Baghdad Science Journal

Volume 23 | Issue 5 Article 15

5-26-2026

Diabetes Case Detection Based on Neuroevolutionary Algorithm and GANet Tool

Ivan V. Stepanyan
Mechanical Engineering Research Institute of the Russian Academy of Sciences (IMASH RAN) 4, M. Kharitonyevskiy Pereulok, 101990, Moscow, Russian Federation, neurocomp.pro@gmail.com

Safa A. Hameed
Department of Mechanics and Control Processes, Academy of Engineering, Peoples' Friendship University of Russia named after Patrice Lumumba (RUDN University), Moscow, 117198, Russian Federation, sa.programmer1@gmail.com

Minhai Hou
Department of Mechanics and Control Processes, Academy of Engineering, Peoples' Friendship University of Russia named after Patrice Lumumba (RUDN University), Moscow, 117198, Russian Federation, houmenhai@gmail.com

How to Cite this Article

Stepanyan, Ivan V.; Hameed, Safa A.; and Hou, Minhai (2026) "Diabetes Case Detection Based on Neuroevolutionary Algorithm and GANet Tool," *Baghdad Science Journal*: Vol. 23: Iss. 5, Article 15.
DOI: https://doi.org/10.21123/2411-7986.5299

BAGHDAD SCIENCE JOURNAL 2026;23(5):1760–1778

Scan the QR to view the full-text article on the journal website

RESEARCH ARTICLE

Diabetes Case Detection Based on Neuroevolutionary Algorithm and GANet Tool

Ivan V. Stepanyan [1], **Safa A. Hameed** [2,*], **Minhai Hou** [2]

[1] Mechanical Engineering Research Institute of the Russian Academy of Sciences (IMASH RAN) 4, M. Kharitonyevskiy Pereulok, 101990, Moscow, Russian Federation
[2] Department of Mechanics and Control Processes, Academy of Engineering, Peoples' Friendship University of Russia named after Patrice Lumumba (RUDN University), Moscow, 117198, Russian Federation

ABSTRACT

This paper proposes a method utilizing an evolutionary algorithm (EA) to improve the learning process within the artificial neural network (ANN). The study utilized two models: the neuroevolutionary algorithm and the GANet tool. These algorithms were trained on two clinical diabetes datasets to discern health conditions. Each Feedforward Artificial Neural Network (FANN) layer in a neuroevolutionary algorithm is broken into multiple ANNs according to the number of neurons. The evolutionary technique later utilizes crossover and mutation operations to produce an ideal FANN. Enhanced artificial neural networks were subsequently incorporated to provide an innovative and robust functional artificial neural network for practical use. The GANet tool offers a novel mutation methodology. This method enables neurons to transform into multi-layered subnetworks. The tests utilizing diabetes data attained enhanced prediction accuracy. This demonstrates the importance of optimizing neural networks. It creates a strong basis for the automation of AI design and the advancement of medical diagnostics in the future. The proposed tools exhibited superior efficiency relative to alternative methods such as decision trees and support vector machines.

Keywords: Artificial neural networks, Breaking process, Evolutionary algorithm, Genetic algorithm, Optimization

Introduction

Diabetes is a prevalent chronic illness that may be lethal owing to its consequences. Prompt diagnosis is crucial for prompt intervention, since it may impede disease progression.[1] Numerous variables hinder precise projections of diabetes. Deep learning can address this difficulty.[2–4] The most prominent artificial intelligence (AI) algorithms are artificial neural networks (ANN), which draw inspiration from biological neurons, while evolutionary algorithms (EA) are founded on the principles of natural selection in biology.

Artificial neural networks (ANN) require a learning process, which is regarded as the most critical aspect for optimising ANN to address challenges. A comprehensive methodology for connection weight training in artificial neural networks (ANN) entails the evolution of connection weights, particularly when gradient information is difficult or expensive to acquire. The learning technique rectifies inefficiencies by modifying learning topologies during the evolutionary process via non-dominated solutions, facilitating an evolution-driven adjustment of learning rules[5,6] which promotes varied solutions and reduces redundancy.[7] It can discover parameters to rectify issues, including the evaluation of ANN classification and detection model performance.[8] The meticulous design of hyperparameters significantly influences the performance of deep learning models.[9] Recent advancements in machine learning technology may reveal concealed patterns that may facilitate the early

Received 23 August 2025; revised 3 January 2026; accepted 10 January 2026.
Available online 26 May 2026

* Corresponding author.
E-mail addresses: neurocomp.pro@gmail.com (I. V. Stepanyan), sa.programmer1@gmail.com (S. A. Hameed), houmenhai@gmail.com (M. Hou).

https://doi.org/10.21123/2411-7986.5299

detection of diabetes.[10] These models use historical data to forecast fresh samples.[11,12] As network capacity increases, so do the weights and biases.[13–16] Success depends on how we design the architecture, train the model, and fine-tune hyperparameters.[17,18] Evolutionary computation is integral to this process. The objective is to use natural mechanisms like reproduction, mutation, selection, and crossover. This facilitates the transformation of systems into instruments for optimisation and design. Evolutionary computing encompasses several methodologies and algorithms. These include genetic algorithms, evolutionary programming, and evolution techniques.[19] The evolutionary algorithm modifies the weights, topologies, or hyperparameters of neural networks. This seeks to enhance a fitness function that evaluates the efficacy of the neural network's learning process.[20]

The Artificial Neural Network (ANN) can be enhanced through evolutionary algorithms (EA) to improve its framework, particularly for diabetes classification. This problem necessitates accurate tools to differentiate health states based on data features. By employing mutations and crossover procedures, the ANN's parameters can be iteratively optimised to effectively tackle the diabetes classification challenge.

Neural evolution and the GANet technique employ multi-layered feedforward artificial neural networks (ANNs) to analyse diabetic datasets. Crossover and mutation promote learning and training, as well as testing accuracy. They improve the neural architecture for complicated categorisation tasks through structural searches and adaptive designs. Diabetes classification results are more accurate and efficient than other basic tools. Diabetes categorisation technologies outperform existing methods in terms of accuracy and efficiency. The evolutionary method used to optimise artificial neural networks enhances the learning process because potent procedures, like crossover and mutation, significantly refine the ANN's parameters. The primary concept involves deconstructing an artificial neural network (ANN) into individual ANNs to independently optimise each neuron through crossover and mutation operations, thereby achieving a satisfactory outcome. This method enhances the ANN's architecture by reinforcing neural connections, and it accelerates optimisation through the deconstruction process. Furthermore, it aids the learning process by refining the generalisation technique based on the independent optimisation of each neurone.

Deficiencies in deep learning research for diabetes detection include issues such as insufficient and heterogeneous datasets, a surplus of models that hinder interpretability and clinical reliability, challenges in generalisation and robustness across diverse populations, and complications from imbalanced datasets. While models often excel on training data, they typically face difficulties with new data drawn from varied demographics or clinical environments. To address these challenges, the implementation of optimised artificial neural networks (ANN) and evolutionary algorithms (EA) can enhance diabetes classification frameworks. This approach relies on precise tools to differentiate health states based on data features. Through the application of mutation and crossover techniques, the parameters of the ANN can be iteratively optimised, resulting in a powerful structure capable of accurate and rapid classification in tackling diabetes-related challenges.

The neuroevolutionary method employs independent optimisation of individual neurons. This refinement strengthens neural connections and accelerates the optimisation process, addressing challenges in deep learning for diabetes detection such as dataset insufficiencies and model interpretability. By improving the ANN through evolutionary algorithms, it creates a robust framework for accurate and rapid classification of diabetes-related issues. GANet demonstrates that genetic algorithms can evolve diabetic neural network topologies, yielding higher average and best accuracies over generations. The optimized architecture outperformed the original population in classification accuracy and resilience, with its mutation technique producing more complex subnetworks.

Literature review

Artificial intelligence (AI) tools have made significant contributions to performing classification tasks, including diabetes treatment.

Mahdi and Swadi accomplished the classification task using Discrete Wavelet Transform (DWT) in conjunction with Back Propagation Neural Networks and a Fuzzy Logic System, with an accuracy range of 85% to 92%. Artificial neural networks, fuzzy logic, and frequency domain discrete wavelet transform were explored to facilitate automatic analysis utilising personal computers. Chowdary and Kumar used a neuro-fuzzy model, specifically focusing on feature extraction, to improve diabetes classification. The goal was to enhance diabetes prediction methods, making it easier for doctors to diagnose and treat the condition, thereby reducing complications. The model's accuracy on the PIMA dataset[25] was 82.33; however, the study can address the data that is missing in smaller amounts of data. Naz and Ahuja affirmed that deep learning yields optimal outcomes

using the most advantageous extracted characteristics. The accuracy attained by functional classifiers, including Artificial Neural Network (ANN), Naive Bayes (NB), Decision Tree (DT), and Deep Learning (DL), ranges from 90% to 98%. Of the four, DL yields the most favourable outcomes for diabetes onset, with an accuracy rate of 98.07% on the PIMA dataset.[21] Jose et al. introduced a neuroevolutionary method to classify the stages of diabetic retinopathy. It employed ensembles, population reinitialisation, and simulated annealing techniques. The method achieved an accuracy, sensitivity, and specificity, as shown by kappa index ratings of 0.889, 0.889, 0.951, and 0.822. The experiments were conducted using a small data set, resulting in the best-case results. Wijaya et al. analysed how Deep Belief Networks (DBN) and Neuroevolutionary Augmenting Topology (NEAT) handle regression issues in diabetes detection. The DBN served as a starting weight for a backpropagation neural network, achieving an average accuracy of 22.61%. In contrast, the NEAT algorithm tackles the problem with 74.5% confidence. Given the significance of network architecture on the fitness score, the crossover method warrants examination. Al-Badarneh et al. showed three methods. They are stochastic and metaheuristic techniques for training the multilayer perceptron neural network. The focus is on accuracy, F1 score, and G-mean. It used ten imbalanced datasets and found that f1-score and g-mean fitness functions outperform accuracy. This experiment demonstrated that accuracy is a deceptive metric for assessing classification performance in imbalanced datasets.

This paper presents a novel diabetes prediction method using neuroevolutionary techniques and GANet tools. Both models attained high accuracy due to the implementation of the optimisation strategy using an evolutionary algorithm, enhancing prediction performance and managing network complexity. This approach could potentially lead to new research areas in medical diagnosis and AI design automation.

