Electronic
Research Archive

http://www.aimspress.com/journal/era

ERA, 31(6): 3453–3470.
DOI: 10.3934/era.2023175
Received: 27 February 2023
Revised: 30 March 2023
Accepted: 10 April 2023
Published: 19 April 2023

Research article

Classification method for imbalanced LiDAR point cloud based on stack autoencoder

Peng Ren[1,2] **and Qunli Xia**[1,*]

[1] School of Aerospace Engineering, Beijing Institute of Technology, Beijing 100081, China
[2] Southwest Institute of Technical Physics, Chengdu 610041, China

* **Correspondence:** Email:1010@bit.edu.cn; Tel: +8613520113147.

Abstract: The existing classification methods of LiDAR point cloud are almost based on the assumption that each class is balanced, without considering the imbalanced class problem. Moreover, from the perspective of data volume, the LiDAR point cloud classification should be a typical big data classification problem. Therefore, by studying the existing deep network structure and imbalanced sampling methods, this paper proposes an oversampling method based on stack autoencoder. The method realizes automatic generation of synthetic samples by learning the distribution characteristics of the positive class, which solves the problem of imbalance training data well. It only takes the geometric coordinates and intensity information of the point clouds as the input layer and does not need feature construction or fusion, which reduces the computational complexity. This paper also discusses the influence of sampling number, oversampling method and classifier on the classification results, and evaluates the performance from three aspects: true positive rate, positive predictive value and accuracy. The results show that the oversampling method based on stack autoencoder is suitable for imbalanced LiDAR point cloud classification, and has a good ability to improve the effect of positive class. If it is combined with optimized classifier, the classification performance of imbalanced point cloud is greatly improved.

Keywords: LiDAR point cloud; imbalanced classification; deep neural network; stack autoencoder; oversampling

1. Introduction

LiDAR [1] can obtain the geometric coordinates of the spatial points and the intensity information of the laser echo signal. It has the advantages of high spatial resolution, accurate spatial positioning, and objective reality [2], and is widely used in forest ecosystem [3], atmospheric remote sensing [4], autonomous driving technology [5], and 3D city models [6], etc. The necessary guarantee for the realization of these applications is the construction of classification algorithms for LiDAR point clouds with high precision and real-time performance.

Point cloud classification is to divide the point clouds into various object instances, such as buildings, ground and vegetation. The classification methods of LiDAR point clouds mainly fall into two categories: machine learning-based method and deep learning-based method. In the former method, feature vectors are extracted, constructed or fusioned, and then machine classification algorithm is used to classify point clouds. Song et al. [7] extracted geometrical features including volume, density, and eigenvalues in the three principal directions as a basis for classification to classify objects into five types by back-prop agation neural network(ANN) model. Li et al. [8] proposed a point cloud classification algorithm: multilevel features of a single point are constructed by multi-resolution of the point cloud and multi-neighborhood spaces, and then the points are classified by the support vector machine(SVM) based on a Gaussian kernel function. Sun et al. [9] selected 20 statistical features and used the optimized random forest(RF) algorithm to classify the vegetation point cloud. According to the processing mode of point cloud data, classification methods based on deep learning mainly include projection-based method [10], point-based method [11] and graph-based method [12].

These methods are almost based on the assumption that each class is balanced, without considering the imbalanced class problem. Imbalanced data refers to the data with a very small proportion of one or more classes in the original samples [13]. Usually, the class with fewest samples is called the positive class, and the class with a relatively large number of samples is called the negative class. If the data set is imbalanced, the classification algorithm cannot correctly learn the feature of positive sample, and tends to the negative class, which may lead to the bias of classification results and decision errors.

The classification effect of imbalanced data is mainly improved by optimizing the sampling algorithm [14], which is an improvement at the data level. Because the sampling algorithm can effectively solve the problem of imbalanced distribution of positive and negative samples, and the optimization design of the sampling algorithm is relatively easy, the performance of positive sample classification accuracy can be significantly improved. According to the different ways of adjusting data, the traditional sampling algorithms can be divided into three categories: undersampling, oversampling, and hybrid sampling [15]. Undersampling algorithms [16] remove data from the negative class until both contain an equal number of samples, which may suffer from feature loss due to data deletion. Oversampling algorithms [17] generate data in positive class until both contain an equal number of samples, which may have the problem of overfitting the training data, such as synthetic minority oversampling technique(SMOTE) [18] and adaptive semi-unsupervised weighted oversampling (ASUWO) [19]. Hybrid sampling algorithms [13] balance the datasets by combining undersampling and oversampling algorithm, thus having the advantages and disadvantages of these two algorithms.

In addition, there are many improved methods of classifier for processing imbalanced data, which is an improvement at the algorithm level. They can also be divided into three categories [20]: ensemble based on Bagging, Boosting, and Hybrid, it depends on the ensemble learning algorithm they are based on.

At present, there are few researches on imbalanced point cloud classification. Table 1 compares the methods of several research papers in the last three years. From Table 1, we can find some shortcomings of the existing research: 1) The amount of point cloud data is relatively large, but the current data processing methods are all using the traditional imbalanced data sampling method. 2) Excessive selection of classification features increases the complexity and difficulty of calculation. 3) The classifiers are not been improved or optimized to be more suitable for the classification of imbalanced data.

Table 1. Classification methods for Imbalanced Lidar Point Cloud.

Method	[21]	[22]	[23]
Object	Graphical Card	Building Roof	Sea Scene
Number of Points	32,768	25,695	30,340
Number of Features	6	12	15
Sampling Algorithm	Hybrid Sampling	Hybrid Sampling	SMOTE
Classifier	PointNet	SVM	ANN

Therefore, our research motivation is to find a suitable classification method for imbalanced point cloud. The research contributions include:

1) Propose an oversampling algorithm suitable for big data by using deep network. Stack autoencoder(SAE) [24] is an unsupervised deep neural network with excellent properties of learning data rules. It is often used in dimensionality reduction [25,26], denoising [27,28], feature extraction [29,30], outlier detection [31, 32], and so on. In this study, SAE is used for oversampling the class with least samples [33]. The SAE realizes automatic generation of synthetic samples by learning the distribution characteristics of the class with least samples, which improves the balance of training data. Compared to SMOTE and ASUWO, it is not affected by negative class and is better suited to imbalanced multi-class problems.

2) Minimize the dimension of feature vectors used for point cloud classification. As the input layer of the SAE, geometric coordinates and intensity information of the point clouds do not need feature construction or fusion. All kinds of point clouds, such as ground, vegetation, vehicles, and buildings in urban scenes, can be well identified by using only geometric coordinates and intensity information.

