Queries to the Author

Author: Please check and confirm whether the name of corresponding author in the first footnote is correct as set.

When you submit your corrections, please either annotate the IEEE Proof PDF or send a list of corrections. Do not send new source files as we do not reconvert them at this production stage.

Authors: Carefully check the page proofs (and coordinate with all authors); additional changes or updates WILL NOT be accepted after the article is published online/print in its final form. Please check author names and affiliations, funding, as well as the overall article for any errors prior to sending in your author proof corrections. Your article has been peer reviewed, accepted as final, and sent in to IEEE. No text changes have been made to the main part of the article as dictated by the editorial level of service for your publication.

Per IEEE policy, one complimentary proof will be sent to only the Corresponding Author. The Corresponding Author is responsible for uploading one set of corrected proofs to the IEEE Author Gateway

Q1. Please confirm or add details for any funding or financial support for the research of this article.
Q2. Please provide complete bibliographic details for Ref. [33].

Oversampling With Reliably Expanding Minority Class Regions for Imbalanced Data Learning

Tuanfei Zhu, Xinwang Liu, *Senior Member, IEEE*, and En Zhu

Abstract—This paper proposes a simple interpolation Oversampling method with the purpose of Reliably Expanding the Minority class regions (OREM). OREM first finds the candidate minority region around each original minority sample, then exploits this region to further identify those clean subregions without distributing any majority sample. The synthetic samples are only allowed to generate in the clean subregions, so that the regions of the minority class can be broadened reliably. Given that the learning from multiclass imbalanced data is more challenging as compared to two-class scenarios, we also extend OREM to handle multiclass imbalance problems by leveraging an iteration procedure of generating synthetic samples, consequently leading to a multiclass oversampling algorithm OREM-M. The key peculiarity of OREM-M is to reduce the class overlapping not only between the synthetic minority and original samples, but also from the synthetic samples of different minority classes. In this way, OREM-M ensures that the data of each class after oversampling can be modeled well. In addition, we embed OREM into boosting framework to develop a new ensemble method OREMBoost addressing class imbalance problems. Extensive experiments demonstrate the effectiveness of the proposed OREM, OREM-M, and OREMBoost.

Index Terms—Class mbalance problems, oversampling, multiclass imbalance, ensemble learning

1 Introduction

The classification learning from imbalanced data is a challenge yet prevalent issue in machine learning and data mining field, in which the majority class overwhelms the minority class in the sample size. The main affliction of this problem is that the learned models typically show undesirable recognition performance on the minority class, because of the accuracy-oriented design of standard classification learning methods, and the existence of the data difficulty factors (e.g., within-class imbalance, class overlapping, scarce representative samples) [1]. The correct classification of the minority samples, however, is vital in many important real-world applications such as disease diagnosis, network intrusion detection, and fraud detection.

In last two decades, a large number of imbalanced learning methods have been developed [1], [2], [3]. They can be divided into data-level methods, algorithm-level approaches, and hybrid methods. Data-level methods aim to mitigate imbalanced class distribution by adding new minority samples (i.e., oversampling) [4], [5], [6], removing redundant majority samples (i.e., undersampling) [7], [8], or using a combination of both ways. Algorithm-level approaches are devoted into modifying standard classification methods to emphasize the learning of the minority samples, via improving training mechanism (e.g., customization of loss function) or predicted rule (e.g., decision threshold movement of output). In hybrid methods, the developed algorithms alter both the imbalanced class distribution, and the learning mechanism to accommodate the classification of imbalanced data [9], [10].

Unlike algorithm-level and hybrid methods, data-level algorithms have the following peculiarities: 1) Usability; they are generally easy to use and implement due to only operating on the data itself. 2) Effectiveness; the quality improvement of imbalanced data can benefit the training of any classification model, and extensive empirical studies have shown both oversampling and undersampling are all universally valid [8], [11]. 3) Versatility; they are independent of specific classification algorithm, and can also combine with algorithm-level methods to produce elaborate and competitive hybrid approaches.

In this paper, we focus our attention on oversampling techniques. Compared to undersampling, oversampling techniques are more fundamental solutions. Since the source of difficulty in classifying unbalanced data is essentially the lack of the minority class information, oversampling techniques can directly strengthen the concept expression of the minority class by introducing new minority samples. Moreover, oversampling techniques do not suffer from the risk of losing informative majority samples

- *Tuanfei Zhu is with the School of Computer, National University of Defense Technology, Changsha 410073, China, and also with the College of Computer Engineering and Applied Mathematics, Changsha University, Changsha 410073, China. E-mail: zhutuanfei@hnu.edu.cn.*
- *Xinwang Liu and En Zhu are with the School of Computer, National University of Defense Technology, Changsha 410073, China. E-mail: {xinwangliu, enzhu}@nudt.edu.cn.*

Manuscript received 28 Aug. 2021; revised 22 Apr. 2022; accepted 27 Apr. 2022. Date of publication 0 . 0000; date of current version 0 . 0000.
This work was supported in part by the National Natural Science Foundation of China under Grants 61922088 and 62006030, in part by the Natural Science Foundation of Hunan Province, China under Grant 2020JJ5623, and in part by the Scientific Research Foundation of Hunan Provincial Education Department under Grant 20B060.
(Corresponding author: Xinwang Liu.)
Recommended for acceptance by B.C.M. Fung.
Digital Object Identifier no. 10.1109/TKDE.2022.3171706

1.1 Motivations

The motivations of our work contain the following three aspects.

i) *Designing More Effective Interpolation Oversampling Algorithm.* A diverse collection of oversampling approaches have been proposed in the previous works. Among of these methods, the interpolation oversampling algorithms are most simple, robust, and popular [11]. A main reason is that they only need local data information to generate synthetic samples. On the contrary, the math-intensive oversampling algorithms including probability distribution-based methods, structure-preserving approaches, etc., often fail in some edge situations such as the extreme scarcity of the minority samples,[1] and the high ratio of feature dimension to minority sample size.[2]

The interpolation oversampling enhances the concept expression of the minority class by filling synthetic samples into the minority class regions. Due to the rarity of the minority samples, the minority class regions include the observable regions based on the training set and the potential minority regions needed to be inferred. The key issue is how to discover the potential regions of the minority class. In the existing literature, the k-nearest neighbor (k-NN)-based [4], [5], [12] and clustering-based [13], [14] ways are two most common strategies. However, both strategies have their own drawbacks. The k-NN-based way assumes that the areas between any two minority neighbor samples belong to the minority class. This assumption is too strong and would be violated easily, because outliers and rare cases are all far away from most of the other minority samples [13], [15]. These samples and their corresponding minority neighbors might cross the majority areas. Some works attempt to strip out outliers and rare cases in advance by considering k-nearest neighborhood of each minority sample [13], [16]. But it is almost impossible to accurately reflect the position characteristics of all the minority samples through a uniform k-nearest neighborhood [5], [13], [17]. The clustering-based way uses clustering algorithms to identify the data space of the minority class. Its main drawback is that the clustering parameters themselves are difficult to be set appropriately, and the reliability of clustering depends heavily on whether the minority data presents a clear clustering structure.

Therefore, it is necessary to develop a more effective interpolation oversampling algorithm which can reliably fill the synthetic samples into the minority class regions, so that the concept of minority class can be expressed adequately.

ii) *Improving the Ability of Oversampling Technique in Handling Multiclass Imbalanced Data.* Most oversampling algorithms are designed specifically for two-class imbalance problems. They cannot be applied to multiclass imbalanced data directly. However, binary classification is not only scenario in real-world applications, and multiclass data is more likely to appear imbalanced class distribution. A straightforward solution is to convert a multiclass imbalanced problem into several binary subproblems via class decomposition schemes, then the two-class imbalance methods are applied to each of the binary subproblems. However, the subproblems produced by any class decomposition scheme have only partial data knowledge. The solutions combined with class decomposition, usually perform worse than native multiclass solutions, because their learning is not based on the complete data information of all classes [6], [18].

A few works have been devoted into developing dedicated multiclass oversampling techniques [5], [6], [19]. The proposed methods are generally conservative, i.e., they emphasize on avoiding the class overlapping between the synthetic minority and original samples. However, they do not consider the problem of the class overlapping occurred among the synthetic samples of different minority classes. This problem can be serious when a lot of synthetic samples are needed to generate. Hence, it is necessary to design an ad hoc oversampling method which can handle the class overlapping from various types of samples, so that the quality of multiclass imbalanced data could be improved significantly.

iii) *Establishing the Positive Synergy With Ensemble Learning Methods.* Almost all of the existing studies focused the attention on the preprocessed ability of oversampling algorithms, but integrating with other types of approaches to develop hybrid solutions is another important use of oversampling techniques. As a typical representative, oversampling techniques have been widely combined with standard ensemble methods such as bagging and boosting, leading to ensemble-based solutions addressing class imbalance problems [20]. The empirical results have confirmed that ensemble-based solutions generally outperform data preprocessing methods [9], [20]. Therefore, it is meaningful to embed the designed oversampling technique into ensemble learning, so that the positive synergy can be formed to further improve the classification performance of imbalanced data.

1.2 Our Methods

Based on the *motivation i)*, we propose a new interpolation oversampling method OREM. OREM finds the candidate minority regions around the original minority samples, then exploits these candidate regions to identify the clean subregions without containing any majority sample. The clean subregions are considered as the potential minority regions. By filling synthetic samples into the clean subregions, OREM can broaden the minority class regions and enhance the concept expression of minority class. The proposed OREM is simple. Neither the use of clustering algorithms nor the adjustment of the neighbor parameter k is involved.

Based on the *motivation ii)*, we generalize OREM to handle multiclass imbalance problems by utilizing an iteration procedure of synthetic sample generation, leading to a multiclass oversampling algorithm OREM-M. In OREM-M, synthetic samples are created one by one, only the synthetic sample whose nearest neighbor is not an original or synthetic sample from the other minority classes, is accepted. In this way, OREM-M mitigates the problem of class overlapping during oversampling multiple minority classes.

Based on the *motivation iii)*, we embed OREM into boosting framework to obtain a new ensemble solution OREMBoost.

1. Linear algebraic operations might become ill-posed problems in structure-preserving oversampling

2. Estimating accurately the distribution of the minority class is difficult in probability distribution-based oversampling

OREMBoost balances the training data of each round before training a base classifier, so that the constructed base classifiers are less biased towards the majority classes and more diversity.

1.3 Experimental Results

First, we tested the performance of OREM on 28 two-class real-world datasets. The results show that OREM can outperform some state-of-the-art oversampling techniques with support vector machine (SVM), neural network (NN), C4.5 classifiers in terms of F1, G-mean, AUC.

Second, the performance of OREM-M was validated on 21 multiclass real-world datasets. The experimental results demonstrate that OREM-M is often significantly better than existing multiclass oversampling techniques with SVM, NN, C4.5 classifiers in terms of marco-F1, MG, MAUC.

