INDONESIAN JOURNAL OF ENGINEERING RESEARCH
2022 Vol. 3, No. 1, 1 - 10
https://doi.org/10.11594/ijer.v3i1.40

Research Article

Flood Detection Using Backpropagation Neural Network Method

Ani Dwi Ratnasari[1], Muhammad Khoirul Hasin[1], Subagio Soim[2], Bambang Teguh Setiawan[3], A.Z. Arfianto[1]

[1]Dept. of Marine Electrical Engineering, Politeknik Perkapalan Negeri Surabaya, Surabaya, Indonesia
[2]Dept. of Mechanical Engineering, Politeknik Perkapalan Negeri Surabaya, Surabaya, Indonesia
[3] Dept. of Shipbuilding, Politeknik Perkapalan Negeri Surabaya, Surabaya, Indonesia

**Corresponding author:*
E-mail:
khoirul.hasin@ppns.ac.id

Abstract

Lack of river and watershed management will cause problems and disasters. One of it is the flood that can cause physical, social and economic loss. So countermeasures or flood anticipation are needed by using the Early Warning System (EWS) to provide early information if a flood is going to occur. This study uses five input indicators: temperature, humidity, water discharge, water surface altitude and rainfall data that will produce output in the form of notifications and alarms for the Early Warning System (EWS). Then the input and output data configuration will be processed using a Backpropagation Neural Network. Data used is data recorded in real-time on the research object for two weeks with the composition of training and testing data with a percentage of 80% and 20%. The best backpropagation neural network model used has the input of 5 neurons layer architecture, 15 neurons as the hidden layer and three neurons as the output layer. The flood prediction result uses the Backpropagation Neural Network method, has an RMSE score performance of 2.16e-21 and a percentage success testing system of 91.33%. It shows that the model has an excellent accuracy level.

Keywords: Backpropagation Neural Network, flood, prediction

INTRODUCTION

One news page (voaindonesia.com) on 26/08/19 launched that there are 458 watersheds in Indonesia, and from that amount, 108 watersheds have been categorized as critical. The critical category is caused by pollution and lack of water management which can cause floods. So, the flood must be immediately handled effectively and efficiently. One countermeasure or anticipation of the widespread flood is to use IoT, which is equipped with a backpropagation neural network to provide early information about the flood. In this study, the authors focus on the method used for IoT implementation, which will be discussed in the following article. In this method, the author uses five input indicators called temperature, humidity, water discharge, water surface's altitude and rainfall to produce output in the form of a decision to bring up notifications and alarms. This research aims to know the optimal neural network architecture for detecting floods.

LITERATURE REVIEW

A. Previous research

Flood detection topics have already been discussed in some articles in Table I.

Table 1. Summary of Previous Research

Title	Method	Results
Tsukamoto Fuzzy Inference System Modeling For Prediction of Flood In Malang City ".	Fuzzy Inference System Tsukamoto	The test results show that the total flood data in 2016-2017 generated a small RMSE error value of 2.76. Then, by using the result data estimation of bulk rain and intensity for the next three years, Tsukamoto's FIS modelling can be implemented to predict the amount of flood in Malang City for the next three years (2018-2020)
Citarum River Flood Prediction With Fuzzy Logic Algorithm Results Particle Swarm Optimization	Particle Swarm Optimization Algorithm	Swarm algorithm optimization particles and samples used is bulk rain and high water level, obtained accuracy of flood prediction by 73% based on results metrix calculation
Algorithm Evolving Neural Network For Rainfall Prediction Application	Evolving Neural Network	The research results were compared to BPNN BMKG's test and prediction results. From research conducted the step early for test and measurement, results from

		this ENN have predicted bulk rain with a level accuracy prediction of 85%.
Prediction of Water Level (TMA) For Disaster Early Detection Flood Using SVR-TVIWPSO	SVR-TVIWPSO	Test results from 10 different monthly data show that the slightest error value of 0.00755 was obtained using Mean Absolute Error for June 2007 data with the integration SVR-TVIWPSO method.
Application Prediction of Rainfall, Water Discharge, and Genesis Flood Web Based with Machine Learning in Deli Serdang	Learning Machine	By using daily bulk rain and water discharge data from January 1, 2016, to December 31, 2017, the accurate prediction of flood incidents or no use application is 94.4%. However, the application is not yet capable predict the day flood happens.

So, in this research, the author tries to use another method called backpropagation neural network with five inputs. They are temperature, humidity, water discharge, surface altitude, and rainfall. Also, the two outputs are the decision to raise alarms and notifications.

B. Backpropagation Neural Network

Backpropagation neural network has a strong ability from non-linear interpolation. This method is used mainly in Engineering applications for prediction and optimization. The back propagation neural network consists of the input layer, several hidden layers and output layers (Fadhilah and Ginardi, 2017).

Backpropagation algorithm network nerves' clone can minimize error because there is a hidden layer. It occurred because the hidden layer in the Backpropagation algorithm functioned as a place to update and customize weight so that it could get a new score and train close to the expected target. Epoch is one of the weight update or training iterations in backpropagation neural networks (Fitradini et al., 2020).

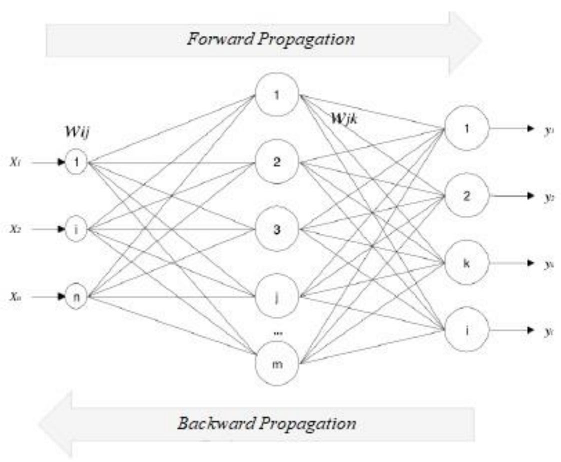


Fig. 1. Backpropagation Neural Network architecture (Abas, Syukur and Soeleman, 2017)

The backpropagation network consists of several layers. The input layer with n neurons (plus a bias), a hidden layer consisting of over j neurons (plus a bias), as well as k outputs to which each neuron in a layer is fully connected to each neuron in the layer above it or below, except on the bias, it only fully connected of neuron layers above it as shown by the backpropagation architecture in Figure 1.

