EMITOR: Jurnal Teknik Elektro

Evaluation of ANN Training Methods: A Comparative Study of Back Propagation, Genetic Algorithm, and Particle Swarm Optimization for Predicting Electrical Energy Consumption

Giovanni Dimas Prenata*, Ahmad Ridhoi
Fakultas Teknologi Elektro dan Informatika Cerdas Program Studi Teknik Elektro – Universitas 17 Agustus 1945 Surabaya, Indonesia
*gprenata@untag-sby.ac.id

Abstract – *This paper compares Artificial Neural Networks (ANNs) trained with three strategies—Backpropagation (BP), Genetic Algorithm (GA), and Particle Swarm Optimization (PSO)—to predict electrical energy consumption at PT PLN UP3 Sidoarjo using five years of data (2015–2019). Inputs comprise six sectoral/customer features and population, with min–max normalization. Models are evaluated by MSE/MAE and validated with two sliding-window scenarios: (S1) train 2015–2017, test 2018; (S2) train 2016–2018, test 2019. Results show* ***ANN–GA*** *consistently attains low error and strong generalization, yielding the lowest average absolute error (MAE ≈ 0.0099; SD ≈ 0.0218) and meeting the target MSE ($\leq 10^{-4}$).* ***ANN–BP*** *is reliable but less accurate (MAE ≈ 0.0185, SD ≈ 0.0231) and tends to require more iterations.* ***ANN–PSO*** *is sensitive to hyperparameters: performance is suboptimal in S1 yet competitive in S2, with overall MAE ≈ 0.0189 and the largest variability (SD ≈ 0.0357). These findings indicate that GA-based weight optimization provides the best accuracy–stability trade-off for this dataset, BP remains a solid baseline, and PSO can be effective with careful parameter tuning. Future work should enlarge the dataset and incorporate exogenous variables (e.g., weather, calendar) while benchmarking modern sequence models (e.g., LSTM, CNN–BiLSTM) under identical preprocessing and validation protocols.*

Keywords – *ANN; Backpropagation; Genetic Algorithm; Particle Swarm Optimization; Electrical Energy Prediction.*

I. Introduction

ELECTRICAL energy is essential for human life, and its absence can hamper various activities. In Indonesia, its supply is managed by PT PLN (State Electricity Company), which requires national electricity demand data to adjust supply. Therefore, PLN needs to forecast electricity demand to generate energy according to demand, both now and in the future.

Research on predicting electrical energy needs has been conducted using various methods. One study analyzed the prediction of electrical load at PT PLN in the West Sumatra region using the Artificial Neural Network method [1]. Monthly electrical energy demand data from 2013 to 2015 was used as training data, while monthly electrical energy demand data in 2016 was used as test data. The results of this study showed an error value of 0.08 with 10,000 epochs in testing electrical energy needs in 2016. Another study conducted electrical load prediction using the back propagation method [2]. The results showed the highest Mean Absolute Percentage Error (MAPE) value of 4.32% and the lowest of 2.71%.

Research on electricity load prediction in Banjarbaru City has used the back propagation artificial neural network method with a 12-12-1 architecture [3]. The variables used in model training include historical electricity load data which is then processed to obtain optimal parameters. The training results show a Mean Absolute Percentage Error (MAPE) value of 6.597% and a Root Mean Square Error (RMSE) of 0.032222. In the testing phase, the model produced a MAPE value of 12.366% and an RMSE of 0.070479, which indicates an increase in errors in data outside the training sample. A similar study in Central Java has proven that the back propagation method has a reliable ability to project long-term peak electricity loads, so it can be used as a basis for decision making in electricity system planning [4]. A similar condition occurs in Rokan Hulu Regency, where population growth triggers an increase in electricity consumption to exceed installed

The manuscript was received on September 2, 2025, revised on September 11, 2025, and published online on November 28, 2025. `Emitor` is a Journal of Electrical Engineering at Universitas Muhammadiyah Surakarta with ISSN (Print) 1411 – 8890 and ISSN (Online) 2541 – 4518, holding Sinta 3 accreditation. It is accessible at `https://journals2.ums.ac.id/index.php/emitor/index`.

capacity, resulting in a supply deficit [5]. To address this issue, a prediction of electricity demand for the next five years was conducted using a time series-based back propagation artificial neural network to support more precise energy supply planning.

The use of artificial neural networks is a relevant approach to predicting power consumption because machine learning technology continues to develop and experience performance improvements. Artificial neural networks have various specialized methods tailored to specific types of problems and conditions. CNN (Convolution Neural Network) is used for object identification, while LSTM (Long Short Term Memory), which is a development of RNN (Recurrent Neural Network), is able to process information based on stored historical data. The results of deep learning implementations to predict electricity consumption depend on the type of method used [6]. The BDLSTM method has the smallest MSE (Mean Squared Error) value compared to other methods. Meanwhile, the CNN-M-BDLSTM method has the smallest MAPE (Mean Absolute Percentage Error) value compared to other methods. Forecasting electricity needs can also be done using the feed-forward back propagation method [7]. The results are quite good, namely 3.29% for the MAPE value, below the maximum requirement of 10%. Accurate electricity consumption prediction is important to support energy distribution and savings. The NAP-BiLSTM model combines attention mechanisms with BiLSTM to effectively process time series data [8]. Tests on electricity consumption and weather data show that this model has a low error rate and outperforms other deep learning methods.

This study uses an Artificial Neural Network (ANN) to predict the electricity load at PT PLN UP3 Sidoarjo. The ANN was trained using a backpropagation mechanism combined with Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) optimization methods to obtain more optimal neuron weights. The weights were updated through optimization methods, while the feedforward process remained consistent with the ANN concept.

II. Literature Review

i. ANN Algorithm Back Propagation

An Artificial Neural Network (ANN) is a computational model that mimics the workings of the human brain. An ANN consists of artificial neurons interconnected by weights. Each neuron receives input, multiplies it by a weight, sums the results, and then passes it through an activation function to produce an output. Typically, an ANN has three main layers: an input layer to receive data, a hidden layer to process the data into more complex representations, and an output layer to produce predictions or final results [9].

