BAREKENG: Journal of Mathematics and Its Applications
December 2024 Volume 18 Issue 4 Page 2435–2448
P-ISSN: 1978-7227 E-ISSN: 2615-3017
https://doi.org/10.30598/barekengvol18iss4pp2435-2448

IMPLEMENTATION OF BACKPROPAGATION AND HYBRID ARIMA-NN METHODS IN PREDICTING ACCURACY LEVELS OF RAINFALL IN MAKASSAR CITY

Hisyam Ihsan[1*], Irwan[2], Andi Illa Erviani Nensi[3]

[1,2,3]*Department of Mathematics, Faculty of Mathematics and Natural Sciences, Universitas Negeri Makassar*
Jl. Dg. Tata Raya Kampus UNM Parangtambung, Makassar, 90224, Indonesia

*Corresponding author's e-mail: * **hisyamihsan@unm.ac.id***

Article History:

Received: 1st, April 2024
Revised: 1st, June 2024
Accepted: 28th, July 2024
Published:14th, October 2024

Keywords:

ARIMA; Backpropagation; Hybrid ARIMA-NN; Makassar; Rainfall

ABSTRACT

Hybrid ARIMA-NN is a combined approach of the ARIMA model used to capture linear patterns in time series data and Artificial Neural Networks (ANN) to handle non-linear and stochastic patterns. Using a gradient descent algorithm, backpropagation adjusts synaptic weights based on the error between the network's prediction and actual training data values. In this study, a comparison was made between the Backpropagation method and Hybrid ARIMA-NN in forecasting rainfall in Makassar City. Rainfall data in Makassar City uses data from the rainfall measuring station at the Paotere Maritime Meteorological Station in Makassar. The activation functions used are ReLU and Leaky ReLU with epoch parameters set at 350, and learning rates of 0.01, 0.001, 0.0001, and 0.00001. The two best methods selected for further evaluation are Backpropagation with architecture 12-32-16-8-1 and Hybrid ARIMA-NN (ARIMA [4,0,1]-NN 12-256-128-64-1). The ARIMA model (4,0,1) with AIC values of 1303.4 and RMSE 162,369 is the best compared to other models, which aligns with the advantages of backpropagation architecture. The results showed that the Backpropagation method excelled with an RMSE value of 137.320 or 0.1149, indicating high accuracy in forecasting changes in seasonal trends and patterns. Hybrid ARIMA-NN gives good results with RMSE 145.834, as residues contain better nonlinearity compared to ARIMA models (4,0,1), although it shows a slightly higher error rate compared to Backpropagation.

How to cite this article:

H. Ihsan, Irwan and A. I. E. Nensi "IMPLEMENTATION OF BACKPROPAGATION AND HYBRID ARIMA-NN METHODS IN PREDICTING ACCURACY LEVELS OF RAINFALL IN MAKASSAR CITY" *BAREKENG: J. Math. & App.*, vol. 18, iss. 4, pp. 2435-2448, December, 2024.

Journal homepage: *https://ojs3.unpatti.ac.id/index.php/barekeng/*
Journal e-mail: *barekeng.math@yahoo.com; barekeng.journal@mail.unpatti.ac.id*

Research Article · Open Access

1. INTRODUCTION

Artificial Neural Networks (ANNs) are a popular machine learning technique that simulates the learning mechanism in biological organisms [1]. Artificial neural networks are capable of learning from data and adapting to changing conditions, as well as capturing nonlinear relationships. Additionally, ANNs consist of several simple processing units called neurons, which are interconnected via synaptic weights and adjust their synaptic weights through the learning process [2] Synaptic weights are trained using the Backpropagation algorithm following the Gradient Descent Method. The common learning method used in ANNs is backpropagation, which utilizes a gradient descent algorithm to optimize synaptic weights to minimize the error between the expected output and the generated output [3].

Despite having advantages such as handling non-linear data, the backpropagation method also has limitations, such as being prone to overfitting and sensitivity to the initial values of synaptic weights and learning rate [4]. To overcome these limitations, one approach that can be taken is to combine the backpropagation method with other methods that have different characteristics or complement each other. One method that can be combined with backpropagation is the ARIMA-NN (Autoregressive Integrated Moving Average-Neural Network) method, which is a hybrid approach that combines the ARIMA model with ANN [5]. The ARIMA model captures linear patterns in time series data through its two main components: the autoregressive (AR) component, which models the linear relationship between the current value and past values, and the moving average (MA) component, which models the linear relationship between the current value and past prediction errors [6]. However, ARIMA only handles linear patterns and cannot capture nonlinear dynamics in the data. To address this limitation, hybrid models combining ARIMA with models like ANN are employed, as ANN can manage the nonlinear and stochastic patterns that ARIMA cannot capture. The ARIMA-NN hybrid approach can improve the prediction accuracy of rainfall intensity in Makassar City by leveraging the strengths of both methods in addressing the different characteristics of time series data.

Rainfall intensity is one of the important factors that affect human life and the environment, especially in Makassar City, which is vulnerable to climate change. High rainfall variability can disrupt various sectors, such as agriculture, transportation, and water availability [7]. Therefore, predicting rainfall intensity in Makassar City is an urgent need. Previous research has been conducted in the context of rainfall prediction using various methods, but this study aims to implement and compare the backpropagation model and the hybrid ARIMA-NN in predicting rainfall intensity in Makassar City.

Previous studies have been conducted [8], regarding rainfall modeling in Banyuwangi using linear method ARIMA and non-linear method Feed Forward Neural Network (FFNN), along with hybrid ARIMA-NN, aiming to compare the three methods in terms of accuracy, error, and computational time. Subsequently, there is research [9], utilizing the ARIMA method and JST method backpropagation in price forecasting, followed by research [10], focusing on modeling and prediction using the hybrid ARIMA-NN method, and a study [11], utilizing the JST backpropagation method in predicting monthly rainfall. However, this research is related to the implementation of JST backpropagation and hybrid ARIMA-NN in predicting rainfall intensity in Makassar City.

