A Differential Evolution-Based Method for Class-Imbalanced Cost-Sensitive Learning

Chen Qiu, Liangxiao Jiang*, and Ganggang Kong
Department of Computer Science, China University of Geosciences, Wuhan 430074, China
Hubei Intelligent Geo-Information Processing Key Laboratory, China University of Geosciences, Wuhan 430074, China
Email: ljiang@cug.edu.cn

***Abstract*—In many real-world applications, it is often the case that the class distribution of example is imbalanced and the costs of misclassification are different. In such circumstances, not classification accuracy but misclassification cost minimization is the primary goal leading to the development of the class-imbalanced cost-sensitive learning. Under-sampling is one of the most important methods in dealing with class-imbalance problems, which uses only a subset of the majority class that is imperfect. In fact, many useful majority class examples are easy to be overlooked. To overcome this deficiency, in this paper, we propose a new adaptive method based on differential evolution for class-imbalanced cost-sensitive learning. We simply denote our proposed method by DE. DE explores the potential useful information contained in these majority class examples and creates an optimal example subset with the goal of low misclassification costs. We compare the performance of DE to its state-of-the-art competitors such as random under-sampling (US) and synthetic minority over-sampling technique (SMOTE) in terms of the total misclassification costs of the resulting base classifiers. The experimental results on 22 class-imbalanced data sets show that our proposed DE is notably better than the widely used random under-sampling (US) and the well-known synthetic minority over-sampling technique (SMOTE).**

I. Introduction

In many real-world applications, the class distribution of examples is typically imbalanced [1], and a correct classification of the minority class often has greater value than that of the majority class. Because the examples that belong to the minority class occur infrequently, the models that describe the minority class tend to be highly specialized. Therefore, detecting the examples that belong to the minority class is very difficult and many of the existing classifiers (models) aren't qualified for this task.

Besides the skewed class distribution, in many application domains, the datasets are characterized by different costs associated with different types of errors. Misclassifying a minority class example carries much higher costs of misclassification [2], [3], [4]. For example, in fault diagnosis, the cost of misclassifying a fault conditions to be a normal conditions is much bigger than the opposite type of error. Cost-sensitive classification considers different misclassification costs for the examples in the learning algorithm. Hence, new related techniques are necessary to ensure that a classifier attaches great importance to these important yet rare class examples and handles class-imbalanced cost-sensitive learning well.

*Corresponding author: ljiang@cug.edu.cn

Many techniques have been developed aiming to address the related problems. These techniques can be broadly categorized into three groups [5]. The first one is to design classifiers that are cost-sensitive in themselves. In second group, the algorithm tries to adapt the decision threshold to impose a bias on the minority class by assigning different misclassification costs for examples. In the third group, the processing techniques do not modify the existing algorithms, on the contrary they resample the natural data aiming to balance the skewed class distribution of examples and weaken the serious impact caused by imbalance [6]. Resampling approaches balance the class distribution in natural data by either generating new examples to the minority class (over-sampling) or removing examples from the majority class (under-sampling). More details about resampling techniques are presented in Section II.

Under-sampling approaches attempt to balance the natural class distribution in order to obtain an accurate classifier when the training data is highly skewed. In the specialized literature, there are several theoretical reasons for the usage of under-sampling methods in dealing with class-imbalance problems [6], [7]. Under-sampling approaches use a subset of the majority class to train a classifier. The way in which many majority class examples are discarded is the key factor to balance data sets and to obtain better performance. The simplest method for under-sampling a data set is random under-sampling (US) [8]. US randomly removes examples from the majority class until the desired balance is achieved. However, the main drawback of under-sampling is the loss of useful information contained in these deleted examples from the training data.

By analyzing its performance, random under-sampling technique is still competitive despite it's randomness. However, for the task of class-imbalanced cost-sensitive learning, we believe that such a random technique might be improved in terms of the misclassification costs in a supervised manner. The intuition of our proposed method is to explore the potential useful information contained in these deleted examples and guide the process of selecting an optimal subset of examples.

Instead of random under-sampling, we propose a differential evolution-based algorithm, denoted by DE, to optimize the process of under-sampling. Our method employs differential evolution to guide the sampling process for reducing the total misclassification costs. After we obtain an optimal subset of examples, the base classifier will be trained on it. Among the genetic algorithms, differential evolution (DE) [9], [10] is a fast and robust population-based search algorithm, which is designed to work on large space with possible solutions.

978-1-4799-1959-8/15/$31.00 ©2015 IEEE

We generate a certain number of subsets corresponding to the chromosomes in the initial population. In the process of evolution, we explore the useful examples of the majority class but not being at the expense of losing randomness with respect to that of US [8]. To tackle the class-imbalanced cost-sensitive learning, we take the use of the misclassification costs as the fitness function. Experimental results on 22 class-imbalanced datasets [11], [12], [13] show that our proposed DE significantly outperforms US and the other well-known SMOTE [14] in terms of the total misclassification cost.

The rest of this paper is organized as follows. We start by a brief introduction of some related work. Then we propose our differential evolution-based method (DE) and report on our experiments, which is followed by the conclusion.

II. Related work

As mentioned above, when one class is under-represented in the dataset, skewed class distribution hinders the classifier learning. Moreover, the minority class is usually the class of interest. A large number of algorithms have been proposed to address the class-imbalance problem. Here, we mainly focus on sampling algorithms and related improvements of algorithms based on evolution algorithm.

