This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3120642, IEEE Transactions on Multimedia

Annealing Genetic GAN for Imbalanced Web Data Learning

Jingyu Hao, Chengjia Wang, Guang Yang, *Senior Member, IEEE,* Zhifan Gao, Jinglin Zhang, and Heye Zhang, *Member, IEEE*

***Abstract*—Class imbalance is one of the most basic and important problems of web data. The key to overcoming the class imbalance problems is to increase the effective instances of the minority, that is, data augmentation. Generative Adversarial Networks (GANs), which have recently been successfully applied in the field of image generation, can be used for data augmentation because they can learn the data distribution given ample training data instances and generate more data. However, learning the distributions from the imbalanced data can make GANs easily get stuck in a local optimum. In this work, we propose a new training strategy called Annealing Genetic GAN (AGGAN), which incorporates simulated annealing genetic algorithm into the training process of GANs. And this can help GANs avoid the local optimum trapping problem, which easily occurs when the training set is imbalanced. Unlike existing GANs, which use a fixed adversarial learning objective alternately training a generator, we use multiple adversarial learning objectives to train a set of generators and use the Metropolis criterion in simulated annealing to decide whether the generator should update. More specifically, the Metropolis criterion accepts worse solutions with a certain probability, so it can make our AGGAN escape from the local optimum and find a better solution. Theory and mathematical analysis provide strong theoretical support for the proposed training strategy. And experiments on several datasets demonstrate that AGGAN achieves convincing ability to solve the class imbalanced problem and reduces the training problems inherent in existing GANs.**

***Index Terms*—Generative adversarial networks, Class imbalance problem, evolutionary computation, data augmentation.**

I. Introduction

CLASS imbalance is one of the most basic and important problems of web data. Although web data are rich and free resources, it is difficult to obtain the same quantity and quality of data for different categories. For example, in the task of identifying melanoma in lesion images[1], it is easy to obtain millions of images of moles from the website, while obtaining the same number of melanomas is almost impossible because the information about unfit patients is scarce compared to that of fit individuals. However, it is still important to learn from the minority classes. On one hand, minority classes often represent the objects of interest in tasks[2][3][4]. On the other hand, it will cause serious consequences if minority instances are incorrectly classified by the classification algorithms[5]. For example, melanoma, specifically, is responsible for 75% of skin cancer deaths[6], and incorrect detection may delay treatment and cause serious consequences.

Data augmentation for minority classes is an effective way to solve the imbalance problem. Because in essence, the reason for the imbalance problem is that the minority classes do not have enough representative instances. And these limited instances are not enough for algorithms to learn their distribution precisely. Therefore, it would be very helpful to generate more representative but realistic data for minority classes. A common method is to directly increase the number of instances by over-sampling (adding repetitive data) for the minority classes. But repetition may lead to over-fitting because the minority instances will be overemphasized. Interpolating is another way to generate more minority instances. But web data are usually located in a high dimensional space and the data manifold is very complex, so interpolation is nontrivial and may generate low-quality samples [7].

Generative Adversarial Networks (GANs) are a powerful class of generative models that can be used for data augmentation. GANs have recently been successfully applied to image generation [8][9], image-to-image synthesis [10], image super-resolution [11] and other applications.

However, GANs can easily get stuck in a local optimum when they try to learn the distributions from the imbalanced data set[12][13][14]. As we all know, GANs learn a mapping between a latent space and a complex target distribution of interest through adversarial training. And we can think of this training process as a process of finding the optimal solution (target distribution). When the generator G learns the target distribution and can generate enough real samples to fool the discriminator D, the training process will stop. However, most real-world data distributions are multimodal. When the generator G has only learned a certain part of this multimodal distribution, but the generated samples can also deceive the discriminator D, the training process will also stop. So the GANs get stuck in a local optimum, which is also known as mode collapse [15]. When the training data is imbalanced, the above situation will occur more easily[16]. This is because comparing with the majority classes, learning distribution from minority classes is much harder. So generator G will prefer to learn more information about majority classes which is easier to deceive discriminator D. As a result, the model stops

Corresponding author: Jinglin Zhang

Jingyu Hao, Zhifan Gao and Heye Zhang are with School of Biomedical Engineering, Sun Yat-sen University, Shenzhen 518107, China (email: haojy5@mail2.sysu.edu.cn; gaozhifan@mail.sysu.edu.cn; zhanghheye@mail.sysu.edu.cn)

Chengjia Wang is with Centre for Cardiovascular Science, University of Edinburgh, Edinburgh EH16 4TJ, UK (email:chengjia.wang@ed.ac.uk)

Guang Yang is with Faculty of Medicine, National Heart Lung Institute, Imperial College London, London SW7 2AZ, UK (email:g.yang@imperial.ac.uk)

Jinglin Zhang is with the School of Artificial Intelligence, Hebei University of Technology, Tianjin 300401, China (e-mail: jinglin.zhang37@gmail.com)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3120642, IEEE Transactions on Multimedia

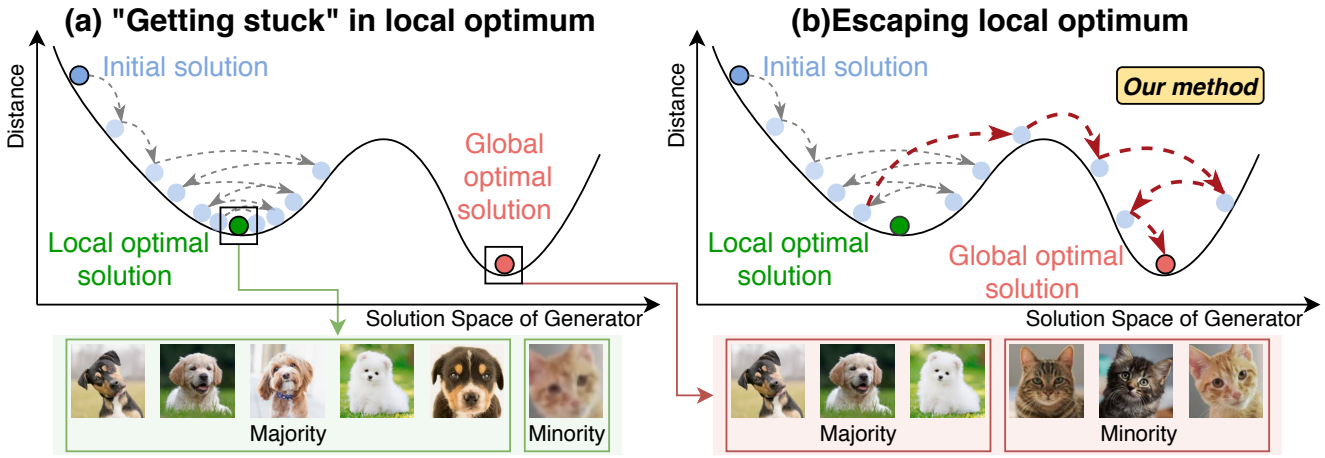

Fig. 1. (a) Schematic illustration of local optimum trapping for GANs. In this situation, GANs fluctuate around the local optimum until the end of training and the learned distribution is still very different from the minority class. Consequently, GANs prefer to generate more majority instances. And the minority instances GANs generated may have low quality. (b) Our method integrates the simulated annealing genetic algorithm into the training of GANs. When GANs get stuck into a local optimum, simulated annealing algorithm can help them update to a worse solution with a certain probability, which can make GANs escape from the local optimum. So GANs can generate more minority instances that are closer to real instances.

training and gets stuck in a local optimum.