Figure 1 shows a network with n input layer x, hidden layer j and output layer k, and bias value, i.e., something enters with the permanent weight of 1. As seen in Figure 1, Wij is the weight line from xi input layer to hidden layer j (is the weight of the line connecting the bias in the input layer to the hidden layer j). Wjk is a value from hidden layer j to output layer k (is weight from bias in the hidden layer to output layer k) (Sari, 2016). There are 3 phases in backpropagation training. They are phase forward (feed-forward), backward (backpropagation), and phase modification value. The value calculation formula is as follows (Fadhilah and Ginardi, 2017):

$$W_{lji}(\mathrm{K}+1) = W_{lji}\,(k) - \mu \frac{\partial MSE(w)}{\partial Wlji}$$

Where W_{lji} is the weight between neuron i in layer l-1 and neuron j in layer l, μ is positive numbers, the learning rate for control of the learning step algorithm is usually set small. In the backpropagation phase, each output unit receives a corresponding pattern target with an input pattern for calculated score error.

Error the will propagated to back off. At the same time, the weight of phase modification aims to lower errors that occur. Three phases the is repeated until the termination condition is fulfilled.

In backpropagation, function activation needs to meet a number of conditions: continuous, easily differentiable, and non-descending function. The frequent function used in backpropagation activation (Sari, 2016) is a binary sigmoid function with a range (0, 1).

$$f_1(x) = \frac{1}{1+e^{-x}}$$

with derivative

$$f_1(x) = f_1(x)\,(1 - f_1(x))$$

Another frequent function used is a bipolar sigmoid function that forms similar to a binary sigmoid function but with a range (-1, 1).

$$f_2(x) = 2f_1 - 1$$

with derivative

$$f_2(x) = \frac{1}{2}\,(1 + f_2(x))(1 - f_2(x))$$

According to Siang (2009: 102) (Sari, 2016), the training algorithm for networks with one hidden layer (with sigmoid activation binary function) is as follows :

Step 0. Initialize all weights with small random numbers.
Step 1. If the termination condition is not yet fulfilled, do steps 2-9.
Step 2. For each pair of training data, do step 3-8.

Phase I: Forward Propagation.

Step 3. Each input layer (xi, i = 1,2,... .,n) receives xi's input signal and pass it to the hidden layer.
Step 4 . Count all output in hidden layer zj (j = 1,2 ,... ,p).

$$z_{in_j} = v_{0j} + \sum_{i=1}^{n} x_i v_{ij}$$

Apply activation function to count output signal.

$$z_j = f\left(z_{in_j}\right) = \frac{1}{1 + e^{-z\_in_j}}$$

Send signal to all output layers.
Step 5. Count all outputs in the output layer yk (k = 1,2, ...,m)

$$y_{in_k} = w_{0k} + \sum_{j=1}^{p} z_j w_{jk}$$

Apply activation function to count output signal.

$$y_k = f\left(y_{in_k}\right) = \frac{1}{1 + e^{-y\_in_k}}$$

Phase II: Backward Propagation

Step 6. Count the δ factor on the output layer based on an error in each output layer yk (k = 1,2,...,).

$$\delta_k = (t_k - y_k) f'\left(y_{in_k}\right) = (t_k - y_k) y_k (1 - y_k)$$

δk is the unit of error that will be used in changing the weight of the layer below it (step 7). Calculate the rate of change of weight wkj (which will be used to change weight $w$jk) with α as the speed acceleration.

$$\Delta w_{jk} = \alpha \delta_{kzj};\ (\mathrm{k} = 1,2, \dots, \mathrm{m};\ \mathrm{j} = 0,1, \dots, \mathrm{p})$$

Count change in bias (which will used to change weight w_{ok})

$$\Delta w_{ok} = \alpha \delta_k;\ (\mathrm{k} = 1,2, \dots, \mathrm{m})$$

Send δ_k to the layer below

Step 7. Count factor on the hidden layer based on error in each hidden zj layer unit (j = 1,2 ,... ,).

$$\delta_{net_j} = \sum_{k=1}^{m} \delta_k w_{jk}$$

Factor δ hidden layers:

$$\delta_j = \delta'_{netjf}\left(\delta_{net_j}\right) = \delta\_net_j z_j (1 - z_j)$$

Count the weight changes vij (used later to change the weight vij)

$$\Delta v_{ij} = \alpha \delta_j x_i;\ (\mathrm{j} = 1,2, \dots, \mathrm{p};\ \mathrm{i} = 0,1, \dots, \mathrm{n})$$

Phase III: Weight Changes

Step 8. Count all the weight changes.
Weights of the lines' changes lead to the output layer.

$$w_{jk}(new\ one) = w_{jk}\ (old\ one) + \Delta w_{jk}, (\mathrm{k} = 1,2, \dots, \mathrm{m}\ ;\ \mathrm{j} = 0,1, \dots, \mathrm{p})$$

Weights of the lines's changes leading to the hidden layer.

$$v_{ij}(new\ one) = v_{ij}\ (old\ one) + \Delta v_{ij}, (\mathrm{j} = 1, \dots, \mathrm{m}\ ;\ \mathrm{i} = 1,2, \dots, \mathrm{n})$$

RESEARCH METHODS

A. *Software Design and Development*

Figure 2 shows the flowchart of software design and development. The first step is to get the dataset from a specified location and enter it into the database. Performed data normalization to get good data (there is no outlier). Next, determine the weight's initial value, the training data percentage, and data testing. Available training data is processed in the forward propagation and backpropagation phases. These two processes are repeated until the error rate comes out smaller from the MSE. In the end, classification results can be found: not flooded, will floods, and floods.