Sampling is one of the widely used methods in many real-world applications, which alters the distribution of the training data so that the minority class is well represented in training data [15], [16], [17], [6], [18]. Sampling either generates new examples to the minority class (over-sampling) or deletes examples from the majority class (under-sampling) until the expected balance is achieved. Weiss and Provost [19] empirically study the effect of the class distribution on decision tree's performance. Their experimental results show that a more balanced class distribution outperforms the natural class distribution.

Many sampling methods have been proposed to balance the class distribution and have been shown to be helpful. The simplest form of under-sampling is random under-sampling, which randomly removes examples from the majority class until a balanced distribution is reached. When the misclassification costs are taken into account, it has been shown that random under-sampling is capable of reducing costs. Jiang et. [8] empirically study the effect of random sampling on state-of-the-art Bayesian network classifiers, such as Naive Bayes (NB) [20], Tree Augmented Naive Bayes (TAN) [21], Averaged One-Dependence Estimators (AODE) [22], Weighted Average of One-Dependence Estimators (WAODE) [23], and Hidden naive Bayes (HNB) [24]. Their proposed under-sampling method, called US, selects a subset by preserving all the minority class and randomly sampling the majority class.

In the same way, random over-sampling balance a dataset by duplicating examples of the minority class, which is also useful for class-imbalanced cost-sensitive learning. Under-sampling results in a loss of information, while over-sampling have the most serious drawback itself. It replicates the minority class. This will lead to overfitting. There are some methods that sample in more complex ways attempting to overcome the drawback. The Synthetic Minority Over-sampling Technique (SMOTE) [14] over-sampling the minority class examples by randomly interpolating pairs of closet neighbors in the minority class. Zhou [7] studies the under-sampling and over-sampling method in training cost-sensitive neural networks.

Genetic algorithm (GA) are another promising and important approach to sampling, which can be represented as an optimization problem. Some genetic-based algorithms have been proposed. In Garicia, cano et al. [29], a evolutionary prototype selection algorithms is proposed for balancing data. Then, a CHC-based evolutionary method is provided. The EUS method [30] is designed for performing a prototype selection process with the aim of balancing data. Besides, based on the ensemble construction learning, Galer et al. [31] propose an evolutionary under-sampling approach, called EUSBoost, to enhance the performance of the base classifier.

III. A Differential evolution-based method

In this paper, for simplicity, we restrict our discussion to binary classification problems. The minority class is denoted as the positive class (+), while the majority class is denoted as the negative class (-). There is a training set S with N examples. Assuming the positive class has N_+ examples and negative class has N_- examples.

Just as the existing literatures [7], [26] show, the class-imbalanced cost-sensitive learning is one of the research hotspots in data mining and machine learning. To achieve the best results at the lowest cost, altering the distribution of training data is a critical way. According to the ratios of size of majority data to the size of minority data, random under-sampling algorithm (US) [8] samples each majority training example with same probability. Such a sampling method may lose some useful information contained in the majority class examples. We believe that the use of more sophisticated methods, such as integrating the sampled majority subset in the framework of differential evolution, could enhance the performance of the random under-sampling (US) and make its advantage stronger.

Differential evolution is a simple but efficient evolutionary algorithm for global optimization. The individual corresponding to the sampled solution is generated randomly and evolved towards the global optimal fitness. The advantage of the algorithm is its simple structure, ease of use, robustness and fast convergence. In this section, we propose a differential evolution-based approach to optimize the sampled subset for class-imbalanced cost-sensitive learning. We firstly describe the frame of differential evolution then present our algorithm for sampling.

Differential evolution follows the general procedure of an evolutionary algorithm. The performance of differential evolution depends on the mutation strategy, the crossover operation, and the control parameters like population size N_p, mutation scaling factor F and cross rate C_r. The setting of parameters in differential evolution are shown in section IV-A. We consider four aspects of the differential evolution here: chromosome representation, fitness function, mutation operator and crossover operator.

A. Chromosome representation

One of the important issues of differential evolution is the representation of the solutions. Let $S \subseteq R^d$ be the d-dimensional search space of the problem. Differential evolution

begins with a randomly initiated population of N_p including d-dimensional real-valued vectors. A property of the sampling optimization problem is expressed here. That is, the solution space which is composed by discrete elements instead of continuous space. We need to map these d real numbers to N_+ integers. As shown in [27][28], using differential evolution for continuous optimization problems to solve the discrete optimization problem directly is an important research branch on discrete optimization. Thus, in our framework of DE, each vector, called chromosome, is coded as integer values and consists of N_+ genes representing the majority examples selected. Each gene in a chromosome takes the value from $[1, N_-]$, which stands for the presence of corresponding majority class examples in model training. Considering the minority class is the primary interest, the search space is only focus on majority class and all minority class are always introduced to the new subset of datasets. We denote subsequent generations in DE by $G = 1, 2 \cdots, G_{max}$. The ith $(i = 1, \ldots, N_p)$ chromosome of the population at the current generations is represented as:

$$\vec{X}_{i,G} = [x_{1,i,G}, x_{2,i,G}, \cdots, x_{j,i,G}, \cdots, x_{N_+,i,G}], \quad (1)$$

where $x_{j,i,G} \in [1, N_-]$, indicating the corresponding index of each majority class examples in the datasets. In this work, the initial population (at $G = 1$) are generated by randomly individuals within the search space, which is constrained by our described range from $[1, N_-]$.

