Noname manuscript No.
(will be inserted by the editor)

Improved Balancing GAN: Minority-class Image Generation

Gaofeng Huang* · **Amir Hossein Jafari***
{ghuang920, ajafari}@email.gwu.edu

Received: date / Accepted: date

Abstract Generative adversarial networks (GANs) are one of the most powerful generative models, but always require a large and balanced dataset to train. Traditional GANs are not applicable to generate minority-class images in a highly imbalanced dataset. Balancing GAN (BAGAN) is proposed to mitigate this problem, but it is unstable when images in different classes look similar, e.g. flowers and cells. In this work, we propose a supervised autoencoder with an intermediate embedding model to disperse the labeled latent vectors. With the improved autoencoder initialization, we also build an architecture of BAGAN with gradient penalty (BAGAN-GP). Our proposed model overcomes the unstable issue in original BAGAN and converges faster to high quality generations. Our model achieves high performance on the imbalanced scale-down version of MNIST Fashion, CIFAR-10, and one small-scale medical image dataset. [1]

Keywords GAN · Imbalanced Data · Image Generation · Data Augmentation · Medical Image

1 Introduction

Image classification is a classical topic in computer vision. There are many state-of-the-art networks proposed in the ImageNet challenge [1]. These deep neural networks commonly require a large and balanced dataset for training. However, in medical image classification, the performance of most networks will deteriorate due to the imbalanced dataset. The underlying idea of neural networks is minimizing the loss function via gradient descent. When training on an imbalanced dataset, the gradients will easily fall into the trap of predicting majority. Apart from reducing majority-class samples, to the best of our knowledge, the only effective solution is increasing the samples of minority. In the field of medical images, collecting pathological cases is time-consuming. The best solution is generating new minority-class images with high quality and with diversity.

Generative adversarial networks (GANs) [2] are currently the most powerful generative models. As one of deep neural networks, GANs also require a large dataset for training. However, the minority-class subset is always insufficient to train a good GAN. In particular, balancing GAN (BAGAN) [3] provided a new method to train GANs on imbalanced datasets while specifically aiming to generate minority-class images in high quality. The main contributions of BAGAN are: 1. using an autoencoder to initialize the GAN training, which gives the GAN a common knowledge of all classes, 2. combining real/fake loss and classification loss fairly into one output at the discriminator, which ensures a balanced training for each class.

- *Problem statement*

Although BAGAN proposed an autoencoder initialization to stabilize the GAN training, sometimes the performance of BAGAN is still unstable especially on medical image datasets. Medical image datasets are always: 1. highly imbalanced due to the rare pathological cases, 2. hard to distinguish the difference among classes. As shown in [3], the imbalanced *Flowers* dateset has many similar classes so that BAGAN performs not well. In our experiments, BAGAN fails to generate good samples on a small-scale medical image dataset. We consider that the encoder fails to separate images by class

*Data Science Program, The George Washington University, Washington, 20052, DC, USA

[1] https://github.com/GH920/improved-bagan-gp

when translating them into latent vectors. Furthermore, similar to traditional GANs, BAGAN is hard to train and sensitive to its architecture and hyperparameters. Our objective of this work is to generate minority-class images in high quality even with a small-scale imbalanced dataset. Our contributions are:

- We improve the loss function of BAGAN with gradient penalty and build the corresponding architecture of generator and discriminator (BAGAN-GP).
- We propose a novel architecture of autoencoder with an intermediate embedding model, which helps the autoencoder learn the label information directly.
- We discuss the drawbacks of the original BAGAN and exemplify performance improvements over the original BAGAN and demonstrate the potential reasons.

2 Background

Literature review of GANs. Generative adversarial networks (GANs) [2, 4] is a minimax problem, which is one of zero-sum non-cooperative games. A typical GAN model contains a generator and a discriminator. The generator wants to maximize its performance, which works to generate images as real as possible to confuse the discriminator. The discriminator works to distinguish a mixture of original and generated images whether real or fake. In this game, the generator attempts to mimic the distribution of the real data.