Backpropagation is a supervised learning algorithm commonly used in ANN training. It iteratively adjusts the network weights to reduce the difference between the output and the target. The process consists of two stages: a forward pass, where the data is propagated forward to generate the output, and a backward pass, where the error is calculated using a loss function (e.g., MSE or MAPE) and then propagated backward to update the weights using gradient descent [10]. An

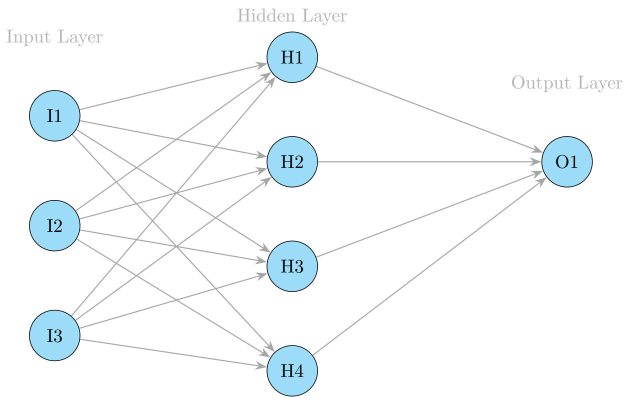


Figure 1: ANN Back Propagation Architecture

artificial neural network (ANN) consists of an input layer, one or more hidden layers, and an output layer. The main processes in an ANN are forward propagation and back propagation. In forward propagation, the incoming signal is calculated by summing the inputs multiplied by the weights, followed by a bias:

$$Z_j = \sum_{i=1}^{n} W_{ij} X_i + b_j \tag{1}$$

The value is then passed to an activation function, for example sigmoid:

$$f(Z_j) = \frac{1}{1+e^{-Z_j}} \tag{2}$$

The network output is compared to the target using Mean Squared Error (MSE):

$$E = \frac{1}{N} \sum_{k=1}^{N} (t_k - y_k)^2 \tag{3}$$

In the backpropagation method, the weights are updated by decreasing the error gradient:

$$W_{ij}(t+1) = W_{ij}(t) - \eta \frac{\partial E}{\partial W_{ij}} \tag{4}$$

where W_{ij} denotes the weight between neurons, η is the learning rate, E is the error (loss) function, and $\frac{\partial E}{\partial W_{ij}}$ is the derivative of the error function with respect to the weight W_{ij}.

ii. Genetic Algorithm (GA)

A genetic algorithm (GA) is a population-based optimization method inspired by biological evolution and natural selection. Introduced by John Holland in 1975, GAs search for optimal solutions using reproduction, mutation, and selection processes in each generation. Solutions are represented as chromosomes (e.g., binary strings or real numbers) that are evaluated with a fitness function to assess their quality [11].

The main processes in Genetic Algorithm (GA) include [12]:

1. **Population initialization**: form the initial population randomly.
2. **Fitness evaluation**: assess the quality of each solution with an objective function.
3. **Selection**: choose the best individuals as parents (e.g., roulette wheel, tournament, or rank selection).
4. **Crossover**: combine genes of two parents to produce new offspring.
5. **Mutation**: randomly change some genes to maintain diversity.
6. **New generation formation**: repeat the process until a certain number of generations or an optimal solution is reached.

In the context of ANN, network weights are represented as chromosomes:

$$Chromosome = [w_1, w_2, \dots, w_n] \tag{5}$$

Each chromosome is evaluated with a fitness function, which in this study is defined from the error value:

$$F = \frac{1}{1+MSE} \tag{6}$$

The evolutionary process is carried out through selection (choosing the best chromosome based on fitness value), crossover (combining two parent chromosomes, for example with one-point crossover), and mutation (changing a small part of the gene with a random value).

iii. Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) is a population-based optimization algorithm introduced by Kennedy and Eberhart in 1995. Inspired by the collective behavior of birds or fish, PSO searches for optimal solutions by modeling each solution as a particle with a position and velocity. At each iteration, the particle's motion is updated based on its own best experience (personal best, *pbest*) and the best experience of the entire population (global best, *gbest*) [13].

The main steps in PSO include: (1) particle initialization, which places particles randomly with a certain initial velocity; (2) fitness evaluation, which calculates the quality of each particle with an objective function; (3) personal best (*pbest*) update, which stores the best position of each particle; (4) global best (*gbest*) update, which determines the best position in the entire population; and (5) velocity update, which adjusts the particle velocity with the following formula:

$$\begin{aligned} v_i(t+1) = w \cdot v_i(t) + c_1 r_1 (pbest_i - x_i(t)) \\ + c_2 r_2 (gbest - x_i(t)) \end{aligned} \tag{7}$$

The particle position is then updated with:

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{8}$$

Where w is the inertia weight, c_1 and c_2 are the acceleration coefficients, while r_1 and r_2 are random numbers between 0 and 1. The last step is the stopping criteria: the iteration is stopped if the maximum number of iterations is reached or if the optimal fitness value has been found [14].

III. Results and Discussion

i. ANN Algorithm Back Propagation

This study used 5 years of electricity consumption data (2015–2019). The training and test data went through a min-max normalization process before being used [15]. The training data was then trained on an Artificial Neural Network Back Propagation. The following are the training and test data after normalization.

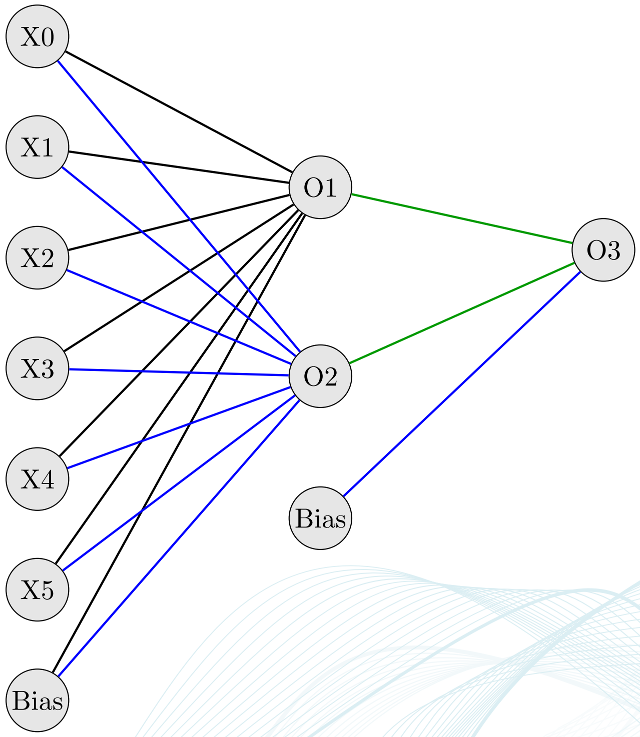


Figure 2: ANN Back Propagation Architecture for Prediction

The Back Propagation method begins by randomly determining the weights for each neuron, then calculating the output for the second layer O_1 and O_2 (hidden layer), then continuing to calculate the output on layer 3 O_3 (output layer). The ANN architecture used in the first layer (input layer) consists of 6 input neurons and 1 bias neuron. These 6 input neurons are based on the parameters of population (X_0), social sector customers (X_1), business sector customers (X_2), industrial sector customers (X_3), household sector customers (X_4), and government sector customers (X_5). The training data includes data from 2015 to 2018. The 2019 data is used as test data that will be tested on the model when the training stage is complete.

