International Journal on Advanced Science Engineering Information Technology

Vol.12 (2022) No. 6
ISSN: 2088-5334

Comparing Restricted Boltzmann Machine – Backpropagation Neural Networks, Artificial Neural Network – Genetic Algorithm and Artificial Neural Network – Particle Swarm Optimization for Predicting DHF Cases in DKI Jakarta

Bevina D. Handari [a,*], Dewi Wulandari [a], Nessa A. Aquita [a], Shafira Leandra [a], Devvi Sarwinda [a], Gatot F. Hartono [a]

[a] *Department of Mathematics, Universitas Indonesia, Depok, 16424, Indonesia*
Corresponding author: [*]bevina@sci.ui.ac.id

Abstract*— Dengue hemorrhagic fever (DHF) is a common disease in tropical countries such as Indonesia that is often fatal. Early predictions of DHF case numbers help reduce the risk of community transmission and help related authorities develop prevention plans and strategies. Previous research shows that temperature, rainfall, and humidity indirectly affect DHF spread patterns. Therefore, this research uses and compares three machine learning models—restricted Boltzmann machine-backpropagation neural network (RBM-BPNN), artificial neural network-genetic algorithm (ANN-GA), and artificial neural network-particle swarm optimization (ANN-PSO)—to predict DHF case numbers in DKI Jakarta, the capital of Indonesia, which is in the DHF red zone. RBM and PSO are used to calculate optimal initial weight and bias before starting the prediction stage with ANN; meanwhile, GA updates weight and bias during the backward pass in ANN. The data includes temperature, rainfall, and humidity, plus previous DHF case data for five districts in DKI Jakarta from Jan. 6, 2009, to Sept. 25, 2017. We used Arima, Autocorrelation, and Pearson correlation for pre-processing data. The DHF case data fluctuates strongly and requires the moving averages method. The data consists of 70% training data and 30% testing data. The results show that each district requires different model architectures for the best predictions. `The best RMSE prediction of DHF cases with RBM-BPNN in Central Jakarta is 3,78%; the best RMSEs using ANN-GA in North and East Jakarta are 5,65% and 5,99%, respectively. The ANN-PSO model had the largest RMSE value in every district, with an average of 8,43%.

***Keywords*—Dengue hemorrhagic fever; machine learning; restricted boltzmann machine – backpropagation neural network; artificial neural network – genetic algorithm; artificial neural network – particle swarm optimization.**

Manuscript received 4 Sep. 2021; revised 15 Dec. 2021; accepted 17 Jan. 2022. Date of publication 31 Dec. 2022.

I. INTRODUCTION

Dengue hemorrhagic fever (DHF) is a contagious disease that affects humans and is spread by the *Aedes aegypti* mosquito, the primary vector in some parts of the world, and *A. albopictus*, which is a secondary vector [1]. DHF is most common in tropical countries such as Indonesia and is often fatal. According to Maula et al. [2], Southeast Asia and the western Pacific are where the majority of DHF cases are found. Indonesia has one of the highest DHF case numbers in the world due to its tropical climate, which is ideal for mosquito growth. The overall case-fatality and morbidity rates for DHF in Indonesia are relatively high [3]. Therefore, an outbreak in a densely populated city will adversely affect the social life and economy in that city. The Health Ministry [4] announced that in 2019, twice as many DHF cases were recorded in Indonesia (over 110,000) than were the previous year, with the province of DKI Jakarta having the fourth-highest number of cases. In DKI Jakarta, the Health Department recorded 971 DHF cases in the first quarter of 2020, with West Jakarta having the highest cases with 269 cases [5]. Some of the factors that affect the spread of DHF include rainfall [6], air temperature [6], humidity [6], soil type [6], elevation [7], population density [7], and amount of sunlight [8]. According to Lai [8], climate factors play a crucial part in the transmission cycles of DHF. Climate can also help or hinder the growth and development of mosquitoes [7]. According to Herath et al. [9], humidity affects mosquitoes as it determines the resilience of the tracheas through which they breathe. Based on this explanation, much

research has been done to predict the number of DHF cases using weather variables.

Machine learning is a popular tool for prediction-making. It uses data for learning and uses the results to obtain information. Machine learning algorithms are ideally suited for such tasks. Indeed, they have a profound impact across a wide range of application fields because of their ability to aid learning and discovery. One such complex system is the interplay of human, climate and mosquito dynamics that give rise to the transmission of mosquito-borne diseases such as dengue [10], and yields information in the form of a prediction of DHF case numbers. One such form of machine learning research was carried out in Colombia by Zhao et al. [1], comparing artificial neural networks (ANN) and random forrest to predict the probable number of DHF cases based on environmental and meteorological conditions predictors. Another research by Herath et al. [9] involved designing an ANN model based on the effects of average temperature and humidity with a lag of 1-4 weeks, cumulative rainfall with a lag of 4 weeks, and the number of previously reported cases to predict the number of DHF cases in Sri Lanka.

The most often used machine learning model is an artificial neural network (ANN) with a backpropagation (BP) algorithm, also referred to as a backpropagation neural network (BPNN). BPNN is adapted from the mechanism underpinning the human nervous system and is highly effective in studying large-scale input data [9]. This adaptation is the mechanism by which an ANN transmits information between neurons through weighted connections. However, the main drawback of a BPNN is that it randomly determines the initial weight of connections [11]. This can be overcome with pre-training using restricted Boltzmann machines (RBM) [12] during the early training stage before training with the BPNN.

Another problem with implementing an ANN that can be encountered during predictions is its slow convergence with the use of gradient descent-based backpropagation in the training process, and it is likely to be stuck in a local minimum. Therefore, the use of well-developed metaheuristic methods can yield a solution. One of these is particle swarm optimization (PSO), as proposed by Kennedy and Eberhart. PSO algorithms are widely used for training neural networks. In addition, PSO-based ANN has a superior training performance with faster convergence rates and better prediction capabilities than any traditional ANN [13].

