International Journal of Innovative
Computing, Information and Control
Volume **18**, Number **3**, June **2022**

ICIC International ©2022 ISSN 1349-4198
pp. **703–721**

ACWGAN: AN AUXILIARY CLASSIFIER WASSERSTEIN GAN-BASED OVERSAMPLING APPROACH FOR MULTI-CLASS IMBALANCED LEARNING

Chen Liao[1] and Minggang Dong[2,*]

[1]School of Information Science and Engineering
[2]Guangxi Key Laboratory of Embedded Technology and Intelligent System
Guilin University of Technology
No. 319, Yanshan Street, Yanshan District, Guilin 541006, P. R. China
2120190618@glut.edu.cn; *Corresponding author: 2000025@glut.edu.cn

Received November 2021; revised March 2022

Abstract. *Learning from multi-class imbalance data is a common but challenging task in machine learning community. Oversampling method based on Generative Adversarial Networks (GAN) is an effective countermeasure. However, due to the scarce number of trainable minority samples, existing methods may produce noise or low-quality minority samples; besides, they may suffer from mode collapse. To address the issues, we propose an Auxiliary Classifier Wasserstein Generative Adversarial Networks (ACWGAN) for imbalanced dataset. An independent auxiliary classifier is introduced to help discriminator determine whether the minority samples match the corresponding labels, more importantly, to improve the quality of generated minority samples. Furthermore, we use Wasserstein distance instead of Jensen-Shannon divergence in ACWGAN as the distance measure of the probability distribution to alleviate the mode collapse. Extensive experimental testing is performed on 16 multi-class imbalanced benchmarks and two real imbalanced datasets in comparison with several popular oversampling approaches. The experiment result demonstrates that our method is superior to other oversampling approaches.*

Keywords: Imbalanced learning, Oversampling approach, GAN, Multi-class, Independent auxilary classifier

1. **Introduction.** In the field of machine learning, classification from imbalanced data is an important but challenging problem for research community. Imbalanced learning can be defined as a learning problem from a binary or multi-class dataset where the certain classes (majority classes) possess much more instances than other classes (minority classes) [1]. Imbalanced learning has many real-world applications, such as fraud detection of credit card [2], cancer medical diagnosis [3], computer vision [4] and protein identification in the biomedical field [5].

Traditional classification method usually assumes the dataset is balanced. When the dataset is imbalanced, the classifier seeking to minimize overall training errors tends to bias toward the majority class. In this situation, the minority classes samples are easy to be misclassified. However, the minority classes often have smaller number and higher cost of misclassification, resulting in the standard methods performing poorly. Therefore, the key to solving imbalanced problem is to improve the classification accuracy of minority classes without reducing the majority class.

At present, there are three main approaches to addressing the imbalanced classification problem: data level methods, algorithm level methods and hybrid methods. The data level

DOI: 10.24507/ijicic.18.03.703

methods balance the data distribution through oversampling [1] or undersampling [6]. The purpose of oversampling is to increase samples for the minority class, while undersampling reduces the samples for the majority class. The algorithm level methods directly modify traditional algorithms to enhance the ability of them to handle the imbalance problem [7]. Hybrid methods combine the advantages of previous two methods by simultaneously sampling in the dataset and modifying the algorithm [8].

In this paper, what we focus is the oversampling method, which generates artificial instances for minority classes to balance the dataset. Synthetic Minority Oversampling Techniques (SMOTE) [1] is the standard oversampling method, and the algorithm uses the sample in the minority class as a seed, seeks for the k-nearest minority neighbors, and then inserts the generated sample between it and the seed sample according to a certain ratio. Although SMOTE decreases the risk of overfitting effectively, it can also raise the probability of overgeneration. To avoid this scenario, various oversampling methods were proposed following the steps of SMOTE, such as Borderline-SMOTE [9], ADASYN [10], Kmean-SMOTE [11], and MWMOTE [12]. Besides, some oversampling methods for multi-class imbalanced dataset were proposed [13,14]. However, most above existing traditional oversampling methods generate synthetic samples along the line segment that joins minority class samples, and do not consider the whole data distribution.

Recently, Generative Adversarial Networks (GAN) [15] has become a popular generative model due to its powerful ability to capture the real data distribution. Douzas and Bacao first applied conditional GAN (cGAN) [16] as an oversampling approach to addressing the imbalanced problem [17], and improved the performance of classification effectively. The success of cGAN has inspired several improvements: Aiming at addressing the possible mode collapse [18] and unstable training [19] in cGAN, Conditional Wasserstein GAN with Gradient Penalty (CWGAN-GP) attempted to introduce objective function of WGAN-GP [20] into cGAN [21]. Auxiliary Classifier GAN (ACGAN) added an additional classification output in the output player in discriminator to improve the quality of generated samples [22]. Works like [23,24] extended the output layer of the discriminator and applied it to handling the imbalanced image problem.

In summary, the existing GAN-based oversampling methods mainly have the following insufficiencies. Firstly, the imbalance problem also exists in the training process of GAN, the generator may be biased towards the majority class, resulting in poor quality of the generated minority samples. Secondly, most of them suffer from mode collapse, and the diversity of generated samples is insufficient.

Based on the above considerations, we propose Auxiliary Classifier Wasserstein Generative Adversarial Networks (ACWGAN) as a novel GAN model. It has an extra independent auxiliary classifier that can help discriminator to better distinguish whether the minority samples match the corresponding labels; in this way, the ability of generator to generate minority samples will be enhanced. Compared with the Jensen-Shannon divergence used in traditional GAN, the Wasserstein distance proposed by [19] can measure the distance between probability distributions more smoothly. Therefore, we use Wasserstein distance as the distance metric of ACWGAN to alleviate mode collapse in training and increase the diversity of generated samples. The performance of ACWGAN is compared against seven popular oversampling approaches over 16 multi-class benchmark datasets and two real world datasets. The experimental results reveal that ACWGAN is superior to other popular oversampling approaches.

The main contributions of the paper are summarized as follows.

- A novel GAN structure that has an independent classifier is proposed, to enhance the ability of generator to generate minority samples with more features for classification.

- The distance metric that measures the distribution of real data and generated data in GAN is replaced with Wasserstein distance, which effectively improves the diversity and quality of generated minority samples.
- Extensive experiments are performed on 16 multi-class imbalanced benchmarks and two real imbalanced datasets in comparison of several popular oversampling approaches.

The remainder of this paper is organized as follows. In Section 2, a brief introduction of related previous works is given. Section 3 presents our method. The experimental result is shown in Section 4. Section 5 concludes the paper.

2. Related Works.

In this section, we provide a brief summary of GAN framework and its related variants.

2.1. GAN, cGAN and ACGAN.

Generative Adversarial Networks (GAN) [15] is a powerful generative model that has been successfully applied to image generation, super-resolution and other fields. The idea of GAN is derived from game theory, in which a generator G and a discriminator D are trying to outperform each other. The objective of generator is to generate fake samples and fool the discriminator. The objective of the discriminator is to determine whether the sample is from a real dataset or generated by the generator. More formally, for the generator G, $G(z)$ denotes the synthetic samples generated by G. For the discriminator D, $D(x)$ represents the probability that x came from the real data. D and G play the following two-player min-max game with value function $V(D, G)$:

$$\min_D \max_G V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \tag{1}$$

The generator and the discriminator update the parameters by their respective objective functions during the adversarial learning. Finally, the discriminator cannot judge whether the samples are real or fake, and the generator can generate realistic samples.

Since GAN cannot control the mode of generated data, conditional GAN (cGAN) [16] introduces additional information into both generator and discriminator respectively to guide the data generation. Hence, the objective function of cGAN is defined as Equation (2):

$$\min_D \max_G V(D,G) = \mathbb{E}_{x,y\sim p(x,y)}[\log D(x \mid y)] + \mathbb{E}_{z\sim p_z(z),y\sim p(y)}[\log(1 - D(G(z \mid y)))] \tag{2}$$

where y is the conditional variable, and it is combined with noise z and data x as the input of generator and discriminator respectively.