Materials and methods

The study used two diabetes datasets[22,23] to implement a diagnosis using neuroevolutionary and GANet tools. The first dataset has 235 samples. It covers health and illness status, including pregnancy, glucose, insulin, BMI, diabetes pedigree function, and age. The second dataset has 448 samples with six features: blood glucose level, age, heart disease, BMI, hypertension, and HbA1c level. The goal was to localize feature search and categorize a two-dimensional data array. The normalization function was used to optimize classification performance.[24,25] The scaling method, also known as "min-max feature scaling," scaled data within the [0–1] range, as expressed in Eq. (1).

$$x_i = (x_{max} - x_{min}) * \frac{x_i - x_{min}}{x_{max} - x_{min}} + x_{min} \quad (1)$$

The suggested model comprises a Functional Artificial Neural Network (FANN) in conjunction with an Evolutionary Algorithm (EA). The EA method was employed to improve the FANN learning process through three primary steps: first, deconstructing the ANN layers into multiple smaller ANNs based on the neuron count in each layer to optimize each neuron independently; second, executing the EA algorithm; and finally, conducting the combination process.

Neuroevolutionary algorithm

A suggested neuro-evolutionary phenotypic model comprises a multi-layer feedforward artificial neural network (FANN) and an evolutionary algorithm (EA). The FANN architecture comprises an input layer, four hidden layers containing three neurons each, and an output layer. The ANN architecture is optimized in stages, as shown in Fig. 1, which illustrates the optimization process cycle. The optimization persists until the outcomes meet or exceed a predetermined threshold value. The procedure encompasses the subsequent steps: (a) deconstructing the FANN layers into several ANNs according to the quantity of neu-

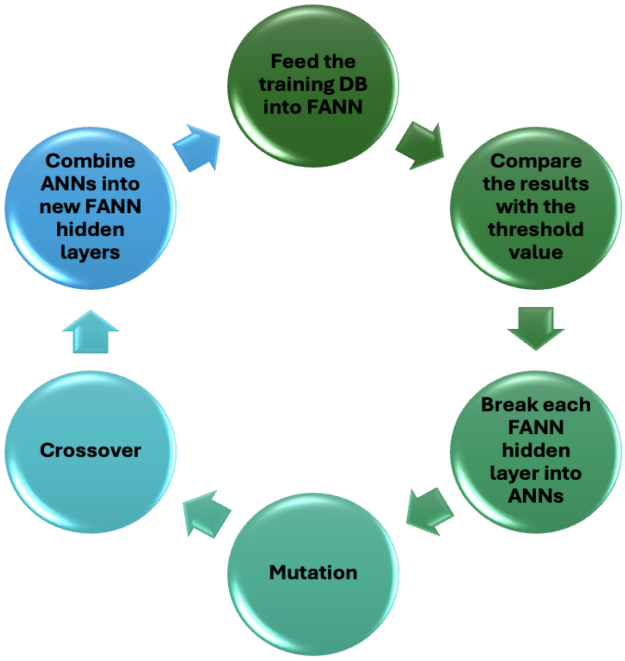

Fig. 1. A visual representation of the optimization method cycle.

rons in the hidden layers; (b) mutating each ANN; (c) executing crossover among the synapses of the artificial neural networks; and (d) amalgamating the optimized ANNs into a novel, resilient hidden layer. The phases persist until the intended objectives are achieved. The iterative approach to data testing post-goal attainment improves the model's efficiency and inventiveness.

The range of data characteristics is used to construct a weight matrix, which is subsequently multiplied by the input augmented with the bias, as delineated in Eq. (2). The neurons in the hidden layer process incoming data utilizing a ReLU function, as delineated in Eqs. (3) and (4). The output layer generates results through a ReLU function, as specified in Eq. (5). The output is then evaluated against a threshold value of 0.5; an illness status is indicated by 1 if the values are greater than or equal to 0.5, whereas the health status is indicated by 0 if the values are less than 0.5, as articulated in Eq. (6), for classification purposes. If the results are likely to be insufficient, thus rendering the learning process essential.[26–28]

$$s = \left(\sum_{i=1}^{f} W_i.X_i\right) + b \tag{2}$$

$$\sigma = \max(0,\ s) \tag{3}$$

$$\text{neuron} = \sigma(s) = \sigma\left(\left(\sum_{i=1}^{f} W_i.X_i\right) + b\right) \tag{4}$$

$$\hat{z} = \max\left(\left(\sum_{j=1}^{n} W_j.\text{neuron}_j\right) + b,\ 0\right) \tag{5}$$

$$f(x) = \begin{cases} 1, & x \geq 0.5 \\ 0, & x < 0.5 \end{cases} \tag{6}$$

i: The index of feature for each sample of data
f: The number of features
W: The weight
X: The features
b: The bias
σ: The activation function (ReLU)
neuron: The output from the neuron
$\hat{z}$: The output from FANN
n: The number of neurons
j: The index of neuron in layer

The system's implementation consists of two phases: training and testing for each indication. Eight percent of the data was designated for training, and the remaining twenty percent was allocated for testing. In the training phase, the training data is segmented into several batches, each comprising the two opposing classes. The system identifies features and implements regularization across all data attributes. As the model trains on each batch, it gains proficiency from the contending classes; with each subsequent batch, it assimilates knowledge from the varied characteristics of the competing samples. The system persistently trains on the provided data batches through various optimization phases. The learning process comprises several stages aimed at improving the efficacy of the FANN. Each layer is divided into different artificial neural networks according to the number of neurons present. To optimize an artificial neural network, an evolutionary strategy incorporating several mutation and exchange operations is essential. The advanced artificial neural networks are subsequently integrated to generate an enhanced FANN layer with greater performance.

The optimization process persists until satisfactory results are achieved, culminating in a new, extremely efficient FANN. The system validates the outcomes of each iteration of the ANN during the optimization process to guarantee consistent enhancements. Feedback from the output layer during training is crucial for making improvements that enhance model accuracy.

The first stage of the optimisation approach is dividing the FANN layers into many artificial neural networks, with each network assigned to a certain layer, depending on the number of neurons in that layer. Upon the attainment of an optimal solution by one artificial neural network, supplementary artificial neural networks begin to refine it. This incremental technique enhances the learning process by enabling each artificial neural network to function autonomously while sharing its discoveries as it advances. This enhances the functionality of the artificial neural network (FANN), yielding a swifter and more precise answer. The artificial neural networks at the input layer need an effective optimisation technique to get satisfying results. Eq. (7) indicates that the list retains the optimal artificial neural networks from the current hidden layer. This process continues until all neurons in the current hidden layer are optimised. The optimisation procedure for each neuron is repeated for every hidden layer, as seen in Fig. 2(a). The breaking process enhances and enables the optimisation of the ANN using a neuron-independent learning mechanism via mutation operations. The neuron optimisation continues until the desired output is attained, as elucidated in Fig. 3. Each layer disassembles to evaluate each neurone separately; if all neurons are optimised and provide satisfactory results, the optimised neurons amalgamate to create a new optimised layer, which proceeds to the subsequent layer for further optimisation.

Algorithm 1 elucidates the pseudocode of the neuroevolutionary algorithm, detailing the procedural

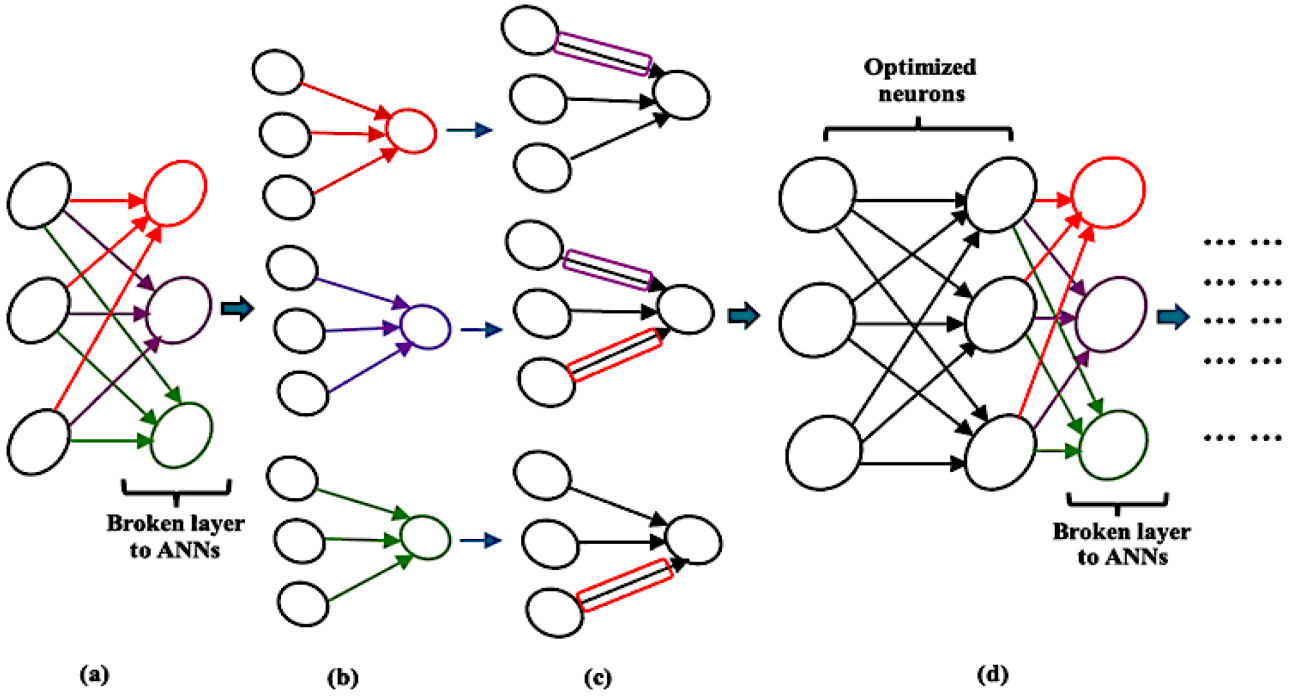

Fig. 2. The optimization stages: (a) breaking, (b) mutation, (c) crossover, and (d) combination.