3) Select a classification method suitable for imbalanced point cloud by comparing eight classifiers. Kaur et al. [17] pointed out that the optimization of classifier can also improve the classification performance of imbalanced data to a certain extent. The experimental study in this paper also verified this point. The improvement and optimization of classifier had a significant effect on the increase of ACC value. If the oversampling algorithm is combined with classifier optimization, the classification performance of imbalanced point cloud is greatly improved.

The rest of this paper is organized as follows. Section 2 reviews two oversampling algorithms(SMOTE and ASUWO) for imbalanced data and expounds the oversampling method based on SAE for imbalanced LiDAR point cloud. Section 3 introduces the source of experimental data and the evaluation index of classification performance. Section 4 analyzes the influence of classifiers, sampling number and oversampling algorithms on classification results. In Section 5, the thesis is summarized and the future research direction is pointed out.

2. Oversampling method

After reviewing the sampling principle of the two algorithms and the network structure of SAE, the idea and step of oversampling method based on SAE for imbalanced LiDAR point cloud are described.

2.1. SMOTE

SMOTE is the most representative oversampling method, the basic idea of which is to randomly generate a new sample between each positive sample and its k nearest neighbors. Since the generated samples are random values between two samples, SMOTE solves the problem of sample diversity. The main process of SMOTE algorithm is shown as follows.

Algorithm 1. SMOTE

```
Input
1) Positive sample set $X^P = \{x_1^P, \cdots, x_m^P\}$;
2) Negative sample set $X^N = \{x_1^N, \cdots, x_t^N\}$, $m < t$;
3) Number of nearest neighbors $k$;
Output New positive sample set $G^P = \{x_1^P, \cdots, x_m^P, x_1^{new}, \cdots, x_{t-m}^{new}\}$.
Procedure
1. Using Euclidean distance, compute $k$ nearest neighbors $T_i^P = \{x_{i1}^P, \cdots, x_{ik}^P\}$ for each positive sample $x_i^P, i = 1, \cdots, m$.
2. Using formula $x_{ij}^{new} = x_i^P + rand(0,1) \times ||x_i^P - x_{ij}^P||$, generate the synthetic samples $x_{ij}^{new}, j = 1, \cdots, k$; $rand(0,1)$ is a random number between 0 and 1.
3. Repeat steps 1 to 2 until generate $t - m$ samples $x_1^{new}, \cdots, x_{t-m}^{new}$ to get $G^P$.
```

In order to improve the quality of synthetic samples, researchers proposed a series of improved algorithms based on smote. Han et al. [34] put forward the Borderline SMOTE to solve the problem that samples in the boundary area are easy to be misclassified by the classifier. Bunkhumpornpat et al. [35] came up with Safe-level SMOTE to generate positive samples by calculating the weight of k nearest neighbors, which avoided the problem of synthetic data crossing the boundary of positive samples. Douzas et al. [36] presented an oversampling method based on k-means clustering and SMOTE, which avoids the generation of noise and effectively overcomes imbalances between and within classes.

2.2. ASUWO

ASUWO clusters the positive samples using a semi-unsupervised hierarchical clustering approach and adaptively determines the size to oversample each subcluster using its classification complexity and cross validation. Then, the positive samples are oversampled depending on their Euclidean distance to the negative class. ASUWO address the problem of over-generalization by identifying positive boundary samples and assigning them different weights, which also mitigate the between-class imbalances and within-class imbalances by combining with clustering algorithms [15]. The main process of ASUWO algorithm is shown as follows.

Algorithm 2. ASUWO

```
Input
1) Positive sample set $X^P = \{x_1^P, \cdots, x_m^P\}$;
2) Negative sample set $X^N = \{x_1^N, \cdots, x_t^N\}$, $m < t$;
```

3) Number of nearest neighbors k;

Output New positive sample set $G^P = \{x_1^P, \cdots, x_m^P, x_1^{new}, \cdots, x_l^{new}\}$.

Procedure

1. Cluster X^P using the improved hierarchical clustering method, and determine the sizes S_i for all subclusters $C_i^P, i = 1, .., n$.
2. For $\forall x_{ih}^P \in C_i^P$, find k nearest neighbors $\{x_{ih_1}^N, \cdots, x_{ih_k}^N\} \in X^N$ according to Euclidean distance.
3. Determine the weights $W(x_{ih}^P)$ and probability distribution $P(x_{ih}^P)$.
4. Select a positive sample $\alpha \in C_i^P$ by sampling from $P(x_{ih}^P)$, and randomly one of its k nearest neighbors $\beta \in C_i^P$.
5. Generate a new synthetic sample x^{new} between α and β using the formula $x^{new} = rand(0, 1) \times \alpha + (1 - rand(0, 1)) \times \beta$.
6. Repeat steps 2 to 5 until the subcluster size reaches S_i and generate l samples $x_1^{new}, \cdots, x_l^{new}$.

2.3. SAE-based oversampling method

The above-mentioned oversampling methods are almost based on the imbalanced two-classification problem, and the generation of synthetic samples is greatly affected by the neighborhood and boundary problems.

An autoencoder(AE) [37] is a self-supervised network whose output is equal to its input under ideal conditions. By establishing a reasonable loss term, the hidden layer features can be extracted under unsupervised learning. Especially when the input features are correlated with each other, the autocoding learning algorithm can extract the correlation in the input data. Furthermore, the extracted features are taken as the new input, and the corresponding encoding features can be obtained through parameter learning. The AE structure is shown in Figure 1(a).

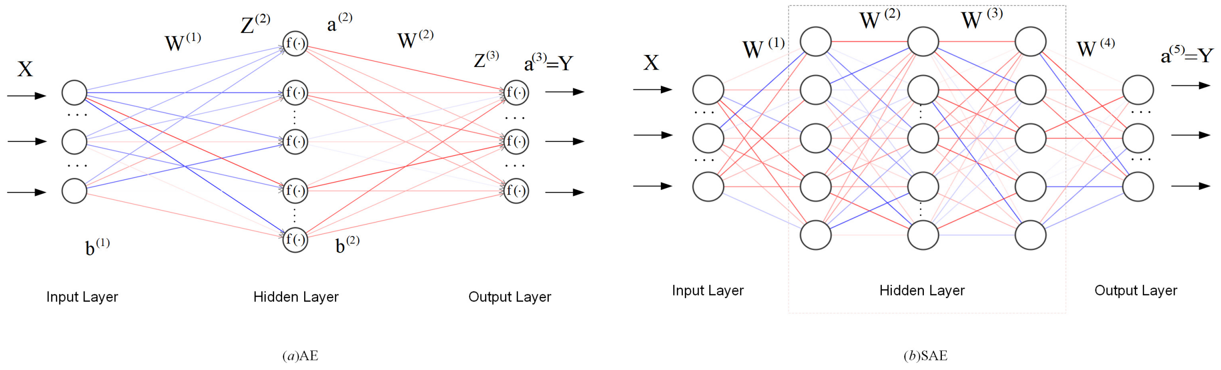

Figure 1. The network structure.

