Constrained Oversampling: An Oversampling Approach to Reduce Noise Generation in Imbalanced Datasets With Class Overlapping

CHANGHUI LIU1, SUN JIN2, DONGHONG WANG3, ZICHAO LUO4, JIANBO YU1, BINGHAI ZHOU1, AND CHANGLIN YANG5

1School of Mechanical Engineering, Tongji University, Shanghai 201804, China
2School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
3School of Materials Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
4Yoshino and Yamamoto Laboratory, Tokyo Institute of Technology, Tokyo 152-8550, Japan
5State Key Laboratory of Solidification Processing, Northwestern Polytechnical University, Xi’an 710072, China

Corresponding author: Jianbo Yu (jbyu@tongji.edu.cn)

This work was supported in part by the National Science and Technology Major Project under Grant 2017-VII-0008-0102, in part by the China Postdoctoral Science Foundation under Grant17Z102060069, in part by the National Natural Science Foundation of China under Grant 51705314 and Grant 51821001, in part by the Zhejiang Province Postdoctoral Project under Grant 228188, and in part by the Startup Fund for Youngman Research, Shanghai Jiao Tong University, under Grant 18× 1000040025.

ABSTRACT

Imbalanced datasets are pervasive in classification tasks and would cause degradation of the performance of classifiers in predicting minority samples. Oversampling is effective in resolving the class imbalance problem. However, existing oversampling methods generally introduce noise examples into original datasets, especially when the datasets contain class overlapping regions. In this study, a new oversampling method named Constrained Oversampling is proposed to reduce noise generation in oversampling. This algorithm first extracts overlapping regions in the dataset. Then Ant Colony Optimization is applied to define the boundaries of minority regions. Third, oversampling under constraints is employed to synthesize new samples to get a balanced dataset. Our proposal distinguishes itself from other techniques by incorporating constraints in the oversampling process to inhibit noise generation. Experiments show that it outperforms various benchmark oversampling approaches. The explanation for the effectiveness of our method is given by studying the impact of class overlapping on imbalanced learning.

INDEX TERMS

Constrained oversampling, oversampling, class overlapping, imbalanced dataset.

I. INTRODUCTION

The class imbalance problem occurs when some of the classes in a dataset have significantly more samples than the others. The discrepancy in sample number between classes brings about the imbalanced data structure in such kinds of datasets. Researchers have reported that this imbalanced structure hampers learning, resulting in an undesirable performance of data mining algorithms [1]–[4].

Imbalanced datasets exist in many real-world classification applications, such as detecting oil spills in satellite radar images [5], conducting medical diagnosis [6], detecting credit card frauds [7], analyzing neuroimaging data [8], predicting binding site [9] and so on. By convention, classes with a larger quantity of samples are called the negative classes or majority classes, while the others are referred to as the positive classes or minority classes [10]. In these domains, minority samples (e.g. oil spills) are often the ones of interest and misclassifying minority samples is costly [3], [4]. However, traditional classification algorithms, constructed under the assumption or expectation of balanced data distribution, are usually inadequate in tackling imbalanced classification problems. Disappointing performances of classifiers are often derived on recognizing and predicting minority instances because the imbalanced structure of training set biases the decision boundary of the classifiers toward the majority class. Class imbalance issue has received considerable attention and is identified as one of the main challenges in data mining [11].

Methods designated to remedy this pervasive problem roughly fall into two categories: data-based methods [12]–[15], and algorithm-based methods [16]–[19]. Algorithm-based methods attempt to modify learning algorithms to adjust the classifier to imbalanced datasets [20]–[22]. They rely...
on selected classifiers so once we demand the use of a different classification algorithm, the extra computation cost is inevitable [23]. Moreover, the research presented by Maloof suggests that operations of re-sampling and adjusting classification algorithms by varying the cost matrix produce similar sets of classifiers [24]. Thus, we narrow the scope of our study to data-level solutions to imbalanced classification problem without loss of generality.

Data-based methods usually refer to re-sampling approaches that preprocess datasets to achieve a balanced data structure. Before data-level approaches are applied, we should notice that datasets differ from each other in the number of classes: binary-class datasets contain only one majority class and one minority class while multi-class datasets compose of several majority classes and minority classes. In multiple-class conditions, we can either decompose the problem into multiple two-class classification problems [25] or extend the schemes developed on binary-class scenarios to multi-class problems via pair-wise coupling techniques [26]. Therefore, we focus our efforts on imbalanced classification with binary classes the same as most researches in the literature.

Data-level strategies on binary-class imbalanced classification problems can be further divided into undersampling and oversampling. Undersampling removes samples belonging to the majority class according to designed criteria to reduce the degree of imbalance in the dataset. The least intricate undersampling proposal is Random Under Sampling (RUS) which randomly discards samples in the majority category to adjust the imbalanced data distribution. One of the more advanced undersampling techniques is the One-Sided Selection proposed by Kubat and Matwin [27]. This method detects and then deletes noisy and redundant samples in the majority category and leaves minority category untouched. By choosing a representative subset of the negative examples, it balances the dataset. Thus, the classifier learns a clean boundary and shows better predicting performances. Undersampling based on clustering put forward by Yen and Lee [28], is an approach that clusters all the training samples in the first step. Then this method decides the number of majority instances that should remain in each cluster according to the ratio of the number of majority samples to the number of minority samples in the particular cluster. In the last step, it removes the calculated amount of majority samples in each cluster to mitigate the imbalance between classes.

ACOSampling, presented by Yu et al. [29], employs an ant colony optimization algorithm to remove less informative instances and search for optimal subsets of a randomly divided part of the original dataset. By doing that repeatedly, the statistical results from all local optimal training subsets are achieved and presented in a frequency list. Then high-frequency majority samples are extracted and they are combined with the original minority set to form a new training set.

Unlike undersampling, oversampling concentrates on increasing the number of minority samples. Japkowicz and Stephen [30] conducted a series of experiments on how re-sampling methods perform on data sets of different complexity and concluded that undersampling methods appear to be less effective than oversampling when two classes were assigned a symmetric cost. Meanwhile, oversampling was shown to help quite dramatically at all complexity and training set size levels. Similar results were reported by Bastista et al. [31]. We are more concerned with oversampling methods because they are reported to be more effective and are also more related to our work.

Random Over Sampling is a simple oversampling method that replicates minority class samples at random to get prescribed imbalance level. It was pointed out in previous work that ROS is prone to incur overfitting problem in learning [32]. Based on this intuitive method, more sophisticated oversampling techniques have been developed in researches. Some of the improved methods are as below.

Synthetic Minority Oversampling Technique (SMOTE), proposed by Chawla et al., forms new minority samples by linearly interpolating between minority samples that lie close to each other in feature space [33]. Despite its efficaciousness, this method has a major shortcoming: it blindly generates new samples for minority class examples without considering the distribution of original data, so sometimes noise samples are added into the dataset. This problem can be more serious when the original data set has one or more overlapping regions or holds noise samples in the original minority class.

