Interval-based Cost-sensitive Classification Tree Induction as a Bi-level Optimization Problem

Rihab Said[1], Maha Elarbi[1], Slim Bechikh[1], Carlos A. Coello Coello[2], Lamjed Ben Said[1]
[1] SMART Lab, ISG, University of Tunis, Tunisia
[2] CINVESTAV-IPN, Mexico
rihabsaid.edu@gmail.com

***Abstract*—Cost-sensitive learning is one of the most adopted approaches to deal with data imbalance in classification. Unfortunately, the manual definition of misclassification costs is still a very complicated task, especially with the lack of domain knowledge. To deal with the issue of costs' uncertainty, some researchers proposed the use of intervals instead of scalar values. This way, each cost would be delimited by two bounds. Nevertheless, the definition of these bounds remains as a very complicated and challenging task. Recently, some researches proposed the use of genetic programming to simultaneously build classification trees and search for optimal costs' bounds. As for any classification tree there is a whole search space of costs' bounds, we propose in this paper a bi-level evolutionary approach for interval-based cost-sensitive classification tree induction where the trees are constructed at the upper level while misclassification costs intervals bounds are optimized at the lower level. This ensures not only a precise evaluation of each tree but also an effective approximation of optimal costs intervals bounds. The performance and merits of our proposal are shown through a detailed comparative experimental study on commonly used imbalanced benchmark data sets with respect to several existing works.**

***Index Terms*—Cost-sensitive learning, misclassification costs' intervals, classification tree induction, bi-level optimization, evolutionary algorithms.**

I. Introduction

Several real-world classification applications encouter the class imbalance problem. This fact leads to biased classifiers that achieve low accuracy on the minority class with high accuracy on the majority one. These biased classifiers cannot be decisive for applications such as bioinformatics and medical diagnosis [1]. This is because the minority class is more important than the majority one in several applications. Cost sensitive learning is an important technique that is used for the improvement of classifiers to make them sensitive to several misclassification costs [2], [3]. However, existing cost sensitive methods share the same shortcoming that consists in manually designing cost matrices. Unfortunately, it cannot be easy for humains to correcly specify misclassification costs for several mistakes.

Cost intervals are easier to be provided by experts because they can provide information about the most serious type of mistakes. Cost information encompasses cost values, interval of costs, and costs distribution [4]. The latter ones are usually unknown. This way, cost sensitive approaches that are based on a manual design for cost matrices cannot be applied. For this reason, the main goal consists in searching for methods that are able to simultaneously learn costs in order to construct cost-sensitive classifiers.

Genetic Programming (GP) could be applied for the classification task as an evolutionary algorithm that is able to automatically evolve solutions [5]. Indeed, a tree or several generated trees could represent a classifier. The main goal consists in selecting informative features in an automatic way. However, the generated classifiers by a GP method could be biased due to the disproportionate class distribution.

Recently, researchers have investigated GP with cost sensitive learning in order to simultaneously build classification trees and search for optimal costs' bounds [6], [7]. In fact, by the use of intervals, possible mistakes could be tolerated in the decision making process due to the uncertainty consideration [4], [7]. In spite of obtaining promising results, proposed approaches have a single level model that does not optimize interval-based cost matrices [7]. However, there is a whole search space of costs' bounds for any classification tree. In other words, the evaluation of a classifier necessitates trying several interval-based cost values in order to be more fair and precise. Motivated by this observation, we propose a bi-level modeling in which the trees are constructed at the upper level while misclassification costs intervals bounds are optimized at the lower level. The proposed model is solved using an enhanced version of an existing co-evolutionary algorithm called CEMBA [8]. The resulting approach is named Bi-ICOS (Bi-level Interval-based COst Sensitive). The main contributions of the paper are the following:

- Proposing a bi-level modeling of the interval-based cost-sensitive classification tree induction problem that evolves classification trees at the upper level and optimizes the misclassification costs intervals bounds for each classifier at the lower level. In fact, the classification tree is passed as a fixed parameter to the lower level and then a whole lower level evolutionary process is executed to approximate the optimal misclassification costs intervals bounds of the considered classifier. Compared to single level modeling, the bi-level one ensures a more precise and fair evaluation of the evolved classification tree;
- Designing an improved and enhanced version of an existing co-evolutionary algorithm in order to solve the proposed bi-level model by modifying the migration strategy. This latter one ensures both efficacious variation and diversification of the evolved classification trees.

- Assessing the performance of Bi-ICOS on ten commonly-used imbalanced datasets with up to 336 instances and 12600 features. The obtained results illustrate the ability of Bi-ICOS in outperforming recent cost sensitive GP methods and GP-based methods in terms of the evolved costs intervals bounds and the Area Under a Curve (AUC) results.

II. Backround and Previous Related Work

When the imbalanced data classification task suffers from high dimensionality, it is very difficult to improve the accuracy of both minority and majority classes by selecting good-quality features. Indeed, biased classifiers could be constructed based on the use of features that are biased towards the majority class.

Several real-world applications confront different mistakes that lead to different losses. For instance, in medical analysis, the fact of classifying a cancer patient as a healthy one is a more serious mistake than the fact of classifying a healthy patient as a cancer person. One of the important methods that have been successfully used for imbalanced data classification is cost sensitive learning. Indeed, cost values are considered in order to treat the possible mistakes. The main goal is to minimize the total cost of a cost-sensitive classifier. Cost sensitive approaches are based on a cost matrix in order to indicate the possible cost of misclassification. It is important to mention here that there are two types of misclassification costs: (1) instance-dependent cost values (i.e., every used instance has its own cost) and (2) class-dependent cost (i.e., different classes have different misclassification costs).