The AE consists of the input layer, the encoding layer, and the decoding layer. Suppose $X = \{X(1), \cdots, X(m)\}$ is the input data and $f(x) = 1/(1+e^{-x})$ represents the activation function, the encoding layer is defined as:

$$Z^{(2)} = W^{(1)}X + b^{(1)}, a^{(2)} = f(Z^{(2)}), \tag{2.1}$$

where $Z^{(2)}$, $W^{(1)}$, $b^{(1)}$ and $a^{(2)}$ denote the input data, the weight matrix, the bias vector, and the output data of encoding layer, respectively.

In a similar way, the decoding layer is defined as:

$$Z^{(3)} = W^{(2)}a^{(2)} + b^{(2)}, a^{(3)} = f(Z^{(3)}), \tag{2.2}$$

where $Z^{(3)}$, $W^{(2)}$, $b^{(2)}$ and $a^{(3)}$ denote the input data, the weight matrix, the bias vector, and the output data of decoding layer, respectively.

The multiple AEs are connected in succession to obtain a stacked autoencoder(SAE) model [24]. The network structure of the SAE with three hidden layers is shown in Figure 1(b).

SAE is typically trained using a layer-by-layer greedy approach. The outputs of encoding layer of an AE are fed to the next AE as their inputs. Each AE learns in the same way as a single AE and the first AE uses the original input from the data as inputs. The network calculation formula of SAE is as follows:

$$a^{(1)} = X, Z^{(K)} = W^{(K-1)}a^{(K-1)} + b^{(K-1)}, a^{(K)} = f(Z^{(K)}), k \geqq 2. \tag{2.3}$$

The training process of SAE is to find a set of parameters($W^{(k)}, b^{(k)}, k = 1, \cdots, K$) to solve the following optimization problems [37]:

$$arg\,\min_{\theta} \frac{1}{2}\sum_{i=1}^{m} ||X(i) - a^{(K)}(i)||_2 + \frac{\lambda}{2}||W||_2. \tag{2.4}$$

The first term is the reconstruction error between the input X and output $a^{(K)}$. The second term aims to reduce the magnitude of weight and prevent overfitting. λ is the regularization parameter, W is a matrix consisting of the weight matrices ($W^{(k)}, k = 1, \cdots, K$). Usually, the parameters of SAE are optimized using the back-propagation method.

In order to overcome the influence of negative samples and mine the depth features of positive samples in big data, to generate better synthetic samples for solving unbalanced multi-classification problems, we propose a SAE-based oversampling method, the main steps of which are as follows: 1) Construct the network structure of SAE; 2) Take the samples with least samples as the input data of the input layer; 3) Extract features in the hidden layer; 4) Generate synthetic samples in the output layer. The main process of SAE-based oversampling is shown Algorithm 3.

Algorithm 3. SAE-based oversampling method

Input

1) Smallest class sample set $X^P = \{x_1^P, \cdots, x_m^P\}$;
2) Number of hidden layers K.
3) Number of hidden layer neurons s_K.
4) Maximum number of iterations T.

Output New positive sample set $G^P = \{x_1^P, \cdots, x_m^P, x_1^{new}, \cdots, x_m^{new}\}$.

Procedure

1. Construct the structure of SAE. The nodes of input layer are determined by the dimension of x_i^P. The output layer is the regression layer.

2. Take X^P as the input of SAE, train the parameters $W^{(1)}$ and $b^{(1)}$ of the first hidden layer with the greedy method, and use the trained parameters to calculate the output $a^{(2)}$ of the first hidden layer.

3. Take the output $a^{(k)}$ of the $k-1$ layer as the input of k layer, and train the parameters $W^{(k)}$ and $b^{(k)}$ of k layers.

4. Repeat step 3 until the last hidden layer is trained.

5. Use the optimization algorithm to iteratively find the parameter values near the minimum value of the cost function, and take them as the final optimal parameter values of the whole network.

6. When the training time is T and the network is stable, $\{x_1^{new}, \cdots, x_m^{new}\}$ is the generated synthetic data to get G^P.

The process of proposed method for classification of imbalanced LiDAR point cloud is as follows:

Step 1. Preprocess the point cloud data, and extract the geometric coordinates and intensity information (x, y, z, i) of each point. The training set and the testing set are randomly generated in a certain ratio, such as 3:1.

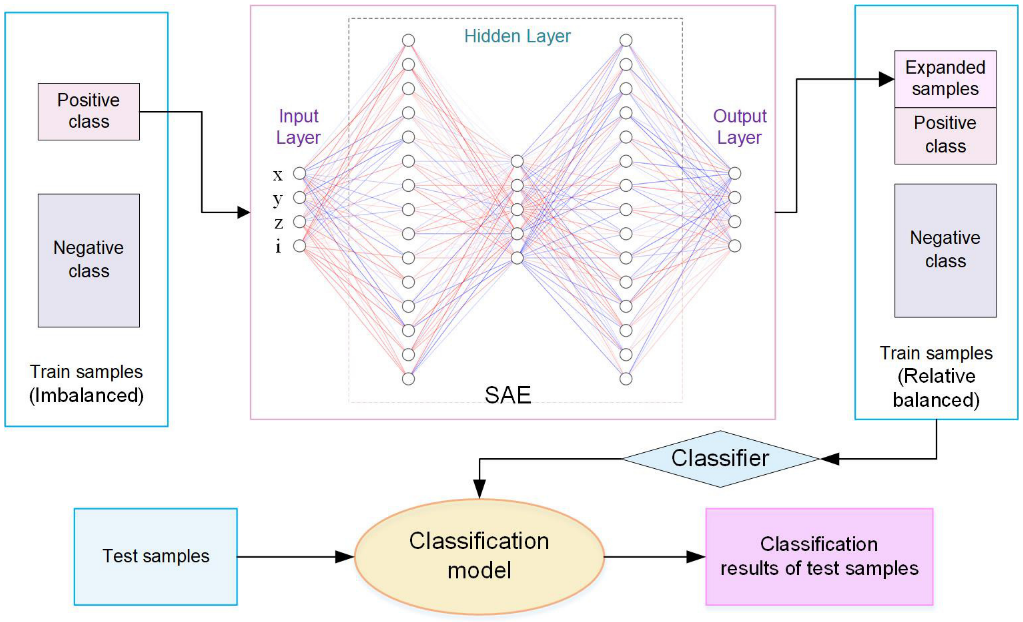

Figure 2. Flowchart of proposed method for classification.