Auxiliary Classifier GAN (ACGAN) [22] is a variant of cGAN. It expands the function of the discriminator to simultaneously distinguish real or false and classify. Therefore, the loss function of ACGAN consists of both discriminator loss L_S and classifier loss L_C:

$$L_S = E[\log P(S = real|X_{real})] + E[\log P(S = fake|X_{fake})] \tag{3}$$

$$L_C = E[\log P(C = c|X_{real})] + E[\log P(C = c|X_{fake})] \tag{4}$$

where c denotes the class label of sample and $C(c|x)$ denotes a probability over the class label calculated by the classifier C in discriminator, L_S is the log-likelihood of the correct source and L_C is the log-likelihood of the correct class. The discriminator is trained to maximize $L_S + L_C$ while generator is trained to maximize $L_C - L_S$.

However, as cGAN and ACGAN are extensions of GAN, they also use Jensen-Shannon Divergence (JSD) [15] as the distance metric, so they exhibit the same problematic behaviors: mode collapse and unstable training [18]. It is worth noting that the problem of mode collapse will be worse in imbalanced dataset.

2.2. **WGAN and WGAN-GP.** Aiming at addressing the problems mentioned above, Wasserstein GAN is proposed by Arjovsky et al. [19], which optimizes Wassertein distance (also known as Earth Mover (EM) distance) rather than JSD. Compared with the JSD that may not be able to provide useful gradients for the generator, the EM distance is not affected by vanishing gradient, and can more appropriately measure the distance between two distributions. Some previous works have proved that WGAN can effectively alleviate the problem of mode collapse in GAN [20,21]. The definition of EM distance is provided in Equation (5).

$$W\left(\mathbb{P}_r, \mathbb{P}_g\right) = \inf_{\gamma \in \Pi\left(\mathbb{P}_r, \mathbb{P}_g\right)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|] \tag{5}$$

where $\Pi\left(\mathbb{P}_r, \mathbb{P}_g\right)$ denotes the set of all joint distributions $\gamma(x, y)$ of real data distribution $\mathbb{P}_r$ and generated data distribution $\mathbb{P}_g$, and $W\left(\mathbb{P}_r, \mathbb{P}_g\right)$ is defined as the minimum cost of transporting mass from $\mathbb{P}_r$ into $\mathbb{P}_g$. However, Equation (5) cannot be solved directly, the K-Lipschitz functions are used to determine the supremum. Equation (5) could be redefined as

$$W\left(\mathbb{P}_r, \mathbb{P}_g\right) = \frac{1}{K} \sup_{||f_w||L \leqslant K} \mathbb{E}_{x\sim\mathbb{P}_r}[f_w(x)] - \mathbb{E}_{x\sim\mathbb{P}_g}[f_w(x)] \tag{6}$$

where K denotes K-Lipschitz for constant K, and $f_w(x)$ denotes the weight of discriminator. To ensure $||f_w||L \leq K$, the weights of the discriminator are clipped into $[-c, c]$. The new loss function of discriminator is defined as Equation (7).

$$L = \mathbb{E}_{x\sim\mathbb{P}_r}\left[f_w(x)\right] - \mathbb{E}_{x\sim\mathbb{P}_g}\left[f_w(x)\right] \tag{7}$$

However, WGAN still cannot converge to high quality solutions, and [20] pointed out that weight clipping makes WGAN unable to adapt to the complex distribution, leading to gradient vanishing or exploding. Thus, in order to satisfy the Lipschitz constraint without weight clipping, they proposed the Wasserstein GAN with Gradient Penalty (WGAN-GP). The loss function of discriminator in WGAN-GP is defined as follows:

$$L = \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})] - \mathbb{E}_{x\sim\mathbb{P}_r}[D(x)] + \lambda\mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}\left[\left(\left\|\nabla_{\hat{x}}D(\hat{x})\right\|_2 - 1\right)^2\right] \tag{8}$$

where λ is the gradient penalty coefficient and $\hat{x}$ denotes random sampling along straight lines between the real data distribution $\mathbb{P}_r$ and generated data distribution $\mathbb{P}_g$:

$$\hat{x} = \varepsilon x_r + (1 - \varepsilon)x_g,\ x_g \sim \mathbb{P}_g,\ x_r \sim \mathbb{P}_r,\ \varepsilon \sim \mathit{Uniform}[0, 1] \tag{9}$$

In this work, we couple ACGAN and WGAN-GP. More specifically, in the proposed ACWGAN, we apply the Wasserstein distance as the distance metric in ACGAN to measuring the distance between real data distribution and generated data distribution. Since our goal is to generate samples for the specific minority classes, we also modify the structure of ACGAN and introduce an independent classifier to help the generator to generate samples with more classification features.

3. The Proposed Method.

In this section, we describe the details of our proposed method. We first introduce the motivation for improving the model in Subsection 3.1. Afterward, we present the proposed GAN model in Subsection 3.2.

3.1. **Motivation.** Conditional GAN (cGAN) [16] is a powerful subclass of generative models that has been successfully applied to binary imbalance problem [17,21]. When training multi-class imbalanced dataset, in order for the GAN to better distinguish between different classes explicitly, ACGAN with classification loss is a better alternative approach. However, the original ACGAN has some limits under imbalanced scenario. We next describe some insufficiencies of ACGAN under multi-class imbalanced scenario as the motivational examples of our work.

Limit 1: In ACGAN, the discriminator has two objectives: one is to judge whether the sample is real or fake, and the other is to classify the sample. However, as the minority class samples are scarce in multi-class imbalanced dataset, there is a conflict between the two goals [23]. Due to the lack of trainable minority samples, the discriminator may intuitively classify the generated minority samples as fake samples. Therefore, to optimize the loss function, the generator may mistakenly generate majority samples with minority class labels as input. In the meanwhile, the discriminator may consider the generated samples as real samples and ignore the labels mismatch, encouraging the generator to further generate such noise samples that may damage the performance of classifier.

Limit 2: Due to the additional classification loss, ACGAN tends to generate samples far away from the decision boundary. Next, we use an example to illustrate this problem. For the convenience, we convert the ecoli dataset from UCI repository into a binary imbalanced dataset. Then, the generator in ACGAN generates samples for the minority class in the dataset. Figure 1 depicts the distribution of the samples generated by ACGAN after PCA dimensionality reduction. As is shown in the figure, even in the binary-class imbalance scenario, synthesized samples are almost all concentrated in the safe area far away from the decision boundary. It means that ACGAN completely abandons the generation of borderline samples. Therefore, in the multi-class imbalance problem, the more complex data distribution will squeeze the generated samples to a very limited safe area, thereby reducing the diversity of generated samples. More seriously, that may cause mode collapse. Another problem with ACGAN is that the classifier and the discriminator share the same network. When optimizing the network, the two may affect each other, resulting in insufficient network optimization.

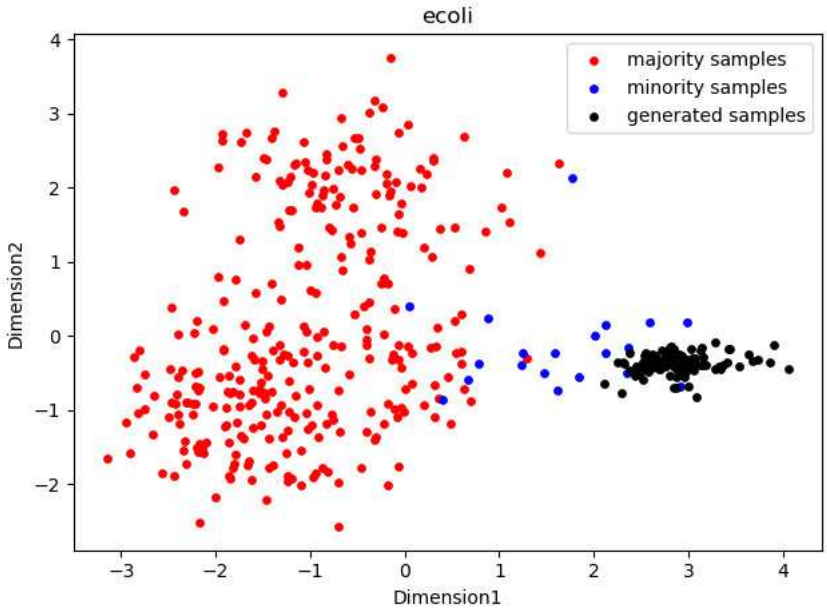


FIGURE 1. The distribution of synthetic data generated by generator in ACGAN

3.2. **ACWGAN.** Motivated by the problems stated in the previous subsection, we propose Auxiliary Classifier Wasserstein GAN (ACWGAN), a novel GAN model that can generate high quality samples for minority classes for multi-class imbalanced datasets. It can be constructed by applying the objective function of WGAN-GP to a modified version of ACGAN.