Finally, we evaluated the usefulness of OREMBoost on both two-class and multiclass datasets. The obtained results confirm the effectiveness of OREMBoost through comparing with prevailing ensemble-based solutions.

2 Related Work

Although a lot of works have been done to combat class imbalance problems, we only provide here a brief overview for the category of oversampling algorithms, as oversampling techniques are our interest.

According to the difference of motivation, existing oversampling algorithms can be roughly classified into the interpolated oversampling, kernel-based oversampling, structure-preserving oversampling, distribution-based oversampling, majority generated oversampling, and multiclass-oriented oversampling.

The interpolated oversampling yields the synthetic samples through linearly interpolating between two original minority samples [4], [5], [13], [14], [17]. The designed motivation behind is to broaden the minority class region for alleviating the overfitting problem [4]. However, when the imbalanced data in feature space is non-linearly separable, the synthetic samples created by linear interpolation approaches might fall in the majority class region. To deal with this problem, kernel-based oversampling methods utilize kernel techniques to map the original imbalanced data into a high-dimensional linear separable space, then generate synthetic samples in the kernel induced feature space [21], [22]. Since synthetic samples are manufactured in some way of stochastic disturbance, they might destroy the useful structures implied in the original data. Structure-preserving oversampling methods aim to produce informative synthetic samples on the premise of protecting a certain structure characteristic [19], [23], [24], [25]. The motivation of distribution-based methods is that the minority class distribution modeled from the global information of minority samples can be used to yield the synthetic samples which express the concept of the minority class adequately [26], [27]. This kind of oversampling algorithms first attempt to acquire the underlying distribution of the minority class, then yield synthetic samples relying on this distribution.

All the methods mentioned above are based on the original minority samples to create synthetic samples. If the minority samples are extremely scarce, the minority class would not have enough information to support the generation of synthetic samples. The majority generated oversampling methods directly utilize an abundance of the majority samples to yield synthetic minority samples, so that the difficulty of generating synthetic samples with the minority samples can be bypassed [28], [29].

Compared to two-class imbalanced scenarios, multiclass imbalance problems are more challenging [5]. Multiclass-oriented oversampling techniques are devoted to combatting multiclass imbalance classification [5], [6], [19]. The difficulty factor of class overlapping is in great need of being specially treated in multiclass-oriented oversampling, as multiclass overlapping can become extremely severe when multiple minority classes are required to oversample.

Table 1 summarizes the characteristics of these types of oversampling algorithms. A complete overview is provided in Section S1 of the supplementary materials associated with this paper, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TKDE.2022.3171706.

3 Method

In this section, we first detail the oversampling method with reliably expanding minority class regions, OREM. Then, we descript the multiclass version of OREM, OREM-M, and the ensemble method OREMBoost. Finally, the computational complexity is analyzed for the proposed methods.

3.1 OREM: Oversampling With Reliably Expanding Minority Class Regions

3.1.1 Description of OREM Algoirthm

As the description of Section 1.1, the key of interpolated oversampling methods is to find the potential minority class regions. Our OREM contains two steps for discovering the potential minority regions, i.e., exploring the Candidate Minority class Regions (CMRs) and further identifying the clean subregions within CMRs.

In stage of exploring CMRs, OREM expands outwards the CMR around each original minority sample, by examining the distribution of neighboring samples. For the convenience of explanation, Fig. 1a illustrates an example of finding the CMR near x_1. The circular region $R(x_1x_7)$, centered at x_1 with radius the distance between x_1 and x_7, can be regarded as a possible minority class region, because the minority samples have a high likelihood to appear in this region. However, an abundance of neighbor samples subsequent to x_7 are from the majority class, which indicates the minority samples rarely occur in the adjacent area outside $R(x_1x_7)$. Hence, $R(x_1x_7)$ can be considered a maximal consecutive CMR around x_1.

In stage of identifying the clean subregions, the CMR of each minority sample is further exploited to find those clean regions without distributing any majority sample. The reason behind is that there are abundant majority samples in imbalanced training data, if the majority samples have not appeared in some subregions of CMR, these subregions would also have a low probability to occur the majority samples in testing data. Therefore, it is relatively safe that the clean subregions within CMR are classified into the

TABLE 1
Summary of Characteristics of Existing Oversampling Methods

Types of oversampling	Motivation	Main challenge	Fine-grained categorization	Weaknesses or limitations	Representative references
Interpolated oversampling	Broadening the regions of minority class to alleviate overfitting	The identification of the potential minority class regions	k-nearest neighbor -based interpolation way	Being easy to incur the over generalization and over constraint	[4], [17] [5]
			Clustering-based interpolation way	Requiring a clean clustering structure	[13], [14]
Kernel-based oversampling	Dealing with the non-linear separable problem in the imbalanced data	The generation of synthetic samples in kernel space		Having the information loss in the process of feature space projection and reconstruction	[21], [22]
Structure preserving oversampling	Preserving useful structure information	The acquisition of structure information	Retaining covariance structure	Unfit for handling the high -dimensional extreme imbalance data with multimodality	[19], [23]
			Maintaining manifold structure	Unsuitable to handling the data without a low-dimensional manifold space	[24], [25]
Distribution -based oversampling	Exploiting the global information to model the minority class distribution	The acquisition of underlying distribution		Unsuitable for dealing with the extreme imbalanced and high-dimensional data	[26], [27]
Majority-based oversampling	Bypassing the difficulty of generating informative synthetic samples with minority samples	The identification for the space of non-majority class		Without exploiting the local or global information in the minority class data	[28], [29]
Multiclass -oriented oversampling	Handling more challenging multiclass imbalance problems	The alleviation of class overlapping		Without considering the class overlapping among the synthetic samples of different minority classes	[6], [19] [5]

potential minority regions. Fig. 1b illustrates an example identifying the clean subregions near x_1. In Fig. 1b, $x_2, x_3, \ldots, x_7$ are the samples distributed in the CMR of x_1. We call them the candidate assistant seeds[3] of x_1, denoted by $\mathcal{C}(x_1)$. For the subregion formed between x_1 and each of $\mathcal{C}(x_1)$, OREM inspects whether it contains the majority samples. If the subregion does not exist any majority sample, it is a clean area, e.g., those circular shaded areas (including $S(x_1x_2)$, $S(x_1x_3)$, $S(x_1x_6)$, etc) are the clean subregions within the CMR of x_1, while $S(x_1x_7)$ is not.

The concrete implementation of OREM is descripted in Algorithm 1. First, OREM for each minority sample finds its CMR, which is implemented by function DiscovCMR(). In DiscovCMR(), $A(x_i) = \{x_{i1}, \ldots, x_{i(|S|-1)}\}$ is the rearranged sample set for $S \setminus \{x_i\}$ according to the distances of the samples to x_i, which x_{i1} is the nearest sample from x_1, x_{i2} is the second nearest, and so on. If successional q samples, $x_{i(k-q+1)}, x_{i(k-q+2)}, \ldots, x_{ik}$, in $A(x_i)$ are from the majority class, the hypersphere region $S(x_i x_{i(k-q)})$, centered at x_i with radius the distance between x_i and $x_{i(k-q)}$, is the discovered CMR of x_i (lines 7-18 of Algorithm 1). $\mathcal{C}(x_i) = \{x_{i1}, x_{i2}, \ldots, x_{i(k-q)}\}$ is just the sample set distributed in the CMR of x_i, i.e., the candidate assistant seeds of x_i. We can see that the CMR of a minority sample is either an area of class overlapping or a pure area of the minority class.

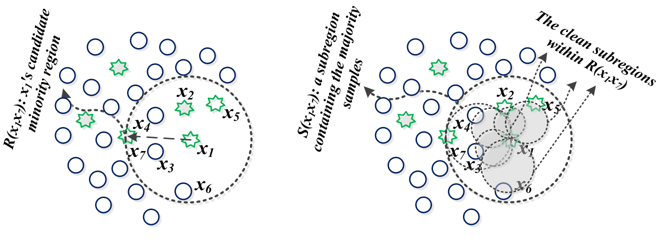
Then, OREM further exploits the CMR of each original minority sample to identify the clean subregions, which is implemented by function IdeCleanReg(). For each sample x_{ip} in $\mathcal{C}(x_i)$, if the hypersphere area, centered at the midpoint of x_i and x_{ip} with radius half the distance between x_i and x_{ip}, does not contain any majority sample, this area is a clean subregion (lines 24-32). It is worth pointing out that the samples of $\{x_{i(p+1)}, x_{i(p+2)}, \ldots, x_{i|\mathcal{C}(x_i)|}\}$ do not need to be considered in this process, because they have greater distances from x_i as compared to x_{ip} (lines 27-32). The sample x_{ip} corresponding to a clean subregion should be added into the assistant seed set of x_i (lines 33-35), so that the synthetic samples can be filled in the potential minority subregion between x_i and x_{ip}.

Finally, OREM generates an equal number of synthetic samples for each original minority sample, which is implemented by function Generate().

3.1.2 In-Depth Dissection of the Proposed OREM

In this subsection, We further analyse the oversampling mechanism of OREM, and discuss the advantages of OREM compared to existing interpolation oversampling methods.

The minority samples can be divided into three types: the inland samples resided inside the clusters of the minority class, the borderline samples settled in the class boundaries, and the outliers away from most of the other minority samples [17], [30]. As an example illustrating the characteristics of these three types of samples, x_i, x_b, and x_o in Fig. 2a are an inland sample, a borderline sample, and an outlier, respectively. According to their position peculiarities, the following observations can be obtained: 1) For an inland minority sample, almost all of the vicinal minority samples can be served as its assistant seeds. 2) It should be cautious



(a) (b)

Fig. 1. (a) Figure illustrating the candidate minority class region of a minority sample x_1. (b) Figure illustrating the subregions within x_1's candidate minority region, where the clean subregions are shaded with gray background.

3. The assistant seeds of a minority sample are those samples used to create synthetic samples with the considered minority sample.

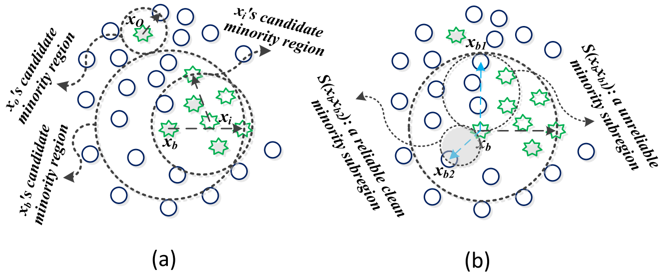


Fig. 2. (a) Figure illustrating an inland minority sample x_i, a borderline minority sample x_b, and an outlier x_o. (b) Figure illustrating an unreliable subregion and a clean subregion within x_1's candidate minority region.

to select the assistant seeds for a borderline minority sample, as there is a high risk of filling synthetic samples into the majority regions if the assistant seeds are selected inappropriately. In addition, it is useful that some borderline majority samples are incorporated into the assistant seeds, because the synthetic samples, generated between the borderline majority and minority samples, can push the distributed front of the minority samples towards the majority class regions, which breaks the limit that the synthetic samples are only created in the convex hull of original minority samples. 3) Since it has a high probability of generating noisy synthetic samples when oversampling an outlier, a prudent way is to restrain the generated synthetic samples from spreading the vicinal region.