Step 2. Oversample the class with least samples in the training set using the proposed method. Network structure of SAE includes: 1 input layer, K hidden layers, 1 output layer. Both the input layer and the output layer have 4 neurons corresponding to four features (x, y, z, i). The neurons of each hidden layer are respectively: $s_1, s_2, \cdots, s_K$. $s_1, s_2, \cdots, s_K$ and K need to be obtained through experimental debugging.

Step 3. Train the relative balanced samples after oversampling with the classifier to get the classification model.

Step 4. Obtain the classification results of the test samples using the model.

The operation process of this method is shown in Figure 2.

3. Experimental study

In order to test the effectiveness and excellence of the proposed method, we perform classification for point clouds in Pandaset-LidarData dataset, and compare the performance of three oversampling methods, eight classifiers and their combinations.

3.1. Data source

PandaSet contains 2560 frame preprocessed organized point clouds of various urban scenes captured using the Pandar64 sensor. Each point cloud contains 13 classes of semantic segmentation labels. Several frame point clouds are randomly selected for this research. Figure 3 shows the first frame point cloud.

The initial 13 classes include unlabelled, vegetation, ground, road, road markings, sidewalk, car, truck, other vehicle, pedestrian, road barriers, signs, buildings. Because some classes have fewer points, the 13 classes are adjusted to 5 classes: ground, vegetation, vehicle, building, and other. Ground, road and sidewalk all fall into the class ”ground”. Car, truck and other vehicle are all included in the class ”vehicle”. Unlabelled, road markings, pedestrian, road barriers and signs are classified into the class ”other”. Figure 4 shows the scatter diagram of the first frame point cloud after class adjustment. Table 2 shows the distribution of each class in some frames.

It can be clearly seen from Table 2 that the classification of each frame is an imbalanced problem. Such as the first frame, the ground class is far lower than other classes and the difference is up to 20 times, which means ”ground” is the positive class. We should generate a fair amount of ”ground” data to balance the training set, then build a classification model to get the results on test set.

3.2. Evaluation indicator

In order to better quantitatively analyze the results, we introduce the confusion matrix [38] and several evaluation indicators including true positive rate(TPR), positive predictive value(PPV), and accuracy(ACC). The confusion matrix is shown in Figure 5. The calculation formulas are as follows:

$$TPR = \frac{TP}{TP + FN} \times 100\%, \tag{3.1}$$

$$PPV = \frac{TP}{TP + FP} \times 100\%, \tag{3.2}$$

$$ACC = \frac{TP + TN}{TP + FN + FP + TN} \times 100\%, \tag{3.3}$$

where TP is to predict the original positive class as the positive class, FN is to predict the original positive class as the negative class, FP is to predict the original negative class as the positive class, TN is to predict the original negative class as the negative class. Higher values of ACC, TPR and PPV indicate better classification effect.

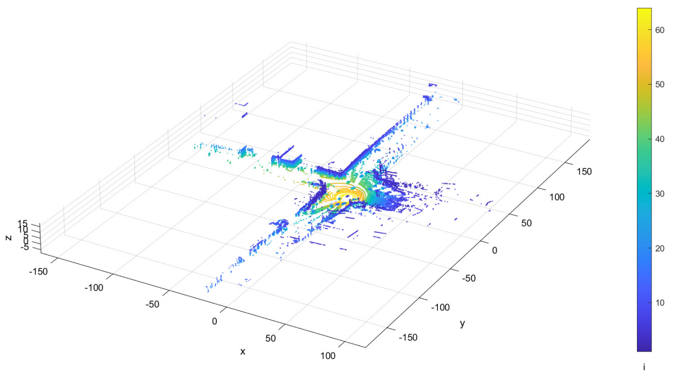

Figure 3. The first frame point cloud.

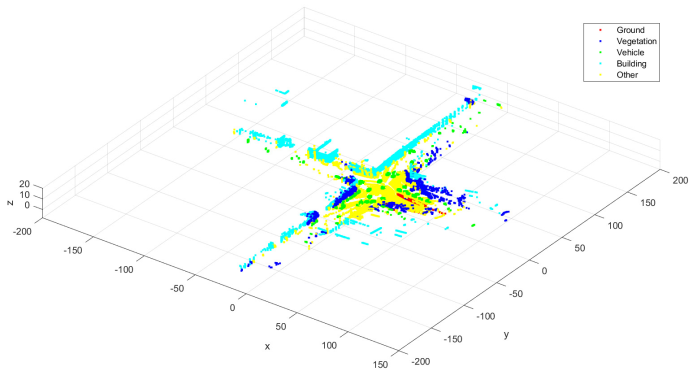

Figure 4. Scatter plot of the pre-processed point cloud.

Table 2. Class distribution of some frames.

Frame	Class	Ground	Vegetation	Vehicle	Building	Other	Total
0001	Number	1631	9560	15,422	21,929	33,881	82,423
	Percentage	2.0%	11.6%	18.7%	26.6%	41.1%	100%
0100	Number	1665	13,709	16,995	24,521	20,740	77,630
	Percentage	2.1%	17.7%	21.9%	31.6%	26.7%	100%
1500	Number	6045	11,102	11,138	51,220	2730	82,235
	Percentage	7.4%	13.5%	13.5%	62.3%	3.3%	100%
2000	Number	18,525	22,097	4365	28,050	10,899	83,936
	Percentage	22.1%	26.3%	5.2%	33.4%	13.0%	100%
2500	Number	11,289	8754	10,396	2032	44,264	76,735
	Percentage	14.7%	11.4%	13.5%	2.7%	57.7%	100%

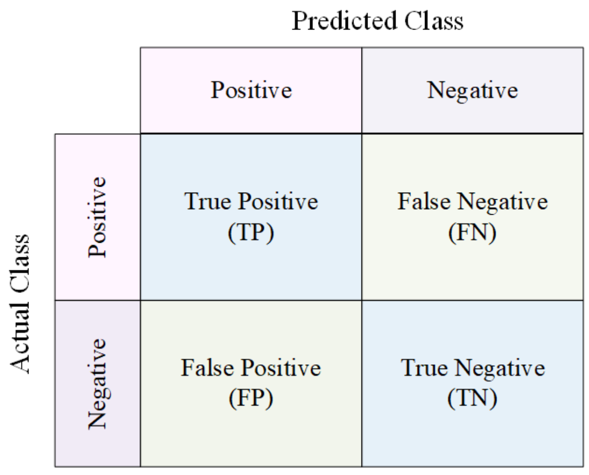

Figure 5. Confusion matrix.

