Performance of Levenberg-Marquardt Algorithm in Backpropagation Network Based on the Number of Neurons in Hidden Layers and Learning Rate

Hindayati Mustafidah[1], Suwarsito[2]

[1] *Informatic Engineering, Universitas Muhammadiyah Purwokerto*
[2] *Geography Education, Universitas Muhammadiyah Purwokerto*
Purwokerto, Central Java, 53182, Indonesia

[1]fida.mustafidah@ump.ac.id
[2]suwarsito.ito@ump.ac.id

Abstract - One of the supervised learning paradigms in artificial neural networks (ANN) that are in great developed is the backpropagation model. Backpropagation is a perceptron learning algorithm with many layers to change weights connected to neurons in hidden layers. The performance of the algorithm is influenced by several network parameters including the number of neurons in the input layer, the maximum epoch used, learning rate (lr) value, the hidden layer configuration, and the resulting error (MSE). Some of the tests conducted in previous studies obtained information that the Levenberg-Marquardt training algorithm has better performance than other algorithms in the backpropagation network, which produces the smallest average error with a test level of α = 5% which used 10 neurons in a hidden layer. The number of neurons in hidden layers varies depending on the number of neurons in the input layer. In this study an analysis of the performance of the Levenberg-Marquardt training algorithm was carried out with 5 neurons in the input layer, a number of n neurons in hidden layers (n = 2, 4, 5, 7, 9), and 1 neuron in the output layer. Performance analysis is based on network-generated errors. This study uses a mixed method, namely development research with quantitative and qualitative testing using ANOVA statistical tests. Based on the analysis, the Levenberg-Marquardt training algorithm produces the smallest error of 0.00014 ± 0.00018 on 9 neurons in hidden layers with lr = 0.5.

Keywords: ***hidden layer, backpropogation, MSE, learning rate, Levenberg-Marquardt***

I. INTRODUCTION

Soft computing has come as an impact of the development of computer science technology which is an approach technique in solving problems [1]. Soft computing is part of an intelligent system which is a model approach to computation by imitating human reason and has the ability to reason and learn in an environment filled with uncertainty and inaccuracy. Artificial Neural Networks (ANN) are biologically inspired computational models. ANN consists of several processing elements (neurons) and there is a relationship between neurons that will transform information received by one neuron to another neuron. This relationship is called *weight*. Deboeck and Kohonen describe ANN as a collection of mathematical techniques that can be used for signal processing, forecasting and grouping, and are referred to as non-linear, multi-layered parallel regression techniques [2]. ANN as one of the main components of forming soft computing have been widely applied in various fields of human life both for the purposes of research and solving technical problems such as forecasting, diagnostics, and pattern recognition [3], [4].

Backpropagation is the most widely used type of learning paradigm with or without supervision in ANN, especially in developing systems to solve problems. Systems known to have used backpropagation have been studied to detect intrusions in the banking system [5] and to estimate the longitudinal velocity fields at open channel junctions [6]. In other cases, backpropagation as a multilayer perceptron was used in simulating the characteristics of open channel bends and subsequently used in prediction of flow parameters in 90° open channel arches [7], [8]. The network structure in this paradigm uses more than one layer (multi-layer) to change the weight associated with neurons in the hidden layer. Learning for ANN is a process in which free parameters of ANN are adapted through a continuous stimulation process by the environment in which the network is located [9]. ANN learns from its experience. The usual learning process includes three tasks, namely: 1) network output, 2) comparing the output with the desired target, and 3) adjusting the weight and repeating the process.

There are 12 training algorithms in the backpropagation model that can be used [10], namely the Fletcher-Reeves Update algorithm, Polak-Ribiere, Powell-Beale Restarts, Scaled Conjugate Gradient, Gradient Descent with Momentum and Adaptive Learning Rate, Resilent Backpropagation, BFGS, One Step Secant, Levenberg-Marquardt. Some researches related to the application of this training algorithm are [11]; [12]; [13]; [14]; [15]. Up to this stage a training algorithm has been implemented to help solve a case and has not yet been tested for other training algorithms.

