JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 38, 761-774 (2022)
DOI: 10.6688/JISE.202207_38(4).0004

DCGAEL: An Optimized Ensemble Learning using a Discrete-Continuous Bi-Level Genetic Algorithm

MOHAMMAD AMIN ADIBI
Faculty of Industrial and Mechanical Engineering, Qazvin Branch
Islamic Azad University
Qazvin, 3419915195 Iran
E-mail: ma.adibi@iau.ac.ir

Ensemble learning encompasses methods that generate many well-diversified predictors and aggregates their results to perform a better prediction. These predictors are usually weak and low-cost for obtaining when they are alone. However, they reveal excellent performance when they are skillfully used together in the form of a learning architecture. Metaheuristic methods have been used to form such architecture optimally during recent years. Along this stream, in this paper, a bi-level optimization based on discrete-continuous genetic algorithm is utilized to enhance the performance of an ensemble learning meta-algorithm which benefits decision tree classification. Feature selection and tree model constructing for any ensemble member are done by the metaheuristic method. It allows us to have advantages of tree-based prediction models, ensemble learning, and solution optimality simultaneously. The proposed system is compared to some well-known ensemble learning methods. Results show significant superiority of the proposed system in terms of prediction accuracy.

Keywords: ensemble learning, genetic algorithm, bi-level optimization, feature selection, classification, decision tree

1. INTRODUCTION

Ensemble learning encompasses methods that generate many classifiers (ensemble members) and aggregates their results. It improves our confidence that we are making the right decision by weighing different opinions and combining them by means of some thought process to reach a final decision. However, if all ensemble members offer the same output, there is nothing to be gained from their combination. Therefore, we need diversity in the decisions of ensemble members. Diversity in ensemble members can be achieved through several strategies. These strategies are reported in [17] as using bootstrapped replicas of the training data (bagging), sampling from a distribution that regards previously misclassified samples (boosting), using different subsets of the available features to train each classifier (random subspace methods), using different parameters of the base classifier, and using different base classifiers as the ensemble members.

On the other hand, metaheuristics and evolutionary algorithms have been extensively used to enhance ensemble learning performance in recent years. For example, [5] used multi-objective evolutionary algorithms for the construction of neural ensembles. In [13], an ensemble classification scheme based on static classifier selection with majority voting error and the multi-objective differential evolution algorithm for sentiment analysis was presented. Based on the static classifier selection scheme, their proposed ensemble clas-

Received December 27, 2020; revised February 14, 2021; accepted March 7, 2021.
Communicated by Shyi-Ming Chen.

sification involves Bayesian logistic regression, the Naïve Bayes algorithm, linear discriminant analysis, logistic regression, and support vector machines. Authors in [28] used several evolutionary multi-objective algorithms to find sparse ensemble classifiers with a firm performance. Malhotra and Khanna in 2018 [12] used particle swarm optimization to determine proper weights for some ensemble classifiers. Improved particle swarm optimization and deep learning models was used by [22] to intelligent skin cancer diagnosis.

By reviewing the literature, it can be found that genetic algorithm (GA) has a crucial role in the enhancement of ensemble learning performance. In this way, many researchers have used GA to an optimal determination of the classifiers in an ensemble. In this case, several types of research can be mentioned. Also, a GA-based ensemble approach named GASEN for assembling a selective subset of individual artificial neural networks was proposed in [23]. In [8], authors used a GA to optimize an ensemble of some SVM classifier models. They used their proposed method to heart disease classification. Kim and Kang in 2012 [9] proposed a genetic algorithm-based coverage optimization technique to classifier selection in ensemble learning for bankruptcy prediction. A research published in [20] compared GA and simulated annealing performance in neural network ensembles. It used the metaheuristics to select the best subset among a group of neural network models. Yin *et al.* in 2014 [27] first proposed a mathematical framework of classifier ensemble with a sparsity and diversity learning strategy. Then, they used GA to solve the developed a mathematical programming to obtain the best ensemble of classifiers. Adnan and Islam in 2016 [3] used a GA in order to construct a small but accurate sub forest based on an initial decision forest, which is an ensemble of some decision trees. In [26] a GA was used to select appropriate classifiers to compose a final ensemble. In their work, the classifiers have been created based on SVM. Authors in [18] proposed a combination of sampling and GA to deal with multiclass imbalanced data through an ensemble learning architecture. GA was utilized in [15] and [16] for an ensemble learning structure to select appropriate combination of classifiers to solve credit scoring problem. They also utilized deep learning to enhance prediction accuracy via optimizing the parameters of modeling.

