# A PSO-optimized Oversampling Method for Imbalance Classification


1st Xi-Bin Dong
*Computer Science and Engineering*
*South China University of Technology*
Guangzhou, China
201721041390@mail.scut.edu.cn

2st Xian-Bing Meng
*Computer Science and Engineering*
*South China University of Technology*
Guangzhou, China
axbmeng@gmail.com

3st Zhi-Wen Yu
*Computer Science and Engineering*
*South China University of Technology*
Guangzhou, China
zhwyu@scut.edu.cn

4st Philip Chen
*Computer Science and Engineering*
*South China University of Technology*
Guangzhou, China
philipchen@scut.edu.cn

5st Guo-Qiang Han
*Computer Science and Engineering*
*South China University of Technology*
Guangzhou, China
csgqhan@scut.edu.cn



***Abstract*—Traditional oversampling methods have been proven to effectively solve imbalance classification. However, how to approximate the original data distribution as much as possible after oversampling remains to be solved. In this paper, an oversampling method optimized by particle swarm optimization (PSO) is proposed for imbalance classification. As same as the traditional oversampling method, synthetic samples are first generated from the minority classes. As a distinctive feature, the synthetic samples will not directly insert into the minority classes, but would be further evaluated by decision tree classifier. Through integration with PSO, the synthetic samples with best fitness value can then be used to expand the amount of minority classes. As a result, the imbalance ratio can be significantly decreased. Simulations and comparisons based on 17 datasets demonstrate the effectiveness and superiority of the proposed method.**



***Index Terms*—imbalance classification, oversampling method, particle swarm optimization**


## I. Introduction

Traditional machine learning methods are mostly based on the assumption that the data distribution is balanced. However, when facing with imbalanced class distribution, the traditional machine learning methods cannot often achieve satisfactory performance. Imbalance learning methods have been proposed and achieve satisfactory performance on imbalanced data [1]. As one of the hot spots in imbalance learning, oversampling method has been proven to be effective, but it often faces the problem of destroying distribution of the original data. For example, synthetic minority oversampling technique(SMOTE) [2] generates samples of minority classes via linear interpolation in the sample space, however, the sample space of majority class is often invaded by the newly generated samples. The intrusive sample will also affect the subsequent data process [3], which will affect the classification performance. To address the aforementioned problem, a PSO-optimized oversampling method is proposed to optimize the synthetic samples to expand the amount of minority classes and approximate the original data distribution as much as possible. Specifically, SMOTE is used to generate the synthetic samples. Then, an evolutionary algorithm, PSO [4], is integrated into decision tree classifier to select the best synthetic samples to expand the amount of minority classes. To investigate the effectiveness of proposed method, mainstream oversampling methods and hybrid sampling methods are used as comparison methods. Experiments based on 17 imbalanced datasets demonstrate the effectiveness of proposed method.

The rest of the paper is organized as follows. The related work is presented in section 2. Section 3 illustrates the proposed method and its computational complexity. Simulations will be conducted in section 4. Conclusion will be finally drawn.


The work described in this paper was partially funded by grants with Key-Area Research and Development Program of Guangdong Province No. 2018B010107002


## II. Related Works

This section reviews related works on sampling methods of imbalance learning and PSO algorithm.

Oversampling methods such as ADASYN [5] generates different numbers of minority samples according to their distribution. SMOTE [2] randomly selects a sample from its nearest neighbor, and interpolates between them to construct minority samples. Thus, the problem of over-fitting on majority class can be mitigated by improving the imbalance ratio. SMOTE assigns equal importance to all the samples of the minority class, however, in the actual modeling process, it is noticed that samples at the boundary are more likely to be misclassified. In this setting, Borderline SMOTE [6], which combines the SMOTE with the information about the boundary samples is proposed. Experiments showed that using the information about samples located in the boundary to generate new samples can improve the model performance. Recent


978-1-7281-6929-3/20/$31.00 ©2020 IEEE

works on oversampling such as Rastogi tried to bring SMOTE to distributed computing environments under Spark for large datasets tasks [7].Gosain proposed FSMOTE [8] which generates samples by Interpolating between minority samples and its k minority class farthest neighbors.Binghao modified SMOTE and presented Mean-SMOTE [9] for non p2p traffic classification.Hamdy utilized SMOTE to predict fine-grained bug severity levels [10].Wu proposed an efficient imbalance learning algorithm called Easy-SMT [11], which combines SMOTE-based oversampling policy with EasyEnsemble to divide imbalnace problem into balanced learning subproblem.

Undersampling methods [12]–[17] usually eliminate noisy or redundant samples of majority classes to improve imbalance ratio. Recent work such as Han applied Gaussian mixture model to for undersampling [18]. However, the removal of samples in the majority classes usually results in information loss. Besides, neighborhood information based algorithms are sensitive to noisy samples which produce negative effects on performance.

To solve the limitation of oversampling and undersampling, in [3], the combination of Tomek Links [19] and ENN [20] with SMOTE are proposed. More specifically, Tomek Links cleans expanded dataset by deleting the sample whose nearest neighbor is different from its own category. In Smote + ENN, ENN is used to predict each sample's label in the expanded dataset. When the prediction is inconsistent with its actual label, the sample will be removed. However, we notice that the sample invasion, which will destroy local structure information of majority class samples and effect the undersampling process, still exist in hybrid sampling methods. So there needs an optimization for the generated samples after oversampling to refine the data distribution.