Hence, we propose a training strategy that incorporates simulated annealing genetic algorithm into the training process of GANs to avoid the local optimum trapping problem. And we call this model Annealing Genetic GAN(AGGAN). We regard the adversarial learning of GANs as the evolutionary process of a species (acted by the generator) in the environment (acted by the discriminator). The iteration of the generator is regarded as the process of individual genetic evolution, which is mainly divided into two parts: generating the best offspring and updating the generator. In each iteration, G gives birth to different offspring G_c with different adversarial learning objectives. Then we calculate the individual fitness of different offspring based on environment D. Similar to the "survival of the fittest" in natural selection, poor individuals (generators) are continuously eliminated with training, and only the individuals most adapted to the environment G_{cBest} (that is, the solution closest to the global optimum) will be retained. Updating the generator uses the idea of simulated annealing. If the individual fitness of G_{cBest} is higher than G, AGGAN will update it to G like other GANs. If the individual fitness of G_{cBest} is lower than G, AGGAN will use metropolis criterion, a random acceptance criterion, to update G_{cBest} to G. This training strategy of updating G with a certain probability in a worse direction enables AGGAN to escape the local optimum as shown in Figure 1(b). In summary, we make the following contributions in this paper:

- In order to make GANs avoid the local optimum, we propose a strategy to incorporate simulated annealing genetic algorithm into the training process of GANs.
- Through analyzing the training process AGGAN, we prove that our proposed AGGAN can reproduce the distributions closest to the target distribution. To the best of our knowledge, it's the first work of a complete theoretical proof of the GANs combined with evolutionary algorithms.
- Experiments evaluated on several imbalanced datasets show that convincing results can be achieved by our proposed AGGAN.

II. Related Work

A. Class Imbalance Problem

Class imbalance is a common problem in web data classification tasks. The limited instances of the minority classes make it difficult for the classifier to learn the minority classes distribution correctly. Therefore, the key to solve class imbalance is to learn an accurate distribution of the minority classes. Data augmentation for minority classes is an effective method to tackle the class imbalance problem. Because it can improve the accuracy of learning minority classes distribution[2]. Random oversampling, Synthetic Minority Over-Sampling Technique (SMOTE) [17] and Borderline SMOTE[18] are commonly used data augmentation methods in class imbalance problems. Random oversampling was the easiest to implement because it adds instances by simply replicating or slightly modifying them. However, the instances obtained using this strategy might not be generalized, which could lead to over-fitting. Interpolation is another way of oversampling, e.g., using Synthetic Minority Over-Sampling Technique (SMOTE) [17]. The SMOTE could alleviate the phenomenon of over-fitting, but it assumed that all kinds of samples were surrounded by the samples of minority class and had strong blindness [18]. Border-line SMOTE was then developed based on the SMOTE, in which only the samples of minority class near the borderline were over-sampled. It could mitigate marginality and blindness problems of the SMOTE. However, when dealing with data in high dimensional space, the quality of the synthesized new data points could still be compromised due to noise and poor distance measurement in the high dimensional space[19].

B. Generative Adversarial Networks

Existing GANs have achieved great success in image generation [8][9], image-to-image synthesis [10], image super-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3120642, IEEE Transactions on Multimedia

resolution [11] and other applications because they can capture the data distribution through adversarial learning. In addition, the success of GANs has led some researchers to investigate the utility of using GANs to solve class imbalance problems[16]. But the number of minority instances is scarce. When the training stop, GANs cannot learn the complete distribution. They may only learn certain distributions and can only generate instances of a single mode, that is, getting stuck into a local optimum, also known as mode collapse. Some studies have been done to help GANs learn more accurate imbalance distribution by improving adversarial learning objectives (e.g. LSGAN [20], WGAN [21]). However, there still exist limitations by using fixed adversarial training objectives. For example, since WGAN measured the Wasserstein distance between the generated distribution and the data distribution, it had to enforce the Lipschitz constraint on the discriminator, which might result in difficulties for optimization [22]. More recently, Wang et al proposed Evolutionary-GAN (EGAN)[23], which uses different adversarial objectives to evolve a population of generators to play the adversarial game with the discriminator. However, the problem of getting stuck in local optimum still has not been addressed yet.

C. Evolutionary Algorithms in Neural Networks

Evolutionary algorithms are a family of algorithms inspired by the idea of biological evolution, which are used for complex optimization problems. They have achieved great performance in various tasks including modeling, optimization, and design [24]. Genetic Algorithm (GA) belongs to evolutionary algorithms and is commonly used to generate high-quality solutions to optimization and search problems[25]. Besides GA, Simulated Annealing (SA) is also the main method for solving search and optimization problems in high dimensional spaces [26]. Recently, they have been introduced to optimize deep learning hyper-parameters and design deep network architectures. Galar et al. [27] proposed to use Cartesian genetic programming to design convolutional neural network (CNN) architectures. Rere et al. [28] used SA to improve the performance of CNN, as an alternative approach for optimal deep learning using modern optimization techniques. Tao et al. [29] introduced a novel variational annealing strategy to stabilize and facilitate general training of GANs. However, there are also significant differences between GA and SA. GA is a population based search algorithm, which uses the same strategy to select the solution. While SA works on one solution per iteration, and its selection strategy is based on temperature and annealing coefficient parameters. These differences lead to the different advantages and disadvantages of GA and SA. GA can rapidly find a solution, but it is not guaranteed to find the exact minimum, i.e. its local search ability is weak. SA has a strong local search ability, but when it gets trapped in a local minimum, it will take longer to escape[30][31][32]. The simulated annealing genetic algorithm tries to combine the advantages of GA and SA. And we incorporate it into the training process of GANs to improve the quality of solutions and reduce execution time.

Algorithm 1 Annealing Genetic GAN (AGGAN)

Input: The imbalanced dataset X, label Y.
Parameter: the number of mutations n_m, different training strategies $\mathcal{M}_c(c = 1, ..n_m)$, the log-likelihood of the correct source $\mathcal{L}_S$, the log-likelihood of the correct class $\mathcal{L}_{Cls}$, initial temperature T, annealing coefficient α.
Output: The classification results.

$\theta_D, \theta_G \leftarrow$ initialise network parameters
repeat
  Calculate individual fitness of $G : \mathcal{F}_G$
  for $c = 1, ..n_m$ **do**
    Individual G generate offspring G_c with $\mathcal{M}_c$:
    $\mathcal{L}_{Gc} = \mathcal{M}_c + \mathcal{L}_{Cls}$
    $\theta_{Gc} \leftarrow \nabla_{\theta Gc}\mathcal{L}_{Gc}$
    Calculate individual fitness of $G_c : \mathcal{F}_c$
  end for
  Sort $\{\mathcal{F}_c\}$, and expressed the largest one as $\mathcal{F}_{cbest}$
  if $\mathcal{F}_{cbest} > \mathcal{F}_G$ **then**
    Update G_{cbest} to G_{new}.
  else
    $\Delta = \mathcal{F}_{cbest} - \mathcal{F}_G$
    $P = e^{-\Delta/T}$
    Update G_{cbest} to G_{new} with probability P
  end if
  $T = \alpha * T$
  $\mathcal{L}_{\mathcal{D}} = \mathcal{L}_S + \mathcal{L}_{Cls}$
  $\theta_D \leftarrow \nabla_{\theta_D}\mathcal{L}_{\mathcal{D}}$
until Convergence

III. Methods

In this section, we first review the original GAN formulation. Then, we introduce the proposed Annealing Genetic GAN algorithm. By illustrating the simulated annealing genetic algorithm of AGGAN, we further discuss the advantage of the proposed framework.

