2025 8th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech) | 979-8-3315-2076-2/25/$31.00 ©2025 IEEE | DOI: 10.1109/IEMENTECH65115.2025.10959522

Long term Load Forecasting using Machine Learning Approach

Debashis Jana
Electrical Engineering, Institute of Engineering & Management, University of Engineering & Management, Kolkata, India
debashis.jana@iem.edu.in

Rajdeep Das
Electrical Engineering, Institute of Engineering & Management, University of Engineering & Management, Kolkata, India
jdas00172@gmail.com

Debojyoti Jha
Electrical Engineering, Institute of Engineering & Management, University of Engineering & Management, Kolkata, India
debojyotijha72@gmail.com

Prajit Kumar Barik
Electrical Engineering, Institute of Engineering & Management, University of Engineering & Management, Kolkata, India
prajitbarik2004@gmail.com

***Abstract*— Electrical load forecasting is an vital module of power system planning and operation because accurate predictions regarding future electricity demand are imperative in the efficient management of resources and ensuring the reliability and stability of the power grid. Long-term load forecasting using machine learning approaches like ANNs in MATLAB offers effective resource management and maintaining reliability of the grid. Based on a historical load demand data set of 8,867 points, with a 24-hour lag feature to take care of temporal fluctuations, the study utilized an ANN model having 20 hidden layers and optimized it for both accuracy and computational efficiency. Three training algorithms were evaluated to check the convergence, accuracy, and generalization by Levenberg-Marquardt, Scaled Conjugate Gradient, and Bayesian Regulation. Results directed to identification of the appropriate algorithm that enhances energy distribution and management.**

Keywords Load Forecasting, Artificial Neural Network, Machine Learning, time series, MSE

I. INTRODUCTION

Over the past few decades, the rate of growth in electricity demand has been accompanied by power system restructuring. Electricity demand is inherently uncertain and intermittent [1] in nature, coupled with traditional error in load forecasting for significant challenges on the operation and planning of bulk power grids. Load forecasting is one of the key elements that must be designed into power systems as it helps manage energy appropriately, facilitates economic dispatch [2, 3], and also ensures system stability [4]. It comes more challenging when renewable resources have been introduced in the grid along with continuously varying load [5, 6]. In the face of the escalating intricacy of power grids and the increasing popularity of renewable energy resources, sophisticated load forecasting techniques have been essential for power grids. The techniques of load forecasting can be differentiated into two types that are traditionally statistical methods and machine learning-based methods [7]. Traditional methods, such as time series models-ARIMA and exponential smoothing have been widely applied in load forecasting for a long time. However, capturing nonlinearities and complexities in the inherent load patterns makes these models less effective, especially under various conditions of rapidly changing load demand or high variability associated with weather influences [8]. ANNs have proved themselves to be very flexible and can be used effectively in load forecasting studies with a high predictability level. Some of the architectures used in the neural network are feed-forward neural networks (FNN), convolutional neural networks (CNN) and recurrent neural networks (RNN). Among them, most researchers have used FNN along with back-propagation in STLF. The choice of algorithms for neural network training is critical for high prediction accuracy [9]. Various algorithms, such as Levenberg-Marquardt (LM), Bayesian Regulation (BR), and Scaled Conjugate Gradient (SC), have been used for this optimization. Bayesian Regulation has demonstrated superior performance in avoiding overfitting and ensuring better generalization based on a probabilistic framework that adjusts the weights of the neural network. Feature selection is important to improve performance in load forecasting models. The most common input features used include historical load data, weather conditions, calendar variables (for example, day of the week and holidays) and socioeconomic factors [10]. There are also some of the disadvantages of neural network-based models. The large amounts of data for training, are sensitive to selection of hyperparameters, and computationally heavy. The future research requires deep learning techniques, hybrid models that combine the benefits of machine learning with the power of statistical methods, and the use of sophisticated feature engineering techniques. It includes further improvement of forecasting accuracy by ensemble models and applying transfer learning [11]. In order to address these issues, the use of a novel approach using machine learning for electrical demand forecasting is proposed herein. The ANN has an input layer having one or more hidden layers with an output layer. Training is divided into the three big steps: (a) input load data that is used to push forward through the network, (b) errors between the predicted and actual outputs are computed and propagated backward through the network, and (c) the weights of the connections between neurons get modified in a

