COMPUTATIONAL MODELING OF ELECTRICITY CONSUMPTION USING ECONOMETRIC VARIABLES BASED ON NEURAL NETWORK TRAINING ALGORITHMS

T.M. Usha,[*] *S. Appavu alias Balamurugan*[†]

Abstract: Recently, there has been a significant emphasis in the forecasting of the electricity demand due to the increase in the power consumption. Energy demand forecasting is a very important task in the electric power distribution system to enable appropriate planning for future power generation. Quantitative and qualitative methods have been utilized previously for the electricity demand forecasting. Due to the limitations in the availability of data, these methods fail to provide effective results. With the development of the advanced tools, these methods are replaced by efficient forecasting techniques. This paper presents the computational modeling of electricity consumption based on the Neural Network (NN) training algorithms. The main aim of the work is to determine the optimal training algorithm for electricity demand forecasting. From the experimental analysis, it is concluded that the Bayesian regularization training algorithm exhibits low relative error and high correlation coefficient than other training algorithms. Thus, the Bayesian Regularization training algorithm is selected as the optimal training algorithm for the effective prediction of the electricity demand. Finally, the economic input attributes are forecasted for next 15 years using time series forecasting. Using this forecasted economic attributes and with the optimal Bayesian Regularization training algorithm, the electricity demand for the next 15 years is predicted. The comparative analysis of the NN training algorithms for the proposed dataset and larger datasets obtained from the UCI repository and American Statistical Association shows that the Bayesian Regularization training algorithm yields higher correlation value and lower relative error than other training algorithms.

Key words: *activation functions, neural network model, neural network training algorithms, Bayesian regularization training algorithm, time series forecasting, electricity demand forecasting*

Received: February 24, 2015 **DOI:** 10.14311/NNW.2017.27.007
Revised and accepted: October 1, 2016

[*] T.M. Usha – Corresponding author; Information and Communication Engineering, Anna University, India, E-mail: asharininiharish@yahoo.co.in

[†] S. Appavu alias Balamurugan; Department of Information Technology, K.L.N. College of Information Technology, Tamil Nadu, India, E-mail: app_s@yahoo.com

1. Introduction

Electricity demand forecasting [55] plays a vital role in the planning of the electricity production, since it determines the quantity of the required resources to operate the electricity plants. Further, it is the cornerstone of planning for electric plants and networks. Prediction of the electric load demand pattern is very complex, due to the irregular nature of the energy markets. It is therefore necessary to develop new prediction methods to reduce the uncertainty of the predictions. Accurate demand forecasting enables to make correct decisions for future planning and development. This enables significant reduction in the operation and maintenance costs and improved reliability of the power supply and distribution system. It is really a critical task to find an appropriate forecasting model for the electricity demand forecasting. Many forecasting methods are developed to solve this problem.

Traditional forecasting techniques such as time series, regression, econometric and soft computing techniques such as Fuzzy logic, genetic algorithm and NNs are utilized for the electricity demand forecasting [43]. But, none of these methods have been generalized for the demand patterns of the power distribution network. Among all the existing time series prediction models, the NN [4] shows better performance in terms of the relative error and correlation coefficient than any other model.

The NN model [5] is a recently developed class of nonlinear models, based on the principles derived from the structure of the brain. The NNs are trained by using the training algorithms. There are three types of training algorithms that are found to be effective for variable range of weights.

- The Newton algorithm is found to be efficient for a small number of weights. The memory requirement of this algorithm is directly proportional to the square of the number of weights.

- Various Quasi-Newton algorithms are efficient for a moderate number of weights. The memory requirement of these algorithms is directly proportional to the square of the number of weights.

- Various conjugate-gradient algorithms are efficient for a large number of weights. The memory requirement of these algorithms is directly proportional to the number of weights.

The main disadvantage of the existing prediction models is its difficulty with infinite recursion and structured representations. Hence, an efficient prediction model is required to overcome these shortcomings. The purpose of this work is to determine the optimal prediction method for forecasting electricity demand using an economic data set. The main focus is to find an optimal training algorithm for the forecasting purpose. All the prediction techniques are applied to the dataset compiled from the Tamil Nadu Electricity Board and Statistical Department.

The empirical comparison results show that the Bayesian Regularization training algorithm achieves significant reduction in the relative error and high correlation coefficient, among all other training algorithms. Based on this concept, the optimal training algorithm is determined based on the correlation coefficient and relative error of the training algorithms. Therefore, it is concluded that the Bayesian Regularization training algorithm is the optimal training algorithm for electricity demand forecasting.

The rest of the paper is systematized as follows: Section 2 describes the existing works related to the electricity demand forecasting techniques. Section 3 illustrates the NN model used in this work. Section 4 describes the activation functions for NN. Section 5 explains about the classification of the NN training algorithms. Section 6 presents the dataset description and Section 7 involves the results and discussion including the comparative analysis of the relative error and correlation coefficient of various training algorithms and ranking of the training algorithms. The conclusion and future implementation of this work are discussed in the Section 8.

2. Literature survey

This section explains about the existing electricity demand forecasting techniques. Various types of classifications based on the forecasting methods were introduced over a period of time. [55] developed an improved hybrid model including moving average, combined method, hybrid model and adaptive PSO algorithm (MA-C-WH) for forecasting electricity demand in China. The performance of the proposed MA-C-WH model was compared with the existing seasonal ARIMA (SARIMA). Based on the results of popular forecasting precision indexes, the proposed model was found to be effective for seasonal time series with nonlinear trend. [43] reviewed the various energy demand forecasting models including traditional methods and soft computing techniques. The support vector regression, optimization techniques such as Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO) were also adopted for energy demand forecasting. The Market Allocation (MARKAL) and Long range Energy Alternatives Planning (LEAP) models were also used for the energy demand management. [14] proposed a NN based approach for selecting the best prediction method depending on small number of customers. The proposed approach did not require frequent retraining.

[5] developed a weighted fuzzy NN for monthly electricity demand forecasting in Taiwan. The fuzzy NN framework was modified and the significance of every factor amongst the different rules was calculated using a weighted factor. The NN was trained using the historical data, to forecast the future electricity demands. The monthly electricity demand forecasting accuracy of the NN model was higher than the other approaches. [2] proposed a novel approach that combined multi-output feed forward NN with filtering and seasonal adjustment. Empirical mode decomposition (EMD) based signal filtering was performed for reducing the noise signals. The seasonal component was removed from the denoised series and the resultant series was modeled with a multi-output strategy. The season indexes were restored to the forecasts and final prediction was obtained. The forecasting accuracy was improved compared to the existing models.

[49] analyzed the PSO optimal Fourier Method, seasonal Autoregressive Integrated Moving Average Model (ARIMA) model and also combined models of both techniques for correcting the seasonal ARIMA forecasting results. The prediction accuracy of the three residual modification models was better than single seasonal

ARIMA model. The combined model was found to be more satisfactory than the models. [45] introduced a decomposition approach for modeling the variation in the electricity demand trend for medium and long-term forecasting. The historical time series was decomposed into a number of components according to the seasonality variation, daily activity and day of the week. However, the decomposition approach was relatively feasible for implementation, since it did not require the structural models or time series analysis and reduced the complex non-linear parameter estimation efforts.

[32] proposed a hybrid forecasting framework including a Multi-Input Multi-Output (MIMO) forecasting engine to predict the electricity demand and price. A Data Association Mining (DAM)-based rule extraction mechanism was employed for determining and extracting the customer reaction patterns to the price forecast. These extracted rules were used for tuning the initial forecasts of the MIMO engine. [44] introduced a semi-functional partial linear model for forecasting the electricity demand and price. The new forecasting model was compared with a naïve method and seasonal ARIMA model. The performance of the new forecasting model was better for the electricity demand forecasting than the price forecast. [39] presented a new mid-term electricity demand forecasting framework for the practical and reliable forecast using the measurable amount of external variables. The performance result of the proposed approach was better than the decomposition forecasting methods. [47] proposed a novel approach that combined the first-order gray differential equation and seasonal fluctuation from time series method. The proposed model achieved a better performance than the original gray differential equation model.

A nonlinear time series modeling technique was applied to analyze the electricity demand. A weighted largest Lyapunov exponent forecasting method was proposed to improve the prediction accuracy. The PSO algorithm was used for determining the optimal weight parameters of the forecasting method. The mean absolute relative error (MARE) of the prediction model was relatively lower than the forecasting errors of the existing methods [46]. [18] discussed the most relevant studies on electricity demand prediction over the last 40 years, and presented different models for the future trends. Additionally, the latest studies on the demand forecasting were analyzed in the future environments [53] modeled the electricity demand in the Ontario province by using a neuro Fuzzy inference system. A neuro-fuzzy model was created for the electricity demand, based on the data collected using statistical methods [8] investigated the relationship between Turkish residential electricity consumption, using the application of the structural time series model to the annual data. Finally, the Turkish residential electricity demand was predicted accurately, based on different forecast assumptions.

[52] developed a hybrid energy demand forecasting procedure with higher precision, using PSO-Genetic Algorithm (GA) approach. The superior performance of the proposed forecast method was higher than the single optimization methods and multiple linear regressions. [19] presented a Multi-Agent System (MAS) model for virtual power plants. A set of agents was embedded with artificial NNs for the collaborative forecasting of disaggregated energy demand of the domestic users. The MAS was fed with relevant data to make informed decision, due to the reduction in the error rate. [25] proposed a new hybrid method including PSO and

ACO for estimating energy demand of Turkey. Estimation of the future energy demand was done under different scenarios. The relative estimation errors of the HAPE model were low and fitting solutions were provided. [40] proposed a Pattern Forecasting Ensemble Model for day-ahead prediction of electricity demand. Five forecasting models were implemented using different clustering techniques based on the Pattern Sequence Similarity algorithm. The performance of the proposed model was evaluated on electricity demand datasets and compared with five forecasting models. The performance of the proposed model was better in terms of Mean Relative Error (MRE) and Mean Absolute Error (MAE), when compared to the five forecasting models.

[41] presented a regression-SARIMA model with generalized autoregressive conditional heteroskedastic (GARCH) for the electricity demand forecasting. Due to the non-constant mean value and variance of the daily peak demand data and multiple seasonality variation corresponding to weekly and monthly data, the possibility of serial correlation in the instability was solved using the GARCH modeling methodology. The forecasting accuracy of the proposed model was improved, while enabling reduction in the mean absolute percent error. [9] investigated the relationship between the variables such as Gross Domestic Product, aggregate electricity consumption rate and price to forecast the future aggregate electricity demand in Turkey. The structural time series technique was applied to the annual data over a certain time period to estimate aggregate electricity demand function for Turkey. Efficient prediction of aggregate electricity demand was performed based on the estimated equation.

3. NN model

The model that predicts the relationship between the input and output values with the sufficient learning repetitions is called Neural network (NN) models. The approximation of non-linear decision boundaries required the high precision with less computational time. The non-linear functions are utilized to compute the most fitting during the transformation of input variables. The NN model includes the input layer, hidden layers and an output layer with summation and activation function. The correlation estimation calculates the linear relationship between the input and output variables. The number of neurons in the hidden layers is responsible for relationship measurement. The NN model comprises of an input layer, few hidden layers, and an output layer. Fig. 1 shows the single neuron. A feedforward artifician NN model includes a summation function and activation function '**G**' as shown in Fig. 2. This network is made up of many nodes connected parallelly and in series.

$$n_i = \sum_{j=1}^{k} w_{ji} x_j + \theta_i, \tag{1}$$

$$y_i = \mathbf{G}_i = \mathbf{G}\left(\sum_{j=1}^{k} w_{ji}\, x_j + \theta_i\right), \tag{2}$$

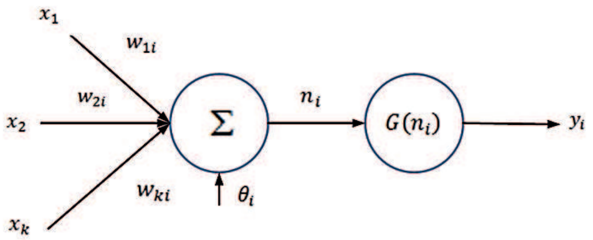

Fig. 1 Single Neuron.