Algorithm 1. OREM$(S_{min}, S_{maj}, q, \zeta)$

Input: S_{min}: set of minority samples; S_{maj}: set of majority samples; q: a counting parameter used to discover candidate minority class regions; ζ: number of synthetic samples needed to be generated for the minority class.

Output: S_{syn}: set of generated synthetic samples

For each minority sample $x_i \in S_{min}$, find the samples distributed in its candidate minority class region, $\mathcal{C}(x_i) \leftarrow$ DiscovCMR(x_i, S_{min}, S_{maj}, q).
For each minority sample $x_i \in S_{min}$, identify the assistant seeds corresponding to its clean subregions, $\mathcal{A}(x_i) \leftarrow$ IdeCleanReg(x_i, $\mathcal{C}(x_i)$, S_{min}).
Generate the synthetic sample set S_{syn}, $S_{syn} \leftarrow$ Generate($\mathcal{A}$, S_{min}, S_{maj}, ζ).
function DiscovCMR$(x_i, S_{min}, S_{maj}, q)$
  $S \leftarrow S_{min} \cup S_{maj}$
  Rearrange the samples of $S \setminus \{x_i\}$ according to their distances to x_i in ascending order, and obtain $A(x_i) = \{x_{i1}, \ldots, x_{i(|S|-1)}\}$, where x_{i1} and $x_{i(|S|-1)}$ are the nearest and farthest samples from x_i, respectively.
  for $k \leftarrow 1 : |S| - 1$ **do**
    $count \leftarrow 0$
    if $x_{ik} \in S_{maj}$ **then**
      $count \leftarrow count + 1$
      if $count == q$ **then**
        $t \leftarrow \max\{1, k - q\}$
        break
      end if
    else
      $count \leftarrow 0$
    end if
  end for
  $\mathcal{C}(x_i) \leftarrow \{x_{i1}, x_{i2}, \ldots, x_{it}\}$
  return $\mathcal{C}(x_i)$
end Function
function IdeCleanReg$(x_i, \mathcal{C}(x_i), S_{min})$
  for $p \leftarrow 1 : |\mathcal{C}(x_i)|$ **do**
    $x_c \leftarrow (x_i + x_{ip})/2$
    $r_p \leftarrow \frac{1}{2} Distance(x_i, x_{ip})$
    $flag\_clean \leftarrow 1$
    for $l \leftarrow 1 : p - 1$ **do**
      if $x_{il} \notin S_{min}$ & $Distance(x_c, x_{il}) \leq r_p$ **then**
        $flag\_clean \leftarrow 0$
        break
      end if
    end for
    if $flag\_clean$ **then**
      $\mathcal{A}(x_i) \leftarrow \mathcal{A}(x_i) \cup \{x_{ip}\}$
    end if
  end for
  return $\mathcal{A}(x_i)$
end Function
function Generate$(\mathcal{A}, S_{min}, S_{maj}, \zeta)$
  $S_{syn} \leftarrow \varnothing$
  while $|S_{syn}| < \zeta$ **do**
    for each $x_i \in S_{min}$ **do**
      $x_s \leftarrow$ select a sample from $\mathcal{A}(x_i)$ randomly
      $\gamma \leftarrow$ generate a random number at [0,1]
      if $x_s \in S_{maj}$ **then**
        $\gamma \leftarrow \frac{1}{2}\gamma$
      end if
      $x_{syn} \leftarrow x_i + \gamma * (x_s - x_i)$
      $S_{syn} \leftarrow S_{syn} \cup \{x_{syn}\}$
      if $|S_{syn}| == \zeta$ **then**
        break
      end if
    end for
  end while
  return S_{syn}
end Function

Therefore, these three types of minority samples should be treated differently in the process of oversampling. Previous studies have attempted to explicitly divide the minority samples into the inland samples, borderline samples, and outliers [17], [30]. A common partition criterion is considering the distribution of k-NNs of each minority sample, i.e., a minority sample whose all/most/a small part of k-NNs are the majority samples is classified into the outliers/borderline samples/inland samples. However, it is difficult or even impossible to find a proper k value to distinguish all these three types of samples correctly, as a single uniform k-nearest neighborhood is hard to completely reflect the position characteristics of all the minority samples [5], [17], [31].

Our OREM does not need to partition the minority samples explicitly, it deals with the inland samples, borderline samples, and outliers discriminatively by adaptively determine the set of assistant seeds. For the inland minority samples, a large number of neighboring minority samples would be taken as their assistant seeds, because the CMR of an inland minority sample is almost a chunk of pure minority class region (e.g., x_i's CMR shown in Fig. 2a). For the borderline minority samples, OREM can select the assistant seeds prudently via identifying the clean subregions (e.g., consider Fig. 2b, x_{b1} would not be an assistant seed of x_b, as $S(x_b x_{b1})$ is a subregion containing the majority samples).

Moreover, since the borderline majority samples settled in the front of the majority class can form the clean regions with the borderline minority samples, they have a high likelihood of being the assistant seeds of the borderline minority samples. It is beneficial to expanding the distributed front of the minority samples towards the majority class regions ($S(x_b x_{b2})$ in Fig. 2b shows such a scenario). The CMRs of outliers would be small and specific because almost all of the neighboring samples are from the majority class (e.g., x_o's CMR shown in Fig. 2a). Hence, OREM can prevent the generated synthetic samples from spreading to outliers' vicinal regions.

Algorithm 2. OREM-M$(S, q, \boldsymbol{\zeta})$

Input: The whole sample set S; The counting parameter q; a $1 \times |C|$ vector $\boldsymbol{\zeta}$, which represents the number of synthetic samples generated for each class.
Output: S_{syn}: set of generated synthetic samples

1: $S_{syn} \leftarrow \varnothing$
2: For all the minority samples, find their corresponding assistant seed sets, $\mathcal{A} \leftarrow$ PrepareCleanReg$(S, \boldsymbol{\zeta})$
3: **while** $sum(\boldsymbol{\zeta}) \neq 0$ **do**
4: $\quad sampling\_distr \leftarrow \boldsymbol{\zeta}./sum(\boldsymbol{\zeta})$
5: $\quad c \leftarrow$ sampling a class with $sampling\_distr$
6: $\quad x_i^c \leftarrow$ sampling a sample from the sample subset S^c
7: $\quad x_{syn} \leftarrow$ Generate$(\mathcal{A}(x_i^c), x_i^c, S \setminus S^c, 1)$
8: $\quad x_{nn} \leftarrow$find the nearest neighbor of x_{syn} in $S \cup S_{syn}$
9: $\quad y_{nn} \leftarrow$acquire the class lable of x_{nn}
10: $\quad$ **if** $y_{nn} \neq c$ & x_{nn} is a synthetic minority sample **then**
11: $\quad\quad S_{syn} \leftarrow S_{syn} \setminus \{x_{nn}\}$
12: $\quad$ **else**
13: $\quad\quad S_{syn} \leftarrow S_{syn} \cup \{x_{syn}\}$
14: $\quad$ **end if**
15: $\quad \boldsymbol{\zeta}(c) \leftarrow \boldsymbol{\zeta}(c) - 1$
16: **end while**
17: **function** PrepareCleanReg$(S, \boldsymbol{\zeta})$
18: $\quad$ Divide S into the sample sets of classes $S^1, \ldots, S^{|C|}$
19: $\quad$ **for** $c \leftarrow 1 : |C|$ **do**
20: $\quad\quad$ **if** $\boldsymbol{\zeta}[c] == 0$ **then**
21: $\quad\quad\quad$ continue
22: $\quad\quad$ **end if**
23: $\quad\quad$ **for** each $x_i^c \in S^c$ **do**
24: $\quad\quad\quad \mathcal{C}(x_i^c) \leftarrow$ DiscovCMR$(x_i^c, S^c, S \setminus S^c, q)$
25: $\quad\quad\quad \mathcal{A}(x_i^c) \leftarrow$ IdeCleanReg$(x_i^c, \mathcal{C}(x_i^c), S^c)$
26: $\quad\quad$ **end for**
27: $\quad$ **end for**
28: $\quad$ **return** $\mathcal{A}$
29: **end Function**

Compared to k-NN-based interpolation oversampling methods, OREM avoided the use of neighbor parameter k. The parameter k is hard to set. If the value of k is small, nearly duplicate synthetic samples would be generated, especially when oversampling the inland minority samples. If k is set to be large, the problem of over generalization might be serious when the borderline minority samples and outliers are oversampled.

Different from clustering-based interpolation oversampling methods, OREM does not require that the minority samples present a clear clustering structure, and breaks down the barrier that the synthetic samples only fall in the convex hull of the original minority samples.

3.2 OREM-M: Extension of OREM to Multiclass Imbalance Problems

Multiclass imbalance problems are more challenging as compared to two-class imbalanced scenarios. The challenging factors include more complex class concepts, more severe class overlapping, within-class imbalance and small disjuncts due to the existence of multiple classes.

Among the above challenging factors, the class overlapping is an issue that needs to be specifically handled in terms of designing oversampling techniques, because there are significant differences between two-class and multiclass imbalances. In two-class imbalanced scenarios, the new class overlapping only occurs when the synthetic samples mistakenly fall in the regions of the majority class, while, in multiclass imbalance problems, the synthetic samples of a minority class can overlap with not only the majority samples, but also the samples of the other minority classes. Hence, the oversampling methods designed for multiclass imbalance problems should be focused more on addressing the problem of class overlapping.

Since OREM generates the synthetic samples in the clean regions without containing any majority sample, it can reduce the class overlapping appeared between the synthetic minority and majority samples effectively. However, there is no mechanism to avoid the class overlapping from the synthetic samples of different minority classes. To solve this problem, we combine OREM with an iteration procedure of synthetic sample generation to produce a multiclass-oriented oversampling algorithm OREM-M.

