ISSN: 0067-2904

Comparison of Performance Metrics Level of Restricted Boltzmann Machine and Backpropagation Algorithms in Detecting Diabetes Mellitus Disease

Dimas Aryo Anggoro*, Sri Hajiati
Department of Informatics, Universitas Muhammadiyah Surakarta, Indonesia

Received: 1/9/2021 Accepted: 9/12/2021 Published: 28/2/2023

Abstract

Diabetes is a disease caused by high sugar levels. Currently, diabetes is one of the most common diseases in the number of people with diabetes worldwide. The increase in diabetes is caused by the delay in establishing the diagnosis of the disease. Therefore, an initial action is needed as a solution that requires the most appropriate and accurate data mining to manage diabetes mellitus. The algorithms used are artificial neural network algorithms, namely Restricted Boltzmann Machine and Backpropagation. This research aims to compare the two algorithms to find which algorithm can produce high accuracy, and determine which algorithm is more accurate in detecting diabetes mellitus. Several stages were involved in this research, including data collection, data pre-processing, data processing, and evaluation models. This research shows that the Restricted Boltzmann Machine algorithm achieved accuracy of 82.02% while the Backpropagation algorithm reached87.01% when using the normalization method. Thus, the diabetes mellitus dataset used can be said to have a better value for the backpropagation algorithm than the restricted Boltzmann machine algorithm.

Keywords: Accuracy, Backpropagation, Diabetes Mellitus, Restricted Boltzmann Machine

1. Introduction

Diabetes is a life-threatening disease. It is nowadays one of the most common diseases, with the number of people with diabetes worldwide increasing[1]. Diabetes is a dangerous disease that develops when the body's ability to create insulin is hampered by the pancreas' inability to meet the body's insulin needs[2]. In addition, other variables, such as an unbalanced diet and an unhealthy lifestyle, contribute to diabetes mellitus. Therefore, to prevent the risk of diabetes mellitus, patients with this disease are advised to keep a healthy diet and lifestyle.

According to the International Diabetes Federation (IDF), the number of people with diabetes continues to rise. One out of every eleven adults is currently estimated to have diabetes, with an estimated 465 million people worldwide suffering from type 2 diabetes. The IDF report estimates that India carries nearly 17 percent [72.9 million] of the global burden of diabetes mellitus.

Type 2 diabetes is the most common case suffered by diabetics. Women are more at risk in this instance because they have a higher likelihood of raising their body mass index or BMI

*Email: dimas.a.anggoro@ums.ac.id

[3]. Diabetes is a complicated disease similar to coronary heart disease and hypertension that can lead to death [4].

The delay in establishing a diagnosis of diabetes is from year to year causing an increase in the number of diabetics. The public is often unaware of many of the signs of diabetes mellitus. As a result, the most appropriate and accurate data mining strategy for diagnosing diabetes mellitus is required as a first step. Data mining methods have long been used to locate or discover patterns in large datasets [5]. For the purpose of this research, the artificial neural network algorithm Restricted Boltzmann Machine (RBM) and Backpropagation are employed.

Susilawati & Muhathir research was conducted to analyze the results of the mean square error (MSE) in the restricted Boltzmann machine algorithm [6]. The best percentage value in this study was 93.42%, with a binary sigmoid activation function and a learning rate of 0.05, and a momentum of 0.7. As a result, the activation function, momentum, and learning rate are all factors that influenced the RBM algorithm's ability to produce a high proportion of values. Subsequent research, entitled Partus referral classification using backpropagation neural network, was conducted by A G Pertiwi. The research compared several algorithmic methods, namely C4.5, Backpropagation Neural Network, and Nave Bayes Classifier. This research found an accuracy rate of 80.91% higher from backpropagation neural network compared to C4.5 89.17% and Naïve Bayes Classifier 65.94% [7].

Previous research has not applied the normalization method to the algorithm's performance process. In this study, adding a method that has not been used by previous research, namely the normalization method, which is a supporting method in the pre-processing process. The normalization method will be very helpful in producing high accuracy values in the restricted Boltzmann machine and backpropagation algorithms. The data that needs to be normalized is data that has a different range [8]. The normalization method used in this study is min-max normalization. This can balance the comparison value between the data before the normalization process. Furthermore, data processing is carried out, namely processing the performance of the two algorithms using the anaconda navigator. The algorithm used is the Restricted Boltzmann Machine (RBM) and Backpropagation algorithm. The backpropagation algorithm is a development of the multilayer perceptron (MLP) algorithm while RBM is an algorithm that has Stochastic properties. Both are artificial neural network algorithms. This research also adds confusion matrix performance.

This paper explores whether the backpropagation algorithm produced a higher accuracy value with the use architecture and parameters, such as the activation function, learning rate. It also discusses whether RBM performance can be improved when the right activation function, momentum, and learning rate are used. The study aims to compare the two algorithms to evaluate which one can provide the highest level of accuracy and is more accurate in detecting diabetes mellitus. The findings of this research are expected to help the community in diagnosing diabetes mellitus by offering the most up-to-date and accurate algorithm.

2. Methodology

The stages in this research will be described in the following flowchart diagram.

