Supervised Contrastive Learning with TPE-based Bayesian Optimization of Tabular Data for Imbalanced Learning

Shuting Tao, Peng Peng*, Hongwei Wang*

Zhejiang, China

Abstract

Class imbalance has a detrimental effect on the predictive performance of most supervised learning algorithms as the imbalanced distribution can lead to a bias preferring the majority class. To solve this problem, we propose a Supervised Contrastive Learning (SCL) method with Bayesian optimization technique based on Tree-structured Parzen Estimator (TPE) for imbalanced tabular datasets. Compared with supervised learning, contrastive learning can avoid “label bias” by extracting the information hidden in data. Based on contrastive loss, SCL can exploit the label information to address insufficient data augmentation of tabular data, and is thus used in the proposed SCL-TPE method to learn a discriminative representation of data. Additionally, as the hyper-parameter $\tau$ has a decisive influence on the SCL performance and is difficult to tune, TPE-based Bayesian optimization is introduced to automatically select the best $\tau$. Experiments are conducted on both binary and multi-class imbalanced tabular datasets. As shown in the results obtained, TPE outperforms other hyper-parameter optimization (HPO) methods such as grid search, random search, and genetic algorithm. More importantly, the proposed SCL-TPE method achieves much-improved performance compared with the state-of-the-art methods.

Keywords: Imbalanced learning, Supervised contrastive learning, Bayesian optimization, Representation learning, Deep learning

*Corresponding author

Email addresses: pengp17@mails.tsinghua.edu.cn (Peng Peng), hongweiwang@intl.zju.edu.cn (Hongwei Wang)
1. Introduction

With excellent performance on uniformly distributed data, supervised learning has become the most popular method for data classification. However, uneven distribution of data, i.e., class imbalance, is very common in most datasets collected from real-world scenarios, which inevitably undermines the effectiveness of supervised algorithm. This class imbalance makes it intractable for the supervised models to represent the distribution characteristics of skewed data correctly, and thus results in very low prediction accuracy for the minority classes. A well-known example is the mammography dataset [1], in which positive samples only account for 2.3% of the total samples. While the prediction accuracy of the positive class is crucial in this case, the traditional supervised learning-based classifiers tend to predict that all samples are negative.

Improving classification accuracy of both the majority and minority classes has become a great challenge. To address this challenge, considerable solutions have been put forward and they can be broadly divided into three categories: data preprocessing, [2, 3, 4, 5], feature learning [6, 7], and classifier design [8, 9, 10]. For the solutions based on data preprocessing, scholars attempt to rebalance data distribution through data sampling. In terms of classifier design, there are two kinds of methods. Specifically, the algorithm-level methods modify algorithms to increase the low accuracy of the minority class - the most popular one is cost-sensitive learning (CSL) that uses a weighted cost for different classes. Another kind is the model-level methods which combine the classification results from multiple base models like ensemble learning.

However, the methods mentioned above have some inherent drawbacks. For example, the data-level methods may result in losing useful information [11] or overestimating the minority data [12]. For CSL, it is difficult to set the value of misclassification cost which in most cases is unknown from the data and cannot be given by experts [13]. These issues have motivated researchers to develop strategies based on feature learning, and the existing methods consider using autoencoders to learn imbalanced data features [7]. In this paper, we propose to use supervised contrastive learning (SCL) [14] to extract features from imbalanced tabular datasets.

Contrastive learning (CL), a kind of self-supervised learning (SSL) [15], has been shown to represent the hidden features that are not conditioned on data labels well in the image domain [16, 17]. CL aims to group an anchor
and a “positive” sample together in the embedding space, and diverse the anchor far from “negative” samples. Here “positive” sample refers to data augmented from the anchor, while “negative” samples are randomly chosen from small batches. Despite the robustness achieved by CL in feature learning of images, techniques of augmenting “positive” samples are not applicable to general tabular data because they heavily rely on the unique structure of the domain datasets, such as spatial correlations in images [18]. The notions like rotation [19], colorization [20], and jigsaw puzzle solving [21] used in image data do not exist in tabular data, so the data augmentation methods applied to the tabular datasets we investigate are limited.

In this work, we fill this gap by adopting SCL to learn the representation of imbalanced tabular data. SCL considers many positives per anchor rather than using only a single positive. These positives are selected from samples belonging to the same class as the anchor, instead of from data augmentations of the anchor. Embeddings of the same class are pulled closer together than those from other classes. By introducing SCL, we solve the domain-specific augmentation problem of CL and extend the success of contrastive loss in the image domain to the tabular domain.

Furthermore, SCL requires fixing the hyper-parameter temperature $\tau$ before model training, which is a crucial hyper-parameter to control the strength of penalties on negative samples. Wang [22] shows a uniformity-tolerance dilemma in CL; a good choice of temperature can compromise the two properties and increase the feature quality significantly. That is to say, a good selection of $\tau$ can make the SCL achieve better performance in imbalanced learning. However, hyper-parameter tuning is often challenging and time-consuming. And among the current studies, little consideration has been provided to details of the hyper-parameter tuning of $\tau$. In this paper, we demonstrate that the setting of the $\tau$ has substantially influences on the performance of SCL. Also, we propose to develop a flexible approach that enables hyper-parameter optimization (HPO) to be conducted as an automatic process.

Classic HPO methods include grid search (GS), random search (RS), evolutionary algorithm, and Bayesian optimization (BO). Compared with uninformed methods like GS and RS, BO considers the previously explored information in each step, which reduces the search space and improves the search efficiency. BO also requires less computational resources than evolutionary algorithm such as genetic algorithm (GA) [23]. In GA, it needs to train the model on multiple hyper-parameters to go from one generation to
the next. In contrast, BO trains a single model and updates the posterior information, shortening training time and not requiring many computational resources. BO has two implementations: Gaussian Process (GP) and Tree-structured Parzen estimator (TPE) \[24\]. TPE has been proven superior to GP since the exploration induced by the TPE’s lack of accuracy turned out to be a good heuristic for search \[25\]. Therefore, we choose TPE to select the SCL model’s best $\tau$ in our work. More empirical work is shown in Section 4 to confirm our choice. More specifically, the main contributions of this paper are listed below:

- Supervised contrastive learning is proposed to learn an embedding space in which samples of the same class pairs stay close to each other while samples belonging to different classes are far apart. In imbalanced learning, we believe that SCL will outperform traditional supervised methods - the reason for this is that, in addition to employing the label information, SCL better captures data features by learning the intrinsic properties from the data itself based on contrastive loss. Therefore, SCL will not suffer from a significant performance drop due to the “label bias” caused by imbalanced data.