B. Fitness function

The main idea of DE algorithm is just to search the optimal or near optimal distribution of training data. The optimization problem in our DE algorithm can be formally stated as:

$$Global\ minf(\vec{X}).\vec{X} = (x_1, x_2, x_3 \cdots, x_{N_+}), \quad (2)$$

where $f(\vec{X})$ is the fitness function.

Cost-sensitive classification is designed to yield classifiers that minimize the total misclassification cost. The cost is defined as:

$$Cost = FN \cdot C(+,-) + FP \cdot C(-,+), \quad (3)$$

where FN is the number of positive examples misclassified as negative, $C(+,-)$ is the cost of false negative, FP is the number of negative examples misclassified as positive, $C(-,+)$ is the cost of false positive. There is no cost for correct prediction. In this paper, $C(-,+)$ is fixed at 1, $C(+,-)$ is assigned by domain experts or set to be 5, 10, 15 and 20 respectively.

Naturally, we consider the total misclassification cost as fitness function directly. The performance is evaluated by across validation. The fitness value of an individual reflects the survival ability for the next generation, where the lower cost reflects the stronger ability. The fitness function is designed as:

$$f(\vec{X}) = FN \cdot C(+,-) + FP \cdot C(-,+). \quad (4)$$

C. Mutation operator

The key operator to differential evolution is differential mutation operators. Price and Storn [10] proposed several classical mutation operators. We briefly introduce a mutation operator which has been employed in our new method. For each target vector $\vec{X}_{i,G}$ at generation G, an associated mutant vector $\vec{V}_{i,G}$ can be generated by the employed mutation operator. The mutation operator, called "$DE/rand-to-best/2$", is as follow

$$\begin{aligned}\vec{V}_{i,G} &= \vec{X}_{r_1^i,G} + F \cdot (\vec{X}_{best,G} - \vec{X}_{r_1^i,G}) \\ &\quad + F \cdot (\vec{X}_{r_2^i,G} - \vec{X}_{r_3^i,G}) + F \cdot (\vec{X}_{r_4^i,G} - \vec{X}_{r_5^i,G}),\end{aligned} \quad (5)$$

where $\vec{X}_{best,G}$ represents the best individual vector in the population at generation G; The indicates $r_1^i, r_2^i, r_3^i, r_4^i$ and r_5^i are mutually exclusive integers randomly chosen from the range $[1, N_p]$, and all are different from the index i.

D. Crossover operator

After the mutation stage, a binominal crossover operation is applied to each pair of the target vector $\vec{X}_{i,G}$ and its corresponding mutation vector $\vec{V}_{i,G}$, which forms the final trail vector $\vec{U}_{i,G} = [u_{1,i,G}, u_{2,i,G}, u_{3,i,G} \cdots, u_{N_+,i,G}]$.

$$u_{j,i,G} = \begin{cases} v_{j,i,G},\ if\ \textit{rand}(0,1) \leq C_r\ or\ j = j_{rand}, \\ x_{j,i,G},\ otherwise \end{cases} \quad (6)$$

where *rand*(0,1) represents a uniformly distributed random variable within the range [0,1), which is generated anew for each jth component of the ith parameter vector. $j_{rand} \in [1, 2 \cdots, N_+]$ is a randomly chosen index, which ensures that $\vec{U}_{i,G}$ gets at least one majority class example from $\vec{V}_{i,G}$.

E. Framework of the proposed method

We integrate the idea of differential evolution into the random under-sampling. Thus, our proposed DE is a heuristic under-sampling method instead of random technique and suits for arbitrary proportion in data sets. Given a training data set $\mathbf{S}$, our algorithm firstly partitions $\mathbf{S}$ into two disjoint subsets, namely $\mathbf{S}_+$ and $\mathbf{S}_-$, according to the class values. Then, our algorithm performs a differential evolution-based sampling on $\mathbf{S}_-$ to create an optimal example subset $\mathbf{NS}_-$. Finally, our algorithm merges the original majority class example set $\mathbf{S}_+$ and the newly created minority class example set $\mathbf{NS}_-$, and then builds a base classifier $\mathbf{L}$ on the merged training data set $\mathbf{D}$ ($\mathbf{D} = \mathbf{S}_+ \bigcup \mathbf{NS}_-$).

In the procedure of differential evolution, the algorithm generates the initial population of individuals corresponding to the selected majority class examples. For each individual, the training datasets and validation datasets are constructed using the selected majority class examples and all the minority class examples. Based on the base classifier $\mathbf{L}$, the performance of the resulting model is evaluated on the training data in terms of the total misclassification costs. The 10-fold cross-validation is embedded into the evaluation process. Through the crossover and mutation operators, the offspring or parents with better fitness values are selected to the next generation. Besides, the evolution is terminated until the stopping condition is satisfied, which is the maximum number of iterations G_{max}. The presented DE algorithm is summarized in Algorithm 1.

