Journal of Information Technology Research
Volume 15 • Issue 1

A Hybrid Approach for Enhancing the Classification Accuracy for Diabetes Disease

Maryam Mohammed Al-Nussairi, Isra University, Jordan
Mohammad Ali H. Eljinini, Isra University, Jordan*
https://orcid.org/0000-0003-1680-2801

ABSTRACT

This paper proposes a new training algorithm for artificial neural networks based on an enhanced version of the grey wolf optimizer (GWO) algorithm. The proposed model is used for classifying the patients of diabetes disease. The results showed that the proposed training algorithm enhanced the performance of ANNs with a better classification accuracy as compared to the other state of art training algorithms for the classification of diabetes on publicly available Pima Indian Diabetes (PID) dataset. Several experiments have been executed on this dataset with variation in size of the population, techniques to handle missing data, and their impact on classification accuracy has been discussed. Finally, the results are compared with other nature-inspired algorithms-trained ANN. EGWO attained better results in terms of classification accuracy than the other algorithms. The convergence curve proved that EGWO had balanced the local and global search abilities because it was faster to reach better positions than the original GWO.

KEYWORDS

Artificial Neural Network, Classification, Diabetes Disease, Enhanced GWO, Grey Wolf Optimizer, Healthcare, Machine Learning, Training Algorithms

INTRODUCTION

Diabetes is a common health problem and is often described by professionals as diabetes mellitus (DM). It comprises a group of metabolic disorders that manifest in hyperglycemia (high blood sugar) in the body of the sufferer either due to insufficient insulin production, insulin intolerance, or both. It has been suggested that it is best to diagnose DM at the early stages. The World Health Organization (WHO) report of November 14, 2016, estimated that about 422 million people had diabetes, while 1.6 million deaths are associated with DM. Based on this report, it is easy to predict the severity of diabetes in patients worldwide.

About 8.5% of adults over 18 years old had diabetes in 2014, and in 2012, hyperglycemia was the primary cause of death (about 2.2 million deaths were reported). The onset of DM comes with various levels of damage to different body parts, such as the eyes, kidneys, nerves, and heart. According to

DOI: 10.4018/JITR.298024 *Corresponding Author

the *Williams Textbook of Endocrinology*, more than 382 million people had DM globally in 2013, and many of them died due to DM-related issues in both poor and wealthy countries of the world.

The US Centers for Disease Control and Prevention (CDC) predicted a 23% increase in type-II DM cases for nine years straight (2001–2009). This disease has become a source of worry to leaders of many countries and organizations, especially relating to its prevalence and prevention. Diabetes is classified into two types: type I and type II. Type I diabetes is commonly described as insulin-dependent diabetes because of the inability of the human body to produce enough insulin. This class accounts for about 10% of all diabetes cases. Type II diabetes, as per the Canadian Diabetes Association, is expected to increase from 2.5 million cases to 3.7 million cases between 2010 and 2020. Hence, diabetes is a health condition that requires early diagnosis and prevention to avoid its life-threatening consequences.

In the last few decades, many health researchers have utilized machine-learning algorithms for the prediction and classification of many diseases. Fast and accurate prediction allows for the development of early and effective treatments. For instance, Abdar et al. (2017) used two well-known data-mining-based algorithms for the detection of liver disease. Their work showed that the algorithms Boosted C5.0 and CHAID are capable of producing new rules for detecting liver disease risk factors.

Moreover, the availability of data and the rise of computational capabilities have encouraged scientists to analyze clinical data to find answers to these pressing problems. This is done using data-mining techniques to discover useful information from large health datasets. Advancements in information technology have made data mining a useful tool in diabetes studies as it has led to the improvement of healthcare delivery and increased decision-making support for better disease supervision. Aljumah et al. (2013) concentrated on predictive analyses of diabetic treatment using regression-based data-mining techniques. They employed Oracle Data Miner (ODM) software as a mining tool for predicting the mode of treating diabetes. In this study, the target variable's identification was based on their percentage. Patients' treatment processes were also considered. Patients were grouped into categories (old or young) based on their age before predicting their treatment. This study observed high predictive percentages for both the young and old control groups. The treatment predictive percentage was calculated using support vector machines (SVM).

Currently, there is no single technique that offers the highest accuracy in predicting all diseases since the excellent performance of a given classifier on one disease dataset can be outdone in another disease dataset. In a study by Bashir et al. (2016), a new hybridization of different classifiers was proposed for the prediction and classification of diabetes. This hybridization approach was proposed to overcome the issues associated with each of the classifiers. In the next section, related work is presented and discussed. Then we introduce the methodology, results, and discussion, and we offer some closing thoughts in the conclusion.

Related Work

Different classification frameworks have been discussed and described by Komi et al. (2017). These frameworks are based on different parameters, such as skin thickness, insulin, glucose, blood pressure, body mass index (BMI), age, and diabetes type. However, pregnancy was not included as a parameter during the study to predict DD. This study only depended on a small sample to predict diabetes using five algorithms: GMM, advanced neural networks (ANN), SVM, EM, and LR. From the results, the study found ANN to be the best-performing algorithm in terms of diabetes prediction accuracy.

