The number of layers in DBN can be increased greedily [26]. Each new layer that is stacked on top of the DBN will generate an output of the previous layer and extract the top-level dependencies between the original input variables. As a result, it would improve the network's capability to catch the underlying regularities in the data. We could also create a classification model by inserting an additional layer to the unsupervised pre-training DBN, and the network that is generated will then be fine-tuned. The Backpropagation algorithm can be used in the fine-tuning phase, where each weight value is changed to give the best results [27, 28].

The training phase of this network is as follows:

1) Set the visible layer values with the values of the binarized features.

2) Do pre-training for each RBM with a greedy learning algorithm [29] to find the hidden states value and positive association. This phase is called the positive phase, which uses the formula shown below.

- Calculate the hidden probability.

$$p(h_j = 1 \mid V) = \sigma(b + \textstyle\sum_i v_i w_{ij}) \qquad (1)$$

where:
$p(h_j = 1 \mid V)$ is the hidden probability.
i is the index that denotes the input unit
j is the index that denotes the hidden unit
b is the bias.
v_i is the input value
w_j is the weight value.

- Calculate the hidden states. If the hidden probability is greater than the random value (from 0-1) then the hidden state value is 1; else, it is 0.
- Calculate the positive association value by doing the dot product between the input data and the hidden states.

$$P\,a = \textstyle\sum_i v_i HS_i \qquad (2)$$

where:
$P\,a$ is the positive association.
HS_i is the hidden states.
i is the index that denotes the visible unit
v_i is the visible units.

3) Reconstruct the visible layer by doing the negative phase, which is similar to the positive phase.

- Calculate visible probability by using the same calculation as a hidden probability. The difference is that this calculation uses the hidden states instead of the input value.
- Calculate the visible states by using random values to determine the state.
- Calculate the hidden probability using the same calculation as that in the positive phase, except that the visible states are used instead of the input value.
- Calculate the hidden states by using a random value to determine the state.
- Calculate the negative association using a similar calculation to the positive association, except that the newest hidden states are now used.

4) Update the weight values using this formula.

$$w_{ij} = w_{ij} + l\big(p(h_j = 1 \mid V) - p(v_i = 1 \mid H)\big) \qquad (3)$$

where l is a learning rate that we multiply with the positive and negative values and then added to the initial weight value.

5) Repeat all the processes until the threshold value is obtained.

6) Each layer takes the output of the previous layer as the input for the next layer. For instance, we use the first hidden layer values as the input for the second hidden layer, and so on.

Transpose the weight for each next RBM. This step ends up providing weights for each layer, and the weights are dedicated to the output cluster. Meanwhile, the output clusters are different from the targeted output. The targeted output is a variable that concludes whether a record is categorized as a diabetic case or not.

3) Neat for Solving Regression Problem

N NeuroEvolution (NE) is a method that develops the ANN algorithms with Genetic algorithms [10]. A topology must first be chosen in the traditional NE approach before an experiment can be carried out. A network topology usually has one hidden layer where each neuron is connected to each input and output node. This is to optimize the connection weights, which determine the network's functionalities. Topology and the network structure can affect the effectiveness of learning.

NEAT has been used for various applications, especially for regression problems. It has been noted that NEAT performs better in distressed financial case detection than AdaBoost algorithm and Discriminant Analysis [30]. Some research using NEAT has been done to solve everyday problems such as health, digital games, and modeling. In particular, NEAT was applied to diagnose breast cancer [31]. This research compared the NEAT algorithm with Backpropagation Neural Network (BNN). The result showed different accuracy mean and standard deviation of the two algorithms. NEAT had an accuracy mean and standard deviation of 98.5 ± 0.004, while BNN 95.3±1.425. Another research used the NEAT algorithm for agent-based evacuation simulation to study the pedestrian dynamics and learning process [32]. NEAT identified the correct fitness function, allowing the agents to learn the simulated environment independently. The research showed that the agents managed to complete their tasks successfully.

In developing ANN, NEAT employs an effective idea, i.e., starting off with a minimal, simple topology, which evolves to be more complex in the following generations like organisms in the nature [33]. The growing complexity of the NEAT structures strengthens the analogy between genetic algorithms and natural evolution. In the evolution process of an ANN topology, the complexity is added when it is needed. The evolution of the topology and weight of an ANN network can improve the performance of neuroevolution [10]. Thus, ANN in NEAT works fast. The procedures in NEAT algorithm are similar to that of a genetic algorithm but with the adaptation of the ANN network structure.