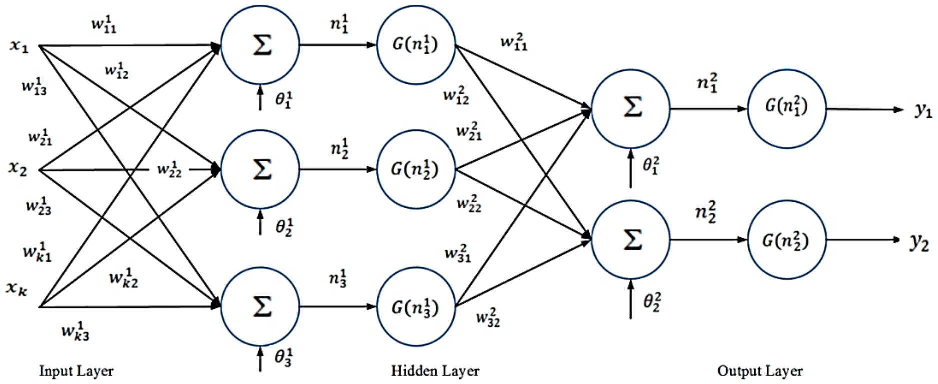

Fig. 2 Feedforward NN model.

$$y_i = g\left(\sum_{j=1}^{3} w^2_{ji} g(n^1_j) + \theta^2_j\right) = g\left(\sum_{j=1}^{3} w^2_{ji} g\left(\sum_{k=1}^{k} w^1_{kj} + \theta^1_j\right) + \theta^2_j\right), \qquad (3)$$

where

$x_k =$ Inputs of k neurons
$w_{ki} =$ Weights of k-th neuron to i-th neuron
$\theta_i =$ Constant bias
$\mathbf{G} =$ Activation function
$n_i =$ The input neurons are multiplied by weights w_{ki} and summed with the constant bias term θ_i and the superscript of 'n', 'θ' and 'w' refers to the first or second layers.
$y_i =$ Output of the Network

The architecture 9-6-5-1 has the highest correlation value as shown in Fig. 3. So, this architecture is chosen for the NN training. The architecture has 9 neurons in the input layer, 6 neurons in the first hidden layer, 5 neurons in the second hidden layer, and 1 neuron in the output layer.

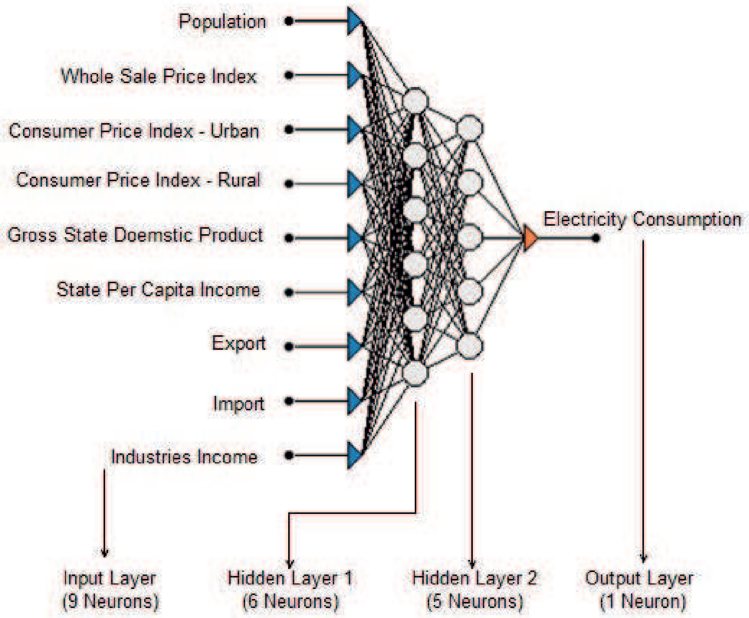

Fig. 3 Optimal Architecture 9-6-5-1.

3.1 NN parameters

NN parameters are used to obtain maximum similarity between the input and output values. So, the error between the desired and actual outputs is reduced. These NN parameters are very important and sensitive to the accuracy of the prediction. Therefore, setting the NN parameters is very significant. While creating a NN, selection of the input variable is mainly important to obtain a meaningful prediction. The variables that influence the target variable are selected as input variable. The correlation value is calculated between the input and target variables to measure their linear relationships. The numbers of hidden layers and the numbers of neurons in the hidden layers are also important for the NN architecture. These values are obtained by continuously changing the configuration during the training process using a number of ancillary algorithms. The correlation value is calculated for different combinations of numbers of hidden layers and numbers of neurons. The NN is trained with a learning algorithm called a training algorithm. Here, the NN is evaluated using different training algorithms including back propagation algorithms, gradient descent methods, conjugate gradient descent, variable metric methods and regularization methods. An activation function specifies the output of a neuron for a given input. Neurons are ‘switches’ that outputs ‘1’ when they are sufficiently activated and ‘0’ when not activated.

Training algorithms are the optimization procedures used for automatic adjustment of the weights and biases of the network. The main objective of the training algorithms is to reduce the global error $\mathbf{G}_\mathrm{E}$ defined as

$$\mathbf{G}_{\mathrm{E}} = \frac{1}{\mathrm{tp}} \sum_{\mathrm{tp}=1}^{\mathrm{tp}} E_{\mathrm{tp}}, \tag{4}$$

where tp is the total amount of training patterns and E_{tp} is the error contained in the training pattern. E_{tp} is calculated as

$$E_{\mathrm{tp}} = \frac{1}{2} \sum_{i=1}^{N} (o_i - t_i)^2, \tag{5}$$

where N is the total number of the output nodes, o_i denotes the output of the network at the i-th output node, and t_i is the target output at the i-th output node. This global error is reduced by adjusting the weights and biases.

4. Activation functions for NN

The NN [6,20,35] is used in the time series prediction to find the best relationship between the input and output values for a sufficient number of learning repetitions. Here the dataset is partitioned into two sets of data for training and testing. They allow the complex nonlinear relationships between the response variable and its predictors. The NN models [17] have concentrated on forecasting future developments of the time series from values of x up to the current time. The advantage of the NN includes automatic learning of dependencies only from measured data without the need to add further information.

4.1 Hyperbolic tangent function

The hyperbolic tangent function [42] is the most common activation function for the NNs, which produces output values ranging from −1 to 1. It is defined as the ratio between the hyperbolic sine and cosine functions or the ratio of the difference and sum of two exponential functions as seen below

$$\tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{\mathrm{e}^{x} - \mathrm{e}^{-x}}{\mathrm{e}^{x} + \mathrm{e}^{-x}}. \tag{6}$$

The sigmoid activation function does not return the values less than zero. But, it is possible to move the sigmoid function towards a region with negative values. This is performed by using the hyperbolic tangent function. Because of this higher numeric range, the hyperbolic activation function is often used instead of the sigmoid activation function. The tangent hyperbolic function is a bipolar version of the sigmoid function. Fig. 4 shows the hyperbolic tangent function.

4.2 Linear activation function

Another common activation function for the NNs is the linear activation function [27]. This function only produces positive numbers over the whole range of real numbers. The mathematical formula for this function is shown as

$$f(x) = x. \tag{7}$$

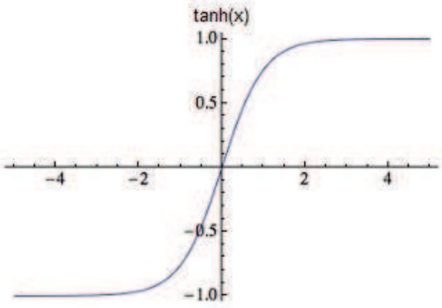

Fig. 4 Hyperbolic tangent function.

4.3 Uni-Polar sigmoid function

The sigmoid function produces the positive output values ranging from 0 to 1. Activation function of the uni-polar sigmoid function [33] is defined as

$$F(x) = \frac{1}{(1 + \mathrm{e}^{-x})}. \tag{8}$$

This function is especially advantageous for NNs trained by back-propagation algorithms. As it is easy to distinguish, it minimizes the computation capacity for training. Since the sigmoid activation function includes single constant derivative, it is suitable only for the output layer of the NN trained with the gradient descent based training methods. Fig. 5 shows the uni-polar sigmoid function.

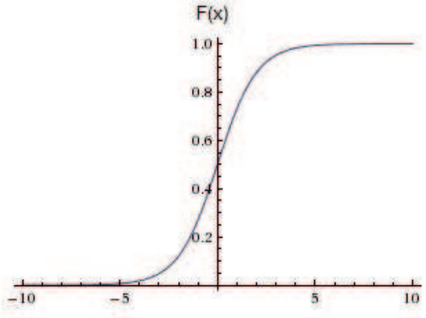

Fig. 5 Uni-Polar Sigmoid Function.

4.4 Bi-Polar sigmoid function

The bi-polar sigmoid function produces the output values in the range of $[-1, 1]$. Fig. 6 shows the bi-polar sigmoid function. Activation function of the Bi-polar sigmoid function [11] is given by

$$F(x) = \frac{1 - \mathrm{e}^{-x}}{1 + \mathrm{e}^{-x}}. \tag{9}$$

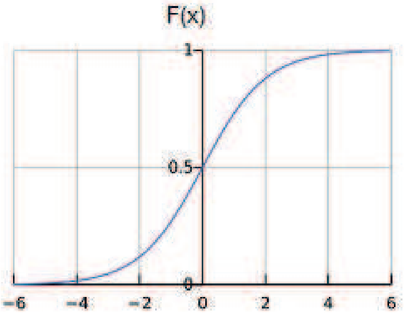

Fig. 6 Bi-polar sigmoid function.

5. Classification of NN training algorithms

Tab. I shows the training algorithms used for training the NN. With the optimal NN architecture, 17 different training algorithms and 9 sets of different combinations of 3 activation functions for the hidden layer and output layer are used in the NN training. This NN predicts the electricity consumption. The main function, advantages and drawbacks of the training algorithms are discussed in this section.

S. No	Training Algorithms
1	Quasi Newton (QN)
2	Limited Memory Quasi-Newton (LM-QN)
3	One Step Secant (OSS)
4	Levenberg Marquardt (LM)
5	Bayesian Regularization (BR)
6	Quick Propagation (QP)
7	Online Back Propagation (OBP)
8	Batch Back Propagation (BBP)
9	Resilient Back Propagation (RBP)
10	Conjugate Gradient Decent (CGD)
11	Scaled Conjugate Gradient (SCG)
12	Fletcher-Powell Conjugate Gradient (FPCG)
13	Polak-Ribiére Conjugate Gradient (PRCG)
14	Powell/Beale Restarts Conjugate Gradient (PBRCG)
15	Variable Learning Rate Gradient Descent (VLRGD)
16	Gradient Descent with Momentum (GDM)
17	Gradient Descent (GD)

Tab. I Neural Network training algorithms.

5.1 Bayesian regularization

Bayesian regularization [31,50] is a mathematical process that converts a nonlinear regression into a well-modeled statistical problem in the manner of a ridge regression. It is the most suitable method for the estimation when a large number of inputs is used for the best output. Bayesian regularization reduces the linear combination of the squared errors and weights. It also adjusts the linear combination to produce the network with good generalization qualities at the end of training. The Bayesian optimization of the regularization parameters requires the computation of the Hessian matrix of $\boldsymbol{F}(\mathbf{w})$ at the minimum point $\mathbf{w}^*$. $\boldsymbol{F}$ is the objective function and $\mathbf{w}$ is the vector of network parameters.

The Bayesian regularization requires numerical approximation of analytically intractable integrals. It provides the estimated values based on the prior approximations of the parameters. These approximations are expressed with the probability density functions. In this technique, the number of subjective choices specifications is required for the prior parameters. The prior confidence about the parameters is approximated before collecting the data. It is also used to estimate the parameters of unknown model by combining the prior knowledge and the observed data for providing a probability distribution. The regularization parameters α and β are calculated as follows:

$$\alpha = \frac{\rho}{2E_{\mathrm{w}}}, \tag{10}$$

$$\beta = \frac{E_{\mathrm{D}} - \rho}{2E_{\mathrm{D}}}, \tag{11}$$

where, α and β are the objective functions, ρ defines the effective number of parameters, E_{w} is the error of weights and E_{D} is the error of data points. It treats the weight values as a random variables and assumes that the prior probabilities of P are Gaussian. The following mathematical model is used to compute the regularization

$$P(\mathbf{v}|S,\alpha,\beta,N) = \frac{P(S|\mathbf{v},\beta,N)P(\mathbf{v}|\alpha,N)}{P(S|\alpha,\beta,N)}, \tag{12}$$

where S denotes the dataset, N represents the particular NN model, $\mathbf{v}$ is the vector of network weights, $P(S|\mathbf{v},\beta,N)$ defines the likelihood function that is the probability of the occurring data for the weights $\mathbf{v}$, $P(\mathbf{v}|\alpha,N)$ represents the prior density of the weights and $P(S|\mathbf{v},\beta,N)$ is the normalization factor that guarantees the probability as 1.

5.2 Levenberg-Marquardt

The Levenberg–Marquardt (LM) algorithm [7,13] is one of the most popular tools for solving the non-linear minimum mean squares problems. This algorithm is designed to attain the second-order training speed, without the need for computing the Hessian matrix. The Hessian matrix can be approximated as

$$\mathbf{H} = \mathbf{J}^{\mathrm{T}}\mathbf{J}. \tag{13}$$

The gradient is computed as

$$\mathbf{G} = \mathbf{J}^{\mathrm{T}}\mathbf{e}, \tag{14}$$

where $\mathbf{e}$ is a vector of network errors and $\mathbf{J}$ is the Jacobian matrix containing first derivatives of the network errors corresponding to the weights and biases. The Jacobian matrix is calculated using a standard back-propagation technique, which is really simpler than the Hessian Matrix computation.