4. Results and analysis

All algorithmic programs covered in this work were implemented using Matlab R2022a. Furthermore, Cross-validation was used for classification. The number of folds in the cross-validation technique was five. After debugging, parameters set for each algorithm in this section are as follows: $K = 50$ in both SMOTE and ASUWO algorithms. There are 3 hidden layers, and the number of neurons is respectively 15, 5, 15 in SAE algorithm. $T = 1000$.

4.1. Influence of Sampling number

SAE-based oversampling method can fully excavate the features of positive samples and randomly generate synthetic data. To verify the effectiveness of SAE-based Oversampling method and discuss the relationship between number of synthetic samples and classification results, the first frame was chosen as the experimental object.

75% of the 82,423 points, namely, 61,818 are randomly selected as the training set and the rest as test set. The training set contains points for each class: 1225, 7195, 11,551, 16,512 and 25,335. We took 1225 ground points as the basis to generate 1, 2, up to 20 times of ground data, conducted separately classification training using RUS-ODT, and got results on the test set. As the number of samples gets larger, the value of ACC is stable around 0.978 without big fluctuation, and the values of TPR and PPV of each class have different changes. Figure 6 presents the variation charts of TPR and PPV with respect to number of synthetic samples, where the abscissa represents a multiple of 1225.

It is easy to see from Figure 6 that: 1) Buildings and vehicles may have obvious classification characteristics, so the values of TPR and PPV are relatively high. Ground and vegetation, especially ground and dwarf vegetation, are not easy to distinguish, so both TPR and PPV values are low. Therefore, other features can be introduced to better realize the classification of ground and vegetation.

2) As the value of abscissa increases, that is, the number of synthetic samples enlarges, TPR and PPV of the ground class show an obvious increase, while indicators of other classes do not change significantly. This indicates that the oversampling method based on SAE can significantly improve the classification effect of the positive class without affecting the other classes. Therefore, in the follow-up

discussion, we only focused on ACC and indicators of positive class.

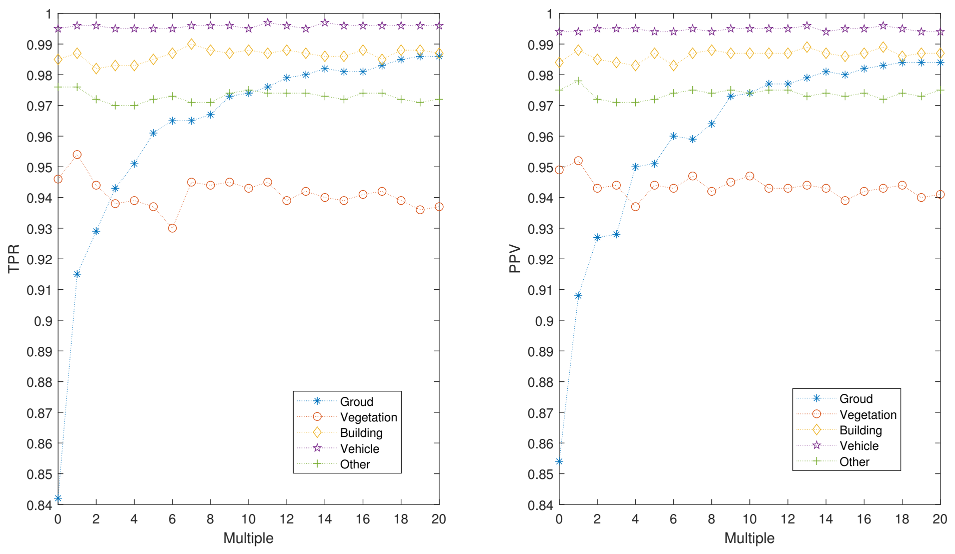

Figure 6. Line charts of TPR and PPV.

3) When the multiple approaches 6,13 and 20, the number of ground points is similar to the vegetation points, buildings and vehicles, respectively. But there is little difference between TPR and PPV. Therefore, which class is taken as the negative class in the oversampling has little influence on the classification effect. Therefore, the proposed method is not affected by negative class, so it can be applied to both imbalanced two-classification and multi-classification problems.

4) When the multiple is equal to 1, the indicators of ground class are greatly improved, and those of other classes reach the maximum. In order to reduce the amount of computation, we can generate twice as much data as the positive class for oversampling.

5) When the sampling multiple is greater than or equal to 5, TPR and PPV of vegetation class become the smallest, which means that the classification effect of vegetation class is the worst. This may be because the number of ground class exceeds that of vegetation, so vegetation class becomes the minimum class.

4.2. *Comparison for classifiers*

In order to select a classifier suitable for imbalanced LiDAR point cloud, we discussed the influence of SAE-based oversampling algorithm on different classifiers, and carried out the following experiments. 1) Taking the vegetation class as the negative class, the proposed method was used to generate the same amount of ground data to obtain the training set. 2) Experimental data was trained with different classifiers to establish the classification model. 3) The models are applied to the test set, and the classification results are obtained.

With the help of Matlab(2022a) Classification Learner App, eight classifiers were selected for comparison, including ANN, SVM, decision tree(DT), RUS-Boost [39], multilayer perception artificial neural network (MPL-ANN) [40], fine Gaussian support vector machine(FG-SVM) [41], Optimal randomized classification trees(OR-CT) [42] and optimizable RUS-Boosted decision tree ensemble(RUS-ODT) [43]. The first four are traditional classifiers, and the last four are improvements of the first four using Bayesian optimizer. Table 3 shows the parameter settings of four optimized classifiers, which are obtained through experimental debugging. Table 4 lists the classification indexes of eight classifiers on the test set.

Table 3. Configurations for four optimized classifiers.

Classifier	Specifications
MPL-ANN	Fully connected layer: 3 layers. The size of each layer: 10. Activation function: ReLU. Iteration limit: 1000. Regularization intensity: 0. Standaridize data: true.
FG-SVM	Kernel fuction: Gaussian. Kernel scale: 0.5. Box constraint level: 1. Multiclass method: One-vs-One. Standaridize data: true.
OR-CT	Learner type: Decision tree. Maximum number of splits: 1. Splitting criterion: Gini diversity index.
RUS-ODT	Preset: RUSBoosted Trees. Ensemble method: RUSBoost. Learner type: Decision tree. Learning rate: 0.1. Maximum number of splits:20. Number of learners: 30.

Table 4. Classification results based on different classifiers.