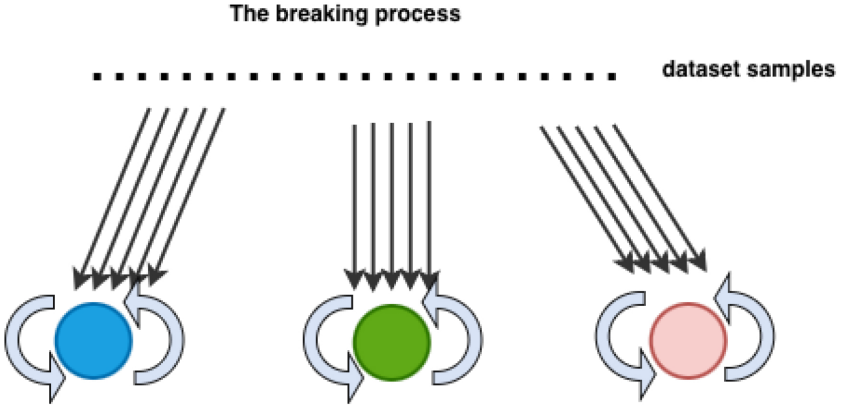

Fig. 3. The breaking process.

stages from Step 2 to Step 21 for optimising each neuron in a hidden layer, which constitutes a critical phase of the algorithm.

$$ANNs = \left[\left[ANN_{i_j}, \ldots, ANN_{n_j}\right], \ldots \left[ANN_{i_j}, \ldots, ANN_{n_L}\right]\right] \quad (7)$$

i: The index of neuron
j: The index of layer
n: Number of neurons in the layer
L: Number of layers

The mutation approach segments artificial neural networks (ANNs) based on threshold criteria to facilitate optimal outcome assessment. Each artificial neural network experiences random mutations, and the altered structure is assessed against the threshold to determine network performance. The modified ANNs are subsequently recorded in the list as delineated in Eq. (7). This iterative technique enhances network efficiency and preserves ideal configurations for future training. It enhances the overall system efficiency, enabling superior predictions and improved performance with varied datasets. The mutation has enhanced the artificial neural network, as illustrated in Fig. 2(b). This technique guarantees that each iteration surpasses the preceding one, yielding an effective framework capable of addressing complex issues. The incorporation of feedback promotes swift modifications, aiding individuals in learning and attaining

Algorithm 1: Neuroevolutionary

```
Btch ← num of batches, x← num of inputs in each batch
Loop (l, Layers)//loop to break each layer into ANNs to optimize each neuron independently
  Loop (i, neurons): //loop for neurons in one layer
    T = False
    While(T==False):
    Loop (j, Btch): //loop for batches to check the outcome for all data samples
      C1← 0
      Loop (k, x): //loop for data samples in one batch
        r ← Relu (sum_of_product (weight, input) + bais)
          if target ≥ threshold and r ≥ threshold: C1 ← C1 + 1 //counter for achieved output
          if target < threshold and r < threshold: C1 ← C1 + 1// counter for achieved output
        if C1 == x: // if the counter equal to the data samples in one batch
          Batch ← 0
          Loop (j, Btch): //loop to check the achieved weight with other batches
           C2← 0
           Loop (k, x): //loop for all data samples in one batch
             r ← Relu (sum_of_product (weight, input) + bais)
             if target ≥ threshold and r ≥ threshold: C2 ← C2 + 1 //counter for achieved output
             if target < threshold and r < threshold: C2 ← C2 + 1 //counter for achieved output
           if C2 = x: Batch ← Batch + 1// counter for achieved batches
          if Batch = Btch: T = True, new-weight ← weight else: weight ← mutation(weight)
       new-weight[i] ← w, new-weight[i] ← new-weight[i-1], new-weight[i-1] ← w //crossover process
      layer.append(new-weight[i])// Combining process
  layers.append(layer)
```

ideal outcomes. The pseudocode and flowchart in Fig. 4 elucidate the process of generating artificial neural networks.

The crossover method utilizes artificial neural networks (ANNs) developed for the current layer subsequent to the mutation phase. Following the development of artificial neural networks (ANNs) for each neuron in the current layer, the crossover functions under the assumption that the ANNs align with the quantity of neurons in that layer. The specified parameters of the ANNs are improved by swapping the synapses of neurons while preserving the same index, as seen in Eq. (8). The crossover process streamlines learning and transfer mechanisms in artificial neural networks, thereby improving their efficacy in precisely predicting and recognizing intricate patterns in datasets, as illustrated in Fig. 2(c).

$$F\left(ANN_{\mathrm{i_j}}, ANN_{\mathrm{i-1_j}}\right) = \mathrm{F}\left(ANN_{\mathrm{k_{i_j}}}, ANN_{\mathrm{k_{i-1_j}}}\right) = ANN_{\mathrm{k_{i_j}}} \Leftrightarrow ANN_{\mathrm{k_{i-1_j}}} \quad (8)$$

i: The index of neuron
j: The index of layer
k: The index of synaptic

The system is subject to periodic updates and enhancements to guarantee compliance with performance standards prior to the final assessment. Tuning facilitates essential modifications, yielding a more resilient neural network. Fig. 2(d) demonstrates the amalgamation of neural networks using enhanced neurons to formulate an optimal hidden layer, employing a threshold value to assess the adequacy of the output, while the process persists in optimizing the subsequent layer. The hidden layer will be systematically modified to improve the overall framework's performance by adjusting each layer appropriately. Consequently, the neural network enhances accuracy while optimizing computational resource utilization, leading to improved result precision. The optimization phase is essential for complex tasks, as even minor modifications can greatly influence the network's efficacy in leveraging the training data. Therefore, it is important to perpetually assess and enhance AI systems to augment their efficacy.

The GANet tool

The study shows GA_NET, a method that uses genetic algorithms. It helps create the best architectures for multilayer perceptrons. It applies structural mutation and crossover on the node and layer levels. Also, it holds on to the best individuals to speed up convergence and improve stability. GA_NET's

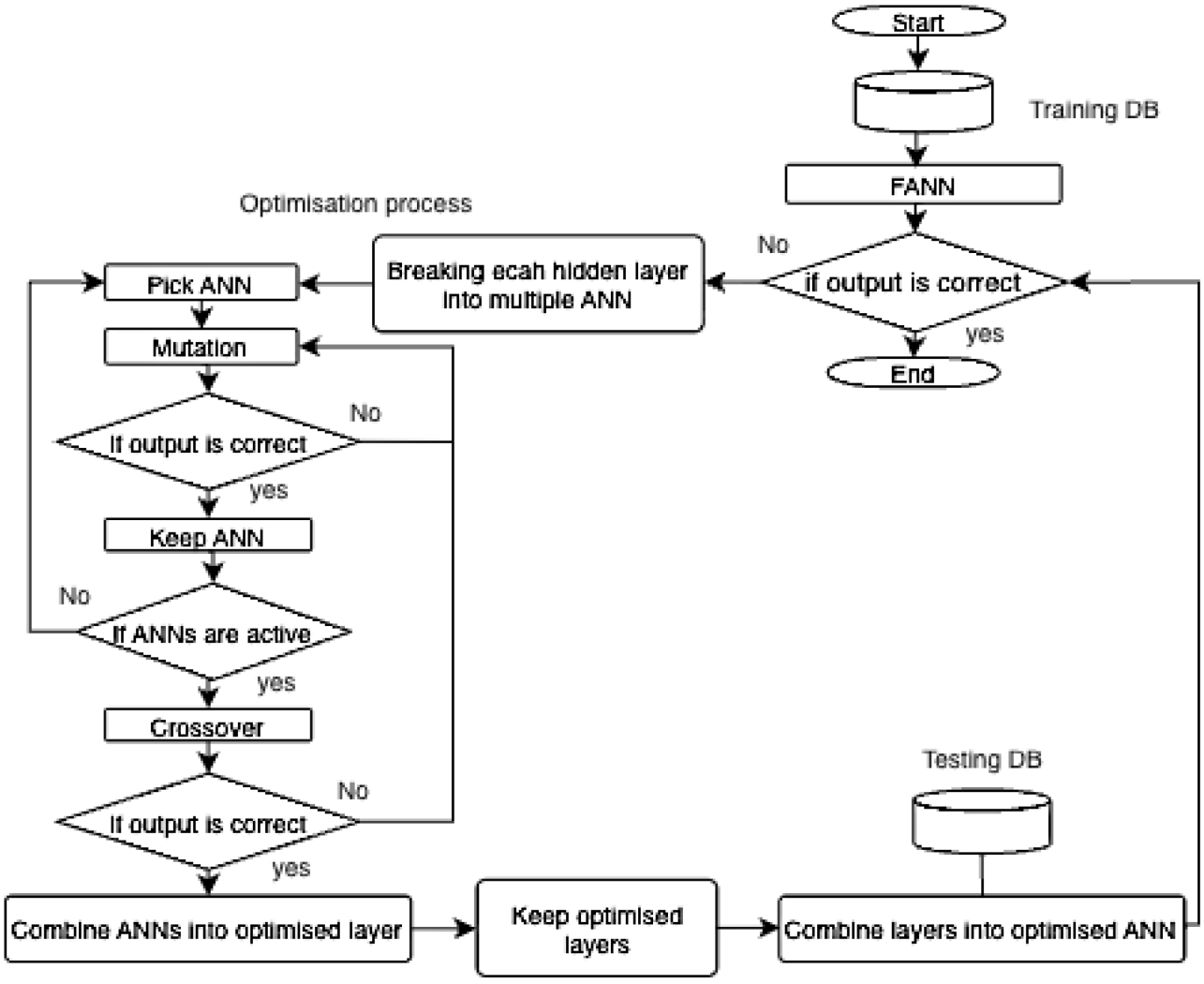

Fig. 4. The Flowchart of neuro evolutionary model.

fitness evaluation considers classification accuracy and structural complexity. The fitness function is Accuracy − 0.002 × (total nodes + 2 × layers). This formula keeps model growth in check. It shows great adaptability. It builds high-performing models using only automated processes, with no manual input.

The algorithm considers up to three hidden layers, determining their presence with decreasing probability. Activated layers are randomly chosen and their size and dropout rate are random. The fitness function evaluates network efficiency, normalizing accuracy to 0-1 and applying a SoftMax function to convert probabilities as in Eq. (9).

$$p_i = \frac{e^{f_i'}}{\sum_{j=1}^{n} e^{f_j'}} \tag{9}$$

The selection process for networks is based on a roulette wheel approach, with higher fitness scores being more suitable for selection.[29,30] GA_NET uses Sigmoid activation in hidden layers and SoftMax for the output layer. It employs the Adam optimizer with a learning rate of 0.01. The training runs for up to 300 epochs, using Early Stopping with a patience of 12. The population size is 16, with a mutation rate of 0.55. The process includes 18 generations of evolution and an elite retention ratio of 30%. The fitness function is based on accuracy minus a complexity penalty. Complexity is calculated as (total number of nodes + 2 × layers) × 0.002. Training is performed with 64 parallel threads and accuracy plots, error matrices, and ROC curves are constructed to visualize results.

The experimental procedure is illustrated in Fig. 5. This process works as follows: for each network to be chosen, selection first occurs with a probability, based on the network's fitness. Then, based on their fitness ranking, the top networks are directly selected as elites.