Han et al. developed an oversampling technology called Borderline-SMOTE based on the observation that misclassified samples usually located on the borderline between minority and majority class [34]. In this proposal, only minority samples that lie near the borderline are operated on, and thus learning on the borderline is reinforced. There were two versions for their proposal: Borderline-SMOTE1 which only generates samples among borderline instances belonging to minority class and Borderline-SMOTE2 which also synthesizes examples between minority samples on the boundary and their nearest negative neighbors.

Another improved oversampling method is Adaptive Synthetic Sampling (ADASYN), which was presented by He et al. [35]. This method considers the original distribution. This method uses the number of samples that belong to the majority class in k nearest neighbors of a minority sample as a criterion to automatically judge the number of samples synthesized around this specific minority sample. More samples would be generated around instances situate close to the majority class region than those far away from the borderline between classes.

Cluster Based Synthetic Oversampling (CBOS), devised by Barua et al. [36], attempts to avoid noise generation by first conducting unsupervised clustering among minority samples and then carrying out SMOTE process inside the clusters. This data generation mechanism avoids the creation of synthetic minority samples in the majority region under some circumstances. However, when overlapping areas between classes occur in the dataset, majority samples would also
be included inside the clusters. Thus, oversampling in these clusters generates minority samples that fall into regions belonging to the majority class and leads to misclassification of majority samples.

Safe-level-SMOTE assigns each positive instance a safe level ratio, which is the ratio of positive samples in k nearest neighbors of that instance [37]. Synthetic samples are generated along the same line segment as SMOTE, but these samples are placed closer to minority instances to which higher safe level ratios are attached. This approach is effective in eliminating noise generation; however, it does not help recognize samples in an overlapping region since it generates no samples in this region. As a result, the usefulness of this proposal is weakened by the fact that overlapping between classes is pervasive in real-world datasets.

All in all, oversampling is widely investigated and used as a remediation strategy to cope with imbalanced datasets. By far, researchers have developed a batch of advanced oversampling methods, such as SMOTE, Borderline-SMOTE, and so on. They have been applied to many real-world imbalanced datasets and are shown to be effective in improving the performance of classifiers. But unfortunately, though it was pointed out that class overlapping is a main cause of misclassification in imbalanced datasets [38]; previous researchers have not yet devised competent oversampling methods to deal with class overlapping regions as far as we know.

In class overlapping regions, majority samples mingle with minority samples so the boundaries between categories become ambiguous. Since classification algorithms are designed to learn the borderline of each class, they are less powerful in overlapping areas where borderlines seem to be confusing. Consequently, the examples situated in these regions are prone to be misclassified. Poor learning in overlapping regions calls for the demand of oversampling in these regions to improve the performance of classifiers. However, most existing oversampling approaches rest on the assumption that samples close to each other in feature space belong to the same class, which is not the case in overlapping regions. So, when existing oversampling methods are applied to these overlapping regions, noise minority samples that fall into the majority region are introduced into the datasets. These noise samples are detrimental to the performance of classification algorithms. As a result, traditional oversampling approaches show a deficiency in handling datasets with overlapping between classes.

In this study, we introduce an oversampling method that distinguishes itself from other oversampling techniques by incorporating constraints in the oversampling process to inhibit noise generation in overlapping regions. This method, namely Constrained Oversampling (CO), composes of three steps. First, samples placed in the overlapping area are identified with a simple k-nearest-neighbors (KNN) based algorithm. Second, in these overlapping regions, we apply the ant colony optimization (ACO) algorithm to search for feasible paths from randomly chosen majority samples to a specific minority sample, and the majority points on the paths which are closest to the destination are picked out as boundary samples. Finally, oversampling under distance constraints is performed between boundary samples and the chosen minority point to attain a more balanced dataset.

The rest of this paper is organized as follows. The proposed constrained oversampling method is described in Section II. In Section III, experiments are conducted on various real-world datasets from the UCI repository to test this method, and the results are presented and discussed. At last, Section IV concludes our work.

II. A NEW OVERSAMPLING METHOD: CONSTRAINED OVERSAMPLING

In order to balance the dataset without producing noise samples, we execute oversampling in overlapping regions, and the oversampling process; constraints are used to prevent noise generation. As is depicted in Fig. 1, our strategy is divided into three stages.

A. THE EXTRACTION OF OVERLAPPING AREA

In the first step of our method, we build a KNN-based method to extract overlapping regions included in the original dataset. As a geometric classifier, the KNN classifier determines the class membership of a data point from its distance to reference data points [39], and it is widely used in clustering [40]–[42]. Thus, KNN is a density-based method to classify the input data by the percentage of samples belonging to a different class in their nearest neighbors. In overlapping regions, minority and majority samples mix with each other. By the definition of KNN, minority samples locate in overlapping regions are more likely to be misclassified. This characteristic of KNN can be taken advantage of to identify samples locate in overlapping regions. When the minority samples in overlapping regions are classified to the majority class by the KNN classifier, we can know they are misclassifications because the data labels are known in our study.

First, we apply a 5-NN classifier to the original data set as recommended by Mani and Zhang [43] and record the misclassified minority samples. For every misclassified minority sample, we choose its m nearest neighbors to form an overlapping set. If there are n misclassified minority samples, then we get m * n data points. In these data points, there are some duplicate samples. The overlapping set O is
Algorithm 1 The ACO-Based Method to Extract Overlapping Regions

1: **Input**: Original training set $S$, controlling parameter of the scope of overlapping region $m$.
2: **Process**:
3:   for $i = 1$: number of minority samples in $S$
4:       find 5 nearest neighbors for minority sample $Min_i$
5:       if (number of positive samples) $< 5/2$
6:           Add $Min_i$ and its $m$ nearest neighbors in overlapping set $O$
7:       endif
8:   endfor
9:   Remove duplicate samples in $O$
10: **Output**: Overlapping set $O$

extracted when these duplicate samples are removed. The similarity metric of the KNN algorithm adopted in this paper is Euclidean distance. The Pseudo-code description of this step is summarized as Algorithm 1.

It should be noted that the total number of overlapping regions do not need to define first. It is determined by the distribution of the misclassified minority samples. It forms when the overlapping set is extracted.

To better illustrate our method, we apply it to a simulated dataset with two features. Fig. 2 (a) displays the binary-class imbalanced dataset with two overlapping regions which consist of 800 majority samples (represented by green circles) and 80 minority instances (represented by red pluses). After the above-mentioned method is applied to the dataset, overlapping regions which contain 191 samples are extracted and shown in Fig. 2 (b). According to the distribution of the misclassified minority samples, two overlapping regions are generated (shown in Fig. 2 (b)). The following steps of our oversampling method will be carried out on these overlapping regions.