Algorithm 2 describes the process of OREM-M. OREM-M first precomputes the assistant seeds for the samples of each minority class by calling PrepareCleanReg() (lines 2). In PrepareCleanReg(), the assistant seeds of each minority sample are found by using DiscovCMR() and IdeCleanReg() of Algorithm 1 (lines 19-27 of Algorithm 2). Next, OREM-M iteratively generates the synthetic samples through the following process: 1) Sampling a minority class according to the distribution $sampling\_distr$ which is normalized from $\boldsymbol{\zeta}$ (lines 4-5). Note that a smaller minority class would have a higher probability of being selected to oversample. Hence, the smaller minority classes possess higher priorities to occupy the clean subregions. 2) Sampling a minority sample from the minority class under consideration, and attempting to create a synthetic sample for it (lines 6-7). 3) If the nearest neighbor of the created synthetic sample belongs to another minority class, this synthetic sample is dropped, so that the synthetic samples would not be positioned close to the regions of the other minority classes (lines 8-14). Furthermore, if the nearest neighbor is a synthetic sample from another minority class, this nearest neighbor sample is also removed (line 11). In this way, the class overlapping from the synthetic samples of different minority classes can be alleviated.

3.3 OREMBoost: A Hybrid Method Combining OREM and Boosting

The existing boosting-based ensemble methods addressing class imbalance problems apply resampling techniques to preprocess the training data of each round in boosting framework before building base classifiers [8], [9], [10].

There are two reasons to embed resampling techniques into boosting: 1) Resampling techniques can balance the class distribution of training data, so as to mitigate the bias of base classifiers towards the majority classes. 2) The diversity of training data can be encouraged by adding new minority samples or dropping part of majority samples, leading to an increase in the predictive diversity of base classifiers.

Algorithm 3. OREMBoost$(S, q, \boldsymbol{\zeta}, T)$

Input: Training set $S = \{x_i, y_i\}, y_i \in C, i = 1, \ldots, n$; base classifier h; number of iterations T; the counting parameter of OREM q; a $1 \times |C|$ vector $\boldsymbol{\zeta}$ indicated the number of synthetic samples generated for each class.
Output: $H(x) = \arg\max_{y \in C} \sum_{t=1}^{T} (\log \frac{1}{\beta_t}) h_t(x, y)$

1: $\mathcal{A} \leftarrow$ PrepareCleanReg$(S, \boldsymbol{\zeta})$
2: Initialize $D_i^1 \leftarrow 1/n, i = 1, 2, \ldots, n$
3: Initialize $w_{i,y}^1 \leftarrow D_i^1/(|C| - 1), i = 1, \ldots, n, y \in C \setminus \{y_i\}$
4: **for** $t \leftarrow 1 : T$ **do**
5: $\quad W_i^t \leftarrow \sum_{y \neq y_i} w_{i,y}^t$
6: $\quad q_{i,y}^t \leftarrow \frac{w_{i,y}^t}{W_i^t}$ for $y \neq y_i$
7: $\quad D_i^t \leftarrow \frac{W_i^t}{\sum_{i=1}^n W_i^t}$
8: $\quad S_t \leftarrow OREMOver(S, D^t, \mathcal{A}, \boldsymbol{\zeta})$
9: $\quad h_t \leftarrow$ train a weak classifier using S_t
10: $\quad$Calculate the pseudo-loss of h_t:
$\quad\quad \epsilon_t \leftarrow \frac{1}{2} \sum_{i=1}^n D_i^t (1 - h_t(x_i, y_i) + \sum_{y \neq y_i} q_{i,y}^t h_t(x_i, y))$
11: $\quad$Set $\beta_t \leftarrow \epsilon_t/(1 - \epsilon_t)$
12: $\quad$Update $w_{i,y}^{t+1} \leftarrow w_{i,y}^t \beta_t^{(1/2)(1 + h_t(x_i, y_i) - h_t(x_i, y))}$ for
$\quad\quad i = 1, \ldots, n, y \in C \setminus \{y_i\}$
13: **end for**
14: **function** OREMOver$(S, D^t, \mathcal{A}, \boldsymbol{\zeta})$
15: $\quad S_t \leftarrow \varnothing$
16: $\quad$Divide S into the sample sets of classes $S^1, \ldots, S^{|C|}$
17: $\quad$**for** $c \leftarrow 1 : |C|$ **do**
18: $\quad\quad \hat{S}^c \leftarrow$ sampling a sample set of size $|S^c|$ with replacement according to the distribution $D^t(S^c)$
19: $\quad\quad$**for** $i \leftarrow 1 : \boldsymbol{\zeta}[c]$ **do**
20: $\quad\quad\quad x_i^c \leftarrow$ sampling a sample from $\hat{S}^c$ randomly
21: $\quad\quad\quad x_{syn} \leftarrow$ Generate$(\mathcal{A}(x_i^c), x_i^c, S \setminus S^c, 1)$
22: $\quad\quad\quad S_t \leftarrow S_t \cup \{x_{syn}\}$
23: $\quad\quad$**end for**
24: $\quad\quad S_t \leftarrow S_t \cup \hat{S}^c$
25: $\quad$**end for**
26: $\quad$**return** S_t
27: **end Function**

The integration way of our OREMBoost with respect to OREM and boosting is similar with traditional boosting-based ensemble methods such as SMOTEBoost [9], RUSBoost [7], and RBBoost [32]. Algorithm 3 presents the details of OREMBoost, in which two places are different with previous ensemble methods, i.e., lines 1 and 8. In line 1, OREMBoost precomputes the assistant seeds based on whole training data for each minority sample of all the minority classes by utilizing function PrepareCleanReg(). In line 8, the precomputed assistant seeds of all the minority samples (i.e., $\mathcal{A}$), and the weight distribution of tth round (i.e., D^t) are together used to oversample the original imbalanced data S, which is implemented by function OREMOver(). By adding these two lines of code, the balanced dataset $\hat{S}_t$ training tth base classifier can be obtained.

By means of the usage of PrepareCleanReg() and OREMOver(), the identification of assistant seeds and the generation of synthetic samples are embedded into boosting framework, respectively. It is based on two following considerations. First, since OREM takes advantage of generating synthetic samples into the clean regions to reliably broaden the minority regions, the correct identification of the clean regions is a key. As shown in function PrepareCleanReg() of Algorithm 2, for any minority sample x_i^c belonging to a minority class c, the assistant seeds are acquired based on the whole training data, i.e., S^c and $S \setminus S^c$ are used as the minority and majority sample sets, respectively. It can ensure that the found clean regions are reliable. Second, the acquisition of assistant seeds is the most time-consuming part of OREM because of requiring to identify the clean subregions near the original minority samples. Placing function OREMOver() outside the iteration process of boosting can reduce the time complexity of OREMBoost dramatically.

In addition, several points are worth mentioning on the implementation of OREMOver(). In OREMOver(), the original data S is divided into the sample subsets of classes, i.e., $S^1, \ldots, S^{|C|}, c = 1, 2, \ldots, |C|$. Then, for each S^c, a sample set $\hat{S}^c$ is sampled with replacement according to the weight distribution on S^c (line 18). Hence, some original samples of S^c may not be included into $\hat{S}^c$, while the samples that are frequently misclassified may occur multiple times in $\hat{S}^c$ due to their high weights. OREMOver() only oversamples the samples of $\hat{S}^c$ (lines 19-23). It helps to augment the diversity of data as $\hat{S}^c$ is variable relative to S, and meanwhile maintains the learning mechanism that the samples with higher weights would receive more attention via generating more synthetic samples for them.

3.4 Computational Complexity Analysis

Let us define the number of samples, minority samples, and synthetic samples by n, n_{min}, and n_s respectively, the number of classes by $|C|$. The most time-consuming basic operation in OREM and OREM-M is the distance computation between samples. For the convenience of analysis, we only count the execution times of basic operation for the proposed methods.

OREM involves two steps: 1) finding the CMR for each minority sample, x_i, and 2) identifying the clean subregions within the CMR of x_i. In the first step, OREM needs to obtain x_i's nearest neighbor list to examine the distribution of neighboring samples. It needs to calculate the distances between x_i and each of the other samples, with the complexity of $(n - 1)$. In the second step, the time complexity depends on the sample size distributed in its CMR. Since there are at most $(q - 1) * (n_{min} - 1)$ majority samples located in x_i's CMR, each minority sample x_i has at most $n_{cmr} = min(q * (n_{min} - 1), n - 1)$ candidate assistant seeds. Further, to identify whether the area between x_{ip} (i.e., pth nearest candidate assistant seed from x_i) and x_i is a clean subregion, OREM first needs to compute the midpoint of x_{ip} and x_i, denoted by x_c, and then to calculate the distances between x_c and each of the majority samples in the first $p - 1$ candidate assistant seeds

TABLE 2
Description of Characteristics of Experimental Two-Class Datasets

ID	Dataset	Minority Class	Majority Class	Class Distribution	|F|	IR
1	Diabetes	'1'	'0'	268/500	8	1.87
2	Breast cancer wisconsin	'4'	'2'	241/458	9	1.90
3	Biodeg	'RB'	'NRB'	356/699	41	1.96
4	Seeds	'1'	other classes	70/140	7	2.00
5	BreastTissue	'Car'&'Con'	other classes	35/71	9	2.03
6	ForestTypes	'h'&'o'	other classes	169/354	27	2.09
7	Yeast	'NUC'	other classes	429/1055	8	2.46
8	Haberman	'died'	'survived'	81/225	3	2.78
9	Wifilocalization2	'2'	other classes	500/1500	7	3
10	Wifilocalization3	'3'	other classes	500/1500	7	3
11	Parkinsons	'0'	'1'	48/147	22	3.06
12	Automobile	'2'&'6'	other classes	49/156	71	3.18
13	Transfusion	'1'	'0'	178/570	4	3.20
14	WPBC	'recur'	'nonrecur'	47/151	33	3.21
15	Vehicle	'van'	other classes	199/647	19	3.25
16	Vertebral	'DH'	other classes	60/250	6	4.17
17	Vowel-28	'2'&'8'	other classes	180/810	9	4.5
18	Ecoli-pp	'pp'	other classes	52/255	7	4.90
19	Win-red	'4'&'7'	'5'&'6'	252/1319	11	5.23
20	Laryngeal	'0'	other classes	53/300	16	5.66
21	Flowmeter C	'2'	other classes	23/158	43	6.87
22	ERA	'7'&'8'	other classes	119/881	4	7.40
23	Voice	'2'&'4'&'9'	other classes	50/378	10	7.56
24	Vowel-6	'6'	other classes	90/900	9	10
25	Vowel-9	'9'	other classes	90/900	9	10
26	Glass	'veh_win_fl'	other classes	17/197	9	11.59
27	Win-white	'4'&'8'	'5'&'6'&'7'	338/4537	11	13.42
28	Risk factor scervical cancer	'1'	'0'	55/803	35	14.60

|F| and IR denote the number of features and the imbalanced ratio (#majority class samples/#minority class samples), respectively.

of x_i. In summary, the times of distance computation is at most $\sum_{p=1}^{n_{cmr}}(p-1) = \frac{n_{cmr}*(n_{cmr}-1)}{2} \leq \frac{(n-1)(n-2)}{2}$. Summing the first and second steps, the worst case complexity of OREM is $n_{min} * ((n-1) + \frac{(n-1)(n-2)}{2}) \in O(n_{min} * n^2)$.