5.3 Scaled Conjugate Gradient

The Scaled Conjugate Gradient (SCG) descent algorithm [22, 24, 38, 51] does not require the computationally expensive line search and at the same time possess the advantage of the Conjugate Gradient descent algorithms. The step size in the conjugate direction in this case is determined using the LM approach. The algorithm starts in the direction of the steepest descent given by the negative of the gradient as

$$P_u = -\nabla V_u, \tag{15}$$

where P_u is the search direction, ∇ is the gradient and V_u is the direction.

The updated weights and biases are then given by

$$X_{k+1} = X_k - \alpha_k P_k, \tag{16}$$

where X_k is the weights, α_k is the step size determined by the Levenberg-Marquardt algorithm.

The next search direction that is conjugate to the previous search directions is determined by the combination of the previous search direction with the new steepest descent direction. This is given by

$$P_k = -\nabla V_k + \beta_k P_{k-1}, \tag{17}$$

where β_k is defined as

$$\beta_k = \frac{|\nabla v_{k+1}|^2 - \nabla V_{k+1}\nabla V_k}{\mu_k}, \tag{18}$$

and where μ_k is given by

$$\mu_k = P_k^T \nabla V_k. \tag{19}$$

5.4 Back propagation

The back propagation algorithms [48, 51] performs the training of a feed-forward multilayer NN for a given set of input patterns with known classifications. The network examines its output response to the sample input pattern, during the presentation of every entry of the sample set to the network. The output response is then compared with the known and desired output and the error value is calculated. The connection weights are adjusted based on the error value.

5.5 Resilient back propagation

The main purpose of the resilient back propagation [28,37] is to eliminate the detrimental effects of the degree of the partial derivatives. The direction of the weight update is determined only by the sign of the partial derivative. The magnitude of the partial derivative does not have any effect on the weight update. By using a different update value, the size of the weight change is determined.

There is an increase in the update value for every weight and bias by a specific factor, when the derivative of the performance function holds the same sign for two successive iterations, with respect to that particular weight. The update value is reduced by the factor, when the derivative changes sign from the previous iteration, with respect to that weight. The update value for the weight and bias remains the same, if the derivative is zero. The weight change is decreased, during variations in the weights. The magnitude of the weight change increases, if the weight change continues in the same direction for multiple iterations.

$$\Delta W_{ij}(t) = \begin{cases} -\Delta P_{ij}, & \text{if } \dfrac{\partial E}{\partial W_{ij}}(t) > 0 \\ -\Delta P_{ij}, & \text{if } \dfrac{\partial E}{\partial W_{ij}}(t) < 0 \\ 0, & \text{if } \dfrac{\partial E}{\partial W_{ij}}(t) = 0 \end{cases}, \tag{20}$$

$$\Delta P_{ij}(t) = \begin{cases} \alpha^{+}.\Delta W_{ij}(t-1), & \text{if } \dfrac{\partial E}{\partial W_{ij}}(t-1).\dfrac{\partial E}{\partial W_{ij}}(t) > 0 \\ \alpha^{-}.\Delta W_{ij}(t-1), & \text{if } \dfrac{\partial E}{\partial W_{ij}}(t-1).\dfrac{\partial E}{\partial W_{ij}}(t) < 0 \\ \Delta W_{ij}(t-1), & \text{if } \dfrac{\partial E}{\partial W_{ij}}(t-1).\dfrac{\partial E}{\partial W_{ij}}(t) = 0 \end{cases}, \tag{21}$$

where α is the learning rate, W is the weight, P is the change in the weight, $\alpha^{+} = 1.2$ and $\alpha^{-} = 0.5$.

5.6 Online back propagation

The online back propagation algorithm is a heuristic method that includes two phases. During the first phase, the learning rate is adjusted after every iteration, so as to quickly attain the minimum value of the error criteria on the validation set. The search process is refined during the second phase, by repeatedly returning to previous weight configurations and reducing the global learning rate. Training is performed online, i.e. the weights of the NN are updated after the presentation of each training sample. The given weight $W_{ij}(t)$ is updated by adding a $\Delta W_{ij}(t)$ at every iteration t,

$$\Delta W_{ij}(t) = -\varepsilon(t)\frac{\partial E(t)}{\partial W_{ij}(t)}, \tag{22}$$

where ε is the learning rate. The validation set is used to control the adjustment of the learning rate $\varepsilon(t)$ after each training iteration t. The true gradient descent is not involved in this method, since the sum of all pattern derivatives over the given iteration is never determined for a particular set of weights. Instead of this, the weights are changed slightly after each pattern, by evaluating the pattern derivatives that are relative to slightly different weight values.

5.7 Batch back propagation

In the batch back propagation approach [54], all patterns are provided for the network before the learning process. In the batch training protocol, initially all the training patterns are presented and their corresponding weight updates are summed. Then the actual weights in the network are updated. This process is repeated, until some stopping criterion is satisfied. In batch back propagation, there is no need to select the patterns randomly, since the weights are updated only after the presentation of all patterns.

In batch mode, the value of $\partial E_P/\partial W_{ij}$ is calculated after the submission of every pattern to the network. Then, the total derivative $\partial E/\partial W_{ij}$ is calculated during the end of a given iteration, by the summation of the individual pattern derivatives. The weights are updated, after the calculation of the total derivative. The batch mode approximates the gradient descent, as far as the learning rate ε is smallest.

5.8 Powell/Beale restarts conjugate gradient

The Beale-Powell restart algorithm [38, 51] is highly useful for the large-scale unconstrained optimization applications. The search direction is reset periodically accoding to the negative values of the gradient, based on the inequality condition. If the number of iterations becomes equal to the number of the network parameters such as weights and biases, there will be the occurrence of the standard reset point. The training efficiency is improved by other reset methods. Powell and Beale have proposed a reset method, if the orthogonality between the current gradient $\mathbf{g}_k$ and previous gradient $\mathbf{g}_{k-1}$ is low. This is validated using the following inequality:

$$|\mathbf{g}_{k-1}^{\mathrm{T}}\mathbf{g}_k| \geq 0.2\|\mathbf{G}_k\|^2. \tag{23}$$

The search direction is reset according to the negative value of the gradient, if this inequality condition is satisfied.

5.9 Polak-Ribiere conjugate gradient

Polak and Ribiere [38, 51] have proposed another version of the conjugate gradient algorithm. The search direction for each iteration is determined by

$$P_k = -\mathbf{G}_k + \beta_k P_{k-1}, \tag{24}$$

where P_k is a search direction.

For the Polak-Ribiére update, the constant β_k is computed by

$$\beta_k = \frac{\Delta g_{k-1}^{\mathrm{T}}\mathbf{G}_k}{g_{k-1}^{\mathrm{T}}\mathbf{G}_{k-1}}. \tag{25}$$

This is defined as the ratio of the product of the previous gradient with the current gradient to the norm squared value of the previous gradient.

5.10 One-step secant

As the Quasi-Newton algorithm exhibits high storage and computational complexity in each iteration when compared to the conjugate gradient algorithms, there arises a need for the secant approximation. The OSS training algorithm [51] needs minimum stotrage and computation requirements than the Quasi-Newton algorithm, and slightly high storage and computation requirements than the conjugate gradient algorithms. Thus, the OSS method is considered as a mutual compromise between the Quasi-Newton algorithms and conjugate gradient algorithms.

5.11 Limited memory Quasi-Newton

The Limited Memory Quasi-Newton method [12,36] is used to update the variables with indices outside the active set. The main idea behind this approach is to use the information from only the most recent iterations only, while the information from earlier iterations is discarded for reducing the memory consumption.

5.12 Quasi-Newton

Newton's method [4], is found to be an alternative for the conjugate gradient approaches because of its rapid optimization. The fundamental step of the Newton's method is

$$X_{k+1} = X_k - \mathbf{A}_k^{-1}\mathbf{G}_k, \tag{26}$$

where $\mathbf{A}$ is the Hessian matrix of the performance index of the present values of the weights and biases.

Quasi-Newton method [10,24] involves a generation of a sequence of matrices $\mathbf{G}$ that represents increasingly accurate approximations to the inverse Hessian. Using only the first derivative information of E, the updated expression is presented as follows:

$$\mathbf{G}_{k+1} = \mathbf{G}_k + \frac{\mathbf{P}\mathbf{P}^{\mathrm{T}}}{\mathbf{P}^{\mathrm{T}}\mathbf{V}} - \frac{(\mathbf{G}_k\mathbf{V})\mathbf{V}^{\mathrm{T}}\mathbf{G}_k}{\mathbf{V}^{\mathrm{T}}\mathbf{G}_k\mathbf{V}}(\mathbf{V}^{\mathrm{T}}\mathbf{G}_k\mathbf{V})\mathbf{U}\mathbf{U}^{\mathrm{T}}, \tag{27}$$

where $\mathbf{G}_k$ is a symmetric positive definite matrix and T represents the transpose of the matrix.

5.13 Variable learning rate gradient descent

The learning rate parameter is used to determine the fast convergence of the Backlinear Propagation (BP) to the minimum solution. The convergence is faster, when the learning rate is large and the step is big. To speed up the convergence time, the variable learning rate gradient descent BP utilizes larger learning rate α, when the NN model is far from the solution and smaller learning rate α, especially when the neural net is near the solution. The new weight vector $\mathbf{w}_{k+1}$ is adjusted$t$ the same as that is in the gradient descent with momentum above but with a varying α_1. Typically, the new weight vector $\mathbf{w}_{k+1}$ is defined as

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha_{k+}1\mathbf{G}_k + \mu\mathbf{w}_{k-1}, \tag{28}$$

$$\alpha_{k+1} = \beta_{\alpha k}, \tag{29}$$

where

$$\beta = \begin{cases} 0.7 & \text{If new error} > 1.04 \text{ (old error)} \\ 1.05 & \text{If new error} < 1.04 \text{ (old error)} \end{cases} . \tag{30}$$

5.14 Conjugate gradient descent

In the conjugate gradient descent algorithms [24,26] the search process is performed along the conjugate directions, to determine the step size. This produces generally faster convergence than the steepest descent directions. During every iteration, the step size is adjusted. The search direction during every iteration is determined by updating the weight vector as

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha P_k, \tag{31}$$

where

$$P_k = -\mathbf{G}_k + \beta_k P_{k-1}, \tag{32}$$

$$\beta_k = \frac{\Delta \mathbf{G}_{k-1}^{\mathrm{T}} \mathbf{G}_k}{\mathbf{G}_{k-1}^{\mathrm{T}} \mathbf{G}_{k-1}}, \tag{33}$$

$$\Delta \mathbf{G}_{k-1}^{\mathrm{T}} {=} \mathbf{G}_k^{\mathrm{T}} \mathbf{G}_{k-1}^{\mathrm{T}}. \tag{34}$$

5.15 Quick propagation

The Quick propagation algorithm [23] computes the weight change by using the quadratic function $f(x) = x^2$. Relating the secant to the quadratic function, it is possible to calculate the minimum point $f'(x) = 0$. The X-coordinate of the minimum point is decided as the new weight value.

$$S(t) = \frac{\partial E}{\partial W_i(t)} = \frac{\Delta w_i(t)}{\alpha}, \tag{35}$$

$$\Delta W_i(t) = \frac{S(t)}{S(t-1) - S(t)} \cdot \Delta W_i(t-1), \tag{36}$$

where W is the weight, i is the neuron, E is the error function, t is the time and α is the learning rate.

5.16 Gradient descent with momentum

Momentum allows the network to respond to the local gradient, and also to the recent trends in the error surface. This also allows the network to ignore the small features in the error surface. The momentum constant is denoted as μ which exists between 0 to 1 range. When the momentum constant is 0, the weight change is performed based only on the gradient. When the momentum constant is 1, the new weight change is set to equal the previous weight change, while simply ignoring the gradient [24, 34]. The new weight vector $\mathbf{w}_{k+1}$ is adjusted as

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha \mathbf{G}_k + \mu \mathbf{w}_{k-1}. \tag{37}$$

5.17 Gradient descent

The gradient descent algorithm [5, 22, 24] updates the weights and biases along the steepest descent direction. The network weights and biases are modified in a direction that reduces the performance function rapidly i.e. the negative of the gradient of the performance function. The updated weights and biases in this algorithm are given by

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \nabla \mathbf{G}_k, \tag{38}$$

where $\mathbf{x}_k$ is the vector of the current weights and biases, α_k is the learning rate and $\nabla \mathbf{G}_k$ is the current gradient of the performance function. Tab II shows the advantages and disadvantages of the NN training algorithms.