At the end, the calculation of the final output value (O_3) will be matched with the target (electrical energy consumption). If the MSE (Mean Squared Error) value for the 4 conditions does not match the MSE target, then the weight update will occur in each neuron as a backward stage. After the update, the O_1, O_2, and O_3 values will be calculated again as a forward stage. The forward and backward stages will be carried out continuously until the MSE target (0.0001) is reached. The MSE value represents the average level of squared error between the model's predicted value and the actual target value.

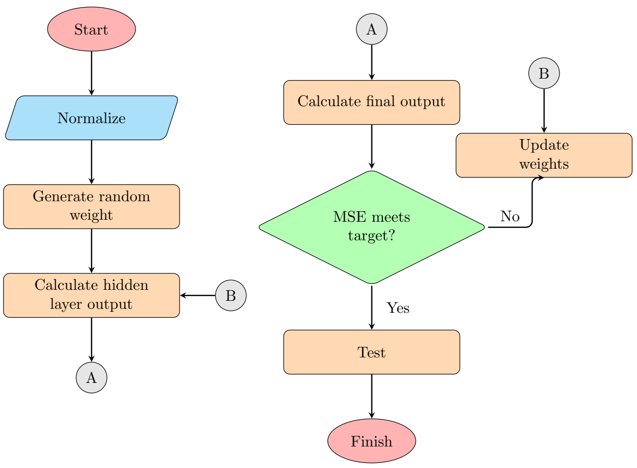


Figure 3: ANN Algorithm Back Propagation Flowchart

Once the MSE value meets the target, the next stage is testing. Using 2019 data, the model's success in predicting electricity consumption will be determined. The resulting predictions will then be calculated for accuracy and used as a basis for further analysis.

ii. Artificial Neural Network – Genetic Algorithm

To optimize the results of using ANN Back Propagation, a Genetic Algorithm (GA) was applied in this study. The ANN architecture remains the same as in Figure 2, with only a difference in the sequence/stages.

The data used is also the same as the previous method and is normalized before being used as training and test data. The GA implementation in this study uses 10 chromosomes with 17 genes each. These 17 genes represent 17 weights in the ANN.

At the beginning of the GA model, the values of the 17 genes for the 10 chromosomes are obtained by generating random values. Then, the fitness function value is calculated using the ANN concept. The ANN concept is to calculate the O_1, O_2, and O_3 values with sigmoid activation for each of these outputs.

Next, calculate the MSE value for each chromosome based on 4 conditions/training data (2015–2019). The next stage is the selection stage, finding the 2 best MSE values (lowest MSE) from the 10 chromosomes to be used as parent chromosomes. After obtaining 2 parent chromosomes, the crossover stage is continued, namely exchanging genes between the 2 parent chromosomes.

Next is the mutation stage, namely replacing genes on the parent chromosomes with random values generated if the mutation rate value is met. The mutation stage produces 2 mutation chromosomes that will be included in the population (regeneration). The regeneration stage will replace 2 chromosomes in the population that have poor (high) MSE values with 2 mutation chromosomes. If the MSE value of 2 mutation chromosomes is worse (higher) than the 10 MSE values of each chromosome in the population, then there is no regeneration, and vice versa.

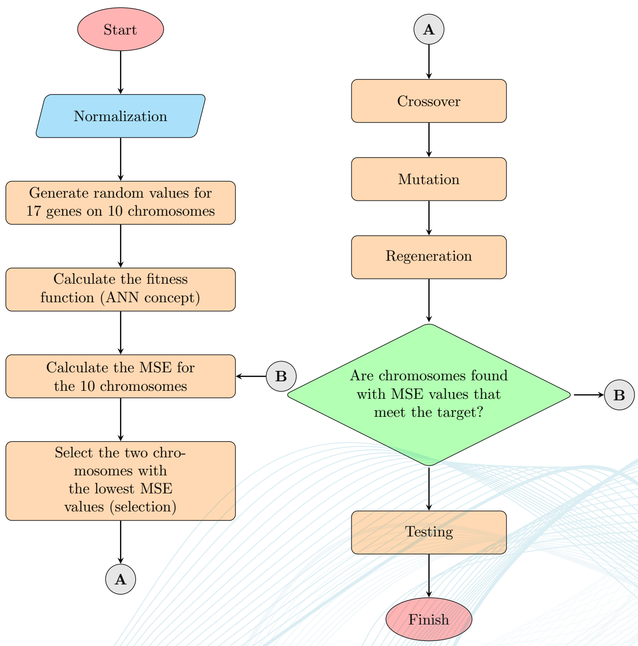


Figure 4: ANN Genetic Algorithm Flowchart

iii. *Artificial Neural Network – Particle Swarm Opt.*

For comparison, this study used another optimization method, Particle Swarm Optimization (PSO). The data used remained the same and underwent a normalization process. The following are the steps for ANN–PSO.

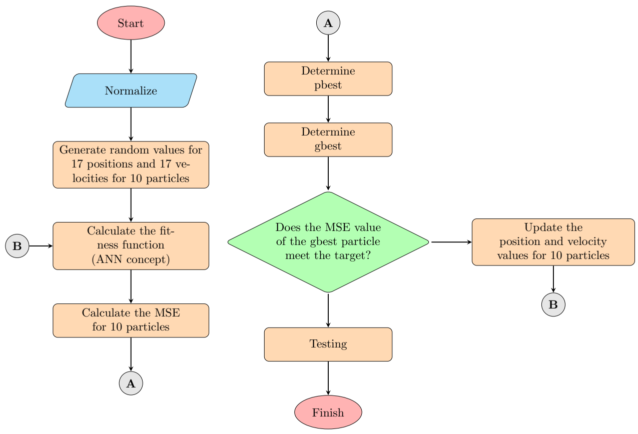


Figure 5: ANN Particle Swarm Optimization Flowchart

This PSO implementation uses 10 particles, each with 17 positions and 17 velocities. This represents the 17 weights in the ANN. Initially, the 17 positions and 17 velocities for the 10 particles are generated randomly. Next, the fitness function for the 10 particles is calculated using the ANN concept. The MSE value is then calculated for all 10 particles, after which the *pbest* and *gbest* values are determined.