Likewise, Kavakiotis et al. (2017) identified ML algorithms suitable for the prediction of different medical datasets, including the DD dataset. This study employed SVM, LR, and NB based on ten-fold cross-validation for the prediction of different medical datasets, including DD. The study also compared the performance and accuracy of the employed algorithms and found SVM to offer the best accuracy compared to the others considered. In a study by Nilashi et al. (2017), CART was employed for fuzzy rule generation. PCA and EM were also used as clustering algorithms for data preprocessing prior to rules application. This study employed different medical datasets (MD), such as those dealing with

diseases of the heart, breast cancer, and diabetes. The study also developed decision support for various diseases, including diabetes. From the results, CART with preprocessed data provided better and more effective performance in disease prediction than CART without the preprocessing stage. Likewise, Pradeep and Naveen (2016) compared the performance of ML techniques on preprocessed and non-processed datasets in terms of accuracy. The study indicated the importance of data preprocessing during disease prediction as it affects the accuracy and performance of the process.

Mercaldo et al. (2017) recommended feature selection as an essential step toward increased accuracy. The study used different algorithms, such as Hoeffding Tree (HT), Multi-layer perceptron (MLP), Jrip, BayeNet, and RF, for the prediction task. In this study, different feature selection algorithms were used for the feature selection task. From the results, HT was found to provide a high level of prediction accuracy.

In contrast, Kandhasamy and Balamurali (2015) focused on different datasets, including DD, during their effort to construct models that can be applied to diverse medical datasets. The proposed classification algorithm in this study was not validated via the cross-validation evaluation method. Among the algorithms used in the study were ANN, KNN, NB, J48, ZeroR, and others. From the evaluations, NB was found to present the best accuracy on DD while KNN and ANN performed better on the other datasets.

Perveen et al. (2016) focused on the early prediction of DD using CPCSSN (the Canadian Primary Care Sentinel Surveillance Network) dataset and three ML methods. This study used Bagging, Adaboost, and J48 for DD prediction. It compared the performance of these frameworks and found the Adaboost method to be the most effective and accurate in using Weka data-mining tools.

Kamadi et al. (2016) focused on the identification of classification problems, emphasizing data reduction as a significant problem in classification tasks due to its influence on prediction accuracy. The study noted the need for reducing the data to obtain better and more accurate performance. PCA was used in this study for data preprocessing, while the modified DT and Fuzzy approaches were used for the prediction task. The study observed that the performance of the system with a reduced dataset was better, thereby highlighting the importance of data reduction.

Among the analyzed ML techniques, DT offered the best DD prediction accuracy on a non-processed dataset, while RF and SVM performed better on the preprocessed dataset. Santhanam and Padmavathi (2015) used GA and K-means for data dimension reduction in a bid to improve performance. They also used SVM for the prediction task, as it performed well in terms of accuracy on small DDs by selecting only five parameters. The study employed ten cross-validation approaches as the evaluation method. From the evaluation, performance was better on the reduced dataset compared to the large dataset.

Meng et al. (2013) used different data-mining methods for DD prediction on real-world datasets using a structured questionnaire. This study employed SPSS and Weka tools during the data analysis and prediction phases. The study compared three techniques (ANN, LR, and J48) and found the J48 ML technique to offer the best efficiency and accuracy.

METHOD

In 2014, Mirjalili et al. (2014) developed the grey wolf optimizer (GWO) as a new SI algorithm based on inspiration from the hunting style and leadership hierarchy of the grey wolf. They proved the superior performance of the standard GWO compared to PSO, GSA, DE, and FEP. However, the GWO can be easily hybridized and used for practical engineering problems due to its natural principle, fast search speed, easy realization, and high search precision. Being that GWO is a newly introduced algorithm, little research has explored it; hence, its process and theoretical development are yet to be perfected. More studies are required to improve the performance of GWO.

Many SI algorithms have been developed with inspiration from the behaviors of individual species. The GWO mimics the internal leadership hierarchy of grey wolves in that, during its searching process,

three solutions are used to assess the position of the best solution. In contrast, the best solution for other SI algorithms is searched using only a single solution. Thus, GWO can significantly reduce the chances of local optimum entrapment. To ensure a proper balance between the exploitation and exploration capabilities of GWO, an improved GWO with evolution and elimination mechanisms is proposed. The new GWO is developed by adding the biological evolution and SOF principles of natural biological updating to the basic GWO. The performance evaluation of the improved GWO is done using 12 typical benchmark functions that are executed on a simulation platform. In contrast, the simulation results are matched with that of the PSO, CS, and ABC algorithms. From the results, the improved GWO performed better in terms of optimization accuracy and convergence velocity.