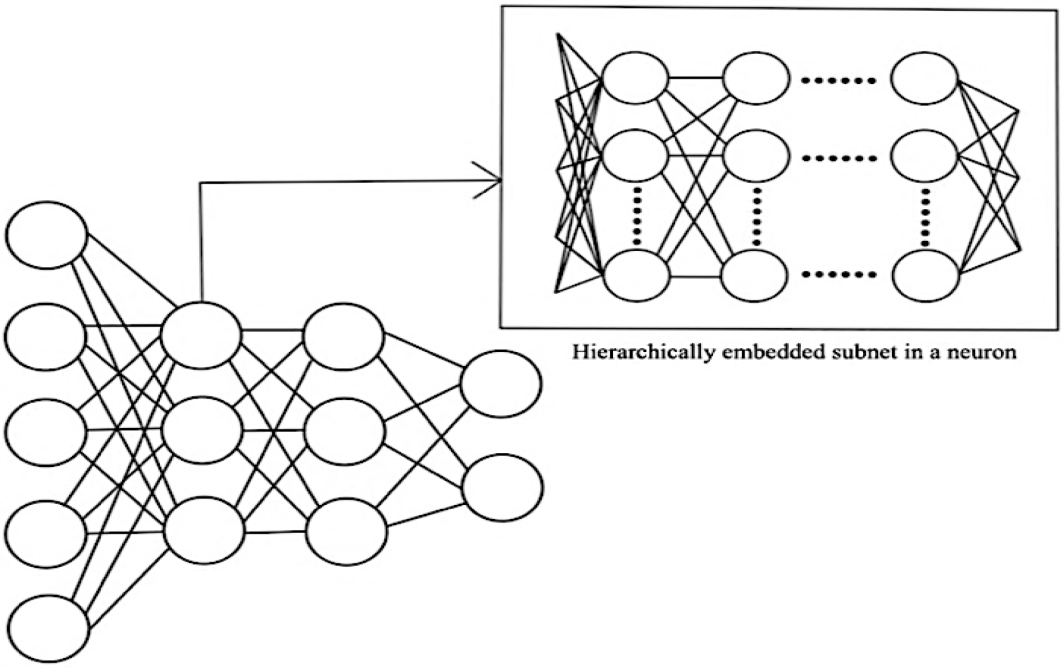

Fig. 5. The smaller mutate neural network.

The crossover step uses either the roulette choice strategy or a tournament strategy for selecting outstanding performers in N_{selected}; pairs are randomly chosen from the selected outstanding individuals to perform crossover operations. If N_{selected} has M individuals, the set of pairs C can be represented as in Eq. (10).

$$C = \left\{ \langle N_x, N_y \rangle | N_x, N_y \in N_{\text{selected}}, x \neq y \right\} \quad (10)$$

The mutation step involves introducing random changes to the binary-encoded parameters of networks, including flipping bits in network architecture parameters with a certain probability. This may lead to changes in layer sizes, number of neurons, or dropout rates. A unique feature of GANet is the ability of individual neurons to mutate into small subnets, allowing neurons to evolve into subnets with varying numbers of layers and neurons, as illustrated in Fig. 5. The flowchart and the pseudocode for the GANet tool explaine the processing steps as in Fig. 6. and in the Algorithm 2.

In addition to the neuroevolutionary algorithm and GANet model, we also evaluated several baseline models that are commonly used for diabetes prediction tasks, namely, Support Vector Machine (SVM), and Decision Tree Classifier. SVM is a powerful classification algorithm effective in high-dimensional spaces. Decision Tree Classifier is a simple yet effective model for classification tasks, par-

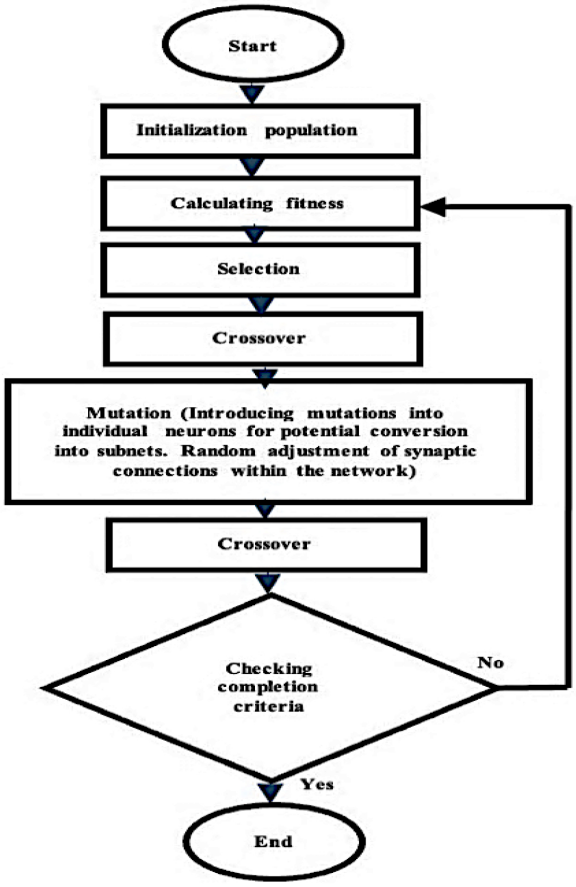

Fig. 6. GANet experimental process flowchart.

Algorithm 2: GANet tool

1: Initialize population ← N big networks (each neuron has a NodeNet)
2: while Generation < max_generations:
3: For each individual: forward pass (including all NodeNets), compute accuracy and complexity penalty
4: fitness ← accuracy – λ× (node count + layer count + total NodeNet parameters)
5: Select top elite_rate% as elites
6: Generate offspring via structural + nested NodeNet crossover
7: Apply mutations: structural and NodeNet-level changes with some probability
8: Replace population with new generation, Generation ← Generation + 1
9: Output best individual's structure (big network + NodeNets) and performance
10: End

ticularly useful for interpreting decision boundaries. These models were chosen due to their widespread use in medical diagnosis and classification problems, providing a solid benchmark for evaluating the neuroevolutionary algorithm and GANet's performance.

Results and discussion

Two binary classification tasks taken from diabetes datasets are used to validate the two suggested approaches, the Neuro Evolutionary and the GANet tool. Training and testing of every dataset are the two stages of the system's implementation. Eighty percent of the data was used for training, and the remaining twenty percent was used for testing.

The dataset has between 235 and 448 samples. The first dataset comprises 235 samples. It encompasses health and sickness indicators, including pregnancy, glucose levels, insulin, body mass index (BMI), diabetic pedigree function, and age. The second dataset has 448 samples, which are characterised by six features: blood glucose levels, age, heart disease, BMI, hypertension, and HbA1c. The objective was to localise a feature search and classify a two-dimensional data array. Each dataset has six numerical input features and a binary class label (0 = healthy, 1 = diabetes). All attributes were considered continuous variables. All trials included preparing the data via the imputation of missing values and the standardisation of input variables. The first assessment method used an 80/20 train-test split with a fixed random seed. Additionally, in the enhanced version, we performed a 5-fold cross-validation to assess the robustness of the proposed model. Four standard metrics were used to assess categorisation performance across all experiments: accuracy, F1 score, precision, and recall. The criteria were consistently applied to the neuroevolutionary algorithm, the GANet model and the baseline models to provide a fair comparison.

Neuroevolutionary algorithm implementation

The proposed neuroevolutionary method used an evolutionary approach to enhance the artificial neural network. The architecture has an input layer, four hidden layers each containing three neurons, and an output layer with a single neuron to signify the classification of health or disease. The processing criteria used the ReLU function to process data in each neuron inside the hidden layers, with each neuron optimised by several rounds of mutation to get satisfactory results. The data was fed in batches of size 8. In the experimental setting, Python was used; this software environment is suitable and efficient for model creation and construction. The technique facilitates quick data processing and analysis, allowing researchers to gain insights more swiftly. The adaptability of Python allows for the integration of many libraries, thereby enhancing the overall use of the system.

The system employs two diabetes datasets to train a feedforward, multilayered artificial neural network. The input is introduced into the network, which produces the desired outcomes. The outputs are then compared to verify their authenticity. The original artificial neural network (ANN) is divided into several ANNs, with each neuron being optimized through crossover and mutation techniques. The output is assessed against a threshold value of 0.5. The system operates under two scenarios: health and sickness status. Two subsets, each consisting of 80% of the data required for training, are generated from the dataset. Each of the three layers of the feedforward artificial neural network (FANN) contains three neurons during the training process. The output is analyzed using a threshold value of 0.5, indicating a health case if the

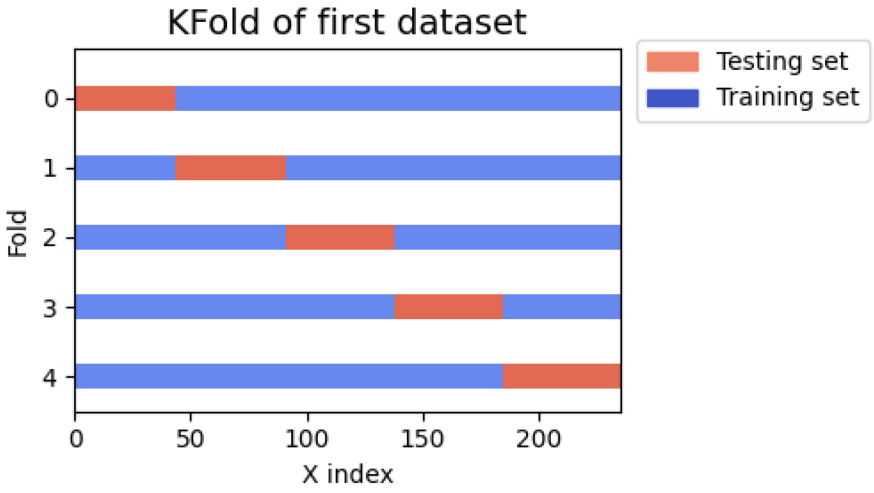

Fig. 7. The Kfold for the first diabetes dataset of neuroevolutionary model.

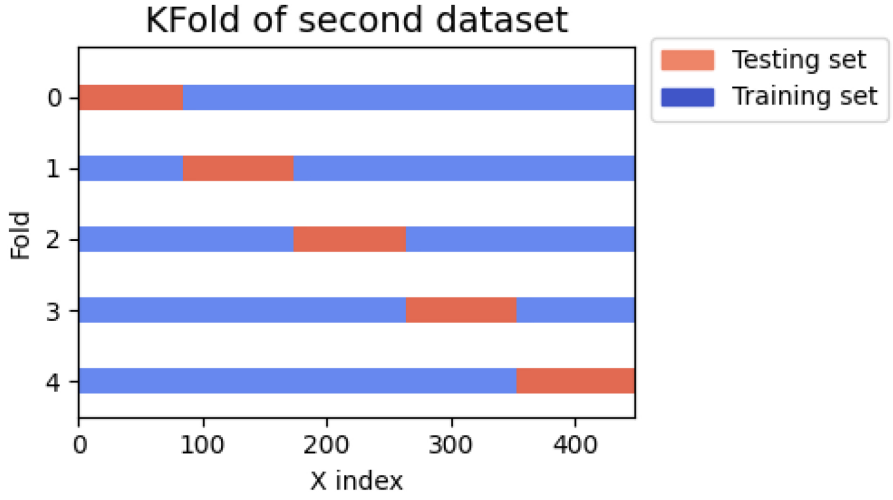

Fig. 8. The Kfold for the second diabetes dataset of neuroevolutionary model.

output meets or exceeds the threshold, and an illness case if it does not.

