AHE: Adaptive hybrid-sampling ensemble for large-scale highly imbalanced data classification

Xingjian Zeng[a], Yali Yuan[a,*], Hantao Mei[a], Guang Cheng[a]

[a]*School of Cyber Science and Engineering, Southeast University, Nanjing, 210000, China*

Abstract

Imbalanced data commonly exist in research fields and bias classifiers toward the class with a dominating sample size, leading to significant degradation in classification performance. Various data-level and ensemble-based methods have demonstrated notable success on small-scale imbalanced datasets with moderate imbalance ratios. However, they frequently encounter challenges like noise interference and information loss when dealing with large-scale and highly imbalanced data. To overcome these limitations, we introduce **AHE**, an **A**daptive **H**ybrid-sampling **E**nsemble in this paper for large-scale highly imbalanced data classification. Aiming at enhancing the scarce representation of the minority class with minimal noise introduction while mitigating information loss from the majority class, a novel hybrid-sampling pipeline is designed to construct informative training sets for base classifiers, which over-samples the minority class for partial augmentation and under-samples the majority class to achieve class balance. In the over-sampling phase, AHE determines an appropriate sampling amount based on within-class imbalance of the minority class and employs an improved SMOTE method for sample generation. In the under-sampling phase, AHE updates a dynamic histogram based on classification hardness to segment the majority class into diversified sampling subspaces and uniformly samples training data from each bin. Finally, base classifiers are combined in a boosting-like paradigm to form an ensemble classification model. Experimental results on 24 benchmark imbalanced datasets demonstrate that AHE achieves state-of-the-art performance over four evaluation metrics compared with 17 competitive baselines.

Keywords:
Imbalanced learning, Hybrid sampling, Ensemble learning, Large-scale data

*Corresponding author

Email addresses: zengxingjian@seu.edu.cn (Xingjian Zeng), yaliyuan@seu.edu.cn (Yali Yuan), meihantao@seu.edu.cn (Hantao Mei), Chengguang@seu.edu.cn (Guang Cheng)

1. Introduction

Imbalanced data commonly appears in research domains such as bot detection [1], intelligent fault prediction [2] and medical diagnosis [3], characterized by uneven representations of different classes. Due to the accuracy-oriented design, classifiers tend to overemphasize the majority class (i.e., the class with dominating sample size) [4] when trained directly on imbalanced datasets. This bias undermines their ability to accurately predict the minority class (i.e., the class with fewer samples), which often carries concepts of greater interest [5]. Furthermore, real-world imbalanced datasets are coupled with additional complexities [6, 7, 8], such as high imbalance ratios, large-scale sample sizes, and severe class overlap, all of these collectively pose significant challenges to imbalanced classification.

To address the imbalance issue, data-level methods are widely adopted as effective strategies. Over-sampling methods [9, 10, 11, 12, 13] augment the minority class to match the size of the majority class via sample generation. Under-sampling methods [14, 15, 16, 17, 18, 19, 20, 21] remove excess samples from the majority class to reduce class imbalance. However, these re-sampling methods possess respective limitations, as noisy samples can be introduced to the minority class [22, 23] and information loss from the majority class inevitably occurs [5, 24, 25, 26]. Considering these pros and cons, hybrid-sampling methods [27, 28, 29, 30, 31, 32] emerge as an balanced solution to leverage strengths while mitigating weaknesses, which first generates synthetic samples through over-sampling to achieve class balance and then removes noisy samples through under-sampling to clean the dataset.

Nevertheless, conventional hybrid-sampling methods exhibit inadequate adaptability and limited performance when applied to large-scale and highly imbalanced datasets. This limitation stems from the adherence to the 'balancing-then-cleaning' re-sampling pipeline, which relies on sample generation to fill the huge sample quantity gap. When over-sampling a dataset with a high imbalance ratio and substantial sample size for class balance, a large number of synthetic samples are generated and can easily distort the data patterns in the minority class, rendering the under-sampling-based noise reduction ineffective. As a result, the hybrid-sampling process almost degrades to its over-sampling component and suffers from noise interference.

Recently, under-sampling-based ensemble methods [33, 24, 25, 26, 34, 35] have garnered attention due to their efficient design and remarkable performance. Despite using only a small majority class subset for training, severe information loss is reduced to some extent as base classifiers learn from various aspects of the majority class and complement each other's limited perspective. However, these methods primarily

concentrate on under-sampling the majority class and overlook the issue of sample scarcity in the minority class. When handling large-scale highly imbalanced datasets, the limited number of minority class samples restricts the size of training sets and constrains the classification performance of base classifiers. This suggests that the learning capacity of the overall under-sampling-based ensemble model can be further elevated by integrating appropriate augmentation for the minority class.

To overcome the aforementioned limitations, in this paper we propose **AHE**, an **A**daptive **H**ybrid-sampling **E**nsemble for large-scale highly imbalanced data classification. The key innovation of AHE lies in redefining the roles of the two components in conventional hybrid-sampling to harness their strengths while mitigating their weaknesses through the integration of an ensemble framework. Instead of expanding the minority class to achieve class balance, AHE over-samples the minority class to a partial size of the majority class, generating an appropriate number of synthetic samples while limiting noise introduction caused by substantial sample generation. Thereafter, AHE under-samples the majority class to achieve class balance with diversified training samples. By incorporating this 'augmenting-then-balancing' pipeline into a boosting-like paradigm, AHE constructs informative training sets and effectively reduces information loss by aggregating multiple base classifiers.

To be specific, in the over-sampling phase, AHE estimates the skewed sub-concept distribution within the minority class through clustering and calculates the number of synthetic samples required to eliminate within-class imbalance. While conventional hybrid-sampling methods [29, 27, 28, 30, 31, 32] fully augment the minority class with a sampling rate of one, AHE adopts a notably lower partial sampling rate, thereby simultaneously expanding the minority class while mitigating noise interference from the root. To maintain a moderate synthesizing pace, a scheduler function is utilized to specify the sampling amount for each iteration. Finally, an improved SMOTE [9] method is designed to augment the minority class and enhance class boundary with rigorously controlled sample generation. In the under-sampling phase, AHE updates a dynamic hardness histogram to capture the classification hardness distribution and segments the majority class samples into multiple bins. Unlike previous under-sampling-based ensemble methods that necessitate a hyper-parameter [24, 34] or adopts an extreme value [25, 26] for the bin count, AHE formulates the segmentation of the majority class samples as a one-dimensional clustering task and minimizes within-bin hardness variance using an adapted Jenks natural breaks method [36] while ensuring a low number of bins. The resulting hardness histogram achieves the finest-grained segmentation of the majority class with the minimum bin count, thus serving as diversified sampling subspaces. Next, AHE extracts a majority class subset equal in size to the augmented minority class by uniformly sampling from each

bin, which guarantees sample diversity across various classification hardness levels.

To conclude, the contributions of this paper are as follows:

- We propose **AHE**, an **A**daptive **H**ybrid-sampling **E**nsemble framework with a novel hybrid-sampling pipeline for large-scale highly imbalanced data classification.
- We propose a partial over-sampling strategy based on an improved SMOTE method to eliminate the within-class imbalance of the minority class, achieving effective data augmentation while reducing noise introduction from the root.
- We introduce a dynamic hardness histogram to adaptively capture the classification hardness distribution of the majority class, based on which a uniform under-sampling strategy is proposed to select diversified training samples.
- We conduct extensive experiments to demonstrate the competence of AHE. Experimental results show that AHE achieves state-of-the-art performance compared with 17 baselines over four evaluation metrics while being adaptive to varying learning complexities of imbalanced data and generalized to various base classifiers.

The remainder of the paper is structured as follows: Section 2 provides an overview of related work in imbalanced classification. Section 3 introduces the proposed AHE in detail. Section 4 presents a comprehensive experimental evaluation. Finally, Section 5 concludes the paper.

2. Related work

Imbalanced learning methods can be categorized into **data-level**, **algorithm-level** and **ensemble-based**. Since algorithm-level methods alleviate classifier bias towards the imbalanced data by modifying existing learning algorithms and are out of the scope of this paper, the data-level and ensemble-based methods are reviewed in this section.

2.1. Data-level methods

Data-level methods mitigate class imbalance by resizing the original dataset and can be divided into three types, i.e., over-sampling, under-sampling and hybrid-sampling, based on the class targeted for re-sampling.

Over-sampling methods conduct sample generation in the minority class to balance the sample counts between classes. SMOTE [9] is one of the most influential

over-sampling techniques and drives the development of numerous over-sampling algorithms based on it [37, 38]. SMOTE performs linear interpolation in the feature space to generate synthetic samples around the sampling center sample based on its nearest neighbors. However, the neglect of specific data distributions and class boundaries makes it prone to noise introduction [4, 7, 38]. To amend this, various SMOTE-based variants seek to optimize sample generation based on neighbor distribution and borderline information [10, 11, 12, 13], achieving impressive performance on imbalanced classification tasks. Nevertheless, these methods suffer from significant computational costs when tackling large-scale highly imbalanced datasets and potentially introduce noise with the substantial sample generation.

Under-sampling methods remove excessive samples or selects a subset from the majority class to match the size of minority class using various heuristics. The simplest yet effective heuristic identifies noisy samples by analyzing the class distribution of their nearest neighbors. ENN [14] identifies samples whose nearest neighbors belong to different classes as noise and removes them. TomekLink [15] removes noisy samples by eliminating 'tomek links', i.e., pairs of nearest neighbors that belong to different classes. OSS [16] utilizes TomekLink to iteratively identify 'tomek links' and refine the imbalanced data. Additionally, clustering-based methods [19, 17, 18] determine the structure of majority class based on cluster priors, allowing for the effective sample selection with data pattern preservation. Evolution-based methods [20, 21] utilize evolutionary algorithms to guide the selection of the majority class samples. Nevertheless, information loss from the majority class inevitably occurs when simply under-sampling massive highly imbalanced data to achieve class balance, due to the sample scarcity in the minority class.

Hybrid-sampling methods combine both aforementioned sampling techniques. The conventional pipeline for hybrid-sampling can be summarized as 'balancing-then-cleaning', which relies on under-sampling to clean noisy samples introduced by sample generation. SMOTEENN [27] and SMOTETomek [28] respectively utilize ENN [14] and TomekLink [15] to filter noise in the over-sampled dataset. SMOTE-RSB [32] draws from rough-set theory and removes synthetic samples that show excessive similarity to the majority class samples based on certain thresholds. SMOTE-IPF [30] trains auxiliary classifiers on the over-sampled dataset and leans on their prediction outputs to identify potential noisy samples. SMOTERkNN [31] leverages reverse k-nearest neighbors [39] to calculate class probability density and filter noise based on density thresholds. However, these methods prove inadequate for large-scale highly imbalanced data classification as the overwhelming synthetic samples can severely distort the original data patterns, rendering under-sampling-based noise reduction ineffective.

2.2. Ensemble-based methods

Ensemble-based methods utilize re-sampling to construct balanced training sets for base classifiers and ensemble them as an overall classification model. SMOTEBoosting [40] and SMOTEBagging [41] balance training sets using SMOTE and aggregates base classifiers through boosting [42] or bagging [43], respectively. Both over-sampling-based ensemble techniques exhibit unacceptably substantial computational costs on large-scale highly imbalanced data [24], rendering them unsuitable for such a scenario [26, 25]. Consequently, under-sampling-based ensemble methods are more explored for datasets with a high imbalance ratio and large sample size.

Easy [33] is an efficient under-sampling-based ensemble method that constructs balanced training sets for a bag of classifier by randomly sampling from the majority class. To guide the under-sampling process with model feedback, Cascade [33] iteratively discards already well-learned training samples in the majority class to gradually shift classifiers' attention to harder-to-learn samples. To avoid overfitting on noisy samples and guarantee model-specific sampling strategy, SPE [24] divides the majority class samples into multiple bins w.r.t. classification hardness and self-paced selects training samples from each bin. EASE [25] adopts a similar binning strategy of SPE but requires abundant bins and a classifier weighting scheme to achieve stable classification performance. TSSE [26] refines the weighting approach in EASE and introduces a two-stage under-sampling process for mining borderline training samples. HUE [35] constructs hash subspaces in the majority class to select diversified training samples. ASE [34] employs an anomaly detector to monitor the classification hardness of the majority class samples and selects informative ones for training. In summary, these under-sampling-based ensemble methods bootstrap from the majority class to construct balanced training sets and leverage the aggregating capabilities of ensemble paradigm to alleviate information loss. Nevertheless, when dealing with large-scale highly imbalanced datasets, the size of the balanced training set is limited by the scarcity of the minority class samples, which constrains learning capacity of base classifiers and leads to suboptimal model performance.

3. Method

Fig. 1 provides an overview of the proposed Adaptive Hybrid-sampling Ensemble. AHE iteratively trains base classifiers and combines them in a serial, boosting-like paradigm to construct the final ensemble classification model. In each iteration, AHE transforms the original imbalanced dataset into a balanced training set through Adaptive Hybrid-sampling, which consists of partial over-sampling and uniform under-sampling. In the over-sampling phase, AHE estimates the skewed

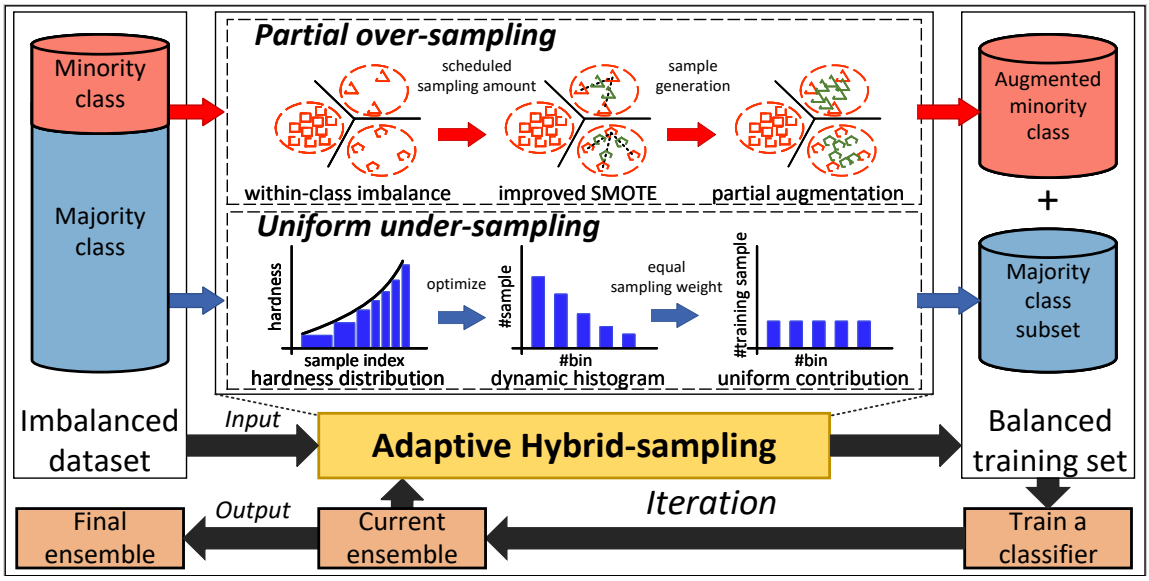


Figure 1: Overview of the proposed **A**daptive **H**ybrid-sampling **E**nsemble.

sub-concept distribution in the minority class through clustering and calculates the number of synthetic samples based on the size differences among sub-clusters. Thereafter, AHE utilizes a scheduler function to specify the sampling amount and guarantee a moderate sampling pace in each iteration. Finally, an improved SMOTE method is designed to augment the minority class and enhance the class boundary, which conducts cluster-wise sample generation with adaptively selected sampling center and bounded interpolation ratio. In the under-sampling phase, AHE initially sorts the majority class samples based on classification hardness and segments them into a dynamic hardness histogram. Specifically, an adapted Jenks natural breaks method [36], which returns a fitting score based on within-bin hardness variance for the histogram with a certain bin count value, is employed to evaluate the quality of the segmentation outcome. The optimal histogram in each iteration is updated by searching for the one with the best fitting score and the minimum bin count value. With the histogram, AHE uniformly samples from each fine-grained bin to obtain diversified majority class samples. A balanced and informative training set is formed with the augmented minority class and majority class subset, on which a base classifier is trained and then added to the current ensemble model. AHE effectively reduces noise interference from the root by rigorously generating an appropriate amount of synthetic samples, and alleviates information loss by training base classifiers on augmented, diversified training data and aggregating them together.

3.1. Partial over-sampling

In the partial over-sampling phase, AHE first calculates the total number of synthetic samples and utilizes a scheduler function to specify the sampling amount for the current iteration. Next, AHE conducts rigorously controlled sample generation with an improved SMOTE [9] method to augment the minority class.

To partially enrich the scarce representation of the minority class, AHE first calculates the number of synthetic samples based on the sample disparity among the sub-concepts. Sub-concepts with unequal sample sizes are commonly observed in imbalanced data, which constitutes within-class imbalance and further complicates the already challenging data distribution under class imbalance [44, 45]. Rare sub-concepts with very few samples are referred to as 'small disjuncts' and significantly contribute to the deterioration of classification performance [46]. Henceforth, to refine the within-class data distribution of the minority class and avoid generating an arbitrary number of synthetic samples, AHE determines the sampling amount required for balancing each sub-concept.