The only point where the coding details require explanation is Step 3, where it is possible that a few genes in newly

Algorithm 1 Framework of the proposed method (S, L)

Input: the original training data $\mathbf{S}$; A base classifier $\mathbf{L}$;
Output: the built classifier $\mathbf{L}$

1. Partitions $\mathbf{S}$ into two disjoint subsets, namely $\mathbf{S}_+$ and $\mathbf{S}_-$, according to the class values;
2. Performs a differential evolution-based sampling on $\mathbf{S}_-$ to create an optimal example subset $\mathbf{NS}_-$;
   (1) Initialize the population, encoding the randomly selected majority class examples $\{x_1, x_2, \cdots, x_{N_+}\}$ for each individual $\vec{X}$;
   (2) Classify the training data based on classification algorithm $\mathbf{L}$ and evaluate the fitness value for each individual;
   (3) If the stopping condition is satisfied, output the optimal solution and go to Step 3;
   (4) Select the best individual and find out five points from population randomly;
   (5) Perform mutation and crossover according to the Equation (5) and (6) on parents.
   (6) Replace the parents with offspring if they have lower misclassification costs;
3. Merges the original majority class example set $\mathbf{S}_+$ and the newly created minority class example set $\mathbf{NS}_-$;
4. Builds a base classifier $\mathbf{L}$ on the merged training data set $\mathbf{D}$ ($\mathbf{S}_+ \bigcup \mathbf{NS}_-$);
5. **Return** the built classifier $\mathbf{L}$.

example set $\mathbf{NS}_-$ will represent the same integer. In other words, the same majority example will be selected more than once. In that special case, we set larger weights for those genes according to the frequency of appearance.

The intention of DE is to select an optimal subset in supervised manner. Genetic algorithm (GA) mainly finds the optimal or near optimal solution to optimization problems. In the research of test costs in terms of money or time, Several works employ different GA strategies to evolution better results. Pan et al. [32] propose a genetic algorithm and Xu [33] present an ant colony algorithm to reduce test cost. Xu [34] develops a genetic algorithm to the multi-objective reduction problem involving multiple types of test costs. It is evident that DE has benefited from the mechanism of evolution to reduce related costs. Both EUS [30] and DE attempt to employ related GA algorithms for exploring better results. The former uses several evolutionary model to balance datasets while our DE is proposed as a class-imbalanced cost-sensitive learning method based on the advanced differential evolution algorithm. EUSBoost [31] also uses the evolved subset of the original training set in the framework of Boosting. EUSBoost sets the geometric mean (GM) as the fitness function. While we consider the misclassification costs as the fitness function with the objective of minimizing costs. In a word, our proposed method is totally different from the previous methods.

IV. Experiments and Results

A. Experimental Setting

The purpose of these experiments is to validate the effectiveness of the proposed DE for class-imbalanced cost-sensitive learning. In this paper, we use the total misclassification costs as the performance evaluation measure. Based on the confusion matrix in Table I, the misclassification costs is defined in Eq. (3). We run our experiments on 22 typical class-imbalance datasets [11], which are downloaded from the home page of KEEL [12], [13]. Table II provides the characteristics of these data sets, including the number of examples (Size), the number of attributes (Attribute), the class distribution (%Class(min., maj.)), and the size of majority class divided by that of majority class (IR). These data sets represent a variety of data set sizes, imbalance levels, and application domains.

TABLE I.
CONFUSION MATRIX.

	Predicted Positive Class	Predicted Negative Class
Actual Positive Class	TP (True Positive)	FN (False Negative)
Actual Negative Class	FP (False Positive)	TN (Ture Negative)

TABLE II.
DATASETS USED IN ●UR EXPERIMENTS.

Dataset	Size	Attribute	%Class(min., maj.)	IR
ecoli-0-vs-1	220	7	(35.00, 65.00)	1.86
Ecoli1	336	7	(22.92, , 77.08)	3.36
Ecoli2	336	7	(15.48, 84.52)	5.46
Ecoli3	336	7	(10.42, 89.56)	8.60
Glass0	214	9	(32.71, 67.29)	2.06
glass-0-1-2-3	214	9	(23.83, 76.17)	3.19
Glass1	214	9	(35.51, 64.49)	1.82
Glass6	214	9	(13.55, 86.45)	6.38
Haberman	306	3	(26.47, 73.53)	2.78
Iris0	150	4	(33.33, 66.67)	2.00
New-thyroid1	215	5	(16.28, 83.72)	5.14
New-thyroid2	215	5	(16.28, 83.72)	5.14
Page-blocks0	5472	10	(10.22, 89.78)	8.79
Pima	768	8	(34.90, 65.10)	1.87
Segment0	2308	19	(14.25, 85.75)	6.02
Vehicle0	846	18	(23.52, 76.48)	3.25
Vehicle1	846	18	(25.65, 74.35)	2.90
Vehicle2	846	18	(25.77, 74.23)	2.88
Vehicle3	846	18	(25.06, 74.94)	2.99
Wisconsin	683	9	(35.00, 65.00)	1.86
Yeast1	1484	8	(28.91, 71.09)	2.46
Yeast3	1484	8	(10.98, 89.02)	8.11

We test our proposed method (simple DE) by choosing two state-of-the-art Bayesian network classifiers, namely naive Bayes (NB) [20] and tree augmented naive Bayes (TAN) [21], as the base classifiers. We compare our proposed method (DE) to the widely used random under-sampling (US) [8] and the well-known synthetic minority over-sampling technique (SMOTE) [14]. The the total misclassification costs of each algorithm on each data set is obtained via 10 runs of 10-fold cross-validation. Runs with the various algorithms are carried out on the same training sets and evaluated on the same test sets. In particular, the cross-validation folds are the same for all the experiments on each data set.

Now, we introduce established algorithms and their abbreviations used in our experiments.

- NB-DE: DE with NB as the base classifier. (The population size N_p, the mutation scaling factor F, the cross rate C_r, and the maximum number of iterations G_{max} are set to 50, 0.5, 0.9 and 50 respectively.)
- NB: the naive Bayesian classifier.
- NB-US: US with NB as the base classifier. (Choosing SBNC-undersampling algorithm (US) as the sampling approach.)