Further testing is carried out by [16]; [17]; [18]; [19]; [20]. The testing was conducted on the twelve training algorithms and generated information that the Levenberg-Marquardt algorithm is the most optimal algorithm using 5, 10, and 15 neurons in the input layer. In the study 10 neurons were used in hidden layers. Meanwhile, the number of neurons in the hidden layer is very influential on network performance, especially in the error or MSE (Mean Squared Error) produced which has an impact on the level of accuracy of network output. MSE is known as a method that produces errors that are likely to be better for small errors, but sometimes make a big difference [21]. In theory, the more neurons in a hidden layer the more accurate the output is produced, but the network performance slows down even though the network speed in carrying out the training process is also influenced by the learning rate (lr) value used. Information about the number of neurons in the hidden layer that has the most optimal performance is unknown. Therefore, in this study an analysis and testing of the performance of the Levenberg– Marquardt training algorithm was conducted based on variations in the number of neurons in hidden layers and learning rate (lr).

II. METHOD

This research is a mixed method research in the form of developing computer programs with quantitative and qualitative testing using ANOVA statistical tests.

A. Research Variables

The research variables in the form of ANN parameters are the maximum epoch of 1000 (10^3), the value of lr = 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, target error = 0.001 (10^{-3}), input neurons (X) as many as 5, and 1 output neuron (Y), as well as 2, 4, 5, 7, 9 neurons in the hidden layer (Z). Network configuration is presented in Fig. 1.

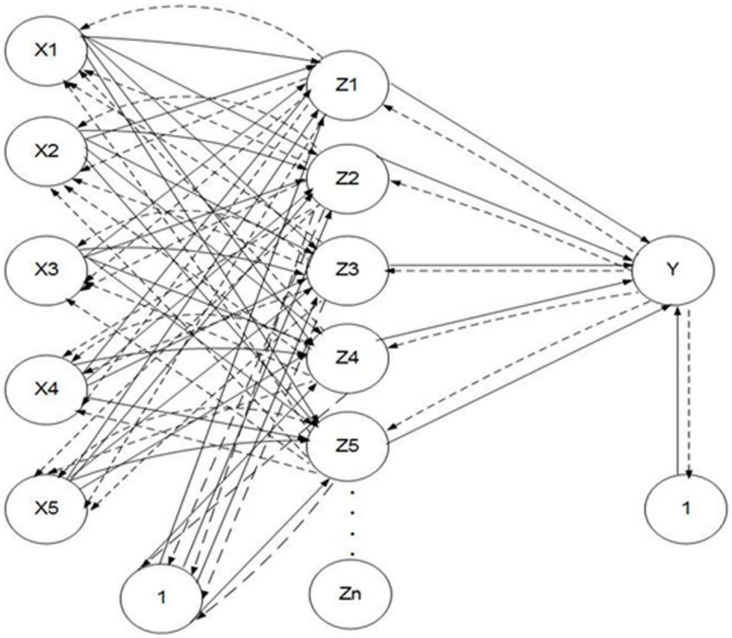

Fig. 1 Design of artificial neural networks with 5 input neurons, n neurons in hidden layers (n = 2, 4, 5, 7, 9) and 1 neuron in the output layer

B. Research Data

Network input data and targets are acquired from research [16].

C. Development of Computer Programs

The design of a computer program to obtain network output data is built as shown in Fig. 2.

D. Data Analysis

Output data of network generated by the Levenberg-Marquardt algorithm were analyzed using ANOVA statistical tests. Tests were carried out on many neurons in hidden layers (n = 2, 4, 5, 7, 9) at each learning rate (Fig. 3). Furthermore, from the results of this test, it is analyzed again to get the smallest MSE. The stages of the ANOVA test were carried out [22]:

1) Determine the hypothesis

H_0: there is no difference in error produced by the Levenberg-Marquardt algorithm on the number of n neurons in hidden layers (n = 2, 4, 5, 7, 9) for each value of lr.

H_1: there are differences in errors generated by the Levenberg-Marquardt algorithm on the number of n neurons in hidden layers (n = 2, 4, 5, 7, 9) for each value of lr

2) Determine the alpha value (α) (in this study used α = 5%)

3) Taking conclusions. Conclusions are taken based on the significant value obtained (sig.) With the provision H_0 rejected if the sig value<α and the provisions of H_0 are accepted if the value of sig≥α.