To be specific, AHE estimates the skewed sub-concept distribution within the minority class through clustering. Given an imbalanced dataset D, we denote the minority class as P. By applying a clustering algorithm $\mathcal{C}$, P is divided into multiple sub-clusters $P = \{SC_1, \ldots, SC_M\}$. The required sampling amount n for balancing each sub-concept is calculated as:

$$n = \sum_{i=1}^{M} n_i = \sum_{i=1}^{M} (\max(|SC_1|, \ldots, |SC_M|) - |SC_i|), \tag{1}$$

where $n_i = \max(|SC_1|, \ldots, |SC_M|) - |SC_i|$ accounts for the total number of synthetic samples to be generated in sub-clusters SC_i. Following this, AHE employs a scheduler function [47] $SF(t, T)$ to specify the sampling amount in each iteration based on n. $SF(t, T)$ monotonically returns values from 0 to 1 with the input t and the number of total iterations T. The amount of synthetic training samples in the minority class for the t-th iteration is:

$$n^t = \sum_{i=1}^{M} n_i^t = \sum_{i=1}^{M} SF(t, T) \times n_i, \tag{2}$$

where $n_i^t = SF(t, T) \times n_i$ denotes the synthetic sample amount of sub-cluster SC_i in the t-th iteration. Consequently, the sample quantity of each sub-concept gradually increases with each iteration from 0 to T, ultimately eliminating the within-class imbalance of the minority class. The specified sampling amount for the t-th iteration

Algorithm 1 SMOTE

Input: Imbalanced dataset D, number of nearest neighbors k
Output: Balanced dataset $D^{'}$

Initialize $N \leftarrow$ majority class of D, $P \leftarrow$ minority class of D.
while $|P| < |N|$ **do**
    Randomly select a seed sample x_{seed} from P.
    Randomly select a $x_{neighbor}$ of x_{seed} from its k-NN.
    Randomly sample $ratio$ from $\mathcal{U}[0, 1]$.
    $x_{synthetic} = x_{seed} + ratio \times (x_{seed} - x_{neighbor})$.
    $P \leftarrow P \cup \{x_{synthetic}\}$.
end while
return $D^{'} = P \cup N$

is calculated as:

$$\Delta n^t = \sum_{i=1}^{M} \Delta n_i^t = \sum_{i=1}^{M} n_i^t - n_i^{t-1}. \tag{3}$$

A constant sampling pace is considered for the minority class, and the scheduler function is defined as $SF(t, T) = t/T$. Since sub-concepts may have disparate sample quantities, balancing them all at once can pose challenges for the classifier to adapt to the original data distribution. Thus, the scheduler function is employed to prevent drastic sample generation and benefit the base classifiers from well adapting to the gradually augmented minority class.

With a determined sampling amount for each iteration, an improved SMOTE method [9] is designed for sample generation. The process of SMOTE can be described in Algorithm 1. SMOTE inherently possesses randomness in sample selection and interpolation, which makes it difficult to capture borderline information and adaptively augment the minority class with relatively limited partial sampling amount in AHE. To amend this, an improved SMOTE method is designed to conduct rigorously controlled sample generation for effectively enhancing the class boundary.

First, AHE generates synthetic samples cluster-wise to preserve the original data distribution within each sub-concept. When generating a new sample $x_{synthetic}$ in a sub-cluster with a seed sample x_{seed}, only $x_{neighbor}$ from the same cluster is eligible for interpolation. Thus $x_{synthetic}$ remains within the geometric space of that cluster.

Second, instead of randomly bootstrapping samples as x_{seed} for interpolation, AHE specifies the sampling amount of x_{seed} from different local hardness levels, which is calculated by the k-Disagreeing Neighbor. For a sample (x, y), its local

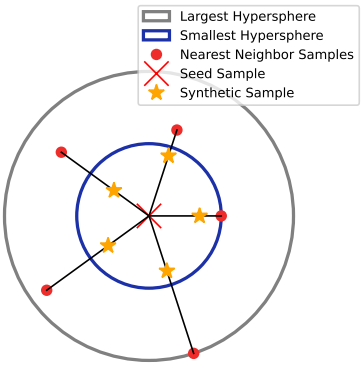


Figure 2: Illustration of the bounded *ratio* for linear interpolation. Whichever neighbor sample is selected for interpolation, the synthetic sample is generated within the smallest hypersphere formed by the sampling center and its nearest neighbor.

hardness value $h^l(x)$ is calculated as

$$h^l(x) = \frac{|x^{'}|x^{'} \in kNN(x) \wedge label(x^{'}) \neq y|}{k}, \tag{4}$$

where k is the number of nearest neighbor. Each sub-cluster is divided into various bins w.r.t. the local hardness level:

$$B_\upsilon = \{(x, y)|h^l(x) = \frac{\upsilon}{k}\}, \forall \upsilon = 0, \ldots, k. \tag{5}$$

A larger local hardness value indicates that the minority class sample has more nearest neighbors from the majority class and is more likely to lie on the class boundary, which is crucial for the classifier to learn a correct class distribution. Thus, AHE prefers to select samples with greater local hardness values as x_{seed} and generates samples locally around them to enhance the class boundary. Given the sampling amount n_i^t for sub-cluster SC_i in the t-th iteration, AHE normalizes sampling weight $w_\upsilon = h_\upsilon^l / \sum_{j=1}^k h_j^k$ for the υ-th bin and bootstraps x_{seed} with an amount of $\Delta n_i^t \times w_\upsilon$ times for sample generation.

Third, AHE limits the region of sample generation by setting an upper bound for *ratio*, ensuring that the synthetic samples are generated locally around the seed samples. To enhance the class boundary near the sampling center x_{seed} through sample generation, setting an upper bound of 1 for *ratio* might result in the generated sample $x_{synthetic}$ deviating significantly from x_{seed}, thus undermining the effectiveness

of data augmentation. To address this, the upper bound for *ratio* is adaptively determined based on the spatial distribution of x_{seed} and its k-nearest neighbors. For a given x_{seed}, AHE randomly selects an $x_{neighbor}$ from its k-nearest neighbor and samples *ratio* from $\mathcal{U}[0, upper\_bound]$ where *upper_bound* is calculated by:

$$upper\_bound = \frac{\min(dist(x_{seed}, x_{neighbor\_\ell}) \mid \forall \ell = 1, \ldots, k)}{dist(x_{seed}, x_{neighbor})}. \tag{6}$$

The synthetic sample $x_{synthetic}$ is generated with:

$$x_{synthetic} = x_{seed} + ratio \times (x_{seed} - x_{neighbor}). \tag{7}$$

Fig. 2 illustrates the impact of the bounded *ratio* on linear interpolation. As $\min(dist(x_{seed}, x_{neighbor\_\ell}))$ represents the radius of the smallest hypersphere encompassing x_{seed} (the blue circle) and its k-nearest neighbors (the red points), $x_{synthetic}$ is generated within this region regardless of which $x_{neighbor}$ is chosen, thus avoiding excessive deviation from x_{seed}. Henceforth, the generation of $x_{synthetic}$ is utilized to maximally enhance the class boundary.

The details of the partial over-sampling phase are presented in Algorithm 2. AHE iterates through each sub-cluster in the minority class and conducts cluster-wise sample generation (step 2). In practice, samples within certain sub-clusters have a local hardness value of 0, indicating that these sub-clusters are located distant from the class boundary. For these sub-clusters, the original SMOTE method is applied for cluster-wise sample generation (step 5-11). Otherwise, the improved SMOTE method is applied (steps 14-23).

3.2. Uniform under-sampling

In the uniform under-sampling phase, AHE calculates classification hardness for the majority class samples and segments them into a dynamic hardness histogram, which is updated by jointly optimizing bin count with a fitting score calculated by an adapted Jenks natural breaks method [36]. The dynamic hardness histogram ensures that samples within each bin exhibit similar levels of classification hardness while maintaining clear distinctions across hardness levels. This histogram design creates diversified under-sampling subspaces. AHE treats each hardness level equally and uniformly samples from each bin to construct a majority class subset that matches the size of the augmented minority class.

Despite AHE increases the sizes of the majority class subsets with the partial augmentation of the minority class, the total number of training samples remains constrained compared with the overall majority class samples, and potential information loss may occur. Therefore, selecting diversified and representative samples of

Algorithm 2 `Partial_over_sampling`

Input: Sub-clusters of the original minority class $SC = \{SC_1, \ldots, SC_M\}$, number of nearest neighbors k, current iteration t

Output: A set $P^{'}$ containing synthetic samples for partial augmentation

Initialize $P^{'} \leftarrow \emptyset$.
for $i = 1$ to M **do** ▷ Cluster-wise sample generation.
  $P_i^{'} \leftarrow \emptyset$.
  Compute the sampling amount Δn_i^t using Eq. (3).
  if $\forall h_\upsilon^l = 0$ for $\upsilon \in \{1, \ldots, k\}$ **then** ▷ Apply the original SMOTE.
    while $|P_i^{'}| < n_i^t$ **do**
      Randomly sample x_{seed} from SC_i.
      Randomly select $x_{neighbor}$ of x_{seed} from its k-NN in SC_i.
      Randomly sample $ratio$ from $\mathcal{U}[0, 1]$.
      Generate $x_{synthetic}$ using Eq. (7).
      $P_i^{'} \leftarrow P_i^{'} \cup \{x_{synthetic}\}$.
    end while
  else ▷ Apply the improved SMOTE.
    Divide SC_i into k bins w.r.t. local hardness h_υ^l using Eq. (4) and Eq. (5).
    Normalize sampling weights: $w_\upsilon = h_\upsilon^l / \sum_{j=1}^{k} h_j^l, \forall \upsilon = 1, \ldots, k$.
    for $\upsilon = 1$ to k **do**
      Bootstrap $\Delta n_i^t \cdot w_\upsilon$ seed samples x_{seed} from the υ-th bin.
      for each x_{seed} bootstrapped from the υ-th bin **do**
        Randomly select $x_{neighbor}$ of x_{seed} from its k-NN in SC_i.
        Compute $upper\_bound$ for $ratio$ using Eq. (6).
        Randomly sample $ratio$ from $\mathcal{U}[0, upper\_bound]$.
        Generate $x_{sythetic}$ using Eq. (7).
        $P_i^{'} \leftarrow P_i^{'} \cup \{x_{synthetic}\}$.
      end for
    end for
  end if
  $P^{'} \leftarrow P^{'} \cup P_i^{'}$.
end for
return $P^{'}$

the majority class is crucial for enabling the base classifiers to acquire a more comprehensive knowledge of the sample distribution from limited training data. To achieve this, AHE utilizes classification hardness [24, 26] to reflect the model's learning state

on the training samples and establishes a dynamic hardness histogram to adaptively characterize sample diversity in the majority class. Classification hardness is defined as $h^c = H(x, y, F(x))$, where H is a "decomposable" error function [24] and $F(x)$ denotes the classification probability of the ensemble model for sample (x, y). In AHE, the absolute error function is employed to calculate classification hardness:

$$h^c = |y - F(x)|. \tag{8}$$

The majority class samples are sorted and segmented into a histogram w.r.t. h^c:

$$B_q = \{(x, y)|h^c \in [\frac{q-1}{b}, \frac{q}{b})\}, \forall q = 1, \ldots, b, \tag{9}$$

where the bin count b is essential for constructing the histogram as it directly impacts the consistency of sample distribution within the same hardness level and the distinction between different hardness levels. Although existing under-sampling ensemble methods utilize similar binning strategies to construct sampling subspaces, they build a static hardness histogram with a fixed bin count, which is either determined by a manually set hyperparameters [24, 34] or simply the number of minority class samples [25, 26]. The static hardness histogram fails to adaptively reflect the classification hardness distribution due to predefined hardness levels. To amend this, AHE determines the optimal segmentation of the majority class based on the dynamic classification hardness distribution and updates the histogram in each iteration

The update of dynamic hardness histogram is essentially a one-dimensional clustering process, aiming to group samples with similar hardness values into the same bin while ensuring that different bins represent distinct hardness levels. The classic one-dimensional clustering method, Jenks natural breaks [36] is employed to cluster the majority class samples based on the classification hardness sequence. For a sorted hardness sequence $h^c = (h^c_1, \ldots, h^c_m)$, Jenks natural breaks calculates clustering fitting scores for all possible partitions $P(h^c, k)$ with k segments:

$$P(h^c, k) = \left\{h^c[i_{k-1} : i_k]|\ 1 \leq i_1 < \cdots < i_{k-1} < m\right\}. \tag{10}$$

For the hardness sequence h^c, the "sum of squared deviations for array mean" $(SDAM)$ is calculated as:

$$SDAM = \sum_{i=1}^{m} (h^c_i - \bar{h^c})^2. \tag{11}$$

The "sum of squared deviations for class means" $(SDCM\_ALL)$ for the partition

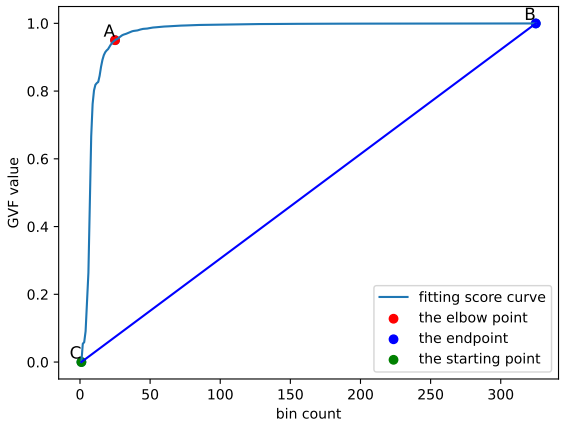


Figure 3: Fitting score graph with the red elbow point marked, where the optimal bin count is determined. The elbow point A is the point with the greatest distance from the line connecting the search endpoint B and the starting point C

$P(h^c, k)$ is calculated as:

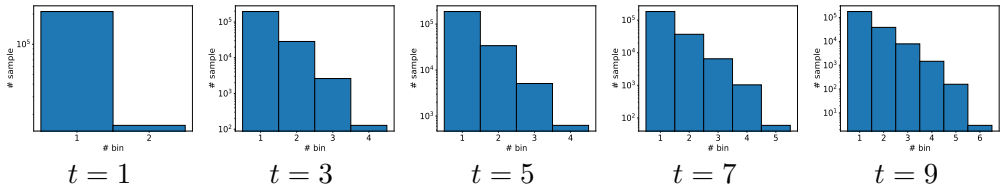
$$SDCM\_ALL = \sum_{i=1}^{k} \sum_{j=1}^{|s_i|} (s_i[j] - \bar{s}_i)^2, \tag{12}$$

where $s_i = h^c[i_{k-1} : i_k]$ denotes the i-th segments in $P(h^c, k)$. $SDCM\_ALL$ represents the overall within-bin hardness variance of h^c. With $SDCM$ and $SDCM\_ALL$, the fitting score "goodness of variance fit" (GVF) is calculated as:

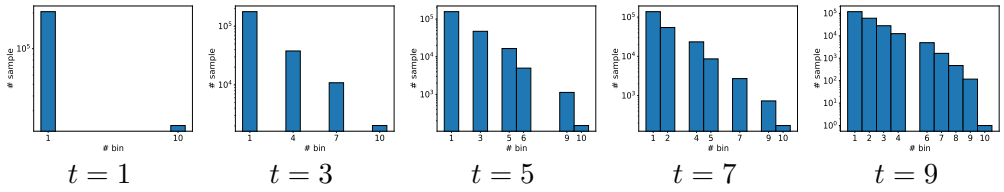
$$GVF = 1 - SDCM\_ALL/SDAM. \tag{13}$$

A higher GVF value indicates a smaller within-bin hardness variance and better partition quality of the input sequence. Since the dynamic hardness histogram is based on Eq. (9) with an equal bin size and a total length of 1, the searching space of candidate partition $P(h^c, k)$ is simplified by setting an equal length $1/k$ for each segment s_j. In other words, a candidate partition of the majority class $P(h^c, k)$ is obtained based on a certain bin count value of k and Eq. (9) with a corresponding GVF value for assessing the partition quality. To ensure that at least one sample is drawn from each bin, the upper limit of the search range for the bin count is set to the sample size of the partially augmented minority class, $|P^{'}|$, which corresponds to the size of the majority class subset to be sampled.

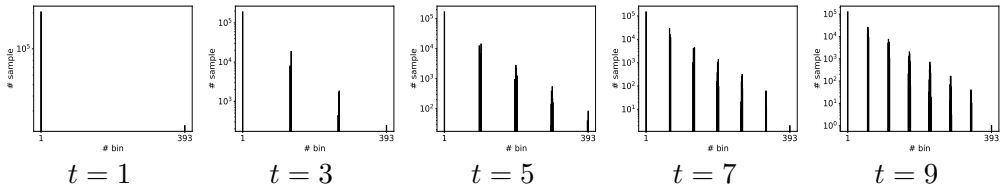
With this adapted Jenks natural breaks method, AHE computes a GVF value for each candidate bin count and establishes a fitness score graph (as shown in Fig. 3)



(a) Dynamic hardness histogram of AHE with adaptive bin count.



(b) Static hardness histogram of SPE with a fixed bin count of 10.



(c) Static hardness histogram of EASE with a fixed bin count of 393.

Figure 4: A comparison of hardness histogram updates in AHE, SPE [24] and EASE [25] on the Credit Card dataset. Each subfigure shows the hardness histograms constructed by AHE and two other methods over five iterations when $t = 1$, $t = 3$, $t = 5$, $t = 7$ and $t = 9$. The total number of training iterations is 10. Decision tree is used as the base classifier.

where the x-axis represents the bin count and the y-axis represents the fitting score GVF. On the one hand, using too few bins fails to achieve a fine-grained partition and results in a low fitting score with a large within-bin hardness variance, increasing the sampling randomness within each bin. On the other hand, using excessive bins is unnecessary and does not significantly improve partition granularity, which reduces the representativeness of hardness levels while increasing the probability of selecting noise samples. To strike a balance, AHE chooses the elbow point in the fitting score graph to determine the appropriate bin count for the dynamic hardness histogram, where adding another bin does not yield a greater improvement on the fitting score. Thus, the dynamic hardness histogram achieves the finest-grained partitioning of the majority class samples with the fewest bins, ensuring a consistent hardness distribution within each bin and distinct hardness levels across bins. Fig. 4 shows the updates of hardness histogram on a real-world large-scale highly imbal-

Algorithm 3 `Uniform_under_sampling`

Input: Majority class data N, size of the majority class subset m, classification hardness array h^c

Output: The majority class subset $N^{'}$

Initialize $SDCM\_ALL \leftarrow 0$, $GVF\_list \leftarrow []$.
Calculate $SDAM$ using Eq. (11).
for $p = 2$ to m **do** ▷ Iterate over candidate bin counts.
    Divide h^c into p segments $(s_1, \ldots, s_p)$ using Eq. (9).
    Calculate $SDCM\_ALL$ for $(s_1, \ldots, s_p)$ using Eq. (12).
    Calculate GVF for $(s_1, \ldots, s_p)$ using Eq. (13).
    Record fitting score: $GVF\_list \leftarrow GVF\_list \cup \{GVF\}$.
end for
Find the elbow point in GVF_list to determine the histogram bin count b.
Divide N into b bins $B_1, \ldots, B_b$ using Eq. (9).
Bootstrap a subset N_q with sample amount of m/b from $B_q, \forall \upsilon = 1, \ldots, b$.
return $N_1 \cup \ldots \cup N_b$

anced dataset [1] in AHE and two other under-sampling-based ensemble methods with binning strategies. SPE [24] requires a hyper-parameter for bin count (default 10) while EASE [25] sets the bin count equal to the size of the minority class, both of which lead to the occurrence of empty bins and overly similar hardness levels. In contrast, the dynamic hardness histogram in AHE offers a more effective and adaptive partitioning of the majority class samples.