Some other researchers used GA to select a proper subset of features and the other related parameters to enhance ensemble learning performance. Researchers in [11] used a GA to generate and optimize a set of feature subsets on which the weak classifiers are constructed in Adaboost learning. They applied their proposed method to image annotation modeled as a classification problem. Rahman and Verma in 2013 [19] used GA to parameter optimization in a cluster oriented ensemble classifier generation method. To ensemble a new artificial neural network, Xue *et al.* in 2014 [24] used a GA. In their research, the main goal was constructing a network with optimal parameters. In [7], a multi-level approach using GA was applied in an ensemble of LS-SVM. Their proposed method aims to act on performing a feature selection, setting the parameters of the classifiers, and finding a weight vector that best represents the importance of each component in the set. Authors in [6] proposed a novel ensemble-based feature selection technique using a bi-objective GA with a dynamic mating pool. The algorithm produced the most precise and informative feature subsets, as rough set theory and information theory are used for defining objective functions of the proposed GA, which could efficiently classify the objects.

According to the literature, using GA as well as the other metaheuristics in ensemble learning is almost limited to find an optimal combination of classifiers each obtained from usual methods while deepening GA role for creating a predictive model based on ensemble

learning would lead to better-quality results. It is what we are looking for in this research.

As the maim contribution of this study, a new learning system is presented in which GA is employed to construct optimal classifiers to be used in learning as well as feature selection and sampling in the context of ensemble learning. Sampling, accurate feature selection, and optimal classifiers in an ensemble learning architecture make the proposed method more effective rather than conventional ensemble learning methods. We call this system as DCGAEL (Discrete-Continuous Genetic Algorithm Ensemble Learning).

The rest of the paper is organized as follows: In Section 2, the overall structure of the proposed method is explained. Discrete-continues GA is presented in Section 3, followed by experiments and results in Section 4, and the conclusion is given in Section 5.

2. THE PROPOSED ENSEMBLE LEARNING ARCHITECTURE

Here, the aim is having advantages of tree-based prediction models, ensemble learning, and solution optimality simultaneously in the form of a learning system. For this, diversification comes from selecting sub-samples of records and features concurrently to construct a lot of decision tree models as ensemble members. For each ensemble member, a discrete-continuous GA is used to perform two functions; Firstly, it helps to find a collection of features to be used in tree construction for each ensemble member. Secondly, the GA will be applied to find optimal tree construction instead of common heuristic procedures which lead to sub-optimal trees. In the proposed framework as presented in Fig. 1, K classifiers are produced by a two-level optimization using a discrete–continuous GA. To produce each of them, the two-level GA uses a subset of training data involving N samples. It also finds proper d features among initial d' features (details are presented in Subsection 3.2.1). In fact, the two-level GA is only allowed to construct a classifier tree with d features at most.

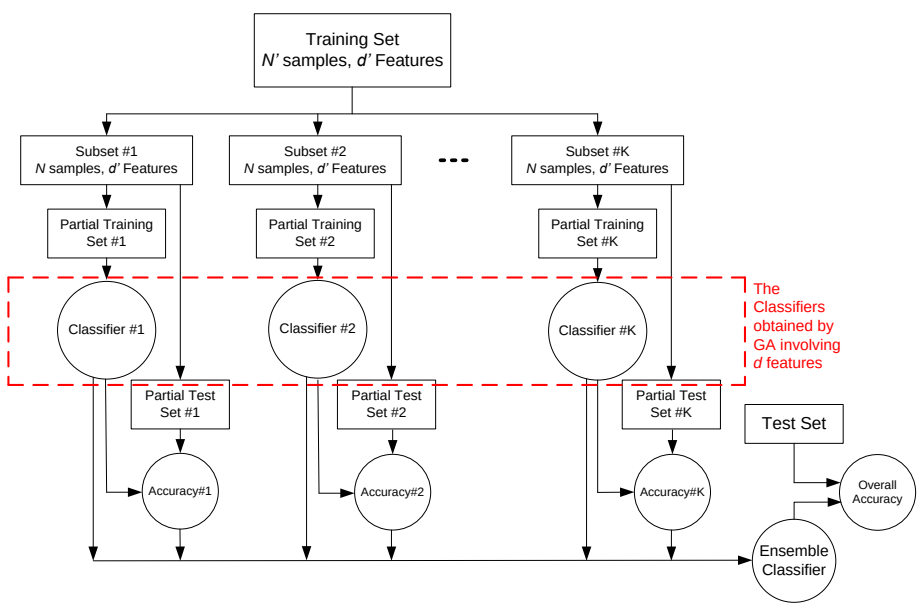

Fig. 1. The overall framework of the proposed method.

By producing K classifiers, the final decision will be made using a weighted voting schema. Each of K classifiers affects the final decision according to a predetermined accuracy. The accuracy for each classifier comes from testing the classifier on a partial test set derived from k^{th} (k = 1, 2, …, K) subset. For this purpose, Eq. (1) is used [2].

$$sample\_class = \arg\max_{c \in C} \left(\sum_{k=1}^{K} a_k \delta_{kc} \right) \tag{1}$$

$$\delta_{kc} = \begin{cases} 1 & \text{classifier } k \text{ puts the sample in class } c \\ 0 & o.w. \end{cases}. \tag{2}$$

Where C is the collection of classes, a_k is the accuracy of the classifier k, and δ_{kc} is defined as Eq. (2).