Bahnsen et al. [9] proposed a framework based on example-dependent cost-sensitive trees. The proposed framework creates several decision trees using random sub-samples from the training set. Moreover, two cost-sensitive combination approaches were proposed where the first approach is a cost-sensitive weighted voting and the second one is a cost-sensitive stacking approach. In [10], authors tackled face recognition by proposing a cost-sensitive kernel logistic regression with cost-sensitive approach based on the k-nearest neighbor. Zhou et al. [11] studied the fact of considering cost-sensitive learning for the multi-class classification case. Indeed, these authors proposed examining costs consistency before utilizing the re-scaling in order to improve the performance of classifiers. When the cost information is unavailable [12], reseachers utilize the class imbalance ratio in order to construct the cost matrix. This approach is oversimplified and does not consider the characteristics of data. In other works [13] [14], the cost matrix is optimized in order to ensure the construction of cost sensitive classifiers in case of unknown cost information.

The investigation of cost sensitive learning and GP has gained the attention of researchers. For instance, Li et al. [15] investigate the performance of GP in order to tackle cost sensitive classification. The main idea consists in manipulating the training data while ensuring the modification of the learning algorithm. The resulting approach is called CGP (Constrained Genetic Programming) that builds decision trees

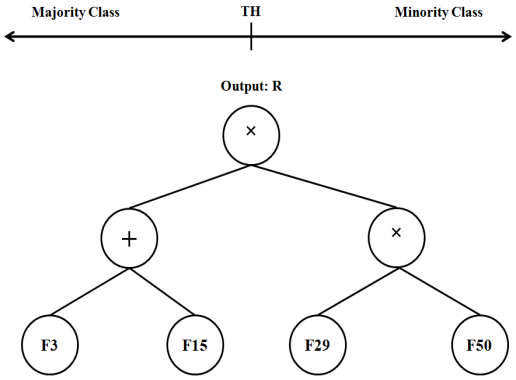

Fig. 1: Example illustrating how GP is utilized for the classification task (inspired by [7]).

while minimizing misclassification costs and errors through the use of a novel fitness function. However, the cost information is provided by domain experts. Recently, Pei et al. [6] propose the use of the class imbalance ratio in order to automatically learn costs without the need of obtaining the cost matrix information from domain experts. Indeed, two cost-sensitive methods based on GP have been introduced. On the one hand, the first method incorporates the learned cost values into the GP fitness function. On the other hand, the second proposed method integrates the cost values into the GP classification process based on the use of a three-way decision idea. More recently, Pei et al. [7] proposed a cost-sensitive genetic programming approach that is able to automatically learn interval costs. These latter ones are utilized by the GP constructed classifiers to make them sensitive to several mistakes.

In cost sensitive learning, the optimal classification predictions are taken based on the cost matrix. Indeed, the minority and the majority class are given by Class 0 (positive) and Class 1 (negative), respectively. In this way, the class-dependent cost matrix could be given as follows:

$$Matrix = \begin{pmatrix} C_{00} & C_{01} \\ C_{10} & C_{11} \end{pmatrix}$$

where C_{01} and C_{10} represent costs of a false positive and a false negative, respectively. C_{00} is the true positive cost while C_{11} denotes the true negative cost. It is important to mention that $C_{01} < C_{10}$, $C_{01} > C_{11}$, and $C_{10} > C_{00}$ [16]. In this way, the prediction of an instance noted *h* into a class noted *a* necessitates finding the lowest expected cost [16]. In fact, when *h* is predicted into *a*, the expected cost *R* is calculated as follows:

$$R(h, a) = \sum_{b} P(b|h) C_{ab} \tag{1}$$

where $P(b|h)$ represents the probability of instance *h* belonging to class b. It is worth mentioning that if b represents the true class label, then the predicting cost of instance h into a is given by C_{ab}. Concerning the correct prediction, it occurs when $a = b$. Conversely, the incorrect prediction occurs when

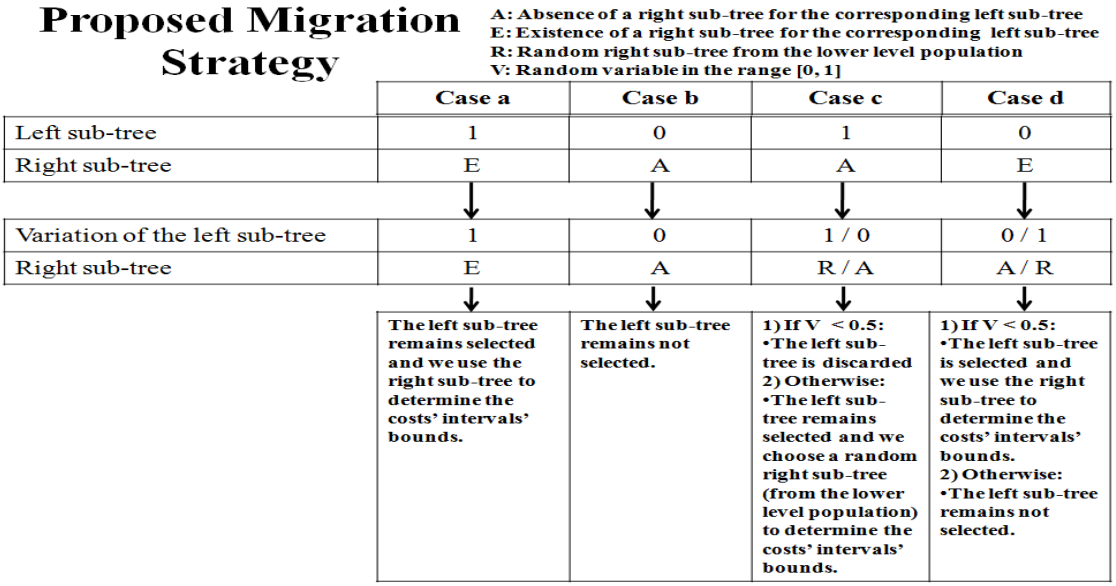

Fig. 2: Illustration of the new migration strategy.