Based on the dynamic hardness histogram, AHE uniformly selects samples from each histogram bin to construct a majority class subset for training. AHE treats each histogram bin equally without any further sophisticated weighting scheme, as the fine-grained histogram already serves as diversified sampling subspaces with similar within-bin hardness distribution and representative hardness levels. Each bin has an equal sample contribution to the under-sampling process, which maximizes the diversity of training samples from adaptively-divided hardness levels and facilitates the comprehensive training of base classifiers. The details of the uniform under-sampling phase are presented in Algorithm 3.

[1] Credit Card dataset, refer to Table 1 for detailed statistics.

Algorithm 4 Adaptive Hybrid-sampling Ensemble

Input: Imbalanced dataset D, number of nearest neighbors k, number of total iterations T, base classifier f, clustering algorithm $\mathcal{C}$

Output: Ensemble classifier F

1: **Initialize:** $N \leftarrow$ majority class of D, $P \leftarrow$ minority class of D, $P_0 \leftarrow P$.
2: Divide P into sub-clusters $SC = \{SC_1, \ldots, SC_M\}$ using $\mathcal{C}$.
3: Train f_0 on $P_0 \cup N_0$, where N_0 is randomly selected from N.
4: **for** $t = 1$ to T **do**
5: Update the current ensemble: $F_t = \frac{1}{t}\sum_{j=0}^{t-1} f_j$.
6: Compute the classification hardness array h^c using Eq. (9).
7: $P_t \leftarrow P_{t-1} \cup$ `Partial_over_sampling`(SC, k, t) ▷ Algorithm 2
8: $N_t \leftarrow$ `Uniform_under_sampling`$(N, |P_t|, h^c)$ ▷ Algorithm 3
9: Train f_t on $P_t \cup N_t$.
10: **end for**
11: **return** $F = \frac{1}{T}\sum_{j=1}^{T} f_j$

3.3. Adaptive Hybrid-sampling Ensemble

The overall process of the proposed AHE is presented in Algorithm 4. To begin with, a clustering algorithm $\mathcal{C}$ is applied to the minority class P to obtain multiple sub-clusters (step 2). The first base classifier is trained on the training set consisting of the original minority class and a randomly selected majority class subset (step 3). In each iteration, the ensemble model is updated (step 5) to calculate the classification hardness distribution for the majority class (step 6). Thereafter, the Adaptive Hybrid-sampling, which consists of partial over-sampling (step 7) and uniform under-sampling (step 8), constructs a balanced training set for the base classifier. Since the scheduler function specifies the synthetic sample amount for each iteration in the partial over-sampling phase, the number of minority class training samples gradually increases from iteration 1 to T, thereby expanding the training set size per iteration and enabling the ensemble model to learn from more comprehensive data. The new classifier is trained on a balanced, informative training set combined with the partially augmented minority class and a diversified majority class subset (step 9).

It should be noted that, though there are many clustering algorithm candidates for $\mathcal{C}$, AHE employs the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [48] to cluster the minority class samples. DBSCAN is a density-based clustering algorithm capable of identifying arbitrarily shaped clusters and is robust to outliers. Hence, it is suitable for estimating the sub-concept distribution within

the minority class, with an automatically determined number of sub-clusters. The required parameters (ϵ and *minPts*) for the DBSCAN algorithm are empirically chosen based on the k-distance graph [49, 50].

4. Experiment

4.1. Setup details

4.1.1. Experimental datasets

For imbalanced datasets, the majority and minority classes are denoted as N and P, respectively. The imbalance ratio (IR) is used to reflect the sample disparity in N and P, calculated as:

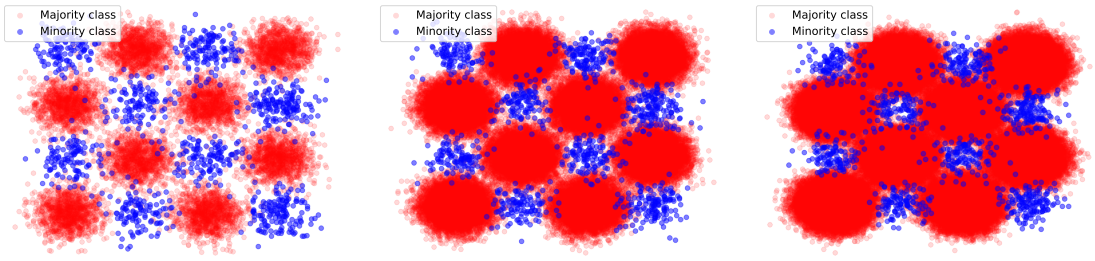
$$IR = \frac{|N|}{|P|} \tag{14}$$

Higher IR values indicate that the dataset is more imbalanced. The experimental datasets are categorized into three groups with different characteristics and evaluation objectives: (1) synthetic datasets with increasing IR values and sample sizes, (2) real-world datasets with high IR values and large sample sizes, and (3) real-world datasets with moderate IR values and sample sizes.

To intuitively validate AHE and highlight the inherent limitations of conventional hybrid-sampling methods, we construct two series of synthetic checkerboard datasets following the approach in [24]. Each dataset consists of 16 Gaussian components, all sharing the same covariance matrix of $cov \times I_2$, where *cov* indicates the degree of overlap between the Gaussian clusters. Initially, we fix $|P| = 1000$ and $cov = 0.5$, then gradually increase IR value from 10 to 300 with a step size of 1. This results in a total of 290 synthetic datasets with increasing IR and sample size, as illustrated in Fig 5. Subsequently, we fix $IR = 100$ and $|P| = 1000$, then gradually increase *cov* from 0.25 to 0.75 in increments of 0.01. This generates a total of 50 synthetic datasets with progressively increasing class overlap degree, as illustrated in Fig 6.

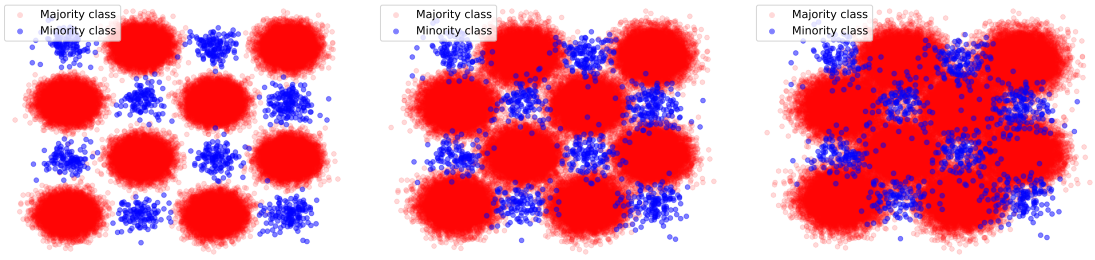
To evaluate the performance of AHE on large-scale, highly imbalanced datasets, we select four benchmark datasets characterized by high IR values and large sample size. These datasets have been widely used in prior studies [24, 25, 26] and serve as the first category for the main experimental evaluation. The details of these large-scale, highly imbalanced datasets are summarized in Table 1. The Normal vs R2L and Dos vs R2L datasets originate from the KDD Cup 1999 Data Mining Competition (KDDCUP-99)[2]. In these datasets, the R2L (Remote-to-Local) attack samples are

[2]https://kdd.ics.uci.edu/databases/kddcup99



(a) IR=10 (b) IR=100 (c) IR=300

Figure 5: Illustration of synthetic checkerboard datasets with varying IR and sample size. Each dataset contains 16 Gaussian components sharing the same covariance matrix of $0.5 \times I_2$. The number of minority class samples is fixed as 1000. Blue/red dots denote the minority/majority class samples.



(a) cov=0.25 (b) cov=0.5 (c) cov=0.75

Figure 6: Illustration of synthetic checkerboard datasets with varying cov. Each dataset contains 16 Gaussian components sharing the same covariance matrix of $cov \times I_2$. The number of minority class samples and IR are respectively fixed as 1000 and 100. Blue/red dots denote the minority/majority class samples.

treated as the minority class, while normal traffic samples and DoS (Denial of Service) attack samples are considered as the majority classes, respectively.

To demonstrate the applicability of AHE for smaller-scale imbalanced datasets, despite its design for the large-scale and highly skewed scenario, we select 20 imbalanced datasets with moderate IR values and sample size, sourced from the University of California, Irvine (UCI) repository[3], the KDD Data Mining Competition (KDD)[4] and the LIBSVM[5] [51]. Statistic details of these small-scale imbalanced datasets are

[3]https://archive.ics.uci.edu/datasets.
[4]https://kdd.org/kdd-cup
[5]https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/

Table 1: Details of datasets with high IR and large sample sizes, arranged in ascending order of IR.

Dataset	**IR**	**#Sample**	**#Feature**	**#Minority**	**#Majority**
Credit Fraud	577.88:1	284,807	31	492	284,314
Payment Simulation	773.70:1	6,362,620	11	8,213	6,354,407
Normal vs R2L	863.92:1	973,905	42	1,126	972,779
Dos vs R2L	3448.82:1	3,884,496	42	1,126	3,883,370

Table 2: Details of datasets with moderate IR and size, arranged in ascending order of IR.

Dataset	**IR**	**#Sample**	**#Feature**	**#Minority**	**#Majority**
optical_digits	9.1:1	5,620	64	1,296	144,455
satimage	9.3:1	6,435	36	260	10,923
sick_euthyroid	9.8:1	3,163	42	981	33,799
spectrometer	11:1	531	93	51	1,433
car_eval_34	12:1	1,728	21	734	19,266
us_crime	12:1	1,994	100	65	1,663
scene	13:1	2,407	294	68	1,321
libras_move	14:1	360	90	25	427
thyroid_sick	15:1	3,772	52	586	9,236
coil_2000	16:1	9,822	85	231	3,541
solar_flare_m0	19:1	1,389	32	177	2,230
oil	22:1	937	49	178	2,239
car_eval_4	26:1	1,728	21	150	1,844
wine_quality	26:1	4,898	11	600	7,197
letter_img	26:1	20,000	16	134	1,594
yeast_me2	28:1	1,484	8	45	486
ozone_level	34:1	2,536	72	1,055	9,937
mammography	42:1	11,183	6	626	5,809
protein_homo	111:1	145,751	74	554	5,066
abalone_19	130:1	4,177	10	35	301

summarized in Table 2.

For datasets containing categorical features (i.e., Normal vs R2L and Dos vs R2L), categorical variables are transformed into numerical features based on their frequencies. Additionally, no further preprocessing measures are applied. To evaluate the performance of different baselines combined with classifiers, each dataset is divided into non-overlapping training and testing sets in an 80%:20% ratio.

4.1.2. Evaluation Metrics

In imbalanced classification tasks, the accuracy metric is inadequate for evaluating model performance due to the disparity in sample sizes between majority and

Table 3: Confusion matrix for binary classification.

Label \ Predict	Positive	Negative
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)

minority classes in testing data. To address this, we adopt the F1, G-mean, MCC, and AUCPRC metric, which are derived from the confusion matrix (Table 3), to comprehensively evaluate the performance of AHE and baseline methods.

F1 and G-mean represent the harmonic mean and geometric mean of precision and recall, respectively. MCC (Matthews Correlation Coefficient) evaluates the model's performance across all categories of the confusion matrix, offering a unbiased measure even in the presence of class imbalance. AUCPRC (Area Under the Precision-Recall Curve) reflects the model's generalization ability to varying classification thresholds.

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

$$Recall = \frac{TP}{TP + FN} \tag{16}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{17}$$

$$G - mean = \sqrt{Precision \times Recall} \tag{18}$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{19}$$

4.1.3. Comparison baselines and classifiers

We select 17 state-of-the-art imbalanced learning baselines for comparison, as summarized in Table 4. These baselines are categorized into four groups: over-sampling, under-sampling, hybrid-sampling, and under-sampling-based ensemble. Among these, 11 belong to data-level methods. SMOTE (*Synthetic Minority Over-sampling TechniquE*) [9] generates synthetic samples through linear interpolation to balance the dataset. ADASYN (*ADAptive SYNthetic over-sampling*) [10] and BSMOTE (*BorderlineSMOTE*) [11] are two SMOTE variants that improve sample generation by incorporating local class distribution and borderline information, respectively. ENN (*Edited Nearest Neighbor*) [14] removes noisy majority class samples

Table 4: Summary of 17 baseline methods, categorized into over-sampling, under-sampling, hybrid-sampling, and under-sampling-based ensemble techniques.

Baselines	**Over-sampling**	**Under-sampling**	**Hybrid-sampling**	**Ensemble-based**
ADASYN [10]	✓	-	-	-
BSMOTE [11]	✓	-	-	-
SMOTE [9]	✓	-	-	-
ENN [14]	-	✓	-	-
TomekLink [15]	-	✓	-	-
OSS [16]	-	✓	-	-
SMOTEENN [27]	-	-	✓	-
SMOTETomek [28]	-	-	✓	-
SMOTEIPF [30]	-	-	✓	-
SMOTERkNN [31]	-	-	✓	-
DE-OS [29]	-	-	✓	-
Easy [33]	-	✓	-	✓
Cascade [33]	-	✓	-	✓
SPE [24]	-	✓	-	✓
EASE [25]	-	✓	-	✓
TSSE [26]	-	✓	-	✓
HUE [35]	-	✓	-	✓

whose labels differ from those of their nearest neighbors. TomekLink [15] identifies and eliminates tomek links which represent borderline instances between classes for noise reduction. OSS (*One Sided Selection*) [16], an extension of TomekLink, iteratively detects tomek links and removes noisy samples from the dataset. SMOTEENN [27] and SMOTETomek [28] combine SMOTE with ENN and TomekLink, respectively, to refine the balanced dataset produced by SMOTE. SMOTEIPF [30] partitions the dataset balanced by SMOTE into subsets, trains auxiliary classifiers, and removes noisy samples based on the classifiers' predictions. SMOTERkNN [31] utilizes reverse k-nearest neighbors [39] to estimate class probability densities for each sample and removes noise based on density thresholds. DE-OS (*Differential Evolution Over-sampling*) [29] generates samples and filters out noise based on the differential evolution algorithm.

For ensemble-based methods, over-sampling-based techniques (e.g., SMOTEBagging [41] and SMOTEBoosting [40]) are not considered as they exhibit substantial computational cost on large-scale, highly imbalanced datasets and are impractical for this classification scenario. Six under-sampling-based methods are considered. Easy [33] randomly selects training samples from the majority class to train an ensemble. Cascade [33] improves Easy by iteratively discarding well-learned samples, thereby encouraging base classifiers to focus on learning borderline samples. SPE (*Self-paced Ensemble*) [24] adopts a bin-division strategy to assess the learning difficulties of majority class samples and self-paced samples from each bin for under-sampling. EASE (*Equalization ensemble*) [25] and TSSE (*Two-step ensemble*

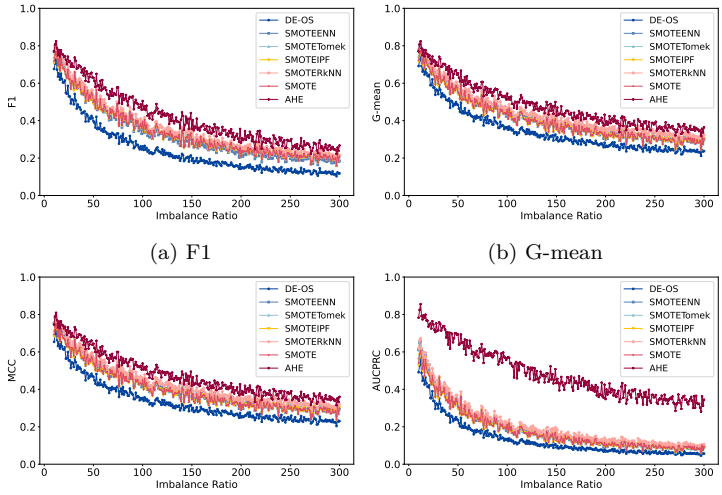
under-sampling) [26] are extensions of SPE, incorporating similar binning strategies along with respective weighting schemes for base classifiers. HUE (*Hashing-based under-sampling ensemble*) creates majority class subplaces using hash methods to ensure sampling diversity when constructing training sets.

To ensure consistency with prior research, we adopt DT (*Decision Tree*) as the primary classifier, and only employ SVM (*Support Vector Machine*), MLP (*Multi-Layer Perceptron*) and LR (*Logistic Regression*) in Section 4.5 to evaluate the generalization ability of AHE across different classifiers. All classifiers are implemented using the scikit-learn Python package [52]. The regularization parameter in SVM is set to 1000, the number of layers of MLP is set to 3, and the rest of the hyper-parameters of base classifiers are set to their default values. Note that the classification threshold for transforming predicted classification probability into label is set to 0.5. All baselines are implemented using the imbalanced-learn Python package [53] and the publicly available code from the research papers. The parameters of the baseline methods are set to their default values. As for AHE, the total number of iterations T is set to 10, which is the same as for the 6 under-sampling-based ensemble baselines. The number of nearest neighbors k is fine-tuned between 1 and 5. DBSCAN is used as the input clustering algorithm $\mathcal{C}$, and its required parameters (ϵ and *minPts*) are empirically determined based on the k-distance graph [50, 49]. To reduce the impact of randomness, the final results report the mean and standard deviation based on 10 independent runs. All experiments were conducted on a machine with an AMD EPYC 7282 processor (1.5GHz) and 377GB RAM.

