial curves and not real measured values; they are statistical parameters and more precisely, the averages of clusters. Validity indicators are needed to assess the clustering errors. These indicators are based on Euclidean distance-based metrics that measure the degree of similarity of patterns in the same clusters, centroids, and patterns of the same cluster and centroids of different clusters. The following metrics are defined [46]:

(i) The Euclidean distance between two vectors $x^{(s)}$ and $x^{(t)}$, with $\left(x^{(s)}, x^{(t)}\right) \in X$, is:

$$d\left(x^{(s)}, x^{(t)}\right) = \sqrt{\frac{1}{D} \sum\nolimits_{h=1}^{D} \left(x_h^{(s)} - x_h^{(t)}\right)^2} \tag{13}$$

(ii) The subset of X that belong to the C_K cluster is denoted as S_K. The Euclidean distance between the centroid $c^{(k)}$ of the k-th cluster and the subset S_K is the geometric mean of the Euclidean distances $d\left(c^{(k)}, S_k\right)$ between $c^{(k)}$ and each member $x^{(k)}$ of S_K:

$$d\left(c^{(k)}, S_k\right) = \sqrt{\frac{\sum_{i=1}^{M_k} x^{(k)} \in S_k d^2\left(c^{(k)}, x^{(k)}\right)}{M_k}} \tag{14}$$

(iii) The geometric mean of the inner distances between the featured members of the subset S_K is:

$$d(S_k) = \sqrt{\frac{1}{2M_k} \sum x^{(k)} \in S_k d^2\left(x^{(k)}, x^{(m)}\right)} \tag{15}$$

The clustering dispersion indicator (CDI), which is the ratio of the mean infra-set distance between the input vectors in the same cluster and the infra-set distance between the cluster's centroids:

$$\mathrm{CDI} = \frac{\sqrt{\frac{1}{K}\sum_{k=1}^{K} d^2(S_k)}}{\sqrt{\frac{1}{2K}\sum_{k=1}^{K} d^2(c_k, C_k)}} \tag{16}$$

The Calinski–Harabasz index (CH) or minimum variance criterion (VRC), which refers to the ratio of the dispersion among the different clusters and the dispersion within the same cluster:

$$\mathrm{CH} = \frac{M-K}{K-1} \cdot \frac{\sum_{k=1}^{K} M_k \cdot \left(c^{(k)} - p\right) \cdot \left(c^{(k)} - p\right)^t}{\sum_{m.k=1}^{K} \sum_{x^{(m)} \in X} \left(x^{(k)} - c^{(k)}\right) \cdot \left(x^{(k)} - c^{(k)}\right)^t} \tag{17}$$

where p is the arithmetic mean of the input vectors. The nominator in (17) refers to the dispersion within the same cluster whereas the denominator refers to the dispersion between the clusters. The Davies–Bouldin index (DBI) is expressed as the ratio of the sum of the most similar clusters to the distance of their centroids:

$$\mathrm{DBI} = \frac{1}{K} \sum_{s,t=1}^{K} \max_{s \neq t} \left\{ \frac{d(S_s) + d(S_t)}{d\left(c^{(s)}, c^{(t)}\right)} \right\} \tag{18}$$

where centroids $c^{(s)}, c^{(t)} \in C$.

Let a_k be the be the mean distance between the pattern $x^{(k)}$ that belongs to the k-th cluster C_k and all other patterns that belong to C_k :

$$a_k = \frac{1}{M_k - 1} \sum_{x^{(l)} \in C_k, x^{(k)} \neq x^{(l)}} d\left(x^{(k)}, x^{(l)}\right) \tag{19}$$

where $x^{(k)} \in C_k$. Let b_k be the mean dissimilarity between $x^{(k)}$ and a cluster C_m, $C_k \neq C_m$. It refers to the mean distance between $x^{(k)}$ and all members of C_m :

$$b_k = \min_{x^{(m)} \neq x^{(k)}} \frac{1}{M_m} \sum_{x^{(m)} \in C_m} d\left(x^{(m)}, x^{(k)}\right) \tag{20}$$

where $x^{(m)} \in C_m$ and M_m is the number of patterns that belong to the cluster C_m. The silhouette index (SI) of pattern $x^{(k)}$ is defined as [47]:

$$S^k = \frac{b_k - a_k}{\max\{a_k, b_k\}} \tag{21}$$

It can be rewritten as:

$$S^k = \left| \begin{array}{ll} 1 - \frac{a_k}{b_k}, & a_k > b_k \\ 0, & a_k = b_k \\ \frac{b_k}{a_k} - 1, & a_k < b_k \end{array} \right| \tag{22}$$