S.No	Training algorithms	advantages	disadvantages
1.	Bayesian regularization	•Bayesian regularization expands the cost function to search not only for the minimal error, but also for the minimal error by using the minimal weights. •By using Bayesian regularization, the need for a costly cross validation is avoided. •This also reduces the need for testing the various numbers of hidden neurons for a problem. •Bayesian regularization provides better generalization performance than others, since it does not require a separate validation dataset.	•The main disadvantage of the Bayesian regularization algorithm is that it generally takes more time to converge than early stopping.
2.	Levenberg-Marquardt	•Because of the properties of the fast convergence and stability, this method is employed in many modeling problems.	•One of the main drawbacks of the LM algorithm is that it needs the large storage of some matrices, for certain problems. •LM algorithm is really dependent on the initial predictions for the network parameters. •Based on the initial weights of the network, this algorithm may converge to the local minima or do not converge at all. •The Levenberg-Marquardt is very sensitive to the initial network weights. It also does not consider outliers in the data,which may lead to overfitting noise.

3.	Scaled Conjugate Gradient	•The SCG algorithm avoids the time-consuming line search. •The calculation complexity and memory consumption of the SCG algorithm are also low. •Hence, the speed-up rate of the SCG algorithm is really high.	•The SCG algorithm does not perform the line search during every iteration.
4.	Back propagation	•Back propagation is a very simple and efficient method for computing the gradient in the NN.	•Slow rate of convergence. •The learning time of the back propagation algorithm increases and recall performance reduces, with the increase in the size and complexity of data.
5.	Resilient Back Propagation	•It is generally much faster than the standard steepest descent algorithm. •The memory requirements are also moderate.	•However, the learning speed and convergence rate are moderate for the artificial hyperbolic test functions.
6.	Online back propagation	•The online back propagation algorithm improves the generalization capacity and shows the good convergence speed. •Further, the online back propagation is superior to the batch back propagation, when there is a maximum degree of redundancy in the training data. •This is used for the dynamic environments that provide a continuous stream of data values.	•The computational complexity of the online back propagation algorithm is high.
7.	Batch Back Propagation	•Batch back propagation yields a highly stable descent to the local minimum.	•It requires a much longer time to converge, as it considers the total training error over all the patterns.
8.	Powell/Beale Restarts Conjugate Gradient	•The performance of the Beale-Powell restart algorithm is higher than the conjugate gradient back propagation.	•The storage requirements for the Powell-Beale algorithm are higher than the storage requirements for Polak-Ribiére.
9.	Polak-Ribiere Conjugate Gradient	•Highly effective and suitable for solving large-scale nonsmooth and free convex optimization problems.	•The storage requirements for the Polak-Ribiére are slightly larger for Fletcher-Reeves.
10.	One-Step Secant	•Requires low storage and computation requirements, when compared to the Quasi-Newton algorithm.	•There is no guaranteed error bound for the computed iterations. •The convergence rate is slower than the Newton algorithms.
11.	Limited Memory Quasi-Newton	•The Limited Memory Quasi-Newton method requires low memory and less computational time.	•However, the Limited Memory Quasi-Newton method still needs to solve subproblems at every iteration.

12.	Newton Method	•Newton's method exhibits faster convergence rate than the conjugate gradient approaches. •This method generalizes more easily for solving the simultaneous types of nonlinear equations.	•Unfortunately, it is more complex and really expensive for computing the Hessian matrix for the NNs.
13.	Quasi-Newton	•Quasi-Newton methods are chosen, since it does not require computation of the Hessian matrix. •Quasi-Newton Method is much faster than the steepest decant method. •Convergence rate of this method is superlinear.	•The problem with this approach is the requirement of computation and storage of the approximate Hessian matrix for every iteration.
14.	Variable Learning Rate Gradient Descent	•A near-optimal learning rate is obtained for the local terrain.	•However, this algorithm becomes unstable, if the learning rate is made too large. On the other hand, if the learning rate is set to be too small, the algorithm will take a longer time to converge.
15.	Conjugate Gradient Descent	•The convergance rate is faster than the gradient descent algorithm. •The conjugate gradient algorithms are usually much quicker than the variable learning rate backpropagation.	•The time consuming line search is required during all the iterations of the weight update.
16.	Gradient Descent with Momentum	•Gradient Descent with Momentum yields better prediction accuracy without requiring more training time. •With the usage of momentum, the stability of the algorithm is maintained during the higher learning rate.	•The results are not accurate, due to the low prediction capability.
17.	Gradient Descent	•Gradient descent is much more faster.	•Convergence rate is low. •Not invariant to the linear transformations.

Tab. II Advantages and disadvantages of NN training algorithms.

6. Dataset description

This section describes the dataset used in our work. Tab. III shows the units and descriptions of the target variable. Tab. IV shows the units and descriptions of the input variables and. Tab. V shows the economic based information and Electricity consumption data for 50 years.

The economic information and electricity consumption data of Tamilnadu from 1964 to 2013 are collected from the Tamil Nadu Electricity Board (TNEB) and Department of Economics and Statistical department, Tamilnadu, India. The experiment utilizes a NN with the economic input factors that influence the electric energy consumption as input variables.

Variable	Target Variable	Simple Description	Units
$Y1$	Electricity Consumption	Electricity Consumption is a measure of the consumption rate of electric energy by the consumers during the period of year. It is measured in kwh. The electricity consumption is used as target variable to be forecast in the neural network.	Million Units

Tab. III Units and description of target variable.

The economic input factors that influence the electricity consumption is selected as the input variables. The economic input variables include the following: population, wholesale price index, urban consumer price index, rural consumer price index, gross state domestic product, state per capita income, exports, imports and industries income. Electricity consumption is the target variable or the dependent variable. The input variables are the independent variables that are used to train the NN to obtain the electricity consumption.

Every independent variable is a time series data belonging to particular economic factors. With the multi-layer perceptron, the parameters of NN are modified in such a way to obtain maximum similarity between inputs and outputs for all training data. The training data is fed at the network input. For every pattern, the error between the desired and actual network output is analyzed. Depending on the value of this error, correction of the neural weights is performed. The process is repeated until certain conditions of training termination are achieved. The results are obtained using an optimal architecture with different training algorithm and with a different combination of input and output activation functions. The results are compared based on the lowest relative error and highest correlation coefficient.

Tab. VI shows the correlation coefficient of the input variable to the target variable. The table indicates that all the selected input variables have the highest correlation coefficient with the dependent variable. From the correlation coefficient values, it is confirmed that the selected economic-based input variables influence the dependent variable. So, these input variables are the optimal input variables for the NN model.

Tab. VII illustrates the correlation coefficient for different NN architectures. From the table, it is clearly evident that the NN architecture 9-6-5-1 has the highest correlation coefficient than the other NN architectures. Hence, this NN architecture is chosen and trained by using the best training algorithm for the effective computation of electricity consumption. The electricity consumption is measured in Megawatt [MW]. Over the years, there is a constant increase in the electricity consumption rate.

Variable	Input Variables	Description	Units
$X1$	Population	With the increase in the population growth, consumption of electric energy for the routine activities also increases.	Number of personnel
$X2$	Wholesale price index	The Wholesale Price Index (WPI) represents the special price of goods that are sold in bulk and traded between the organizations, instead of individual consumers.	Index Numbers
$X3$	Consumer price index – Urban	The Consumer Price Index (CPI) measures the changes in the price level of consumer goods and services purchased by the households. It is a comprehensive measure used for approximating the price changes of the commodities.	Index Numbers
$X4$	Consumer price index – Rural		
$X5$	Gross State Domestic Product	The Gross State Domestic Product (GSDP) is a measure of the economic output of the nation. It represents the total amount of productivity of goods and services during a year.	Amount of Money
$X6$	State Per Capita Income	State per capita income is defined as the average income of individuals in a state.	Amount of Money
$X7$	Exports	Export is the process of selling goods and services produced in the home country to the International markets.	Amount of Money
$X8$	Imports	Import is the process of the receiving goods and services from an external producer.	Amount of Money
$X9$	Industrial income	Industrial income is the total amount of income incurred by the industries.	Amount of Money

Tab. IV Units and description of input variables.

7. Results and discussion

This section explains the comparative analysis of the NN training algorithms. Among these training algorithms, the optimal training algorithm is determined.

Year	Population (Number)	Whole sale price index - Number	Consumer price index – Urban -Number	Consumer price index – Rural -Number	Gross State Domestic Product -Lakhs	State Per Capita Income -Lakhs	Export -Lakhs	Import -Lakhs	Industry income -Lakhs
1964	38156286	80.02	24.12	30.12	286566	105	4153	62367	50950
1965	38283967	85.02	25.35	34.35	329567	133	7128	85156	53550
1966	38411647	88.02	42.35	48.35	368800	192	9465	87077	69640
1967	38539327	99.02	50.35	59.35	405707	267	12814	87774	81360
1968	38667007	105.02	64.35	73.35	454014	338	15849	95101	108520
1969	38794687	115.02	76.35	82.35	488921	416	24383	101937	124950
1970	38922367	127.02	84.35	92.35	532374	491	32517	106792	127770
1971	40199168	137.02	97.35	104.35	557097	548	39963	107921	154930
1972	41008997	142.02	108.35	114.35	606603	617	24535	90122	171020
1973	41818826	149.02	123.35	129.35	645436	688	26660	94372	182740
1974	42628654	154.02	135.35	142.35	672222	744	29635	103076	209900
1975	43438483	171.02	143.35	152.35	694027	822	36860	125864	226330
1976	44248312	166.37	161.35	167.35	720810	899	39198	127785	229150
1977	45058141	178.87	170.35	176.35	744859	967	42547	128482	197910
1978	45867970	181.83	179.35	186.35	784955	1022	45581	135809	230670
1979	46677798	199.34	193.35	200.35	824655	1074	54166	142646	242230
1980	47487627	225.31	210.2	219.2	866113	1151	62300	147500	259640
1981	48297456	263.91	251.09	260.09	898394	1350	81500	209400	230930
1982	49031542	277.73	256.95	263.95	918879	1269	89700	224075	255065
1983	49765628	303.41	302.79	311.79	961097	1373	97900	238750	272645
1984	50499715	333.72	314.31	322.31	998466	1376	106100	253425	313385
1985	51233801	346.28	330.07	335.07	1031660	1827	114300	268100	338030
1986	51967887	381.53	357.26	366.26	1063491	2510	141800	290700	342260
1987	52701973	419.16	389.75	399.75	1229030	2719	181200	298900	295400
1988	53436059	451.74	416.34	425.34	1449271	3141	216900	385100	365800
1989	54170146	493.78	455.34	459.45	1703392	3321	317303	465527	486600
1990	54904232	536.16	491.68	498.19	2014567	3688	412993	522638	580500
1991	55638318	613.25	572.27	572.04	2156190	3966	538988	535921	606500
1992	56285570	693.75	643.91	654.55	2484052	4428	688423	731065	767400
1993	56932822	718.76	689.73	695.71	2886822	5237	908580	811180	884600
1994	57580074	779.23	751.65	766.28	3522440	6935	1246009	1254153	1156200
1995	58227326	851.31	826.35	836.67	4793736	7236	1583283	1741672	1320500
1996	58874579	937.9	934.09	935.51	5815175	9954	1759627	1496654	1348700
1997	59521831	1046.06	1091.42	1093	6613382	11215	1652964	3687789	1036300
1998	60169083	1162.23	1165.5	1164.67	8103589	15152	1877128	2348290	1363900
1999	60816335	1199.27	1217.47	1227.47	10525616	17525	2698375	2560016	1479500
2000	61463587	1223.26	1270.23	1280.23	11564416	18786	1173696	1882137	1653600
2001	62110839	1251.77	1302.34	1310.34	13173056	21229	1263124	1862805	1462200
2002	63032640	1270.32	1339.91	1345.91	13091746	20975	2503900	3569700	1510000
2003	63954440	1340.84	1308.79	1359.95	13678087	21738	3306800	4379800	1910100
2004	64876241	1341.00	1333.76	1365.73	14965415	23476	3478200	7874000	2156700
2005	65798042	1373.00	1351.12	1370.34	16718287	25965	5129800	11298300	2798100
2006	66719843	1377.00	1358.37	1377.41	20750283	31920	6775900	15903500	3778400
2007	67641643	1451.00	1425.72	1443.49	24626587	37635	9176200	19016100	3934100
2008	68563444	1539.00	1540.25	1540.23	27928746	36915	9060300	23098700	3971400
2009	69485245	1640.00	1661.45	1668.45	29458192	49831	11309300	28906600	5914300
2010	70407045	1687.00	1778.3	1772.38	33921164	51097	9176200	19016100	7199300
2011	71328846	1798.00	2122.67	2134.67	42491835	70219	9060300	23098700	7695600
2012	71601574	2026.00	2368.72	2492.13	54726662	72993	11309300	28906600	9136400
2013	71874302	2204.00	2623.07	2770.16	63902460	84496	11309300	28906600	9576510

Tab. V Economic based information and electricity consumption data for 50 years.

S. No	Input Variables	Correlation coefficients
1	Population	0.940
2	Wholesale price index	0.982
3	Urban – Consumer price index	0.978
4	Rural – Consumer price index	0.977
5	Gross State Domestic Product	0.916
6	State Per Capita Income	0.875
7	Exports	0.906
8	Imports	0.859
9	Industries income	0.978

Tab. VI Correlation coefficients of the input variables to the target variable.