4.2. Intuitive evaluation on synthetic datasets

We first evaluate the proposed AHE on synthetic datasets to demonstrate its adaptability to various levels of class imbalance and overlap. To separately investigate the impact of these two characteristics on classification results, we construct two series of 4×4 synthetic checkerboard datasets with increasing IR and *cov* values, respectively, to monitor two situations where with other conditions fixed, the data distribution deteriorates as the degree of class imbalance or class overlap intensifies. The setup details are provided in Section 4.1.1. To validate the effectiveness of the 'augmenting-then-balancing' strategy in AHE, five hybrid-sampling baselines with the conventional sampling pipeline and SMOTE are considered for comparison.

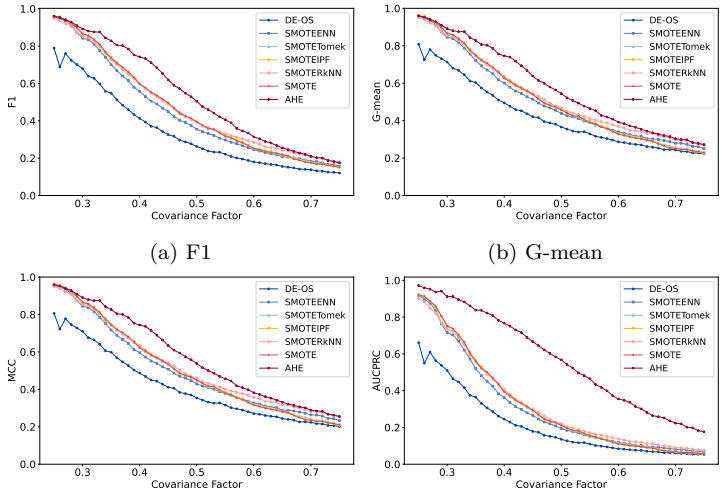
Fig. 7 and Fig. 8 present the evaluation results on synthetic datasets with increasing IR and *cov*, respectively. The vertical axis of the scatter plots represents the metric score, while the horizontal axis represents the evaluated characteristics of the synthetic datasets (class imbalance or class overlap). It can be observed that when the values of IR and *cov* are small, all methods achieve remarkable performance as



(a) F1 (b) G-mean

(c) MCC (d) AUCPRC

Figure 7: Evaluation results across four metrics on synthetic checkerboard datasets with $cov = 0.5$. IR varies from 10 to 300, with a step size of 1.



(a) F1 (b) G-mean

(c) MCC (d) AUCPRC

Figure 8: Evaluation results across four metrics on synthetic checkerboard datasets with $IR = 0.5$. cov varies from 0.25 to 0.75, with a step size of 0.01.

the representation of the majority class is less overwhelming and the class boundaries are clearer. As the values of IR and cov increase, the majority class overlaps more with the minority class and overwhelms it in terms of sample quantity (as illustrated

in Fig. 5 and Fig. 6), resulting in severe performance degradation.

Specifically, in the first series of synthetic datasets, the rising IR value leads to a rapid decline in the performance of all methods. However, AHE generally achieves better performance in terms of F1, G-mean and MCC, and significantly outperforms the hybrid-sampling baselines in AUCPRC. In the second series of synthetic datasets, the F1, G-mean, and MCC scores of the baselines rapidly decrease as *cov* rises, while AHE demonstrates resistance to class overlap and outperforms the baselines in AUCPRC by a large margin. DE-OS relies on an unstable evolution algorithm to generate samples and demonstrates poor performance. Nevertheless, compared using SMOTE alone, four SMOTE-based hybrid-sampling methods fail to steadily and significantly outperform, which indicates that the under-sampling components of these methods have an insignificant effect on noise reduction and performance enhancement, especially when the imbalance ratio is high and the class overlap is severe. The suboptimal performance of the hybrid-sampling baselines on synthetic datasets primarily shows the inadequacy of the conventional 'balancing-then-cleaning' pipeline for imbalanced classification, which is more obvious on large-scale highly imbalanced datasets. In contrast, AHE employs the novel 'augmenting-then-balancing' pipeline and showcases better adaptability with superior generalization ability for deteriorating data distributions, as characterized by rising IR and *cov* values.

4.3. Main results on large scale datasets

We evaluate the proposed AHE compared with 17 competitive baseline methods on four large-scale, highly imbalanced datasets. The statistic details of these benchmark datasets are presented in Table 1. For all baseline methods, DT (*Decision Tree*) is used as the classifier. The evaluation results on these datasets are summarized in four separate tables: Table 5 for the Credit Card dataset, Table 6 for the Payment Simulation dataset, Table 7 for the Normal vs R2L dataset and Table 8 for the Dos vs R2L dataset. The best results are highlighted in bold.

From an overall perspective, AHE ranks first in F1, G-mean, and MCC on all datasets, while achieving first place in AUCPRC on the Credit Card dataset and the Dos vs R2L dataset, second place on the Normal vs R2L dataset, and third place on the Payment Simulation dataset. F1, G-mean and MCC collectively reflect the classifier's prediction quality on testing data since they are calculated based on predicted labels and the confusion matrix (Table 3). The AUCPRC metric reflects the classifier's generalization ability as it represents the area under the precision-recall curve. Therefore, it can be concluded that AHE outperforms all baselines in making correct predictions with the default classification threshold of 0.5 and showcases a strong generalization ability to different classification thresholds. It can

Table 5: Overall results of four metric scores on the Credit Card dataset. The best ones are in bold. DT is used as the base classifier.

Categories	Methods	F1	G-mean	MCC	AUCPRC
Over-sampling	ADASYN	0.523±0.025	0.551±0.021	0.550±0.021	0.304±0.023
	BSMOTE	0.719±0.014	0.722±0.013	0.721±0.013	0.521±0.019
	SMOTE	0.521±0.025	0.554±0.021	0.553±0.021	0.308±0.023
Under-sampling	OSS	0.780±0.015	0.780±0.015	0.780±0.015	0.610±0.023
	ENN	0.774±0.016	0.775±0.015	0.775±0.015	0.601±0.023
	TomekLink	0.782±0.013	0.782±0.013	0.782±0.013	0.613±0.021
Hybrid-sampling	SMOTEENN	0.515±0.018	0.550±0.017	0.549±0.017	0.303±0.018
	SMOTETomek	0.521±0.025	0.554±0.021	0.553±0.021	0.308±0.023
	SMOTEIPF	0.537±0.027	0.566±0.023	0.565±0.023	0.322±0.026
	SMOTERkNN	0.217±0.024	0.328±0.023	0.325±0.024	0.108±0.159
	DE-OS	0.252±0.008	0.361±0.009	0.359±0.009	0.130±0.028
Under-sampling + Ensemble	Cascade	0.527±0.049	0.572±0.039	0.571±0.039	0.752±0.015
	Easy	0.074±0.005	0.190±0.006	0.185±0.007	0.353±0.071
	SPE	0.339±0.029	0.427±0.023	0.425±0.023	0.765±0.016
	EASE	0.425±0.035	0.492±0.027	0.491±0.027	0.789±0.015
	HUE	0.078±0.001	0.197±0.001	0.192±0.001	0.769±0.008
	TSSE	0.606±0.018	0.633±0.015	0.632±0.015	0.766±0.011
Hybrid-sampling + Ensemble	AHE	**0.831**±0.016	**0.831**±0.016	**0.831**±0.016	**0.795**±0.014

Table 6: Overall results of four metric scores on the Payment Simulation dataset. The best ones are in bold. DT is used as the base classifier.

Categories	Methods	F1	G-mean	MCC	AUCPRC
Over-sampling	ADASYN	0.877±0.002	0.882±0.002	0.882±0.002	0.778±0.003
	BSMOTE	0.849±0.002	0.851±0.002	0.851±0.002	0.724±0.004
	SMOTE	0.870±0.003	0.875±0.002	0.875±0.002	0.766±0.004
Under-sampling	OSS	0.898±0.003	0.898±0.003	0.898±0.003	0.806±0.006
	ENN	0.891±0.001	0.891±0.001	0.891±0.001	0.795±0.002
	TomekLink	0.899±0.001	0.899±0.001	0.899±0.001	0.809±0.002
Hybrid-sampling	SMOTEENN	0.820±0.003	0.831±0.002	0.831±0.002	0.690±0.004
	SMOTETomek	0.868±0.003	0.873±0.003	0.873±0.003	0.763±0.005
	SMOTEIPF	0.868±0.002	0.873±0.002	0.873±0.002	0.762±0.003
	SMOTERkNN	0.826±0.003	0.835±0.003	0.835±0.003	0.697±0.005
	DE-OS	0.361±0.002	0.468±0.001	0.467±0.001	0.219±0.001
Under-sampling + Ensemble	Cascade	0.661±0.002	0.701±0.002	0.700±0.002	0.903±0.000
	Easy	0.275±0.000	0.398±0.000	0.397±0.000	0.478±0.004
	SPE	0.889±0.000	0.890±0.000	0.890±0.000	0.951±0.000
	EASE	0.860±0.001	0.864±0.001	0.864±0.001	**0.955**±0.000
	HUE	0.261±0.002	0.387±0.001	0.386±0.001	0.943±0.002
	TSSE	0.882±0.001	0.883±0.001	0.883±0.001	0.954±0.000
Hybrid-sampling + Ensemble	AHE	**0.902**±0.000	**0.902**±0.000	**0.902**±0.000	0.952±0.001

be observed that the performance of under-sampling-based ensemble baselines is not necessarily superior to that of data-level baselines. On the Credit Card dataset, the under-sampling baselines even outperform the ensemble-based methods by a large margin in terms of F1, G-mean, and MCC. Although the under-sampling-based ensemble baselines perform well in terms of AUCPRC, their performance on the

Table 7: Overall results of four metric scores on the Normal vs R2L dataset. The best ones are in bold. DT is used as the base classifier.

Categories	Methods	F1	G-mean	MCC	AUCPRC
Over-sampling	ADASYN	0.935±0.007	0.935±0.007	0.935±0.007	0.874±0.013
	BSMOTE	0.947±0.005	0.947±0.005	0.946±0.005	0.896±0.009
	SMOTE	0.968±0.004	0.968±0.004	0.968±0.004	0.938±0.008
Under-sampling	OSS	0.948±0.010	0.948±0.010	0.948±0.010	0.898±0.020
	ENN	0.957±0.005	0.957±0.005	0.957±0.005	0.916±0.010
	TomekLink	0.948±0.006	0.948±0.006	0.948±0.006	0.898±0.010
Hybrid-sampling	SMOTEENN	0.965±0.005	0.965±0.005	0.965±0.005	0.930±0.009
	SMOTETomek	0.968±0.005	0.969±0.005	0.968±0.005	0.938±0.009
	SMOTEIPF	0.966±0.003	0.966±0.003	0.966±0.003	0.933±0.006
	SMOTERkNN	0.965±0.005	0.965±0.005	0.965±0.005	0.931±0.010
	DE-OS	0.939±0.012	0.939±0.012	0.939±0.012	0.882±0.022
Under-sampling + Ensemble	Cascade	0.843±0.041	0.852±0.035	0.852±0.035	0.978±0.004
	Easy	0.181±0.018	0.315±0.018	0.313±0.018	0.537±0.101
	SPE	**0.981**±0.003	**0.981**±0.003	**0.981**±0.003	0.989±0.003
	EASE	0.980±0.005	0.980±0.005	0.980±0.005	**0.992**±0.002
	HUE	0.167±0.001	0.302±0.001	0.300±0.001	0.970±0.000
	TSSE	0.980±0.002	0.980±0.002	0.980±0.002	0.988±0.003
Hybrid-sampling + Ensemble	AHE	**0.981**±0.004	**0.981**±0.004	**0.981**±0.004	0.990±0.003

Table 8: Overall results of four metric scores on the Dos vs R2L dataset. The best ones are in bold. DT is used as the base classifier.

Categories	Methods	F1	G-mean	MCC	AUCPRC
Over-sampling	ADASYN	0.996±0.005	0.996±0.005	0.996±0.005	0.991±0.010
	BSMOTE	0.998±0.000	0.998±0.000	0.998±0.000	0.996±0.000
	SMOTE	0.990±0.001	0.990±0.001	0.990±0.001	0.981±0.002
Under-sampling	OSS	0.995±0.002	0.995±0.002	0.995±0.002	0.990±0.005
	ENN	0.990±0.002	0.990±0.002	0.990±0.002	0.981±0.004
	TomekLink	0.992±0.002	0.992±0.002	0.992±0.002	0.985±0.004
Hybrid-sampling	SMOTEENN	0.990±0.001	0.990±0.001	0.990±0.001	0.981±0.002
	SMOTETomek	0.990±0.001	0.990±0.001	0.990±0.001	0.981±0.002
	SMOTEIPF	0.990±0.001	0.990±0.001	0.990±0.001	0.981±0.002
	SMOTERkNN	0.988±0.003	0.988±0.003	0.988±0.003	0.976±0.006
	DE-OS	0.906±0.030	0.908±0.028	0.908±0.028	0.826±0.052
Under-sampling + Ensemble	Cascade	0.961±0.023	0.962±0.022	0.962±0.022	0.997±0.004
	Easy	0.369±0.053	0.475±0.042	0.475±0.042	0.785±0.149
	SPE	**0.997**±0.001	**0.997**±0.001	**0.997**±0.001	**1.000**±0.000
	EASE	0.994±0.003	0.994±0.003	0.994±0.003	0.999±0.001
	HUE	0.344±0.021	0.456±0.017	0.456±0.017	0.998±0.001
	TSSE	0.994±0.003	0.994±0.003	0.994±0.003	0.999±0.001
Hybrid-sampling + Ensemble	AHE	**0.997**±0.001	**0.997**±0.001	**0.997**±0.001	**1.000**±0.001

other three metrics, which are closely related to prediction quality and sensitive to threshold selection, are comparatively poor. This suggests the limited applicability of under-sampling-based ensemble methods on large-scale highly imbalanced datasets. In contrast, AHE not only demonstrates acceptable generalization ability but also achieves the best performance on F1, G-mean, and MCC metrics.

The experimental results again highlight the limitations of conventional hybrid-sampling baselines, which are based on the 'balancing-then-cleaning' pipeline and exhibit similar poor performance to SMOTE. It can be seen that except for SMOTEIPF outperforming SMOTE on the Credit Card dataset and SMOTETomek surpassing SMOTE in terms of G-mean on the Normal vs R2L dataset, all hybrid-sampling methods do not achieve better performance than SMOTE in other cases. Meanwhile, these hybrid-sampling methods fail to steadily outperform under-sampling methods. In particular, on the Credit Card dataset, these hybrid-sampling methods perform significantly worse than the under-sampling ones, which suffer from information loss and do not require sample generation. These observations suggest that the conventional hybrid-sampling pipeline is inappropriate for handling large-scale highly imbalanced datasets, not only exacerbating the drawbacks of over-sampling but also failing to leverage the benefits of under-sampling. In contrast, AHE over-samples the minority class for partial augmentation and under-samples the majority class for class balance. The noise introduction is limited from the root as an appropriate number of samples are rigorously generated, and the information loss from the majority class is effectively mitigated as base classfiers are trained on augmented training data and combined as an ensemble model.

4.4. Applicability on small-scale datasets

In real-world scenarios, imbalanced datasets with small sample sizes and moderate imbalance ratios are also commonly encountered as the large-scale, highly imbalanced ones. Therefore, we aim to validate the applicability of AHE on small-scale imbalanced datasets, despite its design being oriented toward the large-scale, highly imbalanced scenarios. To this end, we select 20 benchmark datasets with relatively small sample sizes and imbalance ratios for evaluation. The statistics of these datasets are provided in Table 2, arranged in ascending order of IR. Table 9 presents the average performance of AHE and 17 baselines on all small-scale datasets with respective rankings. For a detailed breakdown, the individual metric scores on each small-scale dataset are presented in four separate tables in the Appendix: Table 16 for F1, Table 17 for G-mean, Table 18 for MCC and Table 19 for AUCPRC. It can be broadly concluded that:

- AHE outperforms on datasets with relatively large sample sizes and high imbalance ratios (wine_quality, letter_img, mammography, ozone_level and protein_homo) in terms of F1, G-mean and MCC. These metrics are closely related to actual prediction quality and highly sensitive to the selection of classification threshold. In practical applications, it is challenging to predefine the optimal classification threshold on testing data to produce the optimal classification

Table 9: Average performance of AHE and 17 baselines across four metrics on 20 small-scale datasets with the rankings of the metric scores shown in parentheses. The best ones are in bold.

Methods	**F1**	**G-mean**	**MCC**	**AUCPRC**	**Average Rank**
SMOTE	0.507 (14)	0.516 (15)	0.488 (15)	0.387 (13)	14.25 (16)
ADASYN	0.508 (13)	0.516 (15)	0.487 (16)	0.388 (12)	14.0 (15)
BSMOTE	0.506 (15)	0.509 (17)	0.481 (17)	0.383 (16)	16.25 (17)
OSS	0.521 (9)	0.525 (12)	0.499 (9)	0.391 (10)	10.0 (8)
ENN	0.550 (6)	0.554 (6)	0.527 (6)	0.410 (8)	6.5 (6)
TomekLink	0.513 (11)	0.517 (14)	0.490 (12)	0.384 (15)	13.0 (14)
SMOTEENN	0.522 (7)	0.539 (7)	0.505 (7)	0.385 (14)	8.75 (7)
SMOTETomek	0.510 (12)	0.519 (13)	0.490 (12)	0.389 (11)	12.0 (13)
SMOTEIPF	0.518 (10)	0.527 (11)	0.498 (10)	0.393 (9)	10.0 (8)
SMOTERkNN	0.522 (7)	0.533 (8)	0.503 (8)	0.374 (17)	10.0 (8)
DE-OS	0.477 (16)	0.501 (18)	0.464 (18)	0.348 (18)	17.5 (18)
Easy	0.477 (16)	0.531 (10)	0.490 (12)	0.546 (7)	11.25 (12)
Cascade	0.564 (5)	0.586 (5)	0.555 (5)	0.588 (6)	5.25 (5)
SPE	0.572 (4)	0.604 (4)	0.570 (4)	0.600 (5)	4.25 (4)
EASE	0.588 (3)	0.611 (3)	0.581 (3)	**0.634 (1)**	2.5 (3)
TSSE	0.600 (2)	0.618 (2)	0.588 (2)	0.618 (3)	2.25 (2)
HUE	0.473 (18)	0.532 (9)	0.491 (11)	0.626 (2)	10.0 (8)
AHE	**0.614 (1)**	**0.631 (1)**	**0.602 (1)**	0.612 (4)	**1.75 (1)**

results. Therefore, the classification performance under the default threshold value of 0.5 can effectively reflect the applicability of different methods on small-scale imbalanced datasets. Though performing slightly worse than the optimal results in terms of AUCPRC, AHE achieves the best or near-optimal performance on the other three metrics.