- NB-SMOTE: SMOTE with NB as the base classifier. (The created examples percentage of SMOTE, the number of nearest neighbors, and the seed used for random sampling are set to 100%, 5 and 1 respectively.)
- TAN-DE: DE with TAN as the base classifier. (The population size N_p, the mutation scaling factor F, the cross rate C_r, and the maximum number of iterations G_{max} are set to 50, 0.5, 0.9 and 50 respectively.)
- TAN: the tree augmented naive Bayesian classifier.
- TAN-US: US with TAN as the base classifier. (Choosing SBNC-undersampling algorithm (US) as the sampling approach.)
- TAN-SMOTE: SMOTE with TAN as the base classifier (The created examples percentage of SMOTE, the number of nearest neighbors, and the seed used for random sampling are set to 100%, 5 and 1 respectively.)

B. Experimental Results

This section presents the results of our experiments with DE and its competitors (US, SMOTE). For comparison purposes, we perform two different statistical analysis tests when cost ratio is 20. Firstly, we conducted the corrected paired two-tailed t-test with the $p = 0.05$ significance level [35] to compare our proposed DE with each of its competitors. Then, we use the Friedman test with the corresponding post-hoc tests [36] such as Bergmann test for comparison of more classifiers over multiple data sets. Finally, to further validate the effectiveness of our proposed method under different cost ratios, we observe its performance under various cost ratios (5, 10, 15, 20) in terms of the total misclassification costs.

1) Paired t-test: Table III- IV show the total misclassification costs of each classifier on each dataset when cost ratio is 20. The symbols $\circ$ and $\bullet$ in the tables respectively denote statistically significant upgrade or degradation over our proposed method with a corrected paired two-tailed t-tests at 95% significance level [35], respectively. The averages and **W/T/L** values are summarized at the bottom of the tables. Please note that, in terms of the total misclassification costs, the lower the number, the better the performance of the classifier. For example, Table III shows that, the **W/T/L** value of NB-DE vs. NB is 11/11/0, that means that NB-DE significantly outperforms NB on 11 datasets, ties on 11 datasets and loses on 0 dataset.

The total misclassification costs of the compared methods are summarized in Table III and Table IV. It is obvious that the misclassification costs of the classifiers using our proposed method are much lower than the original cost-insensitive classifiers. For details, in Table III, NB-DE is significantly superior than NB on 11 datasets; in Table IV, TAN-DE is much better than TAN on 13 datasets. More important, our proposed DE-based method performs significantly better than US and SMOTE on 22 class-imbalanced datasets. In other words, the supervised selection of the optimal example subset based on differential evolution is better than random sampling. NB-DE is significantly better than NB-US and NB-SMOTE on 7 and 9 datasets respectively, TNA-DE is significantly better than TAN-US and TAN-SMOTE 6 and 10 datasets respectively.

2) Friedman test: To present significant statistic differences among these algorithms, we use the KEEL tool [12] for non-parametric statistical analysis. The Friedman test is a non-parametric for comparison of multiple algorithms over multiple data sets. The average rankings of the algorithms obtained by applying the Friedman tests are summarized at Tables V and VII respectively. With 4 algorithms and 22 data sets, F_F is distributed according to the F distribution with 3 and 63 degrees of freedom: 17.239278, 62.448276 respectively, which are all greater than the critical values of F(3, 63) for a = 0.05. So we reject the null hypotheses and proceed with Bergmann test to find out exactly the significant difference among these algorithms. Table VI and VIII report the obtained z-values and the p-values, where the detailed results using the Bergmann test indicates which algorithms are significantly different.

TABLE V.
AVERAGE RANKINGS ●F THE ALG●RITHMS BASED ●N NB

Algorithm	Ranking
NB-DE	1.3864
NB	3.3864
NB-US	2.2955
NB-SMOTE	2.9318

TABLE VI.
P-VALUES F●R $\alpha = 0.05$ BASED ●N NB

i	algorithms	$z = (R_0 - R_i)/SE$	p
6	NB-DE vs. NB	5.138093	0
5	NB-DE vs. NB-SMOTE	3.970345	0.000072
4	NB vs. NB-US	2.802596	0.005069
3	NB-DE vs. NB-US	2.335497	0.019517
2	NB-US vs. NB-SMOTE	1.634848	0.102081
1	NB vs. NB-SMOTE	1.167748	0.242908

Bergmann's procedure rejects these hypotheses:

- NB-DE vs. NB
- NB-DE vs. NB-US
- NB-DE vs. NB-SMOTE
- NB vs. NB-US

TABLE VII.
AVERAGE RANKINGS ●F THE ALG●RITHMS BASED ●N TAN

Algorithm	Ranking
TAN-DE	1.2045
TAN	3.8409
TAN-US	2.1136
TAN-SMOTE	2.8409

TABLE VIII.
P-VALUES F●R $\alpha = 0.05$ BASED ●N TAN

i	algorithms	$z = (R_● - R_i)/SE$	p
6	TAN-DE vs. TAN	6.772941	0
5	TAN vs. TAN-US	4.437444	0.000009
4	TAN-DE vs. TAN-SMOTE	4.203894	0.000026
3	TAN vs. TAN-SMOTE	2.569047	0.010198
2	TAN-DE vs. TAN-US	2.335497	0.019517
1	TAN-US vs. TAN-SMOTE	1.868397	0.061707

Bergmann's procedure rejects these hypotheses:

- TAN-DE vs. TAN
- TAN-DE vs. TAN-US
- TAN-DE vs. TAN-SMOTE
- TAN vs. TAN-US
- TAN vs. TAN-SMOTE

TABLE III.
COMPARISON OF THE TOTAL MISCLASSIFICATION COSTS FOR NB-DE VERSUS NB, NB-US AND NB-SMOTE WHEN COST RATIO=20.