$i \neq j$. Based on the previous equation, the possible costs of classifying instance h into Class 1 or Class 0 are given as follows:

$$R(h,1) = P(0|h)C_{10} + P(1|h)C_{11} \quad (2)$$

$$R(h,0) = P(0|h)C_{00} + P(1|h)C_{01} \quad (3)$$

Consequently, we can say that instance h is predicted to Class 0 when $P(1|h) \geq \frac{C10-C00}{C10-C00+C01-C11}$; however, h is predicted to Class 1 otherwise [16]. Concerning the optimal prediction for h, it is the Class 1 when $R(h,1) \leq R(h,0)$. In order to separate the two classes, Pei et al. [6] introduce the classification threshold as follows:

$$TH = \frac{C_{10} - C_{00}}{C_{10} - C_{00} + C_{01} - C_{11}} \quad (4)$$

that is simplified to [6]:

$$TH = \frac{C}{C+1} \quad (5)$$

because both C_{00} and C_{11} are equal to 0 in order to indicate that there is no misclassification cost that could be caused by correct predictions. Moreover, the false negative cost C_{10} is set to a value (C) greater or equal to 1 while C_{01} (the cost of a false positive) is equal to 1.

III. Proposed approach Bi-ICOS

A. Motivations and main idea

The use of GP was the choice of some researchers to tackle the imbalanced classification. Indeed, an individual is represented by a GP tree based on terminal and function sets. Internal nodes of a tree are constructed using the function set (i.e. operators or functions), while terminals could be the dataset features. To effectively make GP work on imbalanced data classification, Pei et al. [6] introduced the threshold moving idea. As illustrated by Fig. 1, a GP classifier is constructed using $+$ and $\times$ as operators (taken from the adopted function set) and four features: $F3$, $F15$, $F29$, and $F50$ (taken from the terminal set). Each adopted instance is input into the following expression: $(F3 + F15) \times (F29 \times F50)$ and a result value R is generated as the output. In the following, the adopted instance is classified into the majority class if the obtained value R is less than TH (threshold); otherwise, it is classified into the minority class. Recently, an Interval-based Cost-Sensitive Genetic Programming approach (ICS-GP) was proposed by Pei et al. [7] in order to tackle the high-dimensional imbalanced data classification. Indeed, the Strongly Typed GP (STGP) was used to develop a tree where the left sub-tree performs the classifier construction and the right sub-tree performs the cost interval learning. The main goal was to automatically develop classifiers while learning cost intervals. In fact, for each individual, the right sub-tree generates a cost interval that will be utilized by the left sub-tree (i.e., the classifier) in order to evaluate classifiers and to make them sensitive to possible classification errors. This approach has obtained good results beacuse it is based on the fact of automatically learning cost intervals that are needed for the cost sensitive classifiers construction. Unfortunately, the classifier evaluation is not fair and not precise because for each left sub-tree (i.e., classifier), a single right sub-tree (i.e. cost interval) is generated. Indeed, to make the classifier evaluation more precise, it is required to try several cost intervals and then to determine the best generated one that will be used in the evaluation step. In other words, there is a whole search space of costs' bounds for any classification tree. Motivated by this observation, we propose in this paper, a bi-level modeling that performs the trees construction at the upper level while

misclassification costs intervals bounds are optimized at the lower level. By following this bi-level model, a population of right sub-trees (i.e., several cost intervals) are generated, for each clssification tree, at the lower level problem and the best cost interval will be passed to the upper level in order to terminate the classifier evaluation. This fact ensures not only a precise evaluation of each constructed tree but also an effective approximation of optimal costs intervals bounds.

In order to solve the resulting bi-level modeling, we have designed an enhanced version of an existing co-evolutionary algorithm named CEMBA (Co-Evolutionary Migration-Based Algorithm) [8] by modifying its migration strategy in order to ensure efficacious diversification and variation of classification trees. CEMBA [8] is based on decomposition and migration schemes that make it efficient and effective in reducing the number of evaluations while obtaining good upper level and lower level results. To ensure efficacious diversification, we propose a new migration strategy that is explained in Fig. 2. Indeed, we are manipulating four cases as follows. For case (a), the left sub-tree remains selected when it has a right sub-tree at the lower level, and for case (b), the left sub-tree remains discarded when it is discarded at the upper level and its corresponding right sub-tree does not exist at the lower level. Concerning case (c), it occurs when there is a left sub-tree at the upper level while its right sub-tree does not exist at the lower level. In this way, a random variable denoted as V is generated between 0 and 1. Consequently, if $V < 0.5$, then the left sub-tree will be discarded; otherwise, it remains selected at the upper level and a random right sub-tree will be chosen from the lower level population. Case (d) occurs when the left sub-tree is discarded at the upper level while its corresponding right sub-tree exists at the lower level. In this case, if the random variable $V < 0.5$, then this left sub-tree will be selected at the upper level and the right sub-tree will be used to determine cost interval information; otherwise, both left and right sub-trees will be discarded.