Chiang et al. [14] aver that ANN-BP does not often determine the global minimum of an error function, and one of the methods that can determine this during the ANN training process is the genetic algorithm. However, according to Chiang et al. [14], its computational capabilities are slower and more laborious due to finding the fitness value for every chromosome of every generation. One of the researchers who used ANN-GA for the dengue fever problem is Sabara et al. [15], who used ANN-GA to classify the diagnosis of dengue fever in Tegal City, Indonesia, who used ANN-GA to predict a DHF outbreak in Malaysia.

Based on the increasing number of DHF cases in DKI Jakarta, this research compares the implementation of three machine learning methods—RBM-BPNN, ANN-GA, and ANN-PSO—to predict the number of DHF cases in DKI Jakarta (excluding Kepulauan Seribu Regency). In RBM-BPNN, RBM is used in pre-training, or RBM precedes BPNN. In ANN-PSO, PSO is used to calculate optimal initial parameters before moving on to the prediction stage with ANN. Finally, unlike the previous two models, ANN-GA updates weight and bias during the backward pass. To that end, two forms of secondary data were used: the daily weather data in each district in DKI Jakarta, as obtained from the Department of Meteorology, Climatology, and Geophysics from Jan. 1, 2008, to Dec. 31, 2018, and the daily number of DHF cases from Jan. 1, 2008, to Dec. 31, 2017, as obtained from the Epidemiology Surveillance Section website [16].

This article is presented as follows. Section II describes the material and methods (RBM-BPNN, ANN-GA, and ANN-PSO). Section III, the proposed method, includes determining the pre-processing data, smoothing data, and hyperparameters. The results and discussion are presented in section IV. The result is divided into two parts: first, the results are based on each model's evaluation in five districts. Second, we use some visualizations to compare the DHF case predictions obtained from each best model in every district. Finally, section V shows some conclusions.

II. MATERIALS AND METHODS

A. Materials

This research used two sets of secondary data. One of these is the daily weather data from the Meteorology, Climatology and Geophysics Agency (BMKG) as recorded from five observation stations, one for each district in Jakarta: Halim (East Jakarta), Tanjung Priok (North Jakarta), Pondok Betung (South Jakarta), Kemayoran (Central Jakarta) and Cengkareng (West Jakarta) from Jan. 1, 2008, to Dec. 31, 2018. These data consist of average temperature (in degrees Celsius), rainfall (in millimeters), and humidity (%). The other data set is the daily DHF case number, as taken from the website of the Epidemiology Surveillance Section of the Health Department of DKI Jakarta [16]. This data is the number of DHF sufferers as recorded in 163 hospitals in DKI Jakarta, sorted into the same five districts as the first data set, and spans the period from Jan. 1, 2008, to Dec. 31, 2017.

B. Artificial Neural Network and Backpropagation

An artificial neural network (ANN) is a machine learning model that adapts a biological neural network [17]. There are several types of ANN [17]. This research used feedforward neural networks (FFNN) in which the network connects to the output without an inner loop. Specifically, a subset of FFNN is called the multi-layer perceptron (MLP). MLP is an ANN with a hidden layer with a non-linear activation function between the output and input layers. An ANN with a backpropagation (BP) algorithm is called a backpropagation neural network (BPNN). backpropagation is used for weight updating and bias in ANN.

C. Restricted Boltzmann Machine – Backpropagation Neural Network

RBM is a form of unsupervised learning and means of neural network learning that can detect and extract input data features for use in pre-training prior to supervised learning [12]. RBM uses a visible layer as the input layer and a hidden layer. Both layers are used for determining the connection

weight and initial bias for BPNN. In mechanical statistics, RBM uses the concept of energy in which energy is related to the probability of the connection between units in the visible and hidden layers. In RBM-BPNN, pre-training with RBM is used to obtain the connection weight between the visible and hidden units as well as the bias for the hidden unit. Connection weight and bias are used to initiate the BPNN process. The RBM-BPNN scheme is shown in Fig. 1 below:

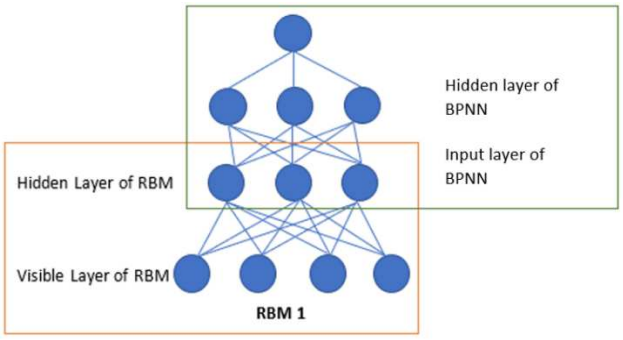

Fig. 1 RBM-BPNN Scheme

The general architecture of an RBM is shown below in Fig. 2.

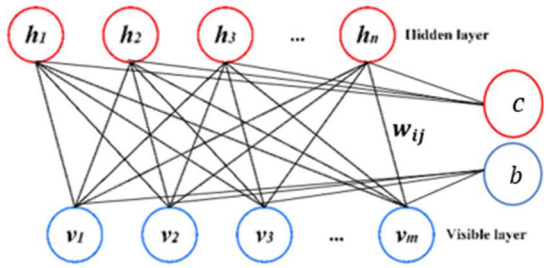

Fig. 2 The general architecture of an RBM [18]

Based on Fig. 2, the RBM has two layers: a visible layer (v) and a hidden layer (h) where $v^T = (v_1, v_2, v_3, \dots, v_m)$ and $h^T = (h_1, h_2, h_3, \dots, h_n)$. While b and c are the vector biases for the visible and hidden units, respectively. According to Fig2, $b = (b)$ and $c = (c)$. Therefore, $b_i = b$ for every i and $c_j = c$ for every $j, i = 1,2,\dots,m, j = 1, 2, \dots, n$. These initial bias values are the ones most often used [19]. The bias values for each neuron can differ in line with the learning process. The connection between the visible (v_i) and hidden units (h_j) is a line with a connection weight of w_{ij}.