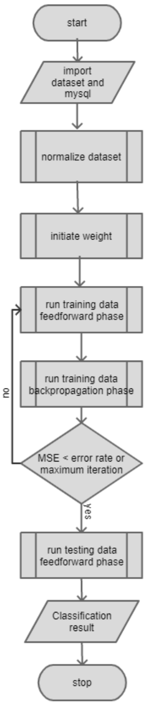


Fig. 2. Backpropagation neural network flowchart

B. *Dataset Retrieval and Data Collection*

Dataset retrieval is carried out along Bengawan Jero River every 5 minutes interval. The proportion of the dataset to be processed is 50:50. Half of the data is taken in rainy conditions, and the other half is taken when it is not rainy. Datasets are later grouped and classified based on condition parameters, as in Table II.

Temperature parameters taken from the DHT11 sensor data with units of degrees Celsius were then classified into three groups based on temperature data daily reported by the bmkg.go.id page in the Lamongan Regency.

Humidity parameters taken from the DHT11 sensor data with units of percent later classified into three groups based on humidity data daily reported by the bmkg.go.id page in the Lamongan Regency.

Bulk parameters of rain were taken from Rain Gauge sensor data in millimeters (mm), then classified into three groups based on previous research data.

Water discharge parameter is taken from ultrasonic sensor data or HC-SR04 in millimeter (mm), which is then classified into three groups based on hardware analysis design built on this research. Water level distance will be displayed on the interface, while measured water level will be used for the training process. It is based on the river watershed's high (in this case is 1.5 meters).

Water discharge parameters are taken from the YF-S201 water flow sensor data in millimeters per second (ml/s) which are then classified into three groups based on the hardware analysis design and sensor datasheet that the authors will build on this research.

The next stage is the process of data collection. The way to collect data is by recording or retrieving data directly. The data needed are temperature, humidity, water level distance, water discharge and the bulk rain that occurred at the research site determined along Bengawan Solo's river branch for approximately three weeks.

Five data sets will be used as input: temperature, humidity, water level distance, water discharge and bulk rain. The actual data is in the Appendix.

Table 2. Parameters Qualification Temperature Humadity, Bulk Rain, Water Level Distance, and Water Dischage

Temperature (°C)	Description
24 - 26	Low
>26-30	Medium
>30-34	High

Humidity (%)	Description
65 – 75	Low
>75 – 85	Medium
>85 – 95	High

Rainfall (%)	Description
0 – 1000	Low
>1000 – 1500	Medium
>1500	High

Water level distance (mm)	Description
500 – 400	Low
>400 – 150	Medium
>150	High

Water discharge (ml/s)	Description
0 – 100	Low
>100 – 300	Medium
>300 – 500	High

C. *Data Labeling and Classification Process*

In this section, data labeling is carried out to make it easier in applicant Provision Classification that has been made previously to determine the output parameters of the later training data used to determine the optimal Backpropagation Neural Network model in search appropriate weights and small MSE values. The data labeling process in the Microsoft Excel application uses the IF function based on Provision Classification for each parameter specified in Table II. The data processing is performed by adding formulas or functions to every data column used under the Terms Classification or Provisions that have been made before. Table III is a data example before and after the labeling process.

After performing the labeling process on the data to determine the classification to be used to the output based on Provision Classification, the following process is to define output into three categories: No Flood, Will be Flood and Flood. Table IV shows the output criteria.

Terms of Data Processing and Classification also use Microsoft Excel to determine output criteria. Data earned after the labeling process is then matched with terms of classification that have been made before. Table V is the result of the implementation of Provision Classification on datasets.

Table 3. Sample Data Before and After Labeling Process

Temperature	Humidity	Distance Water	Water Discharge	Bulk Rain
Before				
29.5	89	378	0	0

After				
Medium	High	Medium	Low	Low

Table 4. Three Output Category

Output	Criteria
0	Not Flood
1	Will be Flood
2	Flood

Table 5. Example of Implementation Results Provision Classification

Temperature	Humidity	Distance Water	Water Discharge	Bulk Rain	Output	
					Criteria	Results
Medium	High	Medium	Low	Low	Not Flood	0

D. *Normalization, Categorization and Data Sharing*

The data normalization process is one of the crucial techniques to understand in data preprocessing, which aims to create several range variables that have the same value, nothing too big nor too small, so statistics analysis becomes manageable. Several data normalization methods include simple feature scaling, min-max and z-score—this research uses the method min-max data normalization using the MinMaxScaler library from scikit-learn. The min-max method works, i.e., every value on a feature minus the minimum value of that feature, then divided by range score or maximum score reduced by the minimum value of that feature. This method will also produce a score of new normalization result between 0 to 1. The equation of the min-max method can be formulated as :

$$x_{new} = \frac{x_{old} - x_{min}}{x_{max} - x_{min}}$$

Representation application method min-max data normalization based on the above equation can be seen in Table VI.

Based on Table VI Scatter Plot on each type of data parameter to be used, can is known that the normalized data with the same scale to the output can represent the original data well without changing the character of these data. Data normalization will be used for the continuation of the following process.

Data categorical are gathering information in the form of a group. This research will use the One-Hot Encoding technique from the scikit-learn library. One-Hot encoding is a technique that changes every value in a column into a new column and fills it with binary values, i.e., 0 and 1. The author uses categorical encoding because the technique of statistics or learning equations only accepts numeric scores, not value categorial. So from that, the categorial data must be changed more formerly into numerical data so that Machine Learning models can process the data.

Frequent categorical data types are shared into 2 (Ordinal and Nominal). Ordinal data is a data where there is an element of ordering intrinsic. An example of ordinal data is when the data score are Low, Medium, and High. On the other hand, nominal data does not have an intrinsic element—for example, the city's name, such as Jakarta, Bandung, or Bali. Because ordinal and nominal data have different characteristics, the treatment given to both types is also different.