This research will explore the training procedures and performance accuracy of the backpropagation model and the hybrid ARIMA-NN approach in the context of rainfall prediction in Makassar City. The use of the ARIMA-ANN method is justified by the nature of rainfall patterns, which exhibit both linear and nonlinear characteristics. The ARIMA model captures the linear components, while the ANN addresses the nonlinear aspects, making the hybrid approach well-suited for accurately modeling and predicting complex rainfall patterns. Thus, this research is expected to make a significant contribution to the understanding and development of more accurate and reliable rainfall prediction methods for disaster mitigation and natural resources. By combining the strengths of both methods, it is hoped that this research can make a significant contribution to the development of more accurate and reliable methods for predicting rainfall in Makassar City, with the potential for real-world applications that require a deep understanding of rainfall behavior in the city.

2. RESEARCH METHODS

2.1 Data Source

The type of research carried out is quantitative applied research by applying Neural Networks, Backpropagation algorithms, and Time Series Analysis ARIMA methods which are then carried out hybrid in the Backpropagation method so that it becomes a hybrid ARIMA-NN method. The data used is secondary data obtained from online data from the Meteorology, Climatology and Geophysics Agency (BMKG), dataonline.bmkg.go.id especially at the Paotere Maritime Station, Makassar which is monthly rainfall data with a time interval of 9 years. The initial observation data from January 2014 to December 2022 consisted of 3,286 daily records, which were then aggregated into 108 monthly records. These monthly records represent the number of rainy days in each month. The training data comprises 96 monthly rainfall records from January 2014 to December 2021, while the testing data consists of 12 monthly rainfall records from January 2022 to December 2022.

2.2 Analysis Technique

The research procedures applied to achieve the objectives of this study are as follows:

1. Conducting a literature review related to the Backpropagation model and hybrid ARIMA-NN model, as well as the applications used in predicting rainfall intensity.
2. Collecting historical monthly rainfall data in Makassar City from 2014 to 2022 obtained from the BMKG online website.
3. Normalizing data to the [0,1] scale rescales values so that they fall between 0 and 1. This is achieved by subtracting the minimum value and dividing by the data range (maximum minus minimum), ensuring consistency and comparability across features.
4. Dividing the dataset into training and testing data is 96 data for training the model and 12 data for testing the model's performance.
5. Forming the model using the Backpropagation method.
    a. Determining the input and output or target, consisting of 12 inputs representing data from the past 12 months and 1 output representing data for the next month, thus predicting rainfall for 1 month using data from the previous 12 months.
    b. Determining the network architectures. This involves selecting the number of layers (including hidden layers), the number of neurons in each layer, and the activation functions used.
    c. Conducting the model training process, followed by model evaluation using testing data.
    d. Selecting the best model with the smallest error value.
6. Forming the model using the Hybrid ARIMA-NN method.
    a. Before using the hybrid model, create an ARIMA model by determining the input based on the ACF and PACF plots of the ARIMA model residuals.
    b. Training the ARIMA model with training data as the main component in the hybrid model.
    c. Designing the neural network model to be used and determining the number of neurons and hidden layers.
    d. Initiating the training process using the Backpropagation algorithm hybridized with the ARIMA model. Selecting the best model with the smallest error value.
7. Testing both models after learning by calculating their error values using Root Mean Squared Error (RMSE) and selecting the best model based on the two selected models, namely the Backpropagation model or the hybrid ARIMA-NN model. The use of RMSE is chosen because RMSE provides a greater penalty for larger errors, making it more sensitive to outliers and particularly useful in prediction cases such as rainfall where large errors need to be minimized. Although Mean Absolute Percentage Error (MAPE) can be used as an alternative, RMSE is more suitable for data with consistent and non-negative scales like rainfall, which is the focus of this study.
8. Denormalizing the data to its original format.
9. Predicting rainfall using the best model to obtain rainfall predictions for the coming years.

2.2.1 Backpropagation

The backpropagation method is a type of ANN that utilizes supervised learning algorithms, involving three layers: input, hidden, and output each layer has specific functions within the network. The input layer is responsible for inputting data into the network, the hidden layer serves as the location for processing data and determining how data is propagated through the network, and the output layer provides the final output based on the input [9]. The equations for the backpropagation algorithm in artificial neural networks are as follows:

1. Compute Output Error: Calculate the error for each output neuron j in the output layer using the derivative of the error function E concerning the output of the neuron a_j:

$$\delta_j = \frac{\partial E}{\partial a_j} \tag{1}$$

2. Propagate Error to Hidden Layers: Calculate the error for each neuron i in the hidden layers using the error signal from the next layer weighted by the synaptic weights connecting the current layer to the next:

$$\delta_i = \frac{\partial E}{\partial a_i} = \sum_k \delta_k \cdot \frac{\partial a_k}{\partial a_i} \tag{2}$$

3. Update Weights: Update the weights of the network using the calculated errors and the learning rate η:

$$\Delta w_{ji} = -\eta \cdot \delta_j \cdot \frac{\partial z_j}{\partial w_{ji}} \tag{3}$$

4. Update Biases: Modify the biases of the network using the computed errors and the learning rate η:

$$\Delta b_j = -\eta \cdot \delta_j \tag{4}$$

With:

δ_j, δ_i : The error signal for neuron j and neuron i
E : Error Function
a_j : Output of neuron j
w_{ji} : Weight of the connection from neuron i to neuron j
z_j : Weighted sum of inputs to the neuron j

2.2.2 Autoregressive Integrated Moving Average (ARIMA)

The most traditional method of non-stationary time series analysis is ARIMA, which allows the explanation of variables depending on previous values through an autoregressive component (AR), a differencing component (I), and a moving average component (MA). ARIMA requires data that has been stationary, where component I is used to make the time series stationary through differencing [12]. The ARIMA equation (p, d, q) is as follows:

$$\phi_p(B)(1-B)^d Z_t = \theta_0 + \theta_q(B)a_t \tag{5}$$

If an ARIMA model has a seasonal pattern, the general form of the model is as follows:

$$\Phi_p(B^s)(1-B^s)^D Z_t = \Theta_Q(B^s)a_t \tag{6}$$

This form of the seasonal ARIMA model is suitable for time series data that exhibit seasonal patterns or periodic fluctuations. In the model equation (5) represents the non-seasonal ARIMA model, where the time series Z_t is modeled as a function of its past values and the white noise error term a_t. This equation includes the autoregressive (AR) component $\phi_p(B)$, the differencing operator $(1-B)^d$, and the moving average (MA) component $\theta_q(B)$. On the other hand, model equation (6) depicts the seasonal ARIMA model, extending the non-seasonal model to account for seasonal patterns. It introduces seasonal counterparts to the AR, differencing, and MA components, represented by $\Phi_p(B^s)$, $(1-B^s)^D$, and $\Theta_Q(B^s)$ respectively, where s denotes the seasonal period. This allows the model to capture both non-seasonal and seasonal dynamics present in the time series data.