979-8-3315-2076-2/25/$31.00 ©2025 IEEE

way to try to minimize the mean squared error. The input data that is entered undergoes transformations by the hidden layers using activation functions till it produces the final output. The architecture of neural networks, in terms of the depth and number of neurons in the hidden layers, affects model performance to a great extent. Although deeper networks can learn more complex features, caution must be exercised to avoid over fitting. Empirical studies indicate that a moderate number of neurons is generally the best for load forecasting, keeping in mind the scope and nature of the dataset. The advantage of the ANN model is its self-learning and self-organizing ability, and such ability can enable the model to be adaptive and enhancing its forecasting capability with time. This method promises to address the obstacles involved in long-term electricity demand forecasting by giving a tool that may help enhance the reliability and stability of power systems amid increasing demand and limited infrastructure expansion. The implementation of such advanced neural network models is crucial for enhancing the efficiency and accuracy of electricity demand forecasting, thereby supporting better operational and planning decisions in power systems [12]. The performance of load forecasting models is measured against mean squared error (MSE), mean absolute error (MAE), and root mean squared error (RMSE). In cross-validation tests, ANNs seem to perform better than traditional models especially when the patterns are nonlinear. However, the success of an ANN model is also determined by the quality of data as well as robust feature engineering practices.

II. Problem Description And Formulation

Accurate load forecasting is crucial in efficient power system operation and planning. Utility firms rely on the precise forecast to monitor generation schedules, optimize operations of the power grid, maintain stability, and subsequently save on generation costs. The process is complex owing to nonlinear patterns caused by changing weather, economic activity, or consumer behavior. This article aims at building a stable ANN-based STLF that provides an accurate hourly estimate for a period of 24 hours. ANNs were selected for their capability to capture complex dependencies beyond the reach of traditional statistical techniques. The past load values are the only input to capture any time of patterns [13]. In this project, a 24-hour lag is utilized to predict the next 24 hours of load demand. Depending upon the availability of data, additional features such as temperature, day-of-the-week, and holiday indicators can be added to improve model performances. A feed forward neural network with an input layer, one or more hidden layers, and an output layer is used. For this project, it has 20 hidden neurons in the model. Bayesian Regulation (trainbr) is the algorithm used to optimize weights of model for finding the best predictive value and preventing over fitting.

Inputs: $X = \{x_1, x_2, x_3, \dots, x_{24}\}$ and represents the input feature for time i.

Outputs: $Y = \{y_1, y_2, y_3, \dots, y_{24}\}$ in which y_j represents load prediction for j^{th} hour of forecast horizon.

The ANN model is defined as a nonlinear function f that maps the input features X into the output $Y: Y = f(X; \theta)$ where θ represents the ANN's parameters, that is, weights and biases. The load forecasting model performance can be assessed using metrics like Mean Absolute Error, MSE, and RMS error. Where y_i, i is the actual load and $(\hat{y}_i), i$ is the predicted load for each time step i, and N is the total number of predictions.

The model assumes that its training is based on high quality historical load data and the underlying patterns in the data are stationary over time. The features in the input are presumed to be adequate to capture the dependencies underlying accurate load forecasting. Machine learning approaches, particularly ANN have shown great promise in load forecasting. ANNs can model complex and nonlinear patterns in data. Therefore, they can be applied to document detailed electricity consumption patterns. Different studies prove that ANNs generally perform better than conventional methods, especially when load demand data are affected by several variables such as humidity, temperature, time of the day, and consumer behavior.