Dataset	NB-DE	NB	NB-US	NB-SMOTE
ecoli-0-vs-1	5.34± 8.83	6.98±10.28	6.13± 9.51	7.20±10.33
ecoli1	16.58±15.15	28.80±17.62 ○	21.28±16.48	25.84±17.61
ecoli2	11.01±12.10	16.74±16.19	11.53±12.66	15.03±14.87
ecoli3	8.52± 6.83	13.52±12.59	8.47± 7.95	11.03±12.03
glass-0-1-2-3	4.84± 6.71	20.90±16.99 ○	19.44±16.73 ○	19.66±18.12 ○
glass0	17.52±13.55	21.03±17.28	19.20±15.38	20.63±16.62
glass1	18.96±14.18	23.65±16.58	26.12±20.80	22.91±16.84
glass6	5.62± 7.18	8.90±12.03	6.17± 9.23	9.90±12.68
haberman	79.30±32.00	129.65±20.17 ○	104.03±28.38 ○	114.79±23.69 ○
iris0	0.00± 0.00	0.00± 0.00	0.00± 0.00	0.00± 0.00
new-thyroid1	0.84± 1.03	0.44± 0.64	1.01± 1.09	0.85± 0.90
newthyroid2	0.97± 1.00	0.44± 0.61	1.07± 0.95	0.87± 0.87
page-blocks0	392.71±71.03	638.94±73.90 ○	541.68±88.16 ○	622.44±71.88 ○
pima	175.14±51.21	221.35±52.12 ○	182.96±48.82	174.42±49.18
segment0	45.69±11.39	42.75±13.98	44.68±13.36	48.56±13.61
vehicle0	42.44±17.92	78.03±37.36 ○	53.36±26.68	60.93±31.17 ○
vehicle1	110.67±38.93	173.19±42.33 ○	149.68±44.36 ○	157.71±41.38 ○
vehicle2	81.24±37.93	209.26±50.40 ○	132.35±49.95 ○	145.88±50.23 ○
vehicle3	113.21±35.03	178.88±39.37 ○	161.54±42.75 ○	166.17±41.12 ○
wisconsin	5.34± 7.23	14.32±14.51	11.68±13.31	12.64±13.55
yeast1	218.05±56.09	453.34±60.43 ○	270.37±60.53 ○	286.25±60.16 ○
yeast3	33.88±20.24	95.25±36.65 ○	40.59±23.38	77.15±33.14 ○
Average	63.08	108.02	82.42	90.95
W/T/L	-	11/11/0	7/15/0	9/13/0

TABLE IV.
COMPARISON OF THE TOTAL MISCLASSIFICATION COSTS FOR TAN-DE VERSUS TAN, TAN-US AND TAN-SMOTE WHEN COST RATIO=20.

Dataset	TAN-DE	TAN	TAN-US	TAN-SMOTE
ecoli-0-vs-1	5.26± 9.05	6.49±10.23	4.98± 8.98	4.18± 8.06
ecoli1	12.23±10.55	19.51±21.32	12.63±12.90	21.77±19.63
ecoli2	14.79±13.17	29.83±20.21 ○	17.28±16.23	28.18±18.69 ○
ecoli3	10.17± 8.89	36.70±16.69 ○	12.10±11.74	18.66±15.84
glass-0-1-2-3	6.20± 9.42	20.53±18.77 ○	11.22±13.59	12.27±13.50
glass0	20.69±17.16	31.84±21.94	22.27±21.26	28.33±21.43
glass1	24.60±18.30	72.47±28.78 ○	60.26±30.46 ○	55.73±28.63 ○
glass6	8.39±11.13	13.36±14.07	9.26±11.86	10.40±12.18
haberman	40.43±21.72	130.06±36.36 ○	74.26±38.82 ○	78.42±27.55 ○
iris0	0.00± 0.00	0.00± 0.00	0.00± 0.00	0.00± 0.00
new-thyroid1	1.64± 4.35	6.61±10.92	3.38± 7.61	2.72± 7.36
newthyroid2	2.59± 6.03	10.01±12.79	2.38± 6.46	4.77± 9.76
page-blocks0	82.50±31.40	144.61±49.46 ○	102.35±40.65	129.06±43.19 ○
pima	115.18±43.44	222.45±49.72 ○	145.61±48.81 ○	162.28±51.63 ○
segment0	8.34±11.38	16.57±16.67	10.22±12.15	13.03±15.41
vehicle0	17.33±13.50	97.94±44.43 ○	55.30±34.95 ○	50.30±27.58 ○
vehicle1	108.56±36.31	239.83±46.77 ○	160.82±45.55 ○	187.92±43.59 ○
vehicle2	19.29±16.12	54.50±28.99 ○	29.29±21.18	34.61±27.30
vehicle3	118.39±39.37	276.50±46.17 ○	148.59±48.37	191.13±43.95 ○
wisconsin	5.12± 7.29	11.30±12.79	9.48±12.80	12.97±15.23
yeast1	161.92±52.75	521.34±69.80 ○	305.97±83.02 ○	341.42±65.35 ○
yeast3	34.02±15.42	73.24±37.55 ○	43.20±25.88	61.75±32.44 ○
Average	37.17	92.53	56.40	65.90
W/T/L	-	13/9/0	6/16/0	10/12/0