A low value of a_k corresponds to a low degree of similarity of the patterns within the same cluster. A high value of b_k corresponds to a high degree of similarity between the pattern $x^{(k)}$ and patterns of other clusters. The overall SI of all clusters is defined as [47]:

$$SI = \frac{1}{K} \sum_{k=1}^{K} S^k \tag{23}$$

There is no a priori information about the number of clusters of the specific data set or other relevant information on data structure such as the number of consumers that are connected to the bus, number of atypical patterns, special conditions, or others. Hence, the present load profiling problem was formulated as a purely unsupervised machine learning task. For this reason, each algorithm was executed separately for 2 to 30 clusters. The scope was to group together daily load curves with similar shapes, i.e., similar daily consumption. For each number of clusters, the values of the validity indicators were checked. The superiority of one algorithm over the other is demonstrated when leading to better performance in all indicators. Figure 6 shows the comparisons of the algorithms per validity indicator. Two crucial attributes of clustering quality are the well-separation and compactness. The well-separation refers to the distances of the clusters' centers, i.e., the centroids. High distances between centroids refer to clusters that are well-separated in the feature space, i.e., the clustering algorithm managed to recognize the dissimilarities between the patterns and distribute them correctly. Compactness is measured either by intra-distances between patterns in the same cluster or by the distances between patterns and the centroid of the cluster they belong to. Low values of a metric that measures compactness refer to clusters that are composed of highly similar patterns.

The CDI is a ratio of a well-separation metric to a compactness one. While the number of clusters increases, robust clustering should lead to lower values of the CDI. While the number of clusters is increasing, the clusters become more compact, i.e., they contain load curves with more similar characteristics. In Figure 6, the PSO/K-medoids results in lower CDI values followed by the Ward hierarchical algorithm. The last in the competition is the FCM. The CH indicator is a measure of compactness. Since the denominator needs to be minimized, the better clustering corresponds to large CH values. Again, the PSO/K-medoids outperforms all the rest. As for the second place in the competitions, there are mixed results; K-means appears more suitable for a low number of clusters and the K-medoids for a medium-range number. For a large number of clusters until 30, SOM is proposed. As in the case of the CDI, fuzzy clustering presents poor performance. Like the CDI, the DBI measures both well-separation and compactness. The best algorithm should lead to lower values. The PSO/K-medoids leads to lower values in mist number of clusters followed by the Ward algorithm. Finally, the SI indicates the proposed clustering algorithm as the winner of the competition. The SOM is the second in the list whereas the FCM fails to provide robust clustering.

Considering the CDI curve as drawn by the PSO/K-medoids algorithm, the optimal number of clusters can be derived using the knee point detection method [48]. The optimal number of clusters is seven. Thus, the 365 daily load curves are optimally grouped together into seven groups. Figure 7 presents the load profiles of the clusters. The day-type distribution per cluster is registered in Table 3.