- Bayesian optimization based on Tree-structured Parzen Estimator (TPE) is firstly used to select the best hyper-parameter for SCL automatically. In this paper, we demonstrate that the hyper-parameter temperature $\tau$ is critical to model performance, and TPE is proved to produce better results than other algorithms for hyper-parameter optimization.

- Extensive experiments are conducted to demonstrate the effectiveness of our method. We compare SCL-TPE’s performance with ten competitive data sampling methods combined with five classifiers and SCL-TPE’s variants on fifteen diverse benchmark datasets that cover both binary and multi-class tasks.

The rest of this article is organized as follows. In Section 2 a brief review of previous research targeting the imbalanced learning problem is described. We also describe supervised contrastive learning and Bayesian optimization as the theoretical foundation of the proposed SCL-TPE method. Section 3 presents the proposed method in detail. Section 4 evaluates the proposed method by conducting experiments on some highly imbalanced datasets. Finally, the main conclusions of this work are drawn and discussed in Section 5.
2. Related work and background theory

Figure 1: The three methods of imbalanced learning.

2.1. Methods for imbalanced learning

This subsection briefly reviews the related work on imbalanced learning methods. As shown in Fig. 1, countermeasures to mitigating class imbalance issues can be divided into three categories: methods based on data preprocessing, methods based on feature learning, and methods based on classifier design.

2.1.1. Methods based on data preprocessing

Data-level methods conduct preprocessing on imbalanced datasets by modifying data distribution through sampling. These methods can be further divided into two sub-groups: undersampling and oversampling.

Undersampling methods rebalance data distribution by removing instances of the majority class. A non-heuristic approach is random undersampling (RUS) which randomly removes some majority class examples. However, such a manner may lead to the loss of important information in those removed examples. To alleviate this problem, more advanced undersampling techniques have been developed, such as Edited Nearest Neighbor (ENN)
ENN investigates k-nearest neighbors for each instance of the majority class. If most of the k neighbors belong to a different class, the instance will be removed. CNN is achieved by enumerating the dataset and adding them to the ‘group’ only if they cannot be classified correctly by the current contents. If there are two instances of different classes whose nearest neighbors are each other, they form a Tomek Link. In the Tomek Link approach, all instances in Tomek links that belong to the majority class are removed. OSS uses the Tomek Link method on top of CNN to eliminate noisy data and rebalance the data distribution.

Oversampling methods aim at increasing the size of the minority class by generating artificial minority instances. The simplest technique is random oversampling (ROS), which randomly replicates the minority class examples to rebalance the original imbalanced datasets. Due to generating duplicate data, ROS is prone to overfitting. In this regard, introduces Synthetic Minority Oversampling (SMOTE). SMOTE generates synthetic instances by interpolating between a minority class instance and its k nearest minority class neighbors. However, the main limitation of SMOTE is that each minority class example generates the same number of artificial samples without considering neighboring examples can come from different classes, which may result in overlap between classes. To accommodate this scenario, Borderline-SMOTE (BSMOTE) and Adaptive Synthetic Sampling (ADASYN) algorithms have been designed. BSMOTE only creates new data for minority instances closed to the border. ADASYN, on the other hand, leverages data distribution to determine the number of samples to be synthesized for each minority sample. Generative models such as generative adversarial networks (GAN) can also be used to generate synthetic minority class samples for data oversampling. This network synthesis-based approach is more complex than traditional techniques, but the generated samples are more diverse.

Data augmentation is also a method of data oversampling, which enhances the size and quality of training datasets by adding slightly modified copies of already existing data like geometric transformations. As mentioned in Section 1, most previous works on handling class imbalance with data augmentation are related to image datasets, but little progress has been made for the tabular datasets. It is insufficient to generate positive samples of contrastive learning only by data augmentation for tabular datasets. Therefore, we adopt the SCL which takes samples of the same class as positive samples.
2.1.2. Methods based on feature learning

Feature-learning based strategy attempts to preserve the key features of data to increase the discrimination between the minority class and the majority class. Using neural networks for feature extraction targeting imbalanced datasets has led to many in-depth studies. In these achievements, deep convolutional neural networks (DCNN) and deep autoencoders (DAE) are employed as the basic models. For example, [7] proposed the Dual Autoencoding Features (DAF), a feature learning method based on the stacked auto-encoder, to learn features with better classification capabilities of the minority and the majority classes.

In this work, we use the SCL method to learn the features of imbalanced data since SCL can utilize the rich implicit information from data as well as the information provided by labels. We will introduce the technical details in Section 2.2.

2.1.3. Methods based on classifier design

Classifier-design based methodology involves algorithm-level methods and model-level methods. Algorithm-level methods assign different weights for the majority and the minority classes and thus ease the optimization difficulty under imbalanced data [30, 31]. Model-level methods focus on constructing models that are less sensitive to imbalanced data. Among them, models built by ensemble approaches have become popular in imbalanced learning due to their better performance than a single learner [32], and the pure ensemble method is usually combined with algorithm-level method or data-preprocess based strategy. For instance, EasyEnsemble is an ensemble solution embedded with RUS.

2.2. Supervised contrastive learning

Many researchers have employed CL methods in previous studies to learn data representations by attracting positive pairs and pushing apart negative pairs. To optimize for this property, self-supervised contrastive learning has been proposed, which contrasts a single positive sample for each anchor against a set of negatives consisting of the entire remainder of the batch in the embedding space. The positive sample is an augmented version of the anchor.

Suppose there is a batch of $N$ samples with their labels, $\{x_k, y_k\}_{k=1,...,N}$. For each input sample $x$, two random augmentations are generated, so the augmented batch used for training comprises $2N$ pairs, $\{\tilde{x}_\ell, \tilde{y}_\ell\}_{\ell=1,...,2N}$, among
which $\tilde{x}_{2k}$ and $\tilde{x}_{2k-1}$ are augmentations of $x_k (k = 1 \ldots N)$ and $\tilde{y}_{2k-1} = \tilde{y}_{2k} = y_k$. In the augmented batch, we assume $i \in I \equiv \{1 \ldots 2N\}$ be the index of an arbitrary augmented sample and $j(i)$ be the index of the other augmented sample originating from the same source sample. The self-supervised contrastive loss [16] is given by:

$$L_{\text{self}} = \sum_{i \in I} L_{\text{self}}^i = -\sum_{i \in I} \log \frac{\exp\left(z_i \cdot z_{j(i)}/\tau\right)}{\sum_{a \in A(i)} \exp\left(z_i \cdot z_a/\tau\right)}$$

In Eq. (1), $z_\ell = \text{Enc}(\tilde{x}_\ell)$, $\text{Enc}(\cdot)$ is the representation vector of $\tilde{x}_\ell$, temperature $\tau \in \mathbb{R}^+$ is a scalar parameter, and $A(i) \equiv I \setminus \{i\}$. The index $i$ denotes the anchor, index $j(i)$ denotes the positive, and the other $2(N-1)$ indices ($k \in A(i) \setminus \{j(i)\}$) denote the negatives. Note that for each anchor $i$, the denominator has a total of $2N - 1$ terms consisting of one positive sample and $2N - 2$ negative samples.