B. Detailed description

1) Upper level (left sub-tree):

The upper level of the proposed approach performs the classifier construction by generating the GP left sub-tree. Terminal and function sets are given by Table I. Fig. 3 illustrates an example of an evolved tree. The left sub-tree in Fig. 3 is utilized as a classifier, that will be transformed to an arithmetic expression Classifier $((F15 + F25) + F59)$. In the previous expression, the adopted features ($F15$, $F25$, and $F59$) are taken from the terminal set while $+$ is an operator chosen from the function set. The output value of the arithmetic expression is normalized into the range [0,1] using the min-max normalization method as follows [7]:

$$pre_a = 1 - \frac{OUTV_a - min(OL)}{max(OL) - min(OL)} \tag{6}$$

where $OUTV_a$ represents the value of the left sub-tree output taking the instance a as an input. The $OUTV_a$ list for all the training instances is given by OL, the minimum value in OL

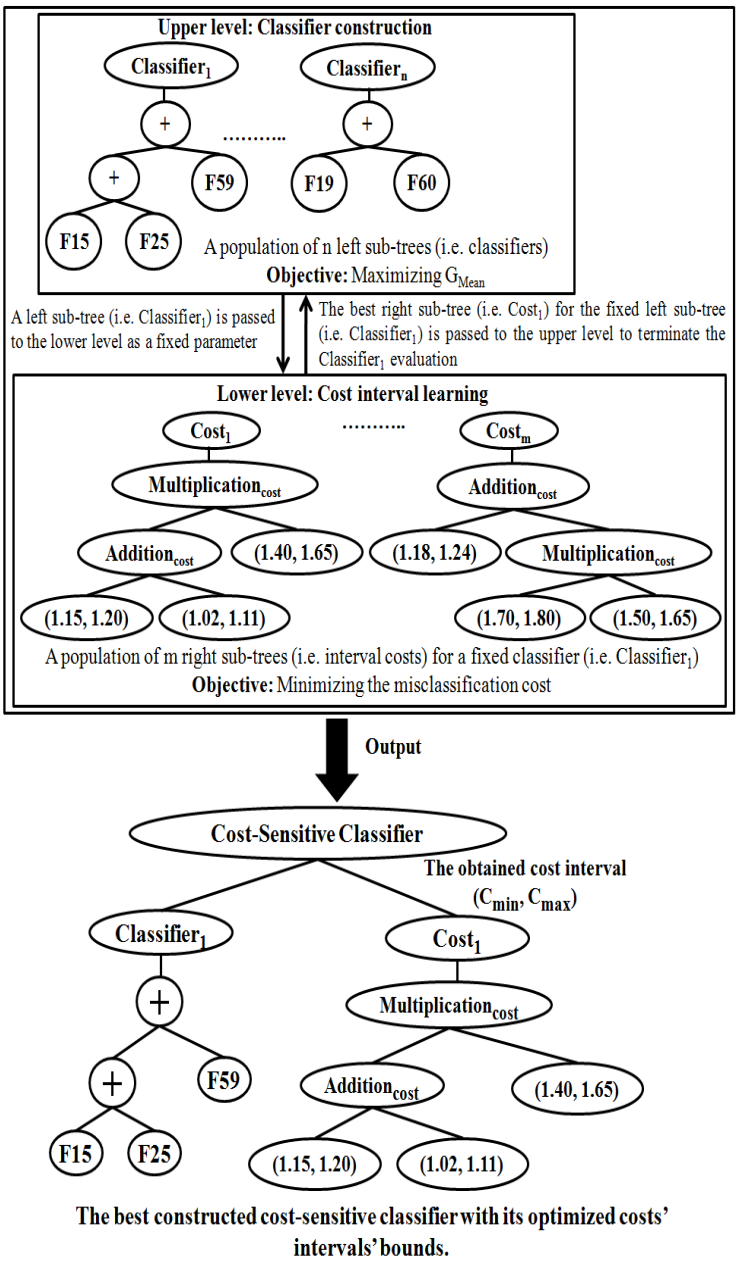

Fig. 3: Example illustrating the main idea of our proposed approach, Bi-ICOS.

is given by $min(OL)$, and the maximum one is represented by $max(OL)$. After that, based on the received cost interval values (i.e., C_{min} and C_{max}) from the lower level, the upper level computes the classification thresholds as follows [7]:

$$Threshold_1 = \frac{C_{max}}{C_{max} + 1} \tag{7}$$

$$Threshold_2 = \frac{C_{middle}}{C_{middle} + 1} \tag{8}$$

After that, the constructed classifier ensures the prediction for the majority and the minority classes based on the two

TABLE I: The used terminal and function sets.

	Left sub-tree	Right sub-tree
Terminal set	Random constant Features taken from the used dataset	Initial cost interval values [C_{min}, C_{max}] (the adopted values are uniformally distributed in [1, 2])
Function set	+, -, ×, %, if, and Classifier	$Addition_{cost}$: C_{max} = C_{max1} + C_{max2}, C_{min} = C_{min1} + C_{min2}, $multiplication_{cost}$: C_{max} = $C_{max1} \times C_{max2}$, C_{min} = $C_{min1} \times C_{min2}$, $Division_{cost}$:if $\frac{C_{max1}}{C_{max2}} > 1$, then $\frac{C_{max1}}{C_{max2}}$; otherwise, $\frac{C_{max2}}{C_{max1}}$ (the same rule for C_{min}), $substruction_{cost}$: C_{max} = max(1, C_{max1} - C_{max2}), C_{min} = max(1, C_{min1} - C_{min2}), and $Cost$

classification thresholds.

a) The training set classification predictions:

On the one hand, when using $Threshold_1$, the instance a is classified into the majority class if $pre_a \geq Threshold_1$; otherwise, a is classified into the minority class. On the other hand, when using $Threshold_2$, the instance a is classified into the majority class if $pre_a \geq Threshold_2$; otherwise, a is classified into the minority class.

b) Evaluation:

Each upper level individual (i.e., a tree) is evaluated using the GMean [14]:

$$GMean = \sqrt{\frac{TP}{TP+FP} \times \frac{TN}{TN+FP}} \tag{9}$$

where FP denotes false positive, TP represents true positive, while TN and FN are true negative and false negative, respectively. In fact, the GMean is computed for each threshold (i.e., $Threshold_1$ and $Threshold_2$): (1) GMean value using $Threshold_1$ ($GMT1$) and GMean value using $Threshold_2$ ($GMT2$). Consequently, the fitness function combines the two GMean values as follows:

$$GMean = GMT1 + GMT2 \tag{10}$$

c) Variation:

Based on the obtained fitness values, best individuals are selected by the tournament selection. After that, the genetic operators (i.e., sub-tree crossover, sub-tree mutation [5], and elitism) are applied in order to generate new populations until reaching a stopping criterion. Finally, the best individual (i.e., tree) is selected in order to ensure the predictions of classification on the test set.

d) The test set classification predictions:

Additionally to $Threshold_1$ and $Threshold_2$, a new $Threshold_3$ is also utilized.

$$Threshold_3 = \frac{C_{min}}{C_{min}+1} \tag{11}$$

The main goal is to ensure the classification decisions for unseen instances [7]. Indeed, the instance a is classified into the majority class if $pre_a \geq Threshold_1$; however a is classified into the minority class if $pre_a \leq Threshold_3$. Moreover, when pre_a belongs to the range [$Threshold_3$, $Threshold_1$], then, the decision is taken based on $Threshold_2$ and pre_a. as follows. Suppose that $max(Pro_{minority})$ is the maximum value in the probabilities list of instances that were predicted to the minority class and $min(Pro_{majority})$ is the minimum value in the probabilities list of instances that were predicted to the majority class. When $pre \geq Threshold_2$, if pre_a is nearer to $max(Pro_{minority})$ than $min(Pro_{majority})$, a is classified to the minority class; otherwise a is classified to the majority class.

2) Lower level (right sub-tree):

The lower level ensures the cost interval learning task. Indeed, several right sub-trees are generated at the lower level for each classifier (i.e., classification tree). Terminal and function sets are given by Table I. The right sub-tree in Fig. 3 generates three cost intervals: (1.15, 1.20), (1.02, 1.11), and (1.40, 1.65). Each interval is represented by (C_{min}, C_{max}) where both C_{min} and C_{max} are obtained through the use of the uniformly distributed random numbers between 1 and 2. As given by Fig. 3, the right sub-tree (i.e. $Cost_1$) is transformed to: Cost($multiplication_{cost}$($addition_{cost}$((1.15, 1.20), (1.02, 1.11))), (1.40, 1.65)). This way, the obtained cost interval is Cost($multiplication_{cost}$((2.17, 2.31), (1.40, 1.65)) = (3.03, 3.81) = (C_{min}, C_{max}).

a) Evaluation:

Each lower level individual (i.e., a right sub-tree) evolves a cost interval. Since the main goal is to minimize the misclassification cost, the best cost interval is the one having the minimum [C_{min}, C_{max}] values.

b) Variation:

Based on the obtained cost interval, best lower level individuals are selected by the tournament selection. After that, the genetic operators (i.e., sub-tree crossover, sub-tree mutation [5], and elitism) are applied in order to generate new lower level populations until reaching a termination criterion. In fact, the best individual (i.e., right sub-tree) is passed to the upper level in order to evaluate the corresponding classification tree.

IV. Experimental study

A. Used datasets

In order to examine the proposed Bi-ICOS, we have used ten datasets (gene expression datasets) in the experiments [17] (cf. https://schlieplab.org/Static/Supplements/CompCancer/datasets.htm and https://sci2s.ugr.es/keel/imbalanced.php). As illustrated by Table II, the adopted datasets may encounter the class imbalance issue with the high-dimensionality aspect. We mention here that IR represents the class imbalance ratio ($\frac{\#maj}{\#min}$, where $\#maj$ is used to denote the instances number in the majority class while $\#min$ is utilized to present the instances number in the minority class).

TABLE II: Datasets details.

Dataset	#Instances	#Features	*IR (Approximately)*
Lung	156	12,600	8
Tomlins-2006-v1	104	2,315	8
Yeoh-2002-v1	248	2,526	5
Gordon-2002	181	1,626	5
DLBCL	77	5,469	3
Shipp-2002-v1	77	798	3
Leukemia	72	7,129	2
Colon	62	2,000	2
Golub-1999-v1	72	1,868	2
Armstrong-2002-v1	72	1,081	2

TABLE III: Default parameters settings for Bi-ICOS.

Upper level Population (UP) size	UP_1 = 25, UP_2 = 25
Lower level Population (LP) size	LP_1 = 25, LP_2 = 25
Upper level generations number	25
Lower level generations number	25
Stopping criterion	781250 evaluations
Initialization	Ramped half-and-half
Crossover type and probability	Sub-tree crossover with a probability of 0.8
Mutation type and probability	Sub-tree mutation with a probability of 0.2
Selection	Tournament selection with a size equal to 6
Maximum tree depth	10
Elitism	1

B. Baseline methods

The proposed Bi-ICOS was compared to a recent interval-based cost sensitive approach (i.e., ICS-GP [7]), GP-based sampling methods (i.e., GP_{SMOTE} [18] and GP_{ADASYN} [19]), and GP approaches that apply several fitness functions (i.e., GP_{ave} based on the weighted-average classification accuracy [20], GP_{G-mean} based on G-mean [14], and GP_{aucw} based on Wilcoxon-Mann-Whitney (aucw) [21]).