2.2.3 Hybrid ARIMA-NN

Hybrid ARIMA-NN modeling is conducted because rainfall data may exhibit both linear and non-linear patterns. Therefore, the combination of both models is expected to better capture the characteristics of rainfall data. Generally, the hybrid ARIMA-NN model can be formulated as follows:

$$Z_t = \hat{L}_t + \hat{Y}_t \tag{7}$$

Where $\hat{Y}_t$ represents the linear component modeled using ARIMA, and $\hat{L}_t$ represents the non-linear component modeled using Neural Network [10].

2.2.4 Error Calculation

1. Root Mean Squared Error (RMSE), is the square root of Mean Square Error (MSE) and is often used because it has the same units as the variable being measured [13]. MSE, calculated as the average of the squared differences between the predicted (y_i) and observed (x_i) values.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}|x_i - y_i|^2} \tag{8}$$

With:

n : The number of observations or data points
x_i : The observed value for the i^{th} data point
y_i : The predicted value for the i^{th} data point

2. Akaike's Information Criterion (AIC) is a statistical measure used for model selection, particularly in the context of regression analysis.

$$AIC = n\ln(\hat{\sigma}_\varepsilon^2) + 2(p+q+1), \hat{\sigma}_\varepsilon^2 = \frac{SEE}{n} \tag{9}$$

$$SEE = \frac{\sum_{i=1}^{m}(x_i - \hat{x}_1)^2}{n}, m < n \tag{10}$$

With:

n : The number of observations or data points
$\hat{\sigma}_\varepsilon^2$: An estimate of the variance of the error terms (ε), calculated as the sum of squared errors (SEE) divided by the number of observations
SEE : Calculated as the sum of squared differences between observed (x_i) and predicted ($\hat{x}_i$) values
p : The number of predictors or independent variables in the model
q : The number of parameters in the model

3. Bayesian Information Criterion (BIC), is a metric used in statistical analysis to compare different statistical models based on the number of observations and the number of parameters in each model. BIC aids in selecting the most appropriate model for the data by considering the trade-off between model accuracy and complexity [14].

$$BIC = nLn(\hat{\sigma}^2) + kLn(n) \tag{11}$$

With:

n : The number of observations or data points
$\hat{\sigma}^2$: An estimate of the variance of the error terms
k : The number of parameters in the model

3. RESULTS AND DISCUSSION

3.1 Backpropagation Method

3.1.1 Data Normalization

In this study, the Backpropagation Artificial Neural Network algorithm employs a binary sigmoid activation function with values ranging from 0 to 1. The choice of the binary sigmoid activation function is due to its non-linear nature, which enables the model to capture complex relationships between input and output, such as those present in rainfall data. Additionally, the function maps input values to the [0, 1] range,

which aligns well with the need to normalize monthly rainfall data. This normalization is crucial to prevent neuron saturation and ensure that the output remains within a realistic and consistent range. Moreover, the binary sigmoid function's easy differentiability is vital for the backpropagation process, allowing efficient gradient calculations necessary for updating the network's weights. Therefore, monthly rainfall data needs to be normalized or transformed to [0, 1] to reduce redundancy and ensure data consistency [15]. The normalization results can be observed in Table 1.

Table 1. Rainfall Data After Normalization

Month	Year								
	2014	2015	2016	2017	2018	2019	2020	2021	2022
January	0.700	0.805	0.322	0.615	0.659	0.533	0.479	1.000	0.640
February	0.262	0.297	0.608	0.339	0.598	0.198	0.451	0.363	0.553
March	0.260	0.256	0.187	0.375	0.480	0.372	0.221	0.569	0.198
April	0.236	0.171	0.101	0.188	0.140	0.290	0.084	0.360	0.053
May	0.088	0.088	0.037	0.063	0.027	0.050	0.156	0.054	0.258
June	0.112	0.046	0.039	0.164	0.101	0.051	0.061	0.063	0.088
July	0.025	0.000	0.012	0.019	0.041	0.002	0.008	0.036	0.007
August	0.005	0.000	0.000	0.044	0.001	0.000	0.013	0.054	0.035
September	0.000	0.000	0.066	0.057	0.001	0.000	0.019	0.096	0.040
October	0.000	0.000	0.356	0.076	0.010	0.000	0.045	0.090	0.326
November	0.098	0.124	0.126	0.384	0.131	0.069	0.208	0.275	0.282
December	0.563	0.518	0.458	0.799	0.718	0.235	0.773	0.807	0.629

Data Source: Data Online BMKG (After Normalization)

3.1.2 Training and Testing Process

After normalization, monthly rainfall data is divided into two parts: training data used to train the model and testing data to evaluate the performance of the model after the training process. In this scenario, the data is divided into two time periods: 2014-2022 with a total of 96 data points for training, and testing data for the year 2022 consisting of 12 data points. The network used is a Multilayer Network for predicting rainfall using feedforward learning methods, comprising three types of layers: input, hidden, and output layers [16]. For the input, 12 neurons representing the 12 months of data are selected, while for the output, 1 neuron representing the subsequent data point is chosen, as shown in Figure 1. There are three types of architectures used: Multilayer Network with one hidden layer, two hidden layers, and three hidden layers.