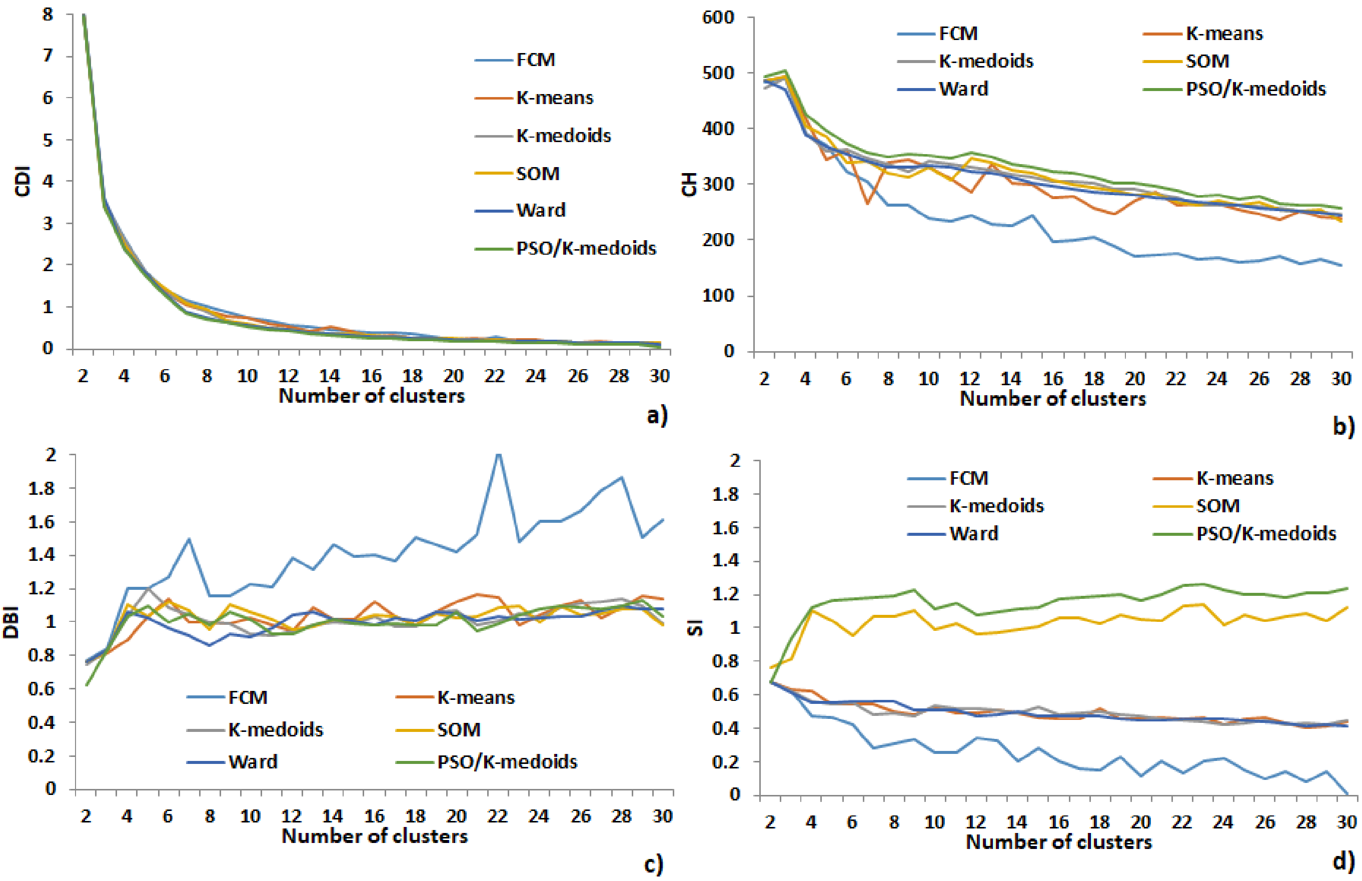

Figure 6. Clustering algorithms comparison using: (**a**) CDI, (**b**) CH, (**c**) DBI, (**d**) SI.

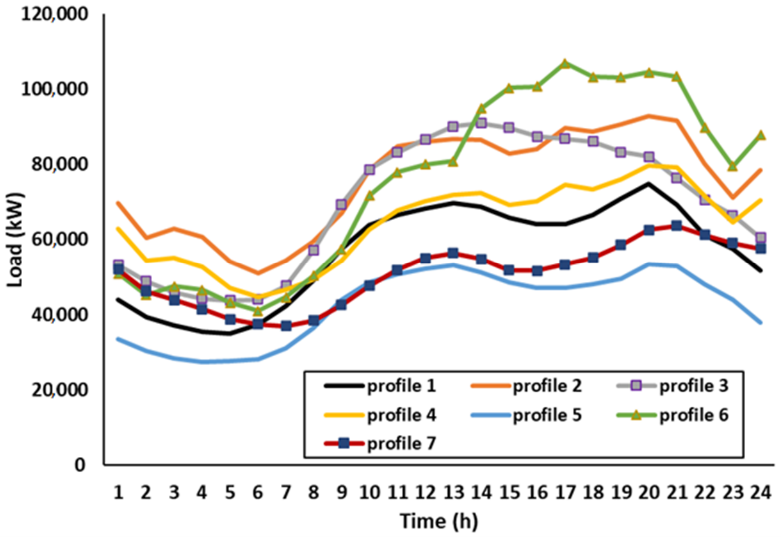

Figure 7. Load profiles of the clusters.