The PSO algorithm [4] is inspired from the behavioral characteristics of biological populations and is used to solve optimization problems. In the PSO algorithm, the potential solution of each optimization problem can be regarded as a particle in the search space. All particles have a fitness value determined by the objective function, and each particle also has a speed that determines the direction and distance of their movement. The particles follow the current optimal particles to get better fitness value in the solution space. There is already relevant work on the application of PSO algorithm to imbalance learning, In [21] and [22], PSO is used for sample subset selection and feature selection. Recently, Hu used PSO to find optimal weight of Weighted Extreme Learning Machine(WELM) parameters [23], experiment showed PSO strategy can improve generalization and performance of WELM on imbalanced datasets.

In our method, we use PSO algorithm to optimize the sample distribution after SMOTE, aiming to introduce minority class samples to improve imbalance ratio while protecting the original distribution of the data.

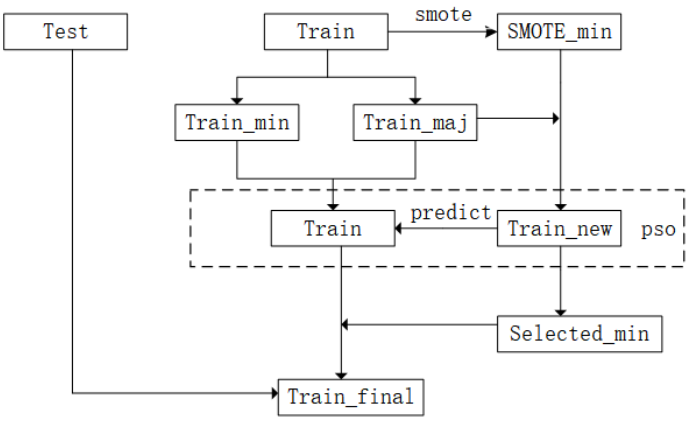



Fig. 1. The overview framework of PSO-optimized oversampling method.

## III. PSO-OPTIMIZED OVERSAMPLING METHOD

### A. Problem Formulation

There is a binary classification task whose dataset $X$ follows an imbalanced distribution, where $X = \{x_1, x_2, \ldots, x_n\} \in \mathbb{R}^{d \times n}$, and $d$ and $n$ denote the numbers of features and samples. $X_{min}$ and $X_{maj}$ denote the sets of samples from the minority and the majority classes, respectively. Our goal is to use PSO to perform selection on samples $X_{smote}$ generated by SMOTE to get optimal samples $X'_{smote}$. The advantage of using PSO is that PSO can regard $X_{smote}$ as a particle and use evolutionary process to optimize the particle globally, which is different from neighborhood information based sample selection methods.

### B. Proposed Approach

The framework of proposed method is represented in Fig.1, in which SMOTE_min represents minority samples generated by SMOTE, Train_min represents minority class samples of training dataset, Train_maj represents majority class samples of training dataset, Train_new represents the combination of Train_maj and SMOTE_min, Selected_min represents minority samples in SMOTE_min selected by PSO, Train_final represents the combination of Train and Selected_min. The pseudo code of PSMOTE is presented in Algorithm 1. Firstly, we adopt SMOTE to conduct oversampling. Given an imbalanced dataset $T$, for each minority class sample $x_i$ in $T$,we calculate the Euclidean distance between $x_i$ and other minority samples and obtain the k-nearest neighbors of $x_i$(in our method, $k$ is 5). Then, we set the sampling rate $r$ based on the imbalance ratio of dataset. Next, for each minority sample $x_i$ of $T$, a number of samples are randomly selected from k-nearest neighbors of $x_i$ according to $r$. For each neighbor $x_n$ of $x_i$, we construct synthetic minority class sample $x_{new}$ as follow:

$$x_{new} = x_i + rand(0,1) * |x_i - x_n|, \tag{1}$$

where $rand(0,1)$ is a function for producing a random number falling into the interval between 0 and 1. Finally, we get a synthetic minority sample set $T_{min\_smote}$.

In PSMOTE, PSO is used to optimize the generated minority class samples. Considering some of the datasets with high dimension and large instance number, we choose PSO with global best topology for its advantage of convergence[29]. Specifically, a particle set $p_{set} \in \mathbb{R}^{p_{num} \times f_{size}}$ is initialized randomly, where $p_{num}$ is the number of particles and $f_{size}$ is the number of minority samples generated by SMOTE. For $(x, y)$ of $p_{set}$, a random variable $r$ is generated to determine the value of $(x, y)$ as follow:

$$p_{set}(x,y) = \begin{cases} 1, & \text{if } r > 0.3 \\ 0, & \text{else} \end{cases}. \tag{2}$$

$p_{set}(x, y) = 1$ indicates that the $y^{th}$ sample of the $x^{th}$ particle is selected.