In the food chain, wolves are top predators and can easily catch their prey. Generally, wolves like to interact with each other, and they have a rigid social hierarchy. To imitate the internal leadership hierarchy among the wolves, four types of wolves are identified—alpha, beta, delta, and omega—depicting the best individual, the second-best, the third-best, and the rest of the pack. The hunting (optimization) in the GWO is steered by alpha, beta, and delta [8] as they are responsible for directing the other wolves (W) to the best search area. The possible position of prey during an iterative search is determined by the three wolves (alpha, beta, and delta). Equations 1 and 2 are used to update the locations of the wolves during an optimization process:

$$\vec{D} = \left| \vec{C} \cdot \overrightarrow{X_P}(t) - \vec{X}(t) \right| \tag{1}$$

$$\vec{D}(x+1) = \overrightarrow{X_P}(t) - \vec{}(t) | \tag{2}$$

where t is the tth iteration, and $\vec{\mathrm{A}}$ and $\vec{\mathrm{C}}$ are the coefficient vectors. The position vector of the prey is represented by $\overrightarrow{\mathrm{X_P}}$ while $\vec{\mathrm{X}}$ is the position of the wolf. Vectors $\vec{\mathrm{A}}$ and $\vec{\mathrm{C}}$ are expressed as follows:

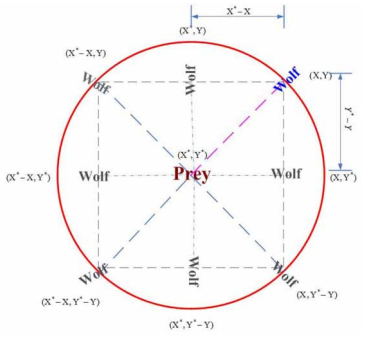
$$\vec{A} = 2a \cdot \vec{r_1} - \vec{a} \tag{3}$$

$$\vec{C} = 2 \cdot \vec{r_2} \tag{4}$$

where $\vec{\mathrm{a}}$ is the coefficient that decreases linearly in the range of [2, 0] as the number of iterations increases. Likewise, $\vec{\mathrm{r}}_1$ and $\vec{\mathrm{r}}_2$ are random vectors that range from [0, 1]. Figure 1 shows the rules for updating position as described in Equations 1 and 2. The figure shows that the wolf at position (X, Y) can move to any position around the prey using the updating formulas presented above. Even though there are only seven positions shown in Figure 1 to which the wolf can move, it is possible to make the wolf move to any position within the space near the prey by manipulating C and A (the random parameters).

In the GWO, it is believed that the optimum position (i.e., the prey) is usually the position of the alpha, beta, and delta wolves. During an ongoing iteration, the currently established best, second-best, and third-best individuals are denoted as alpha, beta, and delta, respectively, while the rest of the wolves (omega) update their positions based on the positions of the three leaders. The positions of the omega wolves are updated using the following relations:

Figure 1. Possible positions in GWO



$$\overrightarrow{D_\vartheta} = \left| \overrightarrow{C_1} \cdot \overrightarrow{X_\vartheta} - \vec{X} \right| \tag{5}$$

$$\overrightarrow{D_\beta} = \left| \overrightarrow{C_2} \cdot \overrightarrow{X_\beta} - \vec{X} \right| \tag{6}$$

$$\overrightarrow{D_\delta} = \left| \overrightarrow{C_3} \cdot \overrightarrow{X_\delta} - \vec{X} \right| \tag{7}$$

where $\overrightarrow{X_a}$, $\overrightarrow{X_2}$, and $\overrightarrow{X}$ are the respective position vectors of the alpha, beta, and delta wolves, and $\overrightarrow{C_1}$, $\overrightarrow{C_2}$, and $\overrightarrow{C_3}$ are vectors that are randomly generated. Equations 5, 6, and 7 are used to calculate the distances between the current individual's position and the position of the alpha, beta, and delta wolves. Hence, the calculation of the final position vectors of the current individual is as follows:

$$\overrightarrow{X_1} = \overrightarrow{X_\vartheta} - \overrightarrow{A_1} \cdot (\overrightarrow{D_\vartheta}) \tag{8}$$

$$\overrightarrow{X_2} = \overrightarrow{X_\beta} - \overrightarrow{A_2} \cdot (\overrightarrow{D_\beta}) \tag{9}$$

$$\overrightarrow{X_3} = \overrightarrow{X_\delta} - \overrightarrow{A_3} \cdot (\overrightarrow{D_\delta}) \tag{10}$$

$$\vec{X}(t+1) = \frac{\overrightarrow{X_1} + \overrightarrow{X_2} + \overrightarrow{X_3}}{3} \tag{11}$$

where $\overrightarrow{A_1}$, $\overrightarrow{A_2}$, $and \overrightarrow{A_3}$ are vectors that are generated randomly, and t is the number of iterations. A region within a plane is determined by the three points; thus, the best three wolves can determine the scope of the position of the prey. Being that the target solution of the GWO is assessed by three solutions, it has a low possibility of local optimum entrapment. From Equations 5–7, it is evident that the step-size of the omega tends toward alpha, beta, and delta. The definition of the omega wolves' final positions follows Equations 8–11.

Enhanced Grey Wolf Optimizer (GWO)

The original version of GWO has been implemented successfully in developing and solving different optimization problems. The parameter a is determined and initialized randomly in the range [0, 1]. The importance of a in the balancing between the exploration (global search) and exploitation (local search) of GWO is clear. Although it has a good search performance, it still needs to be balanced, meaning that the amount of global search and local search should be similar. Therefore, the parameter a should be well tuned to achieve better balancing, which leads to better performance in solving optimization problems.