The Nominal Data such as the output class are: there will be a flood, flood, and not flood that can use One Hot Enconding. Table VII is an output label transformation before and after One Hot Enconding is done.

The data used in this research is the data record of sensor readings consisting of temperature, humidity, water surface distance, water discharge and bulk rain. Then the data is shared into training and data test with a ratio of 80%: to 20%. The total data is 1412, of which the training data is 1130, and the test data is 282.

Table 6. Representation Application Method Min-Max Data Normalization

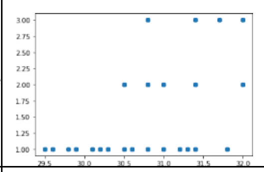
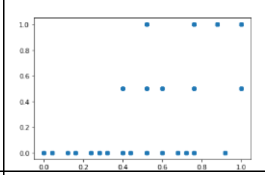
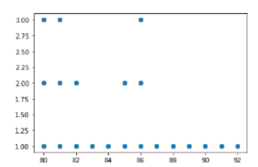
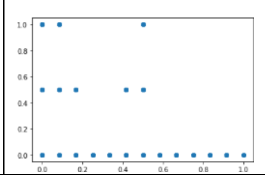
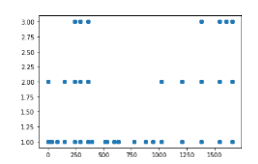
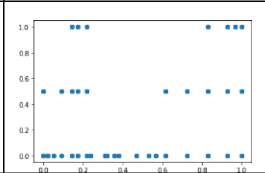
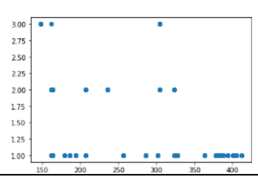
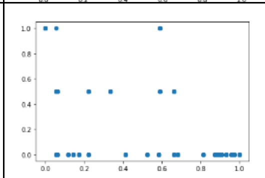
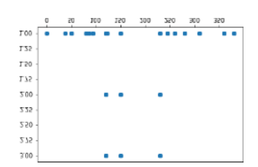
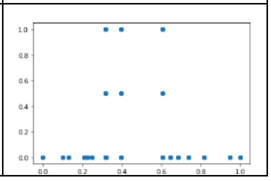
Data	Before Normalization	After Normalization
Temperature		
Humidity		
Bulk Rain		
Water Surface Distance		
Water Discharge		

Table 7. Output Data After Done One-Hot Encoding

No	Output	One Hot Encoding		
1	0	1	0	0
2	0	1	0	0
3	0	1	0	0
4	0	1	0	0
5	0	1	0	0
6	0	1	0	0
7	0	1	0	0
8	0	1	0	0
9	0	1	0	0
10	0	1	0	0

RESULTS AND DISCUSSION

After carrying out the design process backpropagation neural network, then the next step is to do testing to the data training process, testing predictive data with the original dataset, initialize weight as well as determine architecture or which model is suitable as well as have the smallest MSE value so that could make the accurate prediction. The testing process is conducted to know which architecture will be implemented into the system prototype to perform an accurate prediction process. Following is test data starting from the training process with several architectures, predictive data testing process with the original dataset, initialize weight from the model or best architecture, and determine model level accuracy.

A. *BPNN Training Data*

After knowing the data used in the training and testing process, data used during the training process in this section will be displayed. For the process of reading and uploading data in Python, the Pandas library is used with the file format in the form of .xlsx or the Microsoft Excel format. Table VIII shows data used in the process.

Table 8. Training Dataset

No	Temperature	Humidity	Water Distance	Water Discharge	Bulk Rain	Y1	Y2	Y3
1	29.5	89	378	0	0	1	0	0
2	29.5	89	378	0	0	1	0	0
3	29.5	89	378	0	0	1	0	0
4	29.5	89	378	0	0	1	0	0
5	29.5	89	378	86	0	1	0	0
6	29.5	89	378	86	0	1	0	0
7	29.5	89	378	86	0	1	0	0
8	29.5	89	378	86	0	1	0	0
9	29.6	89	378	86	0	1	0	0
10	29.6	89	378	86	0	1	0	0
	.....	.....	.....	.....	.....	.....	.....	.....
	.....	.....	.....	.....	.....	.....	.....	.....
	.....	.....	.....	.....	.....	.....	.....	.....
1121	30.5	84	302	380	288.5924	0	0	1
1122	30.5	84	302	380	288.5924	0	0	1
1123	30.5	85	302	380	288.5924	0	0	1

B. *Training Parameter Design*

In this study, a backpropagation neural network model is the best model if it has the lowest MSE score compared to other models. Next, MAPE calculation conducted to find out the accuracy of the model. It is necessary to conduct testing by changing hidden layer variables and their parameters. Description of each parameters used in building models and designing Backpropagation Neural Network architecture are written below:

a. Input Layer
Training process research uses five input layer nodes consisting of the data temperature, humidity, water surface distance, water discharge and bulk rain.

b. Hidden Layers
This research uses a hidden layer called the single hidden layer model. The number of nodes started from the number of input nodes, i.e., 5. Thus, the experimental process will use a single hidden layer with 5 nodes, 6 nodes, 7 nodes, 8 nodes, 9 nodes, 10 nodes, 15 nodes, 20 nodes, 25 nodes and 30 nodes.

c. Output Layer
This training consists of 3 nodes showing significant output probability in determining proper classification.

d. Optimizer
The type of optimizer used in the research is Adam from the scikit-learns library in Python. Adam is algorithm optimization that can be used as a change from classical stochastic gradient descent procedure for renewing weight by iterative based on training data. Adam is a combination between RMSprop and Stochastic Gradient Descent with momentum.

e. Activation Function
This research uses 2 types of transfer functions types: Sigmoid and Softmax. A sigmoid activation function is placed on the hidden layer node, while the Softmax activation function is placed on the output layer nodes. Sigmoid activation function produces a score in (0, 1), which is helpful in computing and should be interpreted as a probability. At the same time, the Softmax activation function can change the real vector to be a probability vector categorical. Element vector output is in the range (0, 1) and a total 1. Softmax is often used as Activation for final layer from network classification because the result could be interpreted as distribution probability.

f. Epoch
Epoch is many iterations performed when a neural network model is trained. The epoch's amount will influence the results performance of the model because the epoch will determine the end or stop limit of the training process. In this research, the number of epochs used is 100, 500, 1000 and 5000.

g. Learning rate
Learning rate is one of the training parameters to count weight score correction during the training process. This learning rate value ranges from zero (0) to (1). The bigger learning rate value means the training process will walk faster. The values used are also the same, start from 0.1 to 0.9 and with 0.1 , 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9 intervals.