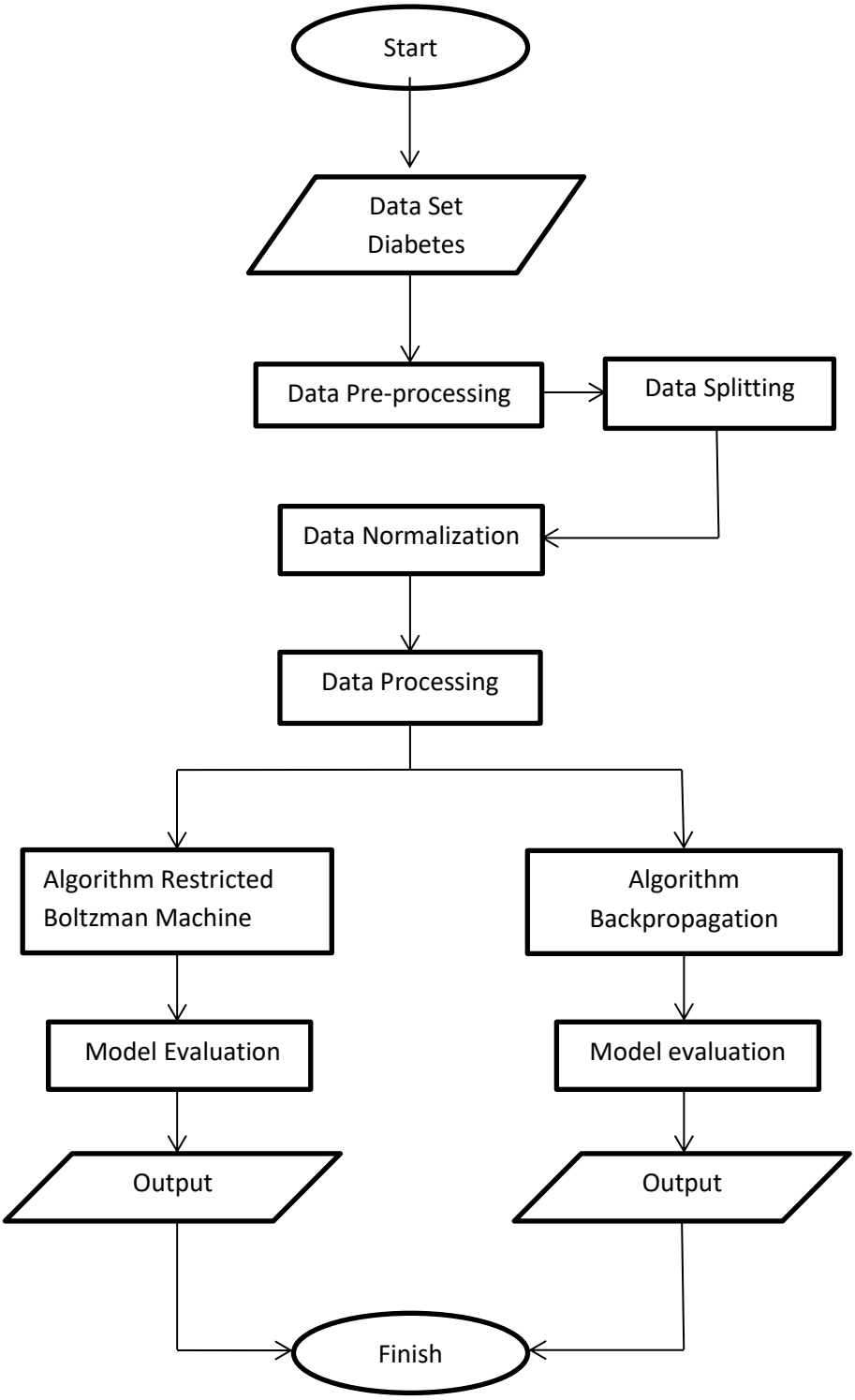

Figure 1: Methodology Stages.

Figure 1 above describes the process of the two algorithms according to the methodology that will be explained in more detail in the next sections. The following is an explanation of the stages of the methodology in the diagram above.

2.1 Data Set

The initial method in the data mining stage was selecting data from a set of data and forming an attribute and variable. The goal was to simplify the data mining process of the research[2]. The dataset used in this research was taken from the Kaggle Dataset that originated from the National Institute of Diabetes and Digestive and Kidney Diseases. The data used focused solely on the female gender because there was a supporting variable for pregnancy. There were 768 data elements in this diabetes dataset. Table 1 shows the attributes and variables for each.

Table 1: Diabetes mellitus data attributes and variables

Variable	Attribute	Description
X1	*Pregnancies*	Number of pregnancies
X2	*Glucose*	Glucose-containing compounds in the patients' body
X3	*Blood Pressure*	Blood pressure of the patients
X4	*Skin thickness*	Skin thickness (mm)
X5	Insulin	The insulin hormone helps the absorption of glucose into the body's cells to control blood sugar.
X6	*BMI*	Body mass index
X7	*DiabetesPedigreeFunction*	Hereditary history of diabetes
X8	*Age*	Age of patients (in year)
Y	*Outcome*	Class variable (0 or 1)

2.2 Data Pre-processing

2.2.1 Data Splitting

In this step was the dataset was split into two parts: training data and testing data. For this research, the dataset was divided into 70% training data and 30% testing data.

2.2.2 Data Normalization

After the splitting stage, the researchers proceeded to the normalization stage. Normalization is part of the pre-processing process. Normalization is the process of transforming data into a specific range. The normalization method used in this research is Min-Max normalization. The following is the equation used in this stage.

$$V' = \frac{(V - MinA)\ X\ (NewMaxA - NewMinA)\ +\ NewMinA}{MaxA - MinA} \qquad (1)$$

Where V' is normalized data; V is preliminary data; MinA is minimum value; MaxA is maximum value; NewMinA is new minimum and NewMaxA is new maximum value.

2.3 Data Processing

2.3.1 Restricted Boltzmann Machine (RBM)

The Restricted Boltzmann Machine algorithm is a two-layer stochastic neural network algorithm. A neural network is a unit of neurons with a binary activation form that depends on the connected neurons, and stochastic meaning that each neuron will react randomly when activated. The visible layer will aid in the observation of the input reconstruction process. Hidden layers are used to capture visible node dependencies that cannot be modelled with visible layer interactions[9]. The hidden layer will result in activation in the form of the forward pass. During the backward pass, the visible layer assists the RBM in reconstructing the input. The array of weights represents the visible units connected to all hidden units.