In view of the drawback of ACGAN, we first modify the structure of ACGAN and separate the classifier from the discriminator. Thus, the Modified ACGAN (MACGAN) has three independent networks: a generator, a discriminator and a classifier. Comparing with sharing weights, the independent discriminator and classifier can focus more on their functions without being affected by each other. The classification ability of independent classifier will also be enhanced. Since similar generating samples do not improve the classification performance of the classifier, in the MACGAN, we only train the classifier with real dataset, the training process of the classifier is independent of adversarial training.

However, ACGAN is more likely to fall into mode collapse due to the additional classification loss. Though MACGAN improves the performance of ACGAN to a certain extent, it does not alleviate the serious mode collapse problem. The Wasserstein distance is proved to be a more appropriate measure of the distance between probability distributions [20], and WGAN-GP using Wassertein distance effectively alleviates mode collapse and improves the diversity of generated samples. [21,25] proved that it is still applicable in the imbalance problem. Therefore, ACWGAN is constructed by applying the object function of WGAN-GP to MACGAN. Figure 2 shows the structure of ACWGAN.

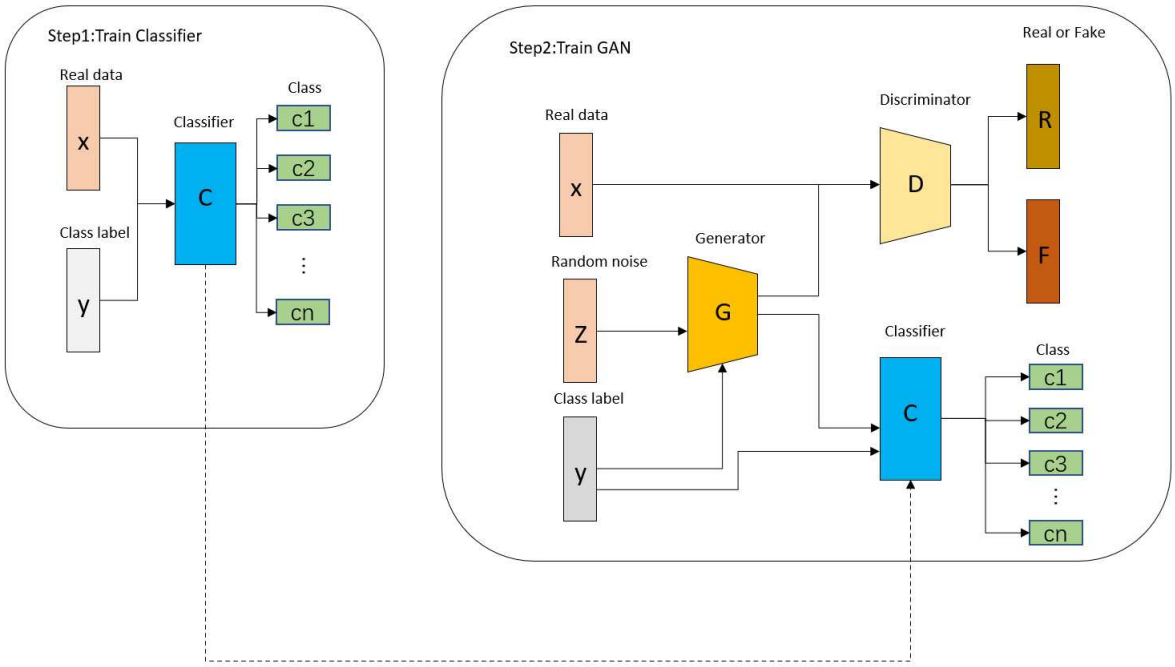


FIGURE 2. Structure of ACWGAN. Here x is the real data, y is a class label, z is a random noise vector, C is the classifier, D is the discriminator and G is the generator. R and F are the discriminator outputs representing the probability of a sample being real or fake, $c1, \ldots, cn$ are the classifier outputs representing the probabilities of samples belonging to class $c1, \ldots, cn$.

The loss functions of ACWGAN can be defined as follows:

$$L_D = \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})] - \mathbb{E}_{x\sim\mathbb{P}_r}[D(x)] + \lambda\mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}\left[\left(\|\nabla_{\hat{x}}D(\hat{x})\|_2 - 1\right)^2\right] \tag{10}$$

$$\hat{x} = \varepsilon x_r + (1-\varepsilon)x_g,\ x_g \sim \mathbb{P}_g,\ x_r \sim \mathbb{P}_r,\ \varepsilon \sim \mathit{Uniform}[0,1] \tag{11}$$

$$L_C = -E_{x\sim\mathbb{P}_r(x),y_r\sim\mathbb{P}(y)}\left[\log C\left(y = y_r \mid x\right)\right] \tag{12}$$

$$L_{adv} = -\mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})] \tag{13}$$

$$L_{cls} = -E_{z\sim\mathbb{P}(z),y_g\sim\mathbb{P}(y)}\left[\log C\left(y = y_g \mid G\left(z, y_g\right)\right)\right] \tag{14}$$

$$L_G = \alpha L_{adv} + (1-\alpha)L_{cls} \tag{15}$$

where L_D, L_G and L_C are the loss functions of discriminator D, generator G and classifier C. x denotes the real sample sampled from real data distribution $\mathbb{P}_r$, $\tilde{x}$ is the synthetic sample generated by generator, and $\hat{x}$ is random sampling along straight lines between the real data distribution $\mathbb{P}_r$ and generated data distribution $\mathbb{P}_g$. y_r denotes the class label of sampled data x from $\mathbb{P}_r(x)$, and y_g denotes the sampled label from $\mathbb{P}_r(y)$. The generator G is trained to minimize both adversarial loss L_{adv} and classification loss L_{cls}, that means the generator should generate samples that will be both judged real by the discriminator and classified correctly by the classifier. α denotes the hyper-parameter that controls the importance of classification loss and adversarial loss.

Algorithm 1 illustrates the training process of ACWGAN. In the first step, we train the auxiliary classifier with real dataset to improve the ability of classifier. In the second step, the weight of classifier will be fixed. The trained classifier will classify the generated samples, and the discriminator is trying to distinguish whether the samples are real or fake, the generator attempts to generate samples that can be discriminated as real by discriminator and classified as the class corresponding to the input label. It is worth noting that when training the generator, in order to ensure the generator's ability to generate minority samples, each label will be generated with the same probability.

Algorithm 1. ACWGAN, we use default values of $\lambda = 10$, $n_d = 5$, $\alpha = 0.66$

Require: S: the train dataset; m: minibatch size for per epoch; l: latent dimension; α: hyperparameter that controls the importance of classifier and discriminator.
Require: Initial discriminator D parameters ω, generator G parameters θ and classifier C parameters μ.
Ensure: A trained Generator Network G
  while μ has not converged **do**
    for $i = 1, 2, \ldots, m$ **do**
      Sample real data $\boldsymbol{x} \sim \mathbb{P}_r$ and their corresponding labels $\boldsymbol{y} \sim \mathbb{P}_r$.
      Update C by $\mu \leftarrow \mathrm{Adam}\big(\nabla_\mu \frac{1}{m}\sum_{i=1}^{m} L_C, \mu\big)$.
    end for
  end while
  while θ has not converged **do**
    for n_d steps **do**
      for $i = 1, 2, \ldots, m$ **do**
        Sample real data $\boldsymbol{x} \sim \mathbb{P}_r$, latent variable $\boldsymbol{z} \sim p(z)$, a random number $\varepsilon \sim U[0, 1]$.
        $\widetilde{\boldsymbol{x}} \leftarrow G(z)$
        $\hat{\boldsymbol{x}} \leftarrow \varepsilon \boldsymbol{x} + (1-\varepsilon)\widetilde{\boldsymbol{x}}$
        $L^{(i)} \leftarrow D_w(\widetilde{\boldsymbol{x}}) - D_w(\boldsymbol{x}) + \lambda\left(\left\|\nabla_{\hat{\boldsymbol{x}}} D_w(\hat{\boldsymbol{x}})\right\|_2 - 1\right)^2$
      end for
      Update D by $\omega \leftarrow \mathrm{Adam}\big(\nabla_\omega \frac{1}{m}\sum_{i=1}^{m} L^{(i)}, \omega\big)$.
    end for
    Sample a batch of latent variables $Z = \left\{z_{(1)}, z_{(2)}, \ldots, z_{(m)}\right\}$ from $p(z)$.
    Generate a batch of labels $Y = \left\{y_{(1)}, y_{(2)}, \ldots, y_{(m)}\right\}$ with equal probability from each class.
    Update G by $\theta \leftarrow \mathrm{Adam}\big(\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} \alpha L_{adv} + (1-\alpha)L_{cls}, \theta\big)$
  end while