The MSE value of the particle selected as the best particle will be compared to the target MSE value. If the value does not match the target, the position and velocity values for 10 particles will be adjusted. This process will be repeated until the MSE value of the best particle reaches the target MSE value. Once it has reached the target, testing and accuracy calculations will be performed.

iv. *Data Normalization*

In the process of training artificial neural networks, one of the crucial stages is data normalization. Normalization aims to equalize the scale between input variables so that the learning process can proceed more stably and efficiently. Without normalization, variables with a large range of values will dominate the weight calculation, while variables with a small range will be less influential. As a result, the training algorithm can converge slowly or even get stuck in a suboptimal solution. In general, there are several normalization methods widely used in the literature, namely min–max normalization, z-score standardization, and logarithmic transformation.

Prior to training, all input features were normalized using min–max normalization to rescale the values into the range [0,1]. This method was chosen because the ANN model in this study uses a sigmoid activation function, which performs best when the input is bounded between 0 and 1, thus avoiding saturation and accelerating convergence. Alternative normalization techniques such as z-score standardization are more suitable for data with a Gaussian distribution, while logarithmic transformation is often applied to highly skewed data. Since the electricity consumption data in this study is strictly positive and varies across customer sectors by different magnitudes, min–max normalization was deemed the most appropriate approach to maintain proportionality while ensuring compatibility with the selected activation function. The min–max normalization formula is written as [16]:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{9}$$

This method maps data to a range of 0–1. The advantage of min–max is that it maintains the proportional relationship between the original data, preserving the relative distribution between samples. This method is particularly suitable for neural networks with sigmoid or tanh activation functions, which perform optimally when inputs fall within a limited range. Mapping to 0–1 minimizes the risk of neuron saturation and accelerates the convergence of the learning process. Z-score standardization transforms data with the formula [16]:

$$x' = \frac{x - \mu}{\sigma} \tag{10}$$

Where μ is the mean and σ is the standard deviation of the variable. This method produces data with a mean of zero and a variance of one. The z-score is suitable when the data is normally distributed, as it can stabilize variables with different variances. However, if the data distribution is far from normal (e.g., highly skewed or contains outliers), this method is less effective and can cause extreme values to remain strongly influential. Logarithmic transformation is used to reduce skewness in data with highly skewed distributions. The most common formula is [17]:

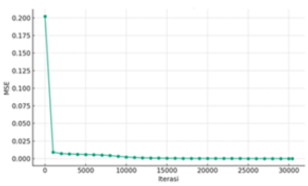
$$x' = \log(1 + x) \tag{11}$$

With this transformation, very large values are compressed so they do not dominate the scale. This method is widely used in financial data, internet traffic, or social data that tend to have a “heavy-tail” distribution. In this study, the normalization method chosen is min–max normalization. This is based on the characteristics of electricity consumption data which is positive, with a range of values that differs between years, and does not strictly follow a normal distribution. In addition, the ANN model used utilizes the sigmoid activation function, which theoretically works best when the input is within the [0,1] domain. With min–max normalization, all input variables can be mapped to a uniform range, resulting in a more stable training process, reduced risk of neuron saturation, increased convergence speed, and improved model generalization on test data.

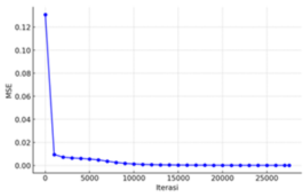
IV. Results and Discussion

i. *Model Performance Without Cross-Validation*

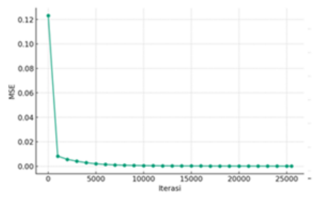
This study uses the concept of MSE (Mean Squared Error) to determine the results of AI model training. The target MSE value is less than or equal to 0.0001. If this value is achieved, the model training process is complete. However, there is also a limit on the number of iterations in model training of 50,000 iterations. If the number of iterations has reached 50,000 but the MSE value obtained is still not less than or equal to 0.0001, model training is also stopped. An MSE value of 0 represents the model's predicted results exactly the same as the target value (without error). The smaller the MSE value, the better the model performance because the average prediction error is smaller. The larger the MSE value, the worse the model performance because the prediction error is large.



(a)



(b)



(c)

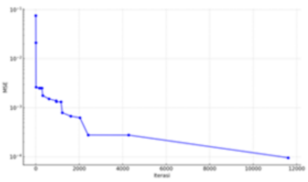
Figure 6: ANN Back Propagation Results

In the ANN Backpropagation method, model training and testing were carried out three times. The first test result (Figure 6a) required 30,420 iterations to obtain an MSE value of less than 0.0001. The second test result (Figure 6b) required 27,524 iterations, and the third test result (Figure 6c) required 25,508 iterations.

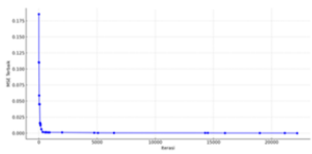
For the ANN GA method, three models were also trained and tested. The first model (Figure 7a) required 11,613 iterations to achieve an MSE value of less than

Table 1: Absolute error from ANN Back Propagation method

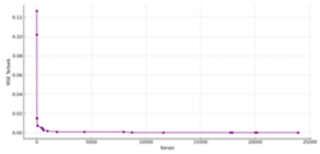
Model	Data	Output target	Output model	Absolute error
1	Train	0	0.0188348	0.019
		0.469862	0.465502	0.004
		0.5674222	0.57171	0.004
		0.804996	0.803193	0.002
	Test	1	0.938455	0.062
2	Train	0	0.0185816	0.019
		0.469862	0.465144	0.005
		0.5674222	0.572214	0.005
		0.804996	0.802954	0.002
	Test	1	0.93886	0.061
3	Train	0	0.0189147	0.019
		0.469862	0.465575	0.004
		0.5674222	0.571548	0.004
		0.804996	0.803347	0.001
	Test	1	0.93424	0.066



(a) Model 1



(b) Model 2



(c) Model 3

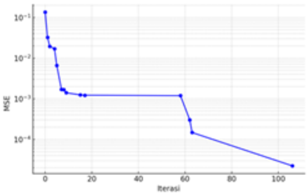
Figure 7: ANN Genetic Algorithm Results

0.0001. The second model (Figure 7b) required 22,189 iterations, and the third model (Figure 7c) required 23,931 iterations.