In this study, we propose to utilize supervised contrastive learning (SCL), a generalization of self-supervised contrastive loss [14]. SCL leverages the label information and contrasts the set of all samples from the same class as positives against the negatives from other classes. The formulation of supervised contrastive loss is:

$$L_{\text{sup}} = \sum_{i \in I} \frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp\left(z_i \cdot z_p/\tau\right)}{\sum_{a \in A(i)} \exp\left(z_i \cdot z_a/\tau\right)}$$

where $P(i) \equiv \{p \in A(i) : \tilde{y}_p = \tilde{y}_i\}$ is the set of indices of all positives in the augmented batch distinct from $i$, and $|P(i)|$ is its cardinality.

2.3. Bayesian optimization for hyper-parameter tuning

It is well recognized that networks are easy to apply but difficult to train. The hyper-parameter tuning can be regarded as a "black art" requiring human experience, trial and error methods, and sometimes even violent search. Therefore, hyper-parameter optimization (HPO) emerges for three purposes: 1. Reduce labor costs; 2. Improve the performance of the model; 3. Help to find more reproducible parameter sets. There are four main methods for HPO: grid search (GS), random search (RS), evolutionary algorithm, and Bayesian Optimization (BO). GS and RS belong to uninformed search. Each step of the search does not learn anything from the previous experiments. Evolutionary algorithm is based on the concepts of biological evolution, which
considers a set of possible candidate solutions that evolves and gives a better result. BO is an informed search, which uses the performance of the previously searched parameters to speculate the best next step, thus reducing the search space and significantly improving the search efficiency.

Assume we want to optimize a function \( f : \mathcal{X} \rightarrow \mathbb{R} \) over a hyper-parameter set \( \mathcal{X} \). But \( f \) is costly to compute, making optimization difficult. The basic idea of Bayesian optimization is that we construct a probability distribution for \( f \). If we have computed \( f \) at hyper-parameter values \( x_1, x_2, \ldots, x_D \), then we consider \( f(x_1), f(x_2), \ldots, f(x_D) \) as observed information that can be exploited to choose the next \( x \) we want to evaluate.

Two choices should be made when performing Bayesian optimization. First, we must select a prior function as the assumptions of the function \( f \) being optimized. [24] proposed two strategies called Gaussian Process (GP) and Tree-structured Parzen Estimator (TPE), mentioning that the overall effect of TPE is better than that of GP. This study uses TPE to optimize the hyper-parameter temperature \( \tau \). Second, we must choose an acquisition function, which is used to construct a utility function from the posterior model to evaluate an expected loss associated with evaluating \( f \) at a hyper-parameter value \( x \). The value with the lowest expected loss will be selected. There are many forms of acquisition function, e.g., the probability of improvement, entropy search, and expected improvement (EI). This paper chooses EI as the criterion due to its excellent and intuitive performance [33]. The details of TPE and EI will be described in Section 3.

3. Methodology

3.1. The training of SCL classification model

In this paper, we propose an SCL with TPE-based Bayesian optimization for imbalanced datasets. From Fig. 2, we can see that the training procedure of the SCL classification model mainly consists of three steps. First, preprocess the original imbalanced dataset. For a multi-feature dataset, we normalize each data feature separately. Second, learn discriminative representations. In this representation learning stage, a data augmentation module is applied to transform each data sample randomly into two correlated instances. Common tabular data transformations, such as gaussian blur, and feature masks, are used. If the two augmented instances are from the same sample, they are regarded as positive pair. They are regarded as negative pair if transformed from different instances. Then we train the contrastive
network with the augmented dataset; the supervised contrastive loss helps us obtain a better feature embedding. Third, train the classifier. After acquiring the embedding of each class from the contrastive network, a three-layer softmax classifier will be trained for classification.

3.2. Determination of temperature $\tau$

In SCL, temperature $\tau \in \mathcal{R}^+$ is a scalar hyper-parameter that has a significant impact on the performance of the model. Different values of $\tau$ can considerably vary in results. In this work, we use TPE-based Bayesian optimization to select the best $\tau$.

Assume $x$ is the value of the hyper-parameter $\tau$. Our goal is to minimize a function $y = f(x)$. Here $y$ represents the negative of the model’s area under the receiver operating characteristic curve (AUC), which is an overall metric of the model’s performance. We will introduce AUC in Section 4.1.2. The smaller value of $y$ means the higher AUC and the better performance of the model. Bayesian optimization uses an iterative method called Sequential Model-Based Global Optimization (SMBO). The algorithm first builds a model $M$ as a surrogate for the function we want to optimize. Then in each iteration, the algorithm selects the local optimal hyper-parameter $x$ based on the current model $M_{t-1}$ according to the acquisition function, and fits a new model $M_t$ based on the updated search history.

Under the framework of SMBO, a modeling strategy called TPE is proposed. Instead of modeling $M$ as $p(y \mid x)$ directly, TPE uses Bayes’ theorem to decompose the $p(y \mid x)$ into $p(x \mid y)$ and $p(y)$. 

Figure 2: The general framework of the SCL classification model.
\[ p(y \mid x) = \frac{p(x \mid y)p(y)}{p(x)} \] (3)

\[ p(x \mid y) = \begin{cases} l(x), & y < y^* \\ g(x), & y > y^* \end{cases} \] (4)

As can be seen in Eq. (4), TPE constructs different \( p(x \mid y) \) on different sides of the threshold, where \( l(x) \) is the density formed by using the observations \( x \) such that corresponding loss \( f(x) \) is less than \( y^* \) and \( g(x) \) consists of the remaining observations. And the algorithm will set \( y^* \) to be some quantile \( \gamma \) of the observed \( y \) values, so that \( p(y < y^*) = \gamma \).