3. DISCRETE-CONTINUOUS GA

3.1 An Overview

The process of obtaining any of the ensemble members can be expressed in the form of two nesting loops. Each of the loops has a particular genetic algorithm while they have different duties. In fact, an outer loop identifies a proper subset of the features, while an inner loop constructs the best classification tree based on a solution of the outer loop. So, it provides a bi-level optimization in which optimal feature selection and optimal tree construction will be achieved concurrently. It should be noted that the bi-level optimization differs from sequential optimization in which optimal solution of a first problem is fed to a second problem to determine a second optimal solution the second solution. Instead, in the bi-level optimization, an optimal combination of features and tree construction will be occurred. However, in each optimization problem, a relatively small problem is solved. In the proposed bi-level optimization process, the fitness value of a solution (a subset of features) in the outer loop is the accuracy of the best constructed tree in the inner loop. The interaction of these two loops is shown in Fig. 2. The operators related to such a GA method were proposed in [1].

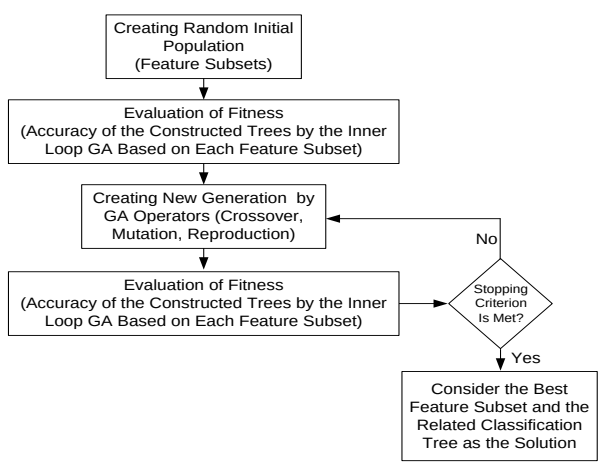

Fig. 2. The outer loop and inner loop interaction in the proposed method.

As shown in this figure, the solutions (each solution is a subset of features) evolve by the operators of the GA during generations. Once the stopping criterion is met, one ensemble member is obtained.

3.2 Outer Loop

3.2.1 Solution representation

The role of the outer loop is to identify the appropriate features to construct the tree in the inner loop. To do this, if there are d' features in the original data, a horizontal vector containing d integers selected at random from intervals 1 to d' are selected as a solution. In such an answer, the number ith ($i = 1, 2, \ldots, d$) represents the feature number that should be used in the tree construction. For instance, if there are 20 features in the original data set and only d features should be considered in the tree construction, the string 2 – 11 – 5 – 3 – 16 could be a solution indicating that the tree is built on features 2, 11, 5, 3, and 16.

3.2.2 Fitness evaluation

Each answer in the outer ring indicates the features that should be used in the tree construction. Consequently, by specifying the features, an optimal tree is constructed by the inner loop. The accuracy of the best solution created by the inner loop (as described in Subsection 3.3.2) is considered as the fitness value of the outer loop solution.

3.2.3 Crossover

Crossover, or recombination, is a genetic operator used to combine the genetic codes of two parents to generate new children in genetic algorithm. Although it generates new solutions from an existing population stochastically, the children should have specifications similar to their both parents. So, if you choose parents with better fitness, there will probably be a better new generation. To execute the crossover operator for the outer loop, two parents are first identified by a selection mechanism that is the Roulette Wheel [25] which gives higher probability to more fit solution to be selected. Then, an integer is randomly selected from interval [1, d]. This number indicates the crossover point. By replacing numbers from the crossover point to the end of the string between parents, two children are created. Note that this replacement occurs for any number when it does not result in duplicates in the new solutions. This operator for an example is shown in Fig. 3.

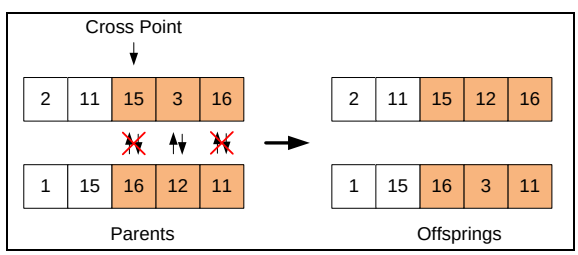

Fig. 3. The proposed crossover operator for outer loop, $d = 5$, $d' = 20$.

3.2.4 Mutation

The mutation operator is used as a mechanism for diversification in genetic algorithm. According to this, given a solution, the mutation operator in the outer loop starts by selecting an integer from interval [1, d] so called mutation point. Then, the current solution numbers from the mutation point to the end of the string are replaced by numbers randomly selected from interval [1, d'] which are not in the current solution. The mutation operator presented in Fig. 4 while it is performed for an example solution.

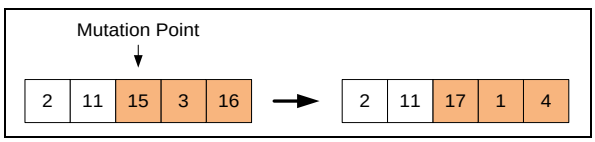

Fig. 4. The proposed mutation operator for outer loop and an example individual when $d = 5$, $d' = 20$.