There are different shapes and magnitudes between the profiles. The most populated cluster is cluster#1 which contains days from all types apart from Sundays. It corresponds to a relatively medium consumption compared with the rest of the profiles. Also, there is an almost equal number of different day types. Cluster#2 is the second largest cluster. Again, there is an almost equal number of working days. It contains six Saturdays and two Sundays. Cluster#6 contains one daily load curve. This includes both the peak demand among the profiles and the maximum consumed energy, i.e., the largest area under the load curve in the power versus time axes. Cluster#7 refers to low consumption and contains the most Sundays of the year.

Table 3. Day-type distribution per cluster.

Cluster	Day							Number of Days per Cluster
	Mon	Tue	Wed	Thu	Fri	Sat	Sun	
1	18	18	20	21	23	23	0	123
2	11	11	10	12	12	6	2	64
3	9	9	8	8	7	3	0	44
4	6	6	6	3	4	10	9	44
5	8	7	7	6	5	7	14	54
6	0	0	1	0	0	0	0	1
7	0	1	0	2	2	3	27	35
			Total number of days					365

3.3. Forecasting

The proposed model with an FFNN. The maximum number of training iterations of the FFNN is set to 500. The initial velocities of the particles are randomly given. The evaluation of the accuracy is done with the mean absolute percentage error (MAPE):

$$\text{MAPE} = \frac{1}{D}\sum_{d=1}^{D} \frac{\left|L_a(d) - L_f(d)\right|}{L_a(d)} \times 100 \tag{24}$$

where D is the total number of days and $L_a(d)$ and $L_f(d)$ are the actual and forecasted loads of day D, respectively. The FFNN is trained with a variable number of neurons in the hidden layer and for each number the MAPE is checked. The comparison of the two models is illustrated in Figure 8. The figure refers to the MAPE of the test set. The test set is used for the model's predictability assessment. For the greatest number of neurons, the reduced set of inputs leads to lower errors. The ANN with the 65 inputs results in the lowest MAPE = 5.77% at six neurons, whereas the one with the 48 inputs results in the lowest MAPE = 5.64% at 12 inputs. The average MAPE of the 65 inputs is 11.84% whereas that of the 48 inputs is 11.12%.

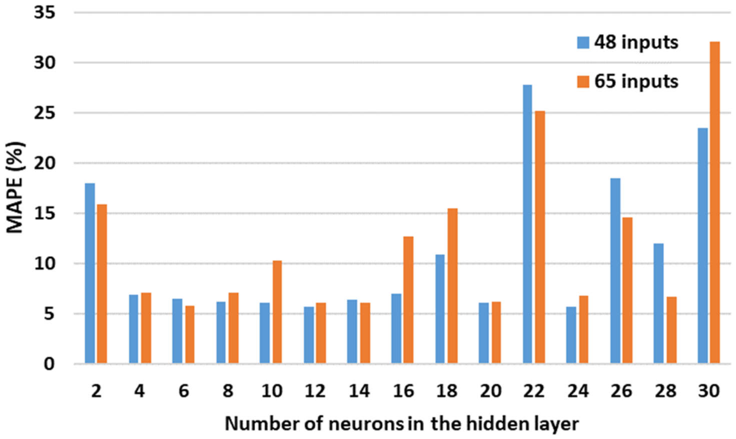

Figure 8. Comparison of the two ANNs for the case of 48 inputs.

The second case refers to a further reduction of the inputs to 42. Figure 9 presents the comparison. The lowest MAPE = 5.63% at 12 neurons, whereas the average MAPE = 12.03%. Therefore, the case with 40 inputs leads to the lowest error. In all cases, the lowest MAPE is not met in the same number of neurons.