Based on the complexity of OREM, the time cost of OREM-M and the computational overhead caused by OREM oversampling in OREMBoost are all $O(n_{min} * n^2)$. However, this worst complexity $O(n_{min} * n^2)$ is analyzed from a very demanding situation, i.e., the CMR of each minority sample contains all of the majority samples. It is extremely hard or even impossible (when the imbalance degree is higher than $q-1$) to appear. In the supplementary material, available online, we investigated the computational complexity of OREM and OREM-M in practice. We found that the distance computation operations of OREM and OREM-M are 7.123 and 3.755 times $n_{min} * n$ respectively, where $O(n_{min} * n)$ is one of most common complexities for oversampling algorithms. The detailed analysis and results can be found in Section S2 of the supplementary material, available online.

4 Experimental Study

We validate the effectiveness of the proposed methods through a set of empirical studies. In Section 4.1, the performance of OREM is assessed on two-class imbalanced datasets. Section 4.2 evaluates the effectiveness of OREM-M on multiclass imbalanced datasets. Finally, we verify the usefulness of OREMBoost in Section 4.3.

It should be highlighted that due to space constraints, only the essential *ingredients* of experimental study are presented in the main paper. There is lots of important content which is placed in a 54-page supplementary material, available online. The content includes the computational complexity analysis for the proposed algorithms (Section S2), an investigation on the performance advantage of OREM (Section S3), an empirical analysis on the impact of varying oversampling degrees (Section S4), a visual comparison of oversampling algorithms (Section S5), and an in-depth analysis of OREMBoost based on accuracy and diversity (Section S7). We highly recommend the reader to reference these elements enhancing the completeness of our work.

4.1 Effectiveness Analysis of OREM on Two-Class Imbalanced Datasets

4.1.1 Experimental Setting

Datasets. 28 two-class imbalanced datasets are selected from UCI [33]. Table 2 shows the detailed characteristics of these datasets (see "Minority Class" column and "Majority Class" column of Table 2).

Classifiers. We selected three commonly used classifiers. They are C4.5, NN, and SVM. The implementations of C4.5 and NN are adopted J48 and MLP in Weka [35], respectively. The default Weka parameters are modified as the following: C4.5 uses Laplace smoothing and unpruned strategy; NN is trained with 500 epochs at a learning rate 0.1, and the number of hidden neurons is set to 10.

TABLE 3
Parameter Settings of the Compared Oversampling Algorithms

Oversampling Algorithms	Parameters of Algorithms
SMOTE [4]	$k = 5$
MWMOTE [13]	$k1 = 5, k2 = 3, k3 = \lvert S_{min} \rvert /2, C_p = 3, C_f(th) = 5, CMAX = 2$
INOS [23]	$k = 15, Q = 5, r = 0.7$
RACOG [26]	$\beta = 100, \alpha = 20$
wRACOG [26]	$slide\_win = 10, threshold = 0.02, wrapper = corresponding\, base\, classifier$
RBO [34]	$\gamma = 0.01, stepsize = 0.0001, iterations = 5000, p = 0.001$
GDO [15]	$k = 5, \alpha = 1$
FWSMOTE [12]	$Feature\, ranking = Fisher, k = 5, r = \frac{\lvert F \rvert}{2}, p = \infty, OWA\, quantifier = Basic\, RIM, \alpha = 0.4$
OREM	$q = 5$
MDO [19]	$K1 = 5, K2 = 10$
SMOM [5]	$k1 = 12, k2 = 8, rTh = 5/8, nTh = 10, w1 = 0.2, w2 = 1/2, r1 = 1/3, r2 = 0.2$
MRBO [6]	$\gamma = 0.01, stepsize = 0.0001, iterations = 5000$
OREM-M	$q = 5$

LibSVM [36] is employed as the implementation of SVM. The linear kernel is used for speeding up training. The penalty parameter C is optimized through the build-in 5-fold cross validation of LIBSVM. The search scope of parameter C is $\{2^{-3}, 2^{-2}, \ldots, 2^{10}\}$.

Assessment Measures. F1, G-mean, and AUC [37] are together used as the skew-insensitive measures to evaluate the performance of classifiers. The performance value on each dataset is the average result of running 10 times with stratified 5-fold cross validation.

Statistical Analysis. The experimental objective is to validate whether the proposed method has a significant advantage compared with the other algorithms. Following the suggestion of [38], the multiple comparisons are performed to achieve this objective of statistical analysis, which each comparison applies the Wilcoxon signed-rank test to compare a pair of approaches.

Rebalanced Way. The minority class of the imbalanced data is oversampled until acquiring a complete balanced class distribution.

4.1.2 Comparison With Representative Oversampling Approaches

In this experiment, we selected eight representative oversampling methods to compare with the proposed OREM. They are k-NN-based interpolation oversampling methods SMOTE [4] and FWSMOTE [12], clustering-based interpolation oversampling MWMOTE [13], structure-preserving oversampling INOS [23], probability distribution-based oversampling RACOG and wRACOG [26], and two recently proposed oversampling algorithms RBO [34] and GDO [15]. All the parameters of these methods are used the recommend values in the corresponding literature, which are summarized in Table 3.

Due to space restrictions, we put the detailed results of all the compared methods in Tables S12, S13, and S14 of the supplementary material, available online. To verify whether there are significant differences between OREM and each of the other oversampling methods, we perform the Wilcoxon signed-rank tests on the performance values of Tables S12, S13, and S14, available online. The results of significance tests are listed in Table 4, where "Original" represents the predicted performance on original imbalanced data. One can see that OREM can achieve statistically superior performance in most of the cases, which demonstrates OREM is highly effective as compared to the other two-class oversampling algorithms.

To trace the performance advantage of OREM over the other oversampling methods, we compare the recall, precision, and balance accuracy (BA) performance for all the considered methods. The experimental results demonstrate that OREM can achieve good, moderate, and best mean rankings

TABLE 4
p-Values of Wilcoxon Signed-Rank Tests for the Comparisons Between OREM and Each of Original, SMOTE, MWMOTE, INOS, RACOG, wRACOG, RBO, GDO, and FWSMOTE

OREM vs	C4.5			NN			SVM		
	F1	G-mean	AUC	F1	G-mean	AUC	F1	G-mean	AUC
Original	**0.00042497++**	**3.2037e-07++**	**0.00029736++**	**9.3192e-05++**	**7.4506e-09++**	0.17752+	**5.6773e-06++**	**7.4506e-09++**	**0.00027929++**
SMOTE	0.30248+	**0.040214++**	**0.039124++**	0.30252+	**7.9788e-05++**	**0.019324++**	0.99553-	**0.039128++**	**0.097964+***
MWMOTE	**0.0038161++**	**2.7508e-05++**	**0.00019822++**	0.43466+	**1.7956e-06++**	**0.0033692++**	0.371+	**0.018143++**	**0.0043194++**
INOS	**0.014341++**	0.4085+	0.31336+	0.28421+	**0.010378++**	0.46161+	**0.040712++**	**0.00022378++**	**0.039124++**
RACOG	**7.9647e-06++**	**3.3304e-06++**	**5.9605e-08++**	**6.5565e-07++**	**7.4506e-09++**	**3.7253e-08++**	**0.0083745++**	**0.00064678++**	**0.0031605++**
wRACOG	**1.4901e-08++**	**3.7253e-08++**	**1.4901e-08++**	**7.4506e-09++**	**7.4506e-09++**	**2.2352e-08++**	**8.2016e-05++**	**1.1049e-05++**	**0.00027351++**
RBO	**0.094564+***	**0.0001057++**	**0.030285++**	0.91522+	**0.0082149++**	**0.040149++**	0.46863-	0.3161+	0.82734+
GDO	**0.0001197++**	**0.0097082++**	**0.0031605++**	**6.4753e-05++**	**0.0019095++**	**0.0017267++**	**1.514e-05++**	**8.2016e-05++**	**0.0001674++**
FWSMOTE	**0.088898+***	**0.017074++**	**0.049785++**	0.35308+	**0.0003904++**	**0.0015174++**	0.11583-	0.52641+	0.24574+

"+*" and "++" signify that OREM is statistically better than the compared algorithm under consideration at a significant level of 0.1 and 0.05, respectively.
"+" denotes that OREM is only quantitatively better, whereas "-" implies the contrary.

TABLE 5
Results of the Wilcoxon Signed-Rank Tests Between OREM and Each of its Variants

Classifier	Evaluation Measure	OREM versus OREM w/o S1			OREM versus OREM w/m S1			OREM versus OREM w/o S2			OREM versus OREM w/m S2		
		R^+	R^-	T	R^+	R^-	T	R^+	R^-	T	R^+	R^-	T
C4.5	F1	277.5	128.5	**128.5(+*)**	250.5	155.5	155.5(+)	214	192	192(+)	181	225	181(-)
	G-mean	143	263	143(-)	147.5	258.5	147.5(-)	199	207	199(-)	181.5	224.5	181.5(-)
	AUC	153.5	252.5	153.5(-)	149	257	149(-)	196.5	209.5	196.5(-)	198	208	198(-)
NN	F1	324	82	**82(++)**	261.5	144.5	144.5(+)	300.5	105.5	**105.5(++)**	327.5	78.5	**78.5(++)**
	G-mean	256	150	150(+)	202	204	202(-)	278	128	**128(+*)**	297.5	108.5	**108.5(++)**
	AUC	171.5	234.5	171.5(-)	186	220	186(-)	157	249	157(-)	202	204	202(-)
SVM	F1	281	125	**125(+*)**	286.5	119.5	**119.5(+*)**	234.5	171.5	171.5(+)	240.5	165.5	165.5(+)
	G-mean	312.5	93.5	**93.5(++)**	286	120	**120(+*)**	228	178	178(+)	240.5	165.5	165.5(+)
	AUC	276.5	129.5	129.5(+)	245	161	161(+)	262.5	143.5	143.5(+)	262	144	144(+)

T(i.e., $min(R^+, R^-)$) no larger than 116 (/130) indicates there is statistically difference with $\alpha = 0.05(/\alpha = 0.1)$, which is highlighted in bold.

in terms of recall, precision, and BA on all three base classifiers, respectively. It suggests that our OREM can provide high accuracy on the minority class without severely jeopardizing the accuracy of the majority class as compared to the other oversampling techniques. The detailed experimental results and analysis are provided in Section S3 of the supplementary material, available online.

4.1.3 Comparison With OREM's Variants

OREM consists of two main steps: 1) It finds the CMR around each minority sample. All the samples distributed in the CMR are the candidate assistant seeds of the considered minority sample. 2) It identifies those clean subregions within the CMR of each minority sample. The candidate assistant seeds corresponding to the clean subregions are eventually used as the assistant seeds of the considered minority sample. Based on the above steps, the following OREM's variants can be straightforwardly introduced.