The process of model optimization entails partitioning the original FANN into several ANNs throughout the training phase, enhancing each neuron via a multi-iteration mutation process, and performing synaptic exchanges for the crossover function. The accuracy of each neuron is evaluated according to the number of learning iterations for each optimized ANN.

The two implemented datasets are divided into five folds each, as seen in Figs. 7 and 8. The performance of the neuroevolutionary algorithm is assessed via five folds to ensure the avoidance of overfitting. The model is applied to both training and testing datasets and evaluated using performance metrics, including F1 score, precision, recall, and accuracy, as shown in Tables 1 and 2. The findings are shown by using the confusion matrix for each fold, as seen in Figs. 9 and 10, and ultimately the model was applied to both entire datasets, as detailed in Figs. 11 and 12. The outcomes for each sample pertain to both health

Table 1. The F1-Score, precision, recall, and accuracy outcomes for the first diabetes dataset of each fold of neuroevolutionary model.

Folds	Datasets	F1 score	Precision	Recall	Accuracy
Fold0	Train data	1.0	1.0	1.0	100.0%
	Test data	1.0	1.0	1.0	100.0%
Fold1	Train data	0.979	0.959	1.0	98.919%
	Test data	1.0	1.0	1.0	100.0%
Fold2	Train data	0.991	0.982	1.0	99.459%
	Test data	0.957	0.917	1.0	97.872%
Fold3	Train data	1.0	1.0	1.0	100.0%
	Test data	1.0	1.0	1.0	100.0%
Fold4	Train data	0.99	0.98	1.0	99.459%
	Test data	0.909	0.833	1.0	93.617%

Table 2. The F1-Score, precision, recall, and accuracy outcomes for the first diabetes dataset of each fold of neuroevolutionary model.

Folds	Datasets	F1 score	Precision	Recall	Accuracy
Fold0	Train data	1.0	1.0	1.0	100.0%
	Test data	0.989	1.0	0.978	98.889%
Fold1	Train data	1.0	1.0	1.0	100.0%
	Test data	1.0	1.0	1.0	100.0%
Fold2	Train data	1.0	1.0	1.0	100.0%
	Test data	1.0	1.0	1.0	100.0%
Fold3	Train data	1.0	1.0	1.0	100.0%
	Test data	1.0	1.0	1.0	100.0%
Fold4	Train data	1.0	1.0	1.0	100.0%
	Test data	1.0	1.0	1.0	100.0%

and disease instances from both datasets. The condition value for the illness status samples is below the threshold value of 0.5, while the condition value for the health status samples is either equal to or exceeds the threshold value.

GAnet model implementation

GA_NET demonstrated robust evolutionary stability and model performance on both datasets. All data undergo the same preprocessing: mean imputation for missing values, feature standardization using StandardScaler, label conversion to integers, and a randomized train-test split in an 8:2 ratio (random_state=42).

GA_NET uses the same hyperparameter settings across both datasets: initial architecture consisting of three hidden layers with 24, 12, and 8 neurons respectively for a three-layer MLP; Sigmoid activation for hidden layers and Softmax for the output layer; Adam optimizer with a learning rate of 0.01; a maximum of 300 training epochs with Early Stopping (patience=12); population size of 16; mutation rate of 0.55; 18 generations of evolution; elite retention ratio of 30%; fitness function defined as accuracy minus complexity penalty, with complexity calculated as (total number of nodes + 2 × layers) × 0.002. Training is performed with a batch size of 64 using 4 parallel threads via the joblib library.

For the evolutionary part, the population size is set to 16 individuals and the algorithm runs for 18 generations. A mutation rate of 0.55 is used to maintain sufficient diversity in the population, while an elite preservation ratio of 30% ensures that the best-performing architectures are always carried over to the next generation.

To further assess the robustness of GANet and to address possible overfitting, diabetes_1 dataset, we

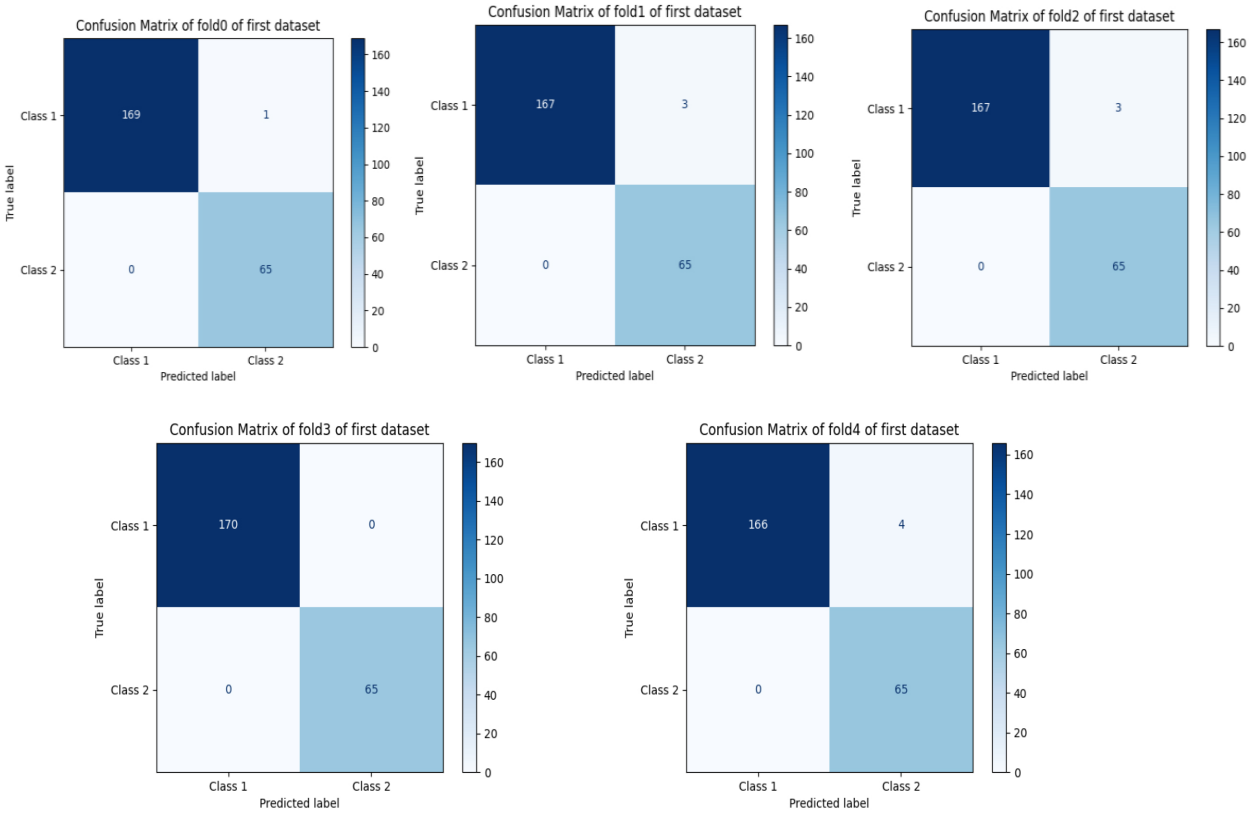

Fig. 9. The confusion matrix for each fold for the first diabetes dataset of neuroevolutionary model.

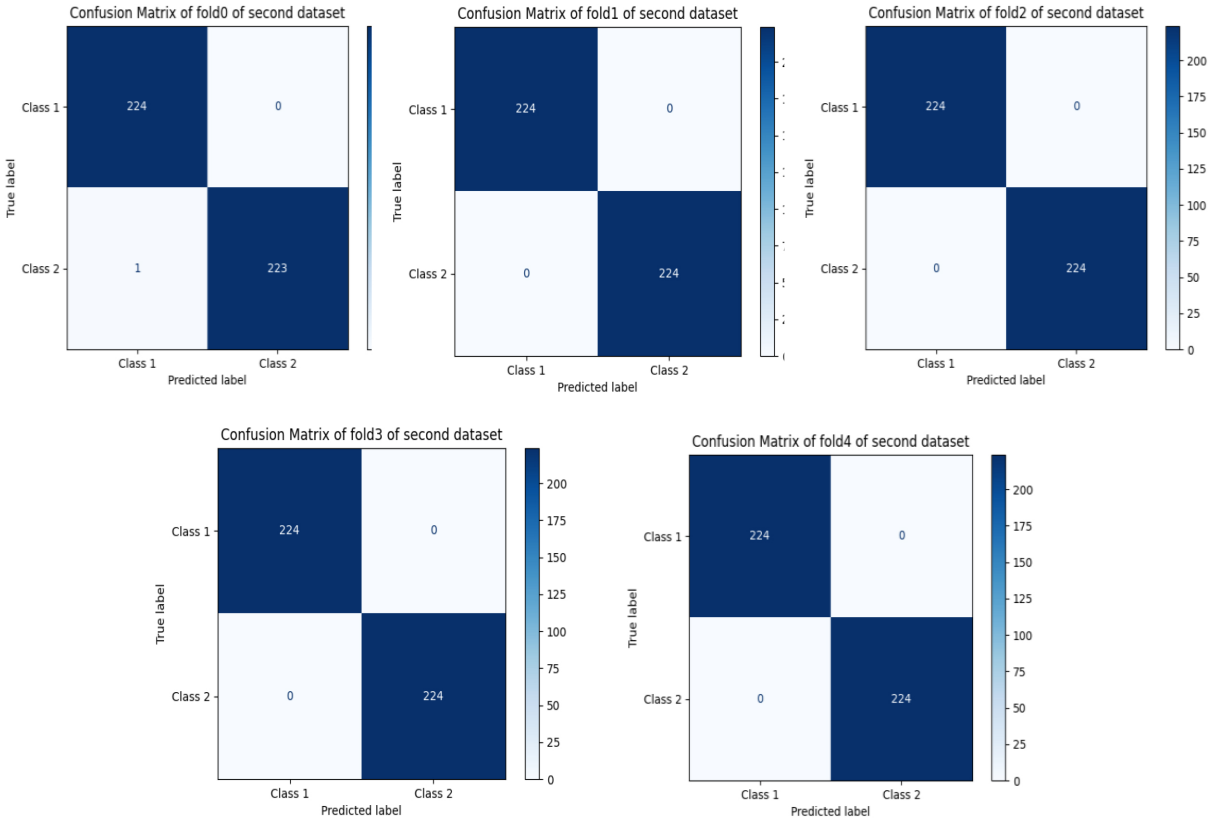

Fig. 10. The confusion matrix for each fold for the second diabetes dataset of neuroevolutionary model.