The probability of the connection between units in the visible and hidden layers can be calculated using the following energy function [12]:

$$E(v_i, h_j) = -\sum_{i=1}^{n} b_i v_i - \sum_{j=1}^{m} c_j h_j - \sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij} v_i h_j, \quad (1)$$

where m is the number of nodes in the visible layer, and n is the number of nodes in the hidden layer. In RBM learning, equation (1) must be minimized; therefore, we must find data for v and h that minimizes equation (1).

This requires implementing the positive and negative phases in Gibbs sampling as follows [12]:

1) Input data $v^T = (v_1, v_2, v_3, \dots, v_m)$ consists of $m =$ 4 variables: temperature, rainfall, humidity, and previous incident numbers. Randomly determine the connection weight w_{ij}, bias b_i and c_j with a value between 0 and 1.
2) Positive Phase
   Let t be an iteration for Gibbs sampling starting from $t = 0$, where $v = v^{(0)}$. Calculate $p(v)$ for every j [27,28] as follows:

$$p\left(h_j^{(0)} = 1 | \mathbf{v}\right) = \frac{e^{c+\sum_{i=1}^{m} w_{ij} v_i}}{1 + e^{c+\sum_{i=1}^{m} w_{ij} v_i}}. \quad (2)$$

   After that, take a sample $h_j^{(0)}$ using Gibbs sampling with the following rule:

$$h_j^{(0)} = \begin{cases} 1, & p\left(h_j^{(0)} = 1 | \mathbf{v}\right) > H_j^{(0)}, \\ 0, & p\left(h_j^{(0)} = 1 | \mathbf{v}\right) \leq H_j^{(0)}, \end{cases} \quad (3)$$

   where $H_j^{(0)}$ is the hidden unit generator value at $t = 0$ which was randomly determined on (0,1).
3) Negative Phase
   Calculate $p(h)$ using the previous results of sample $h_j^{(0)}$, so that $h_j = h_j^{(0)}$, for every i using the following formula

$$p(v_i = 1 | \mathbf{h}) = \frac{e^{b+\sum_{j=1}^{n} w_{ij} h_j}}{1 + e^{b+\sum_{j=1}^{n} h_j w_{ij}}}. \quad (4)$$

   Henceforth, take a sample $v_i^{(1)}$ using Gibbs sampling based on the following rules:

$$v_i^{(1)} = \begin{cases} 1, & p(v_i = 1 | \mathbf{h}) > V_i^{(0)}, \\ 0, & p(v_i = 1 | \mathbf{h}) \leq V_i^{(0)}, \end{cases} \quad (5)$$

   where $V_i^{(0)}$ is the initial visible unit value at $t = 0$ which was randomly determined on (0,1).
4) The last step of Gibbs sampling is to determine the final reconstructed value of $h_j^{(1)}$ using equation (3) with $t = 1$, and the results of $v^{(1)}$ as the value of v. After obtaining the required values of v and h, update the connection weight w_{ij}, bias b_i, and c_j using the gradient ascent method:

$$w'_{ij} = w_{ij} + \alpha\left(v_i^{(0)} h_j^{(0)} - v_i^{(1)} h_j^{(1)}\right), \quad (6)$$

$$b'_i = b_i + \alpha\left(v_i^{(0)} - v_i^{(1)}\right), \quad (7)$$

$$c'_j = c_j + \alpha\left(h_j^{(0)} - h_j^{(1)}\right), \quad (8)$$

   where α is the learning rate.

Resume the process with BPNN training process consisting of the forward and backward passes as follows:

1) Forward Pass:
   Calculate the hidden unit y_j as follows:

$$y_j = f(S_j) = f\left(\sum_{i=1}^{n} x_i w_{ij} + b_j\right), \quad y_j \in (0,1) \tag{9}$$

where w_{ij} is the initial weight value for the connection between the input and hidden layers, which uses w'_{ij} from RBM, and b_j uses the hidden unit bias value from RBM (c'_j). The function f is a sigmoid function determined as follows:

$$f(x) = \frac{1}{1+e^{-x}}, x \in \mathbb{R}. \tag{10}$$

Calculate the output unit z values as follows:

$$z = f(S_z) = f\left(\sum_{i=1}^{n} y_j w_{zj} + c\right), \quad y_j \in (0,1) \tag{11}$$

where w_{zj} is the initial weight value for the connection between the hidden and output layers, and c is the output unit bias value. Both w_{zj} and c are randomly determined on (0,1).

2) Backward Pass:

A Calculate the changes in connection weight w_{ij}, bias b_j with the following gradient descent:

$$w'_{ij} = w_{ij} - \alpha \frac{\partial L(\mathbf{w}, \mathbf{b})}{\partial w_{ij}}, \tag{12}$$

$$b'_j = b_j - \alpha \frac{\partial L(\mathbf{w}, \mathbf{b})}{\partial b_j}, \tag{13}$$

where $L(\mathbf{w}, \mathbf{b})$ is the loss function (RMSE) in which $L(\mathbf{w}, \mathbf{b}) = (f(S_z) - z)^2$ and z is the target data, $i = 1, 2, \ldots, m, j = 1, 2, \ldots, n$. In this case, the target data is the incident numbers. Repeat the BPNN training process until an optimal output is obtained. Note that the value of b'_i in equation (7) is not used in the process since the BPNN model does not need a bias vector in the input layer.

D. Artificial Neural Network with Genetic Algorithm

Unlike RBM-BPNN, ANN-GA updates weight and bias during the backward pass. In ANN training with a GA, a gene in a genetic algorithm is represented with weight and bias and consists of multiple chromosomes, while an individual chromosome is represented as a matrix that already has a weight and bias that is used in the existing neural network. The fitness function is the loss function of the ANN. The end goal of using GA in ANN training is to determine the best individual (chromosome) with the smallest fitness value. The fitness value is determined by RMSE.