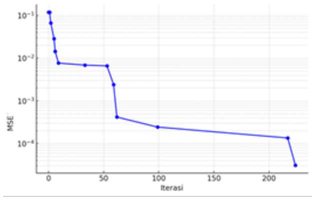
Table 2: Absolute error from ANN GA method

Model	Data	Output target	Output model	Absolute error
1	Train	0	0.0002379	0.000238
		0.469862	0.489103	0.019241
		0.567422	0.565013	0.002409
		0.804996	0.802448	0.002548
	Test	1	1	0.000000
2	Train	0	0.00000037	0.00000037
		0.469862	0.558817	0.088955
		0.567422	0.572214	0.004792
		0.804996	0.814715	0.009719
	Test	1	1	0.000000
3	Train	0	0	0.000000
		0.469862	0.472429	0.002567
		0.567422	0.572721	0.005299
		0.804996	0.792183	0.012813
	Test	1	0.999947	0.000053

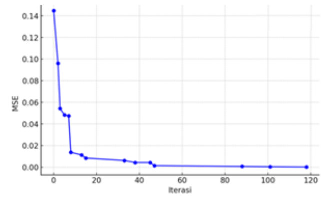
The ANN PSO method also yielded three models. However, the results were poor compared to the



(a) Model 1



(b) Model 2



(c) Model 3

Figure 8: ANN Particle Swarm Optimization Results

ANN Backpropagation and ANN GA methods. With the same target MSE value of 0.0001 and a maximum iteration limit of 50,000 times, it still failed to achieve the MSE target.

Table 3: Absolute error from ANN PSO method

Model	Data	Output target	Output model	Absolute error
1	Train	0	0.0000592609	0.000059260
		0.469862	0.477718	0.0078560
		0.5674222	0.563351	0.0040712
		0.804996	0.808341	0.0033450
	Test	1	0.999964	0.0000360
2	Train	0	0.000085066	0.000085066
		0.469862	0.464859	0.0050030
		0.5674222	0.560096	0.0073262
		0.804996	0.811662	0.0066660
	Test	1	0.880598	0.1194020
3	Train	0	0.0000106824	0.000010682
		0.469862	0.483755	0.0138930
		0.5674222	0.562752	0.0046702
		0.804996	0.791858	0.0131380
	Test	1	0.901576	0.0984240

From the three graphs above, the MSE value experienced convergence at a low iteration (below 2000 iterations), even though the maximum iteration limit is 50,000. This illustrates that even though the iterations were carried out with many opportunities, the resulting MSE value was unable to reach the target (0.0001). Meanwhile, for testing using training data and test data, the results also did not meet the target. The results were all off, this is because the desired MSE target was not met. So when the model was tested, the results were all-off.

Based on Table 4, the absolute error value of the model is getting smaller (approaching zero). This means that the model's prediction/output is getting closer to the target. Meanwhile, if the absolute error value is larger (farther from zero), it means that the prediction/output is getting further from the target. The model output value in the ANN PSO method is based on the *gbest* value obtained at each iteration. If in the next iteration the *gbest* value is not better than the previous *gbest* value, then the *gbest* value does not change and remains the same as in the previous iteration.

The PSO method is based on swarm intelligence, where each particle tends to move toward the *gbest* value obtained by the group. While each particle has its own *pbest*, the *gbest* value is selected from the best *pbest* among all particles.

Table 4: Absolute error from ANN PSO method

Model	Data	Output target	Output model	Absolute error
1	Train	0	0.0000592609	0.000059260
		0.469862	0.477718	0.0078560
		0.5674222	0.563351	0.0040712
		0.804996	0.808341	0.0033450
	Test	1	0.999964	0.0000360
2	Train	0	0.000085066	0.000085066
		0.469862	0.464859	0.0050030
		0.5674222	0.560096	0.0073262
		0.804996	0.811662	0.0066660
	Test	1	0.880598	0.1194020
3	Train	0	0.0000106824	0.000010682
		0.469862	0.483755	0.0138930
		0.5674222	0.562752	0.0046702
		0.804996	0.791858	0.0131380
	Test	1	0.901576	0.0984240

ii. Model Performance with Sliding Window Cross-Validation

The use of two sliding window cross-validation scenarios allows for evaluation of the model's robustness to variations in time periods. If the model's performance is relatively consistent across both scenarios, it can be concluded that the model has good generalization capabilities. Conversely, if significant differences in the results occur, this provides additional insight into the model's limitations in addressing changes in electricity consumption patterns between periods. Thus, this validation approach also addresses reviewers' criticisms regarding the lack of cross-validation in previous studies, as the model is now evaluated on different test data to avoid bias towards the training data.

The test results with the division of training data (2015–2017) and test data (2018) show a significant difference in performance between the ANN Backpropagation, ANN GA, and ANN PSO methods. First, in terms of convergence, the ANN GA method is able to achieve the target MSE faster (417–599 iterations) with the best MSE values ranging from 2.6×10^{-5} to 3.2×10^{-5}. This indicates that the crossover and mutation mechanisms in GA are effective in exploring the

Table 5: Sliding Window Cross-Validation (Training data 2015–2017)

Model Name	Max Iterations	Best MSE
GA-ANN (1)	568	0.0000266615
GA-ANN (2)	417	0.0000304883
GA-ANN (3)	599	0.0000328330
PSO-ANN (1)	206	0.0000912207
PSO-ANN (2)	130	0.0000149662
PSO-ANN (3)	87	0.0000833698
BP (1)	4537	0.0000999995
BP (2)	3342	0.0000999751
BP (3)	3408	0.0000999805

Table 6: Model output on training data (2015–2017) and test data (2018)

Model Name	2015	2016	2017	2018
GA-ANN (1)	0	0.46095	0.568195	0.629789
GA-ANN (2)	0.0018364	0.46143	0.563287	0.611912
GA-ANN (3)	0	0.47827	0.571897	0.774707
PSO-ANN (1)	0.0064848	0.47659	0.553776	0.589952
PSO-ANN (2)	0.0013792	0.46381	0.569948	0.583135
PSO-ANN (3)	0.0027538	0.48338	0.575151	0.583899
BP (1)	0.0169091	0.46699	0.569485	0.579008
BP (2)	0.0169908	0.46712	0.569098	0.578291
BP (3)	0.0170587	0.46739	0.568837	0.581477

solution space and finding optimal ANN weights. In contrast, the Backpropagation method requires a much larger number of iterations (3,300–4,500 iterations) to achieve an MSE of around 1×10^{-4}, so although the results are quite stable, the training process is slower and computationally wasteful. In the ANN PSO method, there are relatively few iterations (87–206 iterations) with varying MSE (1.4×10^{-5} to 9.1×10^{-5}), but they are inconsistent between experiments, thus indicating that there is a trap in a local solution.