Therefore, each hidden unit is also connected to all visible units, whereas the bias unit is connected to all visible and hidden units[10].

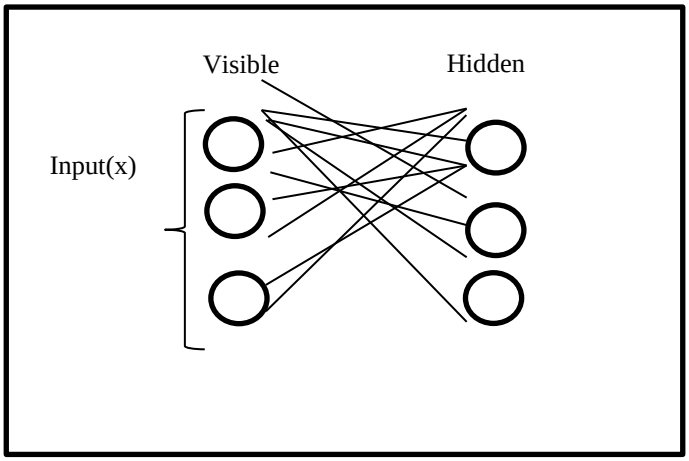

Figure 2: The structure of the RBM algorithm.

Sample data is used as input for the RBM algorithm through visible neurons during training, and then the sample network is reversed between visible and hidden neurons. The purpose is to investigate the connection weights in the visible and hidden layers and the bias in neuron activation so that the RBM algorithm can rebuild the input data from hidden neurons during the visible neurons sample phase.

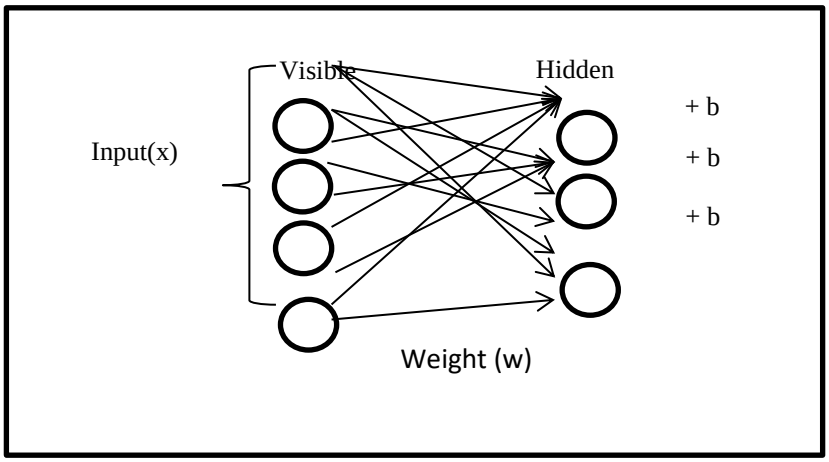

Figure 3: RBM algorithm structure in receiving input.

As shown in the figure above, the input will be multiplied by the weight and then added to the bias. The algorithm or activation function will receive the outcome. The multiplication of the input will be received by the hidden node, which will then be multiplied by the initial weight column before being appended. This process will generate the following equations:

$$H^{(1)} = S(V^{(0)T} W + b_j) \quad (2)$$

Where $H^{(1)}$ and $V^{(0)T}$ are appropriate vectors (matrix column) for *visible* and *hidden* units. S is the iteration superscript iteration, and b_j is the bias of the hidden unit.

$$V^{(1)} = S(H^{(1)} W^T + a_i) \quad (3)$$

$V^{(1)}$ is a vector corresponding to the column matrix for the visible unit, and $H^{(1)}$ is a hidden unit. The superscript is an iteration and a_i the bias vector of the visible unit. There will be a multiplication between the matrices of a set of training samples and the weight matrix during the sampling process, followed by a neuron activation function. The learning rate always controls the visible and hidden layers in this phase to accelerate the learning rate. Each group of data in the training set will be repeated until it reaches convergence[10].

The hidden unit is updated and initialized where the probability of the hidden unit j is expressed using the following equation.

$$P(H_j = 1|\, V) = \sigma\,(b_j + \sum i\ W_{ij}\, V_i) \tag{4}$$

σ(x) is activation function.

$$\sigma(x) = \frac{1}{1 + \ exp^{(-x)}} \tag{5}$$

From the above equation: i (i = 1,2,3……, n) describes the number of visible units and j (j = 1,2, 3,…….n) describes the number of hidden units. b_j is the unit bias for hidden units. W_{ij} is the weight between visible and hidden units, and V_i is the binary state of visible units. Visible units are initialized and updated with the following equation, where V_i of each visible unit i with a value of H (binary state of the hidden unit is set to one with probability).

$$P\ (V_i = 1\ |H) = \sigma\ \left(a_i + \sum j\ W_{ij\ H_j}\right) \tag{6}$$

From the above equation, it is known that a_i is the bias of the visible unit. H_j is the binary state of the hidden unit j, and sigma is the binary sigmoid activation function of equation 5. The followings are the steps of utilizing the Restricted Boltzmann Machine algorithm[10] :

1. Data initialization

a. Carry out the process of searching for weight and bias values with small random values.
b. Determine learning rate and maximum epoch.
c. If epoch < maximum Epoch, do the steps below.
d. If data sample < maximum data sample, do the steps below.