III. Methodology And Implementation

Traditional statistical methods often fail to capture the intricate, nonlinear dynamic inherent in load data, which could be due to economic growth, climate variation, and societal trends. Artificial Neural Networks (ANNs) seem to be the necessary tool that seeks to attack this complexity. ANNs can learn complex patterns from historical data, which is the reason they should be used with good prospects for long-term load forecasting.

In the proposed methodology, training an ANN model on a comprehensive dataset combining historical load data, economic indicators, and meteorological factors is considered. The typical network is composed of an input layer with several hidden layers along with an output layer. The input layer takes the mentioned features, and the hidden layers operate based on nonlinear activation functions to extract complex patterns and relationships. The output layer produces the predicted load values. Advanced techniques such as feature engineering, data normalization, and optimization algorithms are used to improve the accuracy and robustness of the model. Feature engineering is the process of creating meaningful features from raw data, which may include seasonal components, trend components, and holiday effects. Data normalization ensures that features fall within a common range, thus improving training efficiency. Optimization algorithms, such as gradient descent and Adam, were employed to adjust the parameters of the network for optimal error between predicted and actual load values. The proposed ANN model has the following advantages over traditional approaches.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \quad (1)$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \quad (2)$$

$$R = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2(y_i - \bar{y})^2}} \quad (3)$$

$$R^2 = 1 - \frac{\sum_i(y_i - \hat{y}_i)^2}{\sum_i(y_i - \bar{y})^2} \quad (4)$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \quad (5)$$

ANNs can capture nonlinear relationships, adapt to changing load patterns, and handle noisy and uncertain data. Nonetheless, problems in data quality, model complexity, computational cost, and interpretability need to be addressed judiciously to yield accurate and reliable forecasts. In this light, by mitigating these challenges and reaping the effectiveness of ANNs, one is able to reap considerable improvements in long-term load forecasting so that the power industry can make more informed decisions.

A. Data Collection and Preprocessing

Accurate collection of historical load data forms the crux of the beginning stage and has a major impact on the model's overall performance. The data is required to adequately capture all significant features that affect electricity demand, including temperature, humidity, day of the week, seasonality, and other external influences known to have impacts on consumption patterns. This data is usually put in an organized Excel format where the structured columns clearly identify the relevant features and the target variable, i.e., load demand. The data are T imported into MATLAB using read table functions, such as xlsread, or readtable for more flexibility. This allows access to numeric values from multiple sheets, so it can easily extract and analyze these data for model training in practice. Data typically not numeric, like day types or categorical variables, need to be translated into a useful numerical representation. For example, days of the week could be represented as integers in order to make analysis easier. Missing or missing data values, or those that are inconsistent, will badly affect model performance when left alone. Interpolation is one technique used to estimate missing values by relying on neighboring data. Data cleaning can then be done to correct anomalies and outliers that could mislead the algorithm in its predictions. Following import and data cleansing, the raw dataset needs to be transformed into a format suitable for neural network training. MATLAB's tonndata function is very useful in this regard, transforming the data into a time series format such that the neural network could handle sequential inputs aptly. In load forecasting, historical values are highly correlated with future loads.

Input and target series define the data set preparing for modeling. This usually includes past load values along with additional predictive features that could be weather conditions or time-related variables. The targets constitute the future load values towards which the model will direct its prediction. Separating the data allows the network to learn the mapping from the historical inputs to future outputs. Segmentation of data into meaningful input-output pairs ensures the model captures temporal dependencies critical for proper forecasting. Preparation of data in this structured fashion forms the necessary base for effective training and prediction by a neural network.

B. Model Design of the Neural Network

A TDNN is used to capture the relationship between past and future load values; it exploits the capability of TDNN in modeling time-dependent processes. The TDNN architecture is configured such that input delays are set to span from 0 to 24 hours so as to capture load patterns efficiently throughout the day. These input delays would allow the network to understand how values for earlier hours of the day influence both current and future loads to be demanded.