GAN techniques are fast developed in recent years. There are various types of GANs: with different metrics of comparing two distributions (e.g. KL divergence for the original GAN [2], Wasserstein distance for WGAN [5, 6], EBGAN [7], BEGAN [8], Loss-Sensitive GAN [9]), with regularization on the loss function (e.g. WGAN-GP [5], DRAGAN [10]), with different well-designed architecture of GANs (e.g. CycleGAN [11, 12], PGGAN [13], SAGAN [14]), with using a single image for generation (e.g. SinGAN [15]), with conditions (e.g. ACGAN [16]), for augmentation (e.g. AugGAN [17], BAGAN [3]), for reducing mode collapse problem (e.g. VEEGAN [18]).

GAN-based augmentation Data augmentation can extract more information from the original datasets to improve the performance of models. Traditional image augmentation is simply applying linear transformations to the original images, e.g. reflections, rotations and shears. If the linear transformations do not affect the recognition of images, it is effective for the models to learn more information on the original dataset. To extract more information, it is also reasonable to apply some non-linear transformations to the original dataset. GANs are exactly good at create similar images by non-linear transformations inside the network. The literature review [19] compared many data augmentation methods in deep learning, especially the methods based on GANs.

GANs can simulate the distribution of the real dataset and generate new data samples with high quality. Therefore, there are some recent work applying GANs as an augmentation technique. However, the small training set of minority-class images is still a challenge to train a GAN to generate high quality samples. AugGAN [17] and AugCGAN [12] proposed an image-to-image translation framework to generate images in target domain. BAGAN [3] proposed an overall approach to generate minority-class images with high quality to balance the original dataset. [20] used conditional WGAN-GP (cWGAN-GP) to generate face emotion samples for data augmentation. [21] discussed the importance of data augmentation in medical image analysis and considered GANs as the most promising technique. For brain tumor images synthesis, [22] used GANs and [23] used conditional PGGAN for better tumor detection.

3 Methods

3.1 BAGAN architecture.

Autoencoder initialization. Autoencoder initialization helps generator and discriminator to build a common knowledge of the dataset among all classes. Besides, autoencoder will lead the initialized GAN to a good and stable solution. BAGAN uses a typical autoencoder, the encoder translates a given image into a latent vector and the decoder translates a given latent vector back to a reconstructed image. It applies *L2* loss minimization between real images and reconstructed images to train the autoencoder networks. In

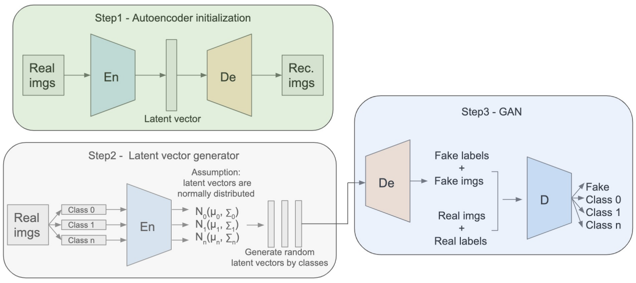

Fig. 1 The architecture of BAGAN. BAGAN proposed three effective steps to improve the quality of generated images when training GANs on imbalanced datasets.

this step, there is no information about classes and the autoencoder learns all images unsupervisedly.

Labeled latent vectors generation. In this step, the class information is attached to each latent vector. The real images can be divided into different classes. Using the encoder to translate these images into latent vectors. With an assumption that these latent vectors are normally distributed within their own classes, a probabilistic generator can be derived by calculating means and covariances w.r.t classes.

Balanced training in GAN. The generator and the discriminator have prior knowledge from the initialized autoencoder. The generator inherits the same architecture and weights from the trained decoder. The discriminator inherits the same weights of the trained encoder as the first part and adds an auxiliary *softmax* layer to identify different classes. Differently from ACGAN [16], the discriminator has only one output but it can classify real/fake and other real classes. Furthermore, in each training batch, the proportion of fake images is the same as any other class. It means the gradients propagated equally for each class and real/fake validity. Although the majority-class images are easier for GAN to learn and to generate real-like images, the balanced training guarantees that the minority-class images will not be ignored.