3.3 Inner Loop

3.3.1 Solution representation

A $2\times(2^d - 1)$ matrix is used to represent the solution in the inner loop that is responsible for the tree model construction. The first row of this matrix consists of randomly selected numbers from interval [1, d]. Since the largest possible tree using d attributes has $2^d - 1$ branching nodes, these numbers and their order will indicate how the tree is to be constructed. In fact, in a tree, the branching feature number in the ith ($i = 1, 2, \ldots, 2^d - 1$) step is the ith number in this row. It is noteworthy that if a path from root node to a branching candidate node contains duplicated number, the node will be considered as leaf node and branching does not take place. For an example solution as $2-3-1-1-2-2-3$ (when $d = 3$), according to the order of numbers presented in the solution, branching is started by feature 2 as the root node. Then branching will be performed on features 3 and 1 in daughters of this node. This process continues until the final number 3 in the row. For this example, the structure of classification tree is as Fig. 5. In this case, no branching has been performed on 5th and 6th nodes because the pathways $2-3-2$ and $2-1-2$ contain the repeated number 2. Finally, the other leaf nodes also are added to the 3rd level branching nodes (4th and 7th nodes) in 4th level.

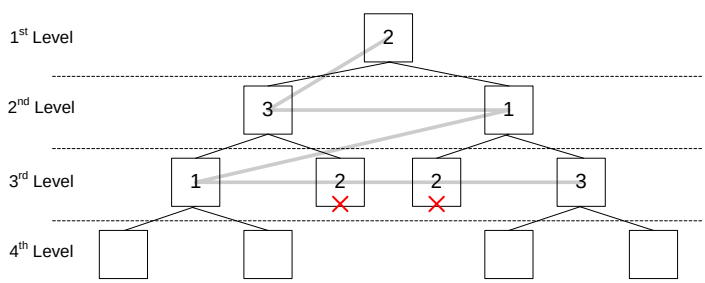

Fig. 5. Structure of the classification tree equivalent to 2-3-1-1-2-2-3 when $d = 3$.

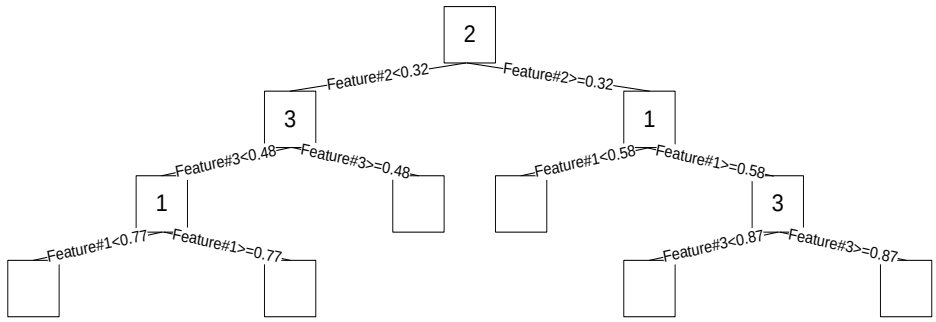

Fig. 6. A completed classification tree model based on the representation method in inner loop.

The second row of the matrix is used to determine the branching point. The second row of the matrix consists of $2^d - 1$ real number that range from 0 to 1. Based on the position of a real number in this string, the branching point for the feature in the same position is specified. In completing the recent example 0.32 – 0.48 – 0.58 – 0.77 – 0.68 – 0.25 – 0.87 can be considered as the second row to represent the branching points. Given the sequence of numbers in the second row and also the numbers in the first row, 0.32 is branching point of feature 2, 0.48 is branching point of feature 3, 0.58 is branching point of feature 1, and the same way for the rest of the numbers. The completed tree model represented by the complete matrix is presented in Fig. 6.

3.3.2 Fitness evaluation

The accuracy of the classification model associated with a solution for a test data set is considered as its fitness value. In fact, Eq. (3) defines accuracy for solution or individual i (i = 1, 2, …, *pop_size*) in which c and m are the number of correctly classified and the total number of instances respectively.

$$f_i = \frac{c}{m} \tag{1}$$

3.3.3 Crossover

For the inner loop, the Roulette Wheel is also used for parent selection. After the parents are selected, two random integers are assigned from 1 to $2^d - 1$. We consider these values as crossover points. Then, values between crossover points on the parents are replaced to create two children. This is done for both rows. However, relevant branching point in the second row will be modified to make the children closer to each other rather than their parents. It preserves the nature of crossover operator in a GA as explained in Subsection 3.2.3. To do this, considering x_i^1 and x_i^2 as branching points of parents 1 and 2 related to a node in which branching should be performed on feature number i, branching points of children 1 and 2, y_i^1 and y_i^2, are obtained using Eqs. (4) and (5). Here, M is a relatively big positive number. The proposed crossover operator is performed for two example solution, and it is presented in Fig. 7.