Designing the hidden layer is a crucial step in determining model complexity. The number of neurons, such as 10 in this case, is chosen based on a trade-off between model expressiveness and overfitting risk. Too few neurons may lead to underfitting, where the model fails to capture essential data patterns, while too many can cause overfitting, where the model becomes too specific to the training data. The neural network is built with MATLAB's timedelaynet function that provides a structured way of building time-series networks. Other options like activation functions, and the parameters for learning, can also be configured to optimize performance.

C. Data Partition

The division of data into training, cross-validation, and testing sets is essential to building an effective model. The division is usually done as follows:

Training Set (70%): Their purpose is to learn model parameters. This portion contains the most data, allowing the model to recognize patterns and relationships within a dataset.
Validation Set (15%): This set is used for tweaking the model parameters, like the number of neurons or learning rates. The validation set thereby prevents the problem of overfitting because it indicates to the model when it learns too many details from the training data, thus lowering its generalization ability.
Testing Set (15%): It is used to test the strength of the machine learning model after training and validation. This is an unbiased measure of how the model would perform on new, unseen data.
In the MATLAB , divideParam is used to define parameters for splitting the data. It ensures that the process is consistent and systematic in dividing the data, which is essential for reliable model performance assessment.

D. Training the Neural Network

The three algorithms train three neural networks, and the best algorithm is chosen for the model of load forecasting.

Levenberg-Marquardt: This algorithm has high efficiency with small datasets, with fast convergence. It minimizes the error by iterating on the model parameters, using a balance between gradient descent and Gauss-Newton method. Although trainlm is extremely efficient it does have significant memory requirements that may not be ideal for large datasets. Bayesian Regularization (trainbr): This algorithm provides regularization to avoid overfitting. It adjusts the complexity of a network using a probabilistic framework by ensuring that, even in the case of new data instances, the model functions well. The Bayesian approach dynamically weights the contribution of every neuron, hence leading to a more refined model. Scaled Conjugate Gradient (trainscg): This training algorithm is especially efficient in larger datasets and it is memory-aware since it never needs to compute a Hessian. The utility of this approach makes it very suitable for scenarios in which there is a serious memory constraint.
The train function in MATLAB is called to start the training, and a structure (tr) returns, displaying the activity and performance of the model at each iteration.

E. Performance Measurement of the Model

Model performance can be measured with a number of metrics and visualizations. Using the perform function MSE has been calculated. It measures the average of the squared error in the predicted and actual load value. The smaller the MSE value, the better the model. The training performance plots used to show the decrease in the error produced as time advances how well the model is learning. The training state displays all significant parameters, including gradients and learning rates throughout the training process to give an image when things start to go wrong or improve. In this article network response plot is also shown. Network response is a plot comparing the actual load values to forecasted values to gauge the model's prediction accuracy.Error Autocorrelation is applied to test the residual (differences between actual and predicted values) for any significant patterns. The absence of autocorrelation establishes that the models have captured the underlying patterns appropriately.Wheres input-error correlation is a measure to check for relationships between input features and errors; it may be an area of improvement with respect to feature selection or model design.

F. Delay Removal and Early Prediction

When processing the result post-training, MATLAB's removedelay function can be used to eliminate any delayed inputs. This step simplifies the trained network by eliminating dependencies that do not arise early in prediction. Preparation of the input data involves preparing the data with respect to early prediction, which is achieved using the command preparets. Since the model still makes good predictions even for fewer input values, this is essential for applications where rapid forecasting is needed, such that early predictions may assist in proactive decision-making.

G. Model Comparison

Once all the models are trained, a thorough comparison is done. Performance is compared in terms of MSE values, Correlation Coefficients, and Visual Plots to perceive which of the models recognizes the load patterns best. Bayesian Regularization is often quoted as a strong contender due to robustness. Levenberg-Marquardt for fast convergence and for memory efficiency Scaled Conjugate Gradient. The strengths and weaknesses of each model are discussed, and the most reliable one is finally selected for deployment into the real world.