From these test results, we can see that, compared to the original cost-insensitive classifiers, our proposed DE algorithm achieves a noticeable gain of performance by introducing the differential evolution-based preprocessing operator. Besides, our proposed DE significantly outperforms all the other comparative approaches. Now, we summarize the high lights as follows:

- Based on the NB classifier, the average rankings of them are NB-DE (1.3864), NB (3.3864), NB-US (2.2955), and NB-SMOTE (2.9318) respectively. According to the Bergmann test, in terms of the total misclassification costs, the performance of NB-DE is notably better than all the other existing algorithms: NB, NB-US and NB-SMOTE.
- Based on the TAN classifier, the average rankings of TAN-DE (1.2045) is much higher than those of TAN (3.8409), TAN-US (2.1136), and TAN-SMOTE (2.8409). According to the Bergmann test, in terms of the total misclassification costs, the performance of TAN-DE is significantly better than all the other existing algorithms: TAN, TAN-US and TAN-SMOTE.
- Our proposed differential evolution-based approach is much better than the existing random under-sampling approach and the over-sampling approach by creating "synthetic" examples in the minority class, called the SMOTE algorithm.

3) Influence on DE with different cost ratios: Table IX shows the **W/T/L** counts of DE versus its competitors under various cost ratios (5, 10, 15, 20) in terms of the total misclassification costs. The experimental results show that our proposed

DE-based method is notably better than its competitors under various cost ratios. As the cost ratio increases, the advantage of our sampled method (NB-DE and TAN-DE) becomes more apparent. For example, when cost ratio is 5, NB-DE slightly outperforms NB-US and NB-SMOTE only on 4 and 3 datasets respectively, TAN-DE slightly outperforms TAN-US and TAN-SMOTE only on 3 and 4 datasets respectively.

V. Conclusion

Many domains involve the problems of class-imbalanced cost-sensitive learning. That is, when the class distribution of examples is imbalanced and the costs of misclassification are different, it can be challenging to identify the examples of the minority class effectively. Many techniques have been proposed for addressing the related problems. In this paper, we present a differential evolution-based method, simply denoted by DE, for class-imbalanced cost-sensitive learning. Instead of random under-sampling, our proposed DE method employs differential evolution to guide the selection of the optimal subset for reducing the total misclassification costs.

Our experiments compare the performance of DE to its competitors: US and SMOTE in terms of the misclassification costs. In our experiments, NB and TAN are chosen as the base classifiers. The experiments results on 22 class-imbalanced data sets from various levels of imbalance show that our proposed DE-based method significantly outperforms the original cost-insensitive algorithms. More important, our proposed DE method is notably better than the widely used random under-sampling (US) and the well-known synthetic minority over-sampling technique (SMOTE) under various cost ratios (5, 10, 15, 20) in terms of the total misclassification costs.

Acknowledgment

The work was partially supported by the National Natural Science Foundation of China (61203287), the Program for New Century Excellent Talents in University (NCET-12-0953), the Chenguang Program of Science and Technology of Wuhan (2015070404010202), and the Fundamental Research Funds for the Central Universities (CUG130504, CUG130414).

References

[1] P. C. R. Lane, D. Clarke, P. Hender, "On developing robust models for favourability analysis: Model choice, feature sets and imbalanced data," Decision Support Systems, vol. 53, pp. 712-18, 2012.

[2] P. N. Tan, M. Steinbach and V. Kumar, "Introduction to data mining," Pearon Education.

[3] N. V. Chawla, "Data Mining for Imbalance Datasets: An Overview," Data Mining and Knowledge Discovery Handbook, vol. 5, pp. 853-867, 2006.

[4] H. He, E. Garcia, "Learning from imbalanced data," IEEE Transactions on Knowledge and Data Engineering, vol. 21, no. 9, pp. 1263-1284, 2009.

[5] C. X. Ling, V. S. Sheng, "Cost-sensitive leanring and the class imbalance problem," Encyclopedia of Machine Learning, Springer, pp. 231-235, 2008.

[6] G. E. A. P. A. Batista, R. C. Prati, M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," SIGKDD Explorations Newsletter, vol. 6, pp. 20-29, 2004.

[7] Z. H. Zhou, X. Y. Liu, "Training cost-sensitive neural networks with methods addressing the class imbalance problem," Knowledge and Data Engineering, IEEE Transactions on, vol. 18, no. 1, pp. 63-77, 2006.

[8] L. Jiang, C. Li, Z. Cai, H. Zhang, "Sampled bayesian network classifiers for class-imbalance and cost-sensitive learning," 2013 IEEE 25th International Conference on, IEEE, pp. 512-517, 2013.

[9] R. Storn, K. Price, "Differential evolution- A simple and efficient heuristic for global optimization over continuous spaces," Journal of Global Optimization, vol. 11, pp.341-359, 1997.

[10] K. Price, R. Storn, J. Lampinen, "Differential evolution: A practical approach for global optimization," Springer-Verlag, 2005.

[11] A. Fernndez, S. Garcła, M. J. del Jesus, et al., " A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets," Fuzzy Sets and Systems, vol. 159, no. 18, pp. 2378-2398, 2008.

[12] J. Alcal-Fdez, L. Snchez, S. Garcła, et al., "KEEL: a software tool to assess evolutionary algorithms for data mining problems," Soft Computing, vol. 13, no. 3, pp. 307-318, 2009.