**B. BOUNDARY DEFINITION BASED ON ANT COLONY OPTIMIZATION ALGORITHM**

Traditional boundary definition methods such as Tomek-links [27] and KNN are density-based approaches that judge the boundary samples by the percentage of samples belonging to a different class in their nearest neighbors. In overlapping regions where minority and majority samples mix with each together, traditional boundary definition methods are prone to take all majority samples in the overlapping regions as boundary samples and are thus imprecise [19].

We developed an ACO-based method to define the boundaries between classes in extracted overlapping regions. With this method, the boundaries between majority and minority regions are expressed in the form of a set of majority samples. Our proposal embodies two major merits: on one hand, it is more robust to noise in the original minority set since it does not solely depend on information conveyed by minority samples; on the other hand, it is advantageous in expanding decision region of minority category to identify boundary majority samples and then introduce them into the oversampling process.

In defining the boundaries, we require an algorithm that can automatically seek feasible itineraries from one class to the other. Ant colony optimization, developed by Colomi et al. [44], has been reported to perform well in solving various discrete combinatorial optimization problems, such as traveling salesman problem [45], protein folding [46], fingerprint matching [47], fuzzy identification [48], map matching [49], route improving [50]. This population stochastic search method was inspired by the communication of ants during the foraging process. Its ability to find an optimal path from nest to food in discrete space through iterative search is what we demand in defining the boundaries surrounding minority regions.

In this work, the ant system version proposed by Dorigo et al is adopted [51] and in particular, the sight of ants is limited to 3 (i.e. an ant can only search in the area of 3 nearest points at once to figure out a way toward the food source). In each iteration of our ACO-based method, $t$ different majority samples are randomly chosen from the pre-defined overlapping region as well as one minority example. Then $na$ ants are put on each majority instance and are ordered to search for the shortest path to that particular minority sample. We get $t$ paths from which we can get to that minority sample from the majority area and then the last station in majority region is picked out and added to boundary set. After each minority sample has been taken as the destination once, the boundaries between majority and minority categories are constructed and described with
FIGURE 3. (a) Pick out three starting majority points (represented by magenta solid circles) and a destination in minority class (displayed by the red solid circle) (b) Generate a feasible path generated with ACO (represented by the black line) (c) Choose the last station in majority region as a boundary point for the particular minority sample (represented by the solid black circle).

Algorithm 2 The ACO-Based Method to Define the Boundaries Between Classes in Extracted Overlapping Regions

1: **Input:** Overlapping set \( O \), number of ants on each node \( n_a \), max iteration number \( NC_{\text{max}} \), number of starting points the ACO-based method in majority region \( t \)

2: **Process:**

3: for \( i = 1 \) : number of minority samples in \( O \)

4: Randomly choose \( t \) different majority samples as ‘nests’

5: for \( j = 1 : t \)

6: Start from majority instance \( \text{nest}_j \) and \( n_a \) ants are simultaneously ordered to search for the shortest path to minority sample \( \text{Min}_i \) using ACO

7: After \( NC_{\text{max}} \) iterations, the optimal path is saved as \( \text{path}_{ij} \)

8: The last majority sample in this path is stored as \( \text{boundary}_{ij} \)

9: endfor

10: endfor

11: **Output:** A set of majority samples defining the borderline of minority region \( \text{boundary}_{ij} \)

route is the optimal one or only one of the sub-optimal solutions.

**C. THE CREATION OF SYNTHETIC SAMPLES BY OVERSAMPLING UNDER CONSTRAINTS**

In step 3, we construct the final training set by oversampling between the boundary samples and the minority instances in overlapping regions.

To reduce the generation of synthetic minority samples in majority regions, we impose distance constraints to the oversampling process. The aim of applying constraints to oversampling is to constrain new samples to the neighborhood of existing minority samples but not to that of majority samples.

Before oversampling, we shall deliberate over three questions: which samples are worthy of oversampling? How many sample points shall be created for each chosen minority sample? Where to place the generated minority samples? Question 1 is responded to in Subsection A and B of Section II. In terms of the second question, ADASYN [26] is integrated into our method to decide the number of new synthetic samples generated for each minority sample in overlapping regions. And question 3 can be answered by the constraints demonstrated as follows.

When a sample between a minority sample in the overlapping regions \( \text{Min} \) and a majority sample belonging to the boundary set \( \text{Maj} \) need to be generated, we denote the Euclidean distance between these two instances as \( \text{Dist} (\text{Min}, \text{Maj}) \), the distance between \( \text{Min} \) and its 5th nearest neighbor as \( \text{Dist} (\text{Min}, 5^{th} \text{NN of Min}) \) and that between \( \text{Maj} \) and its 5th nearest neighbor as \( \text{Dist} (\text{Maj}, 5^{th} \text{NN of Maj}) \). The distance between the new synthetic...
sample and \( \text{Min} \), \( \text{Dist} \), is decided according to rules depicted in Fig. 4.

After \( \text{Dist} \) is calculated, the new sample is synthesized according to:

\[
\text{Dir} = (\text{Maj} - \text{Min})/\text{norm of vector(\text{Maj} - \text{Min})} \quad (1)
\]

\[
\text{Min}_\text{new} = \text{Min} + \alpha \times \text{Dist} \times \text{Dir}, \quad \alpha \in (0, 1) \quad (2)
\]

where \( \text{Min}_\text{new} \) refers to the synthetic instance, \( \text{Dir} \) is a direction vector. The illustrations of the location of the newly synthesized sample on different occasions are as in Fig. 4.

In Subsection A, we applied a 5-NN classifier to the original data set as recommended by Mani and Zhang [43] to extract the overlapping area. For this data set, 5-NN has the best performance in classifying the original data. It means that 5-NN reflects the distribution characteristic of the original data very well when classifying the data. In the creation of synthetic samples for the minority samples, we increase the density of the minority samples in the overlapping area but the distribution characteristic of the original data should be inherited. Therefore, when we define the \( \text{Dist} \) the 5th NN of \( \text{Maj} \) or \( \text{Min} \) is chosen. In this way, when we classify the newly generated data \( \text{Min}_\text{new} \) it will be classified with a low error rate.

The whole procedure of this stage is described in detail as Algorithm 3.