3.2 *Improvements on BAGAN*

3.2.1 Improved loss function.

In this work, we will use two advanced loss functions with gradient penalty (from WGAN-GP [5] and DRAGAN [10]) to compare against the original loss function of BAGAN.

Original GAN. In original GAN model, the loss function is based on KL-JS divergence. Using cross-entropy loss to minimize the difference between two distribution is equivalent to minimizing the KL-JS divergence. However, KL-JS divergence can only give meaningful gradients when two distributions have overlaps. KL-JS divergence cannot measure how far two distributions away when they have no intersections. The loss function $L\left(X_r, X_g\right)$ of original GAN is defined as:

$$\underset{\theta_G}{\min}\underset{\theta_D}{\max} L\left(X_r, X_g\right) = \mathbb{E}_{x_r \sim X_r}\left[\log\left(D\left(x_r\right)\right)\right] + \mathbb{E}_{x_g \sim X_g}\left[\log\left(1 - D\left(x_g\right)\right)\right] \quad (1)$$

where D denotes the discriminator function, G denotes the generator function, θ_G is the parameters of the generator, θ_D is the parameters of the discriminator; x_r is sampled from the real distribution X_r, x_g is sampled from the generated distribution X_g, where $x_g = G\left(z\right)$ and z is a random noise vector sample from normal distribution$z \sim N\left(0, I_{dim(z)}\right)$. The discriminator is minimizing:

$$L^{(D)}\left(X_r, X_g\right) = -\mathbb{E}_{x_r \sim X_r}\left[\log\left(D\left(x_r\right)\right)\right] - \mathbb{E}_{x_g \sim X_g}\left[\log\left(1 - D\left(x_g\right)\right)\right] \quad (2)$$

The generator is minimizing:

$$L^{(G)}\left(X_g\right) = -\mathbb{E}_{x_g \sim X_g}\left[\log\left(D\left(x_g\right)\right)\right] \quad (3)$$

WGAN. For the loss function, we can replace the KL divergence by the Wasserstein distance to improve the performance and training stability. In practice of constructing an original GAN, the architecture of discriminator is not suggested to be very powerful. A powerful discriminator cannot give meaningful gradients when training its generator. WGAN [6] proposed the Wasserstein distance to solve this problem. Wasserstein distance is the minimum transport cost of moving mass from one distribution to another distribution, which is also called as Earth-Mover Distance (EMD). EMD is continuous and differentiable so that the gradients are always meaningful, which ensures the stability of the GAN training. Based on the theory of WGAN, the generator will eventually converge to the performance of the discriminator. Hence, WGAN requires a deep architecture of the discriminator so that the discriminator can reach the optimal critic performance. The EMD is defined as:

$$W\left(X_r, X_g\right) = \inf_{\gamma \sim \Pi\left(X_r, X_g\right)} \mathbb{E}_{\left(x_r, x_g\right) \sim \gamma} \left\| x_r - x_g \right\| \quad (4)$$

where $\Pi\left(X_r, X_g\right)$ denotes all possible joint distributions between the real distribution X_r and the generated distribution X_g. Each γ represents a transport plan.

However, it is impossible to find the lower bound by traversing all the possible γ in this equation. Using the Kantorovich-Rubinstein duality, it is equivalent to find the upper bound in:

$$W\left(X_r, X_g\right) = \sup_{\|D\|_L \leq 1} \left(\mathbb{E}_{x_r \sim X_r}\left[D(x_r)\right] - \mathbb{E}_{x_g \sim X_g}\left[D(x_g)\right]\right) \quad (5)$$

where $\|D\|_L \leq 1$ denotes D belongs to the set of 1-Lipschitz functions. Without the constraint, the objective function for the discriminator is maximizing:

$$W^{(D)}\left(X_r, X_g\right) = \mathbb{E}_{x_r \sim X_r}\left[D\left(x_r\right)\right] - \mathbb{E}_{x_g \sim X_g}\left[D\left(x_g\right)\right] \quad (6)$$

The discriminator in WGAN uses an unconstrained real number rather than a classification probability to measure the validity of real/fake images. The loss function of the WGAN does not have a *log-sigmoid* functions comparing to the original GAN.