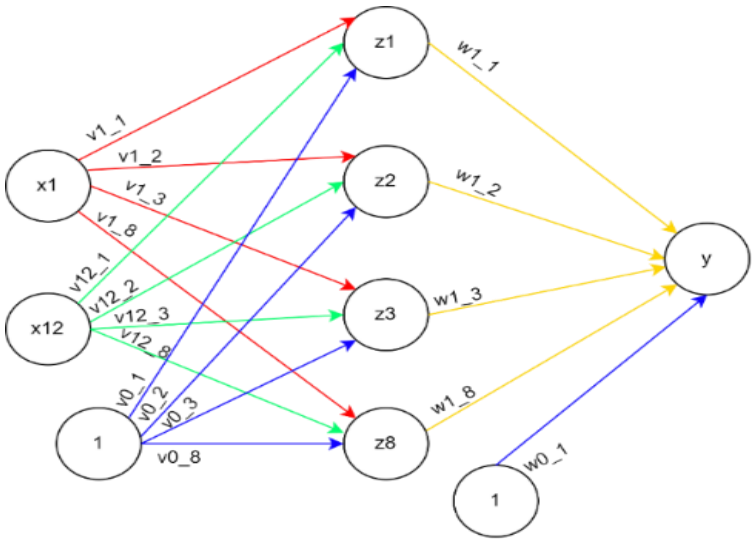


Figure 1. The Architecture of The Network is 12-8-1

In the study, neural network models are trained with uniform parameters: each model undergoes 350 epochs of training, utilizing different learning rates (0.01, 0.001, 0.0001, and 0.00001). The models incorporate Rectified Linear Unit (ReLU) and Leaky ReLU activation functions to introduce non-linearity, following reference [17]. ReLU is chosen for its computational efficiency and ability to mitigate the vanishing gradient problem, which helps accelerate the training of deep networks. However, to address the potential issue of "dying ReLUs," where neurons can become inactive, Leaky ReLU is also employed, allowing a

small, non-zero gradient when the input is negative. To control the model complexity, L2 regularization is applied to all layers except the output layer, with a regularization parameter of 0.01.

The process of determining the initial weights is crucial for the training of neural networks. Specifically, the initial weights from the input layer to the hidden layer (v_{ij}) and from the hidden layer to the output layer (w_{jk}) are specified. This setup ensures that each model starts training with a known set of weights, which helps in comparing the impact of different learning rates and activation functions on the training performance.

Table 2. Rainfall Data After Normalization

	z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8
x_1	0.188	0.179	-0.061	0.142	-0.375	-0.203	-0.005	-0.296
x_2	-0.086	-0.136	0.403	0.144	0.326	0.128	-0.126	-0.004
x_3	-0.542	0.078	-0.246	0.439	-0.481	0.448	0.445	0.231
x_4	-0.248	-0.320	0.507	-0.397	-0.163	0.389	0.137	-0.440
x_5	0.370	-0.360	-0.103	0.446	-0.397	0.182	0.152	-0.192
x_6	-0.495	0.053	0.128	-0.508	-0.362	-0.445	-0.186	-0.017
x_7	0.417	0.213	0.045	-0.165	-0.154	-0.404	0.254	-0.270
x_8	-0.501	0.130	0.376	0.221	0.210	0.200	0.112	-0.009
x_9	-0.240	-0.342	0.482	0.433	0.007	-0.121	-0.249	0.464
x_{10}	-0.284	0.262	-0.486	-0.144	0.263	0.416	0.226	0.409
x_{11}	-0.174	0.287	-0.011	-0.438	0.291	-0.113	0.438	0.155
x_{12}	-0.359	-0.028	-0.446	-0.325	0.245	0.035	-0.277	0.072
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Table 2 presents the normalized rainfall data, which is essential for training neural networks, as normalization ensures that each feature contributes equally to the learning process. Normalization transforms the data into a consistent scale, which helps in speeding up the training process and improving the model's performance. Each column in **Table 2** corresponds to a different feature (z_1 to z_8), representing various attributes of the rainfall data, such as temperature, humidity, wind speed, etc. Each row corresponds to a different data point (x_1 to x_{12}), representing observations collected over different periods or locations. This structured representation of data facilitates the neural network in learning complex patterns and relationships between the input features and the target variable, ultimately enhancing its predictive accuracy. By ensuring that no single feature disproportionately influences the training process, normalization contributes to the stability and convergence of the neural network model.

Table 3. Initial Weights of Hidden-Output Layer

	Y
z_1	0.593
z_2	-0.391
z_3	0.585
z_4	0.325
z_5	0.607
z_6	-0.019
z_7	0.769
z_8	-0.304
1	0

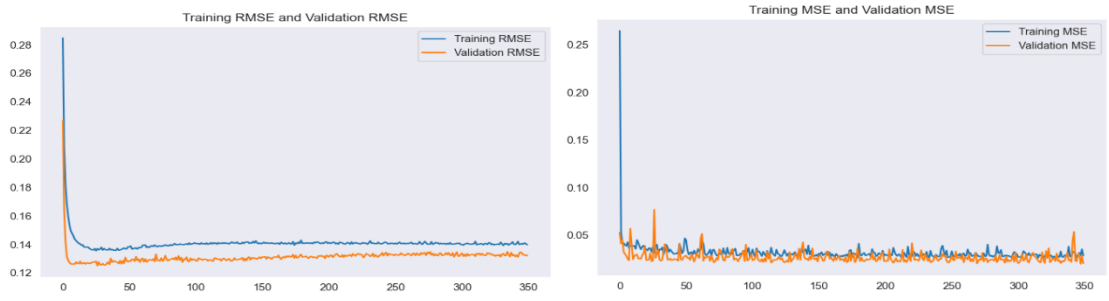
Table 3 provides the initial weights from the hidden layer to the output layer before the training begins. These weights (from z_1 to z_8) are crucial as they influence the initial predictions made by the neural network. Proper initialization of these weights is essential as it sets the starting point for the training process, impacting how quickly and effectively the model converges to an optimal solution. Understanding these initial weights helps in analyzing how the network's performance evolves over the training epochs and provides insight into the model's learning dynamics. The training process of the neural network is conducted for each pair of training data using the updated weights from the previous iteration as the initial weights. This means that after processing one pair of training data, the weights are adjusted based on the error between the predicted and actual outputs, and these updated weights are then used as the starting point for the next iteration. It is an iterative process where after training on one pair of training data, one epoch is considered complete **[18]**. This

approach ensures that the model incrementally improves its performance with each epoch by progressively minimizing the error. The results of this extensive training process, presented in the subsequent table, evaluate the model's performance based on different activation functions and learning rates, providing valuable insights into the effectiveness of various training configurations.