C. Parameters settings and statistical methodology

In order to evaluate our proposed Bi-ICOS, we have used the trial-and-error method [22] for tuning the parameters (cf. Table III). For the other compared algorithms (i.e., ICS-GP, GP_{SMOTE}, GP_{ADASYN}, GP_{ave}, GP_{G-mean}, and GP_{aucw}), we adopted the parameters settings of the following paper [7]. Moreover, we utilized the number of evaluations as a termination criterion which is set to 781,250 evaluations. Furthermore, we have launched 30 runs. Concerning the statistical test, the Friedman statistical test was adopted followed by a posthoc analysis based on the Holm test because we are performing multiple comparisons. Indeed, the first one is used to detect if an algorithm is statistically different from the other algorithms or not, while the second one adjusts the p-values in order to determine the pairwise relationships [23]. It is important to mention that three statistical symbols are used: (1) "+" (better), (2) "-" (worse), and (3) "$\approx$" (no significance).

D. Obtained results and discussions

The Area Under a Curve (AUC) is used in order to compare the proposed Bi-ICOS with ICS-GP, GP_{SMOTE}, GP_{ADASYN}, GP_{ave}, GP_{G-mean}, and GP_{aucw}. AUC is a widely used performance metric in imbalanced data classification that evaluates both true positive and false positive rates many times while ensuring the variation of thresholds in order to provide an accurate curve rendition [1]. The obtained best, median, and standard (std) results of Bi-ICOS with its peer algorithms is given by Table IV. By comparing the obtained AUC results among all runs, our proposed Bi-ICOS achieves the best performance than other used methods in 8 out of the 10 datasets.

Compared with ICS-GP, our proposed Bi-ICOS achieves better results and the superiority of Bi-COS appears on the Leukemia dataset in which Bi-ICOS achieves 7.57% higher AUC. Indeed, both ICS-GP and Bi-ICOS do not require determining a pre-defined classification threshold in order to ensure the classification. However, two classification thresholds are computed by using the evolved cost interval values. In this way, both algorithms will be able to predict both majority and minority classes. The main difference between Bi-ICOS and ICS-GP reveals in the fact that our proposed Bi-ICOS is based on a bi-level modeling in which the misclassification costs intervals bounds are optimized at the lower level. In this way, Bi-ICOS is able to ensure not only a precise evaluation of each generated tree but also an effective approximation of optimal misclassification costs intervals bounds.

Compared with GP_{ave}, our proposed Bi-ICOS obtains significantly better performance in terms of the median AUC in all datasets. Indeed, it is observed from the table that Bi-ICOS achieves higher AUC in the range [1.09%, 16.00%] than GP_{ave}. It is important to mention that GP_{ave} is based on a standard classification strategy related to a weighting coefficient. This latter one is used in order to specify the importance of a majority class to a minority one. However, it is not easy to determine this coefficient because it is given by domain experts which is not the case of our proposed Bi-ICOS. Indeed, Bi-ICOS is based on a combination of GP and cost sensitive learning in order to automatically learn costs without requiring information from domain experts.

Compared with GP_{GMean}, Bi-ICOS achieves better results. Indeed, the superiority of Bi-ICOS appears in Yeoh-2002-v1 dataset (32.72% higher than GP_{GMean}). This observation is explained by the fact that the proposed Bi-ICOS does not require the determination of a predefined threshold prior in order to ensure the classification process. Moreover, two classification thresholds are computed using the evolved cost interval.

Compared with GP_{aucw}, Bi-ICOS is significantly better in 8 datasets. For the other two datasets, GP_{aucw} is slightly higher than Bi-ICOS. It is worth mentioning that GP_{aucw} is able to achieve promising classification results compared to GP_{GMean}, GP_{ave}, GP_{ADASYN}, and GP_{SMOTE}. However, it could be seen from the table that our poposed Bi-ICOS is able to outperform GP_{aucw} due to the proposed bi-level modeling that ensures a precise and fair evaluation. Compared with GP_{SMOTE} and GP_{ADASYN}, our proposed Bi-ICOS is significantly better in all used datasets.

In summary, all the previous observations that demonstrate the merits of the proposed Bi-ICOS are explained by two main

facts. On the one hand, cost-sensitive learning is used to help GP to address the performance bias issue while improving the classification performance by automatically learning the needed interval-based cost information. On the other hand, the proposed bi-level modeling ensures a precise evaluation of each generated tree and an effective approximation of optimal costs intervals bounds.

E. Analysis of GP trees examples for Lung and Colon diseases

To further discuss the obtained results, we have chosen two datasets (Lung with IR = 8 and Colon with IR = 2) in order to analyze examples of the evolved GP trees by Bi-ICOS. First, for the adopted lung disease dataset, there are 156 instances with 12600 features. In the example given by Fig. 4(a), the generated tree has 13 nodes in total, in which 4 features are selected from a totality of 12600 features. The selected features are used in order to develop a classification tree (i.e., the left sub-tree). The generated cost interval is [1.29, 1.11]. It is important to mention that for the test set, Bi-ICOS generates 100% AUC result for the lung dataset on both minority and majority classes. Second, for the adopted colon disease dataset, there are 62 instances with 2000 features. Fig. 4(b) gives an example of a tree that is evolved for the colon disease dataset. This tree is composed from 11 nodes in which two features have been selected while the evolved cost interval is [3.63, 2.78]. It is important to mention that Bi-ICOS achieves 100% AUC result for the colon dataset on both minority and majority classes. One can notice that the left sub-tree of the lung disease case is more complicated than the left sub-tree of the colon disease case. This observation could be explained by the fact that the lung dataset includes more features than the colon dataset.