we initialize the velocity set $v \in \mathbb{R}^{p_{num} \times f_{size}}$, the local best record $l_{best} \in \mathbb{R}^{p_{num} \times f_{size}}$ and the global best record $g_{best} \in \mathbb{R}^{p_{num} \times f_{size}}$ by setting them to 0. Then in the iteration process, we firstly select generated minority class samples according to each particle, and decision tree classifiers will be trained on the dataset formed by selected samples and majority samples from original dataset, auc [25] on original dataset is stored as particles' socre. Then, we update the local best record $l_{best}$ and global best record $g_{best}$ as follow:

$$l_{best}^{i} = \begin{cases} p_i, & \text{if } score(l_{best}^{i}) < score(p_i) \\ l_{best}^{i}, & \text{otherwise} \end{cases} \tag{3}$$

$$g_{best} = \begin{cases} p_i, & \text{if } score(g_{best}) < score(p_i) \\ g_{best}, & \text{otherwise} \end{cases} \tag{4}$$

Next, we utilize $l_{best}$ and $g_{best}$ to update the velocity of particles:

$$v_i = wv_i + c_1 r_1 (l_{best}^{i} - p_i) + c_2 r_2 (g_{best} - p_i), \tag{5}$$

where $w$ is the inertial factor. $c_1$ and $c_2$ are the acceleration constants. $r_1$ and $r_2$ are random variables varying between 0 and 1. $l_{best}^{i}$ and $g_{best}$ are the local best position of particle $i$ and the global best position of all particles, respectively.

$$v_{ij} = \begin{cases} v_{max}, & \text{if } v_{ij} > v_{max} \\ v_{min}, & \text{if } v_{ij} < v_{min} \\ v_{ij}, & \text{otherwise} \end{cases} \tag{6}$$

where $v_{ij}$ is the $j^{th}$ direction of $v_i$.

After getting $v_i$, we update the position of each particle $p_i$ using discrete PSO position update formula(7) from [29], in which $t$ is a random number in (0,1). Mathematically, $v_{ij}$ determines a threshold of the probability that $p_{ij}$ = 1, which means the j-th sample in particle $p_i$ should be kept in the distribution.

$$p_{ij} = \begin{cases} 0, & \text{if } t >= \frac{1}{1+e^{-v_{ij}}} \\ 1, & \text{otherwise} \end{cases} \tag{7}$$

The evolutionary process mentioned above will repeat until reaching given iterations. Considering the method is a single classification framework, and higher auc means better performance on the original dataset, so we select samples according to the global best particle to construct dataset with better imbalance ratio. Finally, we construct classifier and test.

**Algorithm 1** PSMOTE

**Require:**
The training dataset $T$ , sampling ratio $r$, the number $p_{num}$ of particles, the iteration number $it_{num}$

**Ensure:**
initialize $T_{smote\_min}$ to empty set. initialize particles and parameters in PSO.
**for** each minority class sample $x_i$ in $T$ **do**
get $x_i$'s k-nearest neighbors $K$ using Euclidean distance
choose samples in $K$ according to sampling ratio $r$ to form $S$
for each sample $x_n$ in in $S$, generate sample $x_{smote}$ using equation(1), add $x_{smote}$ to $T_{smote\_min}$
**end for**
split $T$ to get major class samples $T_{maj}$ and minority class samples $T_{min}$
combine $T_{smote\_min}$ with $T_{maj}$ to get $T_{new}$
**for** $i = 1$ to $it_{num}$ **do**
**for** $j = 1$ to $p_{num}$ **do**
project $T_{new}$ according to particle $p_j$ in $p_{set}$ to get $T_{new\_selected}$
train decision tree classifier using $T_{new\_selected}$ and do validation on $T$, store the auc score as particles fitness score
update local best record $l_{best}^{j}$ of particle $p_j$ and global best record $g_{best}$
**end for**
adjust particles' velocity and position
**end for**
select generated minority class sample according to global best particle to form $selected\_min$
use $T$ and $selected\_min$ to form $T_{final}$ and train classifier, then predict test data label

### C. Complexity Analysis

We perform a theoretical analysis of PSMOTE concerning the computational cost. The time complexity can be computed by:

$$T_{psmote} = T_{smote} + T_{pso}, \tag{8}$$

$T_{smote}$ is affected by the number $t_m$ of minority class samples in training dataset, and the number $k$ of neighbours used to generate new samples, which is defined as follows:

$$T_{smote} = \mathcal{O}(t_m^2 + t_m * k). \tag{9}$$

$T_{pso}$ is affected by the generated minority class sample number $t_g$, the iteration number $i$, the particle set size $s$, the computation cost for generating a classifier $T_c$,and the cost $T_l$ for particle learning process, which is defined as follows:

$$T_{pso} = \mathcal{O}(i * (s * T_c + T_l)), \tag{10}$$

TABLE I
THE DESCRIPTIONS ABOUT THE IMBALANCED DATASETS

| Dataset | Repository | Target | Ratio | S | F |
|---|---|---|---|---|---|
| spectrometer | UCI | $\geq 44$ | 11:1 | 531 | 93 |
| car_eval_34 | UCI | good, v good | 12:1 | 1728 | 21 |
| us_crime | UCI | $\geq 0.65$ | 12:1 | 1994 | 100 |
| yeast_ml8 | LIBSVM | 8 | 13:1 | 2417 | 103 |
| libras_move | UCI | 1 | 14:1 | 360 | 90 |
| solar_flare_m0 | UCI | M->0 | 19:1 | 1389 | 32 |
| ozone_level | UCI | ozone, data | 34:1 | 2536 | 72 |
| oil | UCI | minority | 22:1 | 937 | 49 |
| ecoil | UCI | imU | 8.6:1 | 336 | 7 |
| yeast_me2 | UCI | ME2 | 28:1 | 1484 | 8 |
| arrhythmia | UCI | 06 | 17:1 | 452 | 278 |
| abalone | UCI | 7 | 9.7:1 | 4177 | 10 |
| sick_euthyroid | UCI | sick euthyroid | 9.8:1 | 3163 | 42 |
| thyroid_sick | UCI | sick | 15:1 | 3772 | 52 |
| wine_quality | UCI | $\leq 4$ | 26:1 | 4898 | 11 |
| ablone_19 | UCI | 19 | 130:1 | 4177 | 10 |
| optical_digits | UCI | 8 | 9.1:1 | 5620 | 64 |

TABLE II
THE PARAMETER OF PARTICLE SWARM OPTIMIZATION (PSO)

| Parameter | Default Value |
|---|---|
| particle number | 50 |
| iteration number | 1000 |
| w | 0.7 |
| c1 | 1.5 |
| c2 | 1.5 |
| vMax | 0.98 |
| vMin | 0.02 |

where $T_c = \mathcal{O}(t * d)$, $d$ is the depth of decision tree and t is the sample number used for training. $T_l = \mathcal{O}(s * t_m)$. The computational cost of PSMOTE is approximately $\mathcal{O}(t^2)$

## IV. EXPERIMENTS

In this section, we conduct experiments to evaluate the effectiveness of PSMOTE on 17 real-world imbalanced datasets. Table I shows the statistical information about these datasets [24]. $S$ and $F$ denote the numbers of data samples and attributes. The imbalance ratio denoted by "Ratio" is also contained.

### A. Evaluation Criterion

The purpose of this paper is to optimize the oversampling method, which is mainly considering the impact of the sample level. For fair comparison, we use Decision Tree classifier for all sampling methods. The evaluation criterion is AUC-ROC (we call it AUC brifely). AUC is a commonly used metric in imbalance learning because it's not affected by imbalance ratio. Usually, higher AUC means better distinguishing ability of algorithm for different class, more details of AUC can refer to [25]. 5 cross-validation is taken to get average AUC of all methods.

### B. Experiment Analysis

*1) Experiment of hyper parameters:* We perform hyper parameters experiment on the combination of sampling ratio and particle number of PSO. The ratio gradually increased from 0.2 to 1 at a growth rate of 0.1, and the particle number increased from 50 to 250. The hyper parameters combinations corresponding to the best results of each dataset were given in tableIII (for example, 0.4/0.7408 represents we get 0.7408 auc with 0.4 sampling ratio). It can be seen that as the particle number increases, the performance of the classifier increases firstly, and then gradually decreases. The reason behind this may be that as the number of particles in the PSO algorithm increases, the population has a greater probability to converge to a better solution. However, the number of particles must be consistent with the complexity to be optimized, and excessive number of particles will affect the optimization of the problem.

*2) SMOTE vs PSMOTE:* We compare our method with SMOTE to prove the effectiveness of PSO strategy. TableIV shows our method has achieved better performance than SMOTE method on 15 out of the 17 dataset. We can see most of the better performance are gained by first setting the PSMOTE with a sampling ratio that is equal to or greater than SMOTE best sampling ratio, and then using PSO to select samples. This shows that compared with directly setting SMOTE sampling ratio, the proposed method can adaptively obtain fewer samples that are beneficial for classification performance, effectively increasing the upper limit of SMOTE. We also give data distribution visualization in Fig.2(we use PCA to reduce attributes to two dimensions). Respectively, figures from left to right are original data distribution, after SMOTE processing, after PSMOTE processing. we can see that our methods can improve the classification results with fewer generated samples, and our method can get closer distribution to original data distribution than SMOTE.