1) Initialization of the Architecture: The network's architecture is initialized by determining the number of nodes needed in the hidden layer and the weight value. The weight value is determined by randomly attempting every possibility without any learning phase. The fitness value in each solution is acquired by training the architecture, then calculating the accuracy rate of the architecture.

2) Crossover of the Architecture: In the crossover process, individuals are chosen to become parents by the roulette wheel selection. Genes with the same innovation number can be processed immediately. However, if one parent has more genes than the other, the extra gene will be carried by the child. The crossover process will select the edge weight of one parent to be set on the child's architecture which is the copy of the other parent's architecture.

3) Mutation: Mutation in the NEAT architecture can be carried out in two ways: either adding the nodes or adding the connection. Mutation by adding the connection is done by adding a connection (a line) between two nodes in different layers. Mutation by adding a node, on the other hand, is done by searching a connection that connects two nodes. Based on that connection, it adds or subtracts one new element in the array node containing one new node and two new nodes.

4) Calculating the Fitness Value in NEAT: The calculation of the fitness value in each solution, which in this case is the architecture, is done by training and evaluating using the multi-layer perceptron (MLP) algorithm. The accuracy value obtained in the architecture will then be used as the fitness value. On the MLP calculation, the threshold binary equation used is shown below.

$$y = \sum_{i=0}^{n} x_i w_i + b \quad (4)$$

where:
i is the incremental variable to denote the input nodes
$x_i$is the diabetes variable used
w_i is the weight value of x_i that is randomly initialized
b is the bias.
If there is a difference with the target value, then the next weight value will be evaluated using these equations.

$$\Delta w = x_i + y \quad (5)$$

$$\Delta b = b + \alpha t \quad (6)$$

where:
α is the learning rate.

After the training phase, the evaluation process is conducted for every node. Every output value (y) that corresponds to the target value will become the fitness value of the ANN architecture.

III. Results and Discussion

A. Neat Accuracy result on Dataset

To examine the consistency of the accuracy rate, this research tested the algorithm 5 times. Each test was done by generating 10 individuals with 50 maximum epochs and 50 iterations for the architecture evolution. The graphs in Fig. 3 show the test results where the x-axis is the number of iterations, and the y-axis is the best fitness value of each iteration. The graphs show 3 examples of developing graphs from 10 trial results. The 0th iteration was the process of assigning a fitness value that preceded the architecture evolution. There was a drastic increase in the initial state of all the five graphs, which then continued with a gradual increase until it reached a convergence point. For example, the best fitness value in the 0th iteration of the first graph in Figure 3 was 66.66%, which then increased in the 1st iteration to 68.91%. The fitness value became 68.92% in the 2nd iteration and kept increasing until the 12th iteration. The graph remained constant from the 13th iteration to the 42nd iteration with a fitness value of 70.03%. Finally, in the 43rd until 50th iteration, the fitness value became 70.41%.