A. Generative Adversarial Networks

GAN was first proposed in [33]. This framework simultaneously trains two networks: a generative network G that captures the data distribution, and a discriminative network D that estimates the probability that a sample came from the training data rather than G. G takes noisy sample $z \sim p_z$ (sampled from a latent space) as the input and outputs the $G(z)$ whose distribution p_g is supposed to be as close as possible to the target data distribution p_{data}. At the same time, D needs to distinguish between real data samples $x \sim p_{data}(x)$ and generated samples $G(z) \sim p_g(G(z))$. D is trained to maximize the probability of assigning the correct label to both x and $G(z)$. Meanwhile, G is trained to minimize $\log(1 - D(G(z))$:

$$\min_G \max_D \mathbb{E}_{x\sim p_{data}}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))] \tag{1}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3120642, IEEE Transactions on Multimedia

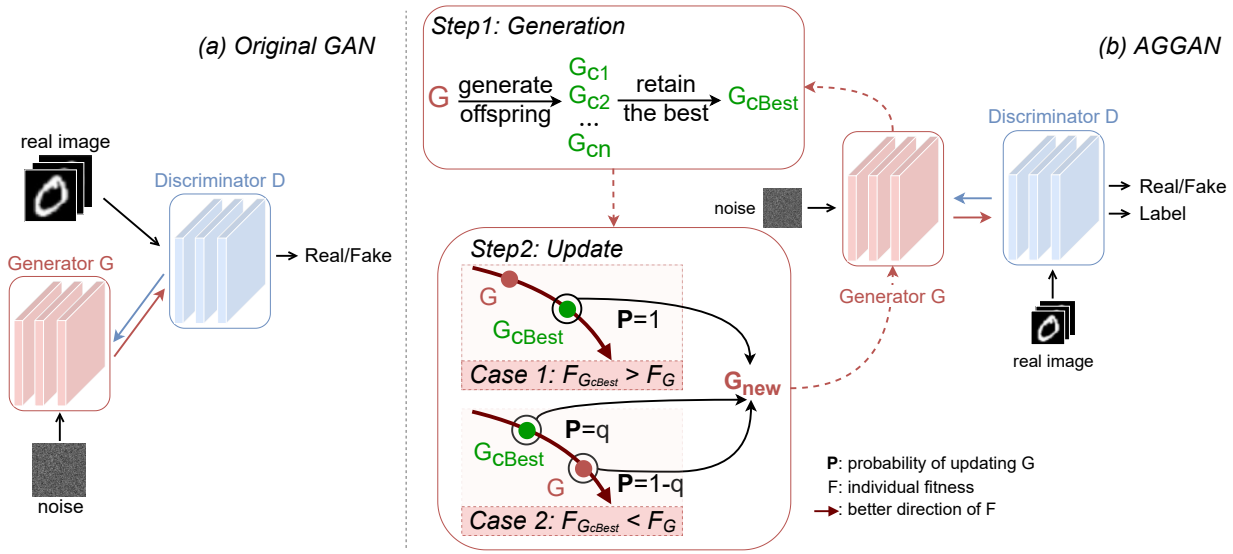

Fig. 2. (a) The original GAN framework. A generator G and a discriminator D play a two-player adversarial game. (b) The proposed AGGAN. The training of the generator is divided into two steps. The first step is to generate the best-fit offspring. The parent generator G generates $G_{c1}, G_{c2}...G_{cn}$ through different adversarial learning objectives. Assessing the individual fitness of all offspring, the offspring G_{cbest} with the highest individual fitness is retained. The second step is to update the generator. We use the simulated annealing strategy to decide whether to update G_{cbest} to the next generator G_{new}. If the individual fitness of G_{cbest} is higher than the parent G, then we update G_{cbest} to G_{new} with a probability of 1. If the individual fitness of G_{cbest} is lower than the parent G, G_{cbest} is updated to G_{new} with a probability of $P = q$, where q is based on the temperature, annealing coefficient and the difference between the two individual fitness. This probability q will continue to decrease with the training so that GANs can converge to the global optimum. After the training of the generator is finished, the samples from G_{new} are input into the discriminator D for training together with the real samples.

B. Annealing Genetic Generative Adversarial Networks

In contrast to conventional GANs, which alternately update a generator and a discriminator, AGGAN uses different adversarial learning objectives to evolves a population of generators to improve their performance. At the same time, the training process of AGGAN incorporates the idea of simulated annealing which can make the model converge to the distribution closest to the target distribution. Specifically, we look at the training process of AGGAN from an evolutionary perspective: evolving individuals (generator G) in the environment (discriminator D). The evolution of the individual consists of two parts: generation and update.

1) Generation: Generating the Best-fit Offspring: In each iteration, we use different adversarial learning objectives to produce offspring G_c. Specifically, these different adversarial learning objectives try to narrow the distances between the real data distribution and the generated distribution in different ways. And each offspring G_c represents a solution in the parameter space of the generator network. Then the environment (D) will evaluate and select offspring G_c. The quality and diversity of the samples generated by the offspring G_c will be the basis for judgment. Specifically, we input the samples into the discriminator, and the average value of the discriminator's output can be used to evaluate the quality of the samples.

$$F_{quality} = \mathbb{E}_z[D(G(z))] \tag{2}$$

This is because the higher the quality score of the samples, the more likely they are to fool the discriminator. And we use the minus log-gradient-norm of optimizing discriminator to evaluate the diversity of the samples.

$$F_{diversity} = -log||\nabla_{\theta_D}\mathcal{L}_{\mathcal{D}}|| \tag{3}$$

where θ_D represents the parameters of discriminator D and $\mathcal{L}_{\mathcal{D}}$ is the loss function of D. Because the discriminator will update with small gradients when the input samples are dispersed enough[22]. We combine quality score and diversity score as individual fitness:

$$F_{fitness} = F_{quality} + \lambda F_{diversity} \tag{4}$$

where λ is used to balance $F_{quality}$ and $F_{diversity}$. Only the offspring with the highest individual fitness participate in the follow-up process. The process above reflects the idea of "survival of the fittest" in the Genetic Algorithm. And using different adversarial objectives can make GANs approach the target distribution from different angles at the same time, which can help the generator to achieve better performance after training.

2) Update: Updating the Generator: In this part, we mainly adopt the idea of simulated annealing. According to the individual fitness of G_{cBest}, we divide it into two cases. (1) the individual fitness of G_{cBest} is higher than that of G; (2) the individual fitness of G_{cBest} is lower than that of G. In case 1, G_{cBest} will be directly updated to G like other GANs. But in case 2, G_{cBest} will be updated to G with a probability P. The probability P is not fixed and is related to the current temperature T_c and the individual fitness of G_{cBest} and G. The temperature T_c will gradually decrease with training, and the probability P will also decrease. In general, updating to a worse solution with a probability P helps AGGAN escape from the local optimum, and the decreasing probability enables AGGAN to asymptotically converge to the global optimum.