$$y_i^1 = x_i^2 + (x_i^1 - x_i^2)/M \tag{4}$$

$$y_i^2 = x_i^1 + (x_i^2 - x_i^1)/M \tag{5}$$

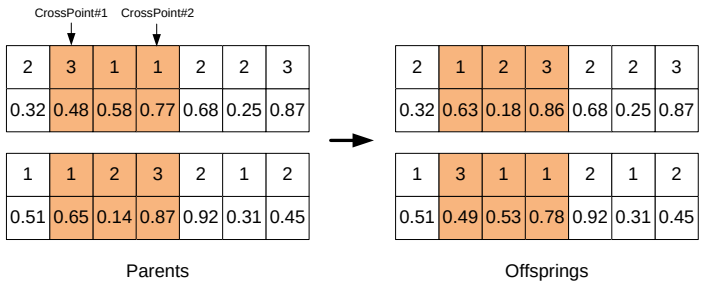

Fig. 7. Applying the proposed crossover operator for two example solution in inner loop, $M = 10$.

3.3.4 Mutation

For mutation operator in inner loop, an individual is selected randomly at first. Then, two mutation points are selected as integers randomly chosen from interval 1 to $2^d - 1$. For both rows in the selected individual, the order of the numbers between two mutation points is reversed. The proposed operator for an example individual is presented in Fig. 8.

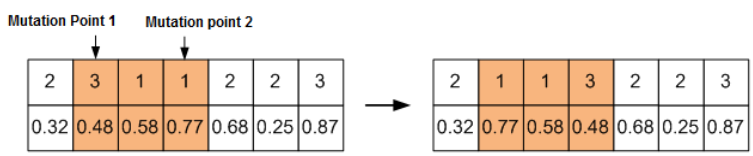

Fig. 8. Applying the mutation operator for inner loop and an example individual when $d = 3$.

4. EXPERIMENTS

4.1 Benchmark Datasets and Parameter Setting

The comparisons have been made for 10 datasets from the UCI repository which are the most common datasets being used by researchers to evaluate newly proposed methods. The characteristics of these datasets are given in Table 1. Please note that among these datasets "size" to "number of attributes" ratio varies from 0.4 (LSVT dataset) to 285 (Wifi dataset). So, the selected datasets can be regarded as a real challenge for any classification method. All datasets were randomly partitioned to train (75%) and test (25%) data.

For data preparation, note that input features for the all datasets are mapped to the interval [0,1] using $(x_{ij} - \min_i x_{ij}) / (\min_i x_{ij} - \min_i x_{ij})$. Each classification method has been performed 30 times over the datasets. Here, for each run, the selected parameters are as presented in Table 2. In both outer and inner loop, to ensure that the best of each generation is preserved, reproduction is used to move 10% of best individuals from one generation to the next generation.

Table 1. Characteristics of the ten UCI datasets used for the benchmarks.

No.	Abbreviation	Description	Attributes	Classes	Size
1	Liver	Liver Disorders	7	2	345
2	Ecoli	Ecoli	9	8	336
3	Breast	Breast Cancer Wisconsin	30	2	569
4	Sonar	Connectionist Bench	60	2	208
5	LSVT	LSVT Voice Rehabilitation Data Set	309	2	126
6	Wifi	Wireless Indoor Localization	7	4	2000
7	Parkinson	Parkinson	22	2	195
8	Colonoscopy	Colonoscopy	63	2	97
9	Glass	Glass	9	6	214
10	Climate	Climate Model Simulation Crashes	18	2	540

Table 2. The selected values for the parameters used in the experiments.

No.	Parameter	Value
1	*Number of ensemble members*	20
2	N	10 percent of N'
3	d	5
4	*Number of generations in the outer level*	10
5	*The outer level population size*	10
6	*The outer level crossover rate*	0.8
7	*The outer level mutation rate*	0.1
8	M	10
9	*Number of generations in the inner level*	50
10	*The inner level population size*	20
11	*The inner level crossover rate*	0.75
12	*The inner level mutation rate*	0.15

4.2 Assessment of Ensemble Architecture Effectiveness

The effectiveness of the proposed method in the case of the first five benchmark datasets namely Breast, Liver, Ecoli, Sonar, and LSVT is illustrated in Fig. 9. In this figure, the accuracy obtained by the proposed ensemble method is compared to maximum and average accuracy over the ensemble members. As we can see, ensemble schema has enhanced the performance of the group of classifiers. When we compare the performance of the method against the maximum performance of each member, we find that this method has been able to properly capture the spirit of ensemble learning over all five benchmark datasets.