Table 4. Comparison of RMSE Value with Backpropagation Model Activation

Architecture	RMSE of Activation Function							
	ReLu				Leaky ReLu			
	0.01	0.001	0.0001	0.00001	0.01	0.001	0.0001	0.00001
12-8-1	165.2	170.2	151.8	350.3	165.1	161.5	156.5	298.2
12-16-1	182.6	172.0	156.2	214.5	182.5	168.7	153.3	242.1
12-32-1	172.7	174.2	156.5	169.1	173.1	180.7	161.2	177.5
12-64-1	182.9	160.1	159.9	172.5	182.2	166.8	160.4	165.6
12-128-1	162.4	181.4	162.9	170.8	162.3	177.1	160.3	160.0
12-256-1	168.5	**153.9**	162.5	167.0	172.0	161.7	166.2	164.4
12-8-8-1	177.9	166.4	157.3	168.1	162.6	165.7	153.3	**140.4**
12-16-16-1	153.8	164.8	155.4	162.2	153.7	161.6	159.4	149.2
12-32-32-1	166.1	166.7	159.7	157.3	163.7	166.3	157.6	146.7
12-64-32-1	162.7	173.8	162.4	153.2	174.8	172.4	164.4	144.8
12-128-64-1	**146.7**	163.2	158.1	**152.2**	**147.3**	159.9	158.2	151.0
12-256-128-1	170.5	171.6	158.8	157.1	160.1	169.6	159.0	157.6
12-8-8-8-1	181.3	157.4	164.0	160.8	178.5	158.5	**151.8**	165.0
12-16-16-16-1	198.5	175.5	154.3	163.9	194.4	170.5	155.5	195.1
12-32-16-8-1	238.3	164.4	**137.3**	373.7	231.8	158.4	156.9	165.2
12-64-32-16-1	191.3	159.2	154.2	153.3	272.1	**153.7**	152.5	145.4
12-128-64-32-1	187.8	166.4	156.4	156.4	179.9	166.2	154.0	151.2

The results of RMSE measurements across various neural network configurations demonstrate performance variations. Generally, lower RMSE values are observed with larger numbers of neurons and layers, along with the ReLU activation function. However, excessive complexity can lead to overfitting [19]. Variations in the learning rate also affect model performance, with smaller values tending to yield better results. For example, in the architecture (12-32-16-8-1) with ReLU and a learning rate of 0.01, a high RMSE of 238.336 is observed, indicating excessive complexity or a learning rate that is too large. The choice of activation function is also crucial. Leaky ReLU can address the issue of dying neurons, but in some cases, ReLU performs better. For instance, in the architecture (12-32-16-8-1) with ReLU and a learning rate of 0.0001, the lowest RMSE of 137.320 is achieved, indicating optimal performance. The importance of selecting the activation function and architecture configuration in neural network development underscores the need for a balance between model complexity and performance. Even models with simple architectures can compete with complex ones if chosen carefully.



(a) **(b)**

Figure 2. Training and Validation MSE, RMSE of Model (12-32-16-8-1)
(a) Train and Val RMSE, (b) Train and Val MSE

The evaluation results in Figure 2 indicate that the model with the architecture (12-32-16-8-1) and 350 epochs does not suffer from overfitting or underfitting. The small difference between the training MSE (0.025) and validation MSE (0.023) signifies the model's ability to generalize patterns effectively from the

training data to the validation data. Similarly, the small difference between the training RMSE (0.139) and validation RMSE (0.132) demonstrates the model's capability to make accurate predictions for both datasets. Therefore, the model can be relied upon to comprehend complex patterns within the dataset and provide accurate predictions.

3.2 Hybrid ARIMA-NN Method

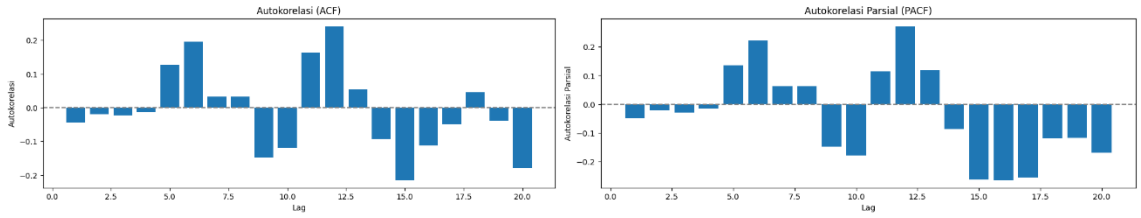
3.2.1 Identification of ARIMA Model

The initial step performed before obtaining an ARIMA model is to test the stationarity of the data to observe the presence of any unit roots in the variables. Stationarity testing is conducted using the unit root test, often referred to as the Augmented Dickey-Fuller Test (ADF Test) [20].

Table 5. Model Parameter Estimation

		t-statistic	Prob
Augmented Dickey-Fuller test statistics		-3.490	0.048
Test critical values:	1%	-3.503	
	5%	-2.89	
	10%	-2.584	

From the stationarity test results in Table 5, it is shown that the rainfall data produces an ADF statistic value of -3.49 with a probability value of 0.048 or < 0.05 and |t-statistic| < |test critical values| [8]. The negative ADF value indicates that the data exhibits stationary tendencies, while the p-value lower than 0.05 suggests statistical significance, leading to the rejection of the null hypothesis: $\gamma = 0$. Therefore, it can be concluded that the rainfall data is stationary within the tested time range, providing a solid basis for modeling using methods such as ARIMA, which requires stationarity assumptions in the modeling process. Further model identification is conducted by examining the lags that experience cutoffs in the ACF plot and PAC plot.



(a) **(b)**

Figure 3. Plot of ACF and PACF on Monthly Rainfall Data (a) ACF, (b) PACF

Figure 3 shows the presence of seasonal spikes at multiples of lag 12, which are more prominent compared to others in the ACF plot. Estimation of the parameters $\emptyset_1$ and θ_1 for each ARIMA model is based on the ACF and PACF functions. A model is considered adequate if the probability values for all variables are less than or equal to 0.05, and the absolute values of the t-statistics for all variables exceed the critical value from the t-table [21]. Here are some possible ARIMA models that are formed.