Referring to the parameter settings of [20], we set the default values of λ and n_d in Algorithm 1 to 10 and 5, respectively. α is the hyper-parameter that we introduced to adjust the importance of adversarial loss and classification loss in generator, and a higher α means that we care more about whether the generated samples are realistic enough rather than whether the samples can be classified correctly. A lower α represents just the opposite. Through our experiments, the ratio of classification loss to adversarial loss can achieve a trade-off at $1 : 2$ in most datasets. At this time, the value of α is 0.66. However, for some specific datasets whose features are simple but difficult to classify, it is recommended to use a lower α value to make the generator focus more on the classification accuracy.

Once the ACWGAN is trained, the generator in ACWGAN can be used in oversampling. The detailed steps of oversampling could be illustrated as follows. First, we use the original imbalanced dataset to train ACWGAN. Next, the corresponding minority class labels are mixed into Gaussian noises input generator to generate samples for each minority class. Last, merge the generated samples with original dataset until the data is integrated into a new balanced dataset. When we obtain a balanced dataset after the oversampling process, we use the balanced dataset to train classification models.

As is shown in Figure 3, in the same binary-class imbalance scenario mentioned in Subsection 3.1, compared with ACGAN, the minority samples generated by generator in ACWGAN possess higher diversity. Furthermore, the distribution area of the generated samples is wider, and the generated samples also include the borderline samples close to the decision boundary. This is because in ACWGAN, the independent classifier has stronger classification performance, so the generator can generate samples closer to the decision boundary without being misclassified.

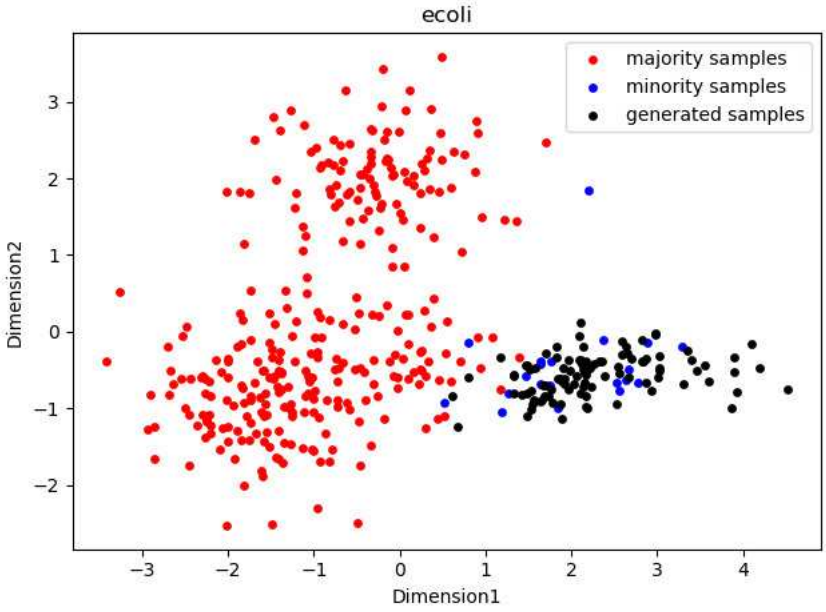


FIGURE 3. The distribution of synthetic data generated by generator in ACWGAN

In addition to the distance measurement, ACWGAN is quite different from original ACGAN: The independent classifier in ACWGAN only trains with real samples, so it will not learn the features of generated samples. Therefore, if the generator wants to generate samples that can be classified by the classifier correctly, it must learn the feature distribution that is similar to the real samples.

4. Experiment Study.

4.1. Setup.

In this subsection, the ACWGAN is evaluated using 16 benchmark multi-class imbalanced datasets. We first introduce the datasets and evaluation metrics in our experiment. Next, we briefly describe the experimental design. Finally, the parameter settings and the statistical tests design are detailed.

Datasets. In this work, 16 multi-class imbalanced benchmark datasets are used in our experiments. Table 1 shows a summary of the benchmark datasets. I, F, and C respectively denote the number of instances, features, and classes in the dataset. IR (Imbalance Ratio) is the imbalanced degree measured for multi-class imbalanced data [1], and it is defined as the ratio between the majority class and the smallest minority class. All the benchmark datasets are real datasets from UCI Machine Learning Repository [26], which vary in size, class distribution, feature number and IR to ensure a reliable assessment of performance.

Table 1. Description of the benchmark datasets from UCI Machine Learning Repository

Dataset	I	F	C	Class distribution	IR
Ecoli	327	7	4	143/77/52/35	4.08
Vowl5	990	10	5	180/90/360/270/90	4
Voice9	413	10	6	38/58/43/100/59/115	3.02
Yeast8	1484	8	8	463/25/35/44/81/163/244/429	17.16
Yeast52	982	8	5	463/25/35/429/30	17.16
SAT	6435	36	6	1358/626/707/1508/703	2.14
SAT4	4374	36	4	1553/626/707/1508	2.14
Plates-faults1	1941	27	7	158/190/391/72/55/402/673	12.23
Wine-red	1571	10	4	681/638/199/53	12.84
Wine-white	4873	10	5	2193/1457/880/175/163	13.45
Newthroid	215	5	3	150/35/30	5
Page-blocks	5445	10	4	4913/329/88/115	42.72
House5	506	13	5	36/123/239/77/31	7.7
Abalone8	2148	10	8	126/203/267/487/634/259/115/57	11.12
Abalone10	2297	10	10	57/115/259/391/634/487/126/103/67/98	11.12
Glass	214	9	6	70/76/17/13/9/29	8.4

Evaluation metrics. Precision, Recall and F-measure [28] are widely used single class metrics to measure the performance of classifier. Precision and Recall represent the exactness and completeness of prediction. F-measure combines both of them and represents their trade-off. Parameter β is used to adjust the relative importance of them, and is usually set to one.

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

$$F\text{-}measure = \frac{(1 + \beta_2)Recall \times Precision}{\beta_2 Recall + Precision} \tag{17}$$

In multi-class imbalanced problems, MG [29] and MAUC [30] are more popular evaluation metrics. They are defined as follows:

$$MG = \left(\prod_{i=1}^{|C|} Recall_{c_i} \right)^{\frac{1}{|C|}} \tag{18}$$

$$MAUC = \frac{2}{|C|(|C|-1)} \sum_{i<j} \left(\widehat{A}\left(c_i, c_j\right) \right) \tag{19}$$

where C is the set of classes, $Recall_{c_i}$ denotes the recall of class C_i, $\hat{A}(c_i, c_j) = \left[\hat{A}(i|j) + \hat{A}(j|i)\right]/2$, and $\hat{A}(i|j)$ denotes the probability that a randomly drawn member class c_j will have a lower estimated probability of belonging to class c_i than a randomly drawn member of class c_i.

In our experiments, we use mean F-measure in every class, MG and MAUC as our performance metrics for evaluating the performance of each oversampling method.

Experimental design and parameter settings. We first perform an ablation experiment on ACWGAN first to determine the effectiveness of several improvements in ACWGAN. Then ACWGAN is compared with those traditional oversampling methods based on SMOTE [1], including SMOTE, Borderline-SMOTE [9], K-means SMOTE [11] and generative models based on GAN, including cGAN [16], ACGAN [22], CWGAN-GP [25], and BAGAN [23]. To those traditional oversampling methods implemented by Python Library Imbalanced Learn, we use default parameter settings.