- The slightly lower AUCPRC metric scores of AHE can be attributed to sample generation. For small-scale imbalanced datasets, the minority class samples are more sparse than the large-scale ones, which makes it harder to accurately estimate the sub-concept distributions. This can lead to noise introduction during the partial over-sampling phase in AHE and cause degradation in generalization performance, as reflected by the AUCPRC score. Such a inherent limitation of sample generation is also evidenced by the poor AUCPRC performance of over-sampling and hybrid-sampling baselines. In comparison, under-sampling-based ensemble methods generally excel in terms of AUCPRC. On some datasets (e.g. satimage and sick_euthyroid), under-sampling methods even exhibit better AUCPRC performance than over-sampling, despite suffering from information loss in the majority class. Additionally, it should be noted that even on small-scale datasets, hybrid-sampling methods fail to con-

Table 10: Total counts of best rankings of AHE and 17 baselines on small-scale datasets over four metric scores. The best ones are in bold.

Methods	**F1**	**G-mean**	**MCC**	**AUCPRC**	**Sum-ranking**
SMOTE	0	0	0	0	0
ADASYN	0	0	0	0	0
BSMOTE	0	0	0	0	0
OSS	0	0	0	0	0
ENN	1	1	1	0	3
TomekLink	0	0	0	0	0
SMOTEENN	0	0	0	0	0
SMOTETomek	0	0	0	0	0
SMOTEIPF	2	2	2	0	6
SMOTERkNN	1	1	1	0	3
DE-OS	0	0	0	0	0
Easy	0	0	0	1	1
Cascade	1	1	1	0	3
SPE	3	3	3	1	10
EASE	0	0	0	**9**	9
TSSE	2	2	2	2	8
HUE	2	2	2	7	13
AHE	**8**	**8**	**8**	1	**25**

sistently outperform SMOTE alone, which again highlights the inadequacy of the 'balancing-then-cleaning' sampling strategy.

- In summary, AHE remains considerably competitive and applicable on small-scale imbalanced datasets. Generally, AHE is more suitable for handling datasets with large sample sizes and high imbalance, due to its intrinsic design. Compared to over-sampling and hybrid-sampling baselines, AHE adopts a partial augmentation strategy that avoids the introduction of a large number of synthetic samples, thereby achieving stronger generalization ability.

Furthermore, we conduct statistical analysis on the performance of AHE and all baselines for a comprehensive evaluation. From Table 9, it can be seen that AHE outperforms all baselines with an average rank score on four metrics of 1.75. AHE achieves the best performance on F1, G-mean and MCC on average with an acceptable average AUCPRC performance in the fourth place. Table 10 presents the total number of best scores across four metrics for AHE and each baseline on small-scale datasets. It can be observed that AHE exhibits a significant advantage in terms of F1, G-mean, and MCC. To measure the extent of the performance difference between AHE and baselines, we conduct separate Friedman tests for each of the AUCPRC, F1, G-mean, and MCC scores based on Table 16 -19 with a significance

Table 11: Significance results of Bonferroni correction test for compared method pairs in terms of F1, G-mean, MCC and AUCPRC.

Method pairs	**F1**		**G-mean**		**MCC**		**AUCPRC**	
	P-values	Hypothesis	P-values	Hypothesis	P-values	Hypothesis	P-values	Hypothesis
AHE vs SMOTE	1.05E-05	Reject	1.05E-05	Reject	9.03E-06	Reject	1.16E-07	Reject
AHE vs ADASYN	1.33E-05	Reject	1.33E-05	Reject	9.82E-06	Reject	1.19E-07	Reject
AHE vs BSMOTE	1.20E-07	Reject	1.20E-07	Reject	2.86E-07	Reject	1.06E-08	Reject
AHE vs OSS	2.70E-04	Reject	2.70E-04	Reject	3.06E-04	Reject	5.00E-06	Reject
AHE vs ENN	5.05E-03	Reject	5.05E-03	Reject	6.56E-03	Reject	5.56E-05	Reject
AHE vs TomekLink	1.61E-04	Reject	1.61E-04	Reject	1.43E-04	Reject	9.01E-07	Reject
AHE vs SMOTEENN	2.60E-04	Reject	2.60E-04	Reject	1.60E-04	Reject	1.06E-06	Reject
AHE vs SMOTETomek	5.35E-05	Reject	5.35E-05	Reject	4.64E-05	Reject	4.94E-07	Reject
AHE vs SMOTEIPF	4.19E-04	Reject	4.19E-04	Reject	3.06E-04	Reject	1.06E-06	Reject
AHE vs SMOTERkNN	1.61E-04	Reject	1.61E-04	Reject	1.29E-04	Reject	1.55E-06	Reject
AHE vs DE-OS	1.99E-06	Reject	1.99E-06	Reject	4.49E-07	Reject	1.47E-08	Reject
AHE vs Easy	2.60E-04	Reject	2.60E-04	Reject	4.64E-05	Reject	6.84E-01	Accept
AHE vs Cascade	1.53E-01	Accept	1.53E-01	Accept	7.72E-02	Accept	1.00E+00	Accept
AHE vs SPE	1.00E+00	Accept	1.00E+00	Accept	6.08E-01	Accept	1.00E+00	Accept
AHE vs EASE	1.00E+00	Accept	1.00E+00	Accept	9.54E-01	Accept	1.00E+00	Accept
AHE vs TSSE	1.00E+00	Accept	1.00E+00	Accept	9.54E-01	Accept	1.00E+00	Accept
AHE vs HUE	4.91E-04	Reject	4.91E-04	Reject	1.43E-04	Reject	1.00E+00	Accept

level $\alpha = 0.05$. The Friedman test initially assumes a null hypothesis that there is no significant difference in the performance of all compared methods (AHE and baselines) for each metric. Based on the original metric score table, we compute the average ranking of each method and obtain the Iman–Davenport statistics for each metric: $F_{AUCPRC} = 53.41$, $F_{F1} = 8.21$, $F_{G-mean} = 9.65$ and $F_{MCC} = 8.87$. When $\alpha = 0.05$, the critical value for the F-distribution with degree of freedom of $(n-1)$ and $(n-1) \times (m-1)$ is $F_F = 1.65$, where $n = 18$ is the number of all compared methods (AHE and 17 baselines) and $m = 20$ is the number of small scale datasets. Clearly, the null hypotheses are rejected as the Iman–Davenport statistics of four metrics are greater than $F_F = 1.65$. Therefore, it can be concluded that there is a significant difference in performance among these methods. Following this, to further validate if significant performance difference exists between AHE and each baseline, we use the Bonferroni correction to conduct a post-hoc test. The choice of post-hoc test [6] is based on the suggestion in [54], as all baselines are compared to a control method (the proposed AHE). The null hypothesis is that AHE does not significantly

[6]Other common post hoc tests like the Nemenyi test and the Wilcoxon signed-rank test are typically used for pairwise comparisons of all methods and for comparing two methods, respectively. However, neither is suitable for the experimental scenario in this paper.

Table 12: Overall results of four metric scores on the Credit Card dataset. The best ones are in bold. SVM is used as the base classifier.

Categories	**Metrics**	**F1**	**G-mean**	**MCC**	**AUCPRC**
Over-sampling	ADASYN	0.551±0.002	0.558±0.002	0.557±0.002	0.609±0.002
	BSMOTE	0.705±0.000	0.706±0.000	0.705±0.000	0.731±0.003
	SMOTE	0.480±0.004	0.512±0.003	0.511±0.003	0.627±0.001
Under-sampling	OSS	0.784±0.003	0.785±0.003	0.784±0.003	0.788±0.001
	ENN	0.761±0.000	0.761±0.000	0.761±0.000	0.729±0.000
	Tomek	0.784±0.000	0.784±0.000	0.783±0.000	0.788±0.000
Hybrid-sampling	SMOTEENN	0.485±0.005	0.527±0.004	0.526±0.004	0.617±0.001
	SMOTETomek	0.480±0.004	0.512±0.003	0.511±0.003	0.627±0.001
	SMOTEIPF	0.513±0.005	0.539±0.004	0.538±0.004	0.631±0.001
	SMOTERkNN	0.543±0.004	0.577±0.004	0.576±0.004	0.641±0.002
	DE-OS	0.426±0.008	0.493±0.007	0.491±0.007	0.689±0.017
Under-sampling + Ensemble	Cascade	0.418±0.097	0.490±0.097	0.489±0.021	0.788±0.122
	Easy	0.067±0.007	0.182±0.007	0.177±0.017	0.589±0.005
	SPE	0.581±0.028	0.610±0.028	0.609±0.010	0.794±0.034
	EASE	0.825±0.006	0.828±0.006	0.828±0.003	0.810±0.006
	HUE	0.068±0.000	0.184±0.000	0.179±0.015	0.764±0.000
	TSSE	0.768±0.035	0.769±0.035	0.769±0.018	0.787±0.036
Hybrid-sampling + Ensemble	AHE	**0.850**±0.004	**0.850**±0.004	**0.850**±0.004	**0.850**±0.002

differ from each compared baseline on a certain metric score. If the P-value obtained from the post-hoc test is less than the significant level ($\alpha = 0.05$), the corresponding null hypothesis is accepted, which indicates that AHE does not have a significant performance difference with the compared baseline. The significance results are presented in Table 11. It can be concluded that AHE significantly outperforms all data-level baselines in terms of four metrics. Moreover, AHE does not exhibit statistically significant performance differences when compared to under-sampling-based ensemble baselines. This aligns with the results presented in Table 9, where AHE demonstrates average performance and rankings comparable to those of the baselines (Cascade, SPE, EASE, and TSSE). Nevertheless, AHE still achieves the best average performance and the most top rankings in F1, G-mean, and MCC, while maintaining an acceptable AUCPRC performance close to the optimal.

4.5. Evaluation on different base classifiers

In the aforementioned experiments, DT is used as the base classifier to ensure consistency with previous research results. However, in recent ensemble-based imbalanced learning studies, the performance of other base classifiers on large-scale, highly imbalanced datasets has bee less investigated. Therefore, we choose SVM (*Support Vector Machine*), MLP (*Multi-Layer Perceptron*) and LR (*Logistic Regression*) as base classifiers to evaluate AHE compared to other baselines. The Credit Card dataset is used for evaluation. Experimental results across four metrics for

Table 13: Overall results of four metric scores on the Credit Card dataset. The best ones are in bold. MLP is used as the base classifier.

Categories	Metrics	F1	G-mean	MCC	AUCPRC
Over-sampling	ADASYN	0.075±0.022	0.186±0.029	0.181±0.030	0.550±0.259
	BSMOTE	0.614±0.029	0.634±0.026	0.633±0.026	0.739±0.028
	SMOTE	0.115±0.020	0.233±0.023	0.230±0.023	0.699±0.070
Under-sampling	OSS	0.799±0.024	0.802±0.022	0.802±0.022	0.784±0.056
	ENN	0.808±0.019	0.809±0.018	0.809±0.018	0.778±0.054
	Tomek	0.812±0.016	0.814±0.015	0.814±0.015	0.784±0.057
Hybrid-sampling	SMOTEENN	0.113±0.023	0.232±0.025	0.229±0.025	0.702±0.062
	SMOTETomek	0.115±0.020	0.233±0.023	0.230±0.023	0.699±0.070
	SMOTEIPF	0.115±0.021	0.234±0.023	0.231±0.023	0.686±0.083
	SMOTERkNN	0.148±0.024	0.271±0.023	0.268±0.024	0.621±0.159
	DE-OS	0.113±0.008	0.237±0.009	0.233±0.009	0.744±0.028
Under-sampling + Ensemble	Cascade	0.656±0.153	0.678±0.124	0.677±0.124	0.770±0.023
	Easy	0.119±0.026	0.244±0.028	0.241±0.028	0.749±0.040
	SPE	0.487±0.240	0.536±0.196	0.535±0.197	0.715±0.029
	EASE	0.793±0.018	0.795±0.017	0.795±0.017	**0.828**±0.004
	HUE	0.089±0.033	0.205±0.049	0.200±0.051	0.291±0.165
	TSSE	0.809±0.015	0.810±0.014	0.809±0.014	0.784±0.014
Hybrid-sampling + Ensemble	AHE	**0.824**±0.015	**0.825**±0.014	**0.825**±0.014	0.810±0.024

Table 14: Overall results of four metric scores on the Credit Card dataset. The best ones are in bold. LR is used as the base classifier.

Categories	Metrics	F1	G-mean	MCC	AUCPRC
Over-sampling	ADASYN	0.033±0.000	0.129±0.000	0.122±0.000	0.707±0.000
	BSMOTE	0.379±0.001	0.457±0.001	0.455±0.001	0.710±0.000
	SMOTE	0.118±0.001	0.243±0.001	0.240±0.001	0.729±0.000
Under-sampling	OSS	0.654±0.000	0.675±0.000	0.674±0.000	0.750±0.000
	ENN	0.679±0.000	0.696±0.000	0.696±0.000	0.761±0.000
	Tomek	0.654±0.000	0.675±0.000	0.674±0.000	0.750±0.000
Hybrid-sampling	SMOTEENN	0.115±0.001	0.239±0.001	0.236±0.001	0.724±0.001
	SMOTETomek	0.118±0.001	0.243±0.001	0.240±0.001	0.729±0.000
	SMOTEIPF	0.537±0.027	0.566±0.023	0.565±0.023	0.322±0.026
	SMOTERkNN	0.098±0.001	0.223±0.001	0.219±0.001	0.697±0.001
	DE-OS	0.113±0.008	0.237±0.009	0.233±0.009	0.744±0.028
Under-sampling + Ensemble	Cascade	0.495±0.064	0.547±0.051	0.546±0.051	0.775±0.010
	Easy	0.099±0.007	0.223±0.008	0.219±0.009	0.717±0.027
	SPE	0.768±0.013	0.768±0.013	0.767±0.013	0.719±0.014
	EASE	0.772±0.013	0.776±0.011	0.776±0.011	**0.797**±0.006
	HUE	0.106±0.000	0.232±0.000	0.228±0.000	0.274±0.010
	TSSE	0.810±0.015	0.811±0.014	0.810±0.014	0.749±0.006
Hybrid-sampling + Ensemble	AHE	**0.813**±0.001	**0.814**±0.001	**0.813**±0.001	0.779±0.002

each classifier are presented in three separate tables: Table 12 for SVM, Table 13 for MLP and Table 14 for LR. It can be seen that although AHE performs slightly worse than EASE on the AUCPRC metric when MLP and LR are used as the base classifiers, it achieves the best metric scores in all other evaluation scenarios. Moreover, all baselines exhibit a certain degree of inadequacy in adapting to different

Table 15: Ablation results on the Credit Card dataset. The best ones are in bold.

Methods	F1	G-mean	MCC	AUCPRC
AHE	**0.831**±0.016	**0.831**±0.016	**0.831**±0.016	**0.795**±0.014
w/o partial over-sampling	0.429±0.041	0.496±0.033	0.495±0.033	0.770±0.017
w/o cluster-wise sample generation	0.807±0.022	0.807±0.022	0.807±0.022	0.790±0.015
w/o specified sampling amount	0.818±0.009	0.818±0.009	0.818±0.009	0.786±0.019
w/o bounded ratio	0.826±0.013	0.826±0.015	0.826±0.013	0.789±0.013
w/o dynamic hardness histogram	0.829±0.012	0.829±0.012	0.829±0.012	0.786±0.014
w/o uniform sampling weight	0.819±0.015	0.819±0.015	0.819±0.015	0.785±0.014

classifiers. Over-sampling and hybrid-sampling methods typically achieve satisfactory performance with SVM, but demonstrate poor performance with MLP and LR. The performance of under-sampling methods on LR is generally not as good as that on the other three base classifiers. When using more powerful base classifiers than DT, under-sampling-based ensemble methods struggle to achieve better performance than individual under-sampling, which indicates their deficiency in adapting to different base classifiers. In comparison, AHE surpasses all baselines in terms of F1, G-mean, and MCC, while achieving acceptable AUCPRC performance close to the optimal results and demonstrating independence from certain base classifiers to achieve superior classification performance.

4.6. Ablation study

We emphasize the importance of the partial over-sampling phase and the uniform under-sampling phase in AHE for learning a well-performing classification model from massive highly imbalanced data, respectively. The partial over-sampling phase is characterized by three modifications to the original SMOTE method: (1) cluster-wise sample generation, (2) sampling amount for x_{seed} specified by local hardness and (3) bounded ratio for interpolation. The uniform under-sampling phase is characterized by the dynamic hardness histogram and the uniform sampling weight. To verify the effectiveness of the above modules, we use DT as the base classifier and conduct an ablation experiment on the Credit Card dataset to evaluate the full AHE and its six variants. The metric scores are presented in Table 15 where (1) w/o partial over-sampling, (2) w/o cluster-wise sample generation, (3) w/o sampling amount specified by local hardness (abbreviated as specified sampling amount), (4) w/o bounded ratio, (5) w/o dynamic hardness histogram and (6) w/o uniform sampling weight. Remarkably, the partial over-sampling phase significantly improves the classification performance of AHE, and each module in the improved SMOTE method contributes to the superior performance of AHE. Since AHE does not adopt a weighting mechanism for base classifiers as in EASE [25] and TSSE [26], its variant without the partial over-sampling phase shares the same boosting-like ensemble framework with

SPE [24], only with a different under-sampling strategy. Without the auxiliary synthetic samples, updating dynamic hardness histogram and uniformly sampling from each bin results in better performance than the baseline method SPE, which sets the bin counts to a fixed number and samples from each bin with self-paced sampling weights, thereby evidencing the effectiveness of the uniform under-sampling phase. In the w/o uniform sampling weight setting, the self-paced sampling weight in SPE is employed for substitution, which results in performance degradation. Also, using fixed bin counts instead of the dynamic hardness histogram in the w/o dynamic hardness histogram setting leads to suboptimal performance. Henceforth, it can be concluded from the ablation evaluation results that none of the above components of AHE is dispensable.