Gradient penalty. 1-Lipschitz constraint is equivalent to the norm of gradients $\|\nabla_x D(x)\|_2 \leq 1$ everywhere. The gradient penalty term is defined as:

$$GP = \mathbb{E}_{x \sim X}\left[(\|\nabla_x D(x)\|_2 - 1)^2\right] \tag{7}$$

In WGAN-GP [5], they add an extra gradient penalty term to the discriminator loss function. The loss function for the discriminator is minimizing:

$$\begin{aligned} W^{(D)}(X_r, X_g) = \mathbb{E}_{x_r \sim X_r}[D(x_r)] - \mathbb{E}_{x_g \sim X_g}[D(x_g)] \\ + \lambda \mathbb{E}_{\hat{x} \sim \hat{X}}\left[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2\right] \end{aligned} \tag{8}$$

where $\widehat{x} = \alpha x_r + (1-\alpha)\, x_g, \alpha \sim U(0,1)$, which we refer to as "model interpolation," λ is a hyperparameter of the penalty extent.

Gradient penalty is only applied in the discriminator loss. The loss function for generator is minimizing:

$$W^{(G)}(X_g) = -\mathbb{E}_{x_g \sim X_g}[D(x_g)] \tag{9}$$

DRAGAN [10] borrowed the idea of gradient penalty from WGAN-GP [5] . [5] indicated the gradient penalty term can be adapted to standard GAN loss function Equation 1. [10] applied the gradient penalty based on the Wasserstein distance to the original *log-sigmoid* loss function and [24] demonstrated it is also effective. The loss function for the discriminator is minimizing:

$$\begin{aligned} L^{(D)}(X_r, X_g) = & - \mathbb{E}_{x_r \sim X_r}[\log(D(x_r))] \\ & - \mathbb{E}_{x_g \sim X_g}[\log(1 - D(x_g))] \\ & + \lambda \mathbb{E}_{\hat{x} \sim \hat{X}}\left[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2\right] \end{aligned} \tag{10}$$

where $\widehat{x} = \alpha x_r + (1-\alpha)\, x_{noise}, \alpha \sim U(0,1), x_{noise} \sim p_{noise}$, which we refer to as "noise interpolation." Although DRAGAN modified the gradient penalty comparing with WGAN-GP, we will not discuss deeply on the difference.

There is no gradient penalty in the generator loss, so the loss function is the same as the original GAN:

$$L^{(G)}(X_g) = -\mathbb{E}_{x_g \sim X_g}[\log(D(x_g))] \tag{11}$$

With comparison of these loss functions in practice, our improved BAGAN uses a DRAGAN-like loss function with the "model interpolation" gradient penalty.

With conditionality. For data augmentation, we need to apply conditional GAN to generate minority-class samples. The architecture of DRAGAN and WGAN-GP are almost the same. Referring to ACGAN [16] and cWGAN-GP [20], we built a feasible architecture for conditional DRAGAN (cDRAGAN). Due to the existence of gradient penalty, we cannot add *softmax* layer to the end of the discriminator to identify different classes. The output of the discriminator still needs to be an unconstrained real number. In our work, we keep the output of the generator and the discriminator the same as WGAN-GP whereas we attach the label information into the input of the generator and the discriminator. The label information is expanded by an *embedding* layer and combined with other inputs by a *multiply* layer. The loss function for the discriminator:

$$\begin{aligned} L^{(D)}(X_r, X_g, Y_r) = -\mathbb{E}_{(x_r, y_r) \sim (X_r, Y_r)}[\log(D(x_r, y_r))] \\ -\mathbb{E}_{(x_g, y_r) \sim (X_g, Y_r)}[\log(1 - D(x_g, y_r))] \\ +\lambda \mathbb{E}_{(\hat{x}, y_r) \sim (\hat{X}, Y_r)}\left[(\|\nabla_{(\hat{x}, y_r)} D(\hat{x}, y_r)\|_2 - 1)^2\right] \end{aligned} \tag{12}$$