Table 9. Backpropagation Neural Network Model Design

Parameter	Score	Information
Input layers (n)	5 nodes	Temperature, humidity, Water surface distance, water discharge, bulk rain
Hidden layers	2 – 10 nodes	Trial and error
Output layers	3 nodes	Prediction Flood
Optimizer	1	Adam
Transfer Function	2	Sigmoid , Softmax
Epoch	100, 500, 1000, 5000	Trial and error
Learning rate	0.1 until 0.9	Trial and error

C. *Training Process*

The training process uses 80 % of the total data in the dataset. The data sharing process automatically uses the split feature of the scikit-learn library. However, before conducting sharing, the first data is normalized for input data and one hot encoding for output data because the data in the dataset is still divided in the form of classification. Normalization process using min max scaler from the scikit-learn library as explained in the previous section. The training process also performed initialization to save loss value and accuracy in each iteration, so authors know the model's amount of loss and accuracy.

After all, the configuration went well without an error, and next is a training simulation process. The training simulation time depends on the amount of data and iterations performed. After completing the training simulation, the output is in the form of three probability data, where the highest probability is the result of classification. And the use of the Argmax function of Numpy makes it easier to send the output result so that the result of the decision could more simple and easy to process to the next stage.

In this section, we will discuss the results of test modeling that has been carried out in the research. Modeling is performed by changing the number of neurons in the input layer, the number of neurons in the hidden layer and the parameters involved. Due to the dataset being categorical data, so categorical cross-entropy metric is used. Cross-entropy is the default loss function to use for trouble multi-class classification.

Cross-entropy will count, summarizing the average score difference between actual distribution probability and prediction for all classes in the practical problem. Minimized score and perfect cross-entropy score is 0. Cross-entropy can be specified as a loss function in Keras by specifying 'categorical_crossentropy' when composing models. This function requires that the output layer is configured with n nodes (one for each class) in three nodes and 'softmax' activation to predict the probability for each class.

Through that process, many Neural Network models will obtained, which a model that has an accuracy score approaching or equal to 1 and a loss value close to zero or equals zero is the best model. So the results of the model have a slight error rate and are under the significant or desired output value. The results of the training process shows in Table X.

Table 10. Results of The Training Process

Architecture	Lowest / Highest	Accuracy	Loss
5-5-3	Highest	1.0	4.96e-01
	Lowest	0.875	0
5-6-3	Highest	1.0	0.5432
	Lowest	0.5423	0
5-7-3	Highest	1.0	0.899
	Lowest	0.84375	0
5-8-3	Highest	1.0	2.12e-02
	Lowest	1.0	0
5-9-3	Highest	1.0	0.4740
	Lowest	0.875	0
5-10-3	Highest	1.0	0.298
	Lowest	0.90625	0
5-15-3	Highest	1.0	0.5154
	Lowest	0.875	0
5-20-3	Highest	1.0	0.5287
	Lowest	0.84375	0
5-25-3	Highest	1.0	0.1192
	Lowest	1.0	0
5-30-3	Highest	1.0	0.45983
	Lowest	0.90625	0

D. Testing Process

After determining the quantity, the amount of data used in the training during the testing process will be displayed. The Pandas library is used in the form of .xlsx or Microsoft Excel for reading and uploading data in Python. The data table used in the training process shows in Table X.

Authors explained that the search value that is lost must be close to or equal to 0 and near accuracy score or equal to 1. Next, compare prediction results conducted during the Neural Network process against the available data, the MSE value search of each model, and MAPE to determine the accuracy of prediction result (in percentage) to the actual data. The model used in this testing process has a good level of accuracy and loss, meaning it has 0.0 loss and 1.0 accuracy. Model selection is made to speed up computing and reduce errors so that the model can be well implemented in the system. Table XI shows MSE and MAPE data on testing data.

In forecasting the testing process results, the RMSE value of 4.6E-11 means the score is excellent. After the training and testing process, the best Neural Network model, called architecture 5 – 15 – 3 (5 input nodes, 15 hidden layer nodes, and 3 output layer nodes) with a learning rate of 0.1 at 5000 epochs, is obtained. The weights and biases identification in the model is conducted. Matrix W1 (weight from the input layer to hidden layer) with order 5x15, matrix B1 (bias from the input layer to hidden layer) with order 1x15, matrix W2 (weight from hidden layer to output layer) with order 15x3 and matrix B2 (bias from hidden layer to output layer) with order 1x3. Tables XII and XIII are matrices from weights and biases on the implemented model.

A confusion matrix is one of the predictive analytic tools that shows and compares actual scores with the results of score prediction model, which could produce metric evaluations like accuracy, precision, recall, and F1-Score or F-Measures. Accuracy describes how a model can classify correctly. Precision describes the level of accuracy between the requested data with prediction results given by the model. Recall describes the success of the model in retrieving information. While the F1 score is a comparison of average precision and recall weighed.

All four evaluation methods are very beneficial for measuring performance from the classifier or our machine learning algorithm to use a prediction. Table XIV shows the results of four methods with the most optimal model, 5 – 15 -3.

A confusion matrix is also frequently called an error matrix. The confusion matrix compares the classification results carried out by the system (model) with actual classification results. Confusion matrix in the form of matrix table which describes performance model classification on series data value test known. 4 terms represent the classification process in the confusion matrix. They are True Positive (TP), True negative (TN), False Positive (FP) and False Negative (FN).