2. Positive phase

During this phase, data and samples will be taken from the hidden unit.

a. Calculate the activation energy, probability, and state of the hidden unit using equation 4.
b. Calculate the positive associative value using equation 7. To do this, multiply the sample data matrix of the transposed visible neurons by the probability, which results from step 2(a).

$$Pos\_Asso = (data)^T\ * P(H_j) \tag{7}$$

From this equation, $(data)^T$ is the visible neuron transposed from the data matrix. P(Hj) is the probability of hidden unit, which results from step 2(a).

3. Negative Phase

There will be the reconstruction of visible units and sample data from hidden units throughout this phase.

a. Calculate the activation energy and probability of visible units using equation 6.
b. Step 2(a) is used to update hidden units.
c. Calculate negative associative using equation 8. In calculating negative associative, transposed data matrix (probability of visible unit obtained from step 3(a)) is multiplied by the probability of hidden unit resulting from step 3(b).

$$Neg\_Asso = (data)^T * P(H_j) \quad (8)$$

From the equation, the $(data)^T$is a data matrix (probability of units obtained from step 3(a)), which is transposed. P(Hj) is the hidden unit probability resulting from step 3(b).

4. Weight update

Weight update is done by using equation 9.

$$W_{ij(new)} = W_{ij(old)} + \Delta W_{ij} \quad (9)$$

$$\Delta W_{ij} = \varepsilon\,(Poss\_Asso - Neg\_Asso) \quad (10)$$

As seen in the above equation, Wij is the weight between the visible unit and hidden unit. ΔW_{ij} s the difference between the old and new weights, as well as the difference in the calculated weight derived for equation 10, which is the weight change calculation. Meanwhile, ε is the learning rate. The value of the change in weight ΔW_{ij} is calculated by multiplying the learning rate by the difference in value between Poss_Asso and Nega_Asso.

If the desired value (convergence) has been achieved by obtaining good weights during the process, the RBM algorithm will stop learning. When the cycle of learning patterns occurs, each weight changes, and it is expected to obtain an accurate value with a stable weight [6].

2.3.2 Backpropagation

The Backpropagation algorithm is a supervised learning algorithm. The backpropagation algorithm uses a perceptron that has many layers to change the weights connected to the neurons in the hidden layer[11].The backpropagation algorithm is the development of the multilayer perceptron (MLP) algorithm[12]. Backpropagation is a simple and clear iterative algorithm that usually works well even with complex data[13]. The backpropagation algorithm also eliminates the problems associated with learning rules that are similar to spike-timing-dependent plasticity[14]. There are several parts of the Backpropagation algorithm architecture called input layers, hidden layers, and outputs.

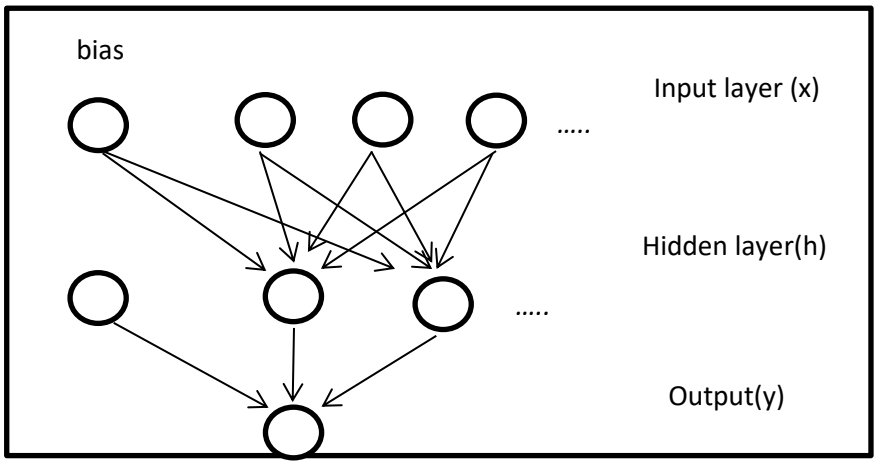

Figure 4: The structure of the Backpropagation algorithm.

Figure 4 illustrates the backpropagation algorithm architecture, which has 3 layer nodes in the form of the input layer, 2 layer nodes in the form of hidden and 1 output layer, and 2 bias. One bias towards the hidden layer and the other towards the output layer.

There are three layers in backpropagation, namely:

1. The input layer contains a value that will be passed on to the hidden layer.
2. Hidden layer. This layer takes input from the input layer and passes it on to the output layer.
3. Output. The output will be the data processing results.

The Backpropagation algorithm solves the problem in numerous steps, each of which minimizes the error value in the algorithm's output. The stages of backpropagation are listed below[12] :

1. Feedforward stage

a. Initialize the initial weight with a small random value.

b. Distribute each input unit.

Each unit in the hidden layer H_j (j= 1,2,3…. , p) is multiplied by the input layer. Then, the wight is summed up.

$$H_{inj} = V_{oj} + \sum_{i=1}^{n} Xi\ W_{ij} \quad (11)$$

Where H_{inj} is the value to calculate hidden layer. V_{oj} is bias. X_i is the value of the input layer in the hidden layer. For units that have entered a value, the activation function is used to calculate the output value and propagate the signal to all units in the output. The following is the equation of the activation function.

$$H_j = \frac{1}{1+e^{-Hinj}} \quad (12)$$

c. Each output unit (Y_k, k= 1, … m) will add a weighted input signal.

$$Y_{ink} = W_{ko} + \sum_{j=1}^{p} H_j\ W_{jk} \quad (13)$$

Where Y_{ink} is the input value of k unit. W_{ko} is the weight of bias. H_j is the activation value of H_j . W_{jk} is the connection value from H_{inj} to Y_k . The activation function is used to calculate the output signal:

$$Y_k = \frac{1}{1+e^{-yink}} \quad (14)$$

2. Backpropagation stage

a. Each output unit ($Y_k, k = 1,2,3 \ldots . m$) receives a pattern related to the learning input. The following is the equation to calculate error information.

$$\delta_k = (t_k - Y_k) f'(Y_{ink}) = (t_1 - y_k)\, y_k\, (1 - y_k) \quad (15)$$

Where δ_k is the setting factor of the connection in the outer layer. t_k is target data. Y_k is the k-th unit of the outer layer. Y_{ink} is the input value of k-unit.