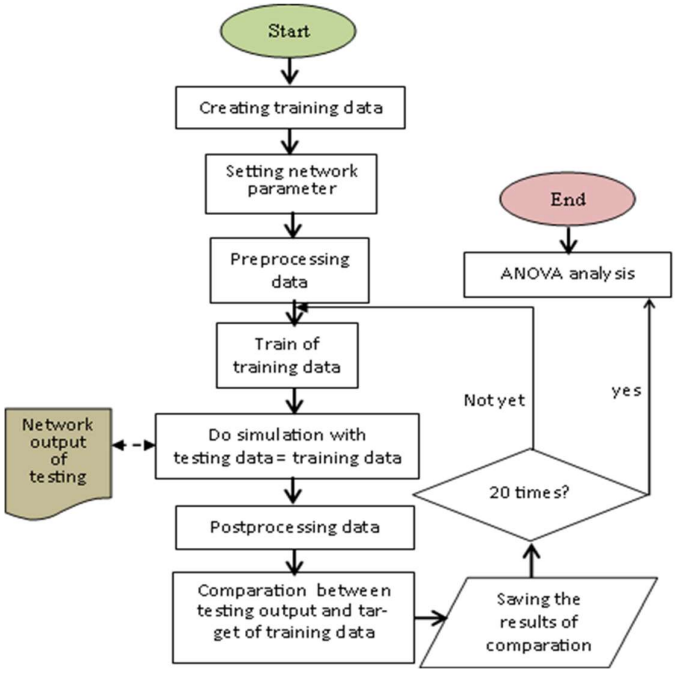

Fig. 2 Flowchart of ANN program development

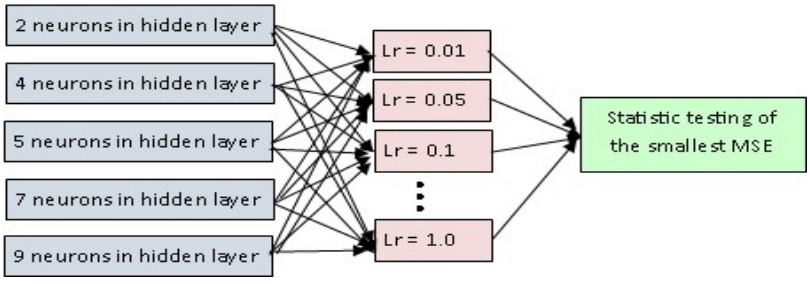

Fig. 3 Testing statistics design of many neurons n in hidden layers (n = 2, 4, 5, 7, 9) at each learning rate

III. RESULTS AND DISCUSSION

A. Research Data

Network input data (X) is the value of 5 neurons in the input layer and target (Y) are random data acquired from the research of [16] as in Table 1. Data input and target of network are run on the Levenberg-Marquardt algorithm to obtain MSE data. The Levenberg-Marquardt algorithm is run 20 times for each number of n neurons in the hidden layer and every lr as the design in Figure 2. The computer program was coded with MATLAB as in Fig. 4.

B. Data Analysis

ANOVA statistical tests were performed using SPSS software. The test results for errors generated by the Levenberg-Marquardt algorithm on the number of n neurons in the hidden layer for each value of lr with n = 2, 4, 5, 7, 9 are presented in Table 2, 3, 4, 5, and Table 6 respectively.

TABLE I
INPUT AND TARGET DATA

X_1	X_2	X_3	X_4	X_5	Y
9.5013	7.6210	6.1543	4.0571	0.5789	2.0277
2.3114	4.5647	7.9194	9.3547	3.5287	1.9872
6.0684	0.1850	9.2181	9.1690	8.1317	6.0379
4.8598	8.2141	7.3821	4.1027	0.0986	2.7219
8.9130	4.4470	1.7627	8.9365	1.3889	1.9881

```
clc;
disp('=======================================')
for i=1:20, fprintf('\nIterate: %d\n', i)
clear; data_5; //can be replaced with data_10 or data_15
P = P'; T

%running trainlm with 2 neurons in hidden layer
net = newff(minmax(P), [2 1], {'tansig' 'purelin'}, 'trainlm');

%first weights
FirstWeightInput = net.IW{1,1}; FirstWeightBiasInput = net.b{1,1};
FirstWeightLayer = net.LW{2,1}; FirstWeightBiasLayer = net.b{2,1};

%set training function
net.trainParam.epochs = 1000; net.trainParam.goal = 1e-3;
net.trainParam.lr = Learning_rate;
net.trainParam.min_grad = 1e-10; net.trainParam.show = 50;

% training function
net = train(net, P, T);

%end weights
EndWeightInput = net.IW{1,1}; EndWeightBiasInput = net.b{1,1};
EndWeightLayer = net.LW{2,1}; EndWeightBiasLayer = net.b{2,1};

%simulation
a = sim(net, P) fprintf('\n'); end
```