There is a more straightforward method to read the confusion matrix to get a conclusion, if started with True, so prediction is correct, doesn't know something that predicted occur or not. If it starts with False, so prediction is wrong, as well positive and negative is the prediction result from the model.

Based on Figure 3 above, the authors could conclude that the prediction could be well done. It can be seen in the True Negative and True Positive columns, which show a more dominant value compared to the False Positive and False Negative columns.

Table 11. Smallest RMSE Value Each Architecture

Architecture	Learning rate	RMSE
5-5-3	0.2	0.1082
5-6-3	0.1	2E-08
5-7-3	0.1	2E-08
5-8-3	0.1	3E-08
5-9-3	0.2; 0.3	2E-08
5-10-3	0.1; 0.2; 0.3	2E-08
5-15-3	0.1	4.6E-11
5-20-3	0.1; 0.2	2E-08
5-25-3	0.1; 0.2	2E-08
5-30-3	0.1; 0.2;	2E-08

Table 12. Weights and Bias on The Hidden Layer

W1	1.0413197 -6.4241657 -1.5110763 -5.358439 4.753444 -1.7437297 5.2255955 -6.231856 1.813893 5.959697 -6.141656 -4.493788 4.5108137 0.9519901 1.0421944
	0.08702539 4.2014775 -4.548502 4.4708-3.6467478 8.48717 -3.5695238 4.7446127 - 7.2677927 -4.4900703 4.0773845 4.400976 -3.354172-3.749158-2.125056
	-1.4215258 6.8876786 -1.6814388 4.834753 -4.4587703 31.586603 -5.3602245.4498057 -35.740498 -5.1037765 6.665885 4.4888024 -5.1330676 -23.041878 -7.379085
	2.541483 -11.94959 5.483617-7.9123135 7.6205606 1.713161 8.71318-9.021179-1.0149755 8.537468 -10.647496 -7.6626925 8.125647 -0.6521959 1.8850287
	0.8985326 -4.5003242 -4.296767 -3.2051275 2.7444608 -0.06130471 3.7579496 -3.6500697 -0.049942 3.3869555 -4.1388164 -2.670514 3.4168258 -0.09478827 -1.3551795
B1	-0.47902653 1.8731744 -1.2408442 0.7299269 -0.80554426 -0.53199816 -1.330791 1.1247135 0.2726039 -1.0805436 1.3780107 0.27171108 -0.9318658 0.42377675 -0.25318936

Table 13. Weights and Bias on The Hidden Layer

W2	-1.6629304 0.640485 0.63996184
	8.071886 -7.292526-0.14644678
	-0.20647298 0.7504495 -2.6174343
	5.814152 -4.765791-2.2599676
	-6.9158955 2.7559915 1.6440003
	0.66440994 6.7416644 -13.689418
	-7.5483756 4.6113530.62145877
	8.1538515 -6.719881 -1.7984974
	-0.39757788 -8.951846 12.100247
	-9.192395 3.9971101 1.7706819
	8.325219 -8.256571-0.73897576
	4.442101 -3.843594-2.259305
	-7.769249 4.3207884 0.7575712
	-0.4200939 -6.8140407 9.134321
	-1.1890938 -1.4178838 2.852742
B2	0.00822386 0.02637376 -0.06117225

Table 14. Weights and Bias on The Hidden Layer

Method Evaluation	Score
Accuracy	0.9929328621908127
Precision	0.972972972972973
Recall	1.0
F1-Score	0.9863013698630138

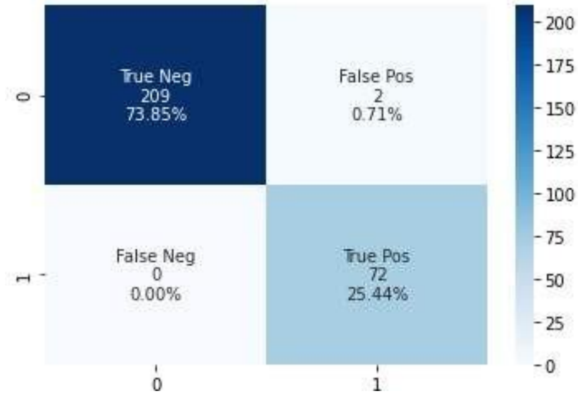


Fig. 3. Confusion Matrix

E. *Results of System Test*

In this section, the authors will discuss the test results that have been conducted previously, like testing each sensor and flood implementation system prediction by using the Backpropagation Neural Network method on the flood, will flood and not a flooded condition. Following is the summary of experiments that have been done previously. "T" in the True Negative and True Positive columns indicate a more dominant value compared to the False Positive and False Negative columns.

Based on the results in Table XV above, four types of sensors used have good accuracy levels for measurement. It can be seen from the average error percentage of the four sensors by 2.358%. The percentage is relatively small, so the testing sensor and overall system will go well.

Next is a whole test result system. In the testing system, there are 3 types of conditions to validate the performance system: not flood condition, will be flood, or flood condition. Three of these conditions will be validated using the Provision Classification that has been made. Table XVI shows test results from non-flood conditions.

Based on the test results in Table XVI above, a significant percentage success system in prediction by 92%. Of course, this figure still belongs well, so the model applied is good enough. Then switch to condition testing with conditions that will flood in Table XVI.

Based on the test results in Table XVI above, the extensive percentage success system in prediction by 88%. This figure still belongs well, so the model applied is good enough. Then switch to condition testing with conditions flood could be seen in Table XVI.

Based on the test results in Table XVI, the extensive percentage success system in prediction is 94%. This figure still belongs well, so the model applied is good enough.

Based on Table XVII, summarizing each condition with a different error range, it can be known that system could go well independently. The average error value is 91.33%, which states that experimenting and testing sensors can go as well as they should.