Algorithm 3 The Constrained Oversampling Method

1: Input: Original data set \( S \), borderline sample set \( \text{boundary}_{i,j} \), minority set in overlapping region \( S_{\text{min}} \), the majority set in overlapping region \( S_{\text{maj}} \), amount of SMOTE \( N\% \), number of nearest neighbors \( k \)
2: Process:
3: Calculate the number of minority samples to be generated: \( G = N\% \times \text{number of minority samples in original data set } S \)
4: for \( i = 1: \text{number of samples in } S_{\text{min}} \)
5: \hspace{1cm} Find \( k \) nearest neighbors based on the Euclidean distance and get the ratio: \( r_i = \text{number of majority samples in } k \text{ nearest neighbors } / k \)
6: \hspace{1cm} endfor
7: for \( i = 1: \text{number of samples in } S_{\text{min}} \)
8: \hspace{1cm} Calculate the number of synthetic samples \( g_i \) that need to be generated for each minority sample according to: \( g_i = r_i \times G / \text{(sum of all } r_i) \)
9: \hspace{1cm} endfor
10: for \( u = 1: g_i \)
11: \hspace{1cm} Except for samples that have been selected in the selected sample set \( S_{\text{selected}} \), randomly choose a minority sample in overlapping region \( \text{Min}_i \), retrieve its \( t \) corresponding majority samples on the borderline in \( \text{boundary}_{i,j} \), randomly pick one out and denote it by \( \text{Maj}_i \).
12: \hspace{1cm} Record the minority sample selected in this iteration to the selected sample set \( S_{\text{selected}} \) so that the samples in this iteration will not be re-selected in the next iterations.
13: \hspace{1cm} Generate a new sample \( \text{Min}_\text{new} \) between \( \text{Min}_i \) and \( \text{Maj}_i \) according to the rules depicted in Fig. 4
14: \hspace{1cm} Add \( \text{Min}_\text{new} \) to \( S_{\text{new}} \)
15: \hspace{1cm} endfor
16: \hspace{1cm} endfor
17: \hspace{1cm} endfor
18: \hspace{1cm} Merge \( S_{\text{new}} \) and original set \( S \) to form a new training set \( S_{\text{balanced}} \)
19: Output: New balanced training set \( S_{\text{balanced}} \)

In the way mentioned above, to enhance learning on minority areas, newly generated minority samples are placed in overlapping regions where learning and predicting minority samples are difficult. Meanwhile, learning on majority regions is not hindered since these synthesized minority samples would not become noise points situated across the boundaries between classes. Our constrain-based oversampling method is powerful in handling class overlapping regions where traditional clustering-based methods do not perform well.

III. EXPERIMENT

In this section, we conducted experiments on five UCI datasets [52] with different numbers of features and imbalance levels. To investigate the performance of the
proposed method, six benchmark oversampling strategies described in Section II, namely original data without oversampling (Origin), SMOTE, Borderline-SMOTE1 (BOS1), Borderline-SMOTE2 (BOS2), ADASYN, CBOS were applied to the datasets as well as Constrained Oversampling (CO) and the results were compared. A well-known binary decision tree learner called Classification And Regression Tree (CART) [53] was chosen as the test classifier in these experiments.

**A. DATASETS**

Among the five benchmark datasets from UCI Repository that were used in our tests, Pima and Haberman are composed of binary-class samples and the others are multi-class. In multi-class problems, we selected one of the classes as the minority class, and the remaineders were merged into the majority class. Table 1 shows the characteristics of the five data sets, namely label of the minority class, number of attributes, number of minority instances and number of majority samples, sorted by the number of minority instances in ascending order. Particularly, imbalance level (IL) in the table refers to the ratio of the number of majority examples to that of minority examples.

**B. EVALUATION METRICS AND PARAMETERS**

The performance of a classifier can simply be shown by the “raw data” produced during testing – the counts of correct and incorrect classifications. Generally, this information would be displayed in a confusion matrix as follows.

In Table 2, \( Tp \) and \( Tn \) are the numbers of true positives and true negatives, respectively. \( Fp \) and \( Fn \) are the numbers of false positives and false negatives, respectively.

A widely adopted measurement of classification performance is overall accuracy calculated as follows:

\[
Overall\text{Accuracy} = \frac{(Tp + Tn)}{(Tp + Fn + Fp + Tn)} \tag{3}
\]

However, in imbalanced data sets, accuracy behaves poorly as a metric to evaluate the performance of classifiers over minority examples. When a data set is highly imbalanced, we attain a high accuracy even if all of the minority examples are misclassified. To overcome the problem mentioned above, other statistics such as \( G-mean \) and \( F-measure \) are also introduced into the assessment of classifiers. They are calculated as follows:

\[
G-mean = \sqrt{Acc^+ \cdot Acc^-} \tag{4}
\]

\[
F-measure = (1 + \beta^2) \cdot \frac{Recall \cdot Precision}{\beta^2 \cdot Recall + Precision} \tag{5}
\]

where \( \beta \) corresponds to relative importance of precision versus recall and is usually set to 1. Both \( F-measure \) and \( G-mean \) are values between 0 and 1. Larger \( F-measure \) and \( G-mean \) indicate better performance. And \( Precision, Recall, Acc^- \) and \( Acc^+ \) are further defined as:

\[
Precision = \frac{Tp}{(Tp + Fp)} \tag{6}
\]

\[
Recall = \frac{Acc^+}{Tp/(Tp + Fn)} \tag{7}
\]

\[
Acc^- = \frac{Tn}{(Tn + Fp)} \tag{8}
\]

They are more reasonable indications of overall precision than accuracy due to some of their unique properties. One is that the magnitude of either \( G-mean \) or \( F-measure \) relies heavily on how the classifier performs on the minority class.

It is determined that in this paper, Overall Accuracy, \( G-mean, F-measure \), and \( Acc^+ \) would all be calculated to provide a comprehensive understanding of the performance of the classifier on given datasets. The initial parameters of the proposed method in this study are listed in Table 3.

**C. RESULTS**

For each dataset presented in Subsection A of Section III, the minority class was over-sampled at 100%, 200%, 300%, 400% and 500% of its original size. We conducted 3 independent 10-fold cross-validation experiments at each percentage of oversampling and the final results were the average value of 3 tests [54]. In each test, all 7 different oversampling strategies — Origin, SMOTE, BOS1, BOS2, ADASYN, CBOS, and CO were applied to the same datasets. Performances of different methods on datasets oversampled at 200% are shown in Table 4, where four different metrics, Overall Accuracy, \( G-mean, Acc^- \), and \( F-measure \) are listed and compared. Other results are displayed in Fig. 5 to 9.

