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

Hidayati Mustafidah¹, Suwarsito²

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

¹fida.mustafidah@ump.ac.id
²suwarsito.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, Resilient 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 \((\alpha)\) (in this study used \(\alpha = 5\%\))

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

![Fig. 2 Flowchart of ANN program development](image)

![Fig. 3 Testing statistics design of many neurons n in hidden layers (n = 2, 4, 5, 7, 9) at each learning rate](image)
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>
<thead>
<tr>
<th>TABLE I</th>
<th>INPUT AND TARGET DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>X2</td>
</tr>
<tr>
<td>9.5013</td>
<td>7.6210</td>
</tr>
<tr>
<td>2.3114</td>
<td>4.5647</td>
</tr>
<tr>
<td>6.0684</td>
<td>0.1850</td>
</tr>
<tr>
<td>4.8598</td>
<td>8.2141</td>
</tr>
<tr>
<td>8.9130</td>
<td>4.4470</td>
</tr>
</tbody>
</table>

Fig. 4 Source code of ANN computer program

<table>
<thead>
<tr>
<th>TABLE II</th>
<th>RESULTS OF ANOVA TEST ON 2 NEURONS IN A HIDDEN LAYER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum of Squares</td>
<td>df</td>
</tr>
<tr>
<td>Between Groups</td>
<td>5.657</td>
</tr>
<tr>
<td>Within Groups</td>
<td>214.404</td>
</tr>
<tr>
<td>Total</td>
<td>220.061</td>
</tr>
</tbody>
</table>
Based on Tables 2, 3, 4, 5, and Table 6, there are 5 Sig. all of which are $\geq \alpha (= 5\%)$ so that $H_0$ 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 $0.00014 \pm 0.00018$. This result is in line with research conducted by [23] which gives the smallest MSE value achieved by the LM algorithm of $0.0001984038 \pm 0.000239300998$. 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.$> \alpha (= 5\%)$, 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].
### Performance of Levenberg-Marquardt …

<table>
<thead>
<tr>
<th>No.</th>
<th>lr</th>
<th>2 neurons Mean ± stddev.</th>
<th>4 neurons Mean ± stddev.</th>
<th>5 neurons Mean ± stddev.</th>
<th>7 neurons Mean ± stddev.</th>
<th>9 neurons Mean ± stddev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.01</td>
<td>0.74591±1.12645</td>
<td>0.00844±0.01963</td>
<td>0.08880±0.36589</td>
<td>0.00040±0.00031</td>
<td>0.00038±0.00028</td>
</tr>
<tr>
<td>2</td>
<td>0.05</td>
<td>0.49334±0.95028</td>
<td>0.08763±0.36596</td>
<td>0.00294±0.01202</td>
<td>0.00596±0.01814</td>
<td>0.00016±0.00021</td>
</tr>
<tr>
<td>3</td>
<td>0.1</td>
<td>0.66188±0.98916</td>
<td>0.0855±0.36588</td>
<td>0.00029±0.00035</td>
<td>0.0016±0.00016</td>
<td>0.00024±0.00032</td>
</tr>
<tr>
<td>4</td>
<td>0.2</td>
<td>0.50436±0.93995</td>
<td>0.12977±0.54840</td>
<td>0.00027±0.00032</td>
<td>0.00020±0.00028</td>
<td>0.00023±0.00030</td>
</tr>
<tr>
<td>5</td>
<td>0.3</td>
<td>0.67330±0.104447</td>
<td>0.12517±0.49946</td>
<td>0.00295±0.01201</td>
<td>0.00290±0.01203</td>
<td>0.00023±0.00023</td>
</tr>
<tr>
<td>6</td>
<td>0.4</td>
<td>0.72417±0.109239</td>
<td>0.01342±0.02758</td>
<td>0.00407±0.01744</td>
<td>0.00017±0.00028</td>
<td>0.00023±0.00023</td>
</tr>
<tr>
<td>7</td>
<td>0.5</td>
<td>0.55946±0.95146</td>
<td>0.12582±0.54907</td>
<td>0.00016±0.00022</td>
<td>0.00008±0.00014</td>
<td>0.00014±0.00018</td>
</tr>
<tr>
<td>8</td>
<td>0.6</td>
<td>0.59960±0.102243</td>
<td>0.00803±0.02399</td>
<td>0.00303±0.01200</td>
<td>0.00027±0.00030</td>
<td>0.00028±0.00034</td>
</tr>
<tr>
<td>9</td>
<td>0.7</td>
<td>0.47074±0.92942</td>
<td>0.00416±0.01742</td>
<td>0.11422±0.49708</td>
<td>0.00027±0.00032</td>
<td>0.00018±0.00021</td>
</tr>
<tr>
<td>10</td>
<td>0.8</td>
<td>0.65068±0.99104</td>
<td>0.09129±0.36541</td>
<td>0.11422±0.49708</td>
<td>0.00028±0.00032</td>
<td>0.00020±0.00026</td>
</tr>
<tr>
<td>11</td>
<td>0.9</td>
<td>0.54508±0.93392</td>
<td>0.00931±0.02250</td>
<td>0.00299±0.01200</td>
<td>0.00028±0.00032</td>
<td>0.00030±0.00032</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>0.14428±0.54541</td>
<td>0.31443±0.74240</td>
<td>0.00024±0.00027</td>
<td>0.00017±0.00022</td>
<td>0.00025±0.00030</td>
</tr>
</tbody>
</table>

### 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


Performance of Levenberg-Marquardt ... | Mustafidah, H., Suwarsito, 29 – 35