Classifier	Sampling	ACC	TPR(Groud)	PPV(Groud)	Time(s)
ANN	Unsampled	0.856	0.379	0.576	589
	SAE	0.864	0.891	0.837	442
MPL-ANN	Unsampled	0.924	0.610	0.727	660
	SAE	0.935	0.937	0.886	738
SVM	Unsampled	0.914	0.840	0.882	6531
	SAE	0.924	0.937	0.926	6744
FG-SVM	Unsampled	0.957	0.713	0.827	6515
	SAE	0.958	0.964	0.918	6486
DT	Unsampled	0.869	0.306	0.487	9
	SAE	0.882	0.872	0.759	5
OR-CT	Unsampled	0.976	0.842	0.850	70
	SAE	0.976	0.961	0.961	104
RUS-Boost	Unsampled	0.799	0.835	0.262	3482
	SAE	0.811	0.873	0.722	3501
RUS-ODT	Unsampled	0.990	0.908	0.943	3579
	SAE	0.991	0.983	0.980	3749

As can be seen from Table 4, 1) SAE-based oversampling method can improve the performance of each classifier, because all the indexes of sampling are higher than those of Unsampled.

2) Compared with the improvement of sampling method, the optimization of classifier has more

obvious effect on the increase of ACC value. Therefore, for imbalanced classification problems, the optimization of classifier and the improvement of sampling method can be combined to obtain better classification results.

3) The training time of each classifier in the unsampled state and the oversampling state does not increase or decrease significantly, so there is no need to worry about the problem of sampling affecting the training time.

4) RUS-ODT is the classifier with highest index value, while OR-CT is the one with relatively high index value and less time consuming. In imbalanced point cloud classification, RUS-ODT can be selected if accuracy is pursued, and OR-CT can be selected if efficiency is pursued.

4.3. Comparison for oversampling methods

To compare the performance of three oversampling methods, we first randomly selected several frames for research, and then carried out experiments on 7 UCI datasets. The smallest class is regarded as the positive class, which is oversampled by the three methods respectively, and then the classification result of the test set is obtained by RUS-ODT. Tables 5 and 6 show the classification results with different oversampling methods of point cloud and UCI data sets respectively. Figure 7 shows the scatter diagram of the positive class of the first frame point cloud before and after oversampling.

Table 5. Classification results based on the point clouds.

Frame	Positive class	Sampling method	Training points	ACC	TPR (Positive)	PPV (Positive)
0001	Ground	Unsampled	61,818	0.982	0.837	0.876
		SMOTE	67,788	0.983	0.879	0.840
		ASUWO	67,749	0.979	0.826	0.873
		SAE	67,943	0.980	0.928	0.926
0100	Ground	Unsampled	58,223	0.991	0.878	0.873
		SMOTE	67,256	0.992	0.886	0.869
		ASUWO	68,301	0.991	0.885	0.871
		SAE	69,461	0.990	0.942	0.948
1500	Other	Unsampled	61,676	0.991	0.843	0.859
		SMOTE	64,163	0.992	0.864	0.869
		ASUWO	64,073	0.991	0.897	0.868
		SAE	65,771	0.990	0.959	0.956
2000	Vehicle	Unsampled	62,952	0.989	0.885	0.880
		SMOTE	67,853	0.990	0.886	0.889
		ASUWO	68,156	0.988	0.910	0.926
		SAE	72,773	0.990	0.983	0.980
2560	Building	Unsampled	57,551	0.993	0.852	0.856
		SMOTE	62,593	0.994	0.899	0.892
		ASUWO	63,285	0.992	0.875	0.893
		SAE	63,647	0.993	0.990	0.991

The following points can be seen from Table 5: 1) The proposed method can solve the imbalanced point cloud classification problem well.

2) Three oversampling methods have improved the values of TPR and PPV, which shows that oversampling is useful to improve the classification effect of positive class. SAE has the most significant improvement effect, as it obtained higher TPR and PPV values than the other two methods.

3) Three methods have little effect on ACC value. Because ACC has become less important in imbalanced classification. If the positive samples are less than 1.0%, the accuracy can still reach 99% even if all positive samples are divided into negative class, but such classification has no practical significance.

4) Both SMOTE and ASUWO need negative class as references to generate samples of positive class, but SAE does not. So the synthetic samples of SAE are generated only by the features of positive class, and will not be affected by negative class samples.

Table 6. Classification results based on 7 UCI datasets.

Datasets	Size of classes	Positive		ACC		
		class	Unsampled	SMOTE	ASUWO	SAE
Balancescale	49,288,288	“1”	0.921	0.894	0.930	0.935
Blood	570,178	“2”	0.738	0.751	0.855	0.859
CMC	629,333,511	“2”	0.751	0.791	0.805	0.868
Ecoli	143,77,2,2,35,20,5,52	“8”	0.922	0.940	0.963	0.966
Vehicle	199,217,218,212	“1”	0.930	0.952	0.954	0.971
Vowel	72,89,172,151,207,180	“1”	0.948	0.962	0.970	0.975
Wine	59,71,48	“3”	0.938	0.988	0.992	0.994

In Table 6, positive classes of each data set are listed, while other classes are taken as negative classes. Therefore, the classification of each data set is an imbalanced binary classification problem. As can be seen from Table 6, the proposed method is also effective on other imbalanced data sets and can obtain better ACC values than the other two methods. So he proposed method performs better than the other two methods for both imbalanced multiclassification (Table 5) and binary classification problems (Table 6).

5. Conclusions

Through the research on classification of LiDAR point cloud in urban scenes, we find that the phenomenon of imbalanced class is quite common. However, existing classification methods are almost based on the assumption that each class is balanced, and rarely directly use the geometric coordinates and intensity information of point clouds as classification features. In order to make up for the lack of existing research, we propose an oversampling method based on SAE for imbalanced point cloud classification. Only 4 features (three-dimensional coordinates and intensity information) are selected to participate in the classification, which ensures the accuracy of classification and reduces the complexity of calculation. And combined with the optimization of classifier, the classification performance of imbalanced point cloud is improved.

Our research also has some shortcomings. 1) It focuses on the improvement effect of positive class, and the improvement of ACC value is not significant enough. 2) The proposed method is an improvement of machine learning-based method for point cloud classification, but does not involve the optimization of deep learning-based method. That is also the direction of our further research in the future.