δ_k is the error unit that will be used to change the weight of the layer below it.

b. Calculate the change in weight that will be used in improving the value of W_{jk} with the acceleration of α.

$$\Delta W_{jk} = \alpha\, \delta_k\, H_j \quad (16)$$

Where ΔW_{jk} is the result of the difference between $W_{jk}\{t\}$ and $W_{jk}(t+1)$, α is learning rate $0 < \alpha < 1$. δ_k is the connection weighting factor in the output layer, and H_j is the activation value of unit H_j.

c. Calculate the value of bias error and distribute the value to the layer below it.

$$\Delta W_{ok} = \alpha\, \delta_k \quad (17)$$

d. Calculate the value of the error of the hidden layer using the following equation.

$$\delta_{inj} = \int_{k=1}^{m} \delta_k\, W_{kj} \quad (18)$$

Where δ_{inj} is the value of error calculation in the hidden layer. δ_k is the connection weight setting factor in the output layer, and W_{kj} is the weighted connection value from H_j to Y_k.

e. Calculate error information from the hidden layer using the derivative activation function.

$$\delta_j = \delta_{inj}\, f'(H_{inj}) = \delta_{inj}\,(1 - H_j) \quad (19)$$

Where δ_{inj} is the value to calculate the error in the hidden layer.

f. Then calculate the weight error that will be used to correct the value of V_{ij}.

$$\Delta V_{ij} = \alpha\, \delta_j\, X_i \quad (20)$$

g. Continue by correcting the bias of V_{oj} .

$$\Delta V_{oj} = \alpha \delta_j \quad (21)$$

3. Stage of weight change

a. Each output unit (Y_k, k = 1,2,3, ……m) correct the bias and its weight (j= 0,1,2,3, …..,p) by adding up old weight and total weight.

$$W_{jk(new)} = W_{jk(old)} + \Delta W_{jk} \quad (22)$$

Where W_{jk} is the weighted connection value from H_{ij} to Y_k. ΔW_{jk} is the difference between W_{jk} (t) and W_{jk} (t+1).

b. Then calculate the change in the weight of the line leading to the hidden layer.

$$V_{ij(new)} = V_{ij(old)} + \Delta V_{ij} \quad (23)$$

Where V_{ij} is the connection value from unit X_i to unit H_i. ΔV_{ij} is the difference between V_{ij}(t) and V_{ij} (t+1).

The training process will experience termination conditions if the iteration is greater than the maximum iteration that has been determined. Iteration is a series of steps in learning artificial neural networks. One is interpreted as one-time learning carried out in the feedforward step until the weight changes.

2.4 Model Evaluation

The purpose of the evaluation is to determine the best way to partition the dataset to provide a classification with a high level of accuracy. In this research, a confusion matrix was employed to evaluate accuracy. The confusion matrix is an evaluation that is usually used in supervised learning algorithms. Confusion matrices are made up of actual and predicted information about the classification outcomes. The benefit of using a confusion matrix is that it can determine the performance of an algorithm.

Accuracy

To provide the best algorithm comparison and determine the proportion of data that is modelled correctly, both algorithms must have the same measurement. Therefore, accuracy is used to find the correct total data [15]. The following is the equation to calculate the accuracy:

$$Accuracy = \frac{(TP+TN)}{(TP+FP+FN+TN)} \quad (24)$$

Where TP is True Positive; TN is True Negative; FP is False Positive and FN is False Negative.

Precision

Precision explains how many positive correct predictions are compared to the overall positive predicted outcome. The following is the equation of the precision formula:

$$Precision = \frac{TP}{TP+FP} \quad (25)$$

TP: True Positif and FP : False Positif

Recall

Recall serves to predict how many positive predictions are true compared to the overall positive predicted results. The following is the equation for the recalled formula:

$$Recall = \frac{TP}{TP+FN} \quad (26)$$

TP: True Positif, FP : False Positif and FN : False Negative.

F1 Score

Used for comparison of average precision and recall. The following is the equation for the F1 score formula:

$$F1\ Score = 2\ \frac{Precission\ x\ Recall}{Precision + Recall} \quad (27)$$

3. Results

The dataset used in this research is a diabetes mellitus dataset. The total number of data entries was 768, with eight input variables and one output variable. The data was collected from Pima Indian female patients who were at least 21 years old. The dataset was divided into two parts for testing at the beginning of the research process, with 70% training data (537 entry) and 30% testing data (231 entry). Training data was a data model generated by a dataset. Furthermore, the testing data was used to store the correct values separately obtained from deleting the training data values [16].

The data was then processed using the python programming language utilizing the anaconda navigator application. In this stage, the number of patients with diabetes was 34.90%, and those who did not have diabetes was 65.10%. After separating the data into two parts, the next stage was data normalization. The advantage of using this method is its simplicity and its ability to modify the original data in its entirety [17]

The Min-Max normalization method was employed in this research. The advantage of utilizing min-max normalization is that the tested data with an unknown class can be categorized correctly and accurately using this method[18]. The normalization method

performs well in terms of generating accuracy values and is quick to produce optimal results. In addition, data within a certain range can be scaled from one range to another using the min-max normalization method[19]. The next stage is data processing, where the datasets and algorithms are entered into the anaconda navigator application. The algorithms used are the Restricted Boltzmann Machine (RBM) and Backpropagation.