Based on the data used, ANN-GA training follows the procedure outlined below:

1) Input an $N \times n$ matrix x and a vector $z = [z_1, z_2, \ldots, z_N]^t$, where n is the number of input variables, and N is the amount of data. Matrix x consisting of variables of temperature, rainfall, humidity, and previous incident numbers. Data target z consists of the numbers of incident data.
2) Determine the population size, the number of generations, retain rate (p), crossover probability (P_c), and mutation probability (P_m).
3) Set the number of generations to 0.
4) Randomly determine the initial population (every chromosome has a matrix with weight $w = [w_1, w_2, \ldots, w_n]^t$ and a bias parameter w_0), where n is the number of neurons in the input layer.
5) Use a forward pass on the ANN for every chromosome in the population, where $i = 1,2,\ldots,N$:

$$a_i = \sum_{j=1}^{n} w_j x_{ij} + w_0, \tag{14}$$

$$y_i = f(a_i), \tag{15}$$

where the activation function f is a sigmoid function as in equation (10). Furthermore, determine the prediction error of y_i on target z with

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - z_i)^2}. \tag{16}$$

6) Use a backward pass on the ANN by updating the weight and bias with a genetic algorithm begin with selecting the numbers of chromosomes for the next generation based on the fitness value (RMSE) according to the retained value of p (truncation selection [20] is used here). Place the results of the chromosome selection in the reproduction pool. After that, create a new population by conducting a gene evolution through crossover and mutation. The crossover operation uses discrete crossover [21] and uses the crossover probability value of $P_c = 0.7$. The mutation operation uses uniform mutation [22] on the chromosomes resulting from crossover and uses the mutation probability value of $P_m = 0.25$. Repeat the process until the reproduction pool is the same size as the initial population before selection.
7) Set generation = generation + 1, and repeat from step 5, until the required number of generations has been reached.
8) Determine the best chromosome that has the best weight and bias values to be used in the forward pass.

After ANN-GA training, continue the process with a forward pass on the testing data using the weight and bias from the best chromosome from the training process.

E. Artificial Neural Network – Particle Swarm Optimization

RBM The basic concept of the PSO algorithm is a simulation of a swarm of animals. As with RBM, PSO is used to calculate optimal initial weight and bias before moving on to the prediction stage with ANN. In the PSO algorithm, we need some parameters: inertia weight (ω), acceleration

coefficient of the cognitive component (c_1), and the acceleration coefficient of the social component (c_2), where both c_1 and c_2 are positive. In this research, the ranges for c_1, c_2 and ω are 0.5-1.5, 0.5-2, and 0.4-1.2, respectively. The optimal weight and bias values are the ones linked to the PSO output with the best fitness function (RMSE) value. The other parameters in the PSO algorithm are r_1 and r_2 which are two random uniformly distributed numbers in [0, 1]. The number of hidden neurons and bias in an ANN must be determined previously. Based on the data, the ANN-PSO procedure is as follows [23]:

1) Input the numbers of data as the swarm population (N), number of iterations (T), inertia weight (ω), acceleration coefficient of the cognitive component (c_1), acceleration coefficient of the social component (c_2), and the target variable data $\boldsymbol{y}$ in the DHF case data. The number of particles in one population is dependent on the number of variables on weather data and the target variable $\boldsymbol{y}$, and the architecture of the corresponding ANN.
2) For every particle in the population, in each iteration t do step from 3-9 below.
3) Randomly create all weight value set and bias value set in PSO model and denoted as the position components ($X^{(t)}$). Initialize the speed component ($V^{(t)}$) randomly. $V^{(t)}$are called the initial positions for every particle.
4) For each particle j, calculate the following formulas by using the $\boldsymbol{m} = \{m_1, m_2, \dots, m_D\}^{\mathrm{T}}$, which is weather data and DHF case data, and the target variable data $\boldsymbol{y}$ is the DHF case data. The initial values for $w_i, i = 1,2,\dots,D$ is randomly selected.

$$a_j = \sum_{i=1}^{D} w_i m_i + w_0, \tag{17}$$

$$y_j = f(a_j), \tag{18}$$

where w_0 is the bias parameter and w_i is the weight and f is the sigmoid function.

Henceforth, evaluate the objective function $F(P_j)$, which are the prediction results of y_j on target $\hat{y}_j$ using the following RMSE:

$$F(P_j), = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(y_j - \hat{y}_j)^2}, \tag{19}$$

where N is the amount of data.

5) Determine the particle best (P_{best}) and global best (Pg_{best}) parameters. P_{best} is the position of the j^{th} particle that yields the smallest evaluation value of the objective function $F(P_j)$ until the $(t-1)^{\text{th}}$ iteration; Pg_{best} is the best position among all particles that yields the smallest value of the objective function P_{best}. For the first iteration, P_{best} is the initial position of every particle.
6) Create two random vectors r_1 and r_2. The number of random vectors is equal to the number of particles.
7) Update the speed $V^{(t)}$ and position $X^{(t)}$ components on the t^{th} iteration for every individual as follows:

$$V^{(t)} = \omega^{(t)}V^{(t-1)} + \left[c_1 r_1\left(P_{best} - X^{(t-1)}\right) + c_2 r_2\left(Pg_{best} - X^{(t-1)}\right)\right], \tag{20}$$

$$X^{(t)} = X^{(t-1)} + V^{(t)}. \tag{21}$$

The initial individual speed $V^{(0)}$ was chosen randomly. Use the results of step 7 to update the weight and bias parameter.

8) Repeat steps 3-7 until an optimal result is obtained or the required number of iterations T has been reached.
9) Use the optimal weight and bias parameter sets from PSO as the weight and bias parameter sets in ANN. This ANN only requires one forward pass. The activation functions for ANN are the Tanh and ReLU functions on the hidden and output layers, respectively.

During implementation, 10 ANN-PSO simulations were carried out; each one had 10 iterations of 15 possible model combinations, each with between 3 and 7 hidden neurons, and populations of 20, 40, or 60 individuals. The ANN-PSO model selected was the one that yielded the smallest average RMSE value from all 10 simulations.