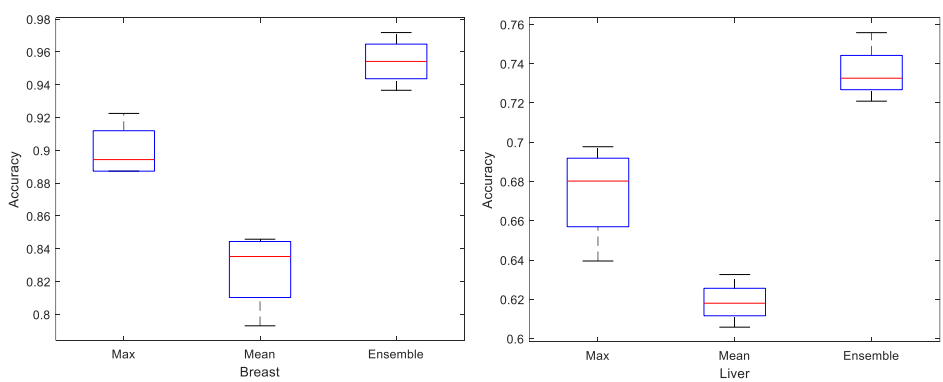

Fig. 9. Effectiveness of the proposed method in case of the five benchmark datasets, Max: performance of the best classifiers obtained in every run, Mean: average performance of the all classifiers, Ensemble: performance of the proposed ensemble method.

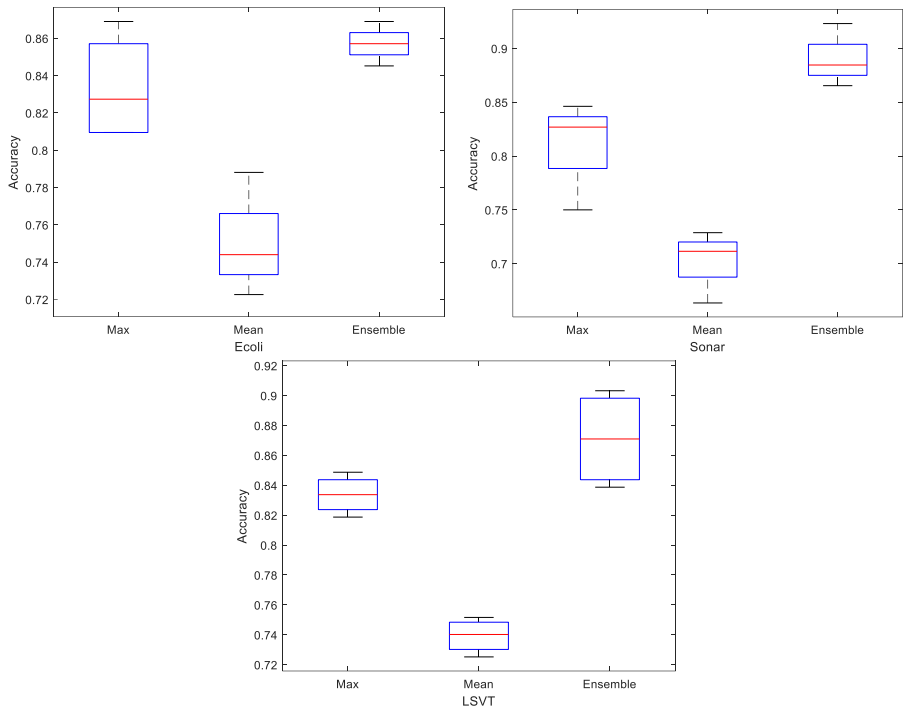

Fig. 9. (Cont'd) Effectiveness of the proposed method in case of the five benchmark datasets, Max: performance of the best classifiers obtained in every run, Mean: average performance of the all classifiers, Ensemble: performance of the proposed ensemble method.

4.3 Tree-Based Benchmark Ensemble Classification Methods

The proposed ensemble method classification was compared against two tree-based ensemble methods, ensemble CART and also Random Forest method both implemented in IBM SPSS Modeler 18. Ensemble Classification and Regression Tree was explained in [21] and Random Forest method in [14]. The comparisons have been made for the above mentioned datasets. Mean accuracy for each method and each dataset is reported in Table 3.

Table 3. Performance of the proposed method (DCGAEL) versus the tree-based benchmarks.

No.	Dataset	DCGAEL	Ensemble CART	Random Forest	Rank of DCGAEL	P-Value
1	Breast	**0.9630**	0.9436	0.9577	1	0.0401
2	Liver	**0.7354**	0.6860	0.7093	1	<0.0001
3	Ecoli	**0.8571**	0.8095	0.7619	1	<0.0001
4	Sonar	**0.8894**	0.7884	0.7692	1	<0.0001
5	LSVT	**0.8709**	0.7419	0.7419	1	<0.0001
6	Wifi	0.8880	0.8960	**0.9120**	3	0.0043
7	Parkinson	0.8888	0.8541	**0.9583**	2	<0.0001
8	Colonoscopy	**0.8437**	0.7916	0.7083	1	<0.0001
9	Glass	**0.7774**	0.7547	0.6792	1	<0.0001
10	Climate	**0.9259**	0.9037	0.9037	1	<0.0001
Mean	–	0.86396	0.81695	0.81015	–	–

Again, for each dataset, the proposed ensemble method has been applied 30 times. To survey statistically significance of the rank related to the proposed method, some statistical tests have also been performed. In the statistical test, H_0 has been defined as equality of the mean accuracy of the proposed method, and the performance of the nearest follower competitor. Each ensemble method was performed 30 times over the datasets and mean accuracy for each method and each dataset was also considered. Please note that by using IBM SPSS Modeler, there is a constant value for the competitors. As a result, the presented rank for the proposed method is completely reliable. Based on the results, it can be said that the ensemble learning method proposed in this paper has been significantly superior in the most cases. In fact, when considering the performance of the method in comparison with other tree-based ensemble learning methods, the significant advantage of the proposed method is considerable. In the recent comparison (Table 3), DCGAEL is superior to other methods in 80% of cases, and it is just the second or third in 20% of the cases.