Acquisition Functions define a balance between exploring new areas in the objective space and exploiting areas already known to obtain favorable values, allowing us to decide the next \( x \) to evaluate. Expected improvement (EI) \[34\] is a popular choice of acquisition function defined as:

\[ EI_{y^*}(x) = \int_{-\infty}^{+\infty} \max(y^* - y, 0) p(y \mid x)dy. \] (5)

\( y^* \) here is a threshold. When \( x \) is given, EI is the expectation that \( f(x) \) will exceed (negatively) \( y^* \). Combining Eqs. (3), (4), (5), optimization of EI in the TPE algorithm is concluded to be:

\[ EI_{y^*}(x) = \frac{\gamma y^* \ell(x) - \ell(x) \int_{-\infty}^{y^*} p(y)dy}{\gamma \ell(x) + (1 - \gamma)g(x)} \propto \left( \frac{g(x)}{\ell(x)}(1 - \gamma) \right)^{-1} \] (6)

We choose the candidate \( x^* \) with the greatest EI as the value for the next iteration.

\[ x_{new} = \arg\max_x EI_{y^*}(x) \] (7)

3.3. The whole procedure of SCL-TPE

The whole process of the proposed method is shown in Algorithm 1. Each iteration in the outer loop corresponds to a search for the temperature \( \tau \). In iteration \( t \), we first uses the data augmentation module to transform the instances. Then we construct the feature extractor and implement the supervised contrastive loss to update the parameters of the extractor network. Subsequently, we train the softmax classifier and apply the trained classifier to test datasets to get the AUC. TPE adds the negative of AUC and the value of \( \tau \) to the history set \( \mathcal{H} \), and forms a probability distribution model \( M_t \) of \( \tau \) and the negative of AUC. In the next iteration \( t + 1 \), we choose the value of \( \tau \) according to \( M_t \) and \( \mathcal{H} \).
Algorithm 1: SCL-TPE for imbalanced datasets

**Input:** imbalanced training set \((x, y)\), imbalanced test set \((x_t, y_t)\),
maximum iterations \(T\), maximum epochs \(n\), batch size \(B\),
learning rate \(\eta\)

**Output:** Encoder network with learned parameters \(\theta_1^*\), Classifier
network with learned parameters \(\theta_2^*\), the best
hyper-parameter \(\tau\)

1. Initialize \(M_0, H \leftarrow \emptyset\);
2. for \(t = 1; t \leq T\) do /\ number of iterations
3. TPE chooses \(\tau\) depends on \(M_{t-1}\) and \(H\);
4. \(\text{Net}, \theta_1 \leftarrow \text{ContrastiveNet}()\);
5. for \(i = 1; i \leq n\) do /\ number of epochs
6. for \(b = 1; b \leq B\) do /\ number of batches
7. \(x_b, y_b \leftarrow \text{sampling} (x, y)\);
8. \(\tilde{x}_b, \tilde{y}_b \leftarrow \text{data_augmentation} (x_b, y_b)\);
9. \(\tilde{z}_b \leftarrow \text{forward}(\tilde{x}_b, \text{Net}, \theta_1, \tau)\);
10. \(\text{grad}_{\theta_1} \leftarrow \text{backward}(\tilde{x}_b, \tilde{z}_b, L_{sup}^b, \text{Net}, \theta_1)\);
11. \(\theta_1^* \leftarrow \text{update_NetParams}(\text{Net}, \theta_1, \text{grad}_{\theta_1}, \eta)\);
12. \(\theta_1 \leftarrow \theta_1^*\);
13. end
14. end
15. \(\text{Classifier} \leftarrow \text{construct_softmaxclassifier}()\);
16. for \(i = 1; i \leq n\) do /\ number of epochs
17. for \(b = 1; b \leq B\) do /\ number of batches
18. \(x_b, y_b \leftarrow \text{sampling} (x, y); x_b \leftarrow \text{Net} (x_b)\);
19. \(o_b \leftarrow \text{forward}(x_b, \text{Classifier}, \theta_2)\);
20. \(\text{grad}_{\theta_2} \leftarrow \text{backward}(y_b, o_b, L_{cross}^b, \text{Classifier}, \theta_2)\);
21. \(\theta_2^* \leftarrow \text{update_ClassifierParams}(\text{Classifier}, \theta_2, \text{grad}_{\theta_2}, \eta)\);
22. \(\theta_2 \leftarrow \theta_2^*\);
23. end
24. end
25. \(\text{AUC} \leftarrow \text{Evaluate} (\text{Classifier}(x_t), y_t)\);
26. \(H \leftarrow H \cup (\tau, -1 \times \text{AUC})\);
27. Fit a new model \(M_t\) to \(H\);
28. end
4. Experiment

4.1. Experimental setup and parameter setting

4.1.1. Datasets and baselines

In this section, two experiments are designed. The first one investigates the effectiveness of TPE. As the baseline, we introduce three HPO algorithms, random search (SCL-RS), grid search (SCL-GS), and genetic algorithm (SCL-GA). TPE and random search are performed using the hyperopt package for Python.

The second one explores whether the proposed SCL-TPE method can outperform other state-of-the-art algorithms. SCL-TPE is evaluated on eight binary and seven multi-class imbalanced datasets collected from the KEEL [35] and UCI [36] repositories. The detailed information of these fifteen datasets is given in Table 1. We compare the results of the proposed method against ten data sampling techniques, including random sampling, random undersampling, ENN, CNN, OSS, random oversampling, SMOTE, ADASYN, BSMOTE and GAN. Each sampling method is tested with five classification algorithms, including multilayer perceptron (MLP) with the same structure as the proposed model, support vector machine (SVM), K nearest neighbors (KNN), decision tree classifier (DTC), random forest classifier (RFC). The package imbalanced-learn [37] is utilized for the implementations of these benchmark undersampling and oversampling methods. In addition, we also conduct an ablation study of SCL, CL-TPE and SCL-TPE with the same structure.