[13] J. Alcal-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. Garcła, L. Snchez, F. Herrera, "KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework," Journal of Multiple-Valued Logic and Soft Computing, vol. 17, pp. 255-287, 2011.

[14] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," Journal of Artificial Intelligence Research, vol. 16, pp. 321-356, 2002.

[15] J. Van Hulse, T. Khoshgoftarr, A. Napolitano, "Experimental Perspectives on Learning from Imbalanced Data," In: Proceedings of the 24nd International Conference on Machine Learning, pp. 935-942, 2007.

[16] K. Napierala, J. Stefanowski, S. Wilk, "Learning from imbalaned data in presence of noisy and borderline examples," RSCTC 2010. LNCS, vol. 6086, pp. 158-167. Springer, Heidelberg, 2010.

[17] A. Estabrooks, T. Jo, N. Japkowicz, "A multiple resampling method for leanring from imbalanced data sets," Computational Intelligence, vol. 20, no. 1, pp. 18-36, 2004.

[18] N. Japkowicz, S. Stephen, "The class imbalance problem: A systematic study," Intelligent Data Analysis, vol. 6, no. 5, pp. 429-250, 2002.

[19] G. Weiss, F. Provost, "Learning when training data are costly: the effect of class distribution on tree induction," Journal of Artifial Intelligence Research, vol. 19, pp. 315-354, 2003.

[20] P. Langley, W. Iba, and K. Thomas, "An analysis of Bayesian classifiers," In Proceedings of the Tenth National Conference of Artificial Intelligence, pp. 223C228. AAAI Press, 1992.

[21] N. Friedman, D. Geiger, M. Goldszmidt, "Bayesian network classifiers," Machine Learning, vol. 29, pp. 131-163, 1997.

[22] G. I. Webb, J. Boughton, Z. Wang, " Not so naive Bayes: Aggregating one-dependence estimators," Journal of Machine Learning, vol. 58, no. 1, pp. 5-24, 2005.

[23] L. Jiang, H. Zhang, Z. Cai, and D. Wang, "Weighted Average of One-Dependence Estimators," Journal of Experimental & Theoretical Artificial Intelligence, vol. 24, no. 2, pp: 219-230, 2012.

[24] L. Jiang, H. Zhang, and Z. Cai, "A Novel Bayes Model: Hidden Naive Bayes," IEEE Transactions on Knowledge and Data Engineering, vol. 21, no. 10, pp: 1361-1371, 2009.

[25] I.H. Witten, E. Frank, "Data Mining: Practical machine learning tools and techniques," Morgan Kaufmann, 2005.

[26] X. Y. Liu, Z. H. Zhou, "The influence of class imbalance on cost-sensitive learning: An empirical study," Data Mining, ICDM'06. Sixth International Conference on. IEEE, pp. 970-974, 2006.

[27] J. Lampinen, I. Zelinka, "Mechanical engineering design optimization by differential evolution," New ideas in optimization. McGraw-Hill Ltd., UK, pp. 127-146, 1999.

[28] S. Das, P N. Suganthan, "Differential evolution: a survey of the state-of-the-art," Evolutionary Computation, IEEE Transactions on, vol. 15, no, 1, pp. 4-31, 2011.

[29] S. Garcia, J. R. Cano, A. Fernandez, F. Herrera, "A proposal of evolutionary prototype selection for class imbalance problems," In IDEAL, pp. 1415-1423, 2006.

[30] S. Garcia, F. Herrera, "Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy," Evolutionary Computation, vol. 17, no. 3, pp. 275-306, 2009.

TABLE IX.
COMPARISON OF THE TOTALS MISCLASSIFICATION COSTS UNDER VARIOUS COST RATIOS ON 22 CLASS-IMBALANCED DATA SETS.

Algorithms	Cost ratio=5	Cost ratio=10	Cost ratio=15	Cost ratio=20
NB-DE vs NB	7/14/0	11/11/0	11/11/0	11/11/0
NB-DE vs NB-US	4/18/0	6/16/0	7/15/0	7/15/0
NB-DE vs NB-SMOTE	3/19/0	8/14/0	10/12/0	9/13/0
TAN-DE vs TAN	8/14/0	12/10/0	13/9/0	13/9/0
TAN-DE vs TAN-US	3/19/0	4/18/0	5/17/0	6/16/0
TAN-DE vs TAN-SMOTE	4/18/0	7/15/0	10/12/0	10/12/0

[31] M. Galar, A. Fernandez, E. Barrenechea, et al., "Eusboost: enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling," Pattern Recognition, vol. 46, no. 12, pp. 3460-3471, 2013.

[32] G. Y. Pan, F. Min, W. Zhu, "A genetic algorithm to the minimal test cost reduct problem," Proceedings of IEEE International Conference on Granular Computing, pp. 539-544, 2011.

[33] Z. Xu, F. Min, J. B. Liu, W. Zhu, "Ant colony optimization to minimal test cost reduction," Proceedings of the 2011 IEEE International Conference on Granular Computing, pp. 688-693, 2012.

[34] B. Xu, F. Min, W Zhu, et al., "A Genetic Algorithm to Multi-objective Cost-sensitive Attribute Reduction," Journal of Computational Information Systems, vol. 10, no. 7, pp. 3011-3022, 2014.

[35] C. Nadeau, Y. Bengio, "Inference for the generalizaiton error," Machine Learning, vol. 52, no. 3, pp: 239-281, 2003.

[36] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," The Journal of Machine Learning Research, vol. 7, pp. 1-30, 2006.