Table 6. Model Parameter Estimation

		AR 1	AR 2	AR 3	AR 4	AR 5	MA 1	MA 2	Jarque-Bera (JB)	Ljung-Box (L1)	Result
ARIMA	(1,0,0)	0.000							0.05	0.31	Significant
	(0,0,1)						0.000		0.05	0.26	Significant
	(2,0,0)	0.000	0.006						0.05	0.59	Significant
	(3,0,0)	0.000	0.289	0.409					0.00	0.48	Insignificant
	(2,0,1)	0.000	0.000				0.000		0.02	0.38	Insignificant
	(1,0,1)	0.001					0.006		0.05	0.72	Significant

	AR 1	AR 2	AR 3	AR 4	AR 5	MA 1	MA 2	Jarque-Bera (JB)	Ljung-Box (L1)	Result
(3,0,1)	0.920	0.745	0.746			0.474		0.00	0.60	Insignificant
(4,0,1)	0.000	0.006	0.001	0.044		0.013		0.06	0.81	Significant
(4,0,0)	0.000	0.108	0.241	0.031				0.04	0.42	Insignificant
(5,0,1)	0.000	0.026	0.240	0.301	0.989	0.065		0.01	0.80	Insignificant
(4,0,2)	0.118	0.669	0.660	0.045		0.48	0.858	0.04	0.76	Insignificant
(5,0,0)	0.000	0.169	0.334	0.255	0.489			0.03	0.60	Insignificant
(5,0,2)	0.847	0.989	0.986	0.802	0.948	0.959	0.91	0.04	0.75	Insignificant

The significance of the parameter is seen from the absolute value of the t_{value} > of the t_{table} [22]. Based on **Table 6**, it is obtained that 5 ARIMA models have significant parameters met and 8 insignificant models. After obtaining a significant ARIMA model, the next step is to select the ARIMA model based on the lowest Akaike Information Criterion (AIC) value. The selected ARIMA model will be used to perform ARIMA-NN hybrid modeling of the best ARIMA residuals in the form of ARIMA which obtained the smallest AIC and BIC values.

Table 7. ARIMA Model AIC, BIC, and RMSE Value

No	ARIMA (p, d, q)	AIC	BIC	RMSE
1	(1,0,0)	1323.706	1331.399	235.580
2	(0,0,1)	1325.29	1332.889	233.752
3	(2,0,0)	1321.226	1331.483	208.583
4	(1,0,1)	1321.582	1331.937	214.408
5	**(4,0,1)**	**1303.491**	**1321.442**	**162.369**

Based on **Table 7** it can be seen that the ARIMA model (4,0,1) is the best because it has the lowest AIC, BIC, and RMSE values. The ARIMA model equation (4,0,1) is as follows.

$$Yt = 261.99 + 1.17\, Y_{t-1} - 0.539\, Y_{t-2} + 0.249\, Y_{t-3} - 0.351\, Y_{t-4} - 0.6110\, \in_{t-1} + \in_t \qquad (11)$$

3.2.2 Hybrid ARIMA-NN

Hybrid modeling using the ARIMA model (4,0,1) which is used as ANN input using residual results in the model. The residuals used then determine the input, hidden layer, and output on the neural network. The architecture used is similar to the architecture in the backpropagation method and the residuals used as inputs are normalized at scale [0,1]. The results of the evaluation using RMSE as an indicator of model accuracy.

Table 8. Comparison of RMSE Value with Hybrid Model Activation

RMSE Activation Function								
ARMA Architecture Hybrid (4,0,1)-NN	**ReLU**				**Leaky ReLU**			
NN	**0.01**	**0.001**	**0.0001**	**0.00001**	**0.01**	**0.001**	**0.0001**	**0.00001**
12-8-1	**145.9**	159	147.6	1051.7	147.8	158.4	185.2	368.4
12-16-1	148.8	161	168.4	202.9	147.2	149.5	153.8	569.7
12-32-1	182.1	159	153	183.1	153.1	148.7	232	194.8
12-64-1	163	159	148.9	363.1	160.5	150.9	146.3	169.9
12-128-1	175.9	169	146.5	177.4	159.6	149.2	145.9	151.6
12-256-1	154.7	158	**145.8**	152.7	161.2	147.7	146.5	148.5
12-8-8-1	156	162	147.6	961.9	146	146.3	180.5	314.6
12-16-16-1	152.8	158	145.8	486.1	160.4	147.6	145.8	298.3
12-32-32-1	155.5	159	146.1	328.5	153.7	148.4	147.4	227.2
12-64-32-1	149.5	161	147	236.4	**145.9**	**146**	146.5	206
12-128-64-1	153.7	164	145.8	148.3	155.5	147.4	147.4	149.6

RMSE Activation Function								
ARMA Architecture Hybrid (4,0,1)-NN	**ReLU**				**Leaky ReLU**			
NN	**0.01**	**0.001**	**0.0001**	**0.00001**	**0.01**	**0.001**	**0.0001**	**0.00001**
12-256-128-1	150.1	162	146.4	**146**	151.3	147.2	147.1	149
12-8-8-8-1	148.2	**157**	148.8	445.4	148.1	146.7	161.7	150.6
12-16-16-16-1	148.2	162	175.4	421.8	148.2	147.5	146.8	174.6
12-36-16-8-1	148.3	162	154.7	321.7	148.2	147.2	146.4	149.2
12-64-32-16-1	148.2	163	145.9	159.2	148.3	147.5	146	**147.8**
12-256-128-64-1	148.4	163	146.2	146.2	148.9	148.3	**145.8**	148

From the RMSE measurement results on the Hybrid ARIMA-NN architecture with ReLU activation function, it is evident that there is significant variation in model performance depending on the architecture configuration and learning rate used. The utilization of the ReLU activation function in **Table 8** indicates that some configurations outperform others. For instance, in the architecture (12-256-1) with a learning rate of 0.00001, an RMSE of 145.837 is obtained, indicating the model's ability to provide more accurate predictions. However, there are also cases where increasing model complexity, such as increasing the number of neurons or layers, does not always result in improved performance and may even lead to a decrease in RMSE. Meanwhile, when using the Leaky ReLU activation function, the results also exhibit significant performance variation depending on the architecture configuration and learning rate. In some cases, Leaky ReLU outperforms ReLU, especially in configurations with smaller learning rates. For example, in the architecture (12-256-128-64-1) with a learning rate of 0.0001, an RMSE of 145.834 is obtained, demonstrating the potential of Leaky ReLU to provide more accurate predictions. Based on **Table 8** the L_t model can be formulated as follows:

$$L_t = W_3 \cdot \sigma_2(W_2 \cdot \sigma_1(W_1 \cdot Y + b_1) + b_2) + b_3 \quad (12)$$