4.1.2. Evaluation Metrics

We evaluate the model’s performance with four metrics: accuracy, F-score, G-mean, area under the receiver operating curve (AUC) [38]. Accuracy is a commonly used metric that summarizes the performance of a classification model as the proportion of correct predictions in the total number of predictions, but it is sensitive to data distributions. Accordingly, we supplement three other metrics to evaluate classifiers in skewed data fields. Single class metrics are calculated for each class and are less susceptible to class imbalance, so they are suitable for imbalanced data classification. For example, precision metric measures the correctly predicted positive class sample and is computed using Eq. (9), and recall quantifies the proportion of correctly identified of all actual positive samples defined by Eq. (10). In general, precision and recall share an inverse relationship. In order to seek a balance between them, F-measure is proposed as shown in Eq. (11). G-mean metric
Table 1: Description of 15 imbalanced datasets.

| Data sets     | Abbreviation | Size | Features | Class | Class Distribution | Data Repository |
|---------------|--------------|------|----------|-------|--------------------|-----------------|
| Glass0        | gl0          | 214  | 8        | 2     | 144/70             | KEEL            |
| Ecoli2        | eo2          | 336  | 7        | 2     | 284/52             | KEEL            |
| Yeast3        | yt3          | 1484 | 8        | 2     | 1321/163           | KEEL            |
| Yeast6        | yt6          | 1484 | 8        | 2     | 1449/35            | KEEL            |
| Vowel0        | vw0          | 988  | 13       | 2     | 90/898             | KEEL            |
| Haberman      | hb           | 306  | 3        | 2     | 225/81             | KEEL            |
| Yeast24       | yt24         | 514  | 8        | 2     | 463/51             | KEEL            |
| Pageblock0    | pa0          | 5472 | 10       | 2     | 4913/559           | KEEL            |
| Scale Balance | bal          | 625  | 4        | 3     | 49/288/288         | KEEL            |
| Wine          | wine         | 178  | 13       | 3     | 59/71/48           | KEEL            |
| lymphography  | lym          | 148  | 18       | 4     | 2/81/61/4          | KEEL            |
| Glass         | gla          | 214  | 9        | 6     | 70/76/17/13/9/29   | UCI             |
| Pageblocks    | page         | 548  | 10       | 5     | 492/33/3/8/12      | KEEL            |
| Dermatology   | dt           | 358  | 34       | 6     | 111/60/71/48/48/20 | KEEL            |
| Penbased      | pb           | 1100 | 16       | 10    | 115/114/114/106/114/106/105/115/105/106 | KEEL |

evaluates the degree of inductive bias between the accuracy of positive and negative classes.

\[
\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN} \tag{8}
\]

\[
\text{Precision} = \frac{TP}{TP + FN} \tag{9}
\]

\[
\text{Recall} = \frac{TP}{TP + FN} \tag{10}
\]

\[
\text{F - score} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Precision} + \text{Recall}} \tag{11}
\]

\[
\text{G - mean} = \sqrt{\frac{TP}{TP + FN}} \times \sqrt{\frac{TN}{TN + FP}} \tag{12}
\]

Besides, we use an overall metric called the area under the receiver operating characteristic curve(AUC). For imbalanced binary datasets, the ROC curve is plotted with TP against the FP where TP is on the y-axis and FP is on the x-axis. AUC metric converts this curve to a value, measuring the entire two-dimensional area underneath the ROC curve. For multi-class imbalanced problems, the MAUC metric averages the AUC value of all pairs of...
classes. This study calculates metrics for each label and find their unweighted mean. The equation is given as follows:

$$\text{MAUC} = \frac{1}{c(c-1)} \sum_{j=1}^{c} \sum_{k>j}^{c} (\text{AUC}(j \mid k) + \text{AUC}(k \mid j))$$

where $c$ is the number of classes and $\text{AUC}(j \mid k)$ is the AUC with class $j$ as the positive class and class $k$ as the negative class. In general, $\text{AUC}(j \mid k) \neq \text{AUC}(k \mid j)$ in the multiclass case. In our experiments, accuracy, F-measure, G-mean, and AUC are used together as assessment metric, in which accuracy and AUC are from sklearn.metrics [39], F-measure and G-mean are from imblearn.metrics[37]. For each metric, the greater the value, the better the performance.

4.1.3. Parameters setting

The detailed parameters for each model we construct are shown in Table 2. And the implementations of SCL are mainly based on PyTorch [40] and scikit-learn [39]. For all the datasets, the number of epochs of training contrastive network is 5000, the number of epochs of training linear classifier is 25. Adam is adopted as an optimizer, and the learning rate is 0.001. The number of TPE iterations is fixed at 75.

Table 2: Parameters for Supervised Contrastive Learning with TPE-based Bayesian optimization for imbalanced datasets.

| Data sets       | # of neurons in each layer of extractor | # of neurons in each layer of linear classifier | Batch size | $\tau$ chosen by TPE |
|-----------------|----------------------------------------|-----------------------------------------------|------------|----------------------|
| Glass0          | (9, 96, 48)                            | (48, 24, 2)                                   | 160        | 0.514                |
| Ecoli2          | (7, 96, 48)                            | (48, 10, 2)                                   | 128        | 0.489                |
| Yeast3          | (8, 128, 64)                           | (64, 32, 2)                                   | 240        | 0.857                |
| Yeast6          | (8, 96, 48)                            | (48, 24, 2)                                   | 320        | 0.947                |
| Vowel0          | (13, 104, 52)                          | (52, 26, 2)                                   | 160        | 0.010                |
| Haberman        | (3, 96, 48)                            | (48, 24, 2)                                   | 128        | 0.153                |
| Yeast24         | (8, 128, 64)                           | (64, 32, 2)                                   | 128        | 0.245                |
| Pageblock0      | (10, 128, 64)                          | (64, 32, 2)                                   | 160        | 0.348                |
| Scale Balance   | (4, 128, 64)                           | (64, 32, 3)                                   | 128        | 0.995                |
| Wine            | (13, 200, 100)                         | (100, 50, 3)                                  | 150        | 0.055                |
| lymphography    | (18, 128, 64)                          | (62, 32, 4)                                   | 150        | 0.854                |
| Glass           | (9, 128, 64)                           | (64, 32, 6)                                   | 128        | 0.352                |
| Pageblocks      | (10, 128, 64)                          | (64, 32, 5)                                   | 128        | 0.122                |
| Dermatology     | (34, 128, 64)                          | (64, 32, 6)                                   | 128        | 0.116                |
| Penbased        | (16, 128, 64)                          | (64, 32, 10)                                  | 160        | 0.059                |
4.2. Experimental results and analysis

4.2.1. Discussion of TPE

In this section, we demonstrate that different values of hyper-parameter $\tau$ can lead to fluctuations in the model’s performance. Bayesian optimization is also proved to be more effective and efficient in selecting a promising hyper-parameter than other HPO like grid search, random search, and genetic algorithm. We take one binary dataset and one multiclass dataset, Glass and Glass0, as examples. Fig. 3 suggests that $\tau$ affects the quality of data embedding. For dataset Glass0, we observe that embeddings of $\tau = 0.05$ present a more reasonable locally clustered and globally separated distribution, while the embeddings trained with $\tau = 0.6$ are chaotic. This phenomenon can also be seen in the dataset Glass. We also evaluate the performances of classification results with different $\tau$ on Glass0, Glass. Tables 3 and 4 show the performance of classification results on the two datasets, respectively. We can see different $\tau$ values will lead to significant differences in final performance.
Figure 3: The t-SNE embeddings of Glass0 and Glass.

Table 3: Performance comparison of models trained with different temperature on Glass0.

| Metrics | $\tau$ | 0.03 | 0.07 | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 1.0 |
|---------|-------|------|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Accuracy |       | 0.721 | 0.625 | 0.674 | 0.674 | 0.674 | 0.674 | 0.676 | 0.697 | 0.744 | 0.628 | 0.628 | 0.674 |
| F-measure |     | 0.73 | 0.61 | 0.68 | 0.68 | 0.68 | 0.66 | 0.77 | 0.59 | 0.75 | 0.64 | 0.64 | 0.54 |
| G-mean |       | 0.74 | 0.48 | 0.71 | 0.71 | 0.69 | 0.73 | 0.73 | 0.77 | 0.67 | 0.67 | 0.00 |     |
| AUC    |       | 0.738 | 0.539 | 0.722 | 0.722 | 0.722 | 0.704 | 0.735 | 0.536 | 0.773 | 0.687 | 0.687 | 0.500 |

Table 4: Performance comparison of models trained with different temperature on Glass.

| Metrics | $\tau$ | 0.03 | 0.07 | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 1.0 |
|---------|-------|------|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Accuracy |       | 0.462 | 0.385 | 0.400 | 0.569 | 0.215 | 0.323 | 0.185 | 0.538 | 0.262 | 0.262 | 0.308 | 0.400 |
| F-measure |     | 0.38 | 0.28 | 0.28 | 0.53 | 0.21 | 0.33 | 0.19 | 0.41 | 0.21 | 0.19 | 0.21 | 0.30 |
| G-mean |       | 0.39 | 0.39 | 0.33 | 0.61 | 0.33 | 0.48 | 0.29 | 0.37 | 0.30 | 0.33 | 0.27 | 0.40 |
| AUC    |       | 0.662 | 0.697 | 0.725 | 0.752 | 0.650 | 0.653 | 0.593 | 0.665 | 0.585 | 0.585 | 0.728 | 0.707 |
Machine learning algorithms such as SCL are rarely parameter-free. Therefore, these hyper-parameters must be optimized to improve the prediction ability of the algorithm. Grid search (GS), random search (RS), and genetic algorithm (GA) are three commonly used HPO strategies. GS defines a search space as a grid of hyper-parameter values and assesses every position in the grid. RS defines a search space as a bounded domain of hyper-parameter values and randomly sample points in that domain. GA is inspired by natural selection, which detects well-performing hyper-parameters in each generation and passes them to the next generation until the best-performing hyper-parameter is identified.

This paper adopts the TPE-based Bayesian optimization approach, and the HPO methods mentioned above are introduced as the baseline. The primary purpose of the experiment is to evaluate the performance of the TPE. The RS and TPE optimizers are iterative processes with the number of iterations set to 75 so as to achieve a good balance between performance and complexity. In the GS, we changed the value of $\tau$ from 0 to 1 in increments of 0.02. In the GA, we set the number of iterations as 5 and the population size as 15. The population size here determines the number of trial solutions in each iteration. Fig. 4 shows the boxplot of the AUC of different optimizers over two datasets, where the X-axis denotes the adopted methods, and the Y-axis represents the AUC. The median results are shown as the red line in the figure. We can see that TPE-based Bayesian optimization’s optimal and medium results greatly exceed those of RS, GS, and GA. The effectiveness of the TPE-based Bayesian optimization is thus proved.

Figure 4: Boxplot of accuracy over four hyper-parameter optimization approach.

To test the efficiency of the TPE, we compared the performances and
running times of these two methods with 75 iterations each. The population size of GA is set to 10. For Glass0, the results obtained by TPE and GA are the same. The running time of TPE is 4695s, and of GA is 23211s. For Glass, the accuracy of GA is 61.5% which is 3.1% lower than TPE; the F1 score is 54% which is 7.0% lower than TPE; the G-mean is 60% which is 9.0% lower than TPE; the AUC is 78.0% which is 0.2% higher than TPE. The running time of TPE is 5369s, and of GA is 26776s. We can see that GA results are slightly worse than TPE and take more time. As we mentioned in Section 1, that’s because, in each iteration, GA needs to train the model on multiple trial solutions.

4.2.2. SCL-TPE vs. state-of-the-art methods

The following experiment is organized in the way below. We consider two cases: binary class imbalanced data classification and multi-class imbalanced data classification. For each case, we compare SCL-TPE with other competitive methods to validate the superiority of the proposed method. The data are divided into two parts during the experiment, including the training and test datasets. We calculate and report the results based on the test part. In imbalanced binary classification, the experimental results in Tables 5 and 6 present an overwhelming improvement of the proposed method over its competitors. In particular, our approach provides the best performance on all eight datasets when considering G-mean and AUC as performance measures. If we take the dataset Glass0 as an example, the proposed method yields an accuracy result of 0.837, which is 2.4% better than the second best method (BSMOTE with MLP), F-measure of 0.84 which is 2% higher than the second best method, G-mean of 0.84 which is 3% higher than the second best method, and AUC 0.842 which is 3.6% superior to the second best method. Regarding accuracy and F1 score, SCL-TPE achieves the optimal value on the five datasets. We found the results are not as good as other methods on the remaining three datasets because other methods tend to classify minority samples into the majority class. For example, in the Haberman dataset, OSS with MLP classifies all samples as the majority. However, in reality, minority groups are usually more important and need to be accurately identified.

The proposed method is also compared with ten sampling methods tested on five base classifiers in the multi-class classification task. From Tables 7 and 8, we can see the proposed method outperforms other approaches on all four metrics. For example, on the Wine dataset, the proposed method yields an accuracy result of 0.972, which is 2.8% better than the second best
method (GAN with SVM); F-measure of 0.97, which is 3% higher than the second best method; G-mean of 0.98, which is 2% higher than the second-best method; and AUC of 0.981, which is 1.8% superior to the second best method. It is worth noting that there are only one or two samples of the minority class in the test dataset of the lymphography, glass, and page block data sets, resulting in the failure of SMOTE and ADASYN. The excellent performance of SCL-TPE in these datasets also proves the robustness of our method.
Table 6: G-mean and AUC for binary imbalanced data.

| Data     | G-mean | None | RUS | ENN | CNN | OSS | ROS | SMO | ADA | BSM | GAN |
|----------|--------|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| MLP      | 0.78   | 0.99 | 0.99 | 0.78 | 0.78 | 0.99 |     |     |     |     |     |
| SVM      | 0.84   | 0.95 |     |     |     |     |     |     |     |     |     |
| KNN      | 1.00   | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |     |     |     |     |     |
| DTC      | 0.94   | 0.94 | 0.94 | 0.94 | 0.94 | 0.94 |     |     |     |     |     |
| RFC      | 0.88   | 0.88 | 0.88 | 0.88 | 0.88 | 0.88 |     |     |     |     |     |
| SCL-TPE  | 0.93   | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 |     |     |     |     |     |

| AUC     | None | RUS | ENN | CNN | OSS | ROS | SMO | ADA | BSM | GAN |
|---------|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| MLP     | 0.78 | 0.99 | 0.99 | 0.78 | 0.78 | 0.99 |     |     |     |     |     |
| SVM     | 0.84 | 0.95 |     |     |     |     |     |     |     |     |     |
| KNN     | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |     |     |     |     |     |
| DTC     | 0.94 | 0.94 | 0.94 | 0.94 | 0.94 | 0.94 |     |     |     |     |     |
| RFC     | 0.88 | 0.88 | 0.88 | 0.88 | 0.88 | 0.88 |     |     |     |     |     |
| SCL-TPE | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 |     |     |     |     |     |

4.2.3. Ablation study

In addition to the advanced resampling methods, we conduct an ablation study to analyze the performance gain of each component in SCL-TPE on the tabular datasets. We define two variants of SCL-TPE: 1. SCL only. We exclude the TPE-based Bayesian optimization and fix temperature $\tau$ as 0.5. 2. CL-TPE. We exclude the supervised contrastive loss and use contrastive loss instead.

Ablations are provided in Table 6. From experimental results, SCL-TPE
always performs better than its variants. We conclude that each of the major elements of our method is valuable and crucial for performance, and the best performance is achieved when they work collaboratively.

Finally, the t-SNE embeddings of two binary datasets and two multiclass datasets are shown in Fig. 5. The discriminative and compact representation learned by SCL is very helpful for the downstream imbalanced dataset classification.

To ensure that the final performance of SCL-TPE is effective, Fig. 6 reports the confusion matrices of the proposed method on these four datasets, where the rows and columns represent prediction classes and actual classes, respectively. The color of each grid represents a specific value, which means

| Data     | Accuracy | F-measure |
|----------|----------|-----------|
| MLP      | 0.944    | 0.912     |
| SVM      | 0.888    | 0.876     |
| KNN      | 0.776    | 0.752     |
| DTC      | 0.776    | 0.756     |
| RFC      | 0.789    | 0.760     |

Table 7: Accuracy and F-measure for multi-class imbalanced data.
Table 8: G-mean and AUC score for multi-class imbalanced data.

| Data | G-mean | AUC |
|------|--------|-----|
| None | RUS | ENN | CNN | OSS | ROS | SMO | ADA | BSM | GAN |
| MLP | 0.94 | 0.96 | 0.90 | 0.98 | 0.94 | 0.99 | 0.98 | 0.93 | 0.99 | 0.96 |
| SVM | 0.85 | 0.84 | 0.86 | 0.88 | 0.84 | 0.93 | 0.94 | 0.94 | 0.94 | 0.85 |
| KNN | 0.79 | 0.85 | 0.84 | 0.86 | 0.80 | 0.78 | 0.79 | 0.80 | 0.80 | 0.80 |
| DTC | 0.84 | 0.79 | 0.83 | 0.74 | 0.84 | 0.74 | 0.79 | 0.83 | 0.83 | 0.84 |
| RFC | 0.78 | 0.82 | 0.80 | 0.82 | 0.77 | 0.73 | 0.80 | 0.79 | 0.78 | 0.80 |
| SCL-TPE | 1.0 | | | | | | | | | |

| Data | G-mean | AUC |
|------|--------|-----|
| None | RUS | ENN | CNN | OSS | ROS | SMO | ADA | BSM | GAN |
| MLP | 0.88 | 0.50 | 0.62 | 0.87 | 0.68 | 0.68 | 0.53 | 0.63 | 0.63 | 0.50 |
| SVM | 0.96 | 0.94 | 0.96 | 0.00 | 0.53 | 0.96 | 0.96 | 0.94 | 0.96 | 0.96 |
| KNN | 0.94 | 0.91 | 0.94 | 0.50 | 0.86 | 0.94 | 0.91 | 0.89 | 0.94 | 0.94 |
| DTC | 0.76 | 0.76 | 0.83 | 0.53 | 0.82 | 0.81 | 0.76 | 0.74 | 0.71 | 0.85 |
| RFC | 0.92 | 0.84 | 0.89 | 0.89 | 0.91 | 0.96 | 0.91 | 0.87 | 0.91 | 0.87 |
| SCL-TPE | 0.98 | | | | | | | | | |

Table 9: Ablation study on fifteen datasets.

| Metrics | Methods | gbl | e2o | ykt | ytb | ved | nb | ytb2 | ptd | bat | wme | lym | gls | gpa | dt | pb |
|---------|---------|-----|-----|-----|-----|-----|----|------|-----|-----|-----|-----|-----|-----|----|----|
| Accuracy | SCL | 0.78 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 |
| SCL-TPE | 0.74 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 |
| SCL-20.00 | 0.83 | 0.56 | 0.56 | 0.56 | 0.56 | 0.56 | 0.56 | 0.56 | 0.56 | 0.56 | 0.56 | 0.56 | 0.56 | 0.56 | 0.56 | 0.56 |
| F-measure | SCL | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 | 0.77 |
| SCL-TPE | 0.74 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 |
| SCL-20.00 | 0.84 | 0.66 | 0.66 | 0.66 | 0.66 | 0.66 | 0.66 | 0.66 | 0.66 | 0.66 | 0.66 | 0.66 | 0.66 | 0.66 | 0.66 | 0.66 |
| G-mean | SCL | 0.77 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 |
| SCL-TPE | 0.84 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 |
| SCL-20.00 | 0.84 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 |
| AUC | SCL | 0.73 | 0.82 | 0.82 | 0.82 | 0.82 | 0.82 | 0.82 | 0.82 | 0.82 | 0.82 | 0.82 | 0.82 | 0.82 | 0.82 | 0.82 |
| SCL-TPE | 0.82 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 |
| SCL-20.00 | 0.82 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 | 0.90 |

the ratio of the number of correctly predicted samples to the total number of actual samples. The darker the color of diagonal blocks and the lighter the
Figure 5: The t-SNE embeddings of yeast3, vowel0, lymphography, glass.

Figure 6: The confusion matrices of yeast3, vowel0, lymphography, glass.

color of other blocks, the better the effect of the model.
5. Conclusions and future work

In this study, we propose a novel SCL with TPE-based Bayesian optimization. In representation learning, SCL not only leverages the label information but also learns the pattern information hidden in the data based on contrastive loss. We reveal the significant influence of the hyper-parameter $\tau$ on the model performance and demonstrate that TPE surpasses grid search, random search, and genetic algorithm in selecting the best $\tau$. Experimental results on binary datasets and multi-class datasets prove that the proposed method obtains better performance than other methods in terms of four metrics, namely Accuracy, F-measure, G-mean, and Area Under the ROC (AUC).

In the future, we plan to explore more data augmentation techniques for tabular datasets and combine them with self-supervised contrastive learning methods. Additionally, a comprehensive comparison will be carried out based on the intention to combine SCL with other newly proposed hyper-parameter optimization methods such as Heteroscedastic and Evolutionary Bayesian Optimisation solver (HEBO) [42].

6. Acknowledgments

This work was supported by the National Key R&D Program of China under Grant No. 2020YFB1707803. This work was also supported in part by the Zhejiang University/University of Illinois at Urbana-Champaign Institute, and was led by Principal Supervisor Prof. Hongwei Wang.
References

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

[2] 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.

[3] I. Mani, I. Zhang, KNN approach to unbalanced data distributions: a case study involving information extraction, in: Proceedings of workshop on learning from imbalanced datasets, volume 126, ICML United States.

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

[5] 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, pp. 1322–1328.

[6] T. Zhang, J. Chen, F. Li, K. Zhang, H. Lv, S. He, E. Xu, Intelligent fault diagnosis of machines with small & imbalanced data: A state-of-the-art review and possible extensions, ISA transactions 119 (2022) 152–171.

[7] W. W. Ng, G. Zeng, J. Zhang, D. S. Yeung, W. Pedrycz, Dual autoencoders features for imbalance classification problem, Pattern Recognition 60 (2016) 875–889.

[8] C. Elkan, The foundations of cost-sensitive learning, in: International joint conference on artificial intelligence, volume 17, Lawrence Erlbaum Associates Ltd, pp. 973–978.

[9] A. Shrivastava, A. Gupta, R. Girshick, Training region-based object detectors with online hard example mining, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 761–769.

[10] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE international conference on computer vision, pp. 2980–2988.
[11] 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 (2008) 539–550.

[12] M. Liu, M. Dong, C. Jing, A modified real-value negative selection detector-based oversampling approach for multiclass imbalance problems, Information Sciences 556 (2021) 160–176.

[13] 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.

[14] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, D. Krishnan, Supervised contrastive learning, Advances in Neural Information Processing Systems 33 (2020) 18661–18673.

[15] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, J. Tang, Self-supervised learning: Generative or contrastive, IEEE Transactions on Knowledge and Data Engineering (2021).

[16] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, in: International conference on machine learning, PMLR, pp. 1597–1607.

[17] V. Sehwag, M. Chiang, P. Mittal, Ssd: A unified framework for self-supervised outlier detection, international conference on learning representations (2021).

[18] J. Yoon, Y. Zhang, J. Jordan, M. van der Schaar, Vime: Extending the success of self-and semi-supervised learning to tabular domain, Advances in Neural Information Processing Systems 33 (2020) 11033–11043.

[19] S. Gidaris, P. Singh, N. Komodakis, Unsupervised representation learning by predicting image rotations, in: International Conference on Learning Representations.

[20] R. Zhang, P. Isola, A. A. Efros, Colorful image colorization, in: European conference on computer vision, Springer, pp. 649–666.
[21] M. Noroozi, P. Favaro, Unsupervised learning of visual representations by solving jigsaw puzzles, in: European conference on computer vision, Springer, pp. 69–84.

[22] F. Wang, H. Liu, Understanding the behaviour of contrastive loss, computer vision and pattern recognition (2021).

[23] D. Whitley, A genetic algorithm tutorial, Statistics and computing 4 (1994) 65–85.

[24] J. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, Algorithms for hyperparameter optimization, Advances in neural information processing systems 24 (2011).

[25] C. Thornton, F. Hutter, H. H. Hoos, K. Leyton-Brown, Auto-weka: Combined selection and hyperparameter optimization of classification algorithms, in: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 847–855.

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

[27] P. Hart, The condensed nearest neighbor rule (corresp.), IEEE transactions on information theory 14 (1968) 515–516.

[28] I. Tomek, Two modifications of cnn, systems man and cybernetics (1976).

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

[30] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, S. Belongie, Class-balanced loss based on effective number of samples, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 9268–9277.

[31] K. Tang, J. Huang, H. Zhang, Long-tailed classification by keeping the good and removing the bad momentum causal effect, Advances in Neural Information Processing Systems 33 (2020) 1513–1524.
[32] J. Tanha, Y. Abdi, N. Samadi, N. Razzaghi, M. Asadpour, Boosting methods for multi-class imbalanced data classification: an experimental review, Journal of Big Data (2020).

[33] J. Snoek, H. Larochelle, R. P. Adams, Practical bayesian optimization of machine learning algorithms, Advances in neural information processing systems 25 (2012).

[34] D. R. Jones, A taxonomy of global optimization methods based on response surfaces, Journal of global optimization 21 (2001) 345–383.

[35] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework., Journal of Multiple-Valued Logic & Soft Computing 17 (2011).

[36] A. Asuncion, D. Newman, Uci machine learning repository, 2007.

[37] G. Lemaître, 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 (2017) 1–5.

[38] A. P. Bradley, The use of the area under the roc curve in the evaluation of machine learning algorithms, Pattern recognition 30 (1997) 1145–1159.

[39] 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.

[40] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., Pytorch: An imperative style, high-performance deep learning library, Advances in neural information processing systems 32 (2019).

[41] L. Van der Maaten, G. Hinton, Visualizing data using t-sne., Journal of machine learning research 9 (2008).

[42] A. I. Cowen-Rivers, W. Lyu, R. Tutunov, Z. Wang, A. Grosnit, R. R. Griffiths, A. M. Maraval, H. Jianye, J. Wang, J. Peters, et al., An
empirical study of assumptions in bayesian optimisation, arXiv preprint arXiv:2012.03826 (2020).