Similar to ACGAN and cWGAN-GP, the generated images use the real labels for training in both G and D. The loss function for the generator:

$$L^{(G)}(X_g, Y_r) = -\mathbb{E}_{(x_g, y_r) \sim (X_g, Y_r)}[\log(D(x_g, y_r))] \tag{13}$$

Combine with BAGAN. BAGAN has state-of-the-art performance of generating minority-class images on imbalanced datasets. The GAN architecture in BAGAN is just a typical conditional GAN. We improved the GAN part in BAGAN by adopting the architecture and loss function from the cDRAGAN proposed in the previous section. The loss function is modified by the idea of balanced training from BAGAN. The loss function of the discriminator:

$$\begin{aligned} L^{(D)}(X_r, Z, Y_r, Y_f, Y_{wrong}) = \\ -\mathbb{E}_{(x_r, y_r) \sim (X_r, Y_r)}[\log(D(x_r, y_r))] \\ -\mathbb{E}_{(z, y_f) \sim (Z, Y_f)}[\log(1 - D(G(z, y_f), y_f))] \\ -\mathbb{E}_{(x_r, y_{wrong}) \sim (X_r, Y_{wrong})}[\log(1 - D(x_r, y_{wrong}))] \\ +\lambda \mathbb{E}_{(\hat{x}, y_r) \sim (\hat{X}, Y_r)}\left[(\|\nabla_{(\hat{x}, y_r)} D(\hat{x}, y_r)\|_2 - 1)^2\right] \end{aligned} \tag{14}$$

where z is a random noise vector $z \sim N\left(0, I_{\dim(z)}\right) \equiv Z$, $y_f \sim U\{0,1,2,...\} \equiv Y_f$ and $y_{wrong} \sim U\{0,1,2,...\ \} \equiv Y_{wrong}$. Previously, the real labels are shared with the real images and the fake images when training the discriminator. In an imbalanced dataset, the real labels randomly sampled from the dataset are still imbalanced. Hence, the GAN will automatically train more on the majority classes. In practice, if we sample from the stratified real labels for training, the GAN will learn slowly. Referring to BAGAN, we randomly sample a fake label from a balanced-label set Y_f for each fake image. In order to enhance the learning of class information from the real dataset, we add an extra cross-entropy loss of wrongly classified cases. For the gradient penalty term, we borrow the "model interpolation" method from WGAN-GP.

In the setting of balanced training, the loss function of the generator becomes:

$$L^{(G)}(Z, Y_f) = -\mathbb{E}[\log(D(G(z, y_f)))] \tag{15}$$

3.2.2 Improved autoencoder

BAGAN has two key steps comparing with ordinary conditional GAN: autoencoder initialization and labeled latent generation. In our work, we design a new autoencoder architecture with an embedding section. In BAGAN, the labeled latent generation is based on the assumption that the latent vectors are normally distributed. This assumption restricts the performance of BAGAN in practice.

1. There might be some overlaps between the latent-vector distributions of different classes Figure 2 . The result is the generated samples based on the intersected latent vectors look like the mixed-class images. In application, we cannot feed a random latent vector into generator to get images by class. Instead, we must calculate a labeled latent vector by means and covariances of encoded training data.