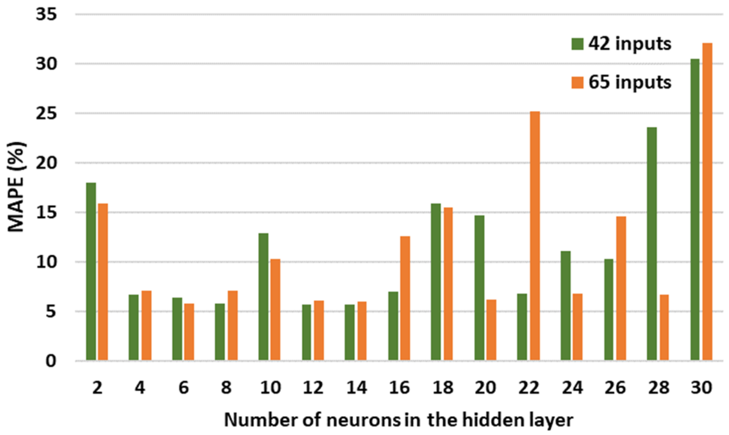

Figure 9. Comparison of the two ANNs for the case of 42 inputs.

4. Conclusions

The forecasting of future demand is an important process in power systems operation and planning in order to ensure a reliable and secure network. Among forecasting horizons, STLF is the basis of intra-day and day-ahead power system operations. Apart from power generation companies and system operators, STLF has also become a valuable tool for retailers and prosumers, leading to the conclusion that STLF supports the decision-making actions of the various competitive market entities.

Among the models that have been proposed in the literature, neural networks are a favorable approach. However, the main issue of the model set-up and application is the proper selection of inputs that will influence forecasting accuracy.

The present paper proposes a novel approach to neural network application for load forecasting. A metaheuristics-type algorithm is used both for optimal input selection and optimal determination of the weights of the neural network. This double operation has not been proposed so far in the literature. Also, the metaheuristics-type algorithm is a new type; it combines two algorithms, a clustering algorithm and an optimization one.

Author Contributions: Conceptualization, I.P.; methodology, I.P.; software, I.P.; validation, I.P.; formal analysis, I.P. and M.K.; investigation, I.P.; resources, M.K.; data curation, I.P. and M.K.; writing—original draft preparation, I.P.; writing—review and editing, I.P. and M.K.; visualization, M.K.; supervision, D.B.; project administration, D.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Ahmed, A.; Khalid, M. A review on the selected applications of forecasting models in renewable power systems. *Renew. Sustain. Energy Rev.* **2019**, *100*, 9–21. [CrossRef]
2. Saksornchaim, T.; Lee, W.J.; Methaprayoon, K.; Liao, J.R.; Ross, R.J. Improve the unit commitment scheduling by using the neural-network-based short-term load forecasting. *IEEE Trans. Ind Appl.* **2005**, *41*, 169–179. [CrossRef]
3. Jarndal, A. Load forecasting for power system planning using a genetic-fuzzy-neural networks approach. In Proceedings of the 2013 7th IEEE GCC Conference and Exhibition, Doha, Qatar, 17–20 November 2013; pp. 44–48.
4. Soliman, S.A.; Al-Kandari, A.M. *Electrical Load Forecasting: Modeling and Model Construction*; Butterworth-Heinemann: Burlington, MA, USA; Oxford, UK, 2010.