- OREM without step 1 (OREM w/o S1) : It removes the step 1 of OREM, i.e., all of the other samples are the candidate assistant seeds of the considered minority sample.
- OREM with modified step 1 (OREM w/m S1): Unlike OREM, this variant determines the CMR for a minority sample as its maximal k-nearest neighborhood dominated by the minority samples.
- OREM without step 2 (OREM w/o S2): It gets rid of the step 2 of OREM, i.e., all candidate assistant seeds are directly identified as qualified assistant seeds.
- OREM with modified step 2 (OREM w/m S2): It relaxes the condition of being the assistant seeds, i.e., the candidate assistant seeds, that correspond to the subregions dominated by the minority samples, can be taken as the assistant seeds.

The performance results of OREM as well as its variants on the experimental two-class imbalanced datasets are summarized in Tables S21, S22, and S23 of the supplementary material, available online. We conduct the Wilcoxon signed-rank tests based on Tables S21, S22, and S23, available online. The significance test results are listed in Table 5. From this table, we can obtain two observations: 1) OREM is not significantly inferior to all the variants in any measure and classifier. 2) OREM is obviously better than OREM w/o S2 and OREM w/m S2 in NN classifier, and there are significance differences between OREM and each of OREM w/o S1 and OREM w/m S1 in terms of F1 and G-mean with SVM classifier. Hence, OREM is more robust as compared to four straightforward variants. A possible explanation is that if OREM's step 1 is eliminated (i.e., OREM w/o S1), the issue, the hollow regions in feature space are identified as the clean regions, would be aggravated as the CMRs are not constrained in the regions near the original minority samples. Naturally, the synthetic samples falling in the hollow regions could weaken the utility of oversampling. In addition, neglecting OREM's step 2 or relaxing the condition of being assistant seeds (i.e., OREM w/o S2 and OREM w/m S2) might increase the risk of filling the synthetic samples into the majority regions, consequently confusing subsequent classification learning.

4.1.4 Empirical Study on the Impact of Counting parameter

The counting parameter q is the only parameter of OREM. It can affect the size of CMRs. A small q (e.g., 3) might result in the CMRs cannot be expanded fully. On the contrary, a very large q can cause the found CMRs contain a considerable portion of hollow regions, and increase the computational cost for further identifying the clean subregions.

We provide the AUC results of OREM on the datasets *diabetes (Db)*, *breastTissue (BT)*, *vehicle (Vc)*, and *glass (Gla)*, *automobile (Am)*, *flowmeter C (FmC)*, and *risk factor scervical cancer (RFSC)* to illustrate how the performance of OREM varies with different q (Fig. 3). The results indicate OREM performs better when q is 5 or 7. The performance of OREM generally degrades from $q \geq 9$. Hence, the values between 5 and 7 would be reasonable for the setting of q.

However, the experimental datasets in Table 2 are low-dimensional. OREM implicitly assumes that if q majority samples continuously appear in the nearest neighbor list, a dense majority region is encountering. In high-dimensional data, the rationality of this assumption would be depreciated, because the discrimination of the distances between samples is weakened. In the Section S6 of the supplementary material, available online, we provided an additional experiment to verify whether OREM is still an outstanding alternative for high-dimensional imbalanced time series

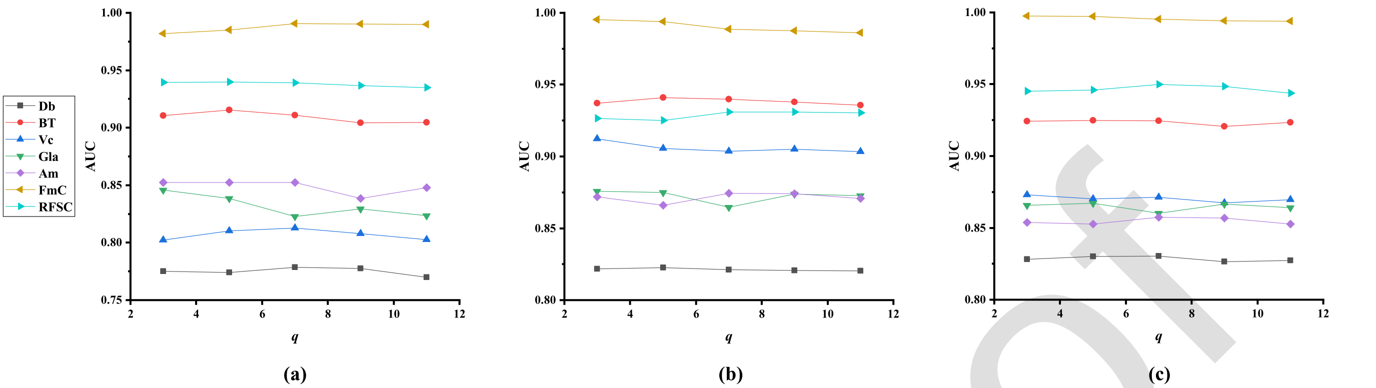


Fig. 3. Figure illustrating how the AUC performance of OREM varies with the counting parameter q. (a) C4.5 classifier. (b) Neural network classifiers. (c) Support vector machine classifier.

data and gene expression microarray data. The results show that OREM does not acquire a clear advantage over other oversampling methods. We deem that the customized oversampling solution should be developed so as to handle the high-dimensional imbalance problems effectively.

4.2 Effectiveness Analysis of OREM-M on Multiclass Imbalanced Datasets

4.2.1 Experimental Setting

Datasets. 21 multiclass real-world imbalanced datasets are selected for this experiment. They contain widely used ordinal regression benchmark datasets [39] and UCI datasets [33]. Table 6 summarizes the characteristics of these datasets.

Classifiers and *statistical analysis* are used the same experimental settings with the effectiveness analysis of OREM. For *assessment measures*, since F1, G-mean, and AUC cannot be directly applied into assessing the classification performance on multiclass imbalanced data, we use their corresponding multiclass extension versions, i.e., macro-F1, MG [40], and MAUC [41].

Rebalanced Way. Given that multiple majority classes may exist in multiclass imbalanced data, the resampling strategy, that all the classes are oversampled to have the same size with the maximum classes, would be time-consuming and might only acquire limited profit when oversampling the majority classes. We follow the suggestion of [42], i.e., only the classes with imbalance degrees greater than 1.2 are oversampled, until the imbalance degree of each class is not higher than 1.2. We define the imbalance degree of individual class as (1).

$$ID_{c_i} = \frac{\sum_{q \neq i} n_{c_q}}{(|C| - 1) \cdot n_{c_i}}, \tag{1}$$

where |C| is the number of classes, n_{c_i} and n_{c_q} are the number of samples in classes c_i and c_q, respectively. To ensure all the classes having imbalanced degrees lower than 1.2, the number of synthetic samples needed to generate for each class is iteratively calculated as [42].

TABLE 6
Description of Characteristics of Experimental Multiclass Datasets

ID	Dataset	Class Distribution	|C|	|F|	IR
1	Vertebral	**60**/150/**100**	3	6	2.67
2	Laryngeal	**53**/218/**82**	3	16	5.32
3	New-thyroid	150/**35**/**30**	3	5	7.45
4	Toy	68/87/79/**35**/**31**	5	2	8.9
5	Voice3	273/**102**/**38**	3	10	9.58
6	Wisconsin5	67/41/43/**24**/**19**	5	32	9.74
7	Cleveland	160/**54**/**35**/**35**	4	13	10.35
8	Voice6	100/**43**/**58**/115/**59**/**38**	6	10	12.11
9	Vowel5	**180**/**90**/360/270/**90**	5	10	13.83
10	Vowel4	360/**90**/**90**/450	4	10	14.25
11	Dermatology	111/**60**/71/**48**/**48**/**20**	6	35	17.58
12	Page-blocks	329/**28**/**88**/**115**	4	10	20.9
13	SWD	**32**/352/399/**217**	4	4	28.84
14	Yeast	463/429/**244**/**163**/**86**/**74**	6	8	30.53
15	Auto5	91/131/101/**59**/**10**	5	7	37.52
16	LEV	**93**/280/403/**197**/**27**	5	4	40.41
17	Leaves plant	128/**16**/**16**/**16**/**16**/80	6	64	44.6
18	Stock10	104/119/110/**64**/108/168/103/104/**48**/**22**	10	9	47.1
19	Plates faults	**158**/**190**/391/**72**/**55**/402/673	7	27	61.1
20	Abalone	**391**/568/689/**103**/**67**/**58**	6	10	62.9
21	Housing10	**22**/**55**/85/154/84/**39**/**29**/**7**/**10**/**21**	10	13	152.35

The minority classes are highlighted in ***bold*** in "Class Distribution".
The overall imbalance ratio (IR) on multiclass imbalanced datasets is computed as [5].

TABLE 7
p-Values of the Wilcoxon Signed-Rank Tests for the Comparisons Between OREM-M and Each of Original, SMOTE, SMOM, MDO, MC-RBO

OREM-M vs	C4.5			NN			SVM		
	macro-F1	MG	MAUC	macro-F1	MG	MAUC	macro-F1	MG	MAUC
Original	0.19977+	**1.812e-05++**	**0.014166++**	**0.035823++**	**2.861e-06++**	**0.0094357++**	**0.01578++**	**7.6294e-06++**	**3.1471e-05++**
SMOTE	0.18182+	**0.090437+***	**0.031147++**	**0.0014067++**	0.13961+	0.22552+	**0.04114++**	**0.0053291++**	**0.03899++**
SMOM	0.23927+	**0.034555++**	**0.087003+***	**0.025691++**	0.45237+	0.77857+	**0.037539++**	**0.010361++**	**0.074398+***
MDO	**0.045213++**	**2.3842e-05++**	**0.088799+***	**0.039237++**	**1.3351e-05++**	**0.048361++**	0.14696+	**0.022335++**	**0.002387++**
MC-RBO	**0.028319++**	**6.6757e-06++**	**3.8147e-06++**	**0.037848++**	**0.05354+***	0.29966+	**0.0054455++**	**0.025822++**	**0.0014286++**

"+*" and "++" signify that OREM-M is statistically better than the compared algorithm under consideration at a significant level of 0.1 and 0.05, respectively.
"+" denotes that OREM is only quantitatively better, whereas "-" implies the contrary.

4.2.2 Comparison With Existing Multiclass Oversampling Approaches

Four oversampling algorithms are selected to compare with OREM-M. They are SMOTE, MDO [19], our previous work SMOM [5], and MC-RBO [6], where the latter three methods are exclusively designed for multiclass imbalance problems. All the compared methods employ the recommend parameter values in the corresponding literature. The specific parameter settings are summarized in Table 3.