**D. DISCUSSION**

We present the discussion of the results displayed above in two aspects: the impact of oversampling rate on the behavior of oversampling methods and the influence of class overlapping to the classifier and oversampling methods:

1) THE IMPACT OF OVERSAMPLING RATE ON THE BEHAVIOR OF OVERSAMPLING METHODS

Generally speaking, a larger oversampling rate leads to a more advantageous dataset for learning the minority samples. This point is justified by Fig. 5 (c) to Fig. 9 (c). However, the progress in the performance of predicting minority is usually at the cost of more misclassified majority samples. As is depicted in Fig. 5 (a) and Fig. 6 (a), more synthesized samples sometimes mean more unreliable or noise minority instances that fall into the majority region and lead to the misclassification of majority samples. So a larger oversampling rate

---

**TABLE 1. Datasets description.**

| Dataset | Label of Minority Class | #Attr | #Minority | #Majority | IL |
|---------|------------------------|-------|-----------|-----------|----|
| Glass   | 7                      | 10    | 29        | 185       | 6.38 |
| Abalone | 18                     | 8     | 42        | 689       | 16.4|
| Yeast   | 2                      | 8     | 51        | 1433      | 28.1|
| Haberman| 2                      | 3     | 81        | 225       | 2.78 |
| Pima    | 1                      | 8     | 268       | 300       | 1.87 |

**TABLE 2. Confusion matrix for a two-class problem.**

| Actual Negative | Predicted Negative | Predicted Positive |
|-----------------|--------------------|--------------------|
| \( Tn \)        | \( Fp \)           | \( Tp \)           |
| \( Fn \)        | \( Tp \)           | \( Fp \)           |
TABLE 3. Parameter settings for constrained oversampling.

| Parameters for Constrained Oversampling | Value |
|----------------------------------------|-------|
| \( m \): a parameter controls the size of the overlapping area | 10    |
| \( na \): the size of the ant population put on each starting point | 30    |
| \( NC\_\text{max} \): max number of iteration steps in ACO | 10    |
| \( \alpha \): a parameter controls the relative importance of trail versus heuristic information in ACO | 3    |
| \( \beta \): a parameter controls the relative importance of trail versus heuristic information in ACO | 0.5  |
| \( \varphi \): deposition coefficient in ACO | 3    |
| \( t \): number of starting points in majority region in each iteration | 3    |

TABLE 4. Performance comparison for the 7 sampling methods on datasets oversampled at 200%.

| Data Set | Sampling Method | Overall Accuracy | G-mean | Acc | F-measure |
|----------|-----------------|------------------|--------|-----|----------|
| Origin   | SMOTE           | 0.9439           | 0.8700 | 0.8000 | 0.7848   |
|          | BOS1            | 0.9553           | 0.9452 | 0.9250 | 0.9246   |
|          | BOS2            | 0.9470           | 0.9426 | 0.9375 | 0.9140   |
|          | ADASYN          | 0.9743           | 0.9749 | 0.9778 | 0.9631   |
|          | CBOS            | 0.9561           | 0.9536 | 0.9542 | 0.9352   |
|          | CO              | \textbf{0.9779}  | \textbf{0.9837} | 1    | \textbf{0.9671}   |
| Origin   | SMOTE           | 0.9329           | 0.4576 | 0.2800 | 0.3874   |
|          | BOS1            | 0.8896           | 0.7536 | 0.6122 | 0.6280   |
|          | BOS2            | 0.9097           | 0.7934 | 0.6615 | 0.7002   |
|          | ADASYN          | 0.8834           | 0.7739 | 0.6505 | 0.6460   |
|          | CBOS            | 0.8737           | 0.7494 | 0.6192 | 0.6118   |
|          | CO              | \textbf{0.9375}  | \textbf{0.8967} | \textbf{0.8494} | \textbf{0.8065}   |
| Origin   | SMOTE           | 0.9555           | 0.4338 | 0.2567 | 0.3418   |
|          | BOS1            | 0.9205           | 0.7298 | 0.5613 | 0.5723   |
|          | BOS2            | 0.9472           | 0.8435 | 0.7425 | 0.7312   |
|          | ADASYN          | 0.9268           | 0.7375 | 0.5706 | 0.6122   |
|          | CBOS            | 0.9218           | 0.7683 | 0.6232 | 0.6195   |
|          | CO              | \textbf{0.9591}  | \textbf{0.9388} | \textbf{0.9158} | \textbf{0.8156}   |
| Origin   | SMOTE           | 0.6566           | 0.4401 | 0.2597 | 0.2787   |
|          | BOS1            | 0.7129           | 0.7098 | 0.7027 | 0.7134   |
|          | BOS2            | 0.7559           | 0.7528 | 0.7377 | 0.7531   |
|          | ADASYN          | 0.7171           | 0.7121 | 0.7142 | 0.7218   |
|          | CBOS            | 0.7105           | 0.7093 | 0.6946 | 0.7166   |
|          | CO              | \textbf{0.7687}  | \textbf{0.7572} | \textbf{0.8510} | \textbf{0.7925}   |
| Origin   | SMOTE           | 0.7045           | 0.6570 | 0.5601 | 0.5657   |
|          | BOS1            | 0.7324           | 0.7170 | 0.7723 | 0.7800   |
|          | BOS2            | 0.7692           | 0.7451 | 0.8328 | 0.8154   |
|          | ADASYN          | 0.7794           | 0.7543 | 0.8367 | 0.8255   |
|          | CBOS            | 0.7539           | 0.7367 | 0.7958 | 0.8011   |
|          | CO              | \textbf{0.8963}  | \textbf{0.8617} | \textbf{0.9482} | \textbf{0.9257}   |

in traditional oversampling methods may give rise to lower Overall Accuracy, G-mean, and F-measure.

As to our method in which we try to eliminate the generation of noise in the oversampling process, the figures show ascending performance when the oversampling rate becomes larger. However, this approach expands the decision region of the minority class in a mild manner and this characteristic leads to a relatively low performance when the oversampling rate is not large enough. When more samples are generated, our method is shown to be effective.

2) THE INFLUENCE OF CLASS OVERLAPPING TO THE CLASSIFIER AND OVERSAMPLING METHODS

As mentioned in Section I, we rest our method on the opinion that overlapping between classes is to be blamed for the loss of performance of learning systems in imbalanced datasets.
To examine this stand, we exhibit the characteristics of an overlapping region extracted with the method proposed in Subsection A of Section II in Table 5.

First, we explored the influence of class overlapping on the performance of our classifier. We note that generally $\text{Acc}^+$ of CART on datasets in which most minority samples

---

FIGURE 5. Experimental results on Glass (a) Overall Accuracy (b) G-mean (c) $\text{Acc}^+$ (d) F-measure.

FIGURE 6. Experimental results on Abalone (a) Overall Accuracy (b) G-mean (c) $\text{Acc}^+$ (d) F-measure.

FIGURE 7. Experimental results on Yeast (a) Overall Accuracy (b) G-mean (c) $\text{Acc}^+$ (d) F-measure.
located inside the overlapping region is undesirable. This phenomenon can be easily accepted by observing Fig. 10 which shows the relationship between the overlapping ratio of minority sample and Acc\(^+\) in different datasets. It can be seen that on Haberman, which has as many as 64.20% minority samples placed inside the overlapping area, our decision tree suffers from great loss in precision — only an Acc\(^+\) of 25.97% is achieved. On the contrary, on Pima, a dataset which possesses a similar imbalance level with Haberman but differs in the ratio of minority samples in overlapping region (this ratio is only 38.81% in Pima), our classifier turns out to be far more effective in recognizing minority samples by resulting in an Acc\(^+\) of 56.01%. Similar situations happen to the changes of G-mean and F-measure on various datasets.