With respect to the generation models based on GAN, in order to ensure fairness, most of the hyperparameters such as network depth, and the number of neural units were the same. All the GAN models were two-layers, and the numbers of hidden layers in generator and discriminator were set to 200-200 and 100-100 respectively. Specially, the hidden layer of additional classifier in ACWGAN was set to 100-100. The dimension of noise space was set to 40. Neither Batch Normalization nor Dropout were applied to all networks, the batch size of each epoch was set to 64, and every GAN was trained for 2000 to 10000 epochs, all networks used ReLU as the activation function in the hidden layers and were trained with Adam optimizer in default settings. Besides, in CWGAN-GP and ACWGAN, the gradient penalty coefficient λ was set to 10, five D parameter was updated followed by the single G parameter. All the generative models above were implemented by Python with Google's open source framework TensorFlow.

Four frequently-used base classifiers are selected for our experiments to evaluate the above oversampling methods, including C4.5 Decision Tree (DT) [31], Gradient Boost Decision Tree (GBDT) [32], Support Vector Machine (SVM) [33], and K-Nearest Neighbor (KNN) [34]. We apply 5-fold cross-validation to the dataset. In each stage, the dataset is divided into 80% training data and 20% test dataset. We only use the training set to train the oversampling approaches. Next, the oversampling approaches synthesize new samples for the minority class. Then the synthetic samples and the original dataset are merged into a new balanced training dataset. We train the classifier with the balanced dataset. Finally, we evaluate the performance of oversampling approaches by comparing accuracy of the classifiers on the test dataset. The experimental procedure was repeated 5 times and the reported results include the average values between the experiments to reduce the bias.

Statistical tests. In our experiments, non-parametric tests are employed to analyze and compare whether there are significant differences between different oversampling methods. The Friedman test [35] and Holm's post hoc test [36] are used to evaluate our experimental results. The Friedman test is a non-parametric test for ranking all algorithms over all datasets, and its null hypothesis is that all algorithms are not significantly different. If the statistic exceeds the critical value of the specified significance level, the null hypothesis will be rejected. If the performances of the oversampling approaches are significantly different, the Holm's post hoc test is used to further distinguish whether each oversampling approach is similar. In this study, the specified level of significance α is set to 0.05.

4.2. **Results.** This subsection presents the experimental results of various oversampling methods under different classifiers. We first perform an ablation experiment on ACWGAN to determine which improvement in ACWGAN plays a leading role in improving the quality of the generated samples. Next, the performance evaluation results for 16 benchmark datasets are presented.

4.2.1. *Results of ablation experiments.* Table 2 shows the MAUC values of ACGAN, WACGAN, MACGAN, and ACWGAN in 8 multi-class imbalanced datasets, where MACGAN denotes ACGAN with independent classifier (without Wassertein distance), WACGAN represents ACGAN with Wasserstein distance (without independent classifier). As can be seen in the table, ACGAN has the lowest scores on almost all datasets. While MACGAN can often obtain higher scores than ACGAN, but for the most part it does

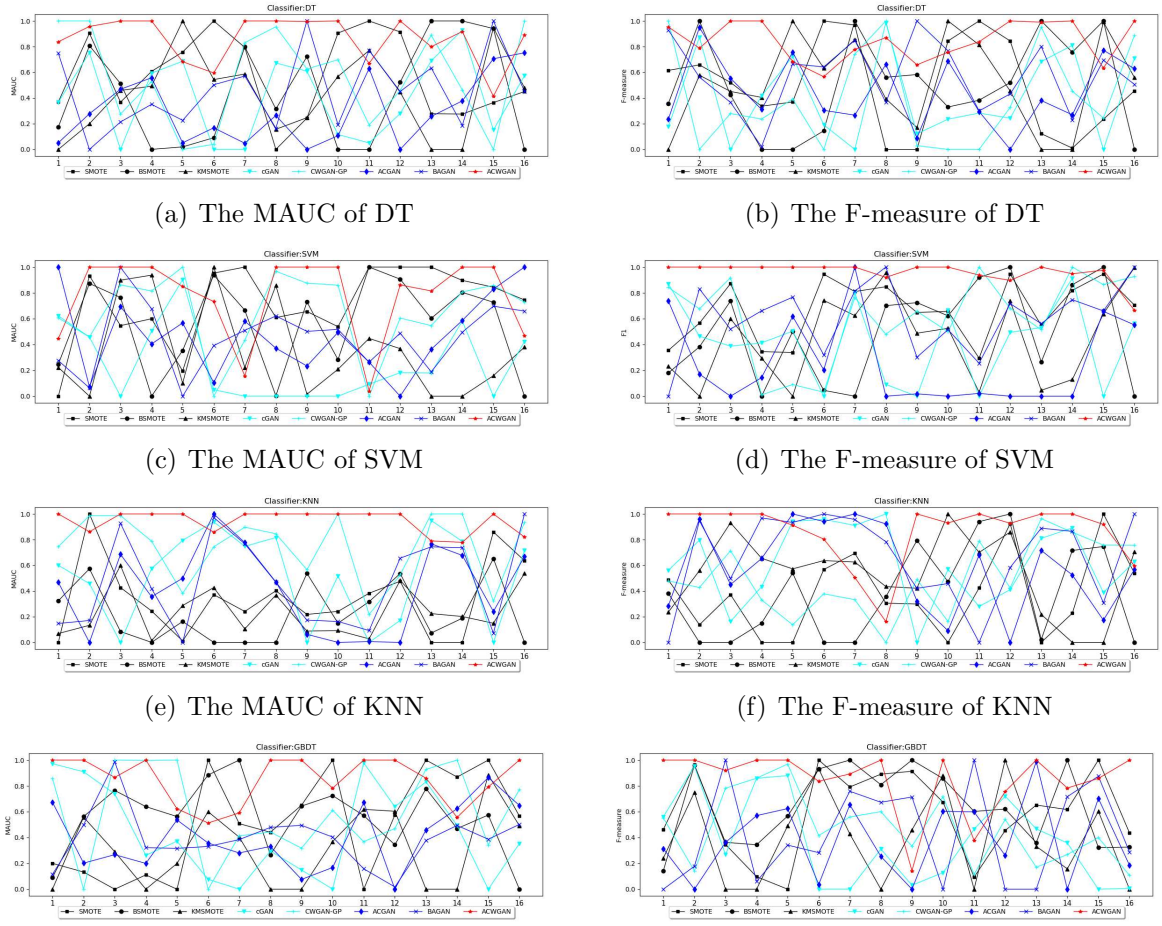
Table 2. The average score of different sampling methods when MAUC is used as the evaluation metric under 8 multi-class imbalanced datasets

Algorithms	Dataset	Classifier			
		DT	SVM	KNN	GBDT
ACGAN	Yeast8	0.708043	0.873261	0.805368	0.869095
WACGAN		0.733671	0.887031	0.839976	0.894299
MACGAN		0.733005	0.885532	0.840302	0.879175
ACWGAN		0.746084	0.895577	0.850107	0.895000
ACGAN	House5	0.772261	0.923427	0.889294	0.937513
WACGAN		0.796112	0.9421	0.906221	0.943818
MACGAN		0.795767	0.938498	0.910413	0.937745
ACWGAN		0.824451	0.950027	0.921493	0.950922
ACGAN	Voice9	0.609615	0.669383	0.645365	0.677051
WACGAN		0.603824	0.674085	0.653771	0.693388
MACGAN		0.601977	0.677877	0.652112	0.693063
ACWGAN		0.625562	0.677935	0.65414	0.690396
ACGAN	Wine-red	0.570878	0.743122	0.653513	0.717553
WACGAN		0.607393	0.763713	0.674795	0.725275
MACGAN		0.593806	0.751726	0.658197	0.730928
ACWGAN		0.60754	0.767708	0.693168	0.747055
ACGAN	Wine-white	0.575036	0.750754	0.646239	0.741113
WACGAN		0.598644	0.759999	0.691884	0.764834
MACGAN		0.575843	0.758996	0.650231	0.746683
ACWGAN		0.607952	0.764324	0.693115	0.763149
ACGAN	Page-Blocks	0.851404	0.956371	0.908819	0.971398
WACGAN		0.88105	0.965442	0.933844	0.968883
MACGAN		0.839296	0.956177	0.913142	0.975093
ACWGAN		0.882925	0.978218	0.93067	0.981352
ACGAN	Plates-faults1	0.73186	0.911253	0.855744	0.897188
WACGAN		0.761198	0.917814	0.860617	0.898802
MACGAN		0.735545	0.914005	0.855714	0.89932
ACWGAN		0.762882	0.918178	0.862625	0.906873
ACGAN	SAT4	0.867582	0.979495	0.966656	0.979051
WACGAN		0.878254	0.979151	0.967394	0.979505
MACGAN		0.874714	0.97933	0.966598	0.979405
ACWGAN		0.880559	0.978509	0.968028	0.980112

not score as well as WACGAN. Finally, in almost all cases, ACWGAN, which has both independent classifier and Wassertein distance as the distance metirc, has the highest scores.