The value of a indicates which searching ability the GWO uses. If the value of the parameter a is small, the wolves search for the better positions near ALPHA (i.e., the best solution), meaning that the algorithm performs a local search. Otherwise, the wolves search for a better position far from ALPHA, meaning that the algorithm performs a global search. In this study, the value of a is updated via the following equation:

$$a = a_{start} - \left(a_{start} - a_{stop}\right) \times \left(\frac{MAXITR}{ITR}\right) \tag{12}$$

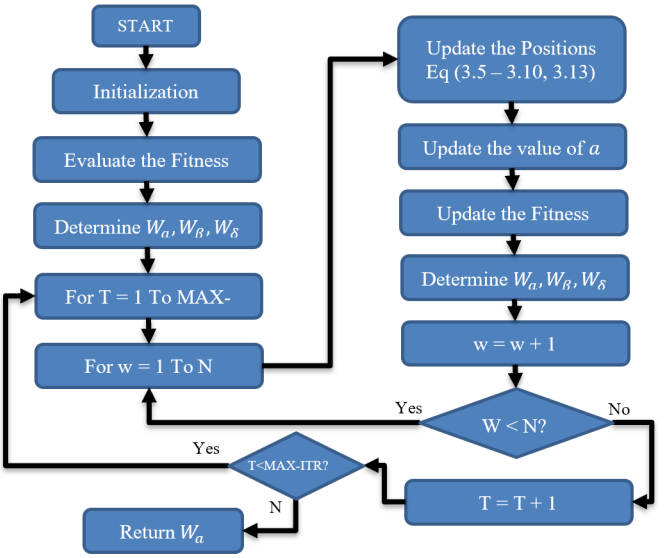
where a_{start} and a_{stop} indicate the first and final values for a, respectively, while $MAXITR$ and ITR indicate the maximum number of iterations and the current iteration, respectively. It can be seen from Equation 12 that the value of a is updated based on the iteration; it starts with a larger value and decreases gradually when the value of ITR increases. This means that the algorithm starts with exploration, but exploitation eventually increases more than exploration.

In addition to the previous modification, there is another part of the original version of GWO, Equation 11, that needs to be enhanced. In the original version, the value of X_{t+1} is updated via the average of all Xs calculated based on D_a, D_β, D_δ. In this study, the value of X_{t+1} is updated by determining the maximum value between X_1, X_2, X_3, as follows:

$$X_{t+1} = \max\left(X_1, X_2, X_3\right) \tag{13}$$

It can be seen from Equation 13 that the values of the solutions are updated via the maximum value, which may enhance the exploration ability of the searching process. The flowchart for the proposed algorithm is given in Figure two.

Figure 2. Flowchart illustrating the proposed algorithm



RESULTS AND DISCUSSION

The analyses performed in this work are based on the Pima Indian Diabetes (PID) dataset sourced from the machine-learning database of the University of California, Irvine (UCI). This dataset was initially assembled by the National Institute of Diabetes and Digestive and Kidney Diseases. The recommendations of the WHO were followed during the investigations. The subjects in this study are women who are 21 years of age or older and of Pima heritage. Various researchers have previously used this dataset to develop classification systems. The reason for selecting this dataset is to facilitate the benchmarking process with other previous studies on the problem of PID diagnoses. The dataset consists of 768 instances, and each instance is associated with eight features. The proposed algorithm used for training the MLP network is evaluated using a standard dataset for diabetes (the PID dataset).

The neural network architecture used in this study is a single hidden layer, meaning that the network consists of three main layers (input, hidden, and output). Each layer includes several nodes or neurons, as follows:

1. **Input Layer:** The number of nodes in this layer is equal to the number of features in the dataset. Thus, there are eight nodes in the input layer.
2. **Hidden Layer:** The number of nodes in this layer is double the input layer plus one (the bias), which means that there are 17 nodes in this layer.
3. **Output Layer:** The number of nodes in this layer is one, which represents the class.

The architecture of the neural network used in this study has been used in the literature and is presented in Figure 3.

The proposed algorithm is evaluated based on several evaluation matrices. These matrices are calculated based on four evaluation parameters: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). These parameters are derived from the confusion matrix. The confusion matrix is used widely in the evaluation process of binary classification problems and is illustrated in Figure 4.

Figure 3. The architecture of ANN

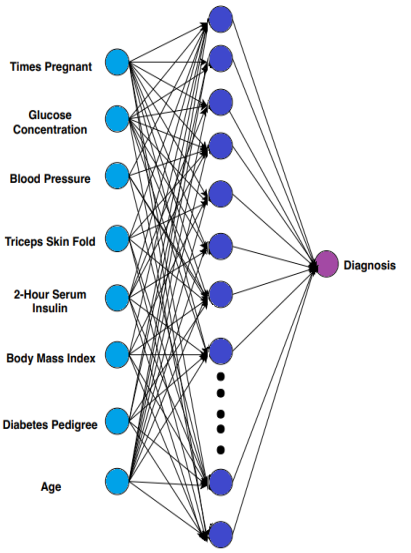


Figure 4. The general form of the confusion matrix

		ACTUAL CLASS	
		Positive	Negative
PREDICTED CLASS	Positive	TP	FP
	Negative	FN	TN

The evaluation metrics used in this work are:

1. **Accuracy:** Indicates the percentage of the correctly classified samples, as follows:

$$Accuracy = \frac{(TP + TN}{TP + TN + FP + FN} \tag{14}$$

2. **Specificity:** Indicates the percentage of the correctly classified negative samples to the actual or original negative samples in the dataset, as follows:

$$Specificity = \frac{TN}{TN + FP} \tag{15}$$

3. **Sensitivity:** Measures the correctly classified positive samples to the actual or original positive samples in the dataset, as follows:

$$Sensitivity = \frac{TP}{FN + TP} \tag{16}$$

4. **Mean Square Error (MSE):** This measure indicates the average error of the prediction or classification model, as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y - \hat{y})^2 \tag{17}$$