5. Hernandez, L.; Baladron, C.; Aguiar, J.M.; Carro, B.; Sanchez-Esguevillas, A.J.; Lloret, J.; Massana, J. A survey on electric power demand forecasting: Future trends in smart grids, microgrids and smart buildings. *IEEE Commun. Surv. Tutorials* **2014**, *16*, 1460–1495. [CrossRef]
6. Ilic, D.; Karnouskos, S.; Goncalves Da Silva, P. Improving load forecast in prosumer clusters by varying energy storage size. In Proceedings of the IEEE Grenoble PowerTech 2013, Grenoble, France, 16–20 June 2013; pp. 1–6.
7. Danti, P.; Magnani, S. Effects of the load forecasts mismatch on the optimized schedule of a real small-size smart prosumer. *Energy Proc.* **2017**, *126*, 406–413. [CrossRef]
8. Hong, T.; Fan, S. Probabilistic electric load forecasting: A tutorial review. *Int. J. Forecast.* **2016**, *32*, 914–938. [CrossRef]
9. Hahn, H.; Meyer-Nieberg, S.; Pickl, S. Electric load forecasting methods: Tools for decision making. *Eur. J. Oper. Res.* **2009**, *199*, 902–907. [CrossRef]
10. Huang, S.J.; Shih, K. Short-term load forecasting via ARMA model identification including non-Gaussian process considerations. *IEEE Trans. Power Syst.* **2009**, *18*, 673–679. [CrossRef]
11. Wei, L.; Zhen-Gang, Z. Based on time sequence of ARIMA model in the application of short-term electricity load forecasting. In Proceedings of the 2009 International Conference on Research Challenges in Computer Science, Shanghai, China, 28–29 December 2009; pp. 11–14.
12. Khuntia, S.R.; Rueda, J.L.; van der Meijden, M.A.M.M. Volatility in electrical load forecasting for long-term horizon—An ARIMA-GARCH approach. In Proceedings of the 2016 International Conference on Probabilistic Methods Applied to Power Systems, Beijing, China, 16–20 October 2016; pp. 1–6.
13. Charytoniuk, W.; Box, E.D.; Lee, W.J.; Chen, M.S.; Kotas, P.; Van Olinda, P. Neural-network-based demand forecasting in a deregulated environment. *IEEE Trans. Ind. Appl.* **2000**, *36*, 893–898. [CrossRef]
14. Yang, X. Comparison of the LS-SVM based load forecasting models. In Proceedings of the 2011 International Conference on Electronic & Mechanical Engineering and Information Technology, Harbin, China, 12–14 August 2011; pp. 2942–2945.
15. Xu, F.Y.; Leung, M.C.; Zhou, L. A RBF network for short-term load forecast on microgrid. In Proceedings of the 2010 International Conference on Machine Learning and Cybernetics, Qingdao, China, 11–14 July 2010; pp. 1–3.
16. Karthika, S.; Margaret, V.; Balaraman, K. Hybrid short term load forecasting using ARIMA-SVM. In Proceedings of the 2017 Innovations in Power and Advanced Computing Technologies, Vellore, India, 21–22 April 2017; pp. 1–7.
17. Zhang, J.; Wei, Y.M.; Li, D.; Tan, Z.; Zhou, J. Short term electricity load forecasting using a hybrid model. *Energy* **2018**, *158*, 774–781. [CrossRef]
18. Wang, J.; Wang, J.; Li, Y.; Zhu, S.; Zhao, J. Techniques of applying wavelet de-noising into a combined model for short-term load forecasting. *Int. J. Electr. Power Energy Syst.* **2014**, *62*, 816–824. [CrossRef]
19. 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]
20. Abiodun, O.I.; Jantan, A.; Omolara, A.E.; Dada, K.V.; Mohamed, N.A.; Arshad, H. State-of-the-art in artificial neural network applications: A survey. *Heliyon* **2018**, *4*, e00938. [CrossRef] [PubMed]
21. Fallah, S.N.; Ganjkhani, M.; Shamshirband, S.; Chau, K.W. Computational intelligence on short-term load forecasting: A methodological overview. *Energies* **2019**, *12*, 393. [CrossRef]
22. Liang, Y.; Niu, D.; Hong, W.C. Short term load forecasting based on feature extraction and improved general regression neural network model. *Energy* **2019**, *166*, 653–663. [CrossRef]
23. Ghadim, N.; Akbarimajd, A.; Shayeghi, H.; Abedinia, O. Two stage forecast engine with feature selection technique and improved meta-heuristic algorithm for electricity load forecasting. *Energy* **2018**, *161*, 130–142. [CrossRef]
24. 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]
25. Hu, Z.; Bao, Y.; Xiong, T. Comprehensive learning particle swarm optimization based memetic algorithm for model selection in short-term load forecasting using support vector regression. *Appl. Soft Comput.* **2014**, *25*, 15–25. [CrossRef]
26. Mori, H.; Yuihara, A. Deterministic annealing clustering for ANN-based short-term load forecasting. *IEEE Trans. Power Syst.* **2001**, *16*, 545–551. [CrossRef]
27. Teixeira, M.A.; Zaverucha, G.; da Silva, V.N.A.L.; Ribeiro, G.F. Recurrent neural gas in electric load forecasting. In Proceedings of the 1999 International Joint Conference on Neural Networks, Washington, DC, USA, 10–16 July 1999; pp. 3468–3473.
28. Kim, C.; Yu, I.K.; Song, Y.H. Kohonen neural network and wavelet transform based approach to short-term load forecasting. *Electr. Power Syst. Res.* **2002**, *63*, 169–176. [CrossRef]
29. Mori, H.; Itagaki, T. A precondition technique with reconstruction of data similarity based classification for short-term load forecasting. In Proceedings of the 2004 IEEE Power Engineering Society General Meeting, Denver, CO, USA, 6–10 June 2004; pp. 1–6.
30. Jin, L.; Ziyang, L.; Jingbo, S.; Xinying, S. An efficient method for peak load forecasting. In Proceedings of the 7th International Power Engineering Conference, Singapore, 29 November–2 December 2005; pp. 1–6.
31. Jin, L.; Feng, Y.; Jilai, Y. Peak load forecasting using hierarchical clustering and RPROP neural network. In Proceedings of the IEEE 2006 Power Systems Conference and Exposition, Atlanta, GA, USA, 29 October–1 November 2006; pp. 1535–1540.
32. Yang, J.; Stenzel, J. Historical load curve correction for short-term load forecasting. In Proceedings of the 7th International Power Engineering Conference, Singapore, 29 November–2 December 2005; pp. 1–6.