3.1 Restricted Boltzmann Machine (RBM)

Before initiating the testing process, it is critical to create a Restricted Boltzmann Machine Algorithm architecture, which was demonstrated in the Figure 3. The architecture used consists of a visible layer according to the number of dataset features and 256 neurons in the hidden layer. The following are the results of the evaluation process for the confusion matrix model of the RBM algorithm.

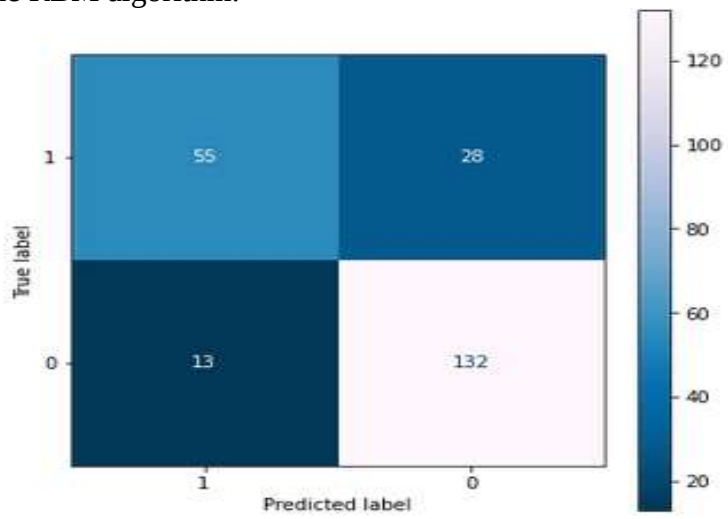

Figure 5: confusion matrix RBM algorithm.

Figure 5 is the confusion matrix of the RBM algorithm that contains the actual and predicted values for the results of classification.

Table 2: Accuracy value of RBM

Learning Rate	Momentum	*Accuracy*	
		Without normalized	**Normalized**
0.01	0.3	63.60 %	82.02 %
0.03	0.3	63.60 %	81.14 %
0.05	0.3	63.60 %	79.82 %

Table 2 shows that the RBM accuracy value using normalization is greater than the value without normalization, at 82.02 % and 63.60 %, respectively. The result of the normalization process gets a higher value. During the normalization process, there will be a spread of data. so that it can produce a high accuracy value compared to those who do not use the normalization method. The anaconda application was used to calculate these results, with PyTorch assisting in the process. PyTorch is quite useful for quickly calculating data. PyTorch and TensorFlow are both open-source libraries that are nearly identical. However, PyTorch takes advantage of dynamic computation, which makes it more complex and flexible.

During the training process, the dataset needs to be divided into several batches. Because there are 768 datasets in the diabetes mellitus dataset, the batch is effective for speeding up the training process. For each batch, the number of 768 training data is divided by 72 data

batches. As a result, 768 datasets are separated into 72 batches to be processed in each epoch. The weights and biases are then initialized with random values, and the calculation procedure begins. The diabetes mellitus dataset sample is then entered. The RBM algorithm will process the visible layer to the hidden layer when the process occurs (positive phase). Then it was calculated and categorized. To speed up the learning process, The learning rate and momentum parameters are added and the weight value is altered. In the performance process of the RBM algorithm, the activation function used is sigmoid. the sigmoid activation function can help speed up the performance of the RBM algorithm and produce stable values.
In the RBM algorithm, the learning rate value has a significant impact on the training process. The effects will be maximum if the learning rate is lower. If the learning rate is too large, there will be a very significant increase in the weight value that can cause errors during the performance process. In the restricted algorithm Boltzmann machine learning rate parameter has a function that is to accelerate the learning rate. Therefore, at this time there is no specific way to determine the exact value in the parameters of the RBM algorithm. For example, table 2 shows that a learning rate of 0.01 produces a result that is 82.02 % than a learning rate of 0.05, which produces 79.82 %. Furthermore, compared to results obtained without employing the normalizing process, the outcomes obtained utilizing the normalization process are higher and more optimal. Hence, it is concluded that the size of the learning rate and the normalization procedure in the Restricted Boltzmann Machine (RBM) algorithm have a significant influence.

Table 3: precision, recall, and F1-score values RBM algorithm

Precision	0.85
Recall	0.77
F1-score	0.81

The table above shows the results of precision, recall, and F1-score of the RBM algorithm with the highest accuracy value of 82.02%. From the table, it can be seen that the precision value is higher than the recall value and F1-score.

3.2 Backpropagation

Before testing, the Backpropagation algorithm, it is very important to build a network architecture. The backpropagation algorithm architecture demonstrated in Figure 4. The following are the results of the evaluation process for the confusion matrix model of the RBM algorithm.

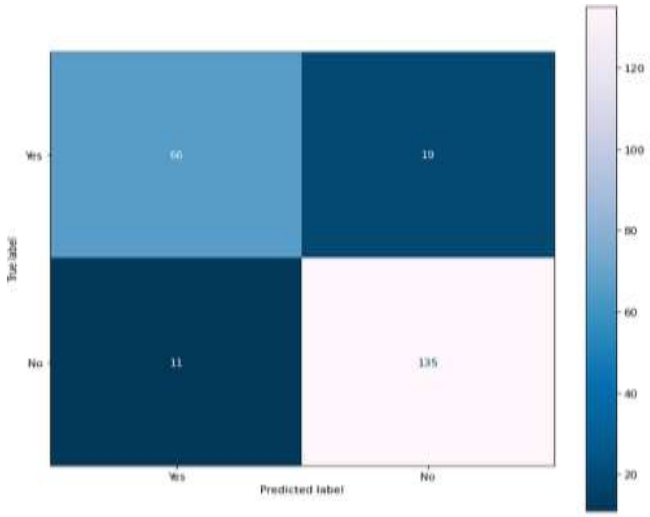

Figure 6: Confusion Matrix algorithm Backpropagation.