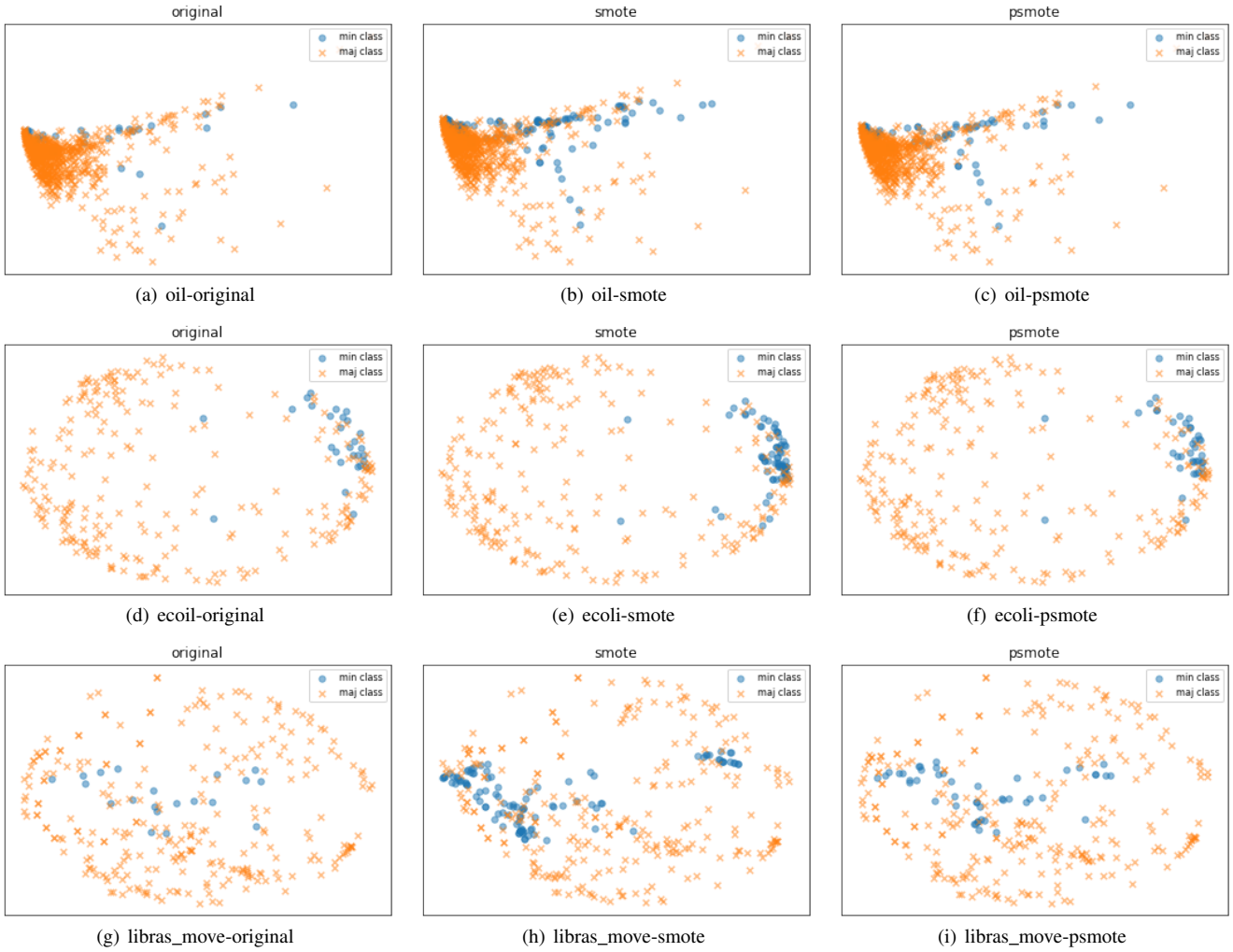
*3) PSMOTE vs other oversampling methods:* For the mainstream oversampling methods such as Rand Oversampling [26], ADASYN [5], BorderLine SMOTE [6], SVM SMOTE [27], we adjust their sampling ratio from 0.2 to 1 at a growth rate of 0.1 and obtain their best AUC. As is shown in tableIV(we list the methods in their abbreviations, / means without sampling strategy), it can be seen that our proposed method performs better than the mainstream oversampling method, which shows that it's necessary to conduct further optimization after oversampling process.

*4) PSMOTE +ENN vs SMOTE + ENN:* As for the hybird sampling methods, tableV shows that PSMOTE + ENN is better than SMOTE + ENN. The reason may be that PSO can delete intrusive samples and optimize data distribution after oversampling, which helps using the local structure information of samples to improve classification performance.

### C. Statistical Tests

*1) Average Ranking:* We conduct the average ranking of PSMOTE based on the experiment result of tableIV. The values in tableVI denote the performance ranks on 17 datasets. We can see that our methods achieve better average rank than other mainstream oversampling methods.

*2) Non-Parametric Test:* Wilcoxon signed-rank test [28] is adopted to determine the significance of our method. TableVII shows the test result. We can see that with the threshold $\alpha = 0.05$, the p values give the conclusion that PSMOTE rejects the