Finally, after the above process, the environment (i.e., the discriminator) D is updated. Note that the D of AGGAN has two outputs, corresponding to Sigmod and Softmax activation

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3120642, IEEE Transactions on Multimedia

TABLE I
DEFINITION OF SYMBOLS

Symbol	Definition
g	Individual
g_c	Offspring of g
g_{cbest}	The best offspring of g
g_b	The best g so far
G	Solution space of g
G_n	Solution space for the n_{th} iteration
$\{G_n \mid n \in N\}_g$	The G_n sequence with initial state g
$succ(g)$	The set of g's possible successors
G_{opt}	Optimal solution

functions. So D can not only judge whether the samples are real, but also judge the class of samples. Algorithm 1 summarises the whole training procedure of the AGGAN.

IV. THEORETICAL ANALYSIS

In this section, we perform a theoretical analysis of AGGAN. We proved that incorporating the simulated annealing genetic algorithm into the training process of GANs can make AGGAN learn the distribution closest to the real data. In other words, if we regard the training process of GANs as a process of constantly searching for the globally optimal solution (i.e. the target distribution), then AGGAN can converge to the global optimal solution with a probability of 1.

For our discussion, we consider the training of GANs as a combinatorial optimization problem. We consider the problem as a pair (G, f), where G is a finite set, the solution space of generator g, f is an objective function. The aim is to find a global optimum G_{opt} to minimize f, that is, to learn the distribution closest to the actual distribution. Notice that the finiteness of G implies that f has at least one minimum over G.

Below we give the definition of the two steps of AGGAN update generator from a mathematical perspective. And then we prove that the simulated annealing genetic algorithm will make AGGAN converge to the optimal solution as the training progresses.

A. Definition

For readability, we first define the symbols we used in Table I. Then we give the mathematical definition of "Generation: generating the best-fit offspring" and "Update: updating the generator" in AGGAN. In addition, in order to obtain monotonicity without changing the training mechanism, we add a second element g_b, which represents the best g so far, to each individual (generator). And then we write this combination as $[g, g_b]$.

Definition 1. *Generation Function F_{gen}*

This part corresponds to the "Generation: generating the best-fit offspring". First, we use a choice function to find the parent g from $[g, g_b]$.

$$F_{cho}(g, g_b) = g \tag{5}$$

Note g_b is the best g so far. And F_{choice} can make g_b not change the training mechanism of AGGAN. Then the parent g generates offspring g_c under the production function,

$$F_{pro}(g) = g_{c1}, g_{c2}, ..., g_{cn} \tag{6}$$

where n is the number of offspring and each g_c comes from a different adversarial learning objective. After that, we use the individual fitness function to calculate the individual fitness of g_c, and retain the offspring g_{cbest} with the highest individual fitness.

$$F_{fit}(g_{c1}, g_{c2}, ..., g_{cn}) = g_{cbest} \tag{7}$$

All above is the whole process of "Generation: generating the best-fit offspring". Since the choice function, production function, and individual fitness function are not mentioned in the analysis below, we collectively represent the above process with F_{gen}:

$$\begin{aligned} &F_{gen}(g, g_b) \\ =&F_{fit}(F_{pro}(F_{cho}(g, g_b)))) \\ =&g_{cbest} \end{aligned} \tag{8}$$

As the parameters of the choice function, production function, and individual fitness function don't depend on the number of iterations, F_{gen} follows the same distribution under different times of iterations.

Definition 2. *Update Function F_{upd}*

The process of "Update: updating the generator" uses the idea of simulated annealing. We define it as follows:

$$F_{upd}(g, g_b, g_{cbest}) = \begin{cases} [g, g_b'], & \text{if } e^{-\frac{\Delta}{T}} < \gamma \\ [g_{cbest}, g_b'], & \text{otherwise} \end{cases} \tag{9}$$

where

$$g_b' = \begin{cases} g_b, & \text{if } f(g_{cbest}) > f(g_b) \\ g_{cbest}, & \text{if } f(g_b) > f(g_{cbest}) \end{cases}$$

and Δ is the difference between the individual fitness of g and g_{cbest}. γ is a random number between 0 and 1. T is the temperature parameter for AGGAN.

We combine F_{upd} and F_{gen} to get F to represent the iterative process of the generator:

$$F(g, g_b) = F_{upd}(g, g_b, F_{gen}(g, g_b)) \tag{10}$$

And we use F_n to stand for the F in the n_{th} iteration.

B. Proof of Convergence

In this section, we will prove that $\{G_n \mid n \in N\}_g$ satisfies the following properties, which can make AGGAN converge to the global optimum with probability 1 according to Corollary 1.

Property 1. *(Monotonicity). G_n is monotonous. The minimum value of G_n on f decreases as n becomes larger.*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3120642, IEEE Transactions on Multimedia

Since F_{upd} always keeps the best f value, for any g

$$\begin{aligned} &min\{f(s)|s \in g\} \\ \geq &min\{f(s)|s \in [g, g_b, g_{cbest}]\} \\ \geq &min\{f(s)|s \in F_{upd}(g, g_b, g_{cbest})\} \end{aligned} \tag{11}$$

holds.

Hence by Eq.10, Eq.11 and

$$G_{n+1} = F_n(G_n, g_b) \tag{12}$$

we have that

$$\begin{aligned} &min\{f(s)|s \in G_{n+1}\} \\ =&min\{f(s)|s \in F_n(G_n, g_b)\} \\ =&min\{f(s)|s \in F_{upd}(G_n, g_b, F_{gen}(G_n, g_b))\} \end{aligned} \tag{13}$$

$$\leq min\{f(s)|s \in G_n\} \tag{14}$$

so $\{G_n|n \in N\}_g$ is monotone.

Property 2. *(Homogeneity) If $\{G_n|n \in N\}_g$ is a Markov chain, and the F_n have the same distribution then the chain is homogeneous.*

Due to the memoryless property of GANs, for any n, the conditional probability distribution of G_{n+1} (conditional on both past and present states) depends only upon the G_n, not on the sequence of G that preceded it. Then we get:

$$\begin{aligned} &P(G_{n+1} = g_{n+1}|(G_n = g_n) \wedge ... \wedge (G_0 = g)) \\ =&P(G_{n+1} = g_{n+1}|G_n = g_n) \end{aligned} \tag{15}$$

so $\{G_n|n \in N\}_g$ has Markov property. Since the F_n have the same distribution then

$$P(G_m = y|G_{m-1} = z) = P(G_n = y|G_{n-1} = z) \tag{16}$$

holds for any $y, z \in G$ and $m, n \in N$. So $\{G_n|n \in N\}_g$ is homogeneous.

Then we use the following corollary[34]:

Corollary 1. *Let $g \in G$ and the following conditions be satisfied:*

(a) $\{G_n|n \in N\}_g$ is monotone

(b) $\{G_n|n \in N\}_g$ is homogeneous

(c) for every $h \in succ(g)$ there exists at least one accessible optimum.

Then $\{G_n|n \in N\}_g$ surely reaches an optimum.

We can prove that AGGAN converges to the global optimum which is closest to the target distribution with probability 1.

V. Experimental Studies and Discussion

In this section, we evaluate our method in six image datasets (MNIST [35], Fashion-MNIST [36], SVHN [37], CIFAR-10 [38], CelebA[39], and LSUN[40]). And we create two imbalanced environments: two-class imbalanced classification and multi-class imbalanced classification. Note that we define the Imbalance Rate (IR) as the number of training samples in the largest category divided by the minimum number of samples in both the two-class and multi-class imbalanced classification. In binary classification, we randomly select two classes from each data set because all the specified data sets are multi-category data sets, and we randomly select a disparate number of samples of these two classes. For each image dataset, we create three two-class imbalanced datasets with IR is 10, 50, and 100. See Table II for details. In multi-classification, we use the same way as the paper of Mullick [13] to create the multi-class imbalanced datasets. See Table III for details. Unlike the training set constructed above, we use fully balanced data as the testing set.