4.4 Non-Tree-Based Benchmark Ensemble Classification Methods

The proposed ensemble method classification was also compared against two non-tree-based ensemble methods, including ensemble ANN and ensemble SVM methods both implemented in IBM SPSS Modeler 18. Fundamentals of ensemble ANN was presented in [10]. Ensemble SVM was explained in [4]. The comparisons have been made for the above mentioned-datasets again. Mean accuracy for each method and each dataset is reported in Table 4.

Table 4. Performance of the proposed method (DCGAEL) versus the non-tree-based benchmarks.

No.	Dataset	DCGAEL	Ensemble ANN	Ensemble SVM	Rank of DCGAEL	P-Value
1	Breast	0.9630	**0.9859**	0.9652	3	–
2	Liver	**0.7354**	0.7209	0.6562	1	0.0014
3	Ecoli	0.8571	0.8214	**0.8671**	2	< 0.0001
4	Sonar	**0.8894**	0.7500	0.8184	1	< 0.0001
5	LSVT	**0.8709**	0.7419	0.8441	1	< 0.0001
6	Wifi	0.8880	**0.8900**	0.8660	2	< 0.0001
7	Parkinson	0.8888	0.8750	**0.9018**	2	0.018
8	Colonoscopy	**0.8437**	0.7500	0.7841	1	< 0.0001
9	Glass	**0.7774**	0.7500	0.7281	1	0.0038
10	Climate	0.9259	0.9185	**0.9333**	2	0.0260
Mean	–	0.86396	0.82036	0.8364	–	–

Based on the results in Table 4 and comparison to the non-tree-based benchmarks, it can be said that DCGAEL totally performed better than ensemble ANN and ensemble SVM. For 10 datasets, the average accuracy is 0.8696 regarding DCGAEL while it is 0.8364 for ensemble SVM and only 0.82036 for ensemble ANN. The proposed method is better than ensemble ANN in 80% cases and better than ensemble SVM in 70% cases. This fact shows that DCGAEL enhances performance of a tree-based ensemble classifier accuracy to the level higher than ANN and SVM. So, the proposed learning architecture incorporating discrete-continuous GA is really efficient and it is ready to host other developments about new optimization methods.

5. CONCLUSIONS

In this paper, a learning system (DCGAEL) that utilized concepts of ensemble learning, decision tree, and solution optimality based on metaheuristics was presented. Diversity in the members was created by sampling records and attributes. Then, outputs were merged based on a weighted voting schema. Each member in the ensemble learning was a decision tree made by a discrete-continuous GA. The GA used two nested loops as a bi-level optimization method, and it was able to select the best features and to construct the optimal structure of a tree classification model. The developed learning system was used for several test datasets and was examined against ensemble ANN, ensemble SVM, ensemble CART, and Random Forest learning method. The results showed a significant advantage of the proposed method in this paper. According to the findings, extending the proposed idea to unsupervised learning like ensemble clustering can be regarded as future research. This is easy because clustering with the decision tree has already been developed. Furthermore, a basic form of genetic algorithm has been utilized in this paper while it can be combined with a local search method as a hybrid genetic algorithm to achieve even more accuracy.

REFERENCES

1. M. A. Adibi, "Single and multiple outputs decision tree classification using bi-level discrete-continues genetic algorithm," *Pattern Recognition Letters*, Vol. 128, 2019, pp. 190-196.
2. M. A. Adibi and J. Shahrabi, "Online anomaly detection based on support vector clustering," *International Journal of Computational Intelligence Systems*, Vol. 8, 2015, pp. 735-746.
3. M. N. Adnan and M. Z. Islam, "Optimizing the number of trees in a decision forest to discover a subforest with high ensemble accuracy using a genetic algorithm," *Knowledge-Based Systems*, Vol. 110, 2016, pp. 86-97.
4. C. Catal and M. Nangir, "A sentiment classification model based on multiple classifiers," *Applied Soft Computing*, Vol. 50, 2017, pp. 135-141.
5. A. Chandra and X. Yao, "Ensemble learning using multi-objective evolutionary algorithms," *Journal of Mathematical Modelling and Algorithms*, Vol. 5, 2006, pp. 417-445.
6. K. Das, S. Das, and A. Ghosh, "Ensemble feature selection using bi-objective genetic algorithm," *Knowledge-Based Systems*, Vol. 123, 2017, pp. 116-127.
7. C. A. de A. Padilha, D. A. C. Barone, and A. D. D. Neto, "A multi-level approach using genetic algorithms in an ensemble of least squares support vector machines," *Knowledge-Based Systems*, Vol. 106, 2016, pp. 85-95.
8. B. Fida, M. Nazir, N. Naveed, and S. Akram, "Heart disease classification ensemble optimization using genetic algorithm," in *Proceedings of IEEE 14th International Multitopic Conference*, 2011, pp. 19-24.
9. M. J. Kim, and D. K. Kang, "Classifiers selection in ensembles using genetic algorithms for bankruptcy prediction," *Expert Systems with Applications*, Vol. 39, 2012, pp. 9308-9314.