NN architectures	Correlation coefficient
[9-3-5-1]	0.940274
[9-4-5-1]	0.998368
[9-5-5-1]	0.897638
[9-6-5-1]	0.999982
[9-7-5-1]	0.996277

Tab. VII Correlation Coefficient for different NN architectures.

Alyuda NeuroIntelligence 2.1 software [1] and MATLAB 2013b software [29] are utilized for experimentation. The comparative analysis of the relative error and correlation coefficient of the training algorithms with different combinations of the activation functions is carried out in this section. The dataset involves the electricity consumption data for 50 years. The dataset is partitioned into two sets of data for training and testing. The dataset is partitioned into two sets of data for training and testing. 90% of data is used for training and 10% is used for testing. Tab. VIII shows the comparative analysis of the correlation coefficient of every training algorithm. The comparative analysis of the relative error of each NN training algorithm is shown in the Tab. IX. Tab. X shows the ranking of the NN training algorithms based on relative error and correlation coefficient.

7.1 Relative error

The Relative Error (RE) is obtained by dividing the difference between the actual and the desired output values. It is calculated by using the following equation

$$\mathrm{RE} = \left(\frac{1}{N} \sum_{i}^{N} \frac{|Y_i - Y_p|}{Y_p} \right) \times 100, \tag{39}$$

where Y_i represents the mean value of the NN output, Y_p is the predicted value of the NN output and N represents the number of cases used for the calculations of statistical parameters.

7.2 Correlation coefficient

The correlation coefficient is defined as the degree of the relationship between the input and output variables. The values of the correlation coefficient ranges from −1.0 to 1.0. If the correlation coefficient is +1.0, there is a high positive correlation between the variables. If the correlation coefficient is −1.0, there is a high negative correlation between the variables. If it is equal to 0, it denotes the non-correlation between the variables.

Fig. 7 shows the graph illustrating the comparison between the correlation coefficients for NN training algorithms. The correlation coefficient of the Bayesian Regularization training algorithm is higher than other training algorithms. Fig. 8 shows the comparison graph of the relative error values of every training algorithm. The relative error of the Bayesian Regularization training algorithm is 0.000257, which is the lowest value among the relative error of other training algorithms. Hence, the Bayesian Regularization training algorithm is selected as the optimal training algorithm for the effective computation of the electricity consumption. It creates the computing model to predict the electricity consumption using the nine economical quantities.

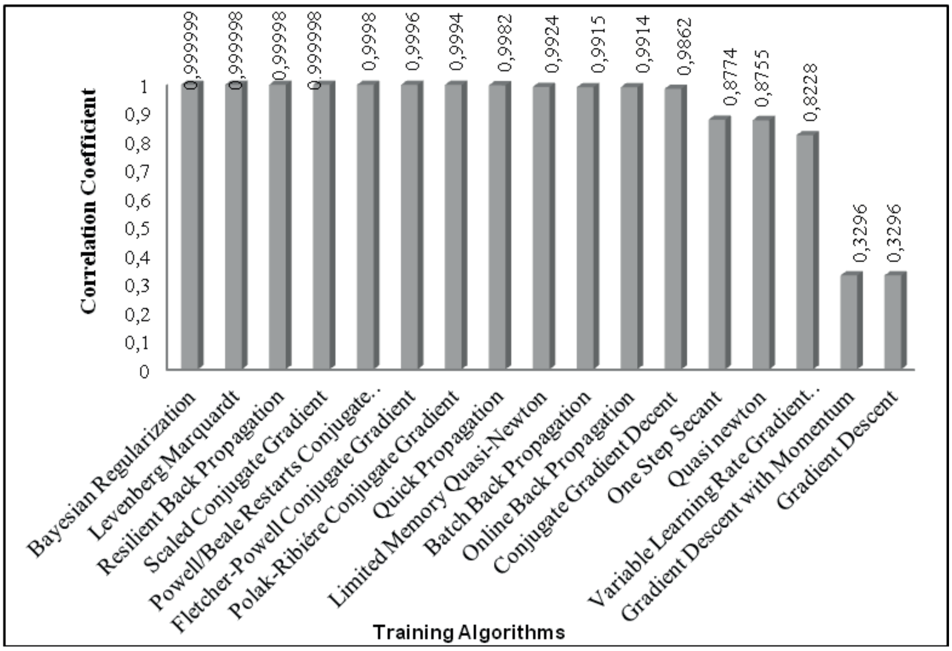

Fig. 7 Correlation coefficient analysis for the NN training algorithms.

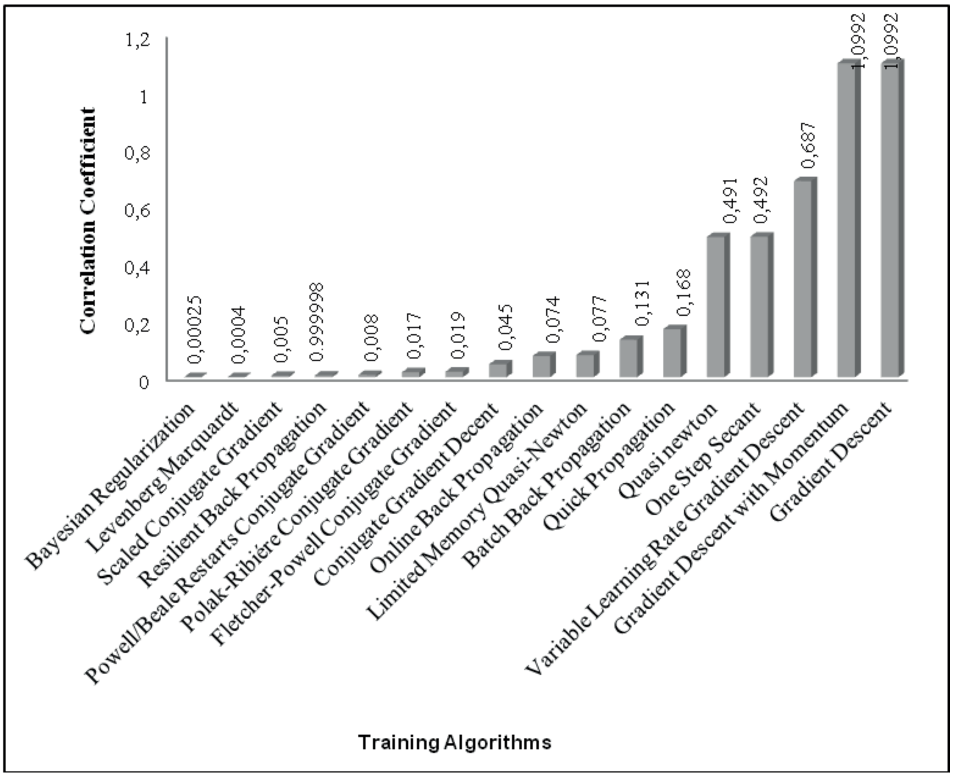

Fig. 8 Relative Error of each training algorithm.

The comparative analysis of correlation coefficient of every NN training algorithm with different combinations of the activation function is shown in Tab. VIII. From the table, it is clearly observed that the correlation coefficient of the Hyperbolic Tangent – Hyperbolic Tangent of the Bayesian Regularization training algorithm is found to be the highest, among all the other activation functions.

The comparative analysis of the relative error of every training algorithm with different combinations of the activation function is depicted in Tab. IX. The relative error of the Hyperbolic Tangent-Hyperbolic Tangent activation function of the Bayesian Regularization training algorithm is found to be the lowest, among all the other activation functions.

This implies that the Bayesian Regularization training algorithm is the optimal training algorithm for the electricity consumption forecasting. The ranking of the training algorithms based on the accuracy parameter is shown in the Tab. X. Based on the lower relative error and higher correlation coefficient in both the training and testing period, the Bayesian Regularization training algorithm obtains the highest rank. The Levenberg Marquardt training algorithm is ranked second. It accurately predicts the electricity demand rate more closely to the actual data. This means that the deviation of the predicted electricity demand rate from the actual demand rate is minimum. Hence, it is concluded that the Bayesian Regularization training algorithm is determined as the optimal training algorithm for the effective computation of electricity consumption.

Training Algorithms	Linear – Linear	Linear – Logistic	Linear – Hyperbolic Tangent	Logistic – Linear	Logistic – Logistic	Logistic – Hyperbolic Tangent	Hyperbolic Tangent – Linear	Hyperbolic Tangent – Logistic	Hyperbolic Tangent – Hyperbolic Tangent
QN	0.9985	0.8656	0.9942	0.9993	0.8565	0.999 96	0.999	0.8011	0.8774
LM-QN	0.9951	0.9975	0.9973	0.993	0.9714	0.9905	0.9947	0.9897	0.9915
OSS	0.9982	0.8667	0.8555	0.9994	0.8565	0.9835	0.9993	0.7411	0.8755
LM	0.9985	0.8656	0.9942	0.9982	0.8603	0.999 97	0.9996	0.8684	0.999 998
BR	0.9987	0.858	0.9943	0.9998	0.8881	0.999 99	0.9998	0.8692	0.999 999
QP	0.9971	0.9878	0.9866	0.9573	0.9818	0.8035	0.9897	0.9983	0.9862
OBP	0.9952	0.9974	0.9966	0.9442	0.9699	0.982	0.9904	0.9987	0.9924
BBP	0.9981	0.9975	0.991	0.9929	0.9713	0.9904	0.9938	0.9889	0.9914
RBP	0.9986	0.8657	0.9943	0.9995	0.8685	0.999 98	0.9992	0.8668	0.9999
CGD	0.9981	0.9982	0.9982	0.9574	0.9982	0.9982	0.9757	0.9897	0.9982
SCG	0.9984	0.8666	0.9945	0.9992	0.9945	0.999 99	0.9987	0.8561	0.999 98
FPCG	0.9979	0.8661	0.9944	0.9988	0.8774	0.9987	0.9989	0.8521	0.9994
PRCG	0.9983	0.8654	0.9943	0.9989	0.8565	0.9954	0.9989	0.8411	0.9996
PBRCG	0.9976	0.8658	0.9942	0.999	0.8774	0.9948	0.9989	0.8566	0.9998
VLRGD	0.9987	0.8865	0.9946	0.9994	0.8565	0.9994	0.9993	0.8041	0.8228
GDM	0.0381	0.2356	0.1831	0.2913	0.2451	0.3296	0.2893	0.2451	0.3296
GD	0.0381	0.2356	0.1831	0.2913	0.2451	0.3296	0.2893	0.2451	0.3296

Tab. VIII Comparative analysis of correlation coefficient of each training algorithm with different combinations of activation functions.

7.3 WEKA time series analysis

After finding an optimal training algorithm, the future electricity consumption is to be forecasted. For further forecasting of electricity consumption, all 9 economical quantities are needed. Since all the variables are time dependent, a Waikato Environment for Knowledge Analysis-version (WEKA) based time series forecasting can be done for every input variable. A WEKA-version 3.7.6 (WEKA 3.7.6) tool [16] is used for time series forecasting. The time series forecasting is reliable for the data that represents the long-time predictions. It can be used easily as the historical observations are readily available from secondary sources. These successive observations are statistically dependent. The time series forecasting is concerned with the analysis of statistical dependencies.

Training Algorithms	Linear – Linear	Linear – Logistic	Linear – Hyperbolic Tangent	Logistic – Linear	Logistic – Logistic	Logistic – Hyperbolic Tangent	Hyperbolic Tangent – Linear	Hyperbolic Tangent – Logistic	Hyperbolic Tangent – Hyperbolic Tangent
QN	0.038	2.075	0.498	0.029	2.143	0.211	0.036	2.143	0.491
LM-QN	0.031	1.848	0.071	0.427	1.732	0.058	0.094	1.922	0.077
OSS	0.045	2.076	0.183	0.035	2.143	0.009	0.034	2.143	0.492
LM	0.046	2.075	0.182	0.062	2.081	0.001	0.033	2.073	0.0004
BR	0.047	2.091	0.184	0.0163	2.084	0.000 27	0.016	2.089	0.000 25
QP	0.043	1.715	0.101	0.099	1.418	0.086	0.377	1.133	0.168
OBP	0.033	1.814	0.08	0.08	1.132	0.111	0.046	1.915	0.074
BBP	0.049	1.939	0.087	0.08	1.232	0.111	0.07	1.769	0.131
RBP	0.047	2.076	0.189	0.048	0.189	0.004	0.051	2.079	0.006
CGD	0.056	1.812	0.09	0.32	1.348	0.201	0.088	1.832	0.045
SCG	0.049	2.075	0.185	0.029	2.073	0.003	0.036	2.073	0.005
FPCG	0.037	2.076	0.184	0.048	2.077	0.116	0.041	2.082	0.019
PRCG	0.044	2.076	0.186	0.047	2.074	0.037	0.041	2.09	0.017
PBRCG	0.044	2.076	0.182	0.044	2.074	0.132	0.042	2.078	0.008
VLRGD	0.046	2.087	0.192	0.031	2.143	0.038	0.036	2.143	0.687
GDM	22.216	2.1031	3.7297	2.0783	2.1431	1.0992	1.9652	2.1431	1.0992
GD	22.216	2.1031	3.7297	2.0783	2.1431	1.0992	1.9652	2.1431	1.0992

Tab. IX Comparative analysis of relative error of each training algorithm with different combinations of activation functions.