TABLE II
Details of two-class imbalance training sets

Dataset	Positive	Negative	Shape
MNIST	Digit 5	Digit 6	$28 \times 28 \times 1$
Fashion-MNIST	Sandal	Sneaker	$28 \times 28 \times 1$
CIFAR-10	Airplane	Automobile	$32 \times 32 \times 3$
SVHN	Digit 8	Digit 9	$32 \times 32 \times 3$
CelebA	Eyeglasses	No-eyeglasses	$64 \times 64 \times 3$
LSUN	Church	Classroom	$128 \times 128 \times 3$

TABLE III
Details of multi-class imbalance training sets

Dataset	IR	Training Set
MNIST	100	4000,2000,1000,750,500,350,200,100,60,40
Fashion-MNIST	100	4000,2000,1000,750,500,350,200,100,60,40
CIFAR-10	56.25	4500,2000,1000,800,600,500,400,250,150,80
SVHN	56.25	4500,2000,1000,800,600,500,400,250,150,80
CelebA	100	15000,1500,750,300,150
LSUN	100	15000,1500,750,300,150

A. Implementation Details

In the binary experiment, we compared AGGAN to the baseline classifier network (CN), Os+CN (oversampled the training set randomly), ACGAN [41] and Evolutionary-GAN[23] (discriminator with classifier) to demonstrate the effectiveness of our method. In the multi-classification, in addition to the methods mentioned above, we also compared AGGAN with another two classical GANs frameworks: LSGAN[20] and WGAN[21], and three state-of-the-art algorithms which are used to tackle class imbalance problems: Class-Balanced[42], DOS[7] and GAMO[13]. For all GANs-based methods, we use the same network structures in each dataset for a fair comparison. Besides that, the discriminator of LSGAN, EGAN, WGAN, and AGGAN all need to judge the class of data like ACGAN. More specifically, our proposed AGGAN uses the same adversarial learning objectives (the minimax loss, the unsaturated loss, and the least-squares loss) with EGAN to generate a population of offspring generators. And we use the same evaluation function to measure the individual fitness of generators in AGGAN and EGAN. The experiments of ACGAN and EGAN are actually ablation experiments for the "generation" and "update" parts in AGGAN.

We use three widely used metrics (Average Class Specific Accuracy (ACSA)[43], F-measure[44] and Geometric Mean(GM)[45]) to validate the imbalance classification performance. In order to eliminate the randomization bias, all

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3120642, IEEE Transactions on Multimedia

TABLE IV
ACCURACY(%) ON IMBALANCED BINARY CLASSIFICATION WITH VARIOUS IMBALANCE RATIO.

Dataset	MNIST			Fashion-MNIST			SVHN		
IR	10	50	100	10	50	100	10	50	100
CN	97.93	94.80	90.27	93.53	86.73	80.00	74.65	58.15	52.00
Os+CN	98.60	96.33	95.53	94.67	92.07	88.13	87.60	66.50	58.90
ACGAN	99.53	98.13	96.60	**97.80**	96.20	94.13	90.50	74.65	70.55
E-GAN	99.53	98.33	97.00	97.67	96.13	94.40	90.85	74.90	75.50
AGGAN	**99.60**	**98.47**	**97.53**	**97.80**	**96.53**	**95.07**	**90.95**	**81.60**	**77.70**
Dataset	CIFAR			CELEBA			LSUN		
IR	10	50	100	10	50	100	10	50	100
CN	72.00	58.85	57.65	71.10	64.40	60.80	81.20	74.60	52.00
Os+CN	90.60	82.75	80.55	86.70	78.85	77.65	92.05	89.95	64.00
ACGAN	92.00	84.70	83.65	91.65	81.40	78.60	95.05	91.10	89.50
E-GAN	92.10	86.55	85.75	91.65	81.75	80.80	95.20	92.40	91.00
AGGAN	**94.00**	**89.25**	**86.25**	**92.05**	**84.20**	**81.40**	**95.35**	**93.10**	**91.70**

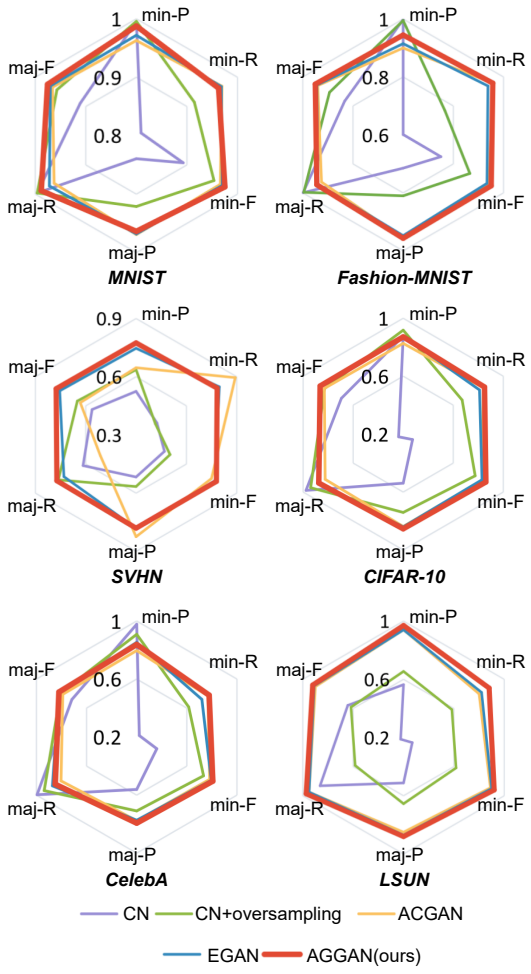

Fig. 3. Precision, recall, F1-score of majority and minority classes in two-class imbalanced datasets with IR=100. "P", "R" and "F" in the figure correspond to the first letters of the three indicators, and "min" and "maj" correspond to the minority and majority class respectively.

Fig. 4. The smallest minority class images generated by GANs methods. The smallest class in MNIST is the Number 9, and Fashion-MNIST is the ankle boot. There were only 40 samples in the training set for these two classes.

the experiments are repeated 5 times with different random seeds for creating imbalanced datasets. And for all experiments based on GANs related methods, in order to make full use of the information from the scarce samples of minority classes, all the samples belonging to the minority classes are oversampled. Note that the over-sampled samples are the same as the original samples, and we do not use any data augmentation techniques to improve the diversity of minority classes.

B. Classification Performance

Table IV reports results obtained on two-class classification under different IR respectively. The results show that the original imbalanced data will seriously affect the performance of the classifier. And consistent with the shortcomings of random oversampling we mentioned before, the results of random oversampling data show that repetition can not improve the performance of the classifier very well. Three experimental

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3120642, IEEE Transactions on Multimedia

TABLE V
ACCURACY (%) OF IMBALANCED MULTI-CLASSIFICATION.