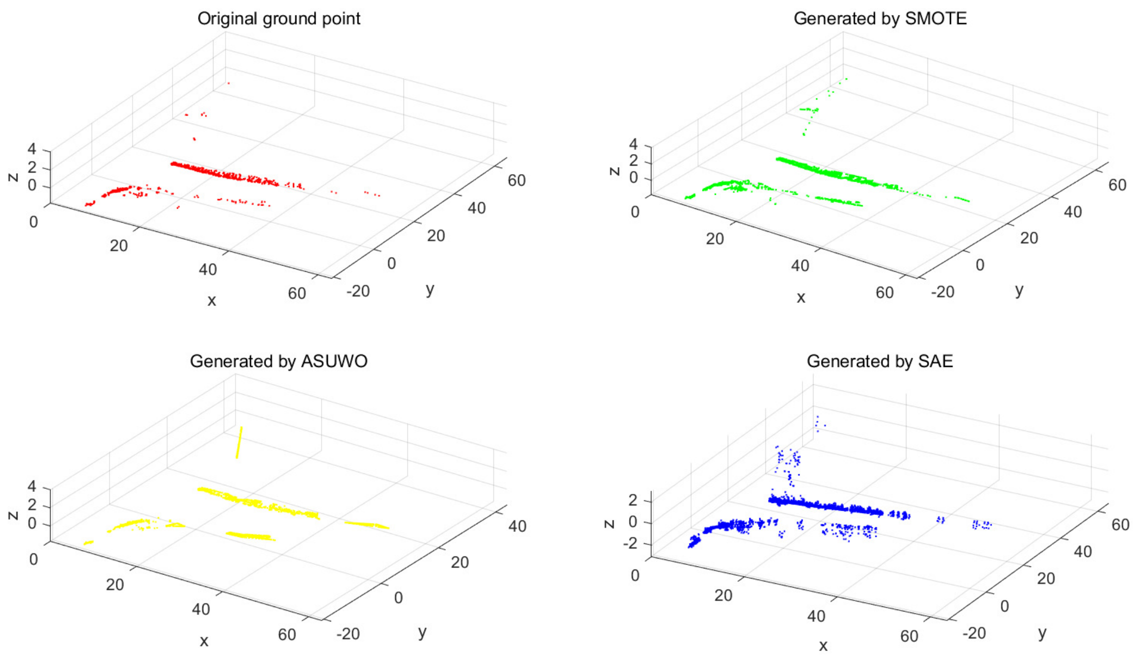

Figure 7. Ground points before and after oversampling.

Conflict of interest

The authors declare there is no conflicts of interest.

References

1. X. L. Li, C. Liu, Z. N. Wang, X. H. Xie, D. Li, L. J. Xu, Airborne LiDAR: state-of-the-art of system design, technology and application, *Meas. Sci. Technol.*, **32** (2020). https://doi.org/10.1088/1361-6501/abc867

2. Y. Benoist, P. Foulon, F. Labourie, Flots d'Anosov a distributions stable et instable differentiables, (French) [Anosov flows with stable and unstable differentiable distributions, *J. Am. Math. Soc.*, **5** (1992), 33–74. https://doi.org/10.1090/S0894-0347-1992-1124979-1

3. M. Beland, G. Parker, B. Sparrow, D. Harding, L. Chasmer, S. Phinn, et al., On promoting the use of lidar systems in forest ecosystem research, *For. Ecol. Manage.*, **450** (2019). https://doi.org/10.1109/10.1016/j.foreco.2019.117484

4. L. Mei, T. Ma, Z. Zhang, R. N. Fei, K. Liu, Z. F. Gong, et al., Experimental calibration of the overlap factor for the pulsed atmospheric lidar by employing a collocated Scheimpflug lidar, *Remote Sens.*, **12** (2020). https://doi.org/10.1016/10.3390/rs12071227

5. S. Muckenhuber, H. Holzer, Z. Bockaj, Automotive lidar modelling approach based on material properties and lidar capabilities, *Sensors*, **20** (2020). https://doi.org/10.3390/s20113309

6. A. Ulvi, Documentation, Three-Dimensional (3D) Modelling and visualization of cultural heritage by using Unmanned Aerial Vehicle (UAV) photogrammetry and terrestrial laser scanners, *Int. J. Remote Sens.*, **42** (2021), 1994–2021. https://doi.org/10.1080/01431161.2020.1834164

7. W. Song, S. H. Zou, Y. F. Tian, S. Fong, K. Cho, Classifying 3D objects in LiDAR point clouds with a back-propagation neural network, *Hum.-centric Comput. Inf. Sci.*, **8** (2018). https://doi.org/10.1186/s13673-018-0152-7

8. Y. Li, G. F. Tong, X. C. Du, X. Yang, J. J. Zhang, L. Yang, A single point-based multilevel features fusion and pyramid neighborhood optimization method for ALS point cloud classification, *Appl. Sci.*, **9** (2019). https://doi.org/10.3390/app9050951

9. T. B. Sun, J. H. Liu, J. M. Kan, T. T. Sui, A study on the classification of vegetation point cloud based on random forest in the straw checkerboard barriers area, *J. Intell. Fuzzy Syst.*, **41** (2021), 4337–4339. https://doi.org/10.3233/JIFS-189694

10. Z. S. Liu, W. Song, Y. F. Tian, S. M. Ji, Y. Sung, L. Wen, et al., Vb-net: Voxel-based broad learning network for 3d object classification, *Appl. Sci.*, **10** (2020). https://doi.org/10.3390/app10196735

11. L. Wang, Y. X. Liu, S. M. Zhang, J. X. Yan, P. J. Tao, Structure-aware convolution for 3D point cloud classification and segmentation, *Remote Sens.*, **12** (2020), 294–302. https://doi.org/10.3390/rs12040634

12. C. C. Lin, C. H. Kuo, H. T. Chiang, CNN-Based Classification for Point Cloud Object with Bearing Angle Image, *IEEE Access*, **22** (2022), 1003–1011. https://doi.org/10.1109/JSEN.2021.3130268

13. X. Li, L. Zhang, Unbalanced data processing using deep sparse learning technique, *Future Gener. Comput. Syst.*, **125** (2021), 480–484. https://doi.org/10.1016/j.future.2021.05.034

14. X. Y. Wang, L. P. Jing, Y. L. Lyu, M. Z. Guo, T. Y. Zeng, Smooth Soft-Balance Discriminative Analysis for imbalanced data, *Knowl.-Based Syst.*, **228** (2020). https://doi.org/10.1016/j.knosys.2020.106604

15. J. N. Wei, H. S. Huang, L. G. Yao, Y. Hu, Q. S. Fan, D. Huang, IA-SUWO: An Improving Adaptive semi-unsupervised weighted oversampling for imbalanced classification problems, *Knowl.-Based Syst.*, **203** (2020). https://doi.org/10.1016/j.knosys.2020.106116

16. W. W. Ng, S. C. Xu, J. J. Zhang, X. Tian, T. W. Rong, S. Kwong, Hashing-based undersampling ensemble for imbalanced pattern classification problems, *IEEE Trans. Cybern.*, **52** (2020), 1269–1279. https://doi.org/10.1109/TCYB.2020.3000754

17. H. Kaur, H. S. Pannu, A. K. Malhi, A systematic review on imbalanced data challenges in machine learning: Applications and solutions, *ACM Comput. Surv.*, **52** (2019). https://doi.org/10.1145/3343440