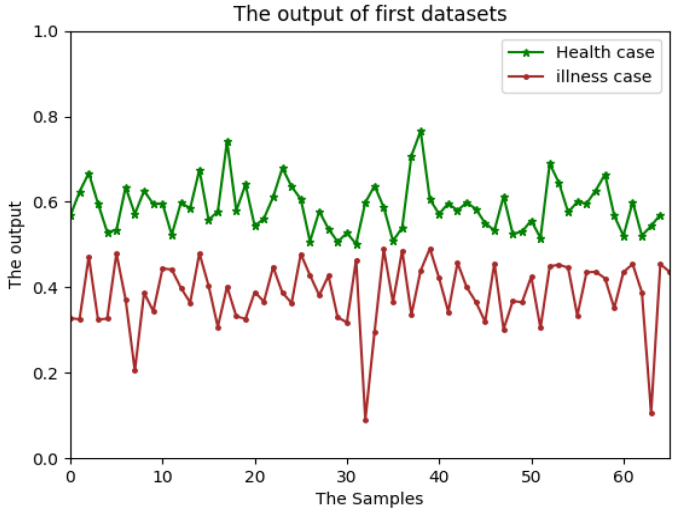

Fig. 11. The neuroevolutionary model outcome for the first diabetes dataset.

additionally performed a 5-fold cross-validation on both datasets. For each fold, we computed accuracy, F1 score, precision, recall, and the corresponding confusion matrix. The detailed fold-wise numerical results are reported in the two cross-validation tables prepared for the revision as explained in Figs. 13 and 14.

GANet achieved consistently high performance across folds on both datasets. On diabetes_1, all folds reached very high accuracy and F1 scores with only a few misclassified samples in the confusion matrices. On diabetes_2, the performance remained stable across folds, with all four metrics staying in a high range, indicating good generalization of the evolved

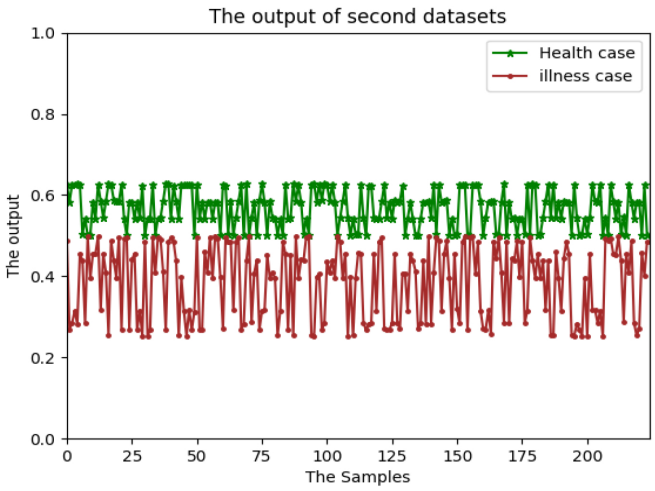

Fig. 12. The neuroevolutionary model outcome for the second diabetes dataset.

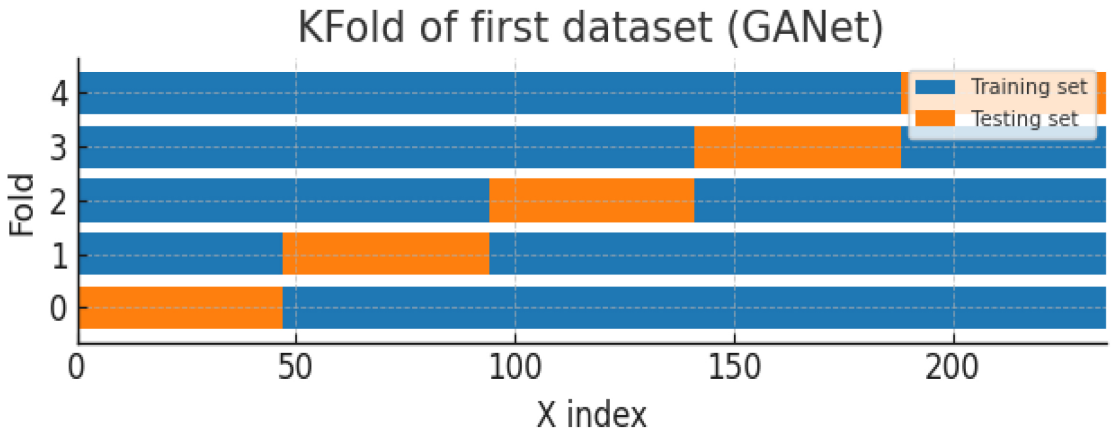

Fig. 13. The Kfold for the first diabetes dataset of GAnet model.

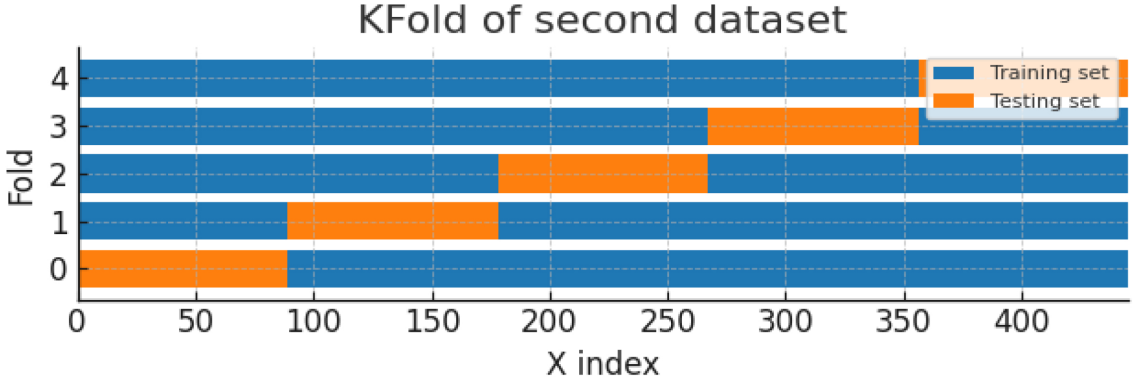

Fig. 14. The Kfold for the second diabetes dataset of GAnet model.

Table 3. The F1-Score, precision, recall, and accuracy outcomes for the first diabetes dataset of each fold GAnet model.

Fold	Accuracy	F1 Score	Precision	Recall
Fold 1	0.996	0.972	0.959	0.986
Fold 2	0.998	0.968	0.958	0.978
Fold 3	0.998	0.972	0.968	0.975
Fold 4	0.996	0.965	0.955	0.975
Fold 5	0.996	0.952	0.941	0.963

Table 4. The F1-Score, precision, recall, and accuracy outcomes for the first diabetes dataset of each fold of GAnet model.

Fold	Accuracy	F1 Score	Precision	Recall
Fold 1	0.972	0.966	0.950	0.983
Fold 2	0.970	0.970	0.946	0.995
Fold 3	0.974	0.963	0.961	0.966
Fold 4	0.972	0.965	0.961	0.968
Fold 5	0.977	0.965	0.954	0.977

architectures. The four main outcome metrics of GANet on the two datasets (aggregated over folds) are summarized in Tables 3 and 4. The confusion matrices for each fold for both diabetes datasets are illustrated in the corresponding Figs. 15 and 16. The GANet model was subsequently applied to the complete datasets, as seen in Figs. 17 and 18. The results for each sample relate to both health and illness occurrences from the two datasets. The condition value for disease status samples is below the threshold of 0.5, whereas the condition value for health status samples is equal to or surpasses the threshold.

To assess the two proposed models, namely the neuroevolutionary algorithm and the GANet model, the results were compared with various baseline tools, including XGBoost, Multi-Layer Perceptron (MLP) utilising Stochastic Gradient Descent (SGD) and Adam, Artificial Neural Networks (ANN) with backpropagation, Random Forest, and Decision Tree models. The performance of the implemented baseline models varied based on the metrics of accuracy, F1 score, precision, and recall. Table 5 presents the results for each baseline model in comparison to our proposed models, the neuroevolutionary algorithm and the GANet model.