Table 1. Experiment 1, search agents = 10 and iterations = 50

Run	TrAcc(%)	TsAcc(%)	Sensitivity	Specificity	MSE
1	0.764822134	0.717557252	0.447916667	0.873493976	0.168839971
2	0.749012	0.687023	0.385417	0.861446	0.169849
3	0.768774704	0.72519084	0.458333333	0.879518072	0.165119449
4	0.774703557	0.713740458	0.489583333	0.843373494	0.160225701
5	0.729249012	0.687022901	0.416666667	0.843373494	0.179859135
6	0.733201581	0.702290076	0.375	0.891566265	0.181002563
7	0.743083004	0.664122137	0.395833333	0.819277108	0.178332549
8	0.743083	0.709924	0.416667	0.879518	0.172803
9	0.756916996	0.713740458	0.479166667	0.84939759	0.163540961
10	0.743083004	0.698473282	0.385416667	0.879518072	0.175690606

Table 2. Experiment 2, search agents = 20 and iterations = 50

Run	TrAcc(%)	TsAcc(%)	Sensitivity	Specificity	MSE
1	0.778656	0.732824	0.489583	0.873494	0.161628
2	0.786561	0.721374	0.447916667	0.772486772	0.160049
3	0.77668	0.736641	0.46875	0.775	0.165567
4	0.764822	0.721374	0.46875	0.761904762	0.161639
5	0.772727	0.729008	0.46875	0.764397906	0.159151
6	0.768775	0.736641	0.489583333	0.756476684	0.159175
7	0.784585	0.748092	0.5	0.755102041	0.156216
8	0.774704	0.732824	0.489583333	0.760204082	0.156045
9	0.768775	0.748092	0.458333333	0.775510204	0.159211
10	0.76284585	0.744274809	0.46875	0.903614458	0.167547385

where y is the actual value, $\hat{y}$ is the predicted value, and n is the number of instances or samples in the training dataset.

The algorithm is evaluated based on different parametric settings. First, three sets of iterations (50, 100, 200) were used for the training algorithm, while the second parameter is the number of search agents or the wolves in the population. In this study, four cases are evaluated for the number of wolves (10, 15, 25, 50). Each experiment is executed ten times, and the best and the average for each experiment are recorded.

The proposed algorithm trained the ANN on the training set (66%). There are 12 total experiments, and they are presented in Tables 1–12. Each table presents the full results, which are: Training Accuracy (TrAcc), Testing Accuracy (TsAcc), Sensitivity, Specificity, and MSE.

Table 3. Experiment 3, search agents = 30 and iterations = 50

Run	TrAcc(%)	TsAcc(%)	Sensitivity	Specificity	MSE
1	0.784585	0.751908	0.489583	0.873494	0.159563
2	0.782609	0.748092	0.520833	0.879518	0.156995
3	0.770751	0.748092	0.5	0.89759	0.156342
4	0.790514	0.755725	0.5	0.903614	0.157444
5	0.774704	0.740458	0.4791667	0.762886598	0.159563
6	0.784585	0.748092	0.5	0.873494	0.158501
7	0.784585	0.751908	0.489583	0.903614	0.15578
8	0.766798	0.751908	0.479167	0.909639	0.161198
9	0.772727	0.751908	0.5	0.89759	0.154367
10	0.780632	0.748092	0.5	0.891566	0.156114

Table 4. Experiment 4, search agents = 50 and iterations = 50

Run	TrAcc(%)	TsAcc(%)	Sensitivity	Specificity	MSE
1	0.770751	0.763359	0.541667	0.891566	0.155849
2	0.774704	0.751908	0.489583	0.903614	0.155671
3	0.786561	0.755725	0.510417	0.89759	0.155564
4	0.784585	0.751908	0.5	0.89759	0.155226
5	0.788538	0.759542	0.510417	0.903614	0.152594
6	0.788538	0.748092	0.5	0.891566	0.154423
7	0.784585	0.763359	0.510417	0.909639	0.153588
8	0.772727	0.755725	0.489583	0.909639	0.157697
9	0.772727	0.763359	0.5	0.891566	0.153924
10	0.784585	0.763359	0.5	0.915663	0.153212

Table 5. Experiment 5, search agents = 10 and iterations = 100

Run	TrAcc(%)	TsAcc(%)	Sensitivity	Specificity	MSE
1	0.770751	0.721374	0.46875	0.86747	0.158888
2	0.772727	0.725191	0.510417	0.849398	0.159668
3	0.764822	0.740458	0.458333	0.903614	0.163697
4	0.784585	0.751908	0.510417	0.891566	0.157043
5	0.782609	0.748092	0.489583	0.89759	0.157952
6	0.788538	0.748092	0.489583333	0.760204082	0.155448
7	0.752964	0.694656	0.395833	0.86747	0.167249
8	0.772727	0.751908	0.489583	0.903614	0.157414
9	0.772727	0.729008	0.458333	0.885542	0.156264
10	0.774704	0.721374	0.46875	0.86747	0.161926