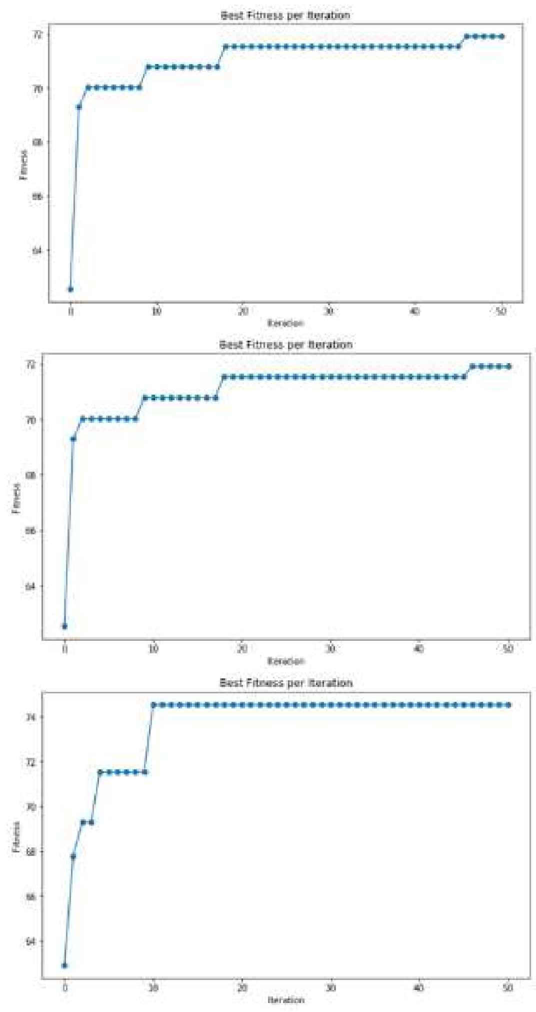

Fig. 3 Best fitness value graphs

TABLE I
BEST FITNESS VALUE PER TESTING

Testing	*Fitness* (%)
1	70.41
2	71.91
3	74.53
4	73.40
5	72.28
6	71.161
7	71.161
8	71.91
9	71.91
10	71.16
Avg	71.9832

Based on the 10 tests that were carried out, the highest fitness value was 74.53%, the lowest value was 70.41%, and the average value was 71.9832% as seen in Table 1.

B. *DBN Result on Diabetes Dataset*

The testing process was divided into two sections: the error evaluation process and the accuracy evaluation process using the backpropagation evaluation function. The error results in the training part of DBN for each layer that represented the output were: 0.3753481, 0.0012443, 0.0031215, 6.21e-05, 2.05e-05, 0.0047576, and 0.0004884, as shown by the graph in Fig 4.4.

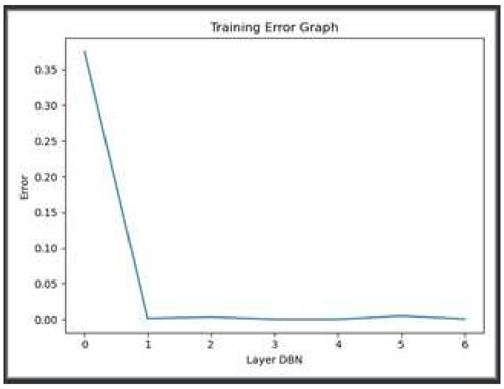

Fig. 4 DBN error rate graph

After the training process, the evaluation phase was then carried out, which is an accuracy evaluation from evaluating function in the Backpropagation algorithm. Based on the evaluation result, a comparison between random and 0 weight values is provided. Table II shows twenty prediction accuracy experiments using the Backpropagation Evaluation function.

TABLE II
ACCURACY RESULTS PER EVALUATION

Test	Accuracy with DBN-initialized weight values	Accuracy with 0 weight values	Accuracy with random weight values
1	18.994 %	0.0 %	0.0 %
2	25.698 %	0.0 %	0.0 %
3	18.063 %	0.0 %	0.0 %
4	17.877 %	0.0 %	0.0 %
5	22.905 %	0.0 %	0.0 %
6	28.678 %	0.0 %	0.0 %
7	17.877 %	0.0 %	0.0 %
8	26.071 %	0.0 %	0.0 %
9	19.367 %	0.0 %	0.0 %
10	19.367 %	0.0 %	0.0 %
11	20.484 %	0.0 %	0.0 %
12	28.119 %	0.0 %	0.0 %
13	18.063 %	0.0 %	0.0 %
14	34.823 %	0.0 %	0.0 %
15	20.298 %	0.0 %	0.0 %
16	25.698 %	0.0 %	0.0 %
17	27.002 %	0.0 %	0.0 %
18	19.367 %	0.0 %	0.0 %
19	21.601 %	0.0 %	0.0 %
20	21.974 %	0.0 %	0.0 %
Avg	22.616 %	0.0 %	0.0 %

This table shows the results or comparison when using the weight from the DBN result, a random weight, and 0 weight as the initial weight using 4 classes as output. As shown in the table, the random and 0 weight values did not provide accurate values. Meanwhile, implementing DBN from the stacked RBM provided the highest accuracy value, i.e., 34.8%. This accuracy value was quite low because DBN algorithm was used for clustering purposes, where the number of clusters was the same as the number of target solutions. Despite the low accuracy value, the weight values generated greatly could be used as the initial weights for the Backpropagation algorithm.

IV. CONCLUSION

Through the design, implementation, and testing of DBN and NEAT, we could compare the roles of both algorithms in detecting diabetes. This research used 8 variables to detect diabetes, namely the number of pregnancies, plasma glucose concentration, diastolic blood pressure, triceps skinfold thickness, insulin serum, body mass index, diabetes pedigree function, and age.