From the ablation experiment in these datasets, we can draw conclusions: On the basis of the original ACGAN, using Wassertein distance or trained independent classifier alone can effectively improve the quality of generated samples. Of these two improvements, the using of Wassertein distance is more critical to improve the quality of generated samples (as WACGAN usually has higher scores than MACGAN). Finally, we observe that ACWGAN outperforms other methods on almost all classifiers, which proves that the effect of two improvements is cumulative.

4.2.2. *Results of performance evaluation for benchmark datasets.* Figure 4 shows the MinMax normalization value of MAUC and F-measure for the ACWGAN and other seven oversampling methods on 16 multi-class benchmark imbalanced datasets. The best performing method will be assigned the value of one while the worst method values zero in MinMax normalization. As is shown in the figure, under most indicators, ACWGAN achieves best results in 8 or more multi-class datasets. In the best case, ACWGAN performs better than all other oversampling methods in 11 of 16 datasets, the situation occurred twice, respectively in F-measure under SVM and MAUC under KNN.



(a) The MAUC of DT (b) The F-measure of DT

(c) The MAUC of SVM (d) The F-measure of SVM

(e) The MAUC of KNN (f) The F-measure of KNN

(g) The MAUC of GBDT (h) The F-measure of GBDT

FIGURE 4. The MinMax normalization value of MAUC and F-measure for the ACWGAN and other seven oversampling methods under 16 multi-class imbalanced datasets

The further results are summarized in Tables 3 and 4. Table 3 shows the average values of the F-measure, MG, and MAUC. Table 4 details the result of mean rank of 8 oversampling methods on 16 imbalanced datasets in terms of three evaluation metrics. For each dataset, the best and the second best performing approaches will be assigned to rank 1 and 2, and so on, the worst performing approach will obtain rank 8, and the best results are highlighted in bold. As is shown in Table 3, ACWGAN achieves the best average scores in 11 of 12 indicators. In terms of the evaluation of MAUC and MG, ACWGAN is significantly better than other seven popular methods. In the evaluation of MG, though ACWGAN appears to be less advantageous than some traditional methods, it is still better than those GAN-based methods under all classifiers. From Table 4, we observe that ACWGAN obtains rank 3 in the MG metric for SVM classifier, whereas, for

Table 3. The average MAUC, F-measure and MG of different oversampling methods over all datasets

Algorithm	Classifier	Average value		
		MAUC	F-measure	MG
SMOTE	DT	0.740571	0.552218	0.490310
BSMOTE		0.736402	0.553276	0.467473
KMSMOTE		0.736177	0.553895	0.463680
cGAN		0.736247	0.549333	0.463080
CWGAN-GP		0.741589	0.547491	0.491916
ACGAN		0.732872	0.552499	0.462097
BAGAN		0.738143	0.555479	0.482820
ACWGAN		**0.750612**	**0.571524**	**0.531642**
SMOTE	SVM	0.881348	0.601593	**0.580866**
BSMOTE		0.879159	0.594877	0.563108
KMSMOTE		0.872986	0.582028	0.494250
cGAN		0.874257	0.581339	0.484820
CWGAN-GP		0.884130	0.598161	0.575744
ACGAN		0.877761	0.570029	0.380590
BAGAN		0.876718	0.599689	0.431624
ACWGAN		**0.886915**	**0.620343**	0.566294
SMOTE	KNN	0.818480	0.567390	0.571343
BSMOTE		0.815787	0.566935	0.546336
KMSMOTE		0.815250	0.576550	0.525502
cGAN		0.824414	0.579988	0.507007
CWGAN-GP		0.833651	0.573941	0.551319
ACGAN		0.821060	0.578000	0.442632
BAGAN		0.823206	0.586021	0.464559
ACWGAN		**0.839281**	**0.598367**	**0.575341**
SMOTE	GBDT	0.874186	0.606160	0.543807
BSMOTE		0.874395	0.608705	0.533829
KMSMOTE		0.869439	0.599744	0.488847
cGAN		0.873669	0.604944	0.494535
CWGAN-GP		0.878731	0.607541	0.530916
ACGAN		0.873205	0.601631	0.450977
BAGAN		0.872442	0.601538	0.460706
ACWGAN		**0.882329**	**0.621400**	**0.546736**

TABLE 4. Results for mean ranking of oversampling methods across the datasets

Classifier	Metircs	SMOTE	BSMOTE	KSMOTE	cGAN	CWGAN-GP	ACGAN	BAGAN	ACWGAN
DT	MAUC	4.125	4.6875	5.125	5.3125	4	5.625	4.875	**2.1875**
	F-measure	4.6875	4.3125	4.3125	5.1875	5.4375	4.75	4.6875	**2.625**
	MG	3.5625	4.875	5.375	5.3125	4.0625	5.75	4.8125	**2.25**
SVM	MAUC	3.3125	4.4375	5.625	6.3125	3.6875	4.8125	5	**2.6875**
	F-measure	3.75	4.875	5.4375	5.4375	4.4375	6.375	4	**1.6875**
	MG	**2.0625**	3.375	5.1875	5.5	2.8125	7	6.6875	3
KNN	MAUC	5.4375	5.875	6.25	4.625	2.5	5.0625	4.5	**1.75**
	F-measure	5.9375	5.5	4.625	4	4.125	4.625	3.875	**2.5**
	MG	2.625	4.125	4.5	4.875	4	7.125	6.3125	**2.4375**
GBDT	MAUC	4.125	4.5	5.875	5.1875	3.875	5.25	5.1875	**2**
	F-measure	3.875	3.5625	5.6875	5.375	4.875	5.5	5	**2.125**
	MG	3.25	3.0625	5.0625	5.75	4	6.375	5.9375	**2.5**

other classifiers, ACWGAN obtains the best mean ranks. Those results show that the performance of ACWGAN is better than other oversampling methods, especially GAN-based methods.

4.2.3. *Results of statistical tests.* In order to verify whether all oversampling methods show the similar performance, we conduct Friedman test [35]. As is shown in Table 5, the hypothesis that all the methods are similar is rejected. Thus, we apply Holm's post hoc test [36] to determining whether ACWGAN is significantly different from each other method at the significance level of $\alpha = 0.05$. As the result presented in Table 6, in most cases, ACWGAN significantly outperforms other oversampling approaches. However, in the MG evaluation metric for classifier SVM, KNN, and GBDT, ACWGAN does not show significantly difference from SMOTE and Borderline-SMOTE. Besides, when using MAUC and MG as the evaluation metrics, ACWGAN has similar performance with CWGAN-GP under some classifiers.

TABLE 5. The Freidman test

Classifier	Metirc	p-value	Classifier	Metirc	p-value
DT	MAUC	0.002407	KNN	MAUC	4.9425e-08
	F-measure	0.060390		F-measure	0.004402
	MG	0.000772		MG	2.0079e-08
SVM	MAUC	0.000290	GBDT	MAUC	0.000314
	F-measure	0.000003		F-measure	0.000226
	MG	2.3295e-12		MG	7.8596e-07

4.3. **Practical application on ACWGAN.** To further verify the effectiveness of ACWGAN for practical application, we evaluate our method in two real imbalanced datasets. The experimental design and comparing methods are consistent with the previous subsection.