Second, on the training data (2015–2017), ANN GA produced the closest output to the target with the smallest absolute error compared to the other two methods. Backpropagation was relatively stable but tended to produce deviations of ± 0.01–0.02 from the target. ANN PSO was also able to produce values quite close to the target, but with lower accuracy than GA. Third, on the test data (2018), only ANN GA was able to produce predictions close to the target value (0.774707 compared to the target of 0.804996 in one model). Meanwhile, the Backpropagation and PSO methods tended to provide lower predictions (~ 0.58), thus indicating limitations in generalization capabilities.

Overall, these results confirm that ANN GA has advantages in both convergence speed and generalization capability, while Backpropagation can still be used but with higher computational costs. However, ANN PSO requires further parameter sensitivity analysis to improve its performance, thus becoming an opportunity for future research.

Table 7: Sliding Window Cross-Validation (Training data 2016–2018)

Model Name	Max Iterations	Best MSE
GA-ANN (1)	7479	0.0000503272
GA-ANN (2)	14949	0.0000651318
GA-ANN (3)	31187	0.0000674051
PSO-ANN (1)	1631	0.0000923351
PSO-ANN (2)	321	0.0000249741
PSO-ANN (3)	135	0.0000390909
BP (1)	1322	0.0000999333
BP (2)	1481	0.0000999496
BP (3)	1434	0.0000998868

Table 8: Model output on training data (2016–2018) and test data (2019)

Model Name	2016	2017	2018	2019 (Test)
GA-ANN (1)	0.466582	0.555619	0.804041	1
GA-ANN (2)	0.46626	0.566734	0.791507	1
GA-ANN (3)	0.475307	0.576436	0.814552	1
PSO-ANN (1)	0.474073	0.551370	0.804996	0.997624
PSO-ANN (2)	0.466042	0.572674	0.799274	0.999796
PSO-ANN (3)	0.472219	0.575236	0.812114	0.996140
BP (1)	0.462679	0.579691	0.796562	0.914100
BP (2)	0.462545	0.579818	0.796782	0.926353
BP (3)	0.463144	0.579170	0.795808	0.908651

Test results with two cross-validation scenarios (Scenario 1: training data 2015–2017, test data 2018; Scenario 2: training data 2016–2018, test data 2019) show significant performance variations between the ANN Backpropagation, ANN GA, and ANN PSO methods. In Scenario 1, the ANN GA method is able to achieve the target MSE in 417–599 iterations with the best MSE values ranging from 2.6×10^{-5} to 3.2×10^{-5}. In contrast, Backpropagation requires 3,300–4,500 iterations to achieve an MSE of around 1×10^{-4}. PSO is relatively fast (87–206 iterations) with varying MSE (1.4×10^{-5} to 9.1×10^{-5}), but results are inconsistent. On the test data (2018), only ANN GA produced predictions close to the target, while Backpropagation and PSO under-predicted.

In Scenario 2, PSO performance improved significantly. PSO converged faster (135–1,631 iterations) with the best MSE of 2.4×10^{-5} – 9.2×10^{-5} and produced predictions on the test data (2019) that were very close to the target (0.996–0.999). ANN GA still provided accurate results (1.0 on the test data), but required 7,479–31,187 iterations. Backpropagation required 1,322–1,481 iterations with an MSE close to 1×10^{-4}, but its predictions on the test data were lower (0.908–0.926).

Overall, the results of both scenarios show that ANN GA excels in accuracy and consistency, despite requiring more iterations. ANN PSO exhibits fluctuating performance: suboptimal in Scenario 1, but nearly perfect in Scenario 2. This confirms that PSO performance is strongly influenced by parameter selection (c_1, c_2, and w) and data conditions. Backpropagation remains stable, but its limitations are seen in the high iteration requirement and lower prediction accuracy. Thus, ANN GA can be recommended as a more reliable method for electricity consumption prediction, while PSO has the potential to be competitive with more systematic parameter tuning.

iii. Comparison with Previous Research

To strengthen the claims of this study, it is important to place the findings within the context of previous literature. Several studies have used population-based optimization methods and deep learning architectures to improve the accuracy of electricity consumption predictions. Chafi and Afrakhte [18] reported that the use of Particle Swarm Optimization (PSO) in tuning Artificial Neural Network (ANN) weights improved short-term load prediction performance compared to standard ANNs. This approach reduces error by averaging several error indices as the objective function, effectively avoiding the local minima trap that often occurs in pure backpropagation. These results are consistent with the findings of this study, where PSO-ANN can produce reasonably good accuracy after proper parameter optimization.

Furthermore, Al-Qaysi [19] showed that the Genetic Algorithm (GA) used to optimize ANN and ANFIS (Adaptive Neuro-Fuzzy Inference System) weights in a case study in Iraq produced lower error rates than conventional ANN methods. GA has the advantage of exploring a wide solution space and avoiding premature convergence, making it more stable in finding optimal weight configurations. This is in line with the results of this study, where GA-ANN consistently provides the best prediction performance compared to backpropagation and PSO.

On the other hand, research trends in the last decade have largely focused on the use of deep learning, particularly Long Short-Term Memory (LSTM)-based Recurrent Neural Network (RNN) architectures. Kong [20] and Tang [21] demonstrated that both LSTM and hybrid CNN-BiLSTM models have superior capabilities in capturing long-term temporal dependencies,

Table 9: Comparison of previous research results related to energy consumption prediction

Study	Main method	Dataset	Key Results
[18]	PSO optimizes NN parameters/architecture (PSO-ANN)	Short-term load data (Math. Problems in Engineering)	Shows that PSO can optimize NN parameters and improve accuracy compared to NN without optimization.
[19]	GA for ANN weight optimization (GA-ANN) and comparison with ANFIS	Iraqi regional electricity load data (real case)	GA-ANN (or hybrid GA) provides improved accuracy vs ANN/ANFIS on the dataset used.
[20]	LSTM (RNN) for short-term residential load forecasting	Residential smart-meter data (public/real)	LSTM outperforms several traditional baselines at the residential level. Evidence that RNN/LSTM is more robust on granular temporal data.
[21]	Hybrid CNN + BiLSTM	Tie-line data / power system uncertainty conditions	CNN-BiLSTM outperforms LSTM and ResNet, showing modern hybrid architectures are often superior on complex temporal/spatial patterns.
[22]	Review and experiments on CNN-LSTM	NTDC dataset (Pakistan)	Comprehensive review shows hybrid DL (CNN-LSTM / BiLSTM) often provides better RMSE/MAPE on STLF; supports claim that DL methods outperform classical ANNs.

thus reducing electricity load prediction errors compared to classical ANN methods. Furthermore, a comprehensive review by Ullah [22] confirmed that hybrid deep learning models, such as CNN-LSTM, consistently outperform in terms of RMSE and MAPE on large, high-resolution datasets (e.g., hourly or daily data).