F. The Proposed Method

The data used is the daily data from Jan. 6, 2009, to Sept. 25, 2017, which omits data from cases between Jan. 1, 2008, to Jan. 5, 2009, due to missing values. Arima is used to replace missing values in both data sets since Arima could generate a time-series model from the existing data to replace the missing values. For the average relative temperature and humidity variables, determine the average weekly value for each week; for the average rainfall and case numbers, find the cumulative value for each week. After missing values and noise have been accounted for, convert the daily data (3185 observations) into weekly data (455 observations).

After pre-processing, determine the time lag for each predictor variable concerning the weekly case numbers as the target variables. This is necessary due to the time it takes for weather variables to affect DHF case numbers based on the life cycle of mosquitoes that are DHF vectors (which are also affected by weather variables). The time lag is between 1 and 8 weeks based on research by Guo et al. [24]. We used autocorrelation to calculate the time lag between the case number for the current week and those of previous weeks [25]. We used cross-correlation using Pearson correlation [25] to calculate the time lag between the number of cases for the current week and previous average temperature, average rainfall, and humidity, respectively.

The final stage is data normalization before it is used in the ANN training process [26]. This is done by mapping input data to a certain scale using the min-max normalization as follows [26]:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}, \tag{22}$$

where x is the data to be normalized on the scale [0,1], x_{norm} is normalized data, and x_{max} and x_{min} are the maximum and minimum values of x, respectively.

The DHF case data fluctuates strongly, prompting the use of moving averages. A moving average is a form of time-series data made by taking the average from several sequential values of another time-series data [27]. According to Raudys et al. [27], moving averages can be used to show trends in fluctuating data to smooth it out, thus improving the stability of the prediction results and removing random variables. The simplest moving average formula is as follows:

$$MA_t^n = \frac{1}{n}\sum_{i=1}^{n} x_{t+i-1}, \tag{23}$$

where $\mathbf{x} = x_1, x_2, x_3, \ldots, x_p$ is time-series p data, n is the moving average period, and t is the index of the smoothed time series data.

We designed the best RBM-BPNN, ANN-GA, and PSO-ANN during the implementation stage through hyperparameter tuning using a grid search. The hyperparameters for RBM-BPNN are the moving average, the number of neurons in the hidden layer, and the learning rate. The other two models do not require a learning rate. However, ANN-GA and PSO-ANN have population hyperparameters of 20, 40, and 60 individuals each. Whereas ANN-GA has 400 generations, PSO-ANN does not need this parameter. The ranges of moving average periods and neurons for all three models were 2 to 5 and 3 to 7, respectively. The RBM-BPNN learning rates were 0.05, 0.025 and 0.01, respectively.

A good model has a small training error with minor differences between the training and testing errors [28]. Therefore, this research determines the five best hyperparameter combinations based on training error, ultimately selecting the hyperparameter combination with the smallest absolute difference between training and testing errors (gap). The model evaluation uses root mean squared error (RMSE) [29], whose formula is as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(Y_i - Y_i')^2}{n}}, \tag{24}$$

where n is the number of observations, Y_i is the actual value, and Y_i' is the predicted value. RMSE is interpreted as the standard deviation (σ) that shows the distribution of values for a data set from the expected value.

III. RESULTS AND DISCUSSION

The analysis is divided into two parts: first, we compare the model evaluation from each model in five districts. Second, we compare the DHF case predictions obtained from each best model in every district using some visualizations. Tables 1, 2, and 3 show the RMSE for incidence prediction using RBM-BPNN, ANN-GA, and ANN-PSO, respectively, in each district with 70% training data (313 data) and 30% testing data (134 data). Here, we use a 70%/30% proportion since the empirical studies show that 20-30% testing data and 70-80% training data yields the best results [30]. The three tables also show information about the hyperparameter resulting from the grid search related to the best model.

TABLE I
RMSE FOR INCIDENCE PREDICTION USING RBM-BPNN (70%/30%)

Districts	Moving Average	Hidden Neuron	Learning Rate	Root Mean Square (RMSE)		
				Training	Testing	Gap
North Jakarta	5	6	0.025	0.027067	0.056797	0.029730
West Jakarta	5	6	0.025	0.034840	0.058598	0.023758
East Jakarta	5	7	0.05	0.017893	0.061574	0.043681
Central Jakarta	5	5	0.01	0.027109	0.037857	0.010748
South Jakarta	5	5	0.05	0.016683	0.037300	0.020616

Table 1 shows the best hyperparameter combinations of the RBM-BPNN model in each district based on the training error. The table shows that not all districts yield the same RBM-BPNN model, as shown in their differing numbers of hidden neurons and learning rates. This also applies to the other two models in Tables 2 and 3. In this table, the smallest gap of the RBM-BPNN model is in Central Jakarta with training and testing RMSEs of 0.027109 and 0.037857, respectively. The largest gap is in East Jakarta, with a value of 0.043681. This shows that RBM-BPNN is not yet optimal for studying incidence in West Jakarta.

TABLE I
RMSE FOR INCIDENCE PREDICTION USING ANN-GA (70%/30%)

Districts	Moving Average	Hidden Neuron	Population Size	Root Mean Square (RMSE)		
				Training	Testing	Gap
North Jakarta	5	5	60	0.026585	0.056518	0.029933
West Jakarta	5	5	60	0.035100	0.056398	0.021298
East Jakarta	5	5	60	0.016933	0.059954	0.043021
Central Jakarta	5	6	60	0.027709	0.038510	0.010801
South Jakarta	5	3	60	0.016046	0.040496	0.024450

Table 2 shows the best hyperparameter combinations of the ANN-GA model in each district based on the training error. The smallest gap of the ANN-GA model is also in Central Jakarta, with a value of 0.010801 with training and testing RMSEs of 0.027709 and 0.038510, respectively. The largest gap is in East Jakarta, with a value of 0.043021. The best models all have the largest population size of 60, requiring the model to learn sufficiently from the data to predict testing data.