H. Implementation

Finally, the chosen model is implemented in a real-time load forecasting system. Testing it through live scenario-based analyses will be crucial, as this integration is expected to perform consistently and accurately. Future directions of research might include feature additions, such as real-time weather data, or even more advanced neural network architectures like the LSTM (Long Short-Term Memory) or GRU (Gated Recurrent Unit), which are specialized for sequential data. Also, ensemble techniques that compose models can make forecasts better while the system is more resilient.

IV. SIMULTAION RESULTS

This paper has considered a set of load data based on hourly voltage (V), current(I) and power factor (pf) information available at 33/11 KV substation (Godishala, huzurabad, Telangana state, India) [14]. Lagged values of the load data are most often used as input to account for the temporal dependencies. Research has shown that it is sufficient to take a 24-hour lag in most cases for daily load patterns.

The input and target data for power consumption have been shown in Figure 1. The results during the implementation of the machine learning technique, have been shown in the Table 1. The Levenberg-Marquardt algorithm, or trainlm, is very efficient and generally used when training a mid-sized neural network. The 6115 data points were used for training and Mean Squared Error (MSE) is 1.1187×10^{5}. The MSE represents an average squared deviation between actual and estimated load values in the training procedure. A low MSE value suggests good performance. A Correlation Coefficient (R) of 0.9664 is obtained. The correlation coefficient is the strength and direction of the linear relation between the predicted and actual values. An R value close to 1 indicates that the model is doing a proper job in fitting the data. In the validation, MSE is 1.3592×10^{5} and R is 0.9602 are obtained for 1310 observation points; and testing phase, MSE is 1.3104×10^{5} and R is 0.9590 for the same observation by this algorithm. The trainlm algorithm achieves a high R value across training, validation,

and test sets, indicating strong performance. However, the increasing MSE from training to validation and test sets suggests potential overfitting, where the model may be slightly too tuned to the training data.

The Bayesian Regulation (trainbr) algorithm is particularly effective at regularizing neural networks, helping to prevent overfitting by incorporating Bayesian principles. The Bayesian Regulation Algorithm (trainbr) algorithm performs an outstandingly good, as its lowest MSE is at the training phase and its R values are consistently high. The absence of a separate validation phase corresponds to the capacity of the algorithm to be internally regularized. The algorithm really performs very well on load forecasting because it handles complexity well and brings strong predictions.

Table I
Observation by ***trainlm, trainbr*** and ***traincg*** algorithms

trainlm	Observations	MSE	R
Training	6115	1.1187e+05	0.9664
Validation	1310	1.3592e+05	0.9602
Test	1310	1.3104e+05	0.9590
trainbr	Observations	MSE	R
Training	7425	8.9432e+04	0.9734
Validation	0	NaN	NaN
Test	1310	1.2021e+05	0.9619
traincg	Observations	MSE	R
Training	6115	1.2501e+05	0.9620
Validation	1310	1.3874e+05	0.9558
Test	1310	1.5729e+05	0.9556

The Scaled Conjugate Gradient Algorithm Results (traincg) is designed to minimize the computational complexity for training large-scale problems. The traincg algorithm realizes the highest values of MSE, as well as the lowest R's, which means it might be worse for this task than other variants, like trainlm and trainbr. Though it is efficient for some tasks, in this case, it's just not good enough for this particular load forecasting.

The output response of power consumption model and error with respect to actual consumption has been shown in Figure 2. A error histogram has been presented in Figure 3 with 20 bins and a comparison of actual and estimated amount of power consumption for 24 hours is also shoen in Figure 4.