Dataset	MNIST			Fashion-MNIST			SVHN		
Method	ACSA	F_1	GM	ACSA	F_1	GM	ACSA	F_1	GM
CN	89.00	89.04	88.36	79.70	78.41	75.03	73.80	73.57	69.96
ACGAN	90.70	90.64	90.49	79.90	79.32	77.61	75.40	74.16	71.71
EGAN	91.30	91.22	91.02	81.90	81.60	81.31	76.80	76.41	75.07
LSGAN	91.19	91.09	90.86	80.61	80.32	78.67	75.30	75.02	73.97
WGAN	91.70	91.67	91.51	81.00	80.55	78.94	75.90	75.89	74.10
CB-loss	92.30	92.34	92.12	81.90	81.41	79.93	75.60	76.35	72.63
DOS	90.60	90.65	90.05	82.74	82.80	79.00	71.20	70.01	69.00
GAMO	91.20	91.13	91.07	82.90	82.16	80.16	76.00	75.43	75.20
AGGAN	**92.41**	**92.34**	**92.26**	**83.20**	**83.23**	**82.60**	**77.10**	**76.53**	**75.58**
Dataset	CIFAR			CELEBA			LSUN		
Method	ACSA	F_1	GM	ACSA	F_1	GM	ACSA	F_1	GM
CN	45.30	41.35	30.16	59.00	53.75	43.78	49.80	38.22	29.96
ACGAN	47.50	45.59	42.27	66.20	64.13	63.96	56.20	56.43	54.18
EGAN	48.10	45.59	41.74	67.20	65.71	63.96	59.60	58.64	57.48
LSGAN	47.50	45.23	39.82	66.80	65.23	63.50	56.40	54.18	50.78
WGAN	48.60	46.61	40.23	67.60	66.44	65.10	57.80	57.13	56.16
CB-loss	49.30	47.07	43.01	60.00	56.99	53.58	55.40	54.27	52.24
DOS	46.02	45.04	38.30	61.10	60.12	50.13	54.02	52.80	50.54
GAMO	49.00	47.18	45.07	66.00	65.37	58.00	57.20	55.18	54.04
AGGAN	**52.40**	**51.29**	**46.74**	**68.80**	**67.06**	**65.01**	**63.60**	**61.60**	**59.15**

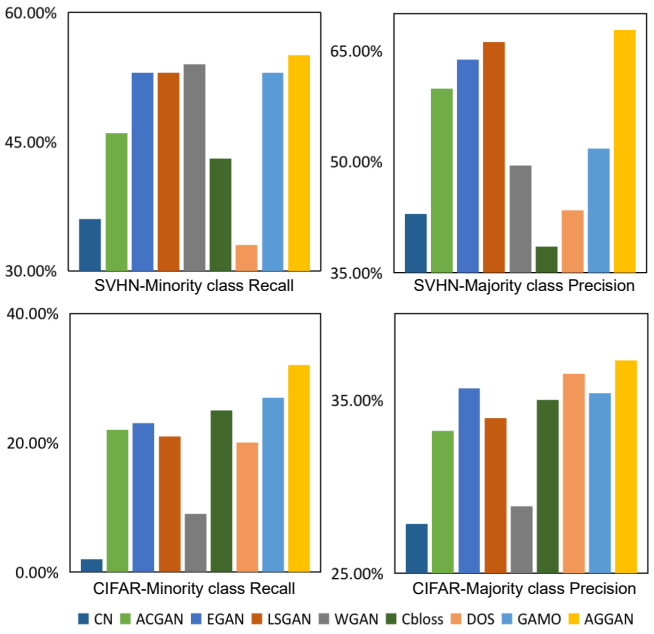

Fig. 5. Performance of the smallest class recall and largest class precision. The first row corresponds to the number 9 (the smallest class) and the number 0 (the largest class) in the SVHN respectively. The second row corresponds to the truck (the smallest class) and the airplane (the largest class) in the CIFAR respectively. Larger value indicates better performance.

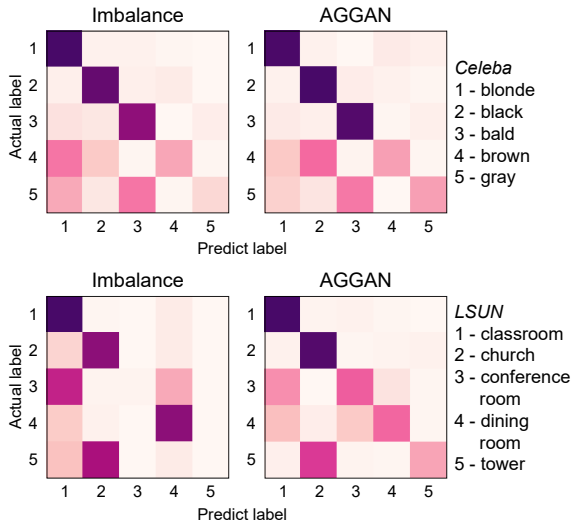

Fig. 6. Confusion matrix for multi-class imbalanced classification results of CelebA and LSUN. The left column shows the classification results of CN that has been trained on the imbalanced data. The right column shows the result of AGGAN.

results using GANs shows that GANs can mitigate the class imbalance problem. From the perspective of IR, we can find that different GANs perform basically the same when the IR is low, 10. And with the degree of imbalance increases, AGGAN's experimental results will be better. When the IR reaches 100, AGGAN performs best in all datasets. Figure 3 shows three commonly used indicators (precision, recall, and F1-scores) for different datasets under different methods. Here we focus on analyzing the recall of the minority class and the precision of the majority class. When the data is imbalanced, those two indicators are very low in both six datasets. This is because the classifier wants to achieve a high accuracy so that it judges more data as the majority class. The methods used in the experiment can improve these indicators to a certain extent, but the proposed method is better than the other three methods, which is especially obvious in the recall of minority classes. This provides solid evidence that AGGAN can generate more realistic minority samples which can improve the performance of the classifier significantly. And the classification results had

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3120642, IEEE Transactions on Multimedia

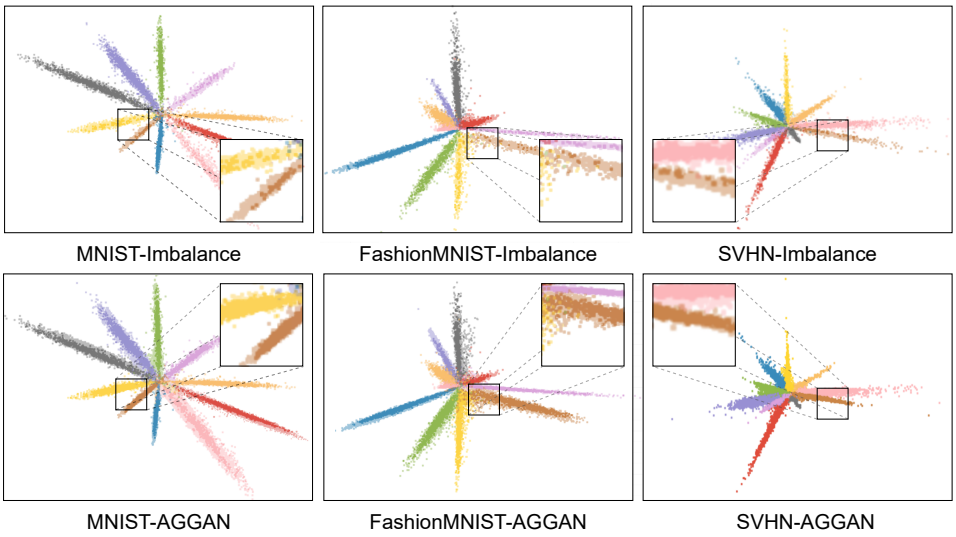

Fig. 7. Feature visualization for MNIST, Fashion-MNIST, and SVHN. The features are obtained by forwarding images to a classifier pre-trained on a balanced dataset.

been demonstrated to be statistically significant based on t-tests (P-values < 0.05).