Table 15. Summary Sensor Test

No	Test	Parameter	Percentage Error

1	DHT11	Temperature	0.49%
2	DHT11	Humidity	1.55%
3	HC-SR04	Distance Water Face	2.66%
4	YF-S201	Water Discharge	3.97%
5	Rain Gauge	Bulk Rain	3.12%
Flat - Flat			2.358%

Table 16. Test System by Whole

Parameter	Range	Provision Classification	Presentation Average Success
Not flooded (0)			
Qualification 1			
Temperature	24 – 26	Low	80%
Humidity	65 – 75	Low	
Distance Water	500 – 400	Low	
Water Discharge	0 – 100	Low	
bulk Rain	0 - 1000	Low	
Qualification 2			
Temperature	>26 - 30	Currently	90%
Humidity	65 – 75	Low	
Distance Water	500 – 400	Low	
Water Discharge	0 – 100	Low	
bulk Rain	0 - 1000	Low	
Qualification 3			
Temperature	24 – 26	Low	90%
Humidity	>75 – 85	Currently	
Distance Water	500 - 400	Low	
Water Discharge	0 – 100	Low	
bulk Rain	>1000 - 1500	Currently	
Qualification 4			
Temperature	>26 – 30	Currently	90%
Humidity	>75 - 85	Currently	
Distance Water	500 – 400	Low	
Water Discharge	0 – 100	Low	
bulk Rain	0 - 1000	Low	
Qualification 5			
Temperature	>26 – 30	Currently	90%
Humidity	65 – 75	Low	
Distance Water	500 – 400	Low	
Water Discharge	>100 – 300	Currently	
bulk Rain	0 - 1000	Low	
Qualification 6			
Temperature	>26 – 30	Currently	90%
Humidity	>75 – 85	Currently	
Distance Water	500 – 400	Low	
Water Discharge	0 – 100	Low	
bulk Rain	>1000 – 1500	Currently	
Qualification 7			
Temperature	>26 – 30	Currently	90%
Humidity	>75 – 85	Currently	
Distance Water	500 – 400	Low	
Water Discharge	>100 - 300	Currently	
bulk Rain	0 - 1000	Low	
Qualification 8			
Temperature	24 – 26	Low	90%
Humidity	65 – 75	Low	
Distance Water	500 – 400	Low	
Water Discharge	>100 - 300	Currently	
bulk Rain	0 - 1000	Low	
Qualification 9			
Temperature	24 – 26	Low	100%
Humidity	65 – 75	Low	
Distance Water	500 – 400	Low	
Water Discharge	0 – 100	Low	
bulk Rain	>1000 - 1500	Currently	
Qualification 10			
Temperature	24 – 26	Low	100%
Humidity	65 – 75	Low	
Distance Water	<400 - 150	Currently	
Water Discharge	0 – 100	Low	
bulk Rain	0 - 1000	Low	
Will flood (1)			
Qualification 1			
Temperature	24 – 26	Low	90%
Humidity	65 – 75	Low	
Distance Water	500 – 400	Low	
Water Discharge	>300 – 500	Tall	
bulk Rain	0 - 1000	Low	
Qualification 2			
Temperature	24 – 26	Low	100%
Humidity	65 – 75	Low	
Distance Water	500 – 400	Low	
Water Discharge	0 – 100	Low	
bulk Rain	>1500	Tall	
Qualification 3			
Temperature	>26 – 30	Currently	100%
Humidity	>75 – 85	Currently	
Distance Water	<400 – 150	Currently	
Water Discharge	>100 – 300	Currently	
bulk Rain	>1000 – 1500	Currently	
Qualification 4			
Temperature	>26 – 30	Currently	80%
Humidity	>75 – 85	Currently	
Distance Water	<150	Tall	
Water Discharge	>100 – 300	Currently	
bulk Rain	>1000 – 1500	Currently	
Qualification 5			
Temperature	>26 – 30	Currently	90%
Humidity	>75 – 85	Currently	
Distance Water	<400 – 150	Currently	
Water Discharge	>300 – 500	Tall	
bulk Rain	>1000 – 1500	Currently	
Qualification 6			
Temperature	24 – 26	Low	80%
Humidity	65 – 75	Low	
Distance Water	<400 – 150	Currently	
Water Discharge	>100 - 300	Currently	
bulk Rain	0 - 1000	Low	
Qualification 7			
Temperature	24 – 26	Low	90%
Humidity	65 – 75	Low	
Distance Water	<150	Tall	
Water Discharge	>300 – 500	Tall	
bulk Rain	0 - 1000	Low	
Qualification 8			
Temperature	24 – 26	Low	90%
Humidity	65 – 75	Low	
Distance Water	<150	Tall	
Water Discharge	0 – 100	Low	
bulk Rain	>1500	Tall	
Qualification 9			
Temperature	>26 – 30	Currently	100%
Humidity	65 – 75	Low	
Distance Water	<400 – 150	Currently	
Water Discharge	>100 – 300	Currently	
bulk Rain	0 - 1000	Low	
Qualification 10			
Temperature	>26 – 30	Currently	100%
Humidity	>75 – 85	Currently	
Distance Water	<150	high	
Water Discharge	>100 – 300	Currently	
bulk Rain	>1500	Tall	

Table 17. Summary of Test Results System

No	Test	Amount	Right	Wrong	Percentage Success
1	Condition Not Flood	100	92	8	92%
2	Condition Will Flood	100	88	12	88%
3	Condition Flood	100	94	6	94%
Flat – Flat					91.33%

CONCLUSIONS AND RECOMMENDATIONS

A. *Conclusion*

Based on the results, in the analysis and testing system that has been created the conclusion is that the Backpropagation Neural Network method that has been through the previous training and testing process could be applied to predict incoming floods in river and watershed real-time. Based on the accuracy level generated by the Backpropagation Neural Network method, it can be seen to predict incoming floods by 94.3%. The most optimal accuracy for Backpropagation Neural Network model design to predict flood is on architectural models 5 – 15 – 3. The model consists of 5 neurons in the input layer, 15 neurons in the hidden layer, and 3 neurons in the output layer, using a learning rate of 0.1, so an RMSE value of 4.65E-11 can be obtained.