TABLE II
RMSE FOR INCIDENCE PREDICTION USING ANN-PSO (70%/30%)

Districts	Moving Average	Hidden Neuron	Population Size	Root Mean Square (RMSE)		
				Training	Testing	Gap
North Jakarta	5	4	60	0.031110	0.103202	0.072092
West Jakarta	5	6	40	0.027651	0.145030	0.117379
East Jakarta	5	4	40	0.021116	0.065179	0.044063
Central Jakarta	5	4	60	0.042866	0.065109	0.022243
South Jakarta	5	3	60	0.031305	0.043379	0.012074

In Table 3, they are the best hyperparameter combinations of the ANN-PSO model in each district based on the training error. The smallest gap of the ANN-PSO model is in South Jakarta with a value of 0.012074 with training and testing RMSEs of 0.031305 and 0.043379, respectively. The largest gap is in West Jakarta, with a value of 0.117379.

From the 3 tables above, the best models all use a moving average of 5. Unlike the results for ANN-GA in Table 2, the best ANN-PSO model does not have to use the largest population. In other words, the best model can have a population size of 40 or 60, and different numbers of hidden neurons (3, 4, or 6). From the 3 tables above, the best RMSE prediction of DHF cases with RBM-BPNN in Central Jakarta is 0.037857; the best RMSE prediction using ANN-GA in North and East Jakarta are 0.056518 and 0.059954, respectively. The ANN-PSO model had the largest RMSE value in most districts, with an average of 0.08438.

The five figures below show visualizations from the three best models in predicting DHF case data on testing data for each district with a data composition of 70% training data and 30% testing data. Generally, all three models can reasonably predict case data on testing data until the 70th week, after which they are less effective. Therefore, analysis of the visualization of the results of the implementation of the model focuses on testing data after the 70th week.

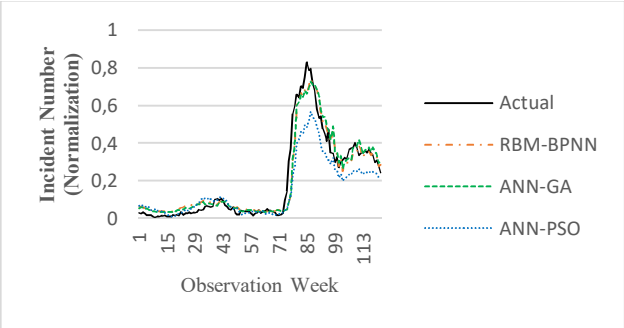

Fig. 3 DHF Prediction in North Jakarta with 70% Training Data

In Fig. 3, both the RBM-BPNN and ANN-GA models provided better testing data prediction than the ANN-PSO model. This prediction is consistent with the results in Tables 1, 2, and 3, where the ANN-PSO model had the highest training and testing RMSE of all three models (0.031110 and 0.103202, respectively). From the training and testing RMSEs shown in Tables 1 and 2, the RMSE of the ANN-GA model is slightly less than that of the RBM-BPNN model.

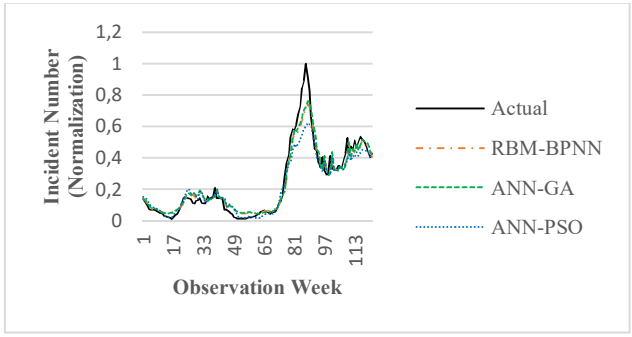

Fig. 4 DHF Prediction in West Jakarta with 70% Training Data

The results of the DHF case prediction in West Jakarta for all three models show that they struggled to predict the significant increase in cases around the 87th week. According to Tables 1, 2, and 3, the ANN-PSO model should have the smallest training RMSE value of 0.027651. However, the testing RMSE is inferior to that of the other two models. This result shows that whatever the ANN-PSO model has learned is still insufficient for predicting data testing.

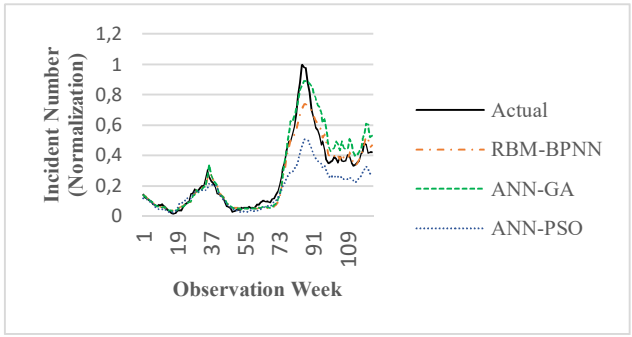

Fig. 5 DHF Prediction in East Jakarta with 70% Training Data

Fig. 5 shows that the ANN-GA model is best at predicting testing data for the highest number of DHF cases, although its prediction is still inferior to that of the RBM-BPNN model in subsequent weeks. This result is consistent with the three tables above, showing that the ANN-GA model yields the smallest training and testing RMSEs in East Jakarta. The ANN-PSO model struggled to predict data after the 76th week, which is consistent with the results shown in all tables, where the ANN-PSO model has the highest training and testing

RMSE values in East Jakarta, at 0.021116 and 0.0652179, respectively.

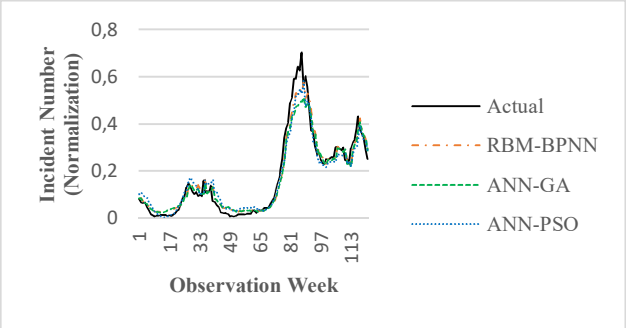

Fig. 6 DHF Prediction in Central Jakarta with 70% Training Data

Fig. 6 shows that around the 80th and 89th weeks, the RBM-BPNN and ANN-PSO models coped with the positive spike in DHF cases better than the ANN-GA model. However, in subsequent weeks, the RBM-BPNN and ANN-GA models yielded better predictions than the ANN-PSO model. This result is consistent with the three tables above, where the RMB-BPNN model had the lowest training and testing RMSEs in Central Jakarta.