Table V reports results obtained on multi-class classification. We can see that AGGAN can still perform well on more complex imbalanced datasets and achieve the state-of-the-art level. Among the five methods of using GANs, E-GAN adopts the idea of evolution, and AGGAN further combines the methods of genetic and simulated annealing. Although AGGAN uses the same adversarial learning objectives as EGAN, the simulated annealing idea is unique to our method, and this is also the key to escape from the local optimum and achieve better performance. Figure 4 shows the smallest minority class images of MNIST and Fashion-MNIST generated by GANs methods. There are only 40 training samples of these classes in the training set. Overall, AGGAN is observed to perform better than other GANs. The quality of images generated by ACGAN is significantly the lowest. The quality of images generated by E-GAN has been improved, but the diversity is obviously insufficient. Especially in the Fashion-MNIST dataset, the images generated by E-GAN are almost all flat boots. On the contrary, images generated by LSGAN have better diversity but lower quality. The quality of images generated by WGAN has been improved compared with LSGAN. But in MNIST dataset, when the given label is the number 9, i.e. the smallest class, WGAN will still generate some images belonging to the largest class, i.e. the number 0. Compared with the other four GANs, the images generated by AGGAN have a higher diversity and quality. These results indicate that, compared with ACGAN, EGAN, LSGAN, and WGAN, AGGAN can learn the distribution closest to the real data, thus generating more realistic and diverse images. Figure 5 shows the performance of different methods in the SVHN and CIFAR experiments on the smallest class recall and the largest class precision. These two indicators will be higher when more minority samples are correctly classified. The results show that AGGAN can significantly improve the performance of the classifier. Figure 6 shows the confusion matrix for multi-class imbalanced classification results of CelebA and LSUN. We observe that under the baseline classifier network, the minority classes can be easily misclassified. And AGGAN can obviously make more minority samples be correctly classified.

C. Visualization

In Figure 7, we visualize the second last layer features of images sampled from multi-class imbalanced datasets, testing datasets, and datasets after the augmentation by the proposed method for the minority classes. We obtain those features by forwarding images to a classifier pre-trained on a balanced dataset, and features with a specific category in each figure are represented in the same color. For a more intuitive display, we overlap the feature maps from the testing datasets with the other two and use translucent dots of corresponding colors to represent the testing datasets. The first row shows the original imbalanced training dataset, where the minority training samples can only cover a part of the minority distribution of the testing dataset. As a result, many samples belonging to the minority class (points that are not covered) will be misclassified with a high probability. The second row shows the feature of datasets that the minority classes are expanded by AGGAN. We can see that the minority samples in the two datasets can almost overlap. That is the reason why the performance of the classifier can be significantly improved. This shows the effectiveness of the AGGAN from the perspective of data distribution.

D. Training stability and complexity analysis

In Figure 8 we visualize the convergence and computational complexity of the five GANs based methods. As shown in Figure 8-left, AGGAN (the yellow line) can achieve higher accuracy with fewer training steps. Meanwhile, AGGAN also shows comparable stability when it goes to convergence. In

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2021.3120642, IEEE Transactions on Multimedia

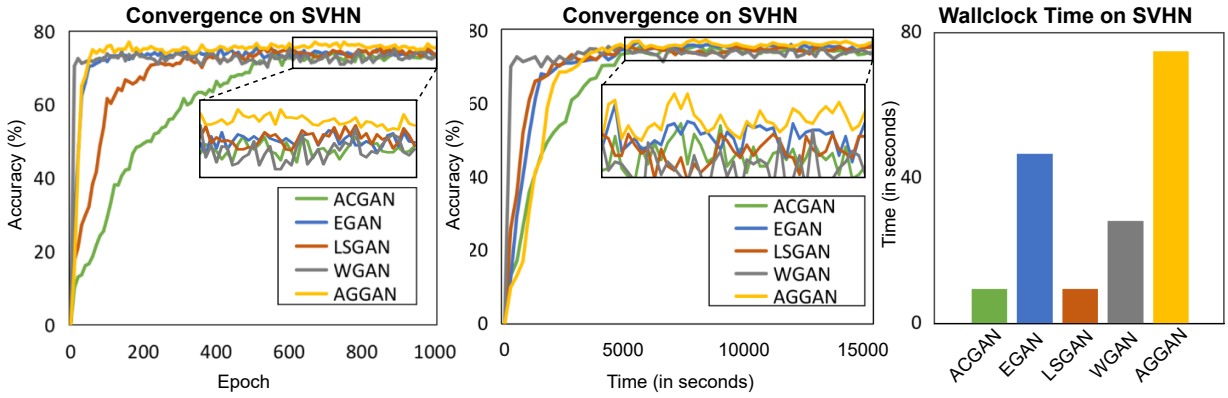

Fig. 8. Experiments on the SVHN dataset. Accuracy over training epoch (left), over wallclock time (middle), and the wallclock time required for a training epoch (right).

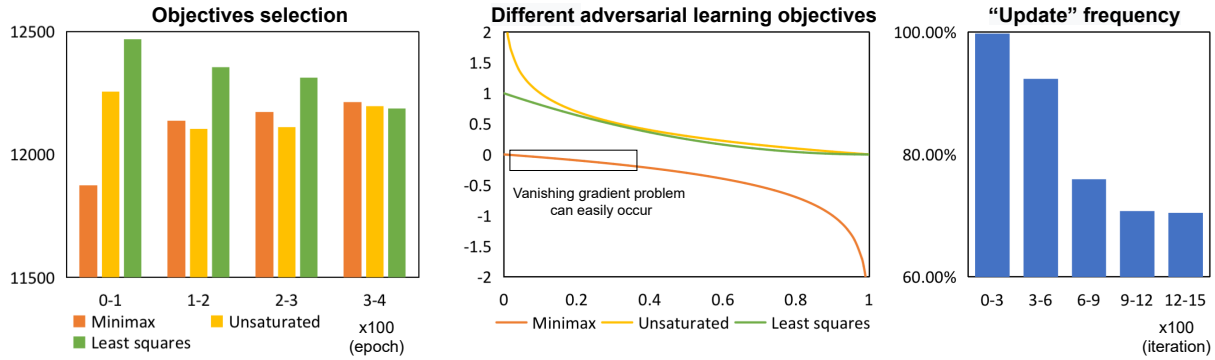

Fig. 9. Experiments on the SVHN dataset. The objectives selection in the training process of AGGAN (left), different objectives used in AGGAN training (middle), and the "Update" frequency in the training process of AGGAN (right).

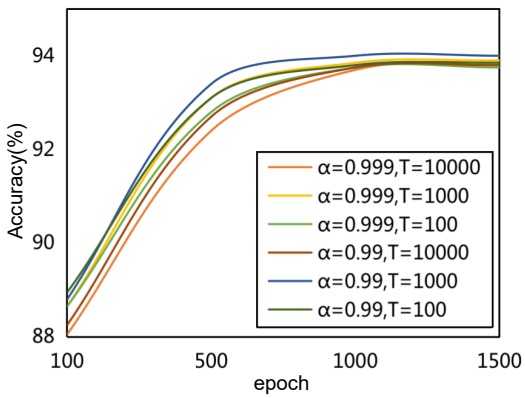

Fig. 10. Simulated annealing hyper-parameter analysis.

wallclock time experiments (Figure 8-middle and Figure 8-right), we can see that although the time required for one epoch of AGGAN is 7.8 times that of ACGAN, 7.9 times that of LSGAN, 2.7times that of WGAN, and 1.6 times that of EGAN, AGGAN still can achieve comparable convergence speed. This experiment further demonstrates the advantages of AGGAN. It can leverage the strengths of different adversarial learning objectives and the idea of simulated annealing algorithm to improve the performance of learning distribution.