In the time series forecasting, the prediction of the future values does not depend on the explanatory variables that affect the system. As the collection of information on the explanatory variables is highly cumbersome, there is a problem in the availability of long-term data. In such conditions, the time series model is a boon for forecasting.

Tab. XI shows the Forecasting of input variable using WEKA Time Series Analysis for the next 15 years. Time series forecasting uses a model to predict the future values based on previously observed values. The forecasted time series data is applied to the computational model to predict the electricity consumption of next 15 years. Tab. XII shows the prediction of the electricity consumption using the Bayesian Regularization training algorithm. The electricity consumption is measured in Megawatt (MW). Fig. 9 shows the predicted electricity consumption using Bayesian regularization training algorithm for the 2014-2028. Fig. 10 shows the actual and predicted electricity consumption. From the graph, it is observed that there is a linear increase in the acutal and predicted electricity consumption from the year 1964 to 2028.

Ranking of training algorithm	Training algorithms	Hidden layer's activation functions	Output layer's activation function	Relative error for train data	Relative error for test data	Relative error	Correlation coefficient
Rank 1	BR	Hyp. Tan.	Hyp. Tan.	0.000 305	0.000004	0.000 25	0.999 999
Rank 2	LM	Hyp. Tan.	Hyp. Tan.	0.000 346	0.00069	0.000 40	0.999 998
Rank 3	SCG	Logistic	Hyp. Tan.	0.002 78	0.0024	0.002 68	0.999 984
Rank 4	RBP	Logistic	Hyp. Tan.	0.005 28	0.0008	0.004 11	0.999 994
Rank 5	PBRCG	Hyp. Tan.	Hyp. Tan.	0.007 49	0.00833	0.007 71	0.999 803
Rank 6	OSS	Logistic	Hyp. Tan.	0.0110	0.00289	0.008 86	0.999 962
Rank 7	PRCG	Hyp. Tan.	Hyp. Tan.	0.0170	0.0155	0.016 58	0.999 436
Rank 8	FPCG	Hyp. Tan.	Hyp. Tan.	0.0204	0.0139	0.018 73	0.999 626
Rank 9	LM-QN	Hyp. Tan.	Logistic	0.0192	0.0286	0.021 64	0.998 668
Rank 10	QN	Logistic	Linear	0.0337	0.0140	0.028 57	0.999 434
Rank 11	VLRGD	Logistic	Linear	0.0372	0.0135	0.031 03	0.999 436
Rank 12	CGD	Hyp. Tan.	Logistic	0.0221	0.0620	0.032 48	0.998 281
Rank 13	QP	Hyp. Tan.	Logistic	0.0198	0.0721	0.033 41	0.989 689
Rank 14	OBP	Hyp. Tan.	Linear	1.0063	0.0647	0.045 52	0.993 832
Rank 15	BBP	Hyp. Tan.	Linear	0.0731	0.0597	0.069 64	0.994 668
Rank 16	GDM	Hyp. Tan.	Hyp. Tan.	1.2715	0.6087	1.099	0.329 695
Rank 17	GD	Hyp. Tan.	Hyp. Tan.	1.2715	0.6087	1.099	0.329 695

Tab. X Ranking of training algorithms based on relative error & correlation coefficient.

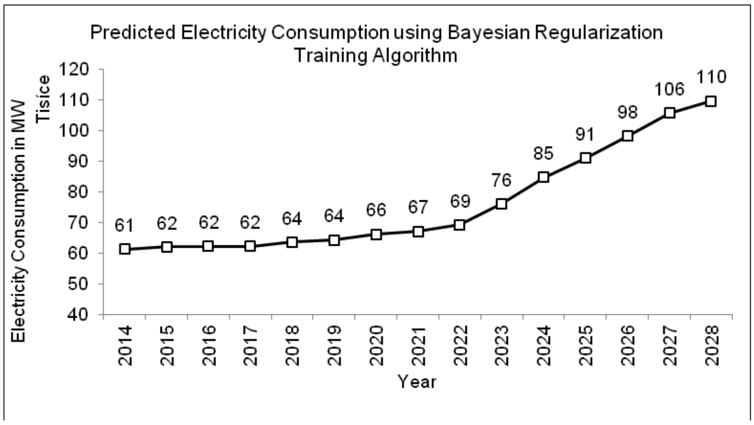

Fig. 9 Predicted electricity consumption using Bayesian regularization training algorithm.

Year	Population (Number)	Wholesale price index (Number)	Consumer price index – Urban (Number)	Consumer price index – Rural (Number)	Gross State Domestic Product (Lakhs)	State Per Capita Income (Lakhs)	Exports (Lakhs)	Imports (Lakhs)	Industries income (Lakhs)
2014	72147030	2183	2957	3173	74447610	98567	11309	28906	9847770
2015	72419758	2684	3071	3191	77467807	102819	12139	30555	10274124
2016	72692486	2811	3223	3223	81699163	117332	13623	31673	10356816
2017	72965214	2958	3387	3387	87547989	128779	15495	32381	10429978
2018	73237942	3145	3567	3567	92735709	139659	18147	33135	10597092
2019	73510670	3437	3888	3888	99681483	155275	22796	35663	10854890
2020	73783398	3686	4160	4160	104527870	166557	26884	38114	11103548
2021	74056126	4115	4658	4658	112559964	177413	33845	41549	11520636
2022	74509893	4937	5544	5544	132939225	197733	51681	48825	12721422
2023	74963660	5500	6159	6159	153340199	210521	72534	55944	13956157
2024	75417427	6552	7077	7377	159871549	233739	113464	71944	15720542
2025	75871194	7247	8357	8357	162012257	256974	143161	103492	18268664
2026	76324961	7962	9909	9909	173860266	269228	160474	166963	19278201
2027	76778728	8798	10716	10716	198558032	298993	184480	184402	21719262
2028	77232495	9030	11218	11238	206584178	319437	198051	214918	22374716

Tab. XI Forecasting of input variable using WEKA Time Series Analysis.

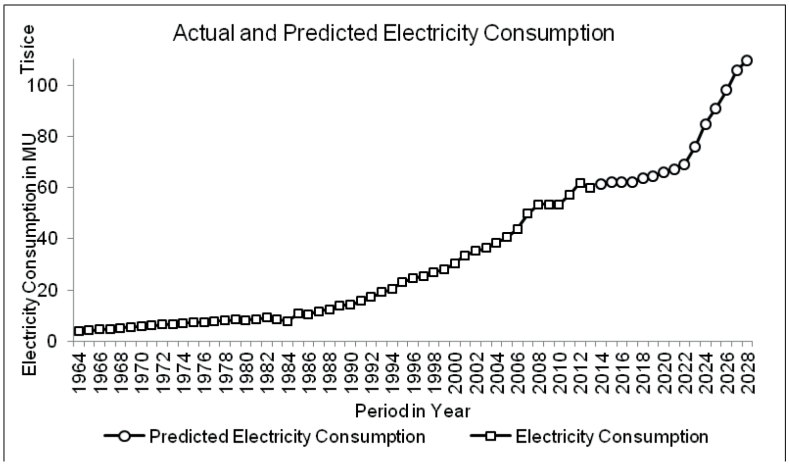

Fig. 10 Actual and Predicted electricity consumption.

Year	Predicted Electricity Consumption [MW]
2014	61,210
2015	61,980
2016	62,100
2017	62,200
2018	63,512
2019	64,248
2020	66,126
2021	67,086
2022	69,156
2023	76,066
2024	84,757
2025	91,009
2026	98,110
2027	105,645
2028	109,645

Tab. XII Prediction of electricity consumption using Bayesian Regularization Training Algorithm.

7.4 Complexity analysis

The complexity of the NN training algorithms is analyzed by using the training speed, number of iterations and computation time. Tab. VIII shows the computation time of every NN training algorithm. The Bayesian Regularization algorithm requires minimum number of iterations and computation time than other training algorithms.

7.5 Evaluation using larger dataset

Different size of datasets such as Energy efficiency dataset [3], Tamilnadu Electricity Board Hourly Readings [21], Individual household electric power consumption dataset [15] are collected from UCI Repository and the Electric bill data [30] is collected from American Statistical Association. They are applied for validating the performance of all NN training algorithms. The missing data is present in an input variable and replaced with the average value of previous and next value of the corresponding input variable. Tab. XIV illustrates the descriptions of UCI and American Statistical Association datasets.

Energy Efficient Dataset comprises of 768 samples and 8 features, to predict two real valued responses such as heating load and cooling load. The features are relative compactness, surface area, wall area, roof area, overall height, orientation, glazing area and glazing area distribution. Tamilnadu Electricity Board Hourly Readings dataset involves the real time readings to find the accuracy consumption of electricity per hour in Tamilnadu around the Thanjavur district. This data is obtained for the residential, commercial, industrial and agriculture applica-

	Metric		
Training Algorithm	Training Speed [iterations/s]	Number of Iterations	Computation Time [s]
QN	107.03	3746	35
LM-QN	485.45	30001	61.8
OSS	81.12	10059	124
LM	52.95	1006	19
BR	51.28	923	18
QP	3409.2	30001	8.8
OBP	3000.1	30001	10
BBP	3658.65	30001	8.2
RBP	358.57	10040	28
CGD	187.5	30001	160.01
SCG	265.42	10086	38
FPCG	114	114	1
PRCG	143	429	3
PBRCG	141	141	1
VLRGD	549.1	11531	21
GDM	5	23	4.6
GD	5	21	4.2

Tab. XIII Computation time of each NN training algorithm.

tions. Individual household electric power consumption dataset contains 2075259 electric power consumption measurements collected in one household with a one-minute sampling rate between December 2006 and November 2010. Electric Bill dataset involves the household electric bill amount and electricity consumption on a monthly basis presented as a time series for January 1991 through December 2000.

The optimal splitting proportion is highly important for successful training. Here, the small size dataset is split into 90% for training and 10% for testing. The medium size dataset is split into 70% for training and 30% for testing. The larger size dataset is split into 60% for training and 40% for testing. The relative error and correlation in the prediction of electricity consumption on different size of datasets are calculated using all the NN training algorithms. The results are compared to analyze the prediction accuracy for different size of datasets.

Tab. XV shows the relative error of the NN training algorithms for different sizes of datasets. Fig. 11 depicts the comparison of the relative error for different datasets. The presence of instances and attributes result in the increase in the relative error. The prediction accuracy decreases with the increase in the size of the dataset. From the Tab. XIV, the number of attributes in the proposed econometric variable dataset is lesser than other datasets. Hence, the relative error of the proposed econometric variable dataset is low and prediction accuracy is high. From the Tab. XV it is observed that the Bayesian regularization training algorithm yields lower relative error than other training algorithms.

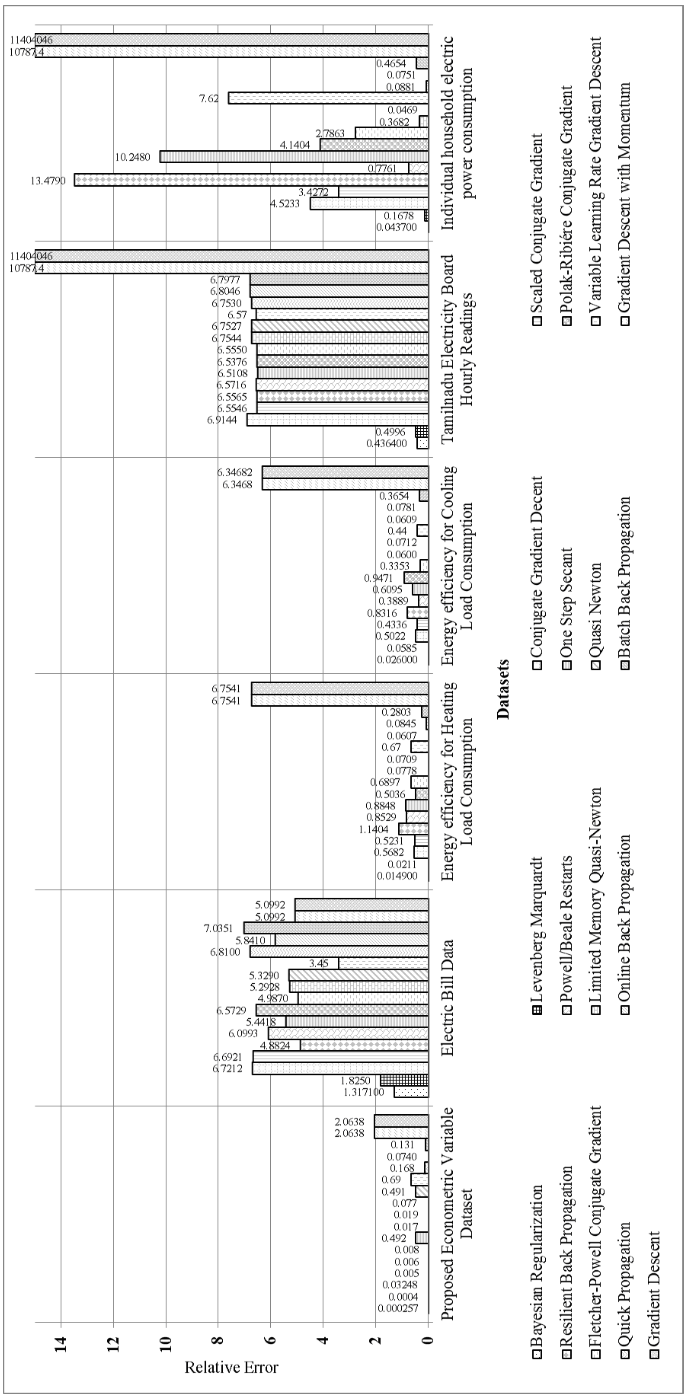

Fig. 11 Comparison of the relative error for different datasets.