5. Conclusion

In this paper, we propose a hybrid-sampling-based ensemble framework (AHE) for large-scale, highly imbalanced data classification. AHE specifies the sampling amount required for eliminating the within-class imbalance of the minority class and employs an improved SMOTE method for sample generation, which is featured as cluster-wise sample generation, sampling center selection and bounded interpolation ratio. Moreover, AHE constructs diversified sampling subspaces by updating a dynamic hardness histogram, which establishes the optimal majority class segmentation based on the learning state of the ensemble model. Thus, balanced and informative training sets are formed with augmented minority class samples and diversified majority class samples from various hardness levels. Extensive experiments are conducted to validate the effectiveness of AHE and the indispensability of each component. In summary, compared to hybrid-sampling methods, AHE avoids substantial sample generation and aggregates multiple classifiers to obtain superior performance; compared to under-sampling-based ensemble methods, AHE expands the training set size through appropriate minority class augmentation and focuses on mining diversified training samples from the majority class, thereby excelling in large-scale, highly imbalanced data classification.

6. Appendix

The appendix presents the complete original experimental results across four evaluation metrics (i.e., F1, G-mean, MCC and AUCPRC) of AHE and 17 baselines on the small-scale datasets. The best results are highlighted in bold.

Table 16: F1 score of AHE and 17 baselines on 20 small-scale datasets. Best results are in bold.

Dataset	optical__digits	satimage	sick__euthyroid	spectrometer	car__eval__34	us__crime	scene	libras__move	thyroid__sick	coil__2000
SMOTE	0.822 ± 0.023	0.559 ± 0.036	0.836 ± 0.030	0.704 ± 0.115	0.986 ± 0.032	0.461 ± 0.048	0.215 ± 0.044	0.558 ± 0.160	0.846 ± 0.015	0.117 ± 0.022
ADASYN	0.809 ± 0.020	0.561 ± 0.039	0.830 ± 0.019	0.707 ± 0.105	0.984 ± 0.032	0.418 ± 0.063	0.228 ± 0.031	0.655 ± 0.170	0.830 ± 0.020	0.082 ± 0.015
BSMOTE	0.794 ± 0.020	0.554 ± 0.028	0.846 ± 0.016	0.684 ± 0.103	0.972 ± 0.041	0.429 ± 0.080	0.231 ± 0.034	0.517 ± 0.146	0.857 ± 0.023	0.079 ± 0.013
OSS	0.795 ± 0.032	0.556 ± 0.021	0.874 ± 0.010	0.641 ± 0.075	0.965 ± 0.018	0.394 ± 0.063	0.280 ± 0.037	0.444 ± 0.083	0.880 ± 0.033	0.100 ± 0.011
ENN	0.812 ± 0.017	0.588 ± 0.014	0.880 ± 0.021	0.618 ± 0.063	0.965 ± 0.018	0.509 ± 0.025	0.172 ± 0.031	0.614 ± 0.107	0.854 ± 0.020	0.144 ± 0.012
TomekLink	0.796 ± 0.013	0.569 ± 0.020	0.871 ± 0.008	0.590 ± 0.067	0.965 ± 0.018	0.377 ± 0.060	0.259 ± 0.032	0.391 ± 0.052	0.872 ± 0.027	0.095 ± 0.009
SMOTEENN	0.829 ± 0.012	0.568 ± 0.019	0.854 ± 0.021	0.729 ± 0.082	0.951 ± 0.011	0.478 ± 0.050	0.217 ± 0.025	0.545 ± 0.095	0.893 ± 0.018	0.189 ± 0.015
SMOTETomek	0.822 ± 0.023	0.560 ± 0.038	0.849 ± 0.032	0.703 ± 0.106	0.986 ± 0.032	0.461 ± 0.048	0.213 ± 0.043	0.558 ± 0.160	0.845 ± 0.016	0.114 ± 0.021
SMOTEIPF	0.807 ± 0.026	0.554 ± 0.020	0.839 ± 0.031	0.729 ± 0.067	**0.996** ± 0.008	0.478 ± 0.054	0.194 ± 0.031	0.565 ± 0.224	0.862 ± 0.022	0.099 ± 0.027
SMOTERkNN	0.793 ± 0.028	0.577 ± 0.023	0.873 ± 0.021	0.642 ± 0.087	0.889 ± 0.046	0.526 ± 0.045	0.231 ± 0.052	0.526 ± 0.095	0.876 ± 0.013	0.180 ± 0.019
DE-OS	0.619 ± 0.057	0.549 ± 0.024	0.843 ± 0.035	0.704 ± 0.105	0.973 ± 0.028	0.426 ± 0.035	0.222 ± 0.035	0.385 ± 0.078	0.798 ± 0.045	0.111 ± 0.015
Cascade	0.913 ± 0.012	0.648 ± 0.023	**0.891** ± 0.013	0.884 ± 0.084	0.957 ± 0.019	0.563 ± 0.036	0.235 ± 0.032	0.657 ± 0.156	0.868 ± 0.018	0.172 ± 0.008
SPE	**0.938** ± 0.007	0.625 ± 0.020	0.882 ± 0.012	0.917 ± 0.041	0.981 ± 0.013	0.515 ± 0.042	0.249 ± 0.030	0.627 ± 0.064	**0.925** ± 0.021	0.180 ± 0.007
EASE	0.922 ± 0.018	0.642 ± 0.022	0.889 ± 0.019	0.891 ± 0.052	0.980 ± 0.020	0.557 ± 0.051	0.278 ± 0.034	0.720 ± 0.066	0.905 ± 0.020	0.180 ± 0.010
TSSE	0.916 ± 0.012	**0.663** ± 0.028	0.889 ± 0.017	0.907 ± 0.057	0.970 ± 0.029	0.540 ± 0.077	0.270 ± 0.061	0.755 ± 0.111	0.900 ± 0.019	0.160 ± 0.016
HUE	0.891 ± 0.011	0.574 ± 0.009	0.817 ± 0.014	0.677 ± 0.046	0.935 ± 0.021	0.461 ± 0.011	**0.284** ± 0.012	0.473 ± 0.050	0.756 ± 0.016	**0.192** ± 0.005
Easy	0.883 ± 0.018	0.564 ± 0.018	0.818 ± 0.024	0.693 ± 0.090	0.904 ± 0.030	0.473 ± 0.020	0.254 ± 0.021	0.514 ± 0.077	0.761 ± 0.024	0.180 ± 0.011
AHE	0.933 ± 0.011	0.652 ± 0.026	0.889 ± 0.016	**0.928** ± 0.079	0.985 ± 0.012	**0.573** ± 0.029	0.271 ± 0.026	**0.772** ± 0.079	0.921 ± 0.028	0.188 ± 0.011

Dataset	solar__flare__m0	oil	car__eval__4	wine__quality	letter__img	yeast__me2	ozone__level	mammography	protein__homo	abalone__19
SMOTE	0.131 ± 0.043	0.392 ± 0.071	0.963 ± 0.000	0.265 ± 0.054	0.913 ± 0.010	0.095 ± 0.081	0.158 ± 0.045	0.531 ± 0.017	0.514 ± 0.010	0.089 ± 0.027
ADASYN	0.094 ± 0.030	0.403 ± 0.098	0.963 ± 0.000	0.306 ± 0.034	0.914 ± 0.007	0.058 ± 0.070	0.185 ± 0.055	0.499 ± 0.026	0.543 ± 0.011	0.101 ± 0.022
BSMOTE	0.143 ± 0.025	0.322 ± 0.104	0.963 ± 0.000	0.329 ± 0.029	0.899 ± 0.009	0.101 ± 0.063	0.149 ± 0.040	0.548 ± 0.028	0.645 ± 0.009	0.066 ± 0.036
OSS	0.167 ± 0.032	0.492 ± 0.080	0.846 ± 0.050	0.372 ± 0.020	0.922 ± 0.009	0.176 ± 0.013	0.175 ± 0.078	0.600 ± 0.027	0.722 ± 0.007	0.033 ± 0.053
ENN	0.286 ± 0.006	**0.557** ± 0.025	0.834 ± 0.058	0.358 ± 0.021	0.927 ± 0.009	0.351 ± 0.041	0.227 ± 0.049	0.592 ± 0.010	0.721 ± 0.009	0.000 ± 0.000
TomekLink	0.167 ± 0.029	0.508 ± 0.084	0.834 ± 0.058	0.365 ± 0.016	0.921 ± 0.009	0.171 ± 0.004	0.205 ± 0.042	0.601 ± 0.019	0.723 ± 0.008	0.000 ± 0.000
SMOTEENN	0.274 ± 0.058	0.371 ± 0.101	0.915 ± 0.044	0.278 ± 0.038	0.909 ± 0.018	0.219 ± 0.042	0.186 ± 0.046	0.463 ± 0.021	0.515 ± 0.016	0.061 ± 0.012
SMOTETomek	0.131 ± 0.043	0.409 ± 0.066	0.963 ± 0.000	0.276 ± 0.048	0.913 ± 0.010	0.109 ± 0.077	0.157 ± 0.038	0.538 ± 0.021	0.514 ± 0.010	0.090 ± 0.027
SMOTEIPF	0.148 ± 0.035	0.478 ± 0.081	0.963 ± 0.000	0.286 ± 0.032	0.919 ± 0.017	0.105 ± 0.032	0.174 ± 0.041	0.533 ± 0.022	0.520 ± 0.013	**0.112** ± 0.042
SMOTERkNN	**0.343** ± 0.036	0.448 ± 0.029	0.840 ± 0.080	0.264 ± 0.047	0.895 ± 0.010	0.222 ± 0.074	0.166 ± 0.035	0.522 ± 0.016	0.555 ± 0.012	0.080 ± 0.029
DE-OS	0.216 ± 0.040	0.428 ± 0.073	0.963 ± 0.000	0.288 ± 0.033	0.830 ± 0.050	0.211 ± 0.047	0.181 ± 0.067	0.398 ± 0.023	0.347 ± 0.005	0.046 ± 0.012
Cascade	0.229 ± 0.038	0.402 ± 0.052	0.801 ± 0.037	0.302 ± 0.034	0.913 ± 0.017	0.359 ± 0.054	0.259 ± 0.046	0.514 ± 0.066	0.689 ± 0.018	0.043 ± 0.015
SPE	0.216 ± 0.025	0.441 ± 0.047	**0.971** ± 0.023	0.260 ± 0.024	0.955 ± 0.007	0.332 ± 0.041	0.227 ± 0.028	0.504 ± 0.019	0.659 ± 0.013	0.042 ± 0.005
EASE	0.262 ± 0.032	0.474 ± 0.049	0.948 ± 0.061	0.309 ± 0.024	0.953 ± 0.007	0.336 ± 0.098	0.252 ± 0.032	0.531 ± 0.035	0.684 ± 0.013	0.054 ± 0.010
TSSE	0.228 ± 0.034	0.488 ± 0.048	0.958 ± 0.086	0.346 ± 0.034	0.956 ± 0.012	**0.393** ± 0.094	0.275 ± 0.042	0.551 ± 0.044	0.784 ± 0.011	0.055 ± 0.014
HUE	0.239 ± 0.013	0.264 ± 0.007	0.743 ± 0.000	0.268 ± 0.007	0.744 ± 0.012	0.293 ± 0.014	0.189 ± 0.006	0.320 ± 0.007	0.315 ± 0.003	0.038 ± 0.000
Easy	0.239 ± 0.014	0.337 ± 0.034	0.760 ± 0.040	0.259 ± 0.020	0.762 ± 0.014	0.277 ± 0.030	0.203 ± 0.013	0.324 ± 0.016	0.310 ± 0.015	0.038 ± 0.005
AHE	0.250 ± 0.003	0.458 ± 0.081	0.957 ± 0.028	**0.407** ± 0.047	**0.959** ± 0.013	0.317 ± 0.089	**0.292** ± 0.045	**0.662** ± 0.030	**0.830** ± 0.011	0.042 ± 0.012

Table 17: G-mean score of AHE and 17 baselines on 20 small-scale datasets. Best results are in bold.

Dataset	**optical_digits**	**satimage**	**sick_euthyroid**	**spectrometer**	**car_eval_34**	**us_crime**	**scene**	**libras_move**	**thyroid_sick**	**coil_2000**
SMOTE	0.823 ± 0.023	0.560 ± 0.036	0.837 ± 0.030	0.717 ± 0.112	0.987 ± 0.030	0.465 ± 0.049	0.221 ± 0.045	0.579 ± 0.148	0.846 ± 0.015	0.117 ± 0.023
ADASYN	0.810 ± 0.020	0.564 ± 0.039	0.830 ± 0.019	0.712 ± 0.103	0.985 ± 0.030	0.423 ± 0.065	0.237 ± 0.035	0.663 ± 0.164	0.830 ± 0.019	0.082 ± 0.015
BSMOTE	0.797 ± 0.020	0.555 ± 0.029	0.846 ± 0.016	0.691 ± 0.107	0.973 ± 0.039	0.432 ± 0.081	0.237 ± 0.035	0.531 ± 0.146	0.857 ± 0.023	0.079 ± 0.013
OSS	0.796 ± 0.031	0.556 ± 0.021	0.875 ± 0.010	0.656 ± 0.076	0.966 ± 0.018	0.397 ± 0.062	0.287 ± 0.037	0.470 ± 0.092	0.881 ± 0.033	0.100 ± 0.011
ENN	0.812 ± 0.017	0.593 ± 0.014	0.881 ± 0.021	0.641 ± 0.070	0.966 ± 0.018	0.516 ± 0.025	0.174 ± 0.032	0.624 ± 0.100	0.854 ± 0.020	0.152 ± 0.014
TomekLink	0.796 ± 0.013	0.570 ± 0.019	0.871 ± 0.008	0.616 ± 0.073	0.966 ± 0.018	0.381 ± 0.058	0.265 ± 0.032	0.394 ± 0.051	0.872 ± 0.026	0.095 ± 0.009
SMOTEENN	0.829 ± 0.012	0.589 ± 0.017	0.857 ± 0.020	0.734 ± 0.083	0.952 ± 0.011	0.510 ± 0.048	0.257 ± 0.030	0.558 ± 0.088	0.894 ± 0.018	0.202 ± 0.016
SMOTETomek	0.823 ± 0.023	0.561 ± 0.038	0.850 ± 0.032	0.714 ± 0.106	0.987 ± 0.030	0.465 ± 0.049	0.219 ± 0.044	0.579 ± 0.148	0.846 ± 0.016	0.114 ± 0.021
SMOTEIPF	0.808 ± 0.026	0.556 ± 0.020	0.839 ± 0.031	0.737 ± 0.065	**0.996** ± 0.008	0.488 ± 0.056	0.200 ± 0.031	0.586 ± 0.215	0.863 ± 0.022	0.099 ± 0.027
SMOTERkNN	0.794 ± 0.028	0.584 ± 0.023	0.874 ± 0.020	0.653 ± 0.080	0.890 ± 0.046	0.537 ± 0.046	0.241 ± 0.055	0.545 ± 0.091	0.876 ± 0.013	0.184 ± 0.019
DE-OS	0.655 ± 0.048	0.559 ± 0.023	0.845 ± 0.035	0.724 ± 0.097	0.973 ± 0.027	0.438 ± 0.038	0.238 ± 0.033	0.447 ± 0.076	0.803 ± 0.042	0.113 ± 0.016
Easy	0.885 ± 0.017	0.598 ± 0.018	0.824 ± 0.022	0.724 ± 0.076	0.908 ± 0.028	0.529 ± 0.022	0.311 ± 0.026	0.559 ± 0.056	0.783 ± 0.020	0.248 ± 0.015
Cascade	0.913 ± 0.012	0.650 ± 0.023	**0.892** ± 0.013	0.888 ± 0.080	0.958 ± 0.019	0.575 ± 0.036	0.249 ± 0.035	0.672 ± 0.150	0.872 ± 0.017	0.203 ± 0.008
SPE	**0.938** ± 0.007	0.636 ± 0.020	0.882 ± 0.012	0.919 ± 0.042	0.982 ± 0.013	0.546 ± 0.040	0.308 ± 0.040	0.650 ± 0.058	**0.925** ± 0.021	0.255 ± 0.011
EASE	0.923 ± 0.018	0.652 ± 0.020	0.890 ± 0.019	0.895 ± 0.050	0.980 ± 0.020	0.580 ± 0.048	0.323 ± 0.036	0.725 ± 0.060	0.906 ± 0.020	0.234 ± 0.016
TSSE	0.916 ± 0.012	**0.663** ± 0.027	0.889 ± 0.017	0.912 ± 0.052	0.970 ± 0.028	0.543 ± 0.079	0.294 ± 0.060	0.762 ± 0.112	0.900 ± 0.019	0.235 ± 0.024
HUE	0.894 ± 0.011	0.615 ± 0.010	0.825 ± 0.012	0.716 ± 0.036	0.937 ± 0.019	0.528 ± 0.009	**0.352** ± 0.016	0.522 ± 0.043	0.779 ± 0.013	**0.269** ± 0.008
AHE	0.934 ± 0.011	0.653 ± 0.026	0.889 ± 0.016	**0.929** ± 0.079	0.985 ± 0.012	**0.588** ± 0.031	0.323 ± 0.025	**0.777** ± 0.078	0.922 ± 0.028	0.247 ± 0.015
Dataset	**solar_flare_m0**	**oil**	**car_eval_4**	**wine_quality**	**letter_img**	**yeast_me2**	**ozone_level**	**mammography**	**protein_homo**	**abalone_19**
SMOTE	0.133 ± 0.044	0.403 ± 0.075	0.964 ± 0.000	0.280 ± 0.054	0.913 ± 0.010	0.096 ± 0.084	0.174 ± 0.050	0.549 ± 0.015	0.552 ± 0.008	0.117 ± 0.037
ADASYN	0.094 ± 0.030	0.405 ± 0.098	0.964 ± 0.000	0.321 ± 0.036	0.915 ± 0.007	0.059 ± 0.071	0.200 ± 0.062	0.521 ± 0.026	0.576 ± 0.009	0.132 ± 0.029
BSMOTE	0.143 ± 0.025	0.325 ± 0.104	0.964 ± 0.000	0.332 ± 0.030	0.899 ± 0.009	0.104 ± 0.065	0.155 ± 0.041	0.551 ± 0.029	0.653 ± 0.008	0.073 ± 0.039
OSS	0.167 ± 0.032	0.504 ± 0.083	0.852 ± 0.052	0.373 ± 0.020	0.923 ± 0.009	0.176 ± 0.014	0.179 ± 0.079	0.602 ± 0.026	0.723 ± 0.007	0.034 ± 0.055
ENN	0.292 ± 0.005	**0.557** ± 0.025	0.840 ± 0.060	0.362 ± 0.022	0.927 ± 0.009	0.354 ± 0.041	0.231 ± 0.050	0.599 ± 0.010	0.722 ± 0.009	0.000 ± 0.000
TomekLink	0.167 ± 0.029	0.513 ± 0.083	0.840 ± 0.060	0.366 ± 0.016	0.921 ± 0.009	0.171 ± 0.004	0.211 ± 0.043	0.603 ± 0.019	0.724 ± 0.008	0.000 ± 0.000
SMOTEENN	0.290 ± 0.059	0.385 ± 0.095	0.918 ± 0.039	0.314 ± 0.040	0.910 ± 0.018	0.226 ± 0.043	0.219 ± 0.057	0.498 ± 0.020	0.555 ± 0.014	0.099 ± 0.021
SMOTETomek	0.133 ± 0.044	0.413 ± 0.065	0.964 ± 0.000	0.292 ± 0.050	0.913 ± 0.010	0.110 ± 0.077	0.172 ± 0.040	0.556 ± 0.022	0.552 ± 0.008	0.119 ± 0.037
SMOTEIPF	0.150 ± 0.036	0.486 ± 0.083	0.964 ± 0.000	0.305 ± 0.036	0.919 ± 0.017	0.105 ± 0.033	0.188 ± 0.042	0.551 ± 0.021	0.559 ± 0.011	**0.144** ± 0.052
SMOTERkNN	**0.352** ± 0.042	0.455 ± 0.027	0.847 ± 0.076	0.285 ± 0.048	0.895 ± 0.010	0.224 ± 0.074	0.184 ± 0.038	0.552 ± 0.017	0.588 ± 0.012	0.115 ± 0.040
DE-OS	0.218 ± 0.041	0.432 ± 0.074	0.964 ± 0.000	0.327 ± 0.034	0.837 ± 0.046	0.237 ± 0.056	0.228 ± 0.078	0.459 ± 0.022	0.441 ± 0.005	0.092 ± 0.021
Easy	0.344 ± 0.014	0.388 ± 0.026	0.783 ± 0.034	0.357 ± 0.026	0.783 ± 0.012	0.361 ± 0.037	0.305 ± 0.012	0.417 ± 0.013	0.415 ± 0.012	0.112 ± 0.016
Cascade	0.295 ± 0.039	0.415 ± 0.054	0.818 ± 0.032	0.350 ± 0.042	0.915 ± 0.016	0.382 ± 0.053	0.311 ± 0.058	0.551 ± 0.053	0.705 ± 0.017	0.115 ± 0.031
SPE	0.278 ± 0.033	0.469 ± 0.038	**0.971** ± 0.022	0.345 ± 0.030	0.955 ± 0.007	0.387 ± 0.042	0.309 ± 0.035	0.547 ± 0.014	0.682 ± 0.011	0.114 ± 0.014
EASE	0.297 ± 0.040	0.487 ± 0.050	0.949 ± 0.059	0.371 ± 0.023	0.953 ± 0.007	0.367 ± 0.113	0.313 ± 0.044	0.565 ± 0.031	0.703 ± 0.011	0.121 ± 0.020
TSSE	0.292 ± 0.033	0.491 ± 0.047	0.961 ± 0.077	0.382 ± 0.036	0.956 ± 0.012	**0.407** ± 0.102	0.316 ± 0.044	0.579 ± 0.033	0.786 ± 0.011	0.113 ± 0.027
HUE	0.324 ± 0.019	0.332 ± 0.005	0.769 ± 0.000	0.374 ± 0.008	0.768 ± 0.010	0.400 ± 0.014	0.295 ± 0.009	0.416 ± 0.007	0.422 ± 0.003	0.118 ± 0.000
AHE	0.289 ± 0.003	0.464 ± 0.078	0.958 ± 0.027	**0.412** ± 0.047	**0.960** ± 0.013	0.343 ± 0.097	**0.351** ± 0.050	**0.665** ± 0.028	**0.831** ± 0.011	0.104 ± 0.031