Based on the discussion above, we conclude that the performance of classifiers, at least CART, on an imbalanced dataset is heavily influenced by the overlapping ratio of minority samples. The kind of data distribution, which most minority samples in the dataset are positioned in the overlapping region, could well lead to performance degradation. This result verifies and extends the conclusion reached by Yu et al. [29] and Jo and Japkowicz [55].
In this paper, we have proposed a new oversampling technique, Constrained Oversampling, to address the class imbalance problem, especially in datasets with overlapping between classes.

This technique is executed in three successive steps: extract overlapping regions based on the KNN algorithm; define boundary samples for each minority instance using ACO; synthesize the required amount of minority samples under constraints. Two major results are expected after this technique is applied: learning on the border is strengthened so that minority samples in imbalanced datasets are more easily recognized and few noise samples are introduced to the original dataset when oversampling the minority class in overlapping regions so that the performance would not be harmed by the generation of noise.

According to our experimental results, CO generally produces satisfying accuracy, $G$-mean, $Acc^+$, and $F$-measure on various datasets differ with each other in imbalance level, number of features, and size. Further analysis of the results also reveals the impact of class overlapping on the performance of the classifier and oversampling strategies.

It should be noted that this study has examined only the influence of class overlapping in the imbalanced classification problem. To get a comprehensive understanding of the impact of other factors in imbalanced datasets, there are still a lot of works to be done. Meanwhile, there remains substantial room for future work considering the excessive computational and storage cost of the proposed method.

### IV. CONCLUSION

Further, we investigated the influence of class overlapping on different oversampling methods. The relationship between the overlapping ratio of minority sample and $F$-measure in different datasets when oversampling rate equals 200% is shown for different oversampling strategies in Fig. 11. In Haberman, Abalone, and Yeast, where the large overlapping ratio of minority samples in these datasets may give birth to a lot of synthetic noise samples. SMOTE, ADASYN, and CBOS, which do not take the noise generation around the borderline into consideration, bear a greater loss in $G$-mean, $Acc^+$, and $F$-measure compared to the other methods. Borderline-SMOTE shows better performance because they are devised to strengthen learning on the boundaries. But Borderline-SMOTE is still undesirable since its assumption is somehow oversimplified in handling overlapping regions. Among these techniques, our CO method renders to be the best solution in datasets with serious class overlapping.

Based on this observation, we believe that noise generation in oversampling imposes a negative effect on learning algorithms and the noise-prevention mechanism would improve the effect of oversampling methods.

### NOMENCLATURE

- $\alpha$: Random number between 0 and 1
- ACO: Ant colony optimization
- ADASYN: Adaptive Synthetic Sampling
- $\beta$: Relative importance of precision versus recall
- $boundary_{ij}$: Borderline of minority region defined by a set of majority samples from $nest_j$ to $Min_i$ in $O$
- BOS1: Borderline-SMOTE1
- BOS2: Borderline-SMOTE2
- CBOS: Cluster Based Synthetic Oversampling
- CART: Classification and Regression Tree
- CO: Constrained Oversampling
- Dir: Direction vector
- Dist: Distance between the new synthetic sample and $Min$
- $Dist(Min, Maj)$: Distance between a minority sample in the overlapping regions $Min$ and a majority sample belonging to the boundary set $Maj$
- $Dist(Min, 5^{th} NN of Min)$: Distance between $Min$ and its $5^{th}$ nearest neighbor
- $Dist(Maj, 5^{th} NN of Maj)$: Distance between $Maj$ and its $5^{th}$ nearest neighbor
- $Fn$: Numbers of false negatives
- $Fp$: Numbers of false positives
the number of synthetic samples needs to be generated for the $i$th minority sample

$G$ Number of minority samples to be generated

IL Imbalance level

KNN K-nearest-neighbors based algorithm

$\text{Maj}$ Majority sample

$\text{Maj}_i$ Corresponding majority sample on the boundary, for $\text{Min}_i$

$\text{Min}$ Minority sample

$\text{Min}_i$ The $i$th minority sample

$\text{Min}_{\text{new}}$ The synthetic instance, the new generated $\text{Min}$

$\text{na}$ Number of ants on each node

$\text{nest}_j$ The $j$th nest for ACO

$\text{NC}_{\text{max}}$ Max iteration number

$\%\$ Amount of SMOTE

$O$ Overlapping set

$\text{path}_{i,j}$ The optimal path from $\text{nest}_j$ to $\text{Min}_i$, in $O$

$\text{ROS}$ Random Over Sampling

$\text{RUS}$ Random Under Sampling

$S$ Original training set $S$

$S_{\text{balanced}}$ New balanced training set

$S_{\text{maj}}$ Majority set in the overlapping region

$S_{\text{min}}$ Minority set in the overlapping region

$\text{SMOTE}$ Synthetic Minority Oversampling Technique

$S_{\text{new}}$ New synthetic samples set

$S_{\text{selected}}$ The minority sample selected in iterations of generating synthetic samples

$T_p$ the numbers of true positives

$T_n$ the numbers of true negatives

REFERENCES

[1] H. He and E. A. Garcia, “Learning from imbalanced data,” *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.

[2] P. Branco, L. Torgo, and R. P. Ribeiro, “A survey of predictive modeling on imbalanced domains,” *ACM Comput. Surv.*, vol. 49, no. 2, pp. 31:1–31:50, 2016.

[3] B. Kacwczryk, “Cost-sensitive one-vs-one ensemble for multi-class imbalanced data,” in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 2447–2452.

[4] C. Zhang, K. C. Tan, and R. Ren, “Training cost-sensitive deep belief networks on imbalanced data problems,” in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 4362–4367.

[5] M. Kubat, R. C. Holte, and S. Matwin, “Machine learning for the detection of oil spills in satellite radar images,” *Mach. Learn.*, vol. 30, nos. 2–3, pp. 195–215, 1998.

[6] C. X. Ling, V. S. Sheng, and Q. Yang, “Test strategies for cost-sensitive decision trees,” *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 8, pp. 1055–1067, Aug. 2006.

[7] P. K. Chan and S. J. Stolfi, “Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection,” in *Proc. 4th Int. Conf. Knowl. Discovery Data Mining*, 1998, pp. 164–168.

[8] P. M. Rasmussen, L. K. Hansen, K. H. Madsen, N. W. Churchill, and S. C. Strother, “Model sparsity and brain pattern interpretation of classification models in neuroimaging,” *Pattern Recognit.*, vol. 45, no. 6, pp. 2085–2100, Jun. 2012.

[9] Y. Sun, C. G. Castellano, M. Robinson, R. Adams, A. G. Rust, and N. Daveya, “Using pre & post-processing methods to improve binding site predictions,” *Pattern Recognit.*, vol. 42, no. 9, pp. 1949–1958, 2009.