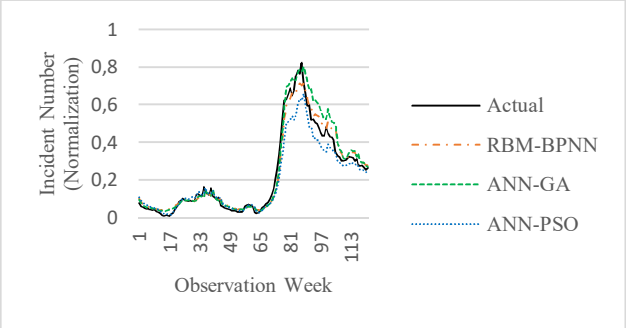

Fig. 7 DHF Prediction in South Jakarta with 70% Training Data

In Fig. 7, the ANN-GA model yielded the best prediction in testing data for the highest number of DHF cases, while the ANN-PSO model struggled to do the same. This is consistent with the results in Table 1, 2, and 3 above, where the ANN-PSO model had the highest RMSE value of all three models.

IV. Conclusion

This research used the RBM-BPNN, ANN-GA and ANN-PSO models to predict DHF case data in five districts throughout DKI Jakarta using weather data variables (rainfall, humidity, and temperature) and DHF case numbers. Based on RMSE values, the results showed that each district required a different best model architecture. The RBM-BPNN model had the best RMSE prediction of DHF cases at 3,78% in Central Jakarta, while the ANN-GA model had the smallest RMSE values in North and East Jakarta, of 5,65% and 0,59%, respectively. However, the ANN-PSO model had the largest value in nearly every district, with an average RMSE of 8,43%. Based on the visualization of the implementation of the testing data, every model struggled to predict the spike in cases in most districts.

Based on the data, generally, the RBM-BPNN and ANN-GA models yielded better predictions than the ANN-PSO model. However, the model with the smallest training RMSE value has yet to guarantee a small testing RMSE. This result shows that whatever a model has learned in training data is still insufficient for predicting testing data. Based on the RMSE values and the visualizations, the results of the RBM-BPNN and ANN-GA models do not differ significantly from each other. Even though the genetic algorithm is a metaheuristic method, in ANN-GA, it has updated weight and updated bias in the backward phase as in RBM-BPNN. Population size affects the results for models with metaheuristic methods (ANN-GA and ANN-PSO). The results also show improvements as population size increases, and this applies to the ANN-GA model, but not the ANN-PSO model.

Moreover, ANN-PSO lacks backpropagation. These differences manifest themselves in the superior prediction resulting from the ANN-GA model compared to the ANN-PSO model. To obtain a better result on data with a significant spike, a data composition other than 70% training data and 30% testing data is recommended. A training data composition that can account for large data spikes is recommended for a model that can learn more easily from training data.

Acknowledgment

Universitas Indonesia funded this research, PUTI grant scheme No.: NKB-1977/UN2.RST/HKP.05.00/2020. We also thank Indonesia's Meteorology, Climatology, and Geophysical Agency and the Jakarta Health Department for their data sets. Without their generous cooperation, this research would not have been possible.

References

[1] N. Zhao, K. Charland, M. Carabali, E. O. Nsoesie, M. M. Giroux, E. Rees, M. Yuan, C. G. Balaguera, G. Jaramillo Ramirez, and K. Zinszer, "Machine learning and dengue forecasting: Comparing random forests and artificial neural networks for predicting dengue burden at national and sub-national scales in Colombia," *PLoS Negl Trop Dis*, vol. 14, no. 9, Sep. 2020. DOI: 10.1371/journal.pntd.0008056.

[2] A. W. Maula, A. Fuad, and A. Utarini, "Ten-years trend of dengue research in Indonesia and South-east Asian countries: a bibliometric analysis," *Global Health Action*, vol. 11, no. 1, Aug. 2018. DOI: 10.1080/16549716.2018.1504398.

[3] A. A. Suwantika, A. P. Kautsar, W. Supadmi, N. Zakiyah, R, Abdulah, M. Ali, and M. J. Postma, "Cost-Effectiveness of Dengue Vaccination in Indonesia: Considering Integrated Programs with Wolbachia-Infected Mosquitos and Health Education," *International Journal of Environmental Research and Public Health*, vol. 17, no. 12, Jun. 2020. DOI: 10.3390/ijerph17124217.

[4] Antaranews, "Health ministry confirms 110.921 dengue fever cases until Oct 2019," 2019, Accessed on: Mar. 9, 2020, [Online] Available: https://en.antaranews.com/news/135984/healt-ministry-confirms.

[5] iNews ID, "Waspada, Januari hingga Maret 2020 sebanyak 971 kasus DBD di DKI," 2020, Accessed on: Mar. 15, 2020, [Online] Available: https://www.inews.id/news/megapolitan/waspada-januari-hingga-maret-2020-sebanyak-971-kasus-dbd-di-dki.

[6] T. W. Kesetyaningsih, S. Andarini, Sudarto, and H. Pramoedyo, "Determination of environmental factors affecting dengue incidence in Sleman district, Yogyakarta, Indonesia," *African Journal of Infectious Disease: AJID,* vol. 12, sec. 1 Suppl, pp. 13-25, Mar. 2018.

[7] S. N. A. Istiqamah, A. A. Arsin, A. U. Salmah, and A. Mallongi, "Correlation study between elevation, population density, and dengue hemorrhagic fever in Kendari city in 2014-2018," *Open Access Macedonian Journal of Medical Sciences*, vol. 8, sec. T2, pp. 63-66, Sep. 2020. DOI: 10.3889/oamjms.2020.5187.