Fig. 4 Source code of ANN computer program

TABLE II
RESULTS OF ANOVA TEST ON 2 NEURONS IN A HIDDEN LAYER

	Sum of Squares	df	Mean Square	f	Sig.
Between Groups	5.657	11	.514	.547	.870
Within Groups	214.404	228	.940		
Total	220.061	239			

TABLE III
RESULTS OF ANOVA TEST ON 4 NEURONS IN A HIDDEN LAYER

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1.743	11	.158	1.052	.401
Within Groups	34.327	228	.151		
Total	36.070	239			

TABLE IV
RESULTS OF ANOVA TEST ON 5 NEURONS IN A HIDDEN LAYER

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	.494	11	.045	.858	.583
Within Groups	11.950	228	.052		
Total	12.444	239			

TABLE V
RESULTS OF ANOVA TEST ON 7 NEURONS IN A HIDDEN LAYER

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	.001	11	.000	1.575	.107
Within Groups	.009	228	.000		
Total	.010	239			

TABLE VI
RESULTS OF ANOVA TEST ON 9 NEURONS IN A HIDDEN LAYER

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	.000	11	.000	1.202	.286
Within Groups	.000	228	.000		
Total	.000	239			

Based on Tables 2, 3, 4, 5, and Table 6, there are 5 Sig. all of which are ≥ α (= 5%) so that H0 is accepted. In accordance with the proposed hypothesis, there is no significant difference in MSE for each n neuron in the hidden layer (n = 2, 4, 5, 7, 9) based on the learning rate. However, the average MSE value generated by the Levenberg-Marquardt algorithm for each of the n neurons in each learning rate (lr) can be known through descriptive analysis. The results of the data description are presented in Table 7.

Table 7 shows the difference in the smallest error rate for each value of lr and the number of neurons in the hidden layer. The data in blue in the table shows the smallest MSE value for each number of neurons in HL at the corresponding lr value. Overall, the smallest error (MSE) was achieved on 9 neurons in the hidden layer with learning rate = 0.5. The MSE value is 0,00014 ± 0,00018. This result is in line with research conducted by [23] which gives the smallest MSE value achieved by the LM algorithm of 0.00019584038 ± 0.000239300998. The MSE results were achieved using a different test direction. In this study, testing was carried out on the number of neurons in HL for each value of lr used. While in research of [23], testing is performed on each value of lr used for each number of neurons in HL.

The MSE difference that occurs is suspected to be a correlation between the value of lr and MSE. Therefore a correlation test is performed using the Pearson method and produce data as in Table 8. From Table 8 can be seen that the correlation between learning rate (lr) and MSE is -0.048. This means that the correlation between lr and MSE is very small and inversely correlated. The greater the value of lr, the smaller the MSE. Because the value of sig.> α (= 5%), it can be said that there is no significant correlation between lr and MSE. This is in line with the results of research by [24] which states that there is no correlation between MSE and lr in backpropagation networks using 10 neurons in hidden layers.

In the studies mentioned, the Levenberg - Marquardt algorithm provides the smallest MSE value compared to other training algorithms. This is reasonable because the algorithm uses a Newtonian method that is very fast and accurate to get the minimum error [10].