4.3.1. *Datasets of practical application.* Two real imbalanced datasets from different domains are used to evaluate the practical significance of our method. The first of datasets (PC4 dataset) is provided by the National Aeronautics and Space Administration (NASA), and describes the software defect status (defective, non-defective). The PC4 dataset contains 1458 samples, of which 1280 samples are non-defective samples and 178 are defective samples. The second dataset (ECG5000 dataset) consists of 5000 heartbeats extracted from a 20-hour long electrocardiogram of a patient with severe congestive heart failure. It

TABLE 6. Result of Holm's post hoc test when ACWGAN is used as the control method, and the p-values which are smaller than 0.05 are marked in bold.

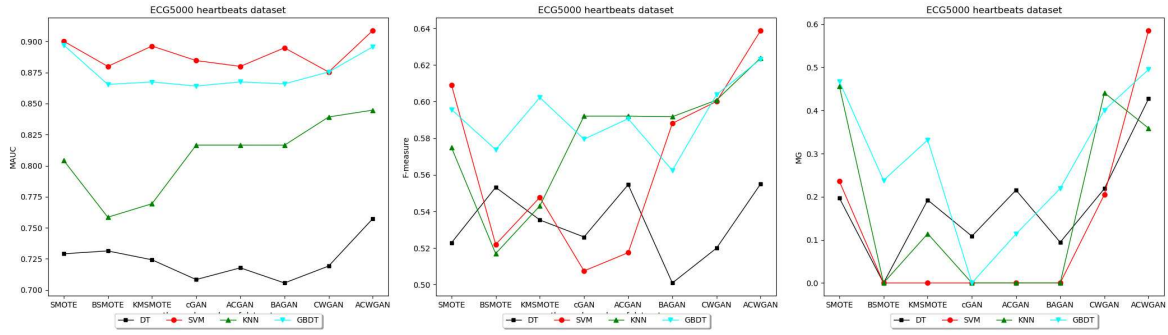
Algorithms	Classifier	p-value		
		MAUC	F-measure	MG
BAGAN	DT	**0.011482**	**0.011591**	**0.000826**
ACGAN		**0.000536**	**0.004394**	**0.001521**
CWGAN-GP		**0.004105**	**0.000917**	**0.016017**
cGAN		**0.000056**	**0.000904**	**0.002440**
KMSMOTE		**0.003684**	**0.009266**	**0.007431**
BSMOTE		**0.006196**	**0.015892**	**0.050191**
SMOTE		**0.005925**	**0.000697**	**0.027478**
BAGAN	SVM	**0.002416**	**0.000382**	**0.002675**
ACGAN		**0.004746**	**0.000127**	**0.000159**
CWGAN-GP		**0.040700**	**0.005081**	0.513952
cGAN		**0.000986**	**0.001660**	**0.049245**
KMSMOTE		**0.001529**	**0.002124**	**0.022074**
BSMOTE		**0.003172**	**0.003272**	0.883256
SMOTE		**0.029622**	**0.004204**	0.315005
BAGAN	KNN	**0.001504**	0.158085	**0.002116**
ACGAN		**0.000158**	**0.032328**	**0.000452**
CWGAN-GP		0.056725	**0.002820**	**0.008639**
cGAN		**0.001018**	**0.024190**	**0.012619**
KMSMOTE		**0.000016**	**0.006089**	**0.021309**
BSMOTE		**0.000012**	**0.001337**	0.109004
SMOTE		**0.000158**	**0.002487**	0.683488
BAGAN	GBDT	**0.002342**	**0.013592**	**0.007382**
ACGAN		**0.001070**	**0.000257**	**0.001750**
CWGAN-GP		0.110223	**0.000651**	**0.044206**
cGAN		**0.002219**	**0.000219**	**0.000309**
KMSMOTE		**0.001607**	**0.003024**	**0.013452**
BSMOTE		**0.014687**	0.052665	0.594322
SMOTE		**0.021683**	**0.040498**	0.879086

TABLE 7. Description of the real imbalanced datasets

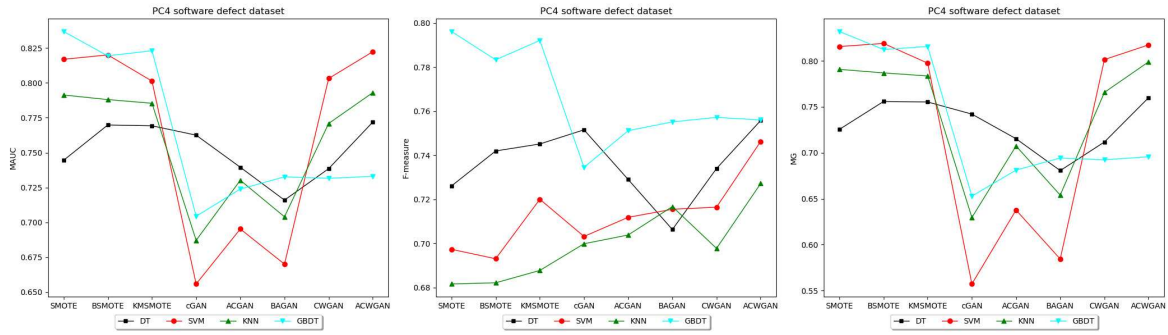
Dataset	I	F	C	Class distribution	IR
PC4	1458	38	2	1280/178	7.19
ECG5000	5000	140	5	2918/1767/96/194/24	121.58

is originally published in [27]. Interpolation was used to make the length of each heartbeat equal. Each heartbeat is classified into one of the five categories: normal (58.4% of the entire data), R-on-T Premature Ventricular Contraction (PVC) (35.3%), PVC (1.9%), supraventricular (3.9%) and unclassifiable (0.5%). The datasets are summarized in Table 7.

4.3.2. *Results of performance evaluation.* Figure 5 shows the performance evaluation results for different oversampling approaches using ECG5000 and PC4 datasets. As is clearly shown in the figure, when applied to the ECG5000 dataset, ACWGAN exhibits better



(a) Multi-class imbalanced ECG5000 heartbeat dataset



(b) Binary imbalanced PC4 software defect dataset

FIGURE 5. The performance evaluation of each oversampling approach in two real datasets

performance than other oversampling methods. We observe that the ACWGAN outperforms other oversampling approaches for all classifiers when F-measure is used as the indicator. For the MAUC metric, ACWGAN also outperforms other approaches except with classifier GBDT. Finally, when MG is used as the evaluation metric, ACWGAN achieves superior performance results except with the classifier KNN. For the imbalanced software defect dataset, in terms of the classifiers DT, SVM and KNN, ACWGAN performs similar to or better than other methods. However, for the classifier GBDT, the performance of ACWGAN is inferior to those SMOTE-like methods.

4.4. **Discussion.** In this subsection, we discuss the results of experiment. The experimental and statistical results indicate that our method outperforms the other oversampling methods especially GAN-based methods in most cases. The reason is that ACWGAN uses an auxiliary independent classifier to help the generator generate high-quality samples with correct class and improve the performance of classifier. Specifically, ACWGAN performs better than other oversampling methods, which can be attributed to the following reasons.

1) Compared with cGAN, BAGAN and ACGAN, ACWGAN uses the Wasserstein distance rather than JS divergence as the measure of distance, which can effectively prevent training of GAN from mode collapse and increase the diversity of generated samples.

2) Compared with CWGAN-GP, ACWGAN introduces an extra classifier to ensure the generator to synthesize high quality samples with correct class. However, the discriminator in CWGAN-GP may not be able to judge whether the samples match the corresponding labels in multi-class imbalanced dataset, causing the generator to generate samples with incorrect class.

3) Compared with SMOTE like method, ACWGAN synthesizes samples based on the actual distribution of whole data rather than based on k-nearest neighbor samples; thus, the synthesized data can better represent the overall distribution of the data.

Though ACWGAN outperforms those methods on most metrics, it is still slightly worse than some SMOTE-like methods in some cases. One possible reason is that for some minority class with scarce samples, ACWGAN could not learn enough features to generate high-quality samples, while SMOTE-like methods could synthesize relatively high-quality samples through k-nearest neighbors. Another disadvantage of ACWGAN is that the GAN needs more time and resource for training compared with those traditional methods.