Figure 6 is the confusion matrix of the backpropagation algorithm that contains the actual and predicted values for the results of classification.

Table 4: Accuracy value of Backpropagation

Neuron, *Hidden layer*	*Learning Rate*					
	Learning Rate (0.01)		**Learning Rate (0.02)**		**Learning Rate (0.03)**	
	Without Normalized	**Normalized**	**Without Normalized**	**Normalized**	**Without Normalized**	**Normalized**
(10, 1)	63.20%	82.68%	63.20%	82.68%	63.60%	82.68%
(10, 2)	79.22%	83.55%	78.79%	81.39%	81.39%	84.42%
(10, 3)	63.20%	85.71%	63.20%	84.85%	63.20%	86.15%
(10, 4)	63.20%	86.15%	63.20%	86.15%	63.20%	86.15%
(10, 5)	76.19%	82.68%	76.19%	83.98%	63.20%	85.28%
(10, 6)	76.19%	85.71%	77.92%	86.58%	63.20%	85.71%
(10, 7)	83.98%	86.15%	85.28%	84.85%	81.82%	87.01%
(10, 8)	87.01%	85.28%	76.62%	85.71%	76.62%	82.25%
(10, 9)	74.89%	86.15%	82.68%	86.58%	82.68%	85.71%
(10, 10)	75.32%	83.55%	76.19%	84.85%	63.20%	83.55%

Table 4 shows the results of the process, which began with the initialization of the initial weight with a random value. Then, utilizing the activation function of all layers, continue the feedforward process by activating the neurons. In this work, the ReLU was chosen as the activation function. The ReLU activation function is a simple activation function that can decrease data processing time when encountering a high number of neurons. The categorization value obtained by the ReLU activation function will range from 0 to 1[2]. Following that, it enters the backpropagation process, where weight optimization takes place, and the weight update is repeated in each iteration to achieve the best result. The process will stop when the iteration has reached the optimal weight and bias results.

The normalizing method generates the highest and most accurate value in the 7th layer, with an accuracy value of 87.01 % to 81.82 % for those who do not utilize the normalization method. The normalizing process in the backpropagation algorithm is better than the normalized RBM, algorithm backpropagation that resulted in an accuracy of 87.01 % at the 7^{th} layer with a learning rate of 0.03, which is higher than algorithm RBM which results 82.02 %, after being normalized.

It demonstrates that the normalizing technique yielded high and accurate results. This happens because the normalization process is used to equalize the scale of each data feature in a smaller range. Data sharing is included in normalization, resulting in a very high and accurate accuracy value when compared to those who do not employ the normalizing approach[2].

In the backpropagation algorithm, the learning rate value has no definite rules, different learning rate values are offered in this research that are used to obtain the best accuracy value. The training process will be accelerated if the learning rate is high. However, if the learning

rate is too high, the value will become unstable, and the accuracy value can also drop dramatically and fluctuate unpredictably[20].

Also, in this research, the number hidden layer (architecture network) has a significant influence, so it is very helpful in accelerating the time during the performance process algorithm. The larger the network architecture, however, the more complicated it will become. Since the hidden layer contains nodes (neurons) that can help speed up the backpropagation algorithm's performance. Nodes in this algorithm will be connected from one node to another. No method can find the optimal number of neurons in the hidden layer. Table 4 shows that a number of neurons as many as 10 combined with the 7th hidden layer can obtain a maximum accuracy value of 87.01% with a short time. Architecture network, parameters and normalization process are very helpful to produce high accuracy values in the backpropagation algorithm process.

Table 5: values precision, recall, and F1-score Backpropagation algorithm.

Precision	**0.80**
Recall	0.66
F1-score	0.79

The table above shows the results of precision, recall, and F1-score of the backpropagation algorithm with the highest accuracy value of 87.01%. From the table, it can be seen that the precision value is higher than the recall value and F1-score.

4. Conclusion

The results of this research, which compares Restricted Boltzmann Machine (RBM) and Backpropagation algorithms, found that the accuracy values of each algorithm using the normalization method were 82.02% and 87.01 %, respectively. When the two algorithms are compared, the Backpropagation algorithm has the highest accuracy, with an accuracy value of 87.01 % when utilizing the normalization method. Thus, both algorithms benefit from the normalizing method in terms of providing high and accurate values. Based on the research findings, it can be concluded that the Backpropagation algorithm has a high and accurate value of accuracy, precision, recall, and F1-score in diagnosing diabetes mellitus with a value of 87.01 %, 0.80, 0.66, and 0.79.

There are still some flaws in this research. Thus, it is required to enhance and improve in all areas. The two algorithms have not been studied using the normalization method for comparison in the case of diabetes disease. This research was conducted to develop methods and algorithms that were not previously used in detecting disease. For this reason, It is envisaged that future research will be able to develop this research better and focus more on the architecture and parameters of each algorithm to create higher and better accuracy values.

References

[1] S. Kang, "Personalized prediction of drug efficacy for diabetes treatment via patient-level sequential modeling with neural networks," *Artif. Intell. Med.*, vol. 85, pp. 1–6, 2018, doi: 10.1016/j.artmed.2018.02.004.

[2] D. A. Anggoro and D. Novitaningrum, "Comparison of accuracy level of support vector machine (SVM) and artificial neural network (ANN) algorithms in predicting diabetes mellitus disease," *ICIC Express Lett.*, vol. 15, no. 1, pp. 9–18, 2021, doi: 10.24507/icicel.15.01.9.