Table 18: MCC score of AHE and 17 baselines on 20 small-scale datasets. Best results are in bold.

Dataset	optical__digits	satimage	sick__euthyroid	spectrometer	car__eval__34	us__crime	scene	libras__move	thyroid__sick	coil__2000
SMOTE	0.803 ± 0.025	0.509 ± 0.040	0.820 ± 0.032	0.687 ± 0.126	0.986 ± 0.033	0.416 ± 0.053	0.140 ± 0.050	0.539 ± 0.168	0.836 ± 0.016	0.060 ± 0.023
ADASYN	0.789 ± 0.023	0.512 ± 0.043	0.813 ± 0.021	0.682 ± 0.115	0.983 ± 0.032	0.369 ± 0.070	0.155 ± 0.035	0.635 ± 0.180	0.819 ± 0.021	0.024 ± 0.014
BSMOTE	0.773 ± 0.022	0.503 ± 0.032	0.830 ± 0.018	0.664 ± 0.119	0.971 ± 0.042	0.380 ± 0.088	0.158 ± 0.038	0.488 ± 0.161	0.848 ± 0.024	0.021 ± 0.015
OSS	0.772 ± 0.036	0.508 ± 0.024	0.861 ± 0.011	0.630 ± 0.083	0.963 ± 0.019	0.355 ± 0.065	0.215 ± 0.042	0.421 ± 0.106	0.873 ± 0.035	0.038 ± 0.012
ENN	0.791 ± 0.019	0.542 ± 0.016	0.868 ± 0.023	0.617 ± 0.078	0.963 ± 0.019	0.470 ± 0.028	0.096 ± 0.032	0.593 ± 0.110	0.844 ± 0.021	0.075 ± 0.014
TomekLink	0.773 ± 0.015	0.523 ± 0.021	0.858 ± 0.009	0.591 ± 0.080	0.963 ± 0.019	0.337 ± 0.060	0.190 ± 0.036	0.347 ± 0.061	0.864 ± 0.028	0.033 ± 0.009
SMOTEENN	0.810 ± 0.014	0.532 ± 0.020	0.841 ± 0.022	0.708 ± 0.092	0.948 ± 0.011	0.454 ± 0.055	0.145 ± 0.036	0.519 ± 0.098	0.887 ± 0.019	0.127 ± 0.018
SMOTETomek	0.803 ± 0.025	0.510 ± 0.042	0.834 ± 0.035	0.684 ± 0.119	0.986 ± 0.033	0.416 ± 0.053	0.138 ± 0.049	0.539 ± 0.168	0.835 ± 0.017	0.057 ± 0.021
SMOTEIPF	0.786 ± 0.029	0.504 ± 0.022	0.823 ± 0.034	0.711 ± 0.074	**0.996** ± 0.009	0.437 ± 0.062	0.116 ± 0.036	0.544 ± 0.241	0.853 ± 0.023	0.041 ± 0.028
SMOTERkNN	0.770 ± 0.031	0.532 ± 0.026	0.860 ± 0.023	0.616 ± 0.092	0.880 ± 0.050	0.491 ± 0.051	0.158 ± 0.059	0.504 ± 0.100	0.867 ± 0.014	0.119 ± 0.022
DE-OS	0.605 ± 0.057	0.501 ± 0.026	0.828 ± 0.038	0.693 ± 0.110	0.971 ± 0.029	0.381 ± 0.040	0.146 ± 0.043	0.380 ± 0.092	0.789 ± 0.046	0.046 ± 0.017
Easy	0.872 ± 0.019	0.539 ± 0.021	0.804 ± 0.025	0.693 ± 0.088	0.900 ± 0.030	0.473 ± 0.026	0.204 ± 0.031	0.514 ± 0.065	0.767 ± 0.022	0.136 ± 0.019
Cascade	0.904 ± 0.013	0.609 ± 0.026	**0.880** ± 0.014	0.879 ± 0.086	0.954 ± 0.020	0.533 ± 0.040	0.162 ± 0.037	0.644 ± 0.166	0.863 ± 0.019	0.109 ± 0.010
SPE	**0.931** ± 0.008	0.589 ± 0.023	0.869 ± 0.013	0.911 ± 0.045	0.980 ± 0.014	0.496 ± 0.046	0.199 ± 0.048	0.618 ± 0.065	**0.920** ± 0.022	0.141 ± 0.014
EASE	0.914 ± 0.020	0.608 ± 0.024	0.878 ± 0.021	0.885 ± 0.053	0.978 ± 0.022	0.535 ± 0.054	0.227 ± 0.044	0.703 ± 0.067	0.899 ± 0.021	0.129 ± 0.017
TSSE	0.907 ± 0.013	**0.626** ± 0.031	0.878 ± 0.018	0.905 ± 0.056	0.968 ± 0.031	0.504 ± 0.083	0.206 ± 0.070	0.745 ± 0.121	0.893 ± 0.021	0.108 ± 0.031
HUE	0.881 ± 0.012	0.558 ± 0.011	0.805 ± 0.014	0.683 ± 0.042	0.931 ± 0.021	0.470 ± 0.010	**0.252** ± 0.019	0.471 ± 0.051	0.762 ± 0.014	**0.160** ± 0.009
AHE	0.926 ± 0.012	0.613 ± 0.029	0.878 ± 0.018	**0.922** ± 0.086	0.984 ± 0.013	**0.547** ± 0.034	0.222 ± 0.033	**0.761** ± 0.085	0.917 ± 0.030	0.142 ± 0.018

Dataset	solar__flare__m0	oil	car__eval__4	wine__quality	letter__img	yeast__me2	ozone__level	mammography	protein__homo	abalone__19
SMOTE	0.094 ± 0.047	0.379 ± 0.079	0.962 ± 0.000	0.240 ± 0.058	0.910 ± 0.010	0.063 ± 0.087	0.135 ± 0.053	0.536 ± 0.016	0.547 ± 0.008	0.102 ± 0.038
ADASYN	0.051 ± 0.031	0.377 ± 0.102	0.962 ± 0.000	0.285 ± 0.038	0.911 ± 0.007	0.021 ± 0.073	0.165 ± 0.065	0.507 ± 0.027	0.571 ± 0.010	0.117 ± 0.029
BSMOTE	0.100 ± 0.027	0.290 ± 0.108	0.962 ± 0.000	0.302 ± 0.031	0.895 ± 0.009	0.076 ± 0.068	0.120 ± 0.043	0.539 ± 0.029	0.650 ± 0.008	0.061 ± 0.039
OSS	0.126 ± 0.033	0.484 ± 0.087	0.847 ± 0.055	0.347 ± 0.021	0.920 ± 0.009	0.143 ± 0.016	0.149 ± 0.082	0.591 ± 0.027	0.720 ± 0.007	0.024 ± 0.055
ENN	0.246 ± 0.007	**0.535**± 0.028	0.834 ± 0.062	0.333 ± 0.023	0.924 ± 0.010	0.326 ± 0.043	0.203 ± 0.052	0.588 ± 0.011	0.719 ± 0.009	-0.011 ± 0.001
TomekLink	0.126 ± 0.030	0.491 ± 0.087	0.834 ± 0.062	0.340 ± 0.017	0.918 ± 0.009	0.138 ± 0.004	0.180 ± 0.044	0.593 ± 0.020	0.721 ± 0.008	-0.010 ± 0.001
SMOTEENN	0.237 ± 0.065	0.344 ± 0.105	0.915 ± 0.041	0.271 ± 0.044	0.906 ± 0.019	0.187 ± 0.045	0.177 ± 0.060	0.482 ± 0.021	0.550 ± 0.014	0.078 ± 0.021
SMOTETomek	0.094 ± 0.047	0.384 ± 0.064	0.962 ± 0.000	0.253 ± 0.053	0.910 ± 0.010	0.077 ± 0.079	0.133 ± 0.044	0.543 ± 0.022	0.547 ± 0.008	0.103 ± 0.038
SMOTEIPF	0.111 ± 0.038	0.463 ± 0.086	0.962 ± 0.000	0.266 ± 0.038	0.916 ± 0.017	0.070 ± 0.034	0.151 ± 0.045	0.538 ± 0.022	0.553 ± 0.012	**0.129** ± 0.054
SMOTERkNN	**0.310** ± 0.043	0.423 ± 0.030	0.841 ± 0.080	0.243 ± 0.052	0.891 ± 0.010	0.191 ± 0.076	0.145 ± 0.041	0.538 ± 0.017	0.583 ± 0.012	0.098 ± 0.041
DE-OS	0.172 ± 0.043	0.400 ± 0.078	0.962 ± 0.000	0.284 ± 0.037	0.830 ± 0.048	0.190 ± 0.059	0.181 ± 0.086	0.439 ± 0.023	0.432 ± 0.005	0.065 ± 0.023
Easy	0.270 ± 0.018	0.338 ± 0.031	0.773 ± 0.036	0.308 ± 0.030	0.773 ± 0.013	0.315 ± 0.041	0.259 ± 0.014	0.393 ± 0.014	0.406 ± 0.012	0.076 ± 0.018
Cascade	0.219 ± 0.048	0.379 ± 0.057	0.810 ± 0.033	0.308 ± 0.045	0.911 ± 0.017	0.349 ± 0.057	0.275 ± 0.061	0.536 ± 0.056	0.702 ± 0.017	0.080 ± 0.040
SPE	0.201 ± 0.038	0.432 ± 0.043	**0.970** ± 0.023	0.295 ± 0.034	0.953 ± 0.008	0.349 ± 0.046	0.267 ± 0.038	0.532 ± 0.014	0.679 ± 0.011	0.081 ± 0.016
EASE	0.237 ± 0.043	0.456 ± 0.052	0.947 ± 0.062	0.328 ± 0.026	0.952 ± 0.007	0.332 ± 0.118	0.275 ± 0.047	0.551 ± 0.032	0.700 ± 0.011	0.093 ± 0.023
TSSE	0.217 ± 0.040	0.464 ± 0.051	0.959 ± 0.081	0.346 ± 0.039	0.954 ± 0.012	**0.379** ± 0.106	0.282 ± 0.047	0.566 ± 0.035	0.784 ± 0.011	0.087 ± 0.030
HUE	0.250 ± 0.023	0.269 ± 0.007	0.758 ± 0.000	0.326 ± 0.009	0.758 ± 0.010	0.358 ± 0.016	0.247 ± 0.010	0.392 ± 0.007	0.414 ± 0.003	0.081 ± 0.001
AHE	0.225 ± 0.003	0.433 ± 0.085	0.956 ± 0.028	**0.385** ± 0.049	**0.958** ± 0.014	0.306 ± 0.102	**0.317**± 0.054	**0.657** ± 0.029	**0.829** ± 0.011	0.072 ± 0.034

Table 19: AUCPRC score of AHE and 17 baselines on 20 small-scale datasets. Best results are in bold.