18. N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, SMOTE: synthetic minority oversampling technique, *J. Artif. Intell. Res.*, **16** (2002), 321–357. https://doi.org/10.1613/jair.953

19. I. Nekooeimehr, S. K. Lai-Yuen, Adaptive semi-unsupervised weighted oversampling (A-SUWO) for imbalanced datasets, *Expert Syst. Appl.*, **46** (2016), 405–416. https://doi.org/10.1016/j.eswa.2015.10.031

20. M. Galar, A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches, *IEEE Trans. Syst., Man, Cybern.*, **42** (2012), 463–484. https://doi.org/10.1109/TSMCC.2011.2161285

21. H. I. Lin, M. C. Nguyen, Boosting minority class prediction on imbalanced point cloud data, *Appl. Sci.*, **10** (2020). https://doi.org/10.3390/app10030973

22. B. E. Aissou, A. B. Aissa, A. Dairi, F. Harrou, A. Wichmann, M. Kada, Building roof superstructures classification from imbalanced and low density airborne LiDAR point cloud, *IEEE Sens. J.*, **21** (2021), 14960–14976. https://doi.org/10.1109/JSEN.2021.3073535

23. T. Kogut, A. Tomczak, A. Sowik, T. Oberski, Seabed modelling by means of airborne laser bathymetry data and imbalanced learning for offshore mapping, *Sensors*, **22** (2022), 14960–14976. https://doi.org/10.3390/s22093121

24. P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P. Manzagol, L. Bottou, Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion, *J. Mach. Learn. Res.*, **11** (2010). https://doi.org/10.1016/j.mechatronics.2010.09.004

25. H. L. Gong, S. B. Cheng, Z. Chen, Q. Li, C. Quilodrán-Casas, D. H. Xiao, et al., An efficient digital twin based on machine learning SVD autoencoder and generalised latent assimilation for nuclear reactor physics, *Ann. Nucl. Energy*, **179** (2022). https://doi.org/10.1016/j.anucene.2022.109431

26. S. B. Cheng, J. H. Chen, C. Anastasiou, P. Angeli, O. K. Matar, Y. Guo, et al., Generalised latent assimilation in heterogeneous reduced spaces with machine learning surrogate models, *J. Sci. Comput.*, **94** (2023). https://doi.org/10.1007/s10915-022-02059-4

27. S. Langarica, F. Nunez, Contrastive blind denoising autoencoder for real time denoising of industrial IoT sensor data, *Eng. Appl. Artif. Intell.*, **120** (2023). https://doi.org/10.1016/j.engappai.2023.105838

28. T. Liu, Y. C. Jin, S. Wang, Q. W. Zheng, G. A. Yang, Denoising method of weak fault acoustic emission signal under strong background noise of engine based on autoencoder and wavelet packet decomposition, *Struct. Health Monit.*, 2023. https://doi.org/10.1177/14759217221143547

29. Z. Salekshahrezaee, J. L. Leevy, T. M. Khoshgoftaar, The effect of feature extraction and data sampling on credit card fraud detection, *J. Big Data*, **10** (2023). https://doi.org/10.1186/s40537-023-00684-w

30. G. Long, Z. X. Zhang, Deep encrypted traffic detection: An anomaly detection framework for encryption traffic based on parallel automatic feature extraction, *Comput. Intell. Neurosci.*, **2023** (2023). https://doi.org/10.1155/2023/3316642

31. X. S. Du, J. Yu, Z. Chu, L. N. Jin, J. Y. Chen, Graph autoencoder-based unsupervised outlier detection, *Inf. Sci.*, **608** (2022), 532–550. https://doi.org/10.1016/j.ins.2022.06.039

32. A. Abhaya, B. K. Patra, An efficient method for autoencoder based outlier detection, *Expert Syst. Appl.*, **213** (2023). https://doi.org/10.1016/j.eswa.2022.118904

33. C. K. Ma, Y. J. Park, A new instance density-based synthetic minority oversampling method for imbalanced classification problems, *Eng. Optimiz.*, **54** (2022), 1743–1757. https://doi.org/10.1080/0305215X.2021.1982929

34. H. Han, W. Y. Wang, B. H. Mao, Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning, *Adv. Intell. Comput.*, **644** (2005), 878–887. https://doi.org/10.1007/11538059-91

35. C. Bunkhumpornpat, K. Sinapiromsaran, C. Lursinsap, Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem, *Adv. Knowl. Discovery Data Min.*, **5476** (2009), 475–482. https://doi.org/10.1007/978-3-642-01307-2-43

36. G. Douzas, F. Bacao, F. Last, Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE, *Inf. Sci.*, **465** (2018). https://doi.org/10.1016/j.ins.2018.06.056

37. W. W. Ng, G. J. Zeng, J. J. Zhang, D. S. Yeung, W. Pedrycz, Dual autoencoders features for imbalance classification problem, *Pattern Recognit.*, **60** (2016), 875–889. https://doi.org/10.1016/j.patcog.2016.06.013

38. J. F. Xu, Y. J. Zhang, D. Q. Miao, Three-way confusion matrix for classification: A measure driven view, *Inf. Sci.*, **507** (2020), 772–794. https://doi.org/10.1016/j.ins.2019.06.064

39. C. Seiffert, T. M. Khoshgoftaar, J. V. Hulse, A. Napolitano, RUSBoost: A hybrid approach to alleviating class imbalance, *IEEE Trans. Syst., Man, Cybernet.-Part A: Syst. Hum.*, **40** (2010), 185–197. https://doi.org/10.1109/TSMCA.2009.2029559

40. X. R. Jin, Z. X. Ding, T. Li, J. Xiong, G. Tian, J. B. Liu, Comparison of MPL-ANN and PLS-DA models for predicting the severity of patients with acute pancreatitis: An exploratory study, *Am. J. Emerg. Med.*, **44** (2021), 85–91. https://doi.org/10.1016/j.ajem.2021.01.044

41. H. Zhou, K. M. Yu, Y. C. Chen, H. P. Hsu, A hybrid feature selection method RFSTL for manufacturing quality prediction based on a high dimensional imbalanced dataset, *IEEE Access*, **9** (2021), 29719–29735. https://doi.org/10.1109/ACCESS.2021.3059298

42. R. Blanquero, E. Carrizosa, C. Molero-Río, D. R. Morales, Optimal randomized classification trees, *Comput. Oper. Res.*, **132** (2021). https://doi.org/10.1016/j.cor.2021.105281

43. Q. A. Al-Haija, M. Krichen, W. A. Elhaija, Machine-learning-based darknet traffic detection system for IoT applications, *Electronics*, **11** (2022). https://doi.org/10.3390/electronics11040556

AIMS Press