The DBN Algorithm could not be utilized directly to detect diabetes due to its unsupervised behavior, and based on the experiment, its highest accuracy was only 34.823%. However, this algorithm can be converted into a supervised case to improve its accuracy by continuing its computation using a backpropagation algorithm. This highest accuracy will be helpful to be used as the initial weight value and performs better against a random or 0 value weight which only resulted in 0% accuracy when tested using the backpropagation evaluation function in the initial state.

On the other hand, The Neat algorithm could be used directly to detect diabetes by exploring the ANN architecture through genetics algorithm and by providing a fitness calculation with multi-layer perceptron function. The crossover and mutation processes in Genetics Algorithm cause the ANN architecture to evolve, and the multi-layer perceptron algorithm will fill the weight of ANN. The experiment was performed 10 times and resulted in an average accuracy of 72.506% and a maximum accuracy of 74.53%.

This research could be further developed concerning both algorithms. An investigation on DBN could be focused on implementing multi-layer backpropagation and utilizing the initial weight provided by DBN. Future research on NEAT could focus on developing the algorithm for determining the fitness score. Backpropagation could substitute the multi-layer perceptron because it has a backward process to evaluate the weight, leading to higher accuracy. Due to the importance of network topology on the fitness score, the crossover strategy also needs to be investigated.

REFERENCES

[1] International Diabetes Federation. *IDF Diabetes Atlas.* Brussels: International Diabetes Federation. 2019

[2] T. Ligita,K Wicking, K Francis, et al. How people living with diabetes in Indonesia learn about their disease: A grounded theory study. *PLOS ONE*, p.1-9, 2009

[3] Ni, X., Wang, H., Che, C., Hong, J. and Sun, Z.. Civil aviation safety evaluation based on deep belief network and principal component analysis. *Safety science*, *112*, pp.90-95, 2019

[4] La Cava, W. and Moore, J.H.. Learning feature spaces for regression with genetic programming. *Genetic Programming and Evolvable Machines*, p.1-35, 2020

[5] Webb, G. I., Fürnkranz, J., Hinton, G., Sammut, C., and Patrick, J. Density-Based Clustering. *Encyclopedia of Machine Learning*, p.270–273. 2011

[6] Zhang, Y. and Huang, G. Traffic flow prediction model based on deep belief network and genetic algorithm. *IET Intelligent Transport Systems*, *12*(6), pp.533-541,2018

[7] Alom, Z., Bontupalli, V., & Taha, Tarek M.. Intrusion Detection using Deep Belief Networks. *IEEE.* Vol. 15, p.339-344, 2015

[8] Kaur, M. and Singh, D. Fusion of medical images using deep belief networks. *Cluster Computing*, p.1-15. 2019. https://doi.org/10.7717/peerj.3809

[9] Sundararajan, S.K., Sankaragomathi, B. and Priya, D.S, Deep Belief CNN Feature Representation Based Content Based Image Retrieval for Medical Images. *Journal of medical systems*, *43*(6), p.174, 2019

[10] Stanley, K.O dan R. Miikkulainen. Evolving Neural through Augmenting Topologies. *Evolutionary Computation,* 10(2): p.99-127.2002

[11] Dua, D. and Graff, C, UCI Machine Learning Repository [http://archive.ics.uci.edu/ml], Irvine, CA: University of California, School of Information and Computer Science, 2019.

[12] M. Pjanic, "The role of polycarbonate monomer bisphenol-A in insulin resistance," PeerJ, vol. 5, September, 2017. https://doi.org/10.7717/peerj.3809.