The macro-F1, MG, and MAUC values of all the compared algorithms with C4.5, NN, and SVM classifiers can be found in Tables S24, S25, and S26 of the supplementary material, available online. Based on the results of these tables, we carry out the Wilcoxon signed-rank tests to assess whether the significant differences between OREM-M and each of SMOTE, MDO, SMOM, and MC-RBO exist. The results are shown in Table 7. From this table, we can observe that OREM-M is statistically better than the other comparative oversampling methods in most of the cases. It suggests that there is a high competitiveness in terms of OREM-M to copy with multiclass imbalance problems.

4.2.3 Empirical Study on Iterative Generation Procedure

By combining the iterative generation procedure, OREM is generalized to OREM-M. Another straightforward extension of OREM is adopting one-vs-all way to handle multiclass imbalance problems, i.e., the class to be oversampled is considered as the minority class in each time, and then all of the remaining classes are taken as the majority class. We call this straightforward extension OREM-S.

To verify whether the iterative generation procedure can promote the performance of OREM, we compare OREM-M with OREM-S. Table 8 presents the average results and mean rankings of OREM-M and OREM-S over the experimental multiclass datasets (note: the complete results are provided in Table S27), available online. We can observe that OREM-M outperforms OREM-S in the most of cases, which demonstrates that the iterative generation way of synthetic samples could indeed boost the ability of OREM to deal with multiclass imbalance datasets. Hence, restraining the class overlapping among the synthetic samples of different minority classes is a point worth exploiting in terms of improving the performance of multiclass oversampling algorithms.

A next natural question is that whether this iterative generation procedure can also benefit to other oversampling algorithms. To answer this question, we combine SMOTE with this procedure (called SMOTE-M), then compare SMOTE with SMOTE-M. The average performance results and mean ranks of SMOTE and SMOTE-M are presented in Table 8 (note: the detailed experimental results can be found in Table S28), available online. It is not inspiring that SMOTE-M does not show more competitive than SMOTE.

TABLE 8
The Average Performance Results and Mean Ranks of OREM (/SMOTE) Combining and Without Combining Iterative Generation Procedure of Synthetic Samples Over the Experimental 21 Multiclass Imbalanced Datasets

Classifiers	Measures	Average performance		Mean ranks		Average performance		Mean ranks	
		OREM-S	OREM-M	OREM-S	OREM-M	SMOTE	SMOTE-M	SMOTE	SMOTE-M
C4.5	macro-F1	0.6329	**0.6341**	1.57	**1.43**	**0.6319**	0.6305	**1.48**	1.52
	MG	0.5578	**0.5646**	**1.48**	1.52	**0.5507**	0.5458	**1.38**	1.62
	MAUC	0.8263	**0.8275**	1.57	**1.43**	**0.8250**	0.8232	**1.43**	1.57
NN	macro-F1	0.6345	**0.6363**	1.71	**1.29**	0.6303	**0.6362**	1.81	**1.19**
	MG	**0.5729**	0.5680	1.52	**1.48**	0.5623	**0.5725**	1.71	**1.29**
	MAUC	0.8559	**0.8566**	1.57	**1.43**	0.8552	**0.8560**	1.55	**1.45**
SVM	macro-F1	0.6174	**0.6212**	1.71	**1.29**	0.6167	**0.6176**	**1.48**	1.52
	MG	0.5317	**0.5408**	1.67	**1.33**	**0.5309**	0.5287	1.55	**1.45**
	MAUC	0.8594	**0.8601**	1.60	**1.40**	0.8585	**0.8589**	**1.50**	**1.50**

TABLE 9
Ensemble Algorithms Used in the Experimental Study

Method (Abbr.)	Brief Description
AdaBoost.M2 (AdaB)	AdaBoost's multiclass extension with confidence estimates
SMOTEBoost (SMOTEB)	AdaBoost.M2 combining with SMOTE in each iteration
RUSBoost (RUSB)	AdaBoost.M2 combining with random undersampling in each iteration
EasyEnsemble (EasyE)	Balanced Bagging with random undersampling of the majority class, and learning each bag with AdaBoost.M2
BalanceCacade (BalanceC)	Similar to BalanceC, but removing correctly classified majority samples in each bagging iteration
RBBoost (RBB)	AdaBoost.M2 combining with random balance resampling in each iteration
SplitBal	Converting imbalanced data into multiple balanced subsets using random splitting, building ensemble on the balanced subsets
MBSBoost (MBSB)	AdaBoost.M2 combining with model-based oversampling in each iteration
Self-paced ensemble (SPE)	Spliting the majority samples into different bins according to hardness levels, then building ensemble via a self-paced undersampling procedure over every bin
Dual-LexiBoost (DLexiB)	The dual to the primal LexiBoost which uses a two staged lexicographic linear programming to determine the component classifier weights
Hybrid data-level ensemble (HDE)	Bootstrapping from the original dataset, then performing a margin-based undersampling and a diversity-enhancing oversampling on each bootstrap

The main difference between SMOTE and OREM is that SMOTE cannot ensure the synthetic samples only fall in the clean regions. Checking whether the nearest neighbor of the generated synthetic sample belongs to a different minority class, might not be enough to effectively suppress the class overlapping, especially occurred between the synthetic minority and majority samples.

4.3 Effectiveness Analysis of OREMBoost

4.3.1 Experimental Setting

Given that most of the ensemble approaches solving class imbalance problems can only deal with two-class cases, we conduct the experiments on two-class and multiclass scenarios based on the datasets of Tables 2 and 6, respectively. A brief description for the ensemble algorithms added into the experiments is summarized in Table 9. In all the ensemble methods, classification and regression tree is employed as base classifier. The number of base classifiers is set to 40 [18], [20]. The trained data in each iteration is resampled into a balanced class distribution.

4.3.2 Comparison With Prevailing Ensemble Methods Addressing Class Imbalance Problems

For two-class imbalanced scenarios, ten representative ensemble methods are added into the experimental comparison. They are AdaB [43], SMOTEB [9], RUSB [7], BalanceC [8], EasyE [8], RBB [32], SplitBal [44], SPE [45], DLexiB [46], and HDE [47].

The detailed performance values of all the compared ensemble methods are provided in Table S29 of the supplementary material, available online. We carry out the Wilcoxon signed-rank tests based on the results of Table S29, available online. The significance test results between OREMBoost and each of the other compared ensemble methods are presented in Table 10. From Table 10, we can find that 1) all the significance differences can be observed between OREMBoost and each of the other methods in F1 and AUC; 2) for G-mean performance, OREMBoost is significantly better than AdaB, SMOTEB, RBB, and SplitBal, while it underperforms the ensemble solutions combining with undersampling, i.e., BalanceC, EasyE, SPE, and HDE. We deem that the excellent G-mean performances in these four ensemble methods is because G-mean value is generally more sensitive to the increase of the accuracy of minority class, and they acquire relatively higher prediction accuracy on the minority class. Concretely, a considerable space of the majority class is emptied due to removing the majority samples. It is directly beneficial to modeling the concept of the minority class in a broader region, consequently improving the recall of the minority samples. However, BalanceC, EasyE, SPE, and HDE do not show superior performance in F1 and AUC. It indicates the performance of the majority class might be seriously damaged, dragging the F1 and AUC down.

For multiclass imbalanced scenarios, we select AdaB, SMOTEB, MBSB [10], and DLexiB to compare our OREMBoost. The experimental results of AdaB, SMOTEB, MBSB, and OREMBoost on 21 multiclass imbalanced datasets are summarized in Table S30, available online. The Wilcoxon signed rank tests are performed on Table S30, available online. Table 11 summarizes the corresponding significance test results. From this

TABLE 10
p-Values of the Wilcoxon Signed-Rank Tests for the Comparisons Between OREMBoost and Each of Ten Representative Ensemble Methods

Metrics	OREMBoost vs									
	AdaB	SMOTEB	RUSB	EasyE	BalanceC	RBB	SplitBal	SPE	DLexiB	HDE
F1	**0.000267++**	**0.012483++**	**0.00400++**	**0.01563++**	**0.010071++**	**0.001274++**	**0.000473++**	**0.03793++**	**8.2e-07++**	**0.000172++**
G-mean	**2.31e-07++**	**0.00063++**	0.2428+	0.12418-	0.17091-	**1.1e-05++**	**0.088911+***	0.15522-	**0.00115++**	0.95532-
AUC	**0.005613++**	**0.085353+***	**8.2e-05++**	**0.017073++**	**0.00471++**	**0.000237++**	**4.2e-05++**	**0.0556+***	**7.4506e-09++**	**0.00244++**

TABLE 11
p-Values of the Wilcoxon Signed-Rank Tests for the Comparisons Between OREMBoost and Each of AdaB, SMOTEB, MBSB, and DLexiB

Measures	OREMBoost vs			
	AdaB	SMOTEB	MBSB	DLexiB
macro-F1	0.32046+	**0.041235++**	**0.012691++**	**9.54e-07++**
MG	**0.000354++**	**0.004416++**	**0.000169++**	**0.000137++**
MAUC	**0.037155++**	0.2683+	**0.028519++**	**9.5e-07++**

table, it can be seen that OREMBoost significantly outperforms the other compared methods in almost all the cases, which demonstrates that OREMBoost is also a competitive ensemble method for combatting multiclass imbalance problems.

In the supplementary material, available online, we provided an in-depth analysis, from the point of view of accuracy and diversity, why OREMBoost can achieve better performance compared to the selected ensemble approaches. The empirical studies demonstrate that the advantage of OREMBoost comes from the substantial improvement of base classifiers in the average prediction accuracy of the minority classes without causing serious damage to the performance of the majority classes. It suggests that the training data processed by OREM is more conducive to modeling the minority class regions accurately in most iterations. The specific analysis is presented in Section S7 of the supplementary material, available online.

5 Conclusion

The interpolation oversampling is one of the most popular types of oversampling. The key issue is the identification of the minority class regions. The proposed OREM consists of two stages to locate the minority regions, i.e., finding the candidate minority regions in the vicinity of original minority samples (stage 1), and identifying the clean subregions within the candidate minority regions (stage 2). The stage 1 can frame the regions having high likelihood of appearing the minority samples. The stage 2 can further exclude those disputed subregions. The experimental studies demonstrate that OREM is often significantly better than state-of-the-art two-class oversampling algorithms, and the united way of the stages 1 and 2 can obtain the most robust performance when compared to several variants derived from the modifications of the stage 1 or stage 2.

However, OREM cannot combat more challenging multiclass imbalance problems. Motivated by this, we designed a dedicated multiclass oversampling algorithm OREM-M. In OREM-M, the synthetic samples are generated iteratively, and only the synthetic sample whose nearest neighbor is not from the other minority classes is accepted. In this way, OREM-M substantially alleviates the problem of class overlapping occurred among the samples of different minority classes. The experimental results show that OREM-M often statistically outperforms the existing multiclass oversampling methods.