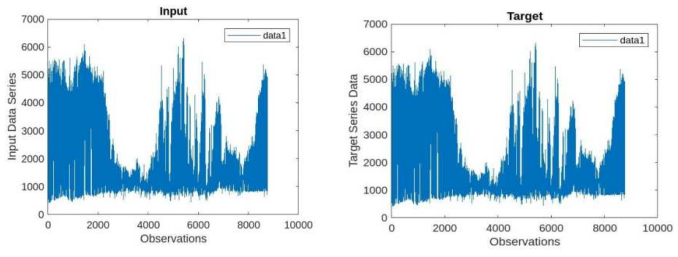

Fig 1. Input and target time-series values of power consumption

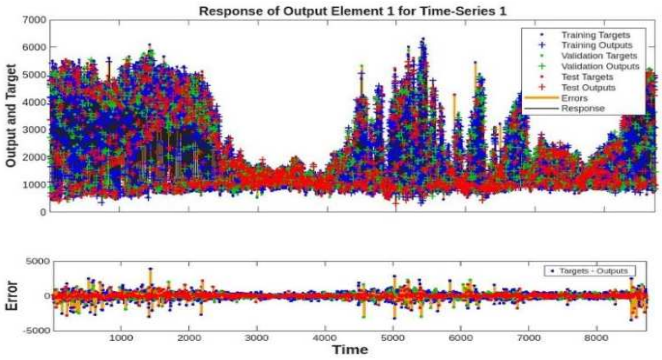

Fig 2. Output response of power consumption model and error with respect to actual consumption.

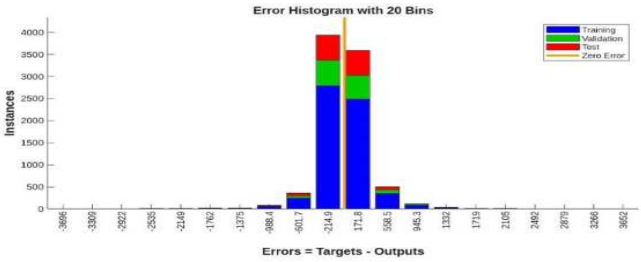

Fig 3. Histogram for error of the proposed model

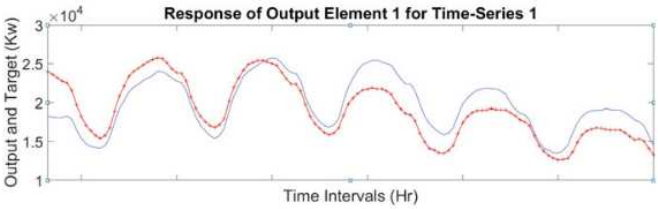

Fig 4. Actual and estimated amount of power consumption for a day.

V. Conclusion

Electricity load forecasting is a critical component of power system planning and operation because accurate predictions regarding future electricity demand are imperative in the efficient management of resources and ensuring the reliability and stability of the power grid. This aids in optimizing generation scheduling, load balancing, and infrastructure maintenance and leads to lower operational costs and enhanced service quality. In this article, machine learning approaches have been introduced for long term load forecasting in MATLAB platform. The article focuses on the improvement of forecasting accuracy based on artificial neural networks to aid optimal decision-making in energy distribution and management. The dataset used in this research comprises 8,867 data points, capturing detailed historical load demand over a specific time horizon. This data contains temporal fluctuations and complex load patterns that are helpful in building a reliable forecasting model. To eliminate daily swing cycles and recurring load trends, 24-hour lag feature has been implemented. It enables the model to use information from the previous day to make intelligent

predictions about future load demand. The ANN architecture was designed with 20 hidden layers, a configuration based on rigorous empirical analysis to balance model complexity, the ability of the model to learn, and computational efficiency. Each algorithm had its unique strengths and limitations in terms of convergence rate, accuracy, and generalization of the model. A thorough comparison was made to assess and confirm the performance of these algorithms, identifying the most suitable choice of algorithm for our load forecasting application.