Table 6. Experiment 6, search agents = 20 and iterations = 100

Run	TrAcc(%)	TsAcc(%)	Sensitivity	Specificity	MSE
1	0.772727	0.748092	0.5	0.891566	0.155464
2	0.77668	0.748092	0.489583	0.89759	0.155048
3	0.77668	0.744275	0.5	0.885542	0.15637
4	0.77668	0.751908	0.5	0.89759	0.155919
5	0.77668	0.744275	0.489583	0.891566	0.157308
6	0.778656	0.744275	0.510417	0.879518	0.157333
7	0.77668	0.748092	0.5	0.891566	0.156796
8	0.778656	0.744275	0.510417	0.879518	0.154937
9	0.77668	0.748092	0.489583	0.89759	0.15489
10	0.784585	0.744275	0.479167	0.89759	0.156322

The tables indicate that the proposed training algorithm is stable, and the classification accuracies are enhanced as the number of search agents increases. Moreover, the number of iterations has the same effect on the searching process. However, the differences are not as large across all experiments. The number of searches has a vital effect on the algorithm because the probability of finding new better solutions is increased. This means that the global search ability of the algorithm may find better positions for the agents in the search space, which decreases the chances for the algorithm to become trapped in the local optimum. In addition, the modification of GWO also helps the search agents perform better balancing between the searching around the alpha—or the local search—and searching far from the alpha—or the global search. Table 13 presents a comparison between the original version of GWO with the enhanced version in terms of classification accuracy and MSE. The number of iterations and the search agents is fixed to (200, 50), respectively.

Figure 5 presents the convergence analysis for the original and the enhanced GWOs. The figure shows that the enhanced version has a faster convergence than the original one, meaning that the solutions find better positions.

Table 7. Experiment 7, search agents = 30 and iterations = 100

Run	TrAcc(%)	TsAcc(%)	Sensitivity	Specificity	MSE
1	0.782609	0.751908	0.5	0.89759	0.155477
2	0.788538	0.736641	0.46875	0.891566	0.155369
3	0.786561	0.751908	0.5	0.89759	0.155058
4	0.784585	0.744275	0.489583	0.891566	0.154225
5	0.784585	0.751908	0.5	0.89759	0.153774
6	0.782609	0.759542	0.5	0.909639	0.155372
7	0.77668	0.755725	0.5	0.903614	0.155085
8	0.77668	0.759542	0.5	0.909639	0.154857
9	0.778656	0.751908	0.510417	0.891566	0.154752
10	0.782609	0.770992	0.520833	0.915663	0.154294

Table 8. Experiment 8, search agents = 50 and iterations = 100

Run	TrAcc(%)	TsAcc(%)	Sensitivity	Specificity	MSE
1	0.780632	0.751908	0.489583	0.903614	0.155127
2	0.780632	0.744275	0.479167	0.89759	0.153917
3	0.778656	0.759542	0.5	0.909639	0.155014
4	0.784585	0.751908	0.489583	0.903614	0.153965
5	0.784585	0.755725	0.5	0.903614	0.154741
6	0.778656	0.748092	0.5	0.891566	0.155325
7	0.782609	0.751908	0.5	0.89759	0.153528
8	0.784585	0.736641	0.46875	0.891566	0.155879
9	0.782609	0.751908	0.510417	0.891566	0.153237
10	0.778656	0.755725	0.5	0.903614	0.15558

The proposed training algorithm is compared with other state-of-the-art algorithms. These algorithms, with their controlling parameters, are presented in Table 14.

The comparison between the algorithms is presented in Table 15. The table provides the average of all ten runs of the algorithms and the standard deviations. The results indicate that the proposed algorithm attained a better average when compared with other state-of-the-art options. Moreover, the standard deviation proved that EGWO is stable because there is not much difference between the experiments. The balancing mechanism of EGWO has led it to avoid becoming trapped in the local optima and to explore the search space better than the other searching algorithms. It can be seen from Table 15 that the range of the classification accuracies for all algorithms is [0.69–0.76].

Table 9. Experiment 9, search agents = 10 and iterations = 200

Run	TrAcc(%)	TsAcc(%)	Sensitivity	Specificity	MSE
1	0.782609	0.751908	0.5	0.89759	0.154887
2	0.782609	0.755725	0.510417	0.89759	0.15662
3	0.774704	0.751908	0.510417	0.891566	0.155349
4	0.770751	0.736641	0.479167	0.885542	0.159355
5	0.774704	0.755725	0.520833	0.891566	0.158846
6	0.782609	0.751908	0.489583	0.903614	0.156236
7	0.788538	0.748092	0.489583	0.89759	0.154565
8	0.780632	0.751908	0.5	0.89759	0.157629
9	0.778656	0.740458	0.489583	0.885542	0.155875
10	0.786561	0.744275	0.5	0.885542	0.156658

Table 10. Experiment 10, search agents = 20 and iterations = 200

Run	TrAcc(%)	TsAcc(%)	Sensitivity	Specificity	MSE
1	0.780632	0.751908	0.520833	0.885542	0.153618
2	0.780632	0.755725	0.5	0.903614	0.155369
3	0.778656	0.751908	0.5	0.89759	0.154635
4	0.780632	0.744275	0.489583	0.891566	0.15402
5	0.790514	0.744275	0.5	0.885542	0.153328
6	0.780632	0.740458	0.489583	0.885542	0.153865
7	0.77668	0.755725	0.5	0.903614	0.155887
8	0.784585	0.748092	0.5	0.891566	0.154745
9	0.77668	0.748092	0.489583	0.89759	0.154585
10	0.786561	0.748092	0.5	0.891566	0.153567