However, it should be noted that these inter-study comparisons are not entirely apple-to-apple. Each study used different datasets, both in terms of time resolution (daily, monthly, seasonal), number of input features (e.g., weather variables, calendar data, or demographic data), and preprocessing methods chosen. Therefore, claims of superiority of one method over another must be viewed in the context of the dataset and experimental setup used. Given the limited data in this study (only five years from a single region), the benchmarking was conducted primarily as a qualitative reference to demonstrate this study's position among previous studies. For a truly fair quantitative comparison, repeat experiments by retraining the benchmark models on the same dataset and preprocessing as this study are necessary.

The evaluation results of the ANN–Backpropagation model show that this method has an average absolute error of 0.0185. The maximum error value obtained is relatively lower compared to other methods, which is 0.0660, while the minimum error is recorded at 0.0020. This indicates that ANN–Backpropagation is quite stable in producing predictions, although the average absolute error value is still higher than the ANN–GA method. The level of error variation indicated by the standard deviation of 0.0231 also shows that the error fluctuations between samples are relatively controlled. Comparisons were made for the ANN Backpropagation method, ANN Genetic Algorithm, and ANN Particle Swarm Optimization using the MAE (Mean Absolute Error) concept.

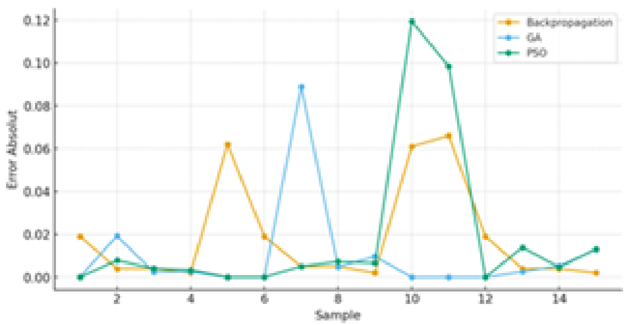


Figure 9: Comparison of ANN BP, ANN GA, and ANN PSO based on MSE values [23]

The ANN–PSO model yielded an average absolute error of 0.0189, slightly higher than both ANN–Backpropagation and ANN–GA. Although this method yielded a minimum error close to zero in some samples, ANN–PSO also demonstrated weaknesses, with the highest maximum error of 0.1194. This indicates prediction instability in some cases. Furthermore, the standard deviation error of 0.0357 was the highest among the three methods, concluding that ANN–PSO tends to produce less consistent errors.

Meanwhile, the analysis results of the ANN–GA model showed better performance compared to the other two methods. The average absolute error value obtained was 0.0099, the lowest among all the models tested. ANN–GA was also able to produce a minimum error close to zero, indicating excellent predictive ability across a number of samples. However, this method still had one case with a relatively large error of 0.0889. Nevertheless, the error variation in ANN–GA was relatively small with a standard deviation of 0.0218, so the resulting predictions were more stable and accurate overall compared to ANN–Backpropagation and ANN–PSO. Thus, ANN–GA provided the lowest absolute error and the most stable distribution compared to ANN–PSO and ANN–Backpropagation.

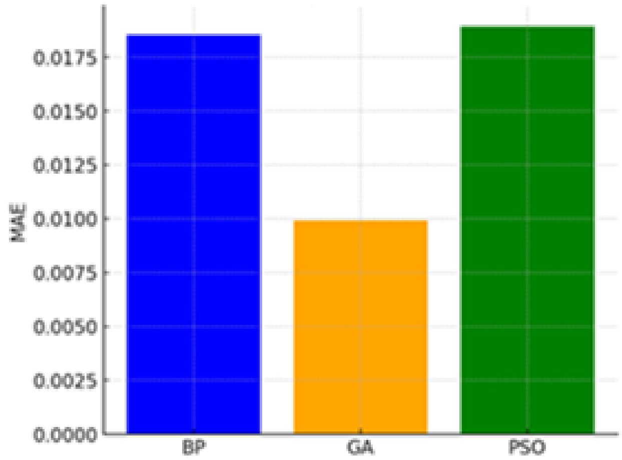


Figure 10: Comparison of MAE for ANN BP, ANN GA, and ANN PSO

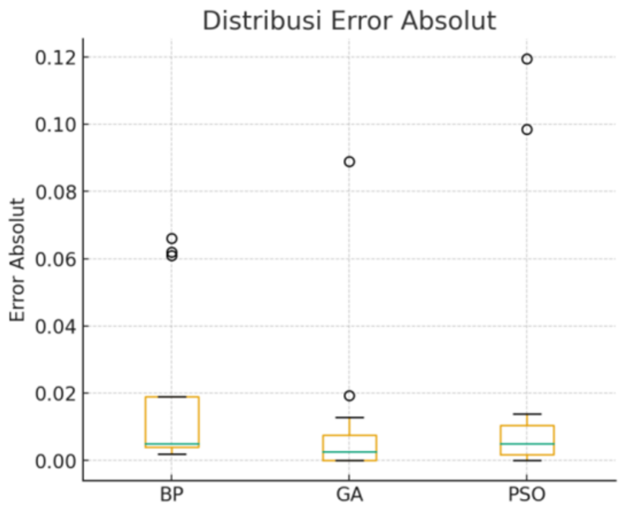


Figure 11: Absolute error distribution for ANN BP, ANN GA, and ANN PSO

V. Conclusion

This study compared three ANN training strategies—Backpropagation (BP), Genetic Algorithm (GA), and Particle Swarm Optimization (PSO)—for predicting electrical energy consumption. Overall, **ANN–GA** delivered the most accurate and stable performance: it consistently reached low MSE targets and produced the smallest average absolute error among all models. **ANN–BP** remained reliable and comparatively stable, though it required more iterations and yielded higher average errors than GA. **ANN–PSO** showed mixed results: it was suboptimal in the first sliding-window scenario but nearly matched the target in the second, confirming that PSO's effectiveness is highly sensitive to parameter choices (e.g., c_1, c_2, and w) and data regime.

Sliding-window cross-validation reinforced these findings. With training on 2015–2017 and testing on 2018, GA converged faster to low MSE and generalized better than BP and PSO. With training on 2016–2018 and testing on 2019, PSO improved markedly yet still depended on careful tuning, while GA retained strong accuracy albeit with higher iteration counts. Across experiments, min–max normalization supported faster convergence and reduced saturation for the sigmoid-based networks.