REFERENCES

[1] D. Jana and N. Chakraborty, "Generalized stochastic Petri nets for uncertain renewable-based hybrid generation and load in a microgrid system," *Int Trans Electr Energ Syst*, vol. 30, no. 4, Apr. 2020, doi: 10.1002/2050-7038.12195.

[2] D. De *et al.*, "Economic load dispatch by optimal scheduling of generating units using improved real coded genetic algorithm, in *2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON)*, Bangkok, Thailand: IEEE, Aug. 2017, pp. 305 308. doi: 10.1109/IEMECON.2017.8079611.

[3] D. Jana *et al.*, Dynamic Economic Emission Dispatch using self-adaptive multivariate distribution based modified Real Coded Genetic Algorithm, in *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA: IEEE, Jan. 2017, pp. 1 5. doi: 10.1109/CCWC.2017.7868490.

[4] H. Daneshi, M. Shahidehpour and A. L. Choobbari, "Long-term load forecasting in electricity market," 2008 IEEE International Conference on Electro/Information Technology, Ames, IA, USA, 2008, pp. 395-400, doi: 10.1109/EIT.2008.4554335.

[5] D. Jana and N. Chakraborty, Probabilistic Transition-Based Optimal Energy Transaction of a Non-Convex CHP-Microgrid during Generation and Load Uncertainty, *Electric Power Components and Systems*, vol. 51, no. 12, pp. 1142 1155, Jul. 2023, doi: 10.1080/15325008.2023.2192717.

[6] D. Jana and N. Chakraborty, Generalized Stochastic Petri Nets (GSPN) for Analysis of Microgrid under Uncertainities, in *2018 20th National Power Systems Conference (NPSC)*, Tiruchirappalli, India: IEEE, Dec. 2018, pp. 1 6. doi: 10.1109/NPSC.2018.8771852.

[7] B. Zhao, L. Zeng, B. Li, Y. Sun, Z. Wang, M. Shahzad, and P. Xi, Collaborative control of thermostatically controlled appliances for balancing renewable generation in smart grid, IEEJ Trans. Electr. Electron. Eng., vol. 15, no. 3, pp. 460 468, Mar. 2020.

[8] J. R. Cancelo, A. Espasa, and R. Grafe, Forecasting the electricity load from one day to one week ahead for the Spanish system operator, Int. J. Forecasting, vol. 24, no. 4, pp. 588 602, 2008.

[9] L. J. Soares and M. C. Medeiros, Modeling and forecasting short-term electricity load: A comparison of methods with an application to Brazilian data, Int. J. Forecasting, vol. 24, no. 4, pp. 630 644, Oct. 2008.

[10] A. Kavousi-Fard, H. Samet, and F. Marzbani, A new hybrid modified firefly algorithm and support vector regression model for accurate short term load forecasting, Expert Syst. Appl., vol. 41, no. 13, pp. 6047 6056, 2014.

[11] L. Hernandez, C. Baladron, J. M. Aguiar, B. Carro, A. J. Sanchez-Esguevillas, J. Lloret, and J. Massana, A survey on electric power demand forecasting: Future trends in smart grids, microgrids and smart buildings, IEEE Commun. Surveys Tuts., vol. 16, no. 3, pp. 1460 1495, 3rd Quart., 2014.

[12] L. Xiao, W. Shao, C. Wang, K. Zhang, and H. Lu, Research and application of a hybrid model based on multi-objective optimization for electrical load forecasting, Energy, vol. 180, pp. 213 233, Oct. 2017.

[13] H. Patel, D. S. Rajput, G. T. Reddy, C. Iwendi, A. K. Bashir, and O. Jo, A review on classification of imbalanced data for wireless sensor networks,
Int. J. Distrib. Sensor Netw., vol. 16, no. 4, 2020, Art. no. 1550147720916404.

[14]Venkataramana Veeramsetty, Electric power load dataset. Mendeley, Feb. 17, 2022. doi: 10.17632/TJ54NV46HJ.1.