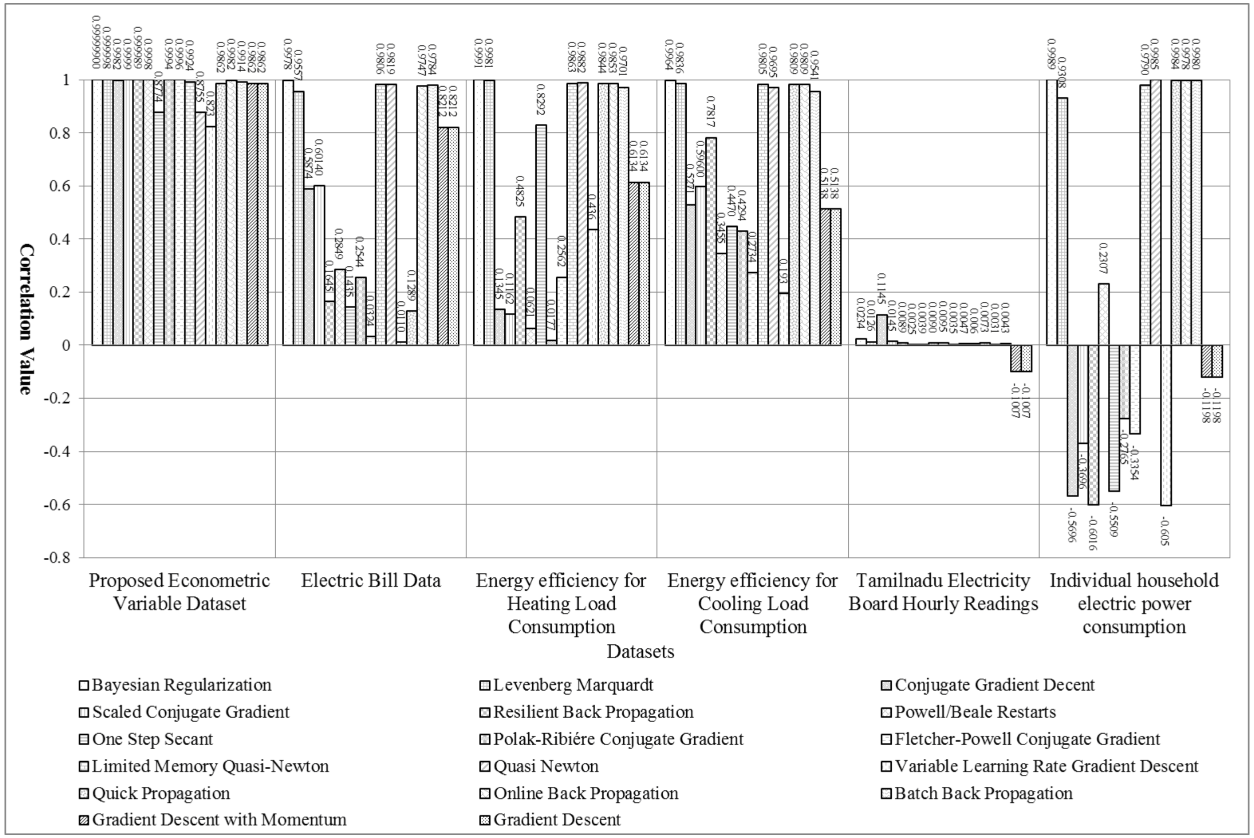

Fig. 12 Comparison of correlation value for different datasets.

Data Source	Name of the Dataset	No. of Instance	No. of Attribute	Response Variable	Data Type
TNEB and Economics & Statistical Deparment	Proposed Econometric Variable Dataset	50	9	Electricity consumption	Real & Integer
American Statistical Association	Electric Bill Data	120	11	Calculated consumption	Real & Integer
UCI	Energy efficiency Data for Heating Load Consumption	768	8	Heating Load consumption	Categorical &, Real
UCI	Energy efficiency Data for Cooling Load Consumption	768	8	Cooling Load consumption	Categorical & Real
UCI	Tamilnadu Electricity Board Hourly Readings	45781	5	Electricity consumption	Categorical & Real
UCI	Individual household electric power consumption	2075259	9	Global active power	Real

Tab. XIV Descriptions of UCI and American Statistical Association Datasets.

Tab. XVI shows the correlation value of the NN training algorithms for different sizes of datasets. Fig. 12 depicts the comparison of the correlation value for different datasets. From the comparative analysis, it is observed that the correlation value of the Bayesian Regularization training algorithm is higher than other training algorithms.

8. Conclusions and future work

The conclusion and future implementation of this survey are discussed in this section. The objective of this survey is to determine the optimal NN training algorithm for achieving effective prediction using multiple time series data. This paper has analyzed a number of NN training algorithms to identify the optimal algorithm. The accuracy parameters such as relative error and correlation coefficient of the training algorithms have been calculated. Various training algorithms are compared to find out the optimal model for Tamilnadu electricity consumption forecasting. The results show that the Bayesian regularization training algorithm with a hyperbolic tangent activation function is proved to be the optimal NN training algorithm. It effectively reduces the relative error to 0.000257 and achieves a high correlation coefficient of 0.99999993. Due to the reduction in the relative error and improved correlation coefficient, the deviation from the actual electricity consumption is minimized. Hence, the Bayesian Regularization training algorithm is selected as the optimal training algorithm for the effective computation of electricity consump-

	Dataset					
Training Algorithm	Proposed Econometric Variable Dataset	Electric Bill Data	Energy efficiency for Heating Load Consumption	Energy efficiency for Cooling Load Consumption	Tamilnadu Electricity Board Hourly Readings	Individual household electric power consumption
BR	0.000 257	1.3171	0.0149	0.0260	0.4364	0.0437
LM	0.0004	1.8250	0.0211	0.0585	0.4996	0.1678
CGD	0.032 48	6.7212	0.5682	0.5022	6.9144	4.5233
SCG	0.005	6.6921	0.5231	0.4336	6.5546	3.4272
RBP	0.006	4.8824	1.1404	0.8316	6.5565	13.479
PBRCG	0.008	6.0993	0.8529	0.3889	6.5716	0.7761
OSS	0.492	5.4418	0.8848	0.6095	6.5108	10.248
PRCG	0.017	6.5729	0.5036	0.9471	6.5376	4.1404
FPCG	0.019	4.9870	0.6897	0.3353	6.5550	2.7863
LM-QN	0.077	5.2928	0.0778	0.0600	6.7544	0.3682
QN	0.491	5.3290	0.0709	0.0712	6.7527	0.0469
VLRCG	0.687	3.4490	0.6695	0.4409	6.5655	7.6193
QP	0.168	6.8100	0.0607	0.0609	6.7530	0.0881
OBP	0.074	5.8410	0.0845	0.0781	6.8046	0.0751
BBP	0.131	7.0351	0.2803	0.3654	6.7977	0.4654
GDM	2.0638	5.0992	6.7541	6.3468	10 787.4	11 404 046
GD	2.0638	5.0992	6.7541	6.346 82	10 787.4	11 404 046

Tab. XV Relative error of NN training algorithms for different sizes of datasets.

tion. It creates the computing model to find the electricity consumption using nine economical quantities. The flexibility of this method is the capacity for using the non-static parameters. The convenience of this method is the insertion of new data and this will continuously improve the estimation process. This approach combines the observed data and the unknown parameters for providing the posterior information. Finally, the economic input attributes are forecasted for the next 15 years using time series forecasting. Using these forecasted economic attributes and with the optimal Bayesian Regularization training algorithm, the electricity consumption for the next 15 years is predicted. The comparative analysis of the NN training algorithms for the proposed dataset and larger datasets obtained from the UCI repository and American Statistical Association shows that the Bayesian regularization training algorithm yields higher correlation value and lower relative error when compared to other training algorithms. The main intention of this future work is to increase the accuracy of forecasting and examining the application of neural networks for the modeling and forecasting of the electricity demand.

	Dataset					
Training Algorithm	Proposed Econometric Variable Dataset	Electric Bill Data	Energy efficiency for Heating Load Consumption	Energy efficiency for Cooling Load Consumption	Tamilnadu Electricity Board Hourly Readings	Individual household electric power consumption
BR	0.999 999	0.9978	0.9991	0.9964	0.0234	0.9989
LM	0.999 998	0.9557	0.9981	0.9836	0.0126	0.9308
CGD	0.9982	0.5874	0.1345	0.5271	0.1145	−0.5696
SCG	0.9999	0.6014	0.1162	0.5960	0.0145	−0.3696
RBP	0.999 989	0.1645	0.4825	0.7817	0.0089	−0.6016
PBRCG	0.9998	0.2849	0.0621	0.3455	0.0025	0.2307
OSS	0.8774	0.1435	0.8292	0.4470	0.0039	−0.5509
PRCG	0.9994	0.2544	0.0177	0.4294	0.0090	−0.2765
FPCG	0.9996	0.0324	0.2562	0.2734	0.0095	−0.3354
LM-QN	0.9924	0.9806	0.9863	0.9805	0.0035	0.9790
QN	0.8755	0.9819	0.9882	0.9695	0.0047	0.9985
VLRCG	0.8228	0.0106	0.4356	0.1933	0.0062	−0.6048
QP	0.9862	0.1289	0.9844	0.9809	0.0073	0.9984
OBP	0.9982	0.9747	0.9853	0.9809	0.0031	0.9978
BBP	0.9914	0.9784	0.9701	0.9541	0.0043	0.9980
GDM	0.9862	0.8212	0.6134	0.5138	−0.1007	−0.1198
GD	0.9862	0.8212	0.6134	0.5138	−0.1007	−0.1198

Tab. XVI Correlation value of NN training algorithms for different sizes of datasets.

Acknowledgement

We would like to thank the reviewers for their insightful comments on the paper, as these comments led us to an improvement of the work. We would also like to thank the Tamil Nadu Electricity Board and the Economics and Statistics Department, Tamilnadu, India for the provision of information and support, without which this research paper would not have been possible.

References

[1] ALYUDA NEUROINTELLIGENCE [Software]. Available from: http://www.alyuda.com/.

[2] AN N., ZHAO W., WANG J., SHANG D., ZHAO E. Using multi-output feedforward neural network with empirical mode decomposition based signal filtering for electricity demand forecasting. *Energy*. 2013, 49, pp. 279–288, doi: 10.1016/j.energy.2012.10.035.

[3] ANGELIKI X. *UCI Machine Learning Repository Energy efficiency Data Set 2012*. Available from: http://archive.ics.uci.edu/ml/datasets/Energy$+$efficiency.

[4] BUHARI M., ADAMU S.S. Short-term load forecasting using artificial neural network. *Proceedings of the IMECS*. 2012.

[5] CHANG P-C., FAN C-Y., LIN J-J. Monthly electricity demand forecasting based on a weighted evolving fuzzy neural network approach. *International Journal of Electrical Power & Energy Systems*. 2011, 33(1), pp. 17–27, doi: 10.1016/j.ijepes.2010.08.008.

[6] CHEN T. A collaborative fuzzy-neural approach for long-term load forecasting in Taiwan. *Computers & Industrial Engineering*. 2012, 63(3), pp. 663–670, doi: 10.1016/j.cie.2011.06.003.

[7] CHITSAZ H., SHAKER H., ZAREIPOUR H., WOOD D., AMJADY N. Short-term electricity load forecasting of buildings in microgrids. *Energy and Buildings*. 2015, 99, pp. 50–60, doi: 10.1016/j.enbuild.2015.04.011.

[8] DILAVER Z., HUNT L.C. Modelling and forecasting Turkish residential electricity demand. *Energy Policy*. 2011, 39(6), pp. 3117–3127, doi: 10.1016/j.enpol.2011.02.059.