(a) oil-original (b) oil-smote (c) oil-psmote

(d) ecoil-original (e) ecoli-smote (f) ecoli-psmote

(g) libras_move-original (h) libras_move-smote (i) libras_move-psmote

Fig. 2. data distribution visualization

TABLE III
HYPER PARAMETERS EXPERIMENT
RESULTS REPRESENT SAMPLING RATIO/AUC

| **Datasets** | **Particle numbers in PSO** | | | | |
|---|---|---|---|---|---|
| | ***50*** | ***100*** | ***150*** | ***200*** | ***250*** |
| oil | 0.4/0.7408 | 0.4/0.7563 | 0.5/0.7452 | **0.4/0.7646** | 0.6/0.7446 |
| ecoli | 0.7/0.79696 | 0.4/0.8079 | 0.4/0.7936 | **0.7/0.7986** | 0.3/0.7860 |
| car_eval_34 | 0.6/0.9713 | 0.3/0.9716 | 0.2/0.9716 | 0.2/0.9712 | **0.3/0.9754** |
| us_crime | 0.4/0.7086 | **0.7/0.7310** | 0.5/0.7212 | 0.9/0.7115 | 0.3/0.7014 |
| solar_flare_m0 | 0.2/0.6449 | **0.2/0.6517** | 0.2/0.6253 | 0.2/0.6443 | 0.2/0.6480 |
| yeast_me2 | 0.9/0.7105 | 0.9/0.7408 | **0.9/0.7498** | 0.8/0.7249 | 0.9/0.7222 |
| spectrometer | 0.6/0.8696 | 0.7/0.8866 | 0.9/0.8825 | 0.7/0.8685 | **0.7/0.8877** |
| libras_move | 0.7/0.8690 | 0.3/0.8815 | **0.3/0.8845** | 0.3/0.8595 | 0.3/0.8815 |
| arrhythmia | 0.6/0.8506 | 0.8/0.8636 | **0.8/0.8671** | 0.6/0.8506 | 0.8/0.8636 |
| abalone | 1.0/0.6159 | **0.5/0.6460** | 1.0/0.6262 | 1.0/0.6250 | 0.5/0.6317 |
| sick_euthyroid | 0.3/0.9153 | **0.9/0.9216** | 0.9/0.9170 | 0.4/0.9176 | 0.5/0.9157 |
| yeast_ml8 | 0.4/0.5216 | **0.9/0.5350** | 0.7/0.5330 | 0.2/0.5292 | 0.6/0.5198 |
| thyroid_sick | 0.4/0.9472 | 0.5/0.9461 | **0.9/0.9538** | 0.9/0.9445 | 0.5/0.9459 |
| wine_quality | 0.7/0.7223 | **0.9/0.7251** | 0.8/0.7199 | 1.0/0.7201 | 0.8/0.7107 |
| ozone_level | 0.5/0.6675 | 0.4/0.6735 | **0.9/0.6738** | 0.5/0.6542 | 0.9/0.6591 |
| abalone_19 | 1.0/ 0.6101 | 1.0/0.5825 | 0.9/0.6093 | 0.9/ 0.5777 | **1.0/0.6114** |
| optical_digits | 0.9/ 0.9028 | 1.0/0.9060 | **0.7/0.9121** | 0.6/0.9077 | 1.0/0.9104 |

TABLE IV
THE COMPARISON OF PSMOTE WITH OTHER OVERSAMPLING METHODS
RESULTS REPRESENT (SAMPLING RATIO)/(PARTICLE NUMBER)/AUC

| **Datasets** | **sampling ratio/AUC** | | | | | | |
|---|---|---|---|---|---|---|---|
| | ***/*** | ***rand_o*** | ***adasyn*** | ***borderline_s*** | ***svm_s*** | ***smote*** | ***psmote*** |
| oil | 0.6869 | 0.5/0.6980 | 0.2/0.7516 | 0.2/0.7633 | 0.7/0.7338 | 0.3/0.7358 | **0.4/200/0.7646** |
| ecoli | 0.7189 | 0.4/0.7691 | 0.4/0.8171 | **0.2/0.8255** | 0.8/0.7448 | 1.0/0.8205 | 0.4/100/0.8079 |
| car_eval_34 | 0.9561 | 0.3/0.9754 | 0.3/0.9718 | 0.2/0.9674 | **0.2/0.9799** | 1.0/0.9720 | 0.3/250/0.9754 |
| us_crime | 0.6687 | 0.2/0.6943 | 0.4/0.7162 | 0.9/0.7112 | 0.2/0.7125 | **0.7/0.7333** | 0.7/100/0.7310 |
| solar_flare_m0 | 0.6021 | 0.6/0.6416 | 0.5/0.6420 | 0.5/0.6482 | 0.2/0.6368 | 0.6/0.6321 | **0.2/100/0.6517** |
| yeast_me2 | 0.6407 | 0.9/0.6660 | 0.9/0.7474 | 0.6/0.7377 | 0.8/0.7068 | 0.7/0.7177 | **0.9/150/0.7498** |
| spectrometer | 0.8231 | 0.2/0.8342 | 0.4/0.8714 | **0.3/0.8918** | 0.8/0.8714 | 0.5/0.8836 | 0.7/250/0.8877 |
| libras_move | 0.8160 | 0.3/0.8485 | 0.3/0.8756 | 0.4/0.8470 | 0.2/0.8756 | 0.8/0.8676 | **0.3/150/0.8845** |
| arrhythmia | 0.7860 | 0.2/0.8118 | **0.6/0.9083** | 0.9/0.9071 | 0.6/0.8495 | 0.8/0.8506 | 0.8/150/0.8671 |
| abalone | 0.5887 | 0.2/0.6087 | 0.7/0.6310 | 0.2/0.6312 | 0.3/0.6393 | 0.7/0.6433 | **0.5/100/0.6460** |
| sick_euthyroid | 0.8947 | 0.6/0.9069 | 0.6/0.9182 | 1.0/0.9134 | **0.9/0.9220** | 0.3/0.9176 | 0.9/100/0.9216 |
| yeast_ml8 | 0.5112 | 0.2/0.5256 | 0.9/0.5266 | 1.0/0.5271 | 0.7/0.5212 | 0.4/0.5290 | **0.9/100/0.5350** |
| thyroid_sick | 0.9357 | 0.5/0.9440 | 0.7/0.9498 | 1.0/0.9414 | 0.7/0.9495 | 0.6/0.9475 | **0.9/150/0.9538** |
| wine_quality | 0.6886 | 0.3/0.6804 | 0.5/0.7151 | 1.0/0.7156 | 0.5/0.7131 | 0.7/0.7199 | **0.9/100/0.7251** |
| ozone_level | 0.6005 | 0.7/0.6355 | 0.2/0.6604 | 0.8/0.6573 | 0.3/0.6257 | 0.6/0.6469 | **0.9/150/0.6738** |
| abalone_19 | 0.5123 | 0.4/0.5454 | 0.7/0.5757 | 0.6/0.5946 | 0.2/0.5845 | 0.3/0.5688 | **1.0/250/0.6114** |
| optical_digits | 0.8858 | 0.7/0.9178 | **1.0/9194** | 0.7/0.9000 | 0.6/0.8971 | 0.8/0.9093 | 0.7/150/0.9121 |