Dataset	**optical_digits**	**satimage**	**sick_euthyroid**	**spectrometer**	**car_eval_34**	**us_crime**	**scene**	**libras_move**	**thyroid_sick**	**coil_2000**
SMOTE	0.693 ± 0.036	0.354 ± 0.038	0.716 ± 0.046	0.538 ± 0.146	0.976 ± 0.054	0.254 ± 0.042	0.104 ± 0.017	0.374 ± 0.170	0.725 ± 0.024	0.071 ± 0.006
ADASYN	0.672 ± 0.031	0.357 ± 0.039	0.706 ± 0.029	0.535 ± 0.128	0.973 ± 0.053	0.221 ± 0.048	0.109 ± 0.012	0.484 ± 0.221	0.699 ± 0.032	0.065 ± 0.002
BSMOTE	0.649 ± 0.030	0.349 ± 0.028	0.731 ± 0.026	0.518 ± 0.144	0.952 ± 0.069	0.231 ± 0.064	0.110 ± 0.014	0.324 ± 0.156	0.743 ± 0.038	0.064 ± 0.002
OSS	0.652 ± 0.050	0.353 ± 0.022	0.776 ± 0.018	0.472 ± 0.095	0.939 ± 0.031	0.210 ± 0.045	0.132 ± 0.018	0.257 ± 0.088	0.784 ± 0.058	0.067 ± 0.002
ENN	0.677 ± 0.025	0.383 ± 0.015	0.783 ± 0.035	0.458 ± 0.084	0.939 ± 0.031	0.297 ± 0.024	0.091 ± 0.008	0.418 ± 0.112	0.737 ± 0.033	0.071 ± 0.003
TomekLink	0.652 ± 0.020	0.367 ± 0.020	0.770 ± 0.015	0.428 ± 0.089	0.939 ± 0.031	0.198 ± 0.041	0.121 ± 0.015	0.199 ± 0.040	0.768 ± 0.046	0.066 ± 0.001
SMOTEENN	0.702 ± 0.019	0.370 ± 0.019	0.741 ± 0.034	0.562 ± 0.113	0.906 ± 0.020	0.282 ± 0.047	0.107 ± 0.012	0.341 ± 0.100	0.804 ± 0.031	0.084 ± 0.005
SMOTETomek	0.693 ± 0.036	0.355 ± 0.039	0.735 ± 0.050	0.533 ± 0.139	0.976 ± 0.054	0.254 ± 0.042	0.104 ± 0.016	0.374 ± 0.170	0.724 ± 0.026	0.071 ± 0.004
SMOTEIPF	0.670 ± 0.040	0.349 ± 0.020	0.720 ± 0.048	0.563 ± 0.094	0.993 ± 0.014	0.271 ± 0.048	0.097 ± 0.010	0.402 ± 0.267	0.752 ± 0.037	0.064 ± 0.004
SMOTERkNN	0.647 ± 0.043	0.373 ± 0.025	0.772 ± 0.034	0.453 ± 0.108	0.801 ± 0.076	0.316 ± 0.047	0.111 ± 0.024	0.327 ± 0.096	0.774 ± 0.022	0.081 ± 0.006
DE-OS	0.439 ± 0.062	0.345 ± 0.024	0.725 ± 0.056	0.541 ± 0.138	0.952 ± 0.049	0.227 ± 0.028	0.106 ± 0.014	0.220 ± 0.061	0.653 ± 0.067	0.066 ± 0.002
Easy	0.949 ± 0.010	0.630 ± 0.016	0.882 ± 0.013	0.868 ± 0.056	0.984 ± 0.019	0.491 ± 0.044	0.203 ± 0.031	0.789 ± 0.096	0.834 ± 0.035	0.109 ± 0.009
Cascade	0.956 ± 0.007	0.696 ± 0.019	0.892 ± 0.019	0.942 ± 0.054	0.979 ± 0.020	0.507 ± 0.041	0.195 ± 0.028	0.776 ± 0.101	0.952 ± 0.016	0.107 ± 0.009
SPE	0.973 ± 0.007	0.695 ± 0.017	**0.898** ± 0.018	0.960 ± 0.041	0.997 ± 0.003	0.490 ± 0.061	0.220 ± 0.027	0.840 ± 0.047	0.971 ± 0.009	0.111 ± 0.007
EASE	**0.982** ± 0.005	**0.752** ± 0.020	0.883 ± 0.018	**0.971** ± 0.021	**0.999** ± 0.002	0.584 ± 0.052	0.278 ± 0.040	0.852 ± 0.034	**0.977** ± 0.017	0.108 ± 0.007
TSSE	0.972 ± 0.007	0.742 ± 0.022	0.876 ± 0.018	0.956 ± 0.041	0.991 ± 0.012	0.530 ± 0.060	0.260 ± 0.071	0.821 ± 0.086	0.965 ± 0.008	0.107 ± 0.011
HUE	0.970 ± 0.005	0.737 ± 0.014	0.893 ± 0.010	0.938 ± 0.029	0.998 ± 0.003	**0.624** ± 0.029	**0.303** ± 0.022	**0.857** ± 0.015	0.950 ± 0.014	**0.119**± 0.003
AHE	0.969 ± 0.007	0.691 ± 0.033	0.893 ± 0.022	0.969 ± 0.036	**0.999** ± 0.002	0.556 ± 0.052	0.243 ± 0.024	0.841 ± 0.072	0.969 ± 0.018	0.111 ± 0.006
Dataset	**solar_flare_m0**	**oil**	**car_eval_4**	**wine_quality**	**letter_img**	**yeast_me2**	**ozone_level**	**mammography**	**protein_homo**	**abalone_19**
SMOTE	0.139 ± 0.024	0.199 ± 0.054	0.929 ± 0.000	0.104 ± 0.029	0.837 ± 0.018	0.049 ± 0.017	0.054 ± 0.014	0.310 ± 0.016	0.307 ± 0.009	0.021 ± 0.008
ADASYN	0.113 ± 0.011	0.202 ± 0.074	0.929 ± 0.000	0.126 ± 0.021	0.838 ± 0.012	0.042 ± 0.010	0.064 ± 0.023	0.280 ± 0.027	0.334 ± 0.011	0.024 ± 0.007
BSMOTE	0.122 ± 0.011	0.147 ± 0.072	0.929 ± 0.000	0.134 ± 0.018	0.812 ± 0.015	0.048 ± 0.013	0.049 ± 0.012	0.314 ± 0.030	0.429 ± 0.010	0.014 ± 0.003
OSS	0.124 ± 0.014	0.288 ± 0.088	0.738 ± 0.084	0.162 ± 0.014	0.855 ± 0.016	0.061 ± 0.005	0.061 ± 0.028	0.372 ± 0.031	0.525 ± 0.010	0.012 ± 0.006
ENN	0.165 ± 0.005	0.332 ± 0.028	0.718 ± 0.097	0.154 ± 0.015	0.862 ± 0.017	0.149 ± 0.027	0.077 ± 0.018	0.368 ± 0.012	0.523 ± 0.013	0.008 ± 0.000
TomekLink	0.120 ± 0.015	0.295 ± 0.083	0.718 ± 0.097	0.157 ± 0.011	0.852 ± 0.016	0.060 ± 0.001	0.068 ± 0.015	0.373 ± 0.023	0.526 ± 0.012	0.008 ± 0.000
SMOTEENN	0.117 ± 0.027	0.180 ± 0.068	0.845 ± 0.069	0.118 ± 0.022	0.830 ± 0.033	0.079 ± 0.017	0.069 ± 0.023	0.255 ± 0.019	0.310 ± 0.015	0.016 ± 0.004
SMOTETomek	0.139 ± 0.024	0.202 ± 0.049	0.929 ± 0.000	0.110 ± 0.026	0.837 ± 0.018	0.050 ± 0.016	0.053 ± 0.012	0.318 ± 0.023	0.307 ± 0.009	0.022 ± 0.008
SMOTEIPF	0.067 ± 0.008	0.269 ± 0.077	0.929 ± 0.000	0.115 ± 0.021	0.848 ± 0.030	0.045 ± 0.007	0.058 ± 0.013	0.311 ± 0.023	0.314 ± 0.013	0.029 ± 0.014
SMOTERkNN	0.154 ± 0.028	0.230 ± 0.025	0.724 ± 0.128	0.105 ± 0.026	0.805 ± 0.017	0.083 ± 0.030	0.056 ± 0.012	0.310 ± 0.018	0.347 ± 0.014	0.021 ± 0.009
DE-OS	0.138 ± 0.023	0.216 ± 0.061	0.929 ± 0.000	0.125 ± 0.021	0.704 ± 0.077	0.082 ± 0.022	0.073 ± 0.031	0.217 ± 0.019	0.195 ± 0.004	0.014 ± 0.004
Easy	**0.213** ± 0.020	0.391 ± 0.074	0.929 ± 0.095	0.268 ± 0.038	0.932 ± 0.019	0.191 ± 0.048	0.196 ± 0.041	0.420 ± 0.025	0.626 ± 0.028	0.028 ± 0.008
Cascade	0.208 ± 0.042	0.410 ± 0.056	0.956 ± 0.046	0.240 ± 0.038	0.978 ± 0.005	0.254 ± 0.072	0.204 ± 0.058	0.654 ± 0.046	0.833 ± 0.013	0.035 ± 0.026
SPE	0.165 ± 0.019	0.418 ± 0.063	0.992 ± 0.022	0.296 ± 0.039	0.983 ± 0.006	0.257 ± 0.032	0.179 ± 0.050	0.676 ± 0.040	0.844 ± 0.009	0.038 ± 0.011
EASE	0.180 ± 0.014	**0.509** ± 0.043	0.995 ± 0.011	0.378 ± 0.036	**0.990** ± 0.003	0.283 ± 0.057	0.250 ± 0.081	**0.739** ± 0.021	0.866 ± 0.009	**0.112** ± 0.079
TSSE	0.181 ± 0.042	0.500 ± 0.094	0.999 ± 0.002	0.347 ± 0.036	0.988 ± 0.005	**0.330** ± 0.053	0.220 ± 0.064	0.651 ± 0.042	**0.870** ± 0.011	0.070 ± 0.058
HUE	0.197 ± 0.010	0.427 ± 0.041	**1.000** ± 0.000	**0.428** ± 0.014	0.985 ± 0.002	0.244 ± 0.024	**0.267** ± 0.024	0.628 ± 0.030	0.865 ± 0.003	0.098 ± 0.024
AHE	0.169 ± 0.004	0.482 ± 0.099	0.998 ± 0.004	0.371 ± 0.043	0.987 ± 0.005	0.227 ± 0.051	0.202 ± 0.044	0.683 ± 0.032	0.852 ± 0.012	0.042 ± 0.020

CRediT authorship contribution statement

Xingjian Zeng: Writing - original draft, Software, Methodology, Data curation, Conceptualization. Yali Yuan: Methodology, Conceptualization, Supervision. Hantao Mei: Writing – review & editing, Software. Guang Cheng: Writing - review & editing, Validation, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported in part by Natural Science Foundation of Jiangsu Province (Grant No. SBK2023041256) and in part by the National Natural Science Foundation of China (Grant No. 62302097).

References

[1] S. Shi, K. Qiao, C. Chen, J. Yang, J. Chen, B. Yan, Over-sampling strategy in feature space for graphs based class-imbalanced bot detection, in: Companion Proceedings of the ACM on Web Conference 2024, 2024, pp. 738–741.

[2] Z. Ren, T. Lin, K. Feng, Y. Zhu, Z. Liu, K. Yan, A systematic review on imbalanced learning methods in intelligent fault diagnosis, IEEE Transactions on Instrumentation and Measurement 72 (2023) 1–35.

[3] S. Fotouhi, S. Asadi, M. W. Kattan, A comprehensive data level analysis for cancer diagnosis on imbalanced data, Journal of biomedical informatics 90 (2019) 103089.

[4] S. Susan, A. Kumar, The balancing trick: Optimized sampling of imbalanced datasets—a brief survey of the recent state of the art, Engineering Reports 3 (4) (2021) e12298.

[5] H. He, E. A. Garcia, Learning from imbalanced data, IEEE Transactions on knowledge and data engineering 21 (9) (2009) 1263–1284.

[6] P. Vuttipittayamongkol, E. Elyan, A. Petrovski, On the class overlap problem in imbalanced data classification, Knowledge-based systems 212 (2021) 106631.

[7] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, G. Bing, Learning from class-imbalanced data: Review of methods and applications, Expert systems with applications 73 (2017) 220–239.

[8] P. Vuttipittayamongkol, E. Elyan, A. Petrovski, On the class overlap problem in imbalanced data classification, Knowledge-based systems 212 (2021) 106631.

[9] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, Smote: synthetic minority over-sampling technique, Journal of artificial intelligence research 16 (2002) 321–357.

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

[11] H. Han, W.-Y. Wang, B.-H. Mao, Borderline-smote: a new over-sampling method in imbalanced data sets learning, in: International conference on intelligent computing, Springer, 2005, pp. 878–887.

[12] C. Bunkhumpornpat, K. Sinapiromsaran, 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: 13th Pacific-Asia conference, PAKDD 2009 Bangkok, Thailand, April 27-30, 2009 proceedings 13, Springer, 2009, pp. 475–482.

[13] C. Bunkhumpornpat, K. Sinapiromsaran, C. Lursinsap, Dbsmote: density-based synthetic minority over-sampling technique, Applied Intelligence 36 (2012) 664–684.

[14] D. L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Transactions on Systems, Man, and Cybernetics (3) (1972) 408–421.

[15] I. Tomek, Two modifications of cnn. (1976).

[16] M. Kubat, S. Matwin, et al., Addressing the curse of imbalanced training sets: one-sided selection, in: Icml, Vol. 97, Citeseer, 1997, p. 179.

[17] S.-J. Yen, Y.-S. Lee, Cluster-based under-sampling approaches for imbalanced data distributions, Expert Systems with Applications 36 (3) (2009) 5718–5727.

[18] A. Farshidvard, F. Hooshmand, S. MirHassani, A novel two-phase clustering-based under-sampling method for imbalanced classification problems, Expert Systems with Applications 213 (2023) 119003.

[19] C.-F. Tsai, W.-C. Lin, Y.-H. Hu, G.-T. Yao, Under-sampling class imbalanced datasets by combining clustering analysis and instance selection, Information Sciences 477 (2019) 47–54.

[20] J. Ha, J.-S. Lee, A new under-sampling method using genetic algorithm for imbalanced data classification, in: Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication, 2016, pp. 1–6.

[21] B. Sun, H. Chen, J. Wang, H. Xie, Evolutionary under-sampling based bagging ensemble method for imbalanced data classification, Frontiers of Computer Science 12 (2018) 331–350.

[22] C. Bunkhumpornpat, K. Sinapiromsaran, 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: 13th Pacific-Asia conference, PAKDD 2009 Bangkok, Thailand, April 27-30, 2009 proceedings 13, Springer, 2009, pp. 475–482.

[23] A. Fernández, S. Garcia, F. Herrera, N. V. Chawla, Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary, Journal of artificial intelligence research 61 (2018) 863–905.

[24] Z. Liu, W. Cao, Z. Gao, J. Bian, H. Chen, Y. Chang, T.-Y. Liu, Self-paced ensemble for highly imbalanced massive data classification, in: 2020 IEEE 36th international conference on data engineering (ICDE), IEEE, 2020, pp. 841–852.

[25] J. Ren, Y. Wang, M. Mao, Y.-m. Cheung, Equalization ensemble for large scale highly imbalanced data classification, Knowledge-Based Systems 242 (2022) 108295.

[26] L. Bai, T. Ju, H. Wang, M. Lei, X. Pan, Two-step ensemble under-sampling algorithm for massive imbalanced data classification, Information Sciences 665 (2024) 120351.

[27] G. E. Batista, R. C. Prati, M. C. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM SIGKDD explorations newsletter 6 (1) (2004) 20–29.

[28] G. E. Batista, A. L. Bazzan, M. C. Monard, et al., Balancing training data for automated annotation of keywords: a case study., Wob 3 (2003) 10–18.

[29] L. Chen, Z. Cai, L. Chen, Q. Gu, A novel differential evolution-clustering hybrid resampling algorithm on imbalanced datasets, in: 2010 third international conference on knowledge discovery and data mining, IEEE, 2010, pp. 81–85.

[30] J. A. Sáez, J. Luengo, J. Stefanowski, F. Herrera, Smote–ipf: Addressing the noisy and borderline examples problem in imbalanced classification by a resampling method with filtering, Information Sciences 291 (2015) 184–203.

[31] A. Zhang, H. Yu, Z. Huan, X. Yang, S. Zheng, S. Gao, Smote-rknn: A hybrid resampling method based on smote and reverse k-nearest neighbors, Information Sciences 595 (2022) 70–88.

[32] E. Ramentol, Y. Caballero, R. Bello, F. Herrera, Smote-rs b*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using smote and rough sets theory, Knowledge and information systems 33 (2012) 245–265.

[33] X.-Y. Liu, J. Wu, Z.-H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 39 (2) (2008) 539–550.

[34] X. Liang, Y. Gao, S. Xu, Ase: Anomaly scoring based ensemble learning for highly imbalanced datasets, Expert Systems with Applications 238 (2024) 122049.

[35] W. W. Ng, S. Xu, J. Zhang, X. Tian, T. Rong, S. Kwong, Hashing-based undersampling ensemble for imbalanced pattern classification problems, IEEE Transactions on Cybernetics 52 (2) (2020) 1269–1279.

[36] G. F. Jenks, The data model concept in statistical mapping, International yearbook of cartography 7 (1967) 186–190.

[37] A. Fernández, S. Garcia, F. Herrera, N. V. Chawla, Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary, Journal of artificial intelligence research 61 (2018) 863–905.

[38] D. Elreedy, A. F. Atiya, A comprehensive analysis of synthetic minority oversampling technique (smote) for handling class imbalance, Information Sciences 505 (2019) 32–64.

[39] P. Sadhukhan, S. Palit, Reverse-nearest neighborhood based oversampling for imbalanced, multi-label datasets, Pattern Recognition Letters 125 (2019) 813–820.

[40] N. V. Chawla, A. Lazarevic, L. O. Hall, K. W. Bowyer, Smoteboost: Improving prediction of the minority class in boosting, in: Knowledge Discovery in Databases: PKDD 2003: 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia, September 22-26, 2003. Proceedings 7, Springer, 2003, pp. 107–119.

[41] S. Wang, X. Yao, Diversity analysis on imbalanced data sets by using ensemble models, in: 2009 IEEE symposium on computational intelligence and data mining, IEEE, 2009, pp. 324–331.

[42] Y. Freund, R. E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, Journal of computer and system sciences 55 (1) (1997) 119–139.

[43] L. Breiman, Bagging predictors, Machine learning 24 (1996) 123–140.

[44] G. Douzas, F. Bacao, F. Last, Improving imbalanced learning through a heuristic oversampling method based on k-means and smote, Information sciences 465 (2018) 1–20.

[45] X. Tao, Y. Zheng, W. Chen, X. Zhang, L. Qi, Z. Fan, S. Huang, Svdd-based weighted oversampling technique for imbalanced and overlapped dataset learning, Information Sciences 588 (2022) 13–51.

[46] Y. Sun, L. Cai, B. Liao, W. Zhu, J. Xu, A robust oversampling approach for class imbalance problem with small disjuncts, IEEE Transactions on Knowledge and Data Engineering 35 (6) (2022) 5550–5562.

[47] Y. Wang, W. Gan, J. Yang, W. Wu, J. Yan, Dynamic curriculum learning for imbalanced data classification, in: Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 5017–5026.

[48] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al., A density-based algorithm for discovering clusters in large spatial databases with noise, in: kdd, Vol. 96, 1996, pp. 226–231.

[49] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, X. Xu, Dbscan revisited, revisited: why and how you should (still) use dbscan, ACM Transactions on Database Systems (TODS) 42 (3) (2017) 1–21.

[50] S. Jahirabadkar, P. Kulkarni, Algorithm to determine ε-distance parameter in density based clustering, Expert systems with applications 41 (6) (2014) 2939–2946.

[51] C.-C. Chang, C.-J. Lin, Libsvm: a library for support vector machines, ACM transactions on intelligent systems and technology (TIST) 2 (3) (2011) 1–27.

[52] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in python, the Journal of machine Learning research 12 (2011) 2825–2830.

[53] G. LemaÃŽtre, F. Nogueira, C. K. Aridas, Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning, Journal of machine learning research 18 (17) (2017) 1–5.

[54] J. Demšar, Statistical comparisons of classifiers over multiple data sets, The Journal of Machine learning research 7 (2006) 1–30.