Table 11. Experiment 11, search agents = 30 and iterations = 200

Run	TrAcc(%)	TsAcc(%)	Sensitivity	Specificity	MSE
1	0.782609	0.748092	0.489583	0.89759	0.153455
2	0.790514	0.755725	0.5	0.903614	0.154647
3	0.77668	0.755725	0.5	0.903614	0.153705
4	0.77668	0.748092	0.5	0.891566	0.155422
5	0.780632	0.751908	0.5	0.89759	0.154264
6	0.77668	0.748092	0.5	0.891566	0.155422
7	0.774704	0.748092	0.5	0.891566	0.153916
8	0.788538	0.748092	0.489583	0.89759	0.154565
9	0.786561	0.751908	0.510417	0.891566	0.155008
10	0.786561	0.751908	0.510417	0.891566	0.154457

Table 12. Experiment 12, search agents = 50 and iterations = 200

Run	TrAcc(%)	TsAcc(%)	Sensitivity	Specificity	MSE
1	0.782609	0.748092	0.489583	0.89759	0.153455
2	0.790514	0.770373	0.51354	0.914848	0.154647
3	0.77668	0.765725	0.51047	0.91243	0.153705
4	0.780632	0.766725	0.51047	0.91243	0.154788
5	0.780632	0.751908	0.5	0.89759	0.154264
6	0.780632	0.755725	0.5	0.903614	0.154788
7	0.774704	0.770342	0.5	0.891566	0.153916
8	**0.778656126**	**0.77481**	**0.520833333**	**0.921686747**	**0.163298332**
9	0.786561	0.764911	0.510417	0.891566	0.155008
10	0.786561	0.764911	0.510417	0.891566	0.154457

Table 13. Experiment 13, search agents = 50 and iterations = 200

Run	Original GWO		Enhanced GWO	
	TrAcc(%)	TsAcc(%)	TrAcc(%)	TsAcc(%)
1	0.784585	0.744275	0.782609	0.748092
2	0.782609	0.751908	0.790514	0.770373
3	0.780632	0.748092	0.77668	0.765725
4	0.788538	0.751908	0.780632	0.766725
5	0.782609	0.748092	0.780632	0.751908
6	0.782609	0.748092	0.77668	0.755725
7	0.782609	0.740458	0.774704	0.770342
8	0.788538	0.755725	0.778656126	**0.77481**
9	0.77668	0.748092	0.786561	0.764911
10	0.778656	0.751908	0.786561	0.764911

Figure 5. Convergence curves between the original and enhanced algorithms

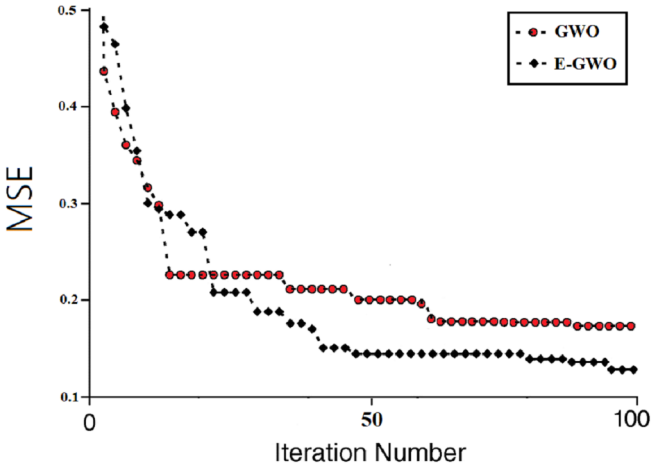


Table 14. Values for the controlling parameters

Algorithm	Parameter	Value	References
Genetic Algorithm (GA)	Crossover probability	0.9	(Faris, Aljarah and Mirjalili, 2018) (Heidari *et al.*, 2019) (Aljarah *et al.*, 2019) (Mirjalili, 2015)
	Mutation Probability	0.1	
	Selection Mechanism	Stochastic	
Deferral Evolution (DE)	Crossover Probability	0.9	
	Differential weight	0.5	
Evolutionary Strategies (ES)	λ	20	
	σ	1	
Particle Swarm Optimization (PSO)	C_1	2.1	
	C_2	2.1	
	w	0.9	
ABC	Acceleration bound	1	
Bat Algorithm (BA)	Loudness	05	
	Pulse rate	0.5	
	Frequency minimum	0	
	Frequency maximum	1	
Grey Wolf Optimizer	a	2 – 0	

Table 15. Comparison of the results

Algorithm	Testing Accuracy	
	Average	Std.
GA	0.7573	0.0128
PSO	0.7359	0.0259
PBIL	0.7436	0.0.196
ES	0.7271	0.314
FPA	0.6932	0.0487
FFA	0.7637	0.0032
BBO	0.7561	0.106
MVO	0.7617	0.0103
BA	0.7684	0.0085
MBO	0.7473	0.0203
EGWO	**0.763352**	**0.002205**