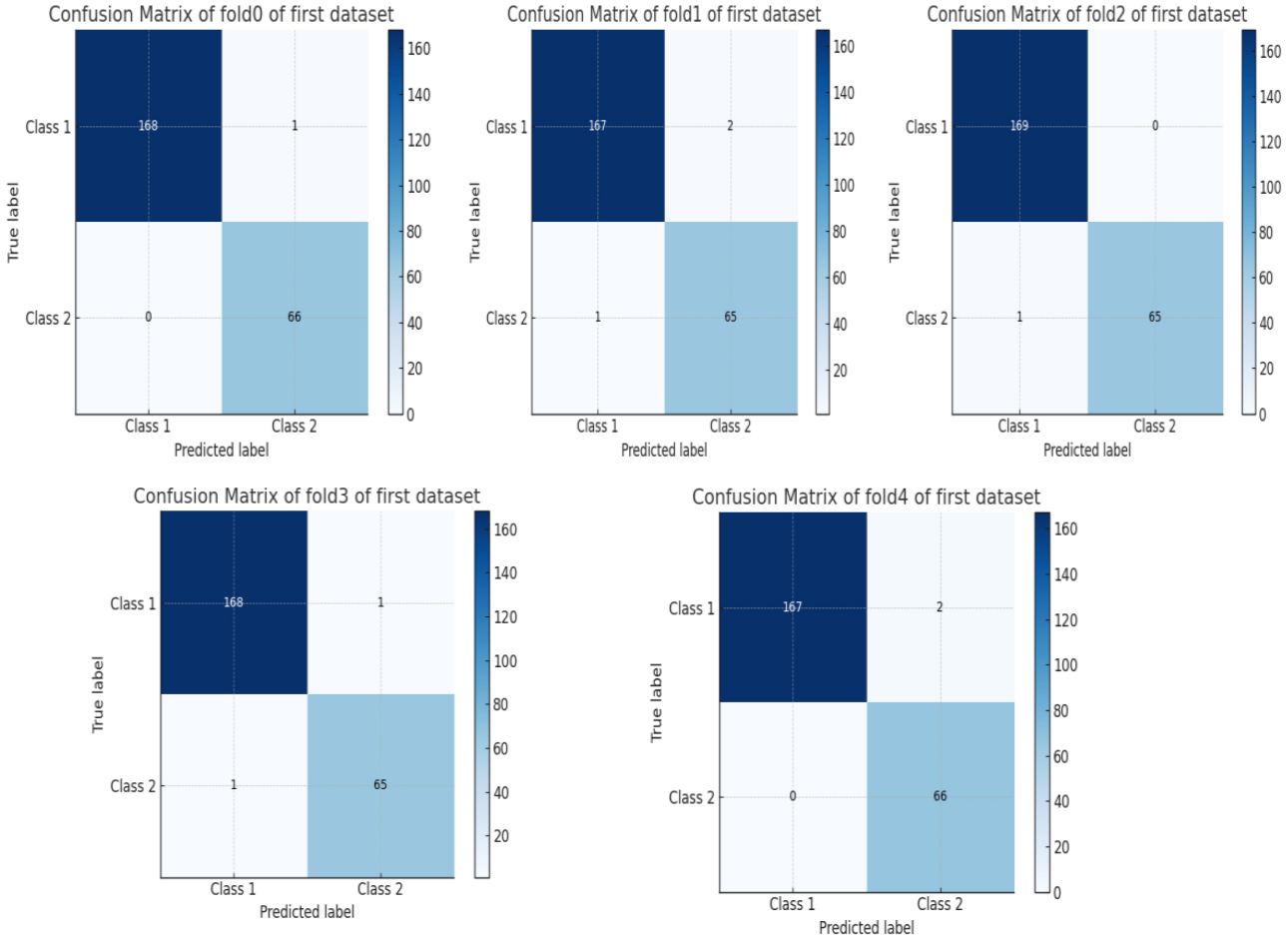

Fig. 15. The confusion matrix for each fold for the first diabetes dataset of GAnet model.

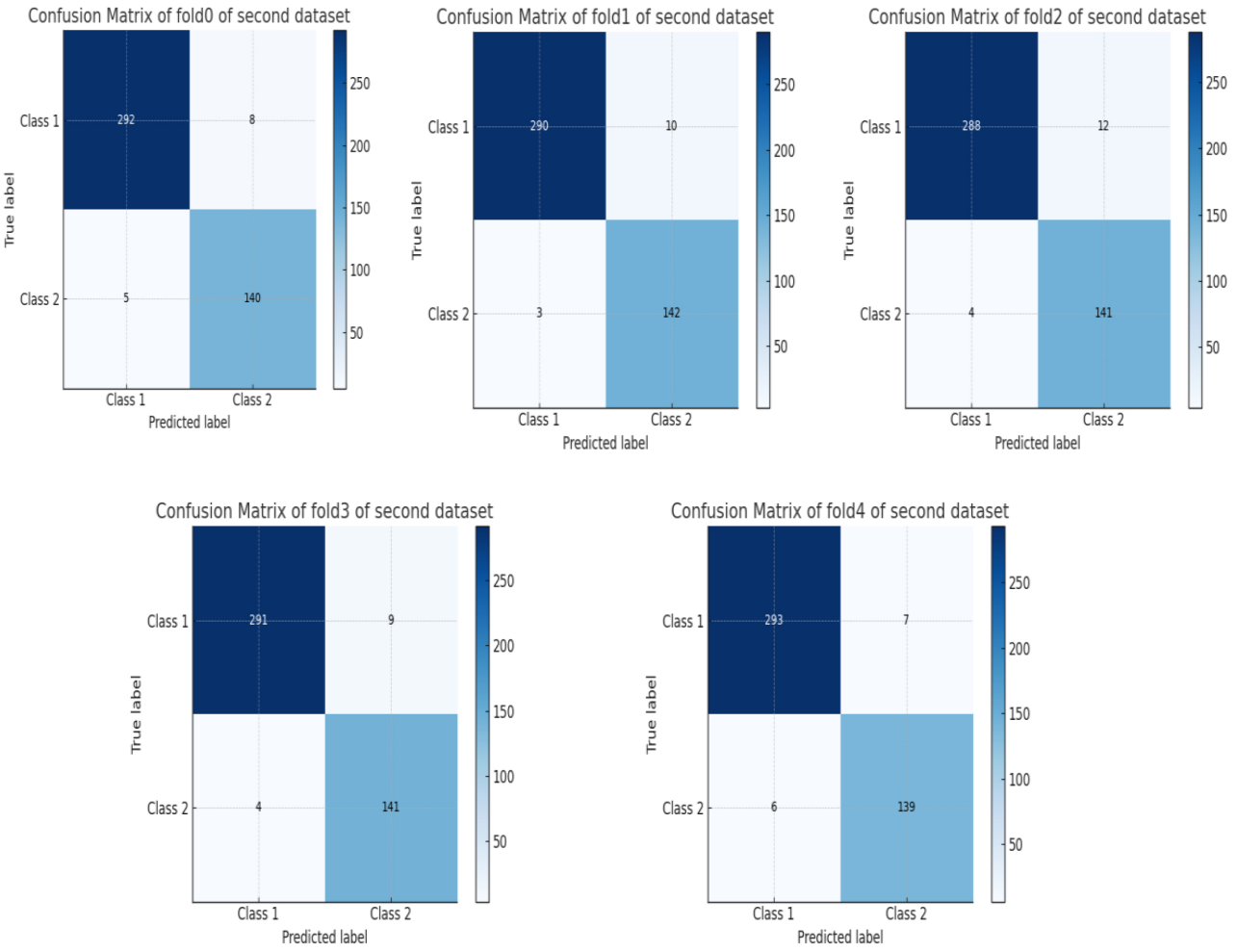

Fig. 16. The confusion matrix for each fold for the second diabetes dataset of GAnet model.

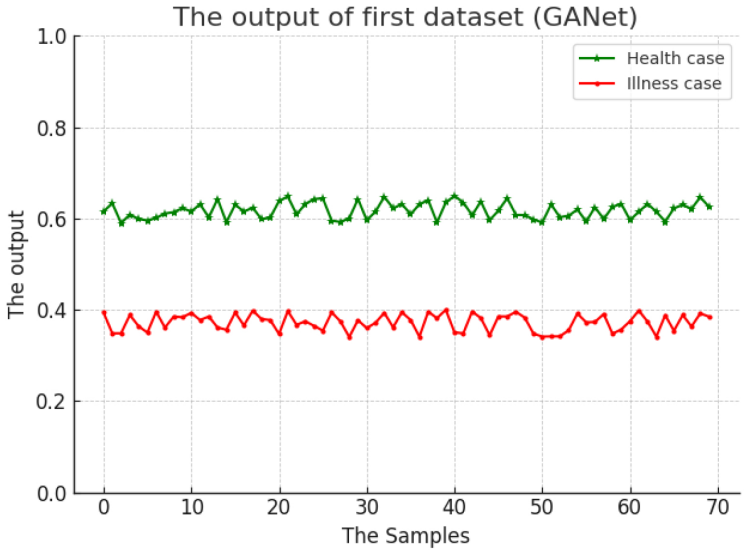

Fig. 17. The GAnet model outcome for the first diabetes dataset.

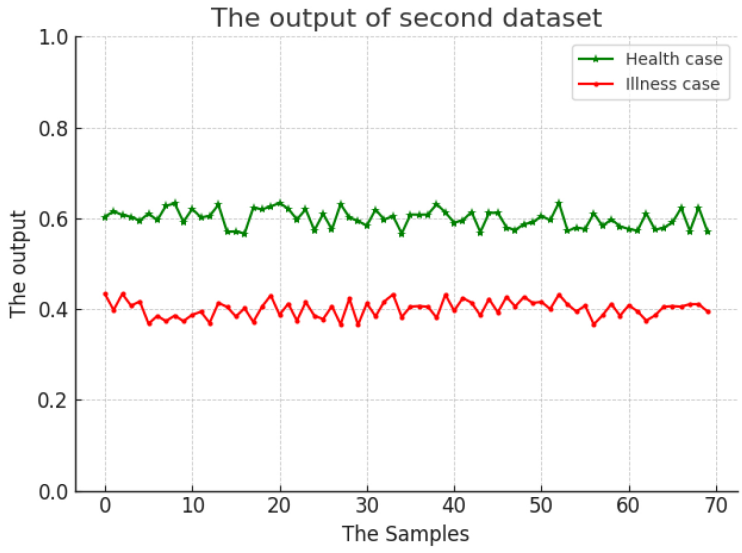

Fig. 18. The GAnet model outcome for the second diabetes dataset.

Table 5. The baseline comparisons on the diabetes datasets.

Models	Datasets	F1 score	Precision	Recall	Accuracy
XGboost	1st dataset	0.953	0.968	0.938	97.45%
	2nd dataset	1.0	1.0	1.0	100%
MLP(SGD)	1st dataset	0.667	0.971	0.508	85.96%
	2nd dataset	0.872	1.0	0.772	88.62%
MLP(Adam)	1st dataset	0.95	0.928	0.973	94.87%
	2nd dataset	0.934	1.0	0.877	96.60%
Random forest	1st dataset	0.963	0.929	1.0	97.87%
	2nd dataset	0.979	0.959	1.0	97.78%
Decision tree	1st dataset	0.881	0.981	0.8	94.04%
	2nd dataset	1.0	1.0	1.0	100.00%
ANN Backpropagation	1st dataset	0.85	1.0	0.738	92.77%
	2nd dataset	0.872	1.0	0.772	88.62%
SVM	1st dataset	0.952	0.984	0.923	97.45%
	2nd dataset	0.954	0.939	0.969	95.31%
Neuro evolutionary (Proposed algo.)	1st dataset	0.97	0.942	1.0	98.29%
	2nd dataset	1.0	1.0	1.0	100%
GAnet (Proposed algo.)	1st dataset	0.966	0.956	0.975	99.7%
	2nd dataset	0.966	0.954	0.978	97.3%

Conclusion

The neuro-evolutionary algorithm and the GANet tool combine two approaches: the evolutionary algorithm (EA) and the artificial neural network algorithm (ANN). The neuro-evolutionary algorithm uses a multi-layered feedforward artificial neural network (FANN) to receive data from two diabetes datasets. The algorithm compares results between health and illness classes, and each hidden layer is divided into multiple ANNs based on the number of neurons. The evolutionary algorithm optimizes crossover and mutation phases in each generation of the learning process for each ANN. The system's implementation yielded a high accuracy rate during training and testing. GA_NET, a self-evolving neural network architecture design method, demonstrated efficient structural search capability and stable performance across binary classification tasks of varying complexity. It automatically generates task-adaptive architectures through genetic mechanisms and converges to optimal solutions with elite retention strategies. The neuro-evolutionary algorithm and GA_NET are practical and valuable tools for neural architecture optimization, particularly suitable for classification problems with unknown structural priors.