[8] Y. H. Lai "The climatic factors affecting dengue fever outbreaks in Southern Taiwan: An application of symbolic data analysis," *BioMedical Engineering Online*, vol. 17, no. 2, pp. 148, Nov. 2018. DOI: 10.1186/s12938-018-0575-4.

[9] P. H. M. N. Herath, A. A. I. Perera, and H. P. Wijekoon, "Prediction of dengue outbreaks in Sri Lanka using artificial neural networks," *International Journal of Computer Applications,* vol. 101, no. 15, pp. 0975–8887 Sep. 2014. DOI: 10.5120/17760-8862.

[10] S. F. McGough, L. Clemente, J. N. Kutz, and M. Santillana, "A dynamic, ensemble learning approach to forecast dengue fever epidemic years in Brazil using weather and population susceptibility cycles," *Journal of the Royal Society Interface*, vol.18, pp. 179, Jun. 2021. DOI: 10.1098/rsif.2020.1006.

[11] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi, "Time series forecasting using a deep belief network with restricted Boltzmann machines," *Neurocomputing*, vol. 137, pp. 47–56, Aug. 2014.

[12] A Practical Guide to Training Restricted Boltzmann Machines (Version 1), G. Hinton ed., Department of Computer Science, University of Toronto, Toronto, 2010.

[13] J. Wu, Z. Li, L. Zhu, G. Li, B. Niu, and F. Peng, "Optimized BP neural network for dissolved oxygen prediction," in *IFAC*, pp. 596-601, Jan. 2018. DOI: 10.1016/j.ifacol.2018.08.132.

[14] T. Chiang, Z. G. Che, and Z. H. Che, "Feed-forward neural networks training: a comparison between genetic algorithm and backpropagation learning algorithm," *International Journal of Innovative Computing,* vol. 7, no. 10, pp. 5839–5850, Oct. 2010.

[15] M. A. Sabara, O. Somantri, H. Nurcahyo, N. K. Achmadi, U. Latifah, and Harsono, "Diagnosis classification of dengue fever based on neural networks and genetic algorithms," *Journal of Physics: Conference Series,* vol. 1175, no. 1, Mar. 2019. DOI: 10.1088/1742-6596/1175/1/012065.

[16] Epidemiology Surveillance Section of the Health Department of DKI Jakarta. (2019). [Online]. Available: https://surveilansdinkesdki.net/chart.php.

[17] M. Islam, G. Chen, and S. Jin, "An overview of neural network," *American Journal of Neural Networks and Applications*, vol. 5, no. 1, Jun. 2019. DOI: 10.11648/j.ajnna.20190501.12.

[18] C. Ou, J. Yang, Z. Du, X. Zhang, and D. Zhu, "Integrating cellular automata with unsupervised deep-learning algorithms: a case study of urban-sprawl simulation in the Jingjintang Urban Agglomeration, China," *Sustainability*, vol. 11, no. 9, Jan. 2019. Art. no. 2464.

[19] N. R. Kumar, "Restricted boltzmann machine," 2018, Accessed on: Mar. 2020, [Online] Available: https://sites.google.com/site/nirajatweb/home/deep_learning_tutorials (2018).

[20] N. G. A. P. H. Saptarini, P. I. Ciptayani, N. W. Wisswani, I. W. Suasnawa and N. E. Indrayana, "Comparing selection method in course scheduling using genetic algorithm," in *International Conference on Science and Technology (ICST 2018),* vol. 1, Jan. 2018. DOI:10.2991/icst-18.2018.119.

[21] M. Y. Orong, A. M. Sison, and R. P. Medina, "A new crossover mechanism for genetic algorithm with rank-based selection method," in *5th International Conference on Business and Industrial Research (ICBIR),* May. 2018. DOI:10.1109/ICBIR.2018.8391171.

[22] T. Helmuth, N. F. McPhee, and L. Spector, "Program synthesis using uniform mutation by addition and deletion," in *GECCO '18*, Kyoto, Japan, Jul. 2018. DOI: 10.1145/3205455.3205603.

[23] N. Gómez, L. F. Mingo, J. Bobadilla, F. Serradilla, and J. A. C. Manzano, "Particle swarm optimization models applied to neural networks using the R language," *WSEAS Transactions on Systems*, vol. 2, no. 9, pp. 192-202, Feb. 2010.

[24] Y. Chen, J. H. Y. Ong, J. Rajarethinam, G. Yap, L. C. Ng, and A. R. Cook, "Neighbourhood level real-time forecasting of dengue cases in tropical urban Singapore," *BMC medicine*, vol. 16, no. 1, Aug. 2018.

[25] V. J Jayaraj, R. Avoi, N. Gopalakrishnan, D. B. Raja, and Y. Umasa, "Developing a dengue prediction model based on climate in Tawau, Malaysia," *Acta tropica,* vol. 197, Sep. 2019.

[26] J. M, Jo, "Effectiveness of normalization pre-processing of big data to the machine learning performance," *The Journal of the Korea institute of electronic communication sciences*, vol. *14*, no. 3, pp. 547-552, Jun. 2019. DOI: 10.13067/JKIECS.2019.14.3.547.

[27] A. Raudys, and Ž. Pabarškaitė, "Optimising the smoothness and accuracy of moving average for stock price data," *Technological and economic development of economy*, vol. 24, no. 3. pp. 984-1003, May. 2018.

[28] Y. Bengio, I. Goodfellow, and A. Courville, "Deep learning," in *MIT Press*, Cambridge, Nov. 2015.

[29] T. Chakraborty, S. Chattopadhyay, and I. Ghosh, "Forecasting dengue epidemics using a hybrid methodology," *Physica A: Statistical Mechanics and its Applications,* vol. 527, Aug. 2019. DOI: 10.1016/j.physa.2019.121266.

[30] A. Gholamy, V. Kreinovich, and O. Kosheleva, "Why 70/30 or 80/20 relation between training and testing sets: a pedagogical explanation," *Departmental Technical Reports (CS)*, 2018, [Online] Available: https://scholarworks.utep.edu/cs_techrep/1209/