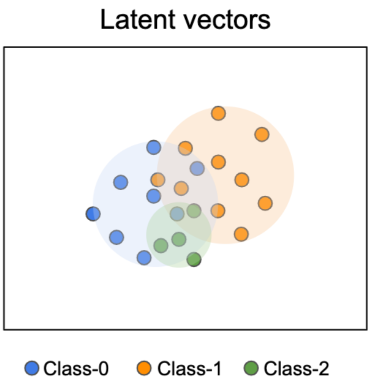

Fig. 2 Distributions of latent vectors in different classes are overlapped

2. The autoencoder does not learn the label information directly in BAGAN. The latent vectors encoded by the autoencoder cannot disperse their own classes. The labeled latent vectors are defined and restricted by their overlapped distributions, i.e. the label information is unclear. Then, the rough label information attached to the latent vectors will mislead the later GAN training. Furthermore, even if we have a perfectly dispersed latent vectors, the labeled latent vectors are only suitable to the trained decoder. Along with the GAN training, the generator (pretrained decoder) will be updated. However, after the autoencoder initialization, the distributions of labeled latent vectors cannot be updated anymore when we train the later GAN model. In our work, we use an embedding model to generate labeled latent vectors.

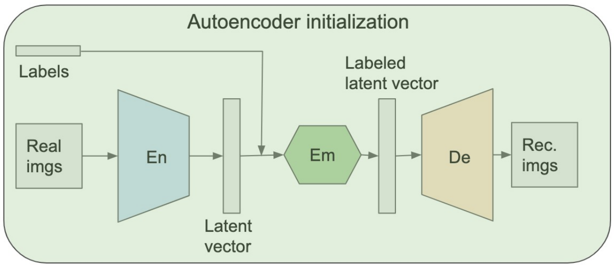

Fig. 3 Autoencoder with an intermediate embedding model. Our proposed autoencoder is supervised. The label information is embedded to a dense vector with the same size of the latent vector. Then, we apply a *multiply* layer to combine these two vectors as a labeled latent vector.

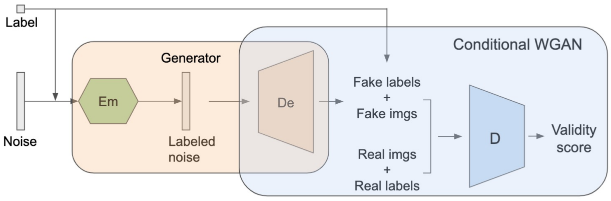

Fig. 4 GAN architecture and our proposed generator. Our proposed generator is an aggregate model of the pretrained embedding model and decoder model. We feed a random latent vector and a random label into the generator and get a generated image in specific class. The embedding model inside the generator can be updated with GAN training.

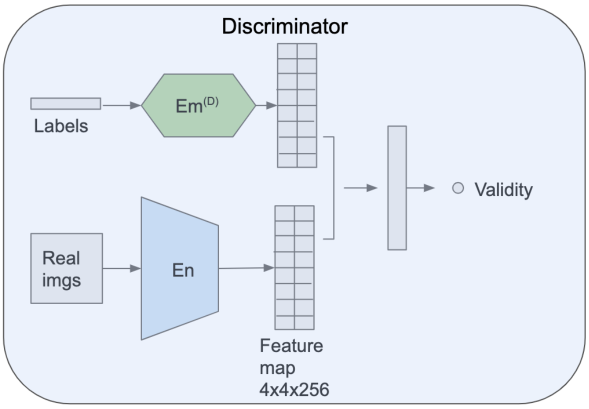

Fig. 5 The discriminator architecture is similar to cWGAN-GP. Our proposed discriminator is an extended model of the pretrained encoder. To note, the discriminator does not use the whole encoder model. Excluding the output layer in decoder, we adopt the second-last output (feature map) and combine the feature map with the embedded labels as a new dense vector. The output of the discriminator is an unconstrained real number, which indicates the total validity of real/fake and class-matching.

4 Experiments and Results

The optimizer for our models in this work is Adam algorithm with learning rate 0.0002 and momentum (0.5, 0.9). The size of mini-batches is 128. All the image inputs will be resized as $64 \times 64 \times channels$. The dimension of default latent vector is 128. We only use batch normalization in the generator/decoder. Except the generator's output activation function is *tanh* while the discriminator's is *linear*, other activation functions are *LeakyReLU* with threshold 0.2. Quality of generated images is measured by Fréchet Inception Distance. The framework of all experiments is Keras with TensorFlow backend. We use an NVIDIA Tesla P4 GPU with 8GB memory. Most of our results are trained within 3600s. For *Cells* dataset, we train 100 epochs and each epoch takes 18s on our device. For *MNIST Fashion* dataset, we train 15 epochs and each epoch takes 154s on our device. For *CIFAR-10* dataset, we train 30 epochs and each epoch takes 129s on our device.

Note. In each figure of representative images at this section, the first row ($row = 0$) shows real images by class. For each column, we feed the generator with class label c_{column}. Start from the second row, we feed the generator with a fixed noise vector z_{row-1}. The generated images in this figure are derived by

$$Im\left(row > 0, column \geq 0\right) = G\left(z_{row-1}, c_{column}\right) \qquad (16)$$

4.1 MNIST Fashion & CIFAR-10

We start with our experiments on two well-known balanced datasets, *MNIST Fashion* and *CIFAR-10*. We first sample 70% of images as the training set for generative models (A for *MNIST Fashion* Table 1, C for *CIFAR-10* Table 2). To exemplify the quality of minority-class generation, an imbalanced version (B for *MNIST Fashion* Table 1, D for *CIFAR-10* Table 2) is created manually for comparison. We observe our model works perfectly not only on the balanced datasets (A, C), but also on the highly imbalanced datasets (B, D). From the representative images Figures 6 and 7 generated with imbalanced datasets, we cannot easily figure out which column is minority class. Therefore, our model has a fair training for each class no matter the imbalanced class weight. The learning outcome only depends on the complexity of the image itself. For example, there are 73 *trousers* and 370 *sandals* in dataset B. Although the training set of *sandals* is 5 times as large as trousers, the generated *trousers* images even have a better quality.

The discriminator in our BAGAN-GP has a similar architecture with WGAN-GP. Hence, we can set the train ratio of the discriminator vs the generator to 5 and boost the training with high stability. In the original BAGAN, we cannot set a train ratio larger than 1. Otherwise, the training of BAGAN will be oscillated. In other words, the stability of BAGAN requires a competitive relation between the generator and the discriminator while our BAGAN-GP only pursues a powerful discriminator to lead the generator. Furthermore, our BAGAN-GP still performs excellently when we only initialize the generator because a good generator will accelerate the learning process of the discriminator.

4.2 Medical image dataset: Cells

Cells dataset is a highly imbalanced medical-image dataset, which contains one majority class and three minority classes Table 3 , i.e. “red blood cell”, “ring”, “schizont” and “trophozoite” respectively. Except the first type, the rest of the cells indicate different stages of malaria infection.

Unlike the images of *MNIST Fashion* and *CIFAR-10*, these four classes are different types of red blood

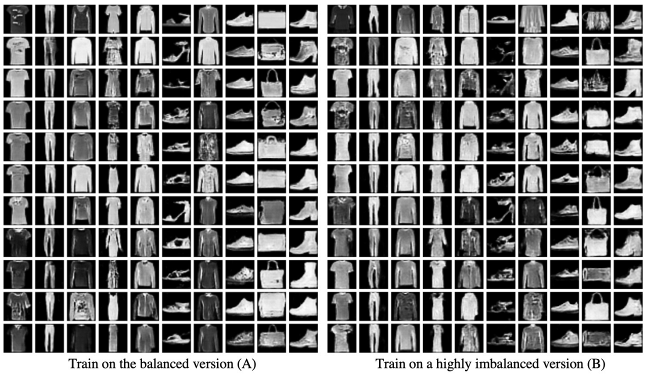
Train on the balanced version (A) Train on a highly imbalanced version (B)

Fig. 6 Representative samples generated in the *MNIST Fashion*. The order of these images follows Equation (16).