[10] M. Ohsaki, P. Wang, K. Matsuda, S. Katagiri, H. Watanabe, and A. Ralescu, “Confusion-matrix-based kernel logistic regression for imbalanced data classification,” *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 9, pp. 1806–1819, Sep. 2017.

[11] Q. Yang and X. Wu, “10 challenging problems in data mining research,” *Int. J. Inf. Technol. Decis. Making*, vol. 5, no. 4, pp. 597–604, Dec. 2006.

[12] M. Galar, A. Fernández, E. Barrenechea, and F. Herrera, “EUSBoost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling,” *Pattern Recognit.*, vol. 46, no. 12, pp. 3460–3471, Dec. 2013.

[13] M. A. Tahir, J. Kittler, and F. Yan, “Inverse random undersampling for class imbalance problem and its application to multi-label classification,” *Pattern Recognit.*, vol. 45, no. 10, pp. 3738–3750, Oct. 2012.

[14] L. Abdi and S. Hashemi, “To combat multi-class imbalanced problems by means of over-sampling techniques,” *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 1, pp. 238–251, Jan. 2016.

[15] B. Das, N. C. Krishnan, and D. J. Cook, “RACOG and wRACOG: Two probabilistic oversampling techniques,” *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 1, pp. 222–234, Jan. 2015.

[16] H.-N. Qu, G.-Z. Li, and W.-S. Xu, “An asymmetric classifier based on partial least squares,” *Pattern Recognit.*, vol. 43, no. 10, pp. 3448–3457, Oct. 2010.

[17] K. M. Ting, “An instance-weighting method to induce cost-sensitive trees,” *IEEE Trans. Knowl. Data Eng.*, vol. 14, no. 3, pp. 659–665, May 2002.

[18] Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser, “SVMs modeling for highly imbalanced classification,” *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 39, no. 1, pp. 281–288, Feb. 2009.

[19] Y. Li, L. Zhang, Y. Xu, Y. Yao, R. Y. K. Lau, and Y. Wu, “Enhancing binary classification by modeling uncertain boundary in three-way decisions,” *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 7, pp. 1438–1451, Jul. 2017.

[20] N. V. Chawla, N. Japkowicz, and A. Kotcz, “Editorial: Special issue on learning from imbalanced data sets,” *ACM SIGKDD Explor. Newslett.*, vol. 6, no. 1, pp. 1–6, Jun. 2004.

[21] J. Wu, H. Xiong, and J. Chen, “COG: Local decomposition for rare class analysis,” *Data Mining Knowl. Discovery*, vol. 20, no. 2, pp. 814–823, 2010.

[22] D. A. Cieslak, T. R. Hoens, N. V. Chawla, and W. P. Kegelmeyer, “Helleringer distance decision trees are robust and skew-insensitive,” *Data Mining Knowl. Discovery*, vol. 24, no. 1, pp. 136–158, Jan. 2012.

[23] D. Zhang, W. Liu, X. Gong, and H. Jin, “A novel improved SMOTE resampling algorithm based on fractal,” *J. Comput. Inf. Syst.*, vol. 7, no. 6, pp. 2204–2211, 2011.

[24] M. A. Maloof, “Learning when data sets are imbalanced and when costs are unequal and unknown,” in *Proc. Workshop Learn. Imbalanced Data Sets II, ICM*, vol. 2, 2003, pp. 1–2.

[25] G. Ou and Y. L. Murphy, “Multi-class pattern classification using neural networks,” *Pattern Recognit.*, vol. 40, no. 1, pp. 4–18, Jan. 2007.

[26] A. Fernández, M. J. Del Jesus, and F. Herrera, “Multi-class imbalanced data-sets with linguistic fuzzy rule based classification systems based on pairwise learning,” in *Computational Intelligence for Knowledge-Based Systems Design*, Berlin, Germany: Springer, 2010, pp. 89–98.

[27] M. Kubat and S. Matwin, “Addressing the curse of imbalanced data-sets: One-sided selection,” in *Proc. 4th Int. Conf. Mach. Learn.*, vol. 97, Jul. 1997, pp. 179–186.

[28] S.-J. Yen and Y.-S. Lee, “Cluster-based under-sampling approaches for imbalanced data distributions,” *Expert Syst. Appl.*, vol. 36, no. 3, pp. 5718–5727, Apr. 2009.

[29] H. Yu, J. Ni, and J. Zhao, “ACOSampling: An ant colony optimization-based undersampling method for classifying imbalanced DNA microarray data,” *Neurocomputing*, vol. 101, pp. 309–318, Feb. 2013.

[30] N. Japkowicz and S. Stephen, “The class imbalance problem: A systematic study,” *Intell. Data Anal.*, vol. 6, no. 5, pp. 429–449, Oct. 2002.

[31] G. A. P. A. Batista, R. C. Prati, and M. C. Monard, “A study of the behavior of several methods for balancing machine learning training data,” *ACM SIGKDD Explor. Newslett.*, vol. 6, no. 1, pp. 20–29, Jun. 2004.

[32] G. M. Weiss and F. Provost, “Learning when training data are costly: The effect of class distribution on tree induction,” *J. Artif. Intell. Res.*, vol. 19, pp. 315–354, Oct. 2003.
[33] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” J. Artif. Intell. Res., vol. 16, pp. 321–357, Jun. 2002.

[34] H. Han, W.-Y. Wang, and B.-H. Mao, “Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning,” in Advances in Intelligent Computing, vol. 4914, Berlin, Germany: Springer, 2008, pp. 878–887.

[35] H. Ye, Y. Bai, E. A. García, and S. Li, “ADASYN: Adaptive synthetic sampling approach for imbalanced learning,” in Proc. IEEE Int. Joint Conf. Neural Netw. (IEEE World Congr. Comput. Intell.), Jun. 2008, pp. 1322–1328.

[36] S. Barua, M. M. Islam, and K. Murase, “A novel synthetic minority over-sampling technique for imbalanced data set learning,” in Proc. Int. Conf. Neural Inf. Process. Berlin, Germany: Springer, 2011, pp. 735–744.

[37] C. Bunkhumpornpat, K. Sinapromsaran, and C. Lursinsap, “Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem,” in Advances in Knowledge Discovery and Data Mining, Berlin, Germany: Springer, 2009, pp. 475–482.

[38] R. C. Prati, G. E. A. P. A. Batista, and M. C. Monard, “Class imbalances versus class overlapping: An analysis of a learning system behavior,” in MICAI 2004: Advances in Artificial Intelligence, Berlin, Germany: Springer, 2004, pp. 312–321.

[39] R. Isermann, Fault-Diagnosis Systems: An Introduction From Fault Detection to Fault Tolerance, Berlin, Germany: Springer, 2006.