In practice, GA is the recommended optimizer for this problem setting due to its balance of accuracy and robustness; BP remains a dependable baseline; and PSO can be competitive with systematic parameter tuning. Future work should enlarge the dataset and feature set (e.g., weather and calendar variables), perform formal sensitivity analyses for PSO/GA hyperparameters, and benchmark against modern sequence models (e.g., LSTM, CNN–BiLSTM) under identical preprocessing and evaluation protocols.

References

[1] S. Arif *et al.*, "Analisa prediksi beban listrik menggunakan artificial neural network (ann)," in *SNESTIK Seminar Nasional Teknik Elektro, Sistem Informasi, dan Teknik Informatika*, 2023, pp. 104–111. [Online]. Available: https://ejurnal.itats.ac.id/snestikdanhttps://snestik.itats.ac.id

[2] F. Rohman, "Prediksi beban listrik dengan menggunakan jaringan syaraf tiruan metode backpropagation," *Jurnal Surya Energy*, vol. 5, no. 2, pp. 55–60, 2022. [Online]. Available: https://doi.org/10.32502/jse.v5i2.3092

[3] M. N. Fadilah, A. Yusuf, and N. Huda, "Prediksi beban listrik di kota banjarbaru menggunakan jaringan syaraf tiruan backpropagation," *Jurnal Matematika Murni dan Terapan Epsilon*, vol. 14, no. 2, p. 81, 2021. [Online]. Available: https://doi.org/10.20527/epsilon.v14i2.2961

[4] N. Fadlilah, I. Harjanto, and M. Novita, "Prediksi beban listrik jangka panjang di wilayah jawa tengah menggunakan algoritma jaringan syaraf tiruan backpropagation," in *Scientific Engineering National Seminar (SENS)*, vol. 6, no. 5, 2021, pp. 598–604. [Online]. Available: http://conference.upgris.ac.id/index.php/sens/article/view/2517

[5] A. F. Aziz *et al.*, "Menggunakan metode jaringan syaraf tiruan (backpropagation) di kabupaten rokan hulu," 2022.

[6] N. Daratha, P. Wardana, and M. K. A. Rosa, "Perbandingan metode-metode deep learning dalam prediksi konsumsi daya listrik rumah jangka pendek," *Jurnal Amplifier: Jurnal Ilmiah Bidang Teknik Elektro dan Komputer*, vol. 14, no. 1, pp. 61–69, 2024. [Online]. Available: https://doi.org/10.33369/jamplifier.v14i1.34367

[7] A. A. Diwanda, I. N. Setiawan, and W. Setiawan, "Peramalan kebutuhan energi listrik jangka panjang di provinsi bali rentang tahun 2020–2030 menggunakan neural network," *Jurnal SPEKTRUM*, vol. 8, no. 2, p. 99, 2021. [Online]. Available: https://doi.org/10.24843/spektrum.2021.v08.i02.p12

[8] D. Li, C. Xiao, X. Zeng, and Q. Shi, "Short-mid term electricity consumption prediction using non-intrusive attention-augmented deep learning model," *Energy Reports*, vol. 8, pp. 10 570–10 581, 2022. [Online]. Available: https://doi.org/10.1016/j.egyr.2022.08.195

[9] A. Lopez, D. Math, M. Professor, and C.-K. Li, "Neural networks: The backpropagation algorithm," pp. 1–7, 2023.

[10] A. N. Rachmayani, "Jaringan saraf tiruan," p. 6, 2015.

[11] P. Waluyo, "Algoritma genetika: Teori dan aplikasinya untuk bisnis dan industri," 2018. [Online]. Available: https://www.academia.edu/15288949/ALGORITMA_GENETIKA_Teori_dan_Aplikasinya_untuk_Bisnis_dan_Industri

[12] A. R. Garcia, S. B. Filipe, C. Fernandes, C. Estevão, and G. Ramos, "Covariance structure analysis of health-related indicators in home-dwelling elderly with a focus on subjective health perception," Unpublished manuscript.

[13] A. Slowik, "Particle swarm optimization," in *Industrial Electronics Handbook - Five Volume Set*, 2011. [Online]. Available: https://doi.org/10.1007/978-3-319-46173-1_2

[14] T. Fearn, "Particle swarm optimisation," *NIR news*, vol. 25, no. 1, pp. 27–27, 2014. [Online]. Available: https://doi.org/10.1255/nirn.1421

[15] I. Permana, "The effect of data normalization on the performance of the classification results of the backpropagation algorithm," *Indonesian Journal of Information Research and Software Engineering*, vol. 2, no. 1, pp. 67–72, 2022. [Online]. Available: https://media.neliti.com/media/publications/485639-pengaruh-normalisasi-data-terhadap-perfo-e19e3a00.pdf

[16] S. Bhanja and A. Das, "Impact of data normalization on deep neural network for time series forecasting," pp. 5–10, 2018. [Online]. Available: http://arxiv.org/abs/1812.05519

[17] C. Feng *et al.*, "Log-transformation and its implications for data analysis," *Shanghai Archives of Psychiatry*, vol. 26, no. 2, pp. 105–109, 2014. [Online]. Available: https://doi.org/10.3969/j.issn.1002-0829.2014.02

[18] Z. S. Chafi and H. Afrakhte, "Short-term load forecasting using neural network and particle swarm optimization (pso) algorithm," *Mathematical Problems in Engineering*, vol. 2021, 2021. [Online]. Available: https://doi.org/10.1155/2021/5598267

[19] A. M. M. Al-Qaysi, A. Bozkurt, and Y. Ates, "Load forecasting based on genetic algorithm–artificial neural network-adaptive neuro-fuzzy inference systems: A case study in iraq," *Energies*, vol. 16, no. 6, 2023. [Online]. Available: https://doi.org/10.3390/en16062919

[20] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-term residential load forecasting based on lstm recurrent neural network," *IEEE Transactions on Smart Grid*, vol. 10, no. 1, pp. 841–851, 2019. [Online]. Available: https://doi.org/10.1109/TSG.2017.2753802

[21] U. P. Systems, "An improved cnn-bilstm model for power load prediction in," 2024.

[22] K. Ullah *et al.*, "Short-term load forecasting: A comprehensive review and simulation study with cnn-lstm hybrids approach," *IEEE Access*, vol. 12, pp. 111 858–111 881, 2024. [Online]. Available: https://doi.org/10.1109/ACCESS.2024.3440631

[23] T. Chai and R. R. Draxler, "Root mean square error (rmse) or mean absolute error (mae)? arguments against avoiding rmse in the literature," *Geoscientific Model Development*, vol. 7, no. 3, pp. 1247–1250, 2014. [Online]. Available: https://doi.org/10.5194/gmd-7-1247-2014