We summarize the selected objectives for each iteration as Figure 9-left. We can see that, at the beginning of training, the minimax objective is selected less frequently than others. This may be because at the beginning of training, when the discriminator can easily determine the data source, the gradient provided by the minimax objective is easy to disappear, as shown in Figure 9-middle. As the generator converges, the frequency of selection of different objectives gradually approaches. And this method of using different objectives to training GANs can improve the stability of convergence. Figure 9-right shows the frequency of updating the best offspring to the generator of the next iteration. The decreasing probability of updating enables AGGAN to asymptotically converge to the global optimum.

E. Hyper-parameters analysis

As shown in Algorithm 1, the initial temperature T and the annealing coefficient α are the key parameters in the simulated annealing algorithm. We conducted experiments with different initial temperatures and annealing coefficients on the CIFAR-10 with IR 10. Figure 10 shows the accuracy of AGGAN under different hyper-parameters at different training epochs. We use the exhaustive search, and the search range for hyper-parameters are $T \in \{100, 1000, 10000\}$, $\alpha \in \{0.99, 0.999\}$. We can see that a smaller value of T and α can make the

model converge faster at the beginning of the training process. To the opposite, the larger T and α ($T = 10000$,$\alpha = 0.999$) will make the model convergence slower at the beginning. But after training for a certain period, the models with different hyper-parameters can all converge to a higher result. These indicate that AGGAN is robust to the different values of hyper-parameters.

VI. Conclusion

The proposed AGGAN is a novel training strategy for GANs, which can help GANs escape from the local optimum especially when the training dataset is imbalanced. We analyzed the AGGAN through theoretical proof. And experiments show that AGGAN achieves convincing performance in several imbalanced classification tasks. An interesting area of future investigation is to help GAN learn the target distribution under more complex and extreme data, such as classification of distorted test set[46].

References

[1] V. Rotemberg, N. Kurtansky, B. Betz-Stablein, L. Caffery, E. Chousakos, N. Codella, M. Combalia, S. Dusza, P. Guitera, D. Gutman *et al.*, "A patient-centric dataset of images and metadata for identifying melanomas using clinical context," *arXiv preprint arXiv:2008.07360*, 2020.

[2] 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.

[3] Y. Sun, M. S. Kamel, A. K. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognition*, vol. 40, no. 12, pp. 3358–3378, 2007.

[4] C. Chen and M.-L. Shyu, "Clustering-based binary-class classification for imbalanced data sets," in *2011 IEEE International Conference on Information Reuse & Integration*. IEEE, 2011, pp. 384–389.

[5] M. A. Mazurowski, P. A. Habas, J. M. Zurada, J. Y. Lo, J. A. Baker, and G. D. Tourassi, "Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance," *Neural networks*, vol. 21, no. 2-3, pp. 427–436, 2008.

[6] A. F. Jerant, J. T. Johnson, C. D. Sheridan, and T. J. Caffrey, "Early detection and treatment of skin cancer," *American family physician*, vol. 62, no. 2, pp. 357–368, 2000.

[7] S. Ando and C. Y. Huang, "Deep over-sampling framework for classifying imbalanced data," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2017, pp. 770–785.

[8] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5907–5915.

[9] Q. Mao, H.-Y. Lee, H.-Y. Tseng, S. Ma, and M.-H. Yang, "Mode seeking generative adversarial networks for diverse image synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[10] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[11] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4681–4690.

[12] G. Mariani, F. Scheidegger, R. Istrate, C. Bekas, and C. Malossi, "Bagan: Data augmentation with balancing gan," *arXiv preprint arXiv:1803.09655*, 2018.

[13] S. S. Mullick, S. Datta, and S. Das, "Generative adversarial minority oversampling," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 1695–1704.

[14] J. Hao, C. Wang, H. Zhang, and G. Yang, "Annealing genetic gan for minority oversampling," in *The 31st British Machine Vision Virtual Conference*, 2020.

[15] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," *Stat*, vol. 1050, 2017.

[16] 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.

[17] 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.

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

[19] R. Blagus and L. Lusa, "Smote for high-dimensional class-imbalanced data," *BMC Bioinformatics*, vol. 14, no. 1, p. 106, Mar 2013.

[20] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2794–2802.

[21] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[22] V. Nagarajan and J. Z. Kolter, "Gradient descent gan optimization is locally stable," in *Advances in Neural Information Processing Systems*, 2017, pp. 5585–5595.

[23] C. Wang, C. Xu, X. Yao, and D. Tao, "Evolutionary generative adversarial networks," *IEEE Transactions on Evolutionary Computation*, 2019.

[24] H.-L. Liu, L. Chen, Q. Zhang, and K. Deb, "Adaptively allocating search effort in challenging many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 3, pp. 433–448, 2017.

[25] M. Mitchell, *An introduction to genetic algorithms*. MIT press, 1998.

[26] D. Adler, "Genetic algorithms and simulated annealing: A marriage proposal," 1993.

[27] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches," *IEEE Transactions on Systems Man & Cybernetics Part C Applications & Reviews*, vol. 42, no. 4, pp. 463–484, 2011.

[28] L. R. Rere, M. I. Fanany, and A. M. Arymurthy, "Simulated annealing algorithm for deep learning," *Procedia Computer Science*, vol. 72, no. 1, pp. 137–144, 2015.

[29] C. Tao, S. Dai, L. Chen, K. Bai, J. Chen, C. Liu, R. Zhang, G. Bobashev, and L. C. Duke, "Variational annealing of gans: A langevin perspective," in *International conference on machine learning*, 2019, pp. 6176–6185.

[30] H. Chen and N. S. Flann, "Parallel simulated annealing and genetic algorithms: a space of hybrid methods," in *International Conference on Parallel Problem Solving from Nature*. Springer, 1994, pp. 428–438.

[31] L. Ingber and B. Rosen, "Genetic algorithms and very fast simulated reannealing: A comparison," *Mathematical and computer modelling*, vol. 16, no. 11, pp. 87–100, 1992.

[32] Y. R. Elhaddad, "Combined simulated annealing and genetic algorithm to solve optimization problems," 2012.

[33] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[34] E. H. L. Aarts, Á. E. Eiben, and K. Van Hee, "A general theory of genetic algorithms," 1989.

[35] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[36] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[37] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," 2011.

[38] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[39] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[40] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop," *arXiv preprint arXiv:1506.03365*, 2015.

[41] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 2642–2651.

[42] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9268–9277.

[43] C. Huang, Y. Li, C. C. Loy, and X. Tang, "Learning deep representation for imbalanced classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5375–5384.

[44] D. POWERS, "Evaluation: From precision, recall and f-measure to roc, informedness, markedness & correlation," 2011.

[45] M. Kubat, S. Matwin *et al.*, "Addressing the curse of imbalanced training sets: one-sided selection," in *Icml*, vol. 97. Citeseer, 1997, pp. 179–186.

[46] T. Guo, C. Xu, B. Shi, C. Xu, and D. Tao, "Learning from bad data via generation," in *Advances in Neural Information Processing Systems*, 2019, pp. 6044–6055.