[40] Y. Yuan, X. Lian, L. Chen, Y. Sun, and G. Wang, “RKNN: KNN search on road networks by incorporating social influence,” IEEE Trans. Knowl. Data Eng., vol. 28, no. 6, pp. 1575–1588, Jun. 2016.

[41] G. Song, J. Rochas, L. E. Beze, F. Hue, and F. Magoules, “K nearest neighbour joins for big data on MapReduce: A theoretical and experimental analysis,” IEEE Trans. Knowl. Data Eng., vol. 28, no. 9, pp. 2376–2392, Sep. 2016.

[42] A. Almalawi, A. Fahad, Z. Tari, M. A. Cheema, and I. Khalil, “KNNWVC: An efficient k-nearest neighbours approach based on various-widths clustering,” IEEE Trans. Knowl. Data Eng., vol. 28, no. 1, pp. 68–81, Jan. 2016.

[43] I. Mani and I. Zhang, “KNN approach to unbalanced data distributions: A case study involving information extraction,” in Proc. Workshop Learn. From Imbalanced Datasets II, ICML, Washington, DC, USA, vol. 126, Aug. 2003, pp. 1–7.

[44] A. Colorni, M. Dorigo, and V. Maniezzo, “Distributed optimization by ant colonies,” in Proc. 1st Eur. Conf. Artif. Life, vol. 142, 1991, pp. 134–142.

[45] A. Ulug and D. Aydin, “An interactive simulation and analysis software for solving TSP using ant colony optimization algorithms,” Adv. Eng. Softw., vol. 40, no. 5, pp. 341–349, May 2009.

[46] A. Shmygelska and H. H. Hoos, “An ant colony optimisation algorithm for the 2D and 3D hydrophobic polar protein folding problem,” BMC Bioinf., vol. 6, no. 1, p. 30, Dec. 2005.

[47] K. Cao, X. Yang, X. Chen, Y. Zang, J. Liang, and J. Tian, “A novel ant colony optimization algorithm for large-distorted fingerprint matching,” Pattern Recognit., vol. 45, no. 1, pp. 151–161, Jan. 2012.

[48] S.-H. Tsai and Y.-W. Chen, “A novel fuzzy method optimization based on ant colony optimization algorithm,” IEEE Access, vol. 4, pp. 3747–3756, 2016.

[49] Y.-J. Gong, E. Chen, X. Zhang, L. M. Ni, and J. Zhang, “AntMapper: An ant colony-based map matching approach for trajectory-based applications,” IEEE Trans. Intell. Transp. Syst., vol. 19, no. 2, pp. 390–401, Feb. 2018.

[50] Y. Sun, W. Dong, and Y. Chen, “An improved routing algorithm based on ant colony optimization in wireless sensor networks,” IEEE Commun. Lett., vol. 21, no. 6, pp. 1317–1320, Jun. 2017.

[51] M. Dorigo, V. Maniezzo, and A. Colorni, “Ant system: Optimization by a colony of cooperating agents,” IEEE Trans. Syst. Man, Cybern. B, Cybern., vol. 26, no. 1, pp. 29–41, Feb. 1996.

[52] C. Blake and C. J. Merz. (1998). UCI Repository of Machine Learning Databases. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[53] D. Steinberg and P. Colla, “CART: Classification and regression trees,” The Top Ten Algorithms in Data Mining, vol. 9, Berlin, Germany: Springer, 2009, p. 179.

[54] T. Raeder, T. R. Hoens, and N. V. Chawla, “Consequences of variability in classifier performance estimates,” in Proc. IEEE Int. Conf. Data Mining, Dec. 2010, pp. 421–430.

[55] T. Jo and N. Japkowicz, “Class imbalances versus small disjuncts,” ACM SIGKDD Explor. Newslett., vol. 6, no. 1, pp. 40–49, Jun. 2004.

CHANGHUI LIU received the B.Sc. degree in mechanical engineering from Hunan University, Changsha, China, in 2010, and the Ph.D. degree from Shanghai Jiao Tong University, Shanghai, China, in 2016. From 2016 to 2018, he held a postdoctoral position with the Shanghai Key Laboratory of Digital Manufacture for Thin-Walled Structures, School of Mechanical Engineering, Institute of Automobile Engineering, Shanghai Jiao Tong University. From 2018 to 2019, he held a postdoctoral position and a Research Associate with the H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, USA. He is currently an Assistant Professor with the School of Mechanical Engineering, Tongji University. His primary research interests include data mining, machine learning, and quality control.

SUN JIN received the B.Sc. and M.Sc. degrees in mechanical engineering from Hunan University, Changsha, China, in 1995 and 1998, respectively, and the Ph.D. degree in mechanical engineering from Shanghai Jiao Tong University, Shanghai, China, in 2001. He is currently a Professor with the Department of Mechanical Engineering, Shanghai Jiao Tong University. His current research interests include data analysis, fault diagnosis, manufacturing quality control, assembly technology development, and industrial application.

DONGHONG WANG received the Ph.D. degree in material processing engineering from Shanghai Jiao Tong University, Shanghai, China, in 2014. He is currently an Assistant Fellow with the Department of Material Science and Engineering, Shanghai Jiao Tong University. His main current research interests include theories of metal solidification in investment casting, data mining and artificial intelligence in material processing, and whole process, cross-level, and multiscale calculation of superalloy castings in the engineering of integrated computing materials for superalloy complex castings.

ZICHAO LUO received the B.Sc. degree in mechanical engineering from the East China University of Science and Technology, Shanghai, China, in 2011, and the M.Sc. degree from Shanghai Jiao Tong University, Shanghai, in 2013. He is currently pursuing the Ph.D. degree with the Yoshino and Yamamoto Laboratory, Tokyo Institute of Technology, Tokyo, Japan. His primary research interests include data mining and machine learning.
JIANBO YU received the B.Eng. degree from the Department of Industrial Engineering, Zhejiang University of Technology, Zhejiang, China, in 2002, the M.Eng. degree from the Department of Mechanical Automation Engineering, Shanghai University, Shanghai, China, in 2005, and the Ph.D. degree from the Department of Industrial Engineering and Management, Shanghai Jiao Tong University, Shanghai, in 2009. In 2008, he joined the Center for Intelligent Maintenance System, University of Cincinnati, Cincinnati, OH, USA, as a Visiting Scholar. From 2009 to 2013, he was an Associate Professor with the Department of Mechanical Automation Engineering, Shanghai University. Since 2016, he has been a Professor with the School of Mechanical Engineering, Tongji University. His current research interests include intelligent condition-based maintenance, machine learning, quality control, and statistical analysis. He serves as the Editorial Board Member for Advances in Mechanical Engineering, Chinese Journal of Engineering, and Journal of Advanced Manufacturing Research.

BINGHAI ZHOU, photograph and biography not available at the time of publication.

CHANGLIN YANG, photograph and biography not available at the time of publication.