Authors' declaration

- Conflicts of Interest: None.
- We hereby confirm that all the Figures and Tables in the manuscript are ours. Furthermore, any Figures and images that are not ours have been

included with the necessary permission for republication, which is attached to the manuscript.
- No animal studies are present in the manuscript.
- Author(s) signed on ethical consideration's approval.
- Ethical Clearance: The project was approved by the local ethical committee at RUDN University, the work was carried out within the framework of the State assignment, the code of the scientific topic FFGU-2024-0019

Authors' contributions statement

I. S., S. H., and M. H. designed the study, S. H. performed the neuroevolutionary experiments and simulations, and M.H. performed the GANet tool experiments and simulations, S. H., and M. H. analyzed the data, and S. H. wrote the paper with input from all authors.

Data availability

The datasets analyzed during the current study are publicly available in the following repositories: https://www.kaggle.com/datasets/iammustafatz/diabetes-prediction-dataset

https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database

References

1. Zhou H, Myrzashova R, Zheng R. Diabetes prediction model based on an enhanced deep neural network. EURASIP JWCN. 2020 Jul 17;2020(1):13. https://doi.org/10.1186/s13638-020-01765-7.
2. Aquino-Brítez D, Gómez JA, Noguera JLV, García-Torres M, Román JCM, Gardel-Sotomayor PE, *et al.* Automatic diagnosis of diabetic retinopathy from fundus images using neuro-evolutionary algorithms. Stud Health Technol Inform. 2022 Jun;290:689–693. https://doi.org/10.3233/shti220166.
3. Balasubramanian K, Ananthamoorthy NP. Improved adaptive neuro-fuzzy inference system based on modified glowworm swarm and differential evolution optimization algorithm for medical diagnosis. Neural Comput Appl. 2020 Nov 24;33(13):7649–60. https://doi.org/10.1007/s00521-020-05507-0.
4. Alasaady MT, Aris TNM, Sharef NM, Hamdan H. A proposed approach for diabetes diagnosis using neuro-fuzzy technique. BEEI. 2022 Sep 29;11(6):3590–7. https://doi.org/10.11591/eei.v11i6.4269.
5. Yao X. Evolving artificial neural networks. Proceedings of the IEEE. 1999;87(9):1423–47. https://doi.org/10.1109/5.784219.
6. Ming F, Gong W, Jin Y. Growing neural gas network-based surrogate-assisted pareto set learning for multimodal multi-objective optimization. Swarm Evol Comput. 2024 Jun 1;87:101541–1.
7. Saadatmand H, Mohammad-R. A. Many-objective jaccard-based evolutionary feature selection for high-dimensional imbalanced data classification. IEEE Trans Pattern Anal Mach Intell. 2024 Jan 1. https://doi.org/10.1109/TPAMI.2024.3416196.
8. Atillah ME, Fazazy KE, Riffi J. Classification of Arabic alphabets using a combination of a convolutional neural network and the morphological gradient method. Baghdad Sci J. 2023 May 20. https://doi.org/10.21123/bsj.2023.7877
9. Chowdary PBK, Kumar RU. Diabetes Classification using an Expert Neuro-fuzzy Feature Extraction Model. Int J Adv Comput Sci Appl. 2021 Jan 1;12(8). https://doi.org/10.14569/ijacsa.2021.0120842.
10. Naz H, Ahuja S. Deep learning approach for diabetes prediction using PIMA Indian dataset. J Diabetes Metab Disord. 2020 Apr 14;19(1):391–403. https://doi.org/10.1007/s40200-020-00520-5.
11. Olisah CC, Smith L, Smith M. Diabetes mellitus prediction and diagnosis from a data preprocessing and machine learning perspective. Comput Methods Programs Biomed. 2022 Mar 31;220:106773. https://doi.org/10.1016/j.cmpb.2022.106773.
12. Vázquez L, Mello-Román JC, Pinto-Roa DP, Gomez S, Ayala J, Aquino-Brítez D, *et al.* Multiclass diabetic retinopathy classification of eye fundus images small datasets performance improvement – a Neuroevolution approach. CLEI 2023. 2023. https://doi.org/10.1109/CLEI60451.2023.10346184.
13. Bhandari S, Pathak S, Jain SA. A literature review of early-stage diabetic retinopathy detection using deep learning and evolutionary computing techniques. Arch Comput Methods Eng. 2022 Sep 13;30(2):799–810. https://doi.org/10.1007/s11831-022-09816-6.
14. De Campos Souza PV. Fuzzy neural networks and neuro-fuzzy networks: A review the main techniques and applications used in the literature. Appl Soft Comput. 2020 Apr 29;92:106275. https://doi.org/10.1016/j.asoc.2020.106275.
15. Wijaya ABM, Ikawahyuni DS, Gea R, Maedjaja F. Role Comparison between deep belief neural network and neuroevolution of augmenting topologies to detect diabetes. JOIV. 2021 May 8;5(2):156–61. https://doi.org/10.30630/joiv.5.2.448.
16. Al-Badarneh I, Habib M, Aljarah I, Faris H. Neuro-evolutionary models for imbalanced classification problems. J King Saud Univ Comput Inf Sci. 2020 Nov 14;34(6):2787–97. https://doi.org/10.1016/j.jksuci.2020.11.005.
17. Buccheri E, Dell'Aquila D, Russo M. Artificial intelligence in health data analysis: The Darwinian evolution theory suggests an extremely simple and zero-cost large-scale screening tool for prediabetes and type 2 diabetes. Diabetes Res Clin Pract. 2021 Feb 27;174:108722. https://doi.org/10.1016/j.diabres.2021.108722.
18. Galvan E, Mooney P. Neuroevolution in deep neural networks: Current trends and future challenges. IEEE Trans Artif Intell. 2021 Jun 30;2(6):476–93. https://doi.org/10.1109/tai.2021.3067574.
19. Slowik A, Kwasnicka H. Evolutionary algorithms and their applications to engineering problems. Neural Comput Appl. 2020 Mar 16;32(16):12363–79. https://doi.org/10.1007/s00521-020-04832-8.
20. Kumar KS, Suryakanth GV, Giridhar K. Evolutionary computation technique combined with ensemble model for classification of diabetes. Afr J Diabetes Med. 2021 Jan 1;29(5). https://doi.org/10.54931/2053-4787.29-5-1.
21. Pima-Indians-Diabetes-Dataset, OpenML, 2022.
22. Diabetes prediction dataset. Kaggle. 2023.
23. PIMA Indians Diabetes Database. Kaggle. 2016.

24. Singh D, Singh B. Investigating the impact of data normalization on classification performance. Appl Soft Comput. 2019 May 23;97:105524. https://doi.org/10.1016/j.asoc.2019.105524.
25. Singh D, Singh B. Feature wise normalization: An effective way of normalizing data. Pattern Recognit. 2021 Sep 20;122:108307. https://doi.org/10.1016/j.patcog.2021.108307.
26. Pothina H, Nagaraja KV. Artificial neural network and math behind it. SmartCom. 2022 July 6;396:205–221. https://doi.org/10.1007/978-981-16-9967-2_21.
27. Ye JC. Geometry of Deep Learning. Singapore: Springer; 2021 Oct 21. Chapter 6, Artificial neural networks and backpropagation; p. 91–112. https://doi.org/10.1007/978-981-16-6046-7_6.
28. Joshi AV. Machine Learning and Artificial Intelligence. 2nd ed. Singapore: Springer; 2022 Sep. 02. Chap 6, Perceptron and Neural Networks; p. 57–72. https://doi.org/10.1007/978-3-031-12282-8_6.
29. Ojha VK, Abraham A, Snášel V. Metaheuristic design of feedforward neural networks: A review of two decades of research. Eng Appl Artif Intell. 2017 Feb 10;60:97–116. https://doi.org/10.1016/j.engappai.2017.01.013.
30. Hashemi ST, E OME, Kaur H. A hybrid conceptual cost estimating model using ANN and GA for power plant projects. Neural Comput Appl. 2017 Aug 29;31(7):2143–54. https://doi.org/10.1007/s00521-017-3175-5.

اكتشاف حالات مرض السكري استنادًا إلى خوارزمية التطور العصبي وأداة الشبكة الجينية

إيفان فيكتوروفيتش ستيبانيان[1]، صفا عبد الوهاب حميد[2]، مينهاي هو[2]

[1] معهد بحوث الهندسة الميكانيكية التابع للأكاديمية الروسية للعلوم, موسكو, روسيا.
[2] قسم الميكانيك وعمليات التحكم، أكاديمية الهندسة، الجامعة الروسية للصداقة بين الشعوب سميت على اسم باتريس لومومبا, موسكو, روسيا.

الخلاصة

تقترح هذه الورقة البحثية طريقةً تستخدم الخوارزميةً التطوريةً لتحسين عملية التعلم داخل الشبكة العصبية الاصطناعية. استخدمت الدراسة نموذجين: خوارزمية التطور العصبي وأداة الشبكة الجينية. تم تدريب هذه الخوارزميات على مجموعتين من البيانات السريرية لمرض السكري لتمييز الحالات الصحية. تكسر كل طبقة من طبقات الشبكة العصبية الاصطناعية في خوارزمية التطور العصبي إلى شبكات عصبية اصطناعية متعددة وفقًا لعدد الخلايا العصبية في الطبقة. تستخدم التقنية التطورية لاحقًا عمليتي التقاطع والطفرة لإنتاج شبكة عصبية اصطناعية مثالية. يتم دمج الشبكات العصبية الاصطناعية المُحسّنة لاحقًا الى شبكة عصبية اصطناعية وظيفية مبتكرة وقوية للاستخدام العملي. تقدم أداة الشبكة الجينية منهجيةً جديدةً للطفرة. تُمكّن هذه الطريقة الخلايا العصبية من التحول إلى شبكات فرعية متعددة الطبقات. حققت الاختبارات التي تستخدم بيانات مرض السكري دقة تنبؤ مُحسّنة. يُظهر هذا أهمية تحسين الشبكات العصبية، ويضع أساسًا قويًا لأتمتة انظمة الذكاء الاصطناعي وتطوير التشخيصات الطبية في المستقبل. أظهرت الأدوات المقترحة كفاءة متفوقة مقارنة بالطرق البديلة مثل أشجار القرار وآلات الدعم المتجه.

الكلمات المفتاحية: الشبكات العصبية الاصطناعية، عملية التكسير، الخوارزمية التطورية، الخوارزمية الجينية، التحسين.