TABLE VII
MEAN OF MSE FOR EACH LEARNING RATE (LR) FOR N NEURONS IN A HIDDEN LAYER (N = 2, 4, 5, 7, 9)

No.	lr	2 neurons	4 neurons	5 neurons	7 neurons	9 neurons
		Mean ± stdev.	Mean ± stdev.	Mean ± stdev.	Mean ± stdev.	Mean ± stdev.
1	0.01	0.74591±1.12645	0.00844±0.01963	0.08880±0.36589	0.00040±0.00031	0.00038±0.00028
2	0.05	0.49334±0.95028	0.08763±0.36596	0.00294±0.01202	0.00596±0.01814	0.00016±0.00021
3	0.1	0.66188±0.98916	0.08855±0.36588	0.00029±0.00035	0.00016±0.00016	0.00024±0.00032
4	0.2	0.50436±0.93995	0.12977±0.54840	0.00027±0.00032	0.00020±0.00028	0.00023±0.00030
5	0.3	0.67330±1.04447	0.12517±0.49946	0.00295±0.01201	0.00290±0.01203	0.00023±0.00023
6	0.4	0.72417±1.09239	0.01342±0.02758	0.00407±0.01744	0.00017±0.00028	0.00023±0.00023
7	0.5	0.55946±0.95146	0.12582±0.54907	0.00016±0.00022	0.00008±0.00014	0.00014±0.00018
8	0.6	0.59960±1.02243	0.00803±0.02399	0.00303±0.01200	0.00027±0.00030	0.00028±0.00034
9	0.7	0.47074±0.92942	0.00416±0.01742	0.11422±0.49708	0.00027±0.00032	0.00018±0.00021
10	0.8	0.65068±0.99104	0.09129±0.36541	0.11422±0.49708	0.00028±0.00032	0.00020±0.00026
11	0.9	0.54508±0.93392	0.00931±0.02250	0.00299±0.01200	0.00028±0.00032	0.00030±0.00032
12	1	0.14428±0.54541	0.31443±0.74240	0.00024±0.00027	0.00017±0.00022	0.00025±0.00030

TABLE VIII
CORRELATIONS TEST BETWEEN LEARNING RATE VALUES AND MSE

		learning rate	MSE
learning rate	Pearson Correlation	1	-.048
	Sig. (2-tailed)		,715
	N	60	60
MSE	Pearson Correlation	-,048	1
	Sig. (2-tailed)	.715	
	N	60	60

IV. CONCLUSION

Based on the results of the research that has been done, it can be concluded that the Levenberg–Marquardt training algorithm has the best performance when using 9 neurons in the hidden layer and lr = 0.5. This performance is indicated by the MSE value of 0.00014 ± 0.00018 from the target error 0.001. With information generated from this study, the Levenberg–Marquardt training algorithm can be used as an alternative in the development of ANN-based applications.

ACKNOWLEDGEMENTS

Authors thank DP2M - The Ministry of Research, Technology, and Directorate General of Higher Education (DIKTI) through KOPERTIS Region VI, who has provided funds and Universitas Muhammadiyah Purwokerto for providing facilities in the implementation of this research.

REFERENCES

[1] S. Kusumadewi and S. Hartati, *Neuro-Fuzzy : Integrasi Sistem Fuzzy dan Jaringan Syaraf*. Yogyakarta: Graha Ilmu, 2006.

[2] S. Shanmuganathan and S. Samarasinghe, *Artificial Neural Network Modelling*, vol. 628. Springer International Publishing, 2016.

[3] M. T. T. Jones, *Artificial Intelligence A Systems Approach*. New Delhi: Infinity Science Press LLC, 2008.

[4] J. J. Siang, *Jaringan Syaraf Tiruan dan Pemrogramannya Menggunakan MATLAB*. Yogyakarta: ANDI, 2009.

[5] A. B. Adetunji, A. Q. Ayinde, and C. O. Akanbi, "Application of Neural Network to Detect Intrusion in Banking System.," *Am. J. Sci. Ind. Res.*, vol. 5, no. 2, pp. 53–59, 2014.

[6] A. H. Zaji and H. Bonakdari, "Application of artificial neural network and genetic programming models for estimating the longitudinal velocity field in open channel junctions," *Flow Meas. Instrum.*, vol. 41, pp. 81–89,

2015.

[7] A. Gholami, H. Bonakdari, A. H. Zaji, and A. A. Akhtari, "Simulation of open channel bend characteristics using computational fluid dynamics and artificial neural networks," *Eng. Appl. Comput. Fluid Mech.*, vol. 9, no. 1, pp. 355–369, 2015.

[8] A. Gholami, H. Bonakdari, A. H. Zaji, S. Ajeel Fenjan, and A. A. Akhtari, "Design of modified structure multi-layer perceptron networks based on decision trees for the prediction of flow parameters in 90° open-channel bends," *Eng. Appl. Comput. Fluid Mech.*, vol. 10, no. 1, pp. 193–208, 2016.