With:
W_1, W_2, W_3 : Weights for each layer
b_1, b_2, b_3 : Biases for each layer
σ_1, σ_2 : Activation function for each layer
Y : Input (lag of Y_t or other variable)

The model L_t is the output of a neural network that uses specific inputs, such as lags of Y_t, to predict residuals. Unlike the ARIMA model, which has a clear linear equation, L_t is generated through a complex neural network process. The network used has an architecture of 12-256-128-64-1, consisting of 12 input neurons, three hidden layers with 256, 128, and 64 neurons respectively, and one output neuron. The selected activation function is Leaky ReLU.

From the testing results, it can be observed that there is performance variation among different Hybrid ARIMA-NN architectures. Increasing the number of neurons in the hidden layers has a significant impact in some cases. There is a tendency that increasing the number of neurons in the hidden layers may help improve model performance. However, it should be noted that this increase does not always result in a significant decrease in RMSE. Some complex architectures may lead to overfitting, indicating that too many parameters can adversely affect model performance on test data **[23]**. It is important to consider the trade-off between model complexity and performance.

Table 9. Method Comparison Based on the Lowest RMSE

Method	Best Model	RMSE
Backpropagation	12-32-16-8-1	137.320
Hybrid ARIMA-NN	ARIMA (4, 0, 1) - NN 12-256-128-64-1	145.834

In this research, the Backpropagation method with the 12-32-16-8-1 model yielded a Root Mean Square Error (RMSE) value of 137.320. This result indicates that the model is capable of providing predictions that closely approximate the true values with a relatively low level of error. The utilization of neural network architectures like 12-32-16-8-1 demonstrates the model's ability to capture complex patterns in the data, and the deep layers can assist in extracting relevant features. The forecasting results from the Hybrid ARIMA-NN model can offer more accurate estimates for cases where the data patterns are difficult to explain by a single linear statistical model **[23]**. Meanwhile, the Hybrid ARIMA-NN method with the

combination of ARIMA (4, 0, 1) and the NN model 12-256-128-64-1 resulted in an RMSE of 145.834. The slightly higher RMSE value compared to Backpropagation indicates that although this approach combines the advantages of ARIMA models and neural networks, there may be some trade-offs between the model's ability to capture trends and seasonality and the complexity introduced by the neural network. From these results, it can be concluded that the Backpropagation method with the 12-32-16-8-1 model has slightly better accuracy compared to the hybrid ARIMA-NN method on the dataset used for evaluation. Here are the rainfall forecasts for the years 2023-2025.

Table 10. Rainfall Forecasting Results for 2023-2025

Month	Year		
	2023	2024	2025
January	598.084	472.490	410.273
February	424.047	366.405	330.517
March	282.017	269.728	254.743
April	198.416	198.097	195.939
May	163.554	143.966	149.323
June	92.237	107.420	116.554
July	70.451	90.547	102.152
August	91.677	105.152	119.494
September	126.041	160.063	186.531
October	296.453	292.228	294.373
November	416.387	398.312	374.710
December	534.214	445.064	401.838

In January 2023, predictions show a value of 598.084 mm. The value then decreased steadily until July 2023 whereas after that it steadily increased until December, and the pattern of decline in value remains stable in general. Backpropagation models can adjust to the general trend of data, despite some noise. Predictions for the following years (2024-2025) also show a consistent downward trend. The data also fluctuates at the end of 2023 which has increased again. This is following frequent rain patterns.

4. CONCLUSIONS

The ARIMA-NN Hybrid training process and the Backpropagation method on Neural Networks have similarities in three main stages, namely feedforward, error calculation, and parameter modification. The forecasting results show that the model of the best Backpropagation method is a model with an architecture of 12-32-16-8-1 with an RMSE value of 137.320 or 0.1149 while the model of the best ARIMA-NN hybrid method is the ARIMA model (4, 0, 1) with an NN architecture (12-256-128-64-1) and an RMSE of 145.834 or 0.195. Through both models of the method, the RMSE value is calculated which results in the Backpropagation method with the architecture 12-128-64-1 having the smallest RMSE value. Thus, the Backpropagation method with the 12-128-64-1 architecture gives better results compared to the ARIMA-NN hybrid method with the ARIMA (4, 0, 1) – NN (12-256-128-64-1) model.

REFERENCES

[1] E. Alpaydin, *Neural Networks and Deep Learning*. 2021. doi: 10.7551/mitpress/13811.003.0007.

[2] H. R. Vega-Carrillo, V. M. Hernández-Dávila, E. Manzanares-Acuña, E. Gallego, A. Lorente, and M. P. Iñiguez, "Artificial neural networks technology for neutron spectrometry and dosimetry," *Radiat. Prot. Dosimetry*, vol. 126, no. 1–4, pp. 408–412, 2007, doi: 10.1093/rpd/ncm084.

[3] Y. Chauvin, D. E. Rumelhart, R. Durbin, and R. Golden, "Backpropagation: The Basic Theory," *Backpropagation Theory, Archit. Appl.*, no. 1995, pp. 0–34, 1995.

[4] B. J. Wythoff, "Backpropagation neural networks. A tutorial," *Chemom. Intell. Lab. Syst.*, vol. 18, no. 2, pp. 115–155, 1993,

doi: 10.1016/0169-7439(93)80052-J.

[5] N. Sadeghi Gargari, H. Akbari, and R. Panahi, "Forecasting Short-term Container Vessel Traffic Volume Using Hybrid ARIMA-NN Model," *Int. J. Coast. offshore Eng.*, vol. 3, no. 3, pp. 47–52, 2019, doi: 10.29252/ijcoe.3.3.47.

[6] S. Ayub and Y. Z. Jafri, "Comparative Study of an ANN-ARIMA Hybrid Model for Predicting Karachi Stock Price," vol. 10, no. 1, pp. 1–9, 2020, doi: 10.5923/j.ajms.20201001.01.