B. *Suggestion*

Based on the experiments, some suggestions exist to continue improving this final project. Including :

- Development to get more accurate results. Using data that has a more extended period is recommended, and adding prototypes to some point or relevant places for data records is recommended.

- In the data processing, author can use other compatible methods for flood prediction compared to a method used in this project.
- To increase the accuracy level of the prediction results generated by the model that has been made, author can add more parameters that support and influence incoming floods.

REFERENCES

1. Abas, MI, Syukur , A. and Soeleman , MA (2017) ' Prediction Time Sequence Amount Airport Passengers Using Neural Network Algorithm Based on Genetic Algorithm', Jurnal Technology Information , 13(January 2017), pp. 101– 114.

2. Ary , M. (2017) ' Application Prediction Flooding with Spade' Algorithm , Indonesian Journal on Computer and Information Technology, 2(1), pp. 11–16. Espressif (2019) 'ESP32 Series Datasheet',

3. Espressif Systems, pp. 1–61. Available at: www.espressif.com.

4. Fadhilah , RN and Ginardi , RVH (2017) ' Pricing Using the Back Propagation Method in the CariKos E-Commerce Application Web-Based ', ITS Engineering Journal , 6(2). doi : 10.12962/j23373539.v6i2.23978.

5. Al Fikri , I. (2016) ' Application Navigation Based on Device Moving Using the Wikitude Platform for Study Case ITS' Environment , ITS Engineering Journal , 5(1), pp. 48–51. doi : 10.12962/j23373539.v5i1.14511.

6. Fitriadini , A. et al. (2020) ' Application of Backpropagation Neural Network in Stock Price Prediction ', Student National Seminar Knowledge Computers and their Applications (SENAMIKA), p. 561.

7. Hokya , S. (2013) ' Python Programming Handbook ', Books , 84, pp. 487–492. Available at: http://ir.obihiro.ac.jp/dspace/handle/10322/3933 .

8. Ike, Y. and L. (2018) ' Application Prediction of Rainfall, Water Discharge, and Flood Web-based with Machine Learning in Deli Serdang', Journal of Communication Research and Public Opinion, 22(2), pp. 150–160.

9. Joni, W. (2019) ' Do'a and Iqro' E - Learning System in Enhancing the Learning Process in Kindergarten Amal Ikhlas', Journal Student Application Technology Computers and Information, 1(3), pp. 154–159. Available at: http://www.ejournal.pelitaindonesia.ac.id/JMApTeKsi/index.php/JOM/articl e/view/526.

10. Junaidi and Prabowo, YD (2018) Project System control Electronic Arduino based, CV Anugrah Utama Raharja. Available at: https://docplayer.info/109709787-Project-sistem-kendali-elektronik-based-arduino-dr-junaidi-s-si-m-sc-yuliyan-dwi-prabowo.html. Ministry of Health (2016) ' Already Are we ready to face Flood ?' Mauliana , P. (2016) ' Prediction Citarum River Flood With Fuzzy Logic Results of Particle Swarm Optimization Algorithm , Informatics, AMIK BSI Bandung, 3(9), pp. 269–276. Available at: file:///C:/Users/Septian/Downloads/citarum teteh /CITARUM RIVER FLOOD PREDICTION WITH FUZZY LOGIC RESULTS OF PARTICLE SWARM OPTIMIZATION ALGORITHM.pdf.

11. Rafiuddin Syam, ST, M.Eng , P. (2013) Basics of Sensor Engineering. Available at: http://anies.staff.gunadarma.ac.id/Downloads/files/73301/Ref_2_Rafiuddin +Syam_Dasar_teknik+sensor.pdf.

12. Republic of Indonesia (2004) Law of the Republic of Indonesia No. 7 of 2004 concerning Source Water Power. Jakarta: State Secretariat .

13. Republic of Indonesia (2012) Regulation Government Republic of Indonesia Number 37 of 2012 concerning Watershed Management , State Secretariat . Available at: https://dspace.ups.edu.ec/bitstream/123456789/5224/1/UPS-QT03885.pdf.

14. Sari, YN (2016) Network Nerves Backpropagation clone to predict attack area . Semarang: Semarang State University.

15. Shah, C. (2020) 'MySQL', A Hands-On Introduction to Data Science, pp. 187–206. doi : 10.1017/9781108560412.008.

16. Soebroto , AA et al. (2015) ' Prediction of Water Level (TMA) for Early Detection of Disasters Flood Using SVR-TVIWPSO', Journal Technology Information and Science Computer , 2(2), p. 79. doi : 10.25126/jtiik.201522126.

17. Subhan Panji Cipta (2016) ' Application Evolving Neural Network Algorithm for Rainfall Prediction ' , Jurnal Technology Lambung University Information Mangkurat (JTIULM), 1(1), pp. 1–8. doi : 10.20527/ jtiulm.v 1i1.2.

18. Sulistyo Ida; Adipraja , Philip Faster Eka, DAW (2020) ' Tsukamoto Fuzzy Inference System Modeling for Prediction Incident Floods in Malang City', Jurnal Technology Information and Science Computer , 7(Vol 7, No 1: February 2020), pp. 189–196. doi : 10.25126/jtiik.202071898.

19. Syahbeni , M. et al. (2018) ' Design Get up Rainfall Detector _ Using the Tipping Bucket Rain Sensor and Arduino Uno', Agrotechnics , 1(2), pp. 51–62. doi : 10.32530/ agtk.v 1i2.22.

20. Villela , lucia maria aversa (2013) ' Cause Floods ', Journal of Chemical Information and Modeling, 53(9), pp. 1689–1699. Water Flow Sensor: YF-S201 Datasheet (2015).

21. Widiawaty , MA and Dede, M. (2018) ' Modeling Spatial Dangers And Vulnerabilities Disaster Flood In the Eastern Region of the District Cirebon', Journal of Dialogue Countermeasures BNPB Disaster , 9(2), pp. 142–153