[9] A. Desiani and M. Arhami, *Konsep kecerdasan buatan*. Yogyakarta: Andi Offset, 2006.

[10] S. Kusumadewi, *Membangun Jaringan Syaraf Tiruan Menggunakan MATLAB & EXCEL LINK*. Yogyakarta: Graha Ilmu, 2004.

[11] H. Harjono and D. Aryanto, "Application of Artificial Neural Networks to Predict Student Achievement Study," *SAINTEKS*, vol. 5, no. 2, 2009.

[12] H. Mustafidah, D. K. Hakim, and S. Sugiyanto, "Tingkat Keoptimalan Algoritma Pelatihan pada Jaringan Syaraf Tiruan (Studi Kasus Prediksi Prestasi Belajar Mahasiswa) Optimization Level of Training Algorithms in Neural Network (Case Studies of Student Learning Achievement Predictions)," *JUITA*, vol. II, no. 3, pp. 159–166, 2013.

[13] H. Mustafidah, D. Aryanto, and D. K. Hakim, "Uji Optimalisasi Algoritma Pelatihan Conjugate Gradient pada Jaringan Syaraf Tiruan," in *Prosiding SENATEK, ISBN: 978-602-14355-0-2, 21 September 2013*, 2013, p. B-9-1.

[14] F. Wibowo, S. Sugiyanto, and H. Mustafidah, "Tingkat Ketelitian Pengenalan Pola Data pada Algoritma Pelatihan Perbaikan Metode Batch Mode dalam Jaringan Syaraf Tiruan," *JUITA (Jurnal Inform.*, vol. II, no. 4, pp. 259 – 264, 2013.

[15] H. Mustafidah, S. Hartati, R. Wardoyo, and A. Harjoko, "Prediction of Test Items Validity Using Artificial Neural Network," in *Proceeding International Conference on Education, Technology, and Science (NETS) 2013, "Improving The Quality Of Education To Face The Impact Of Technology". December 28th, 2013*, 2013.

[16] H. Mustafidah and S. Suwarsito, "Error Rate Testing of Training Algorithm in Back Propagation Network," *Int. J. Soft Comput. Eng.*, vol. 5, no. 4, pp. 46 – 50, 2015.

[17] H. Mustafidah and S. Suwarsito, "Model Parameter Jaringan Syaraf Tiruan untuk Pemilihan Algoritma Pelatihan Jaringan Backpropagation yang Paling Optimal," Purwokerto, Central Java, Indonesia, 2015.

[18] H. Mustafidah and S. Suwarsito, "Uji Keoptimalan Algoritma Pelatihan pada Jaringan Syaraf Tiruan," in *Prosiding Seminar Nasional SENATKOM 2015*, 2015, pp. 243–248.

[19] H. Mustafidah and S. Suwarsito, "Inferensi Tingkat Kesalahan dalam Jaringan Backpropagation Berdasarkan Laju Pemahaman," in *Prosiding Seminar Nasional Aptikom 2016 (SEMNASTIKOM), Hotel Lombok Raya Mataram, 28 – 29 Oktober 2016*, 2016, pp. 576–580.

[20] H. Mustafidah and H. Harjono, "Korelasi Tingkat Kesalahan dan Epoh dalam Jaringan Backpropagation," in *Prosiding SEMNASTIKOM 2017, 3 November 2017, ISBN: 978-602-50434-0-6*, 2017, pp. 55–61.

[21] E. L. Lehmann and G. Casella, *Theory of Point Estimation*, Springer t. Springer, 2003.

[22] T. Taniredja and H. Mustafidah, *Penelitian Kuantitatif (Sebuah Pengantar)*. Bandung: ALFABETA, 2011.

[23] H. Mustafidah, Suwarsito, and S. N. C. Permatasari, "Accuracy of the neurons number in the hidden layer of the levenberg-marquardt algorithm," *Int. J. Recent Technol. Eng.*, vol. 8, no. 4, pp. 2349–2353, 2019.

[24] H. Mustafidah and S. Suwarsito, "Correlation Analysis Between Error Rate of Output and Learning Rate in Backpropagation Network," *Adv. Sci. Lett.*, vol. 24, no. 12, pp. 9182–9185, 2018.