TABLE VI
AVERAGE RANKING.

| **Datasets** | **AUC ranking** | | | | | | |
|---|---|---|---|---|---|---|---|
| | ***/*** | ***rand_o*** | ***adasyn*** | ***borderline_s*** | ***svm_s*** | ***smote*** | ***psmote*** |
| oil | 7 | 6 | 3 | 2 | 5 | 4 | 1 |
| us_crime | 7 | 6 | 3 | 5 | 4 | 1 | 2 |
| solar_flare_m0 | 7 | 4 | 3 | 2 | 5 | 6 | 1 |
| spectrometer | 7 | 6 | 5 | 1 | 4 | 3 | 2 |
| ecoli | 7 | 5 | 3 | 1 | 6 | 2 | 4 |
| car_eval_34 | 7 | 2 | 5 | 6 | 1 | 4 | 2 |
| yeast_me2 | 7 | 6 | 2 | 3 | 5 | 4 | 1 |
| libras_move | 7 | 5 | 2 | 6 | 2 | 4 | 1 |
| arrhythmia | 7 | 6 | 1 | 2 | 5 | 4 | 3 |
| abalone | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| sick_euthyroid | 7 | 6 | 3 | 5 | 1 | 4 | 2 |
| yeast_ml8 | 7 | 5 | 4 | 3 | 6 | 2 | 1 |
| thyroid_sick | 7 | 5 | 2 | 6 | 3 | 4 | 1 |
| wine_quality | 6 | 7 | 4 | 3 | 5 | 2 | 1 |
| ozone_level | 7 | 5 | 2 | 3 | 6 | 4 | 1 |
| abalone_19 | 7 | 6 | 4 | 2 | 3 | 5 | 1 |
| optical_digits | 7 | 2 | 1 | 5 | 6 | 4 | 3 |
| avg_ranking | 6.9 | 5.2 | 3.1 | 3.5 | 4.1 | 3.5 | 1.6 |

TABLE V
THE EFFECT OF PSMOTE IN HYBRID OVERSAMPLING.

| **Datasets** | **AUC** | | |
|---|---|---|---|
| | ***smote*** | ***smote+enn*** | ***psmote+enn*** |
| oil | 0.7516 | **0.7829** | 0.7826 |
| ecoli | 0.8171 | 0.8780 | **0.8929** |
| car_eval_34 | 0.9718 | 0.9880 | **0.9944** |
| us_crime | 0.7162 | 0.7985 | **0.8017** |
| solar_flare_m0 | 0.6420 | 0.7075 | **0.7177** |
| yeast_me2 | 0.7474 | 0.7929 | **0.8041** |
| spectrometer | 0.8714 | **0.9058** | 0.8998 |
| libras_move | 0.8756 | 0.8785 | **0.8850** |
| arrhythmia | **0.9083** | 0.8825 | 0.9020 |
| abalone | 0.631 | 0.7519 | **0.7605** |
| sick_euthyroid | 0.9182 | **0.9494** | 0.9454 |
| yeast_ml8 | 0.5266 | 0.5566 | **0.5721** |
| thyroid_sick | 0.9498 | 0.9495 | **0.9559** |
| wine_quality | 0.7151 | 0.7499 | **0.7629** |
| ozone_level | 0.6604 | **0.6973** | 0.6969 |
| abalone_19 | 0.5757 | **0.6454** | 0.6364 |
| optical_digits | **0.9196** | 0.9194 | 0.9181 |

TABLE VII
WILCOXON SIGNED-RANK TEST

| Hypothesis | P value ($\alpha = 0.05$) |
|---|---|
| psmote & rand_o | 0.0005 |
| psmote & adasyn | 0.0277 |
| psmote & borderline_s | 0.0006 |
| psmote & svm_s | 0.0019 |
| psmote & smote | 0.0005 |
| smote+enn & psmote+enn | 0.0217 |

hypothesis of equivalent performance with other oversampling methods, which proves the significance of our method.

## V. CONCLUSION

In this paper, an oversampling method optimized by evolutionary algorithm is proposed for imbalance classification. Different from traditional oversampling methods, the proposed method used auc as objective function and expand the minority

class according to synthetic samples' fitness value on the original dataset. Through sample distribution visualization and experiment results on 17 imbalanced dataset, we can see that our method has better sample distribution than SMOTE in terms of sample contour and outlier distribution, and has satisfying performance when compared with mainstream imbalance methods. We also proved that data distribution optimization can help for hybird oversampling process. Our future work is to expand the method to multi-label classification with the combination of ensemble learning.

Our method is a wrapper algorithm, so it can be easily expanded with other sampling methods, and we think there are potential research directions such as design appropriate objective function for different imbalanced tasks or approximate sample distribution in a better way.

## REFERENCES


[1] He H, Garcia E A. Learning from Imbalanced Data, *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263-284, 2009.