Finally, to develop the positive synergy between OREM and boosting, we proposed a new ensemble approach addressing class imbalance problems, OREMBoost. We evaluated the performance of OREMBoost and some prevailing ensemble solutions on the two-class and multiclass imbalanced datasets, respectively. OREMBoost can often achieve significantly better performance compared to the others.

Several research issues associated with this work deserve further consideration. First, future studies could address the problem of determining the optimal oversampling degree at both class level and sample level. OREM and OREM-M require a preset total number of synthetic samples, and generate roughly equal synthetic samples for each original minority sample. However, the optimal oversampling degree for a minority class generally depends on the specific data at hand, and different minority samples would have different levels of importance. Second, it can be investigated how to make full use of the majority samples to help identify the minority class areas. OREM mainly utilizes the local information of the minority class to locate the potential minority regions, however, when the minority samples are extremely scarce, both the local and global information in the minority data might not be reliable. Third, the lightweight oversampling algorithms exploiting the whole data can be designed to handle large-scale imbalanced data.

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] S. S. Mullick, S. Datta, and S. Das, "Adaptive learning-based k-nearest neighbor classifiers with resilience to class imbalance," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5713–5725, Nov. 2018.
[3] S. S. Mullick, S. Datta, and S. Das, "Generative adversarial minority oversampling," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1695–1704.
[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Mach. Learn. Res.*, vol. 16, pp. 321–357, 2002.
[5] T. Zhu, Y. Lin, and Y. Liu, "Synthetic minority oversampling technique for multiclass imbalance problems," *Pattern Recognit.*, vol. 72, pp. 327–340, 2017.
[6] B. Krawczyk, M. Koziarski, and M. Woźniak, "Radial-based oversampling for multiclass imbalanced data classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 8, pp. 2818–2831, Aug. 2020.
[7] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," *IEEE Trans. Syst., Man, Cybern.-A, Syst. Hum.*, vol. 40, no. 1, pp. 185–197, Jan. 2010.
[8] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Trans. Syst., Man, Cybern., B (Cybern.)*, vol. 39, no. 2, pp. 539–550, Apr. 2009.
[9] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "SMOTEBoost: Improving prediction of the minority class in boosting," in *Proc. Eur. Conf. Princ. Data Mining Knowl. Discov.*, 2003, pp. 107–119.
[10] C.-L. Liu and P.-Y. Hsieh, "Model-based synthetic sampling for imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 8, pp. 1543–1556, Aug. 2020.
[11] G. Kovács, "An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets," *Appl. Soft Comput.*, vol. 83, 2019, Art. no. 105662.
[12] S. Maldonado, C. Vairetti, A. Fernandez, and F. Herrera, "FW-SMOTE: A feature-weighted oversampling approach for imbalanced classification," *Pattern Recognit.*, vol. 124, 2022, Art. no. 108511.
[13] S. Barua, M. M. Islam, X. Yao, and K. Murase, "MWMOTE-majority weighted minority oversampling technique for imbalanced data set learning," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 2, pp. 405–425, Feb. 2014.
[14] Z. Xu, D. Shen, T. Nie, Y. Kou, N. Yin, and X. Han, "A cluster-based oversampling algorithm combining smote and k-means for imbalanced medical data," *Inf. Sci.*, vol. 572, pp. 574–589, 2021.

[15] Y. Xie, M. Qiu, H. Zhang, L. Peng, and Z. Chen, “Gaussian distribution based oversampling for imbalanced data classification,” *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 2, pp. 667–679, Feb. 2022.

[16] X. Wang, J. Xu, T. Zeng, and L. Jing, “Local distribution-based adaptive minority oversampling for imbalanced data classification,” *Neurocomputing*, vol. 422, pp. 200–213, 2021.

[17] T. Zhu, Y. Lin, and Y. Liu, “Improving interpolation-based oversampling for imbalanced data learning,” *Knowl.-Based Syst.*, vol. 187, 2020, Art. no. 104826.

[18] S. Wang and X. Yao, “Multiclass imbalance problems: Analysis and potential solutions,” *IEEE Trans. Syst., Man, Cybern., B (Cybern.)*, vol. 42, no. 4, pp. 1119–1130, Aug. 2012.

[19] 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.

[20] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, “A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches,” *IEEE Trans. Syst., Man, Cybern., C. (Appl. Rev.)*, vol. 42, no. 4, pp. 463–484, Jul. 2012.

[21] J. Mathew, M. Luo, C. K. Pang, and H. L. Chan, “Kernel-based SMOTE for SVM classification of imbalanced datasets,” in *Proc. 41st Annu. Conf. IEEE Ind. Electron. Soc.*, 2015, pp. 001127–001132.

[22] A. Pourhabib, B. K. Mallick, and Y. Ding, “Absent data generating classifier for imbalanced class sizes,” *J. Mach. Learn. Res.*, vol. 16, pp. 2695–2724, 2015.

[23] H. Cao, X.-L. Li, D. Y.-K. Woon, and S.-K. Ng, “Integrated oversampling for imbalanced time series classification,” *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 12, pp. 2809–2822, Dec. 2013.

[24] L. Yang, Y. Guo, and J. Cheng, “Manifold distance-based oversampling technique for class imbalance learning,” in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 10 071–10 072.

[25] S. Bej, N. Davtyan, M. Wolfien, M. Nassar, and O. Wolkenhauer, “LoRAS: An oversampling approach for imbalanced datasets,” *Mach. Learn.*, vol. 110, no. 2, pp. 279–301, 2021.

[26] 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.

[27] S. K. Roy, J. M. Haut, M. E. Paoletti, S. R. Dubey, and A. Plaza, “Generative adversarial minority oversampling for spectral-spatial hyperspectral image classification,” *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–15, 2021.

[28] C. Bellinger, S. Sharma, N. Japkowicz, and O. R. Zaïane, “Framework for extreme imbalance classification: Swim-sampling with the majority class,” *Knowl. Inf. Syst.*, vol. 62, pp. 841–866, 2020.

[29] M. Liu, M. Dong, and C. Jing, “A modified real-value negative selection detector-based oversampling approach for multiclass imbalance problems,” *Inf. Sci.*, vol. 556, pp. 160–176, 2021.

[30] K. Napierala and J. Stefanowski, “Types of minority class examples and their influence on learning classifiers from imbalanced data,” *J. Intell. Inf. Syst.*, vol. 46, no. 3, pp. 563–597, 2016.

[31] T. Zhu, Y. Lin, Y. Liu, W. Zhang, and J. Zhang, “Minority oversampling for imbalanced ordinal regression,” *Knowl.-Based Syst.*, vol. 166, pp. 140–155, 2019.

[32] J. F. Díez-Pastor, J. J. Rodríguez, C. Garcia-Osorio, and L. I. Kuncheva, “Random balance: Ensembles of variable priors classifiers for imbalanced data,” *Knowl.-Based Syst.*, vol. 85, pp. 96–111, 2015.

[33] M. Lichman *et al.*, “UCI machine learning repository,” 2013.

[34] M. Koziarski, B. Krawczyk, and M. Woźniak, “Radial-based oversampling for noisy imbalanced data classification,” *Neurocomputing*, vol. 343, pp. 19–33, 2019.

[35] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical Machine Learning Tools and Techniques*. Burlington, MA, USA: Morgan Kaufmann, 2016.

[36] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, 2011, Art. no. 27.

[37] T. Fawcett, “ROC graphs: Notes and practical considerations for researchers,” *Mach. Learn.*, vol. 31, no. 1, pp. 1–38, 2004.

[38] A. Benavoli, G. Corani, and F. Mangili, “Should we really use post-hoc tests based on mean-ranks?,” *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 152–161, 2016.

[39] W. Chu and Z. Ghahramani, “Gaussian processes for ordinal regression,” *J. Mach. Learn. Res.*, vol. 6, pp. 1019–1041, 2005.

[40] Y. Sun, M. S. Kamel, and Y. Wang, “Boosting for learning multiple classes with imbalanced class distribution,” in *Proc. Int. Conf. Data Mining*, 2006, pp. 592–602.

[41] D. J. Hand and R. J. Till, “A simple generalisation of the area under the ROC curve for multiple class classification problems,” *Mach. Learn.*, vol. 45, no. 2, pp. 171–186, 2001.

[42] M. Pérez-Ortiz, P. A. Gutiérrez, C. Hervás-Martínez, and X. Yao, “Graph-based approaches for over-sampling in the context of ordinal regression,” *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 5, pp. 1233–1245, May 2015.

[43] Y. Freund and R. E. Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting,” *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.

[44] Z. Sun, Q. Song, X. Zhu, H. Sun, B. Xu, and Y. Zhou, “A novel ensemble method for classifying imbalanced data,” *Pattern Recognit.*, vol. 48, no. 5, pp. 1623–1637, 2015.

[45] Z. Liu *et al.*, “Self-paced ensemble for highly imbalanced massive data classification,” in *Proc. IEEE 36th Int. Conf. Data Eng.*, 2020, pp. 841–852.

[46] S. Datta, S. Nag, and S. Das, “Boosting with lexicographic programming: Addressing class imbalance without cost tuning,” *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 5, pp. 883–897, May 2020.

[47] Z. Chen, J. Duan, L. Kang, and G. Qiu, “A hybrid data-level ensemble to enable learning from highly imbalanced dataset,” *Inf. Sci.*, vol. 554, pp. 157–176, 2021.

Tuanfei Zhu received the PhD degree from Hunan University, China. He is now lecturer with the College of Computer Engineering and Applied Mathematics, Changsha University, and also a postdoctor with the School of Computer, National University of Defense Technology. He has published more than ten peer-reviewed articles. His current research interests include imbalanced data learning, ensemble learning.

Xinwang Liu (Senior Member, IEEE) received the PhD degree from the National University of Defense Technology (NUDT), China. He is now professor with the School of Computer, National University of Defense Technology. His current research interests include kernel learning and unsupervised feature learning. He has published more than 60 peer-reviewed papers, including those in highly regarded journals and conferences such as *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Image Processing*, *IEEE Transactions on Neural Networks and Learning Systems*, *IEEE Transactions on Multimedia*, *IEEE Transactions on Information Forensics and Security*, ICML, NeurIPS, CVPR, ICCV, AAAI, IJCAI, etc. More information can be found at https://xinwangliu.github.io/.

En Zhu received the PhD degree from the National University of Defense Technology (NUDT), China. He is now professor with the School of Computer Science, National University of Defense Technology, China. His main research interests are pattern recognition, image processing, machine vision and machine learning. He has published more than 60 peer-reviewed papers, including *IEEE Transactions on Circuits and Systems for Video Technology*, *IEEE Transactions on Neural Networks and Learning Systems*, *Pattern Recognition*, AAAI, IJCAI, etc. He was awarded China National Excellence Doctoral Dissertation.