33. Fan, S.; Chen, L. Short-term load forecasting based on an adaptive hybrid method. *IEEE Trans. Power Syst.* **2006**, *21*, 392–401. [CrossRef]
34. Fan, S.; Mao, C.; Chen, L. Electricity peak load forecasting with self-organizing map and support vector regression. *IEEJ Trans. Electr. Electron. Eng.* **2006**, *1*, 330–336. [CrossRef]
35. Hu, G.S.; Zhang, Y.Z.; Zhu, F.F. Short-term load forecasting based on fuzzy c-mean clustering and weighted support vector machines. In Proceedings of the Third International Conference on Natural Computation, Haikou, China, 24–27 August 2007; pp. 1–6.
36. Amjady, N. Short-term bus load forecasting of power systems by a new hybrid method. *IEEE Trans. Power Syst.* **2007**, *22*, 333–341. [CrossRef]
37. Kiartzis, S.J.; Zournas, C.E.; Theocharis, J.M.; Bakirtzis, A.G.; Petridis, V. Short term load forecasting in an autonomous power system using artificial neural networks. *IEEE Trans. Power Syst.* **1997**, *12*, 1591–1596. [CrossRef]
38. Independent Power Transmission Operator (IPTO) S.A. Available online: http://www.admie.gr/nc/en/home/ (accessed on 18 February 2022).
39. Hellenic National Meteorological Service. Available online: http://www.hnms.gr/emy/en/index_html? (accessed on 12 February 2022).
40. Xu, R.; Wunsch, D. *Clustering*; John Wiley & Sons Inc.: Hoboken, NJ, USA, 2006.
41. Wu, Y.; Wu, Y.; Liu, X. Couple-based particle swarm optimization for short-term hydrothermal scheduling. *Appl. Soft Comput.* **2019**, *74*, 440–450. [CrossRef]
42. Harikumara, S.; Surya, P.V. K-Medoid clustering for heterogeneous data sets. *Proc. Comput. Sci.* **2015**, *70*, 226–237. [CrossRef]
43. Liu, X.; Jiang, A.; Xu, N.; Xue, J. Increment entropy as a measure of complexity for time series. *Entropy* **2016**, *18*, 22. [CrossRef]
44. Fu, Z.; Han, B.; Chen, Y. Levenberg–Marquardt method with general convex penalty for nonlinear inverse problems. *J. Comput. Appl. Math.* **2022**, *404*, 113771. [CrossRef]
45. Liu, X.; Gu, H. Hyperbolic tangent function based two layers structure neural network. In Proceedings of the 2011 International Conference on Electronics and Optoelectronics, Dalian, China, 29–31 July 2011; pp. 376–379.
46. Tsekouras, G.J.; Hatziargyriou, N.D.; Dialynas, E.N. Two-stage pattern recognition of load curves for classification of electricity customers. *IEEE Trans. Power Syst.* **2007**, *22*, 1120–1128. [CrossRef]
47. Rousseeuw, P. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* **1987**, *20*, 53–65. [CrossRef]
48. Cancino, A.E. Load Profiling of MERALCO Residential Electricity Consumers using Clustering Methods. In Proceedings of the 2010 Conference of the Electric Power Supply Industry Conference, Taipei, Taiwan, 24–28 October 2010; pp. 1–24.