[9] DILAVER Z., HUNT L.C. Turkish aggregate electricity demand: an outlook to 2020. *Energy*. 2011, 36(11), pp. 6686–6696, doi: 10.1016/j.energy.2011.07.043.

[10] DORDONNAT V., KOOPMAN S.J., OOMS M. Dynamic factors in periodic time-varying regressions with an application to hourly electricity load modelling. *Computational Statistics & Data Analysis*. 2012, 56(11), pp. 3134–3152, doi: 10.1016/j.csda.2011.04.002.

[11] DUDEK G., ed. Forecasting time series with multiple seasonal cycles using neural networks with local learning. *Artificial Intelligence and Soft Computing*. 2013, pp. 52–63, doi: 10.1007/978-3-642-38658-9_5.

[12] ERWAY J.B., MARCIA R.F. On efficiently computing the eigenvalues of limited-memory quasi-newton matrices. *SIAM Journal on Matrix Analysis and Applications*. 2015, 36(3), pp. 1338–1359, doi: 10.1137/140997737.

[13] FAN S., HYNDMAN R.J. Short-term load forecasting based on a semi-parametric additive model. *IEEE Transactions on Power Systems*. 2012, 27(1), pp. 134–141, doi: 10.1109/TPWRS.2011.2162082.

[14] FRINCU M., CHELMIS C., NOOR M.U., PRASANNA V., eds. Accurate and efficient selection of the best consumption prediction method in smart grids. *Proceedings of the IEEE international conference on Big Data*, 2014, pp. 721–729, doi: 10.1109/BigData.2014.7004296.

[15] HÉBRAIL G., BÉRARD A. Individual household electric power consumption Data Set. UCI Machine Learning Repository, 2012. Available from: https://archive.ics.uci.edu/ml/datasets/Individual$+$household$+$electric$+$power$+$consumption.

[16] HALL M. Time Series Analysis and Forecasting with Weka-Pentaho Data Mining [Software]. [Accessed: 10-Aug-2013]. Available from: http://wiki.pentaho.com/display/DATAMINING/Time$+$Series$+$Analysis$+$and$+$Forecasting$+$with$+$Weka.

[17] HASSAN S., KHOSRAVI A., JAAFAR J. Examining performance of aggregation algorithms for neural network-based electricity demand forecasting. *International Journal of Electrical Power & Energy Systems*. 2015, 64, pp. 1098–105, doi: 10.1016/j.ijepes.2014.08.025.

[18] HERNANDEZ L., BALADRON C., AGUIAR J.M., CARRO B., SANCHEZ-ESGUEVILLAS A.J., LLORET J. A survey on electric power demand forecasting: Future trends in smart grids, microgrids and smart buildings. *IEEE Communications Surveys & Tutorials*. 2014, 16(3), pp. 1460–1495, doi: 10.1109/SURV.2014.032014.00094.

[19] HERNANDEZ L., BALADRON C., AGUIAR J.M., CARRO B., SANCHEZ-ESGUEVILLAS A.J., LLORET J. A multi-agent system architecture for smart grid management and forecasting of energy demand in virtual power plants. *IEEE Communications Magazine*. 2013, 51(1), pp. 106–113, doi: 10.1109/MCOM.2013.6400446.

[20] JARAMILLO-MORÁN M.A., GONZÁLEZ-ROMERA E., CARMONA-FERNÁNDEZ D. Monthly electric demand forecasting with neural filters. *International Journal of Electrical Power & Energy Systems*. 2013, 49, pp. 253–263, doi: 10.1109/MCOM.2013.6400446.

[21] KALYANI K. Tamilnadu Electricity Board Hourly Readings Data Set Available from: https://archive.ics.uci.edu/ml/datasets/Tamilnadu$+$Electricity$+$Board$+$Hourly$+$Readings.

[22] KAYTEZ F., TAPLAMACIOGLU M.C., CAM E., HARDALAC F. Forecasting electricity consumption: A comparison of regression analysis, neural networks and least squares support vector machines. *International Journal of Electrical Power & Energy Systems.* 2015, 67, pp. 431–438, doi: 10.1016/j.ijepes.2014.12.036.

[23] KELO S., DUDUL S. A wavelet Elman neural network for short-term electrical load prediction under the influence of temperature. *International Journal of Electrical Power & Energy Systems.* 2012, 43(1), pp. 1063–1071, doi: 10.1016/j.ijepes.2012.06.009.

[24] KHEIRKHAH A., AZADEH A., SABERI M., AZARON A., SHAKOURI H. Improved estimation of electricity demand function by using of artificial neural network, principal component analysis and data envelopment analysis. *Computers & Industrial Engineering.* 2013, 64(1), pp. 425–441, doi: 10.1016/j.cie.2012.09.017.

[25] KIRAN M.S., ÖZCEYLAN E., GÜNDÜZ M., PAKSOY T. A novel hybrid approach based on particle swarm optimization and ant colony algorithm to forecast energy demand of Turkey. *Energy conversion and management.* 2012, 53(1), pp. 75–83, doi: 10.1016/j.enconman.2011.08.004.

[26] LIU Z., LI W., SUN W. A novel method of short-term load forecasting based on multiwavelet transform and multiple neural networks. *Neural Computing and Applications.* 2013, 22(2), pp. 271–277, doi: 10.1007/s00521-011-0715-2.

[27] LLANOS J., SÁEZ D., PALMA-BEHNKE R., JIMÉNEZ-ESTÉVEZ G., eds. Load profile generator and load forecasting for a renewable based microgrid using self organizing maps and neural networks. *Proceedings of International Joint Conference on Neural Networks.* 2012, pp. 1–8, doi: 10.1109/IJCNN.2012.6252648.

[28] MARINESCU A., HARRIS C., DUSPARIC I., CLARKE S., CAHILL V., eds. Residential electrical demand forecasting in very small scale: An evaluation of forecasting methods. *Proceedings of 2nd International Workshop on Software Engineering Challenges for the Smart Grid (SE4SG),* 2013, pp. 25–32, doi: 10.1109/SE4SG.2013.6596108.

[29] MATLAB Release 2013b [Software]. The MathWorks, Inc., Natick, Massachusetts, United States. https://in.mathworks.com/products/new_products/release2013b.html.

[30] MCLAREN C., MCLAREN B. *Electric Bill Data.* Available from: http://www.amstat.org/publications/jse/datasets/electricbill.txt.

[31] MELLIT A., PAVAN A.M., LUGHI V. Short-term forecasting of power production in a large-scale photovoltaic plant. *Solar Energy.* 2014, 105, pp. 401–413, doi: 10.1016/j.solener.2014.03.018.

[32] MOTAMEDI A., ZAREIPOUR H., ROSEHART W.D. Electricity price and demand forecasting in smart grids. *IEEE Transactions on Smart Grid.* 2012, 3(2), pp. 664–674, doi: 10.1109/TSG.2011.2171046.

[33] MURUGADOSS R., RAMAKRISHNAN M., editors. Universal approximation using probabilistic neural networks with sigmoid activation functions. *Proceedings of International Conference on Advances in Engineering and Technology Research (ICAETR),* 2014, pp. 1–4, doi: 10.1109/ICAETR.2014.7012920.

[34] NAYAK S, CHOUDHURY B, LENKA SK, eds. Gradient Descent with Momentum Based Backpropagation Neural Network for Selection of Industrial Robot. *Proceedings of First International Conference on Information and Communication Technology for Intelligent Systems.* 2016, 1, pp. 487–496, doi: 10.1007/978-3-319-30933-0_49.

[35] NIU D-X., SHI H-F., WU D.D. Short-term load forecasting using bayesian neural networks learned by Hybrid Monte Carlo algorithm. *Applied Soft Computing.* 2012, 12(6), pp. 1822–1827. doi: 10.1016/j.asoc.2011.07.001.

[36] RAHPEYMAII F., KIMIAEI M., BAGHERI A. A limited memory quasi-Newton trust-region method for box constrained optimization. Journal of Computational and Applied Mathematics. 2016, 303, pp. 105–118, doi: 10.1016/j.cam.2016.02.026.

[37] RAZA M.Q., KHOSRAVI A. A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings. *Renewable and Sustainable Energy Reviews*. 2015, 50, pp. 1352–1372, doi: 10.1016/j.rser.2015.04.065.

[38] SANDHU P.S., CHHABRA S. A comparative analysis of Conjugate Gradient Algorithms & PSO Based Neural Network approaches for reusability evaluation of procedure based software systems. *Chiang Mai j Sci.* 2011, 38(2), pp. 123–35, Available from: http://www.thaiscience.info/journals/Article/CMJS/10905403.pdf.

[39] SHAO Z., GAO F., YANG S-L., YU B-G. A new semiparametric and EEMD based framework for mid-term electricity demand forecasting in China: Hidden characteristic extraction and probability density prediction. *Renewable and Sustainable Energy Reviews*. 2015, 52, pp. 876–89, doi: 10.1016/j.rser.2015.07.159.

[40] SHEN W., BABUSHKIN V., AUNG Z., WOON W.L., editors. An ensemble model for day-ahead electricity demand time series forecasting. *Proceedings of the fourth international conference on Future energy systems*, ACM, 2013, pp. 51–62, doi: 10.1145/2487166.2487173.

[41] SIGAUKE C., CHIKOBVU D. Prediction of daily peak electricity demand in South Africa using volatility forecasting models. *Energy Economics*. 2011, 33(5), pp. 882–888, doi: 10.1016/j.eneco.2011.02.013.

[42] SOUSA J., NEVES L., JORGE H. Assessing the relevance of load profiling information in electrical load forecasting based on neural network models. *International Journal of Electrical Power & Energy Systems*. 2012, 40(1), pp. 85–93. doi: 10.1016/j.ijepes.2012.02.008.

[43] SUGANTHI L., SAMUEL A.A. Energy models for demand forecasting—A review. *Renewable and sustainable energy reviews*. 2012, 16(2), pp. 1223–1240. doi: 10.1016/j.rser.2011.08.014.

[44] VILAR J.M., CAO R., ANEIROS G. Forecasting next-day electricity demand and price using nonparametric functional methods. *International Journal of Electrical Power & Energy Systems*. 2012, 39(1), pp. 48–55, doi: 10.1016/j.ijepes.2012.01.004.

[45] WANG C-H., GROZEV G., SEO S. Decomposition and statistical analysis for regional electricity demand forecasting. *Energy*. 2012, 41(1), pp. 313–25, doi: 10.1016/j.energy.2012.03.011.

[46] WANG J., CHI D., WU J., LU H-Y. Chaotic time series method combined with particle swarm optimization and trend adjustment for electricity demand forecasting. *Expert Systems with Applications*. 2011, 38(7), pp. 8419–8429, doi: 10.1016/j.eswa.2011.01.037.

[47] WANG J., MA X., WU J., DONG Y. Optimization models based on GM (1, 1) and seasonal fluctuation for electricity demand forecasting. *International Journal of Electrical Power & Energy Systems*. 2012, 43(1), pp. 109–117, doi: 10.1016/j.ijepes.2012.04.027.

[48] WANG L., ZENG Y., CHEN T. Back propagation neural network with adaptive differential evolution algorithm for time series forecasting. *Expert Systems with Applications*. 2015, 42(2), pp. 855–863, doi: 10.1016/j.eswa.2014.08.018.

[49] WANG Y., WANG J., ZHAO G., DONG Y. Application of residual modification approach in seasonal ARIMA for electricity demand forecasting: a case study of China. *Energy Policy*. 2012, 48, pp. 284–294, doi: 10.1016/j.enpol.2012.05.026.

[50] WEBBERLEY A., GAO DW. Study of artificial neural network based short term load forecasting. *IEEE Power and Energy Society General Meeting (PES)*. 2013, pp. 1–4, doi: 10.1109/PESMG.2013.6673036.

[51] YOKOYAMA J., CHIANG H-D. Short Term Load Forecasting improved by ensemble and its variations. *IEEE Power and Energy Society General Meeting*. 2012, pp. 1–6, doi: 10.1109/PESGM.2012.6345222.

[52] YU S-W., ZHU K-J. A hybrid procedure for energy demand forecasting in China. *Energy*. 2012, 37(1), pp. 396–404, doi: 10.1016/j.energy.2011.11.015.

[53] ZAHEDI G., AZIZI S., BAHADORI A., ELKAMEL A., ALWI SRW. Electricity demand estimation using an adaptive neuro-fuzzy network: a case study from the Ontario province–Canada. *Energy*. 2013, 49, pp. 323–328, doi: 10.1016/j.energy.2012.10.019.

[54] ZHANG H., WU W., YAO M. Boundedness and convergence of batch back-propagation algorithm with penalty for feedforward neural networks. *Neurocomputing*. 2012, 89, pp. 141–146. doi: 10.1016/j.neucom.2012.02.029.

[55] ZHU S., WANG J., ZHAO W., WANG J. A seasonal hybrid procedure for electricity demand forecasting in China. *Applied Energy*. 2011, 88(11), pp. 3807–3815, doi: 10.1016/j.apenergy.2011.05.005.