CONCLUSION

This paper proposed a new training algorithm for ANNs based on an enhanced version of the GWO algorithm. The proposed model is used for classifying diabetes patients. The results indicate that the proposed training algorithm enhanced the performance of ANNs with better classification accuracy than the other novel training algorithms for the classification of diabetes using the publicly available PID dataset. Experiments have been executed on this dataset with varying population sizes and techniques for handling missing data, and their impact on classification accuracy has been discussed. In addition, the results were compared with other nature-inspired algorithms trained for ANN. EGWO attained better results in terms of classification accuracy than the other algorithms. The proposed GWO version enhanced the balancing between local and global searchability when the parameters were calculated using the proposed equations. The new position-updating mechanism helped the best solutions (alpha, beta, and delta) to update more effectively than the original version of the algorithm because it depends on the maximum changing value of X instead of using the mean method. Proposing new algorithms for training or tuning the weights and bias of ANNs is still an important research area. The proposed training algorithm can be applied in different medical dataset problems, such as brain tumor disease, breast cancer, or leukemia. Furthermore, it can be applied to other machine-learning tasks, such as regression problems, time series problems, SVM, and support vector regression (SVR) for classification and regression problems. It can also be used to tune the controlling parameters for SVM and SVR. Finally, the proposed version of GWO can be used to solve other types of optimization problems in other research areas, such as feature selection and engineering design problems.

FUNDING AGENCY

Publisher has waived the Open Access publishing fee.

REFERENCES

Abdar, M., Zomorodi-Moghadam, M., Das, R., & Ting, I. (2017). Performance analysis of classification algorithms on early detection of liver disease. *Expert Systems with Applications*, *67*, 239–251. doi:10.1016/j.eswa.2016.08.065

Aljumah, A. A., Ahmad, M. G., & Siddiqui, M. K. (2013). Application of data mining: Diabetes health care in young and old patients. *Computer and Information Science*, *25*(2), 127–136.

Bashir, S., Qamar, U., Khan, F. H., & Naseem, L. (2016). HMV: A medical decision support framework using multi-layer classifiers for disease prediction. *Journal of Computational Science*, *13*, 10–25. doi:10.1016/j.jocs.2016.01.001

Kamadi, V. V., Allam, A. R., Thummala, S. M., & P, V. N. R. (2016). A computational intelligence technique for the effective diagnosis of diabetic patients using principal component analysis (PCA) and modified fuzzy SLIQ decision tree approach. *Applied Soft Computing*, *49*, 137–145. doi:10.1016/j.asoc.2016.05.010

Kandhasamy, J. P., & Balamurali, S. (2015). Performance analysis of classifier models to predict diabetes mellitus. *Procedia Computer Science*, *47*, 45–51. doi:10.1016/j.procs.2015.03.182

Kavakiotis, I., Tsave, O., Salifoglou, A., Maglaveras, N., Vlahavas, I., & Chouvarda, I. (2017). Machine learning and data mining methods in diabetes research. *Computational and Structural Biotechnology Journal*, *15*, 104–116. doi:10.1016/j.csbj.2016.12.005 PMID:28138367

Komi, M., Li, J., Zhai, Y., & Zhang, Y. (2017). Application of data mining methods in diabetes prediction. In *Proceedings of the 2nd International Conference on Image, Vision and Computing (ICIVC)* (pp. 1006–1010). IEEE. doi:10.1109/ICIVC.2017.7984706

Meng, X. H., Huang, Y. X., Rao, D. P., Zhang, Q., & Liu, Q. (2013). Comparison of three data mining models for predicting diabetes or prediabetes by risk factors. *The Kaohsiung Journal of Medical Sciences*, *29*(2), 93–99. doi:10.1016/j.kjms.2012.08.016 PMID:23347811

Mercaldo, F., Nardone, V., & Santone, A. (2017). Diabetes mellitus affected patients classification and diagnosis through machine learning techniques. *Procedia Computer Science*, *112*, 2519–2528. doi:10.1016/j.procs.2017.08.193

Mirjalili, S., Mirjalili, S. M., & Lewis, M. (2014). Grey Wolf Optimizer. *Advances in Engineering Software*, *69*, 46–61. doi:10.1016/j.advengsoft.2013.12.007

Nilashi, M., Bin Ibrahim, O., Ahmadi, H., & Shahmoradi, L. (2017). An analytical method for diseases prediction using machine learning techniques. *Computers & Chemical Engineering*, *106*, 212–223. doi:10.1016/j.compchemeng.2017.06.011

Perveen, S., Shahbaz, M., Guergachi, A., & Keshavjee, K. (2016). Performance analysis of data mining classification techniques to predict diabetes. *Procedia Computer Science*, *82*, 115–121. doi:10.1016/j.procs.2016.04.016

Pradeep, K. R., & Naveen, N. C. (2016). Predictive analysis of diabetes using J48 algorithm of classification techniques. In *Proceedings of the 2nd International Conference on Contemporary Computing and Informatics* (pp. 347–352). IEEE. doi:10.1109/IC3I.2016.7917987

Santhanam, T., & Padmavathi, M. S. (2015). Application of K-means and genetic algorithms for dimension reduction by integrating SVM for diabetes diagnosis. *Procedia Computer Science*, *47*, 76–83. doi:10.1016/j.procs.2015.03.185