[13] Z. Punthakee, R. Goldenberg, and P. Katz, "Definition, Classification and Diagnosis of Diabetes, Prediabetes and Metabolic Syndrome," Canadian Journal of Diabetes, vol. 42, April, 2018. https://doi.org/10.1016/j.jcjd.2017.10.003

[14] R. Nicoll and M. Y. Henein, "Caloric Restriction and Its Effect on Blood Pressure, Heart Rate Variability and Arterial Stiffness and Dilatation: A Review of the Evidence," International Journal of Molecular Sciences, vol. 19, Issue 3, March, 2018. https://doi.org/10.3390/ijms19030751

[15] A. Ruiz-Alejos, R. M. Carrillo-Larco. J. J. Miranda, R. H. Gilman, L. Smeeth, and A. Bernabe-Ortiz, "Skinfold thickness and the incidence of type 2 diabetes mellitus and hypertension: an analysis of the PERU MIGRANT study," Public Health Nutrition, vol. 23, Issue 1, June, 2019. doi: 10.1017/S1368980019001307

[16] Williams, L. dan Wilkins. Nurse's Five-Minute Clinical Consult: Signs and Symptoms. Edisi Pertama. Wolters Kluwer. New York.F. 2008

[17] NCD Risk Factor Collaboration (NCD-RisC) – Africa Working Group, Trends in obesity and diabetes across Africa from 1980 to 2014: an analysis of pooled population-based studies. *International journal of epidemiology*, (46), pp.1421-1432 , 2017

[18] Wadhwa, S. and Babber, K., "Artificial Intelligence in Health Care: Predictive Analysis on Diabetes Using Machine Learning Algorithms", International Conference on Computational Science and Its Applications, vol. 12250, 2020. https://doi.org/10.1007/978-3-030-58802-1_26

[19] Vuvor, F. and Egbi, B., "Correlation of diabetes mellitus and body weight of adults above the age of 30 years in a medical facility in Ghana", Diabetes & Metabolic Syndrome: Clinical Research & Reviews, vol. 11, 2017.

[20] G.E Hinton, S. Osindero, & Y.W Teh. A fast learning algorithm for deep belief nets. *Neural Computation,* (18), p.1527 – 1554, 2006

[21] S. Kamada , T. Ichimura, and T. Harada. Knowledge extraction of adaptive structural learning of deep belief network for medical examination data. *International Journal of Semantic Computing*, *13*(01), pp.67-86, 2019

[22] K.P. Lin, P.F Pai, and Y.J. Ting, Deep belief networks with genetic algorithms in forecasting wind speed. *IEEE Access*, *7*, pp.99244-99253, 2019

[23] Xu, H. and Jiang, C. Deep belief network-based support vector regression method for traffic flow forecasting. *Neural Computing and Applications*, pp.1-10, 2019

[24] Ouyang, T., He, Y., Li, H., Sun, Z. and Baek, S. Modeling and forecasting short-term power load with copula model and deep belief network. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *3*(2), pp.127-136, 2019

[25] Khatami A, Khosravi A, Nguyen T, Lim CP, Nahavandi S., Medical image analysis using wavelet transform and deep belief networks Expert Systems with Applications, Elsevier vol 86, pp 190-8, 2017,

[26] Movahedi F, Coyle JL, Sejdić E., Deep belief networks for electroencephalography: A review of recent contributions and future outlooks , IEEE journal of biomedical and health informatics, vol 22(3):pp 642-52, 2017

[27] Movahedi F, Coyle JL, Sejdić E., Deep belief networks for electroencephalography: A review of recent contributions and future outlooks, IEEE journal of biomedical and health informatics, Vol 22(3): pp 642-52, 2017.

[28] R. Jiao, X. Huang, X. Ma, L. Han and W. Tian, "A Model Combining Stacked Auto Encoder and Back Propagation Algorithm for Short-Term Wind Power Forecasting," *IEEE Access*, vol. 6, pp. 17851-17858, 2018, doi: 10.1109/ACCESS.2018.2818108

[29] Khandelwal, Renu. Deep Learning – Deep Belief Network (DBN). *Data Driven Investor*. Available: https://medium.com/datadriveninvestor/deep-learning-deep-belief-network-dbn-ab715b5b8afc.2018

[30] Belhaj Slimene, S. and Mamoghli, C. NeuroEvolution of Augmenting Topologies for predicting financial distress: A multicriteria decision analysis. *Journal of Multi-Criteria Decision Analysis*, *26*(5-6), p. 320-328, 2019

[31] H. Turabieh. Comparison of NEAT and Backpropagation Neural Network on Breast Cancer Diagnosis. *International Journal of Computer Applications*. 139(8): p.40-44. 2016.

[32] M. E. Yuksel, Advanced Engineering Informatics Agent-based Evacuation Modeling with Multiple Exits using NeuroEvolution of Augmenting Topologies. *Advanced Engineering Informatics*, (35): p.30–55, 2018.