[3] H. Bhatt, S. Saklani, and K. Upadhayay, "Anti-oxidant and anti-diabetic activities of ethanolic extract of Primula Denticulata Flowers," *Indones. J. Pharm.*, vol. 27, no. 2, pp. 74–79, 2016, doi:

10.14499/indonesianjpharm27iss2pp74.

[4] H. Fatemidokht and M. K. Rafsanjani, "Development of a hybrid neuro-fuzzy system as a diagnostic tool for Type 2 Diabetes Mellitus," *2018 6th Iran. Jt. Congr. Fuzzy Intell. Syst. CFIS 2018*, vol. 2018-Janua, pp. 54–56, 2018, doi: 10.1109/CFIS.2018.8336627.

[5] M. Thiyagarajan, C. Raveendra, P. Thulasi, and S. K. Priya, "Role of association rules in medical examination records of Gestational Diabetes Mellitus," *Proceeding - IEEE Int. Conf. Comput. Commun. Autom. ICCCA 2017*, vol. 2017-Janua, pp. 78–80, 2017, doi: 10.1109/ CCAA. 2017.8229775.

[6] S. Susilawati and M. Muhathir, "Analisis Pengaruh Fungsi Aktivasi, Learning Rate Dan Momentum Dalam Menentukan Mean Square Error (MSE) Pada Jaringan Saraf Restricted Boltzmann Machines (RBM)," *J. Informatics Telecommun. Eng.*, vol. 2, no. 2, p. 77, 2019, doi: 10.31289/jite.v2i2.2162.

[7] A. G. Pertiwi, A. P. Wibawa, and U. Pujianto, "Partus referral classification using backpropagation neural network," *J. Phys. Conf. Ser.*, vol. 1193, no. 1, 2019, doi: 10.1088/1742-6596/1193/1/012010.

[8] D. A. Nasution, H. H. Khotimah, and N. Chamidah, "Perbandingan Normalisasi Data untuk Klasifikasi Wine Menggunakan Algoritma K-NN," *Comput. Eng. Sci. Syst. J.*, vol. 4, no. 1, p. 78, 2019, doi: 10.24114/cess.v4i1.11458.

[9] T. Aldwairi, D. Perera, and M. A. Novotny, "An evaluation of the performance of Restricted Boltzmann Machines as a model for anomaly network intrusion detection," *Comput. Networks*, vol. 144, pp. 111–119, 2018, doi: 10.1016/j.comnet.2018.07.025.

[10] S. Nasional, T. Informatika, U. M. Area, I. L. Belakang, and S. Deshmukh, "Algoritma Restricted Boltzmann Machines (RBM) untuk Pengenalan Tulisan Tangan Angka," pp. 140–148, 2017.

[11] M. D. Yalidhan, "Implementasi Algoritma Backpropagation Untuk Memprediksi Kelulusan Mahasiswa," *Klik - Kumpul. J. Ilmu Komput.*, vol. 5, no. 2, p. 169, 2018, doi: 10.20527 /klik.v5i2.152.

[12] Rudianto, "Penentuan Penyakit Peradangan Hati Dengan Menggunakan Neural Network Backpropagation," *Indones. J. Comput. Inf. Technol. Vol 1 No 1*, vol. 1, no. 1, pp. 27–33, 2016.

[13] M. Adi *et al.*, "Penerapan Algoritma Backpropagation Dalam Memprediksi Produksi Tanaman Padi Sawah Menurut Kabupaten/Kota Di Sumatera Utara," vol. 4, no. 1, pp. 77–86, 2018.

[14] A. Wanto and A. P. Windarto, "Analisis Prediksi Indeks Harga Konsumen Berdasarkan Kelompok Kesehatan Dengan Menggunakan Metode Backpropagation," *J. Penelit. Tek. Inform. Sink.*, vol. 2, no. 2, pp. 37–43, 2017.

[15] A. Joshi, E. J. Dangra, and M. K. Rawat, "A Decision Tree Based Classification Technique for Accurate Heart Disease Classification & Prediction," *nternational J. Technol. Res. Manag. ISSN*, vol. 3, no. 11, pp. 1–4, 2016.

[16] D. A. Anggoro and W. Supriyanti, "Improving accuracy Bb applying Z-score normalization in linear regression and polynomial regression model for real estate data," *Int. J. Emerg. Trends Eng. Res.*, vol. 7, no. 11, pp. 549–555, 2019, doi: 10.30534/ijeter/2019/247112019.

[17] U. Rofiqoh, R. S. Perdana, and M. A. Fauzi, "Analisis Sentimen Tingkat Kepuasan Pengguna Penyedia Layanan Telekomunikasi Seluler Indonesia Pada Twitter Dengan Metode Support Vector Machine dan Lexion Based Feature," *J. Pengemb. Teknol. Inf. dan Ilmu Komput. Univ. Brawijaya*, vol. 1, no. 12, pp. 1725–1732, 2017.

[18] R. Sistem, "JURNAL RESTI Peningkatan Hasil Klasifikasi pada Algoritma Random Forest untuk," vol. 1, no. 10, pp. 114–122, 2021.

[19] N. Chamidah, . W., and U. Salamah, "Pengaruh Normalisasi Data pada Jaringan Syaraf Tiruan Backpropagasi Gradient Descent Adaptive Gain (BPGDAG) untuk Klasifikasi," *J. Teknol. Inf. ITSmart*, vol. 1, no. 1, p. 28, 2016, doi: 10.20961/its.v1i1.582.

[20] A. Rahmadhiat, I. I. Tritoasmoro, and I. Wijayanto, "Analisis Penggunaan Algoritma Genetika Untuk Meningkatkan Performansi Dari Klasifikasi Genre Musik Berbasis Jaringan Syaraf Tiruan Back-Propagation," vol. 4, no. 4, pp. 1527–1535, 2016.