V. Conclusions and future work

The aim of this paper was to tackle the imbalanced data classification. The main contributions are given as follows. First, we have proposed a bi-level modeling for the interval-based cost sensitive classification tree induction where trees are constructed at the upper level and misclassification costs intervals bounds are optimized at the lower level. Second, we have designed an improved version of an existing co-evolutionary bi-level algorithm to solve the resulting bi-level modeling. The main goal is to ensure efficacious variation and diversification of the evolved classification trees by modifying the migration strategy of the adopted algorithm. The resulting approach is named Bi-ICOS.

To test the efficiency of the proposed bi-level modeling, Bi-ICOS was compared with respect to recent interval-based cost sensitive approach and GP-based approaches with several fitness functions. Experiments on ten commonly-used imbalanced datasets illustrate the ability of Bi-ICOS on achieving better classification performance. The obtained results show the merits of the bi-level modeling in ensuring not only a precise evaluation of each evolved tree but also an effective approximation of the optimal costs intervals bounds.

TABLE IV: The obtained AUC results (%).

Dataset	Method	Best	Median	Std
Lung	Bi-ICOS	100	**99.49**	**2.03**
	ICS-GP	100(≈)	98.29(-)	3.63(-)
	GP_{SMOTE}	100(≈)	80.99(-)	16.20(-)
	GP_{ADASYN}	100(≈)	82.50(-)	15.01(-)
	GP_{ave}	100(≈)	83.49(-)	14.77(-)
	GP_{G-mean}	99.15(-)	80.97(-)	18.30(-)
*	GP_{aucw}	100(≈)	92.24(-)	13.30(-)
tomlins-2006-v1	Bi-ICOS	100	**98.89**	**1.25**
	ICS-GP	100(≈)	97.37(-)	2.50(-)
	GP_{SMOTE}	100(≈)	83.30(-)	13.40(-)
	GP_{ADASYN}	100(≈)	84.59(-)	13.27(-)
	GP_{ave}	100(≈)	88.80(-)	13.50(-)
	GP_{G-mean}	100(≈)	84.46(-)	14.60(-)
*	GP_{aucw}	100(≈)	91.20(-)	9.80(-)
Yeoh-2002-v1	Bi-ICOS	100	98.92	2.30
	ICS-GP	100(≈)	97.80(-)	3.90(-)
	GP_{SMOTE}	100(≈)	87.37(-)	9.31(-)
	GP_{ADASYN}	100(≈)	84.30(-)	10.26(-)
	GP_{ave}	100(≈)	84.01(-)	11.82(-)
	GP_{G-mean}	95.66(-)	66.20(-)	16.29(-)
*	GP_{aucw}	100(≈)	**99.02(+)**	**2.28(+)**
Gordon-2002	Bi-ICOS	100	99.28	1.98
	ICS-GP	100(≈)	97.10(-)	2.41(-)
	GP_{SMOTE}	100(≈)	97.39(-)	3.01(-)
	GP_{ADASYN}	100(≈)	97.80(-)	2.88(-)
	GP_{ave}	100(≈)	98.19(-)	2.85(-)
	GP_{G-mean}	100(≈)	98.40(-)	2.98(-)
*	GP_{aucw}	100(≈)	**99.35(+)**	**1.97(+)**
DLBCL	Bi-ICOS	100	**87.75**	**9.50**
	ICS-GP	100(-)	81.60(-)	11.90(-)
	GP_{SMOTE}	98.05(-)	83.10(-)	9.63(-)
	GP_{ADASYN}	100(≈)	79.55(-)	10.87(-)
	GP_{ave}	97.97(-)	75.06(-)	15.80(-)
	GP_{G-mean}	100(≈)	76.80(-)	16.01(-)
*	GP_{aucw}	100(≈)	86.30(-)	9.61(-)
SHIPP-2002-V1	Bi-ICOS	100	**88.12**	**4.17**
	ICS-GP	95.10(-)	83.99(-)	9.17(-)
	GP_{SMOTE}	98.10(-)	81.95(-)	12.01(-)
	GP_{ADASYN}	95.90(-)	79.50(-)	12.33(-)
	GP_{ave}	98.82(-)	82.64(-)	9.92(-)
	GP_{G-mean}	95.93(-)	82.79(-)	10.11(-)
*	GP_{aucw}	100(≈)	82.55(-)	9.60(-)
Leukemia	Bi-ICOS	100	**95.17**	**4.66**
	ICS-GP	100(≈)	87.60(-)	9.45(-)
	GP_{SMOTE}	100(≈)	87.61(-)	9.43(-)
	GP_{ADASYN}	100(≈)	89.80(-)	8.50(-)
	GP_{ave}	97.95(-)	88.40(-)	7.93(-)
	GP_{G-mean}	100(≈)	81.12(-)	15.70(-)
*	GP_{aucw}	100(≈)	86.77(-)	9.47(-)
Colon	Bi-ICOS	97.17	**83.65**	**3.64**
	ICS-GP	93.06(-)	78.89(-)	6.45(-)
	GP_{SMOTE}	93.01(-)	76.11(-)	10.30(-)
	GP_{ADASYN}	87.89(-)	74.08(-)	10.65(-)
	GP_{ave}	92.10(-)	76.04(-)	9.77(-)
	GP_{G-mean}	93.24(-)	72.01(-)	12.60(-)
*	GP_{aucw}	92.15(-)	79.31(-)	6.10(-)
GOLUB-1999-V1	Bi-ICOS	100	**99.59**	**1.50**
	ICS-GP	100(≈)	99.02(-)	3.02(-)
	GP_{SMOTE}	100(≈)	91.95(-)	10.51(-)
	GP_{ADASYN}	100(≈)	91.21(-)	9.98(-)
	GP_{ave}	100(≈)	91.86(-)	10.21(-)
	GP_{G-mean}	100(≈)	90.02(-)	11.77(-)