[7] D. Susilokarti, S. S. Arif, S. Susanto, and L. Sutiarso, "Studi Komparasi Prediksi Curah Hujan Metode Fast Fourier Transformation (Fft), Autoregressive Integrated Moving Average (Arima) Dan Artificial Neural Network (Ann)," *J. Agritech*, vol. 35, no. 02, p. 241, 2015, doi: 10.22146/agritech.9412.

[8] Y. Susanto and B. S. S. Ulama, "Pemodelan Curah Hujan dengan Pendekatan Model ARIMA , Feed Forward Neural Network dan Hybrid (ARIMA-NN) di Banyuwangi," *J. Sains dan Seni ITS*, vol. 5, no. 2, p. D-145-D-150, 2016, [Online]. Available: http://ejurnal.its.ac.id/index.php/sains_seni/article/viewFile/16409/3032

[9] Wahyu ngestisari, "The Perbandingan Metode ARIMA dan Jaringan Syaraf Tiruan untuk Peramalan Harga Beras," *Indones. J. Data Sci.*, vol. 1, no. 3, pp. 96–107, 2020, doi: 10.33096/ijodas.v1i3.18.

[10] D. Pratiwi and M. Hadijati, "Inflation modeling in Indonesia using hybrid autoregressive integrated moving average (ARIMA)-neural network (NN)," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1115, no. 1, p. 012058, 2021, doi: 10.1088/1757-899x/1115/1/012058.

[11] A. A. Gifari, Misbahuddin, and Made Sutha Yadnya, "Penggunaan Jaringan Syaraf Tiruan Metode Backpropagation Untuk Prediksi Curah Hujan Berbasis Website," *Dielektrika*, vol. 2, no. 2086–9487, pp. 2–2, 2020.

[12] K. Naveena, S. Singh, S. Rathod, and A. Singh, "Hybrid ARIMA-ANN Modelling for Forecasting the Price of Robusta Coffee in India," *Int. J. Curr. Microbiol. Appl. Sci.*, vol. 6, no. 7, pp. 1721–1726, 2017, doi: 10.20546/ijcmas.2017.607.207.

[13] I. P. Sutawinaya, I. N. Gede, A. Astawa, N. Kadek, and D. Hariyanti, "Perbandingan Metode Jaringan Saraf Tiruan Pada Peramalan Curah Hujan," *J. Log.*, vol. 17, no. 2, pp. 92–97, 2017.

[14] S. Kasus *et al.*, "Peramalan Curah Hujan Dengan Pendekatan Seasonal Autoregressive Integrated Moving Average (Sarima) Rainfall Forecasting Using Seasonal Autoregressive Integrated Moving Average Model (Sarima)," vol. 11, pp. 63–74, 2017.

[15] Y. Andrian and M. R. Wayahdi, "Analisis Algoritma Inisialisasi Nguyen-Widrow Pada Proses Prediksi Curah Hujan Kota Medan Menggunakan Metode Backpropagation Neural Network," *Semin. Nas. Inform. 2014*, pp. 57–63, 2014, [Online]. Available: http://e-journal.potensi-utama.ac.id/ojs/index.php/SNIf/article/view/171/118

[16] D. Learning, *Charu C. Aggarwal A Textbook 123 Neural Networks and Deep Learning*. [Online]. Available: https://doi.org/10.1007/978-3-319-94463-0

[17] I. Guntoro, D. M. Midyanti, and R. Hidayati, "Penerapan Dropout Pada Jaringan Saraf Tiruan Backpropagation Dalam Mengklasifikasi Tingkat Fine Fuel Moisture Code (Ffmc) Untuk Kebakaran Hutan Dan Lahan," *J. Komput. dan Apl.*, vol. 10, no. 01, pp. 114–123, 2022.

[18] R. Ayu, R. Gernowo, D. Fisika, F. Sains, U. Diponegoro, and S. E-, "Metode Autoregressive Integrated Movingaverage (Arima) Dan Metode Adaptive Neuro Fuzzy Inference System (Anfis) Dalam Analisis Curah Hujan," *Berk. Fis.*, vol. 22, no. 1, pp. 41–48, 2019.

[19] C. Twumasi and J. Twumasi, "Machine learning algorithms for forecasting and backcasting blood demand data with missing values and outliers: A study of Tema General Hospital of Ghana," *Int. J. Forecast.*, vol. 38, no. 3, pp. 1258–1277, 2022, doi: 10.1016/j.ijforecast.2021.10.008.

[20] B. A. Safitri, A. Iriany, and N. W. S. Wardhani, "Perbandingan Akurasi Peramalan Curah Hujan dengan menggunakan ARIMA, Hybrid ARIMA-NN, dan FFNN di Kabupaten Malang," *Semin. Nas. Off. Stat.*, vol. 2021, no. 1, pp. 245–253, 2021, doi: 10.34123/semnasoffstat.v2021i1.853.

[21] I. K. Hasan and Ismail Djakaria, "Perbandingan Model Hybrid ARIMA-NN dan Hybrid ARIMA-GARCH untuk Peramalan Data Nilai Tukar Petani di Provinsi Gorontalo," *J. Stat. dan Apl.*, vol. 5, no. 2, pp. 155–165, 2021, doi: 10.21009/jsa.05204.

[22] W. Estiningtyas, F. Ramdhani, and E. Aldrian, "Analisis Korelasi Curah Hujan Dan Suhu Permukaan Laut Wilayah Indonesia, Serta Implikasinya Untuk Prakiraan Curah Hujan (Studi Kasus Kabupaten Cilacap)," *J. Agromet Indones.*, vol. 21, no. September, pp. 46–60, 2007.

[23] R. L. Fraiha Lopes, S. G. C. Fraiha, H. S. Gomes, V. D. Lima, and G. P. S. Cavalcante, "Application of Hybrid ARIMA and Artificial Neural Network Modelling for Electromagnetic Propagation: An Alternative to the Least Squares Method and ITU Recommendation P.1546-5 for Amazon Urbanized Cities," *Int. J. Antennas Propag.*, vol. 2020, 2020, doi: 10.1155/2020/8494185.