5. **Conclusions.** In this paper, we proposed ACWGAN as a novel GAN model. The ACWGAN has an independent auxiliary classifier which can help generator better synthesize correctly minority samples under multi-class imbalanced scenario. In addition, we also proposed an oversampling method based on ACWGAN for multi-class imbalance dataset. To demonstrate the effectiveness of our proposed method, extensive experimental testing is performed on 16 multi-class imbalanced benchmark datasets and two real imbalanced datasets in comparison with several popular oversampling approaches. The experiment results show that ACWGAN is superior to other oversampling approaches.

As for future extension of this work, firstly, the data type can be extended from low-dimensional to high-dimensional (such as image) to improve the practicality of the method. In addition, we plan to improve the classification accuracy of the classifier in ACWGAN on imbalanced data to help the generator synthesize higher quality samples.

REFERENCES

[1] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, SMOTE: Synthetic minority over-sampling technique, *Journal of Artificial Intelligence Research*, vol.16, pp.321-357, 2002.

[2] Y. L. Zhang, J. Zhou, W. Zheng, J. Feng, L. Li, Z. Liu and Z. H. Zhou, Distributed deep forest and its application to automatic detection of cash-out fraud, *ACM Transactions on Intelligent Systems and Technology*, vol.10, no.5, pp.1-19, 2019.

[3] I. B. K. Manuaba, I. Sutedja and R. Bahana, The evaluation of supervised classifier models to develop a machine learning API for predicting cardiovascular disease risk, *ICIC Express Letters*, vol.14, no.3, pp.219-226, 2020.

[4] Z. Wang, Q. She and T. E. Ward, Generative adversarial networks in computer vision: A survey and taxonomy, *ACM Computing Surveys*, vol.54, no.2, pp.1-38, 2021.

[5] M. A. Kuenemann, C. M. Labbe, A. H. Cerdan and O. Sperandio, Imbalance in chemical space: How to facilitate the identication of protein-protein interaction inhibitors, *Scientic Reports*, vol.6, no.1, pp.1-17, 2016.

[6] W. Lin, C. Tsai, Y. Hu and J. Jhang, Clustering-based undersampling in class-imbalanced data, *Information Sciences*, vol.409, pp.17-26, 2017.

[7] Z. Zhou and X. Liu, Training cost-sensitive neural networks with methods addressing the class imbalance problem, *IEEE Transactions on Knowledge and Data Engineering*, vol.21, no.9, pp.1263-1284, 2009.

[8] H. He and E. A. Garcia, Learning from imbalanced data, *IEEE Transactions on Knowledge and Data Engineering*, vol.21, no.9, pp.1263-1284, 2009.

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

[10] H. He, Y. Bai, E. A. Garcia and S. Li, ADASYN: Adaptive synthetic sampling approach for imbalanced learning, *IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pp.1322-1328, 2008.

[11] G. Douzas, F. Bacao and F. Last, Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE, *Information Sciences*, vol.465, pp.1-20, 2018.

[12] S. Barua, M. M. Islam, X. Yao and K. Murase, MWMOTE – Majority weighted minority oversampling technique for imbalanced data set learning, *IEEE Transactions on Knowledge and Data Engineering*, vol.26, no.2, pp.405-425, 2014.

[13] T. Zhu, Y. Lin and Y. Liu, Synthetic minority oversampling technique for multiclass imbalance problems, *Pattern Recognition*, vol.72, pp.327-340, 2017.
[14] L. Abdi and S. Hashemi, To combat multi-class imbalanced problems by means of over-sampling techniques, *IEEE Transactions on Knowledge and Data Engineering*, vol.28, no.1, pp.238-251, 2016.
[15] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, Generative adversarial networks, *arXiv Preprint*, arXiv: 1406.2661, 2014.
[16] M. Mirza and S. Osindero, Conditional generative adversarial networks, *arXiv Preprint*, arXiv: 1411.1784, 2014.
[17] G. Douzas and F. Bacao, Effective data generation for imbalanced learning using conditional generative adversarial networks, *Expert Systems with Applications*, vol.91, pp.464-471, 2018.
[18] A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann and C. Sutton, VEEGAN: Reducing mode collapse in GANs using implicit variational learning, *arXiv Preprint*, arXiv: 1705.07761, 2017.
[19] M. Arjovsky, S. Chintala and L. Bottou, Wasserstein generative adversarial networks, *International Conference on Machine Learning*, pp.214-223, 2017.
[20] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin and A. Courville, Improved training of Wasserstein GANs, *arXiv Preprint*, arXiv: 1704.00028, 2017.
[21] M. Zheng, T. Li, R. Zhu, Y. Tang, M. Tang, L. Lin and Z. Ma, Conditional Wasserstein generative adversarial network-gradient penalty-based approach to alleviating imbalanced data classification, *Information Sciences*, vol.512, pp.1009-1023, 2020.
[22] A. Odena, C. Olah and J. Shlens, Conditional image synthesis with auxiliary classier GANs, *International Conference on Machine Learning*, pp.2642-2651, 2017.
[23] S. Suh, H. Lee, P. Lukowicz and Y. O. Lee, CEGAN: Classification enhancement generative adversarial networks for unraveling data imbalance problems, *Neural Networks*, vol.133, pp.69-86, 2021.
[24] A. Ali-Gombe and E. Elyan, MFC-GAN: Class-imbalanced dataset classification using multiple fake class generative adversarial network, *Neurocomputing*, vol.361, pp.212-221, 2019.
[25] X. Gao, F. Deng and X. Yue, Data augmentation in fault diagnosis based on the Wasserstein generative adversarial network with gradient penalty, *Nerocomputing*, vol.396, pp.487-494, 2020.
[26] R. Akbani, S. Kwek and N. Japkowicz, Applying support vector machines to imbalanced datasets, *European Conference on Machine Learning*, pp.39-50, 2004.
[27] A. Goldberger, L. Amaral, L. Glass, J. Hausdor, P. C. Ivanov, R. Mark, J. Mietus, G. Moody, C. Peng and H. Stanley, Components of a new research resource for complex physiologic signals, *PhysioBank, PhysioToolkit, and Physionet*, 2000.
[28] J. Sander, M. Ester, H. Kriegel and X. Xu, Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications, *Data Mining and Knowledge Discovery*, vol.2, no.2, pp.169-194, 1998.
[29] Y. Sun, M. S. Kamel and Y. Wang, Boosting for learning multiple classes with imbalanced class distribution, *The 6th International Conference on Data Mining (ICDM'06)*, pp.592-602, 2006.
[30] D. J. Hand and R. J. Till, A simple generalisation of the area under the ROC curve for multiple class classification problems, *Machine Learning*, vol.45, no.2, pp.171-186, 2001.
[31] J. R. Quinlan, Induction of decision trees, *Machine Learning*, vol.1, no.1, pp.81-106, 1986.
[32] J. H. Friedman, Greedy function approximation: A gradient boosting machine, *Annals of Statistics*, vol.29, no.5, pp.1189-1232, 2001.
[33] W. S. Noble, What is a support vector machine?, *Nature Biotechnology*, vol.24, pp.1565-1567, 2006.
[34] T. M. Cover and P. E. Hart, Nearest neighbour pattern classification, *IEEE Transactions in Information Theory*, vol.13, 1967.
[35] D. W. Zimmerman and B. D. Zumbo, Relative power of the Wilcoxon test, the Friedman test, and repeated-measures ANOVA on ranks, *Journal of Experimental Education*, 1993.
[36] T. Pohlert, The pairwise multiple comparison of mean ranks package (PMCMR), *R Package*, 2014.

Author Biography

Chen Liao is a postgraduate student in the Department of Information Science and Engineering of Guilin University of Technology. His main research interests include machine learning and imbalanced learning.

Minggang Dong obtained his Ph.D. degree in control science and engineering from Zhejiang University, China in 2012; he visited the University of Rhode Island for joint doctoral training from October 2016 to October 2017.

Prof. Dong is currently the vice dean of the Department of Information Science and Engineering at Guilin University of Technology. His main research interests include intelligent computing and application, and machine learning. He has published more than 50 papers in well-known journals at home and abroad. He is hosting some research projects funded from the National Natural Science Foundation of China.