Fig. 4: Illustration of two examples of GP trees for (a) lung and (b) colon diseases.

Several future paths of research could be followed from this work. For instance, it would be interesting to investigate Bi-ICOS' performance for the multi-class classification case since we have only considered binary classification. Also, it would be interesting to investigate ensemble learning in order to improve the generality of learned cost intervals. Indeed, the main goal is to improve the Bi-ICOS performance with the application of ensemble learning in order to explore how several cost-sensitive classifiers could use the same cost information.

ACKNOWLEDGMENTS

Carlos A. Coello Coello gratefully acknowledges support from CONACyT grant no. 2016-01-1920 (Investigación en Fronteras de la Ciencia 2016).

REFERENCES

[1] U. Bhowan, M. Johnston, and M. Zhang, "Developing new fitness functions in genetic programming for classification with unbalanced data," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 406–421, 2011.

[2] H. Zhao and X. Li, "A cost sensitive decision tree algorithm based on weighted class distribution with batch deleting attribute mechanism," *Information Sciences*, vol. 378, pp. 303–316, 2017.

[3] D. Fuqua and T. Razzaghi, "A cost-sensitive convolution neural network learning for control chart pattern recognition," *Expert Systems with Applications*, vol. 150, p. 113275, 2020.

[4] X.-Y. Liu and Z.-H. Zhou, "Learning with cost intervals," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 403–412.

[5] M. ONeill, "Riccardo poli, william b. langdon, nicholas f. mcphee: a field guide to genetic programming," 2009.

[6] W. Pei, B. Xue, M. Zhang, and L. Shang, "A cost-sensitive genetic programming approach for high-dimensional unbalanced classification," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2019, pp. 1770–1777.

[7] W. Pei, B. Xue, L. Shang, and M. Zhang, "Developing interval-based cost-sensitive classifiers by genetic programming for binary high-dimensional unbalanced classification [research frontier]," *IEEE Computational Intelligence Magazine*, vol. 16, no. 1, pp. 84–98, 2021.

[8] R. Said, M. Elarbi, S. Bechikh, and L. Ben Said, "Solving combinatorial bi-level optimization problems using multiple populations and migration schemes," *Operational Research*, pp. 1–39, 2021.

[9] A. C. Bahnsen, D. Aouada, and B. Ottersten, "Example-dependent cost-sensitive decision trees," *Expert Systems with Applications*, vol. 42, no. 19, pp. 6609–6619, 2015.

[10] Y. Zhang and Z.-H. Zhou, "Cost-sensitive face recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 10, pp. 1758–1769, 2009.

[11] Z.-H. Zhou and X.-Y. Liu, "On multi-class cost-sensitive learning," *Computational Intelligence*, vol. 26, no. 3, pp. 232–257, 2010.

[12] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, "Cost-sensitive learning," in *Learning from Imbalanced Data Sets*. Springer, 2018, pp. 63–78.

[13] L. Zhang and D. Zhang, "Evolutionary cost-sensitive extreme learning machine," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 12, pp. 3045–3060, 2016.

[14] C. Zhang, K. C. Tan, H. Li, and G. S. Hong, "A cost-sensitive deep belief network for imbalanced classification," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 1, pp. 109–122, 2018.

[15] J. Li, X. Li, and X. Yao, "Cost-sensitive classification with genetic programming," in *2005 IEEE congress on evolutionary computation*, vol. 3. IEEE, 2005, pp. 2114–2121.

[16] C. Elkan, "The foundations of cost-sensitive learning," in *International joint conference on artificial intelligence*, vol. 17, no. 1. Lawrence Erlbaum Associates Ltd, 2001, pp. 973–978.

[17] Z. Zhu, Y.-S. Ong, and M. Dash, "Markov blanket-embedded genetic algorithm for gene selection," *Pattern Recognition*, vol. 40, no. 11, pp. 3236–3248, 2007.

[18] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[19] H. He, Y. Bai, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. IEEE, 2008, pp. 1322–1328.

[20] B. Tran, B. Xue, and M. Zhang, "Genetic programming for feature construction and selection in classification on high-dimensional data," *Memetic Computing*, vol. 8, no. 1, pp. 3–15, 2016.

[21] L. Yan, R. H. Dodier, M. Mozer, and R. H. Wolniewicz, "Optimizing classifier performance via an approximation to the wilcoxon-mann-whitney statistic," in *Proceedings of the 20th international conference on machine learning (icml-03)*, 2003, pp. 848–855.

[22] A. E. Eiben and S. K. Smit, "Parameter tuning for configuring and analyzing evolutionary algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 19–31, 2011.

[23] J. Carrasco, S. García, M. Rueda, S. Das, and F. Herrera, "Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review," *Swarm and Evolutionary Computation*, vol. 54, p. 100665, 2020.