10. H. Li, X. Wang, and S. Ding, "Research and development of neural network ensembles: a survey," *Artificial Intelligence Review*, Vol. 49, 2018, pp. 455-479.
11. R. Li, J. Lu, Y. Zhang, and T. Zhao, "Dynamic Adaboost learning with feature selection based on parallel genetic algorithm for image annotation," *Knowledge-Based Systems*, Vol. 23, 2010, pp. 195-201.
12. R. Malhotra and M. Khanna, "Particle swarm optimization-based ensemble learning for software change prediction," *Information and Software Technology*, Vol. 102, 2018, pp. 65-84.
13. A. Onan, S. Korukoğlu, and H. Bulut, "A multiobjective weighted voting ensemble classifier based on differential evolution algorithm for text sentiment classification," *Expert Systems with Applications*, Vol. 62, 2016, pp. 1-16.
14. A. Parmar, R. Katariya, and V. Patel, "A review on random forest: An ensemble classifier," in *Proceedings of International Conference on Intelligent Data Communication Technologies and Internet of Things*, 2018, pp. 758-763.
15. P. Pławiak, M. Abdar, and U. R. Acharya, "Application of new deep genetic cascade ensemble of SVM classifiers to predict the Australian credit scoring," *Applied Soft Computing*, Vol. 84, 2019, pp. 105740.
16. P. Pławiak, M. Abdar, J. Pławiak, V. Makarenkov, and U. R. Acharya, "DGHNL: A new deep genetic hierarchical network of learners for prediction of credit scoring," *Information Sciences*, Vol. 516, 2020, pp. 401-418.
17. R. Polikar, "Ensemble learning," *Ensemble Machine Learning*, Springer, 2012, pp. 1-34.
18. A. Purwar and S. K. Singh, "A novel ensemble classifier by combining sampling and genetic algorithm to combat multiclass imbalanced problems," *International Journal of Data Analysis Techniques and Strategies*, Vol. 12, 2020, pp. 30-42.
19. A. Rahman and B. Verma, "Ensemble classifier generation using non-uniform layered clustering and genetic algorithm," *Knowledge-Based Systems*, Vol. 43, 2013, pp. 30-42.
20. S. Soares, C. H. Antunes, and R. Araújo, "Comparison of a genetic algorithm and simulated annealing for automatic neural network ensemble development," *Neurocomputing*, Vol. 121, 2013, pp. 498-511.
21. C. Strobl, J. Malley, and G. Tutz, "An introduction to recursive partitioning: rationale, application, and characteristics of classification and regression trees, bagging, and random forests," *Psychological Methods*, Vol. 14, 2009, pp. 323-348.
22. T. Y. Tan, L. Zhang, and C. P. Lim, "Intelligent skin cancer diagnosis using improved particle swarm optimization and deep learning models," *Applied Soft Computing*, Vol. 84, 2019, p. 105725.
23. Z. H. Zhou, J. X. Wu, Y. Jiang, and S. F. Chen, "Genetic algorithm based selective neural network ensemble," in *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, Vol. 2, 2001, pp. 797-802.
24. X. Xue, M. Yao, Z. Wu, and J. Yang, "Genetic ensemble of extreme learning machine," *Neurocomputing*, Vol. 129, 2014, pp. 175-184.
25. S. L. Yadav and A. Sohal, "Comparative study of different selection techniques in genetic algorithm," *International Journal of Engineering, Science and Mathematics*, Vol. 6, 2017, pp. 174-179.

26. R. Ye, Q. Dai, and M. L. Li, "A hybrid transfer learning algorithm incorporating TrSVM with GASEN," *Pattern Recognition*, Vol. 92, 2019, pp. 192-202.
27. X. C. Yin, K. Huang, H. W. Hao, K. Iqbal, and Z. B. Wang, "A novel classifier ensemble method with sparsity and diversity," *Neurocomputing*, Vol. 134, 2014, pp. 214-221.
28. J. Zhao, L. Jiao, S. Xia, V. B. Fernandes, I. Yevseyeva, Y. Zhou, and M. T. Emmerich, "Multiobjective sparse ensemble learning by means of evolutionary algorithms," *Decision Support Systems*, Vol. 111, 2018, pp. 86-100.

Mohammad Amin Adibi received the Ph.D. degree in Industrial Engineering at Amirkabir University of Technology, Tehran, Iran. His research interests include machine learning, artificial intelligence, data mining, and their applications.