[2] N. V. Chawla, K. W. Bowyer, L. O.Hall, W. P. Kegelmeyer. SMOTE: synthetic minority over-sampling technique, *Journal of artificial intelligence research*, vol. 16, no. 1, pp. 321-357, 2002.

[3] G. 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*,vol. 6, no. 1, pp 20-29, 2004.

[4] Kennedy J, Eberhart R. Particle swarm optimization, *IEEE International Conference on Neural Networks*, pp. 1942-1948, 1995.

[5] He H, Bai Y, Garcia E A, *et al*. ADASYN: adaptive synthetic sampling approach for imbalanced learning, *IEEE International Joint Conference on Neural Networks*, pp. 1322-1328, 2008.

[6] Han H, Wang W Y, Mao B H. Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning, *International Conference on Intelligent Computing*, pp. 878-887, 2005.

[7] Rastogi A K , Narang N , Siddiqui Z A . Imbalanced big data classification: a distributed implementation of SMOTE, *the Workshop Program of the 19th International Conference. ACM*, 2018.

[8] Gosain, Anjana Sardana, Saanchi. (2019). Farthest SMOTE: A Modified SMOTE Approach. 10.1007/978-981-10-8055-5_28.

[9] Binghao Y , Guodong H , Yajing H , et al. New traffic classification method for imbalanced network data. *Journal of Computer Applications*, 2018.

[10] Hamdy A , El-Laithy A . SMOTE and Feature Selection for More Effective Bug Severity Prediction. *International Journal of Software Engineering and Knowledge Engineering*, 2019, 29(6):897-919.

[11] Wu Z , Lin W , Ji Y . An Integrated Ensemble Learning Model for Imbalanced Fault Diagnostics and Prognostics. *IEEE Access*, 2018, 6:1-1.

[12] M. Kubat, S. Matwin. Addressing the curse of imbalanced training sets: one-sided selection, *International Conference on Machine Learning*, pp. 179-186, 1997.

[13] P. Hart. The condensed nearest neighbor rule, *IEEE Transactions on Information Theory*, vol. 14, no. 3, pp. 515-516, 1968.

[14] 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*, vol. 126, 2003.

[15] Tomek I. An experiment with the edited nearest-neighbour rule, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 6, no. 6, pp. 448-452, 1976.

[16] Liu C, Cao L, Philip S Y. A hybrid coupled k-nearest neighbor algorithm on imbalance data, *IEEE International Joint Conference on Neural Networks*, pp. 2011-2018, 2014.

[17] Naik B, Swetanisha S, Behera D K, *et al*. Cooperative swarm based clustering algorithm based on PSO and k-means to find optimal cluster centroids. *IEEE National Conference on Computing and Communication Systems*, pp. 1-5, 2012.

[18] Han X , Cui R , Lan Y , et al. A Gaussian mixture model based combined resampling algorithm for classification of imbalanced credit data sets. *International Journal of Machine Learning and Cybernetics*, 2019(1).

[19] Tomek I. Two modifications of CNN, *IEEE Transactions on Systems Man & Cybernetics*, vol. 6, no. 11, pp. 769-772, 1976.

[20] D. Wilson, Asymptotic. Properties of Nearest Neighbor Rules Using Edited Data, In *IEEE Transactions on Systems, Man, and Cybernetrics*, vol. 2, no. 3, pp. 408-421, 1972.

[21] Yang P, Yoo P D, Fernando J, *et al*. Sample subset optimization techniques for imbalanced and ensemble learning problems in bioinformatics applications, *IEEE Transactions on Cybernetics*, vol. 44, no. 3, pp. 445-455, 2014.

[22] Wang K J , Makond B , Chen K H , *et al*. A hybrid classifier combining SMOTE with PSO to estimate 5-year survivability of breast cancer patients, *Applied Soft Computing*, 2014, vol. 20, pp. 15-24.

[23] Hu K , Zhou Z , Weng L , et al. An Optimization Strategy for Weighted Extreme Learning Machine based on PSO. *International Journal of Pattern Recognition and Artificial Intelligence*, 2016, 31(1):1751001.

[24] Ding, Zejin. Diversified Ensemble Classifiers for Highly Imbalanced Data Learning and their Application in Bioinformatics, *Dissertation*, Georgia State University, 2011.

[25] Fawcett T. An introduction to ROC analysis, *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861-874, 2005.

[26] Ghazikhani A, Yazdi H S, Monsefi R. Class imbalance handling using wrapper-based random oversampling, *2012 20th Iranian Conference on Electrical Engineering*, pp. 611-616, 2012.

[27] Jin-Hyuk Hong, Sung-Bae Cho. A probabilistic multi-class strategy of one-vs.-rest support vector machines for cancer classification. *Neurocomputing*, vol. 71, no. 16-18, pp. 3275-3281. 2008.

[28] Hernandez W, Maldonado-Correa J L. Power Performance Verification of a Wind Turbine by using the Wilcoxon Signed-Rank Test, *IEEE Transactions on Energy Conversion*, pp. 1-1, 2016.

[29] Del Valle Y, Venayagamoorthy G K, Mohagheghi S, et al. Particle swarm optimization: Basic concepts, variants and applications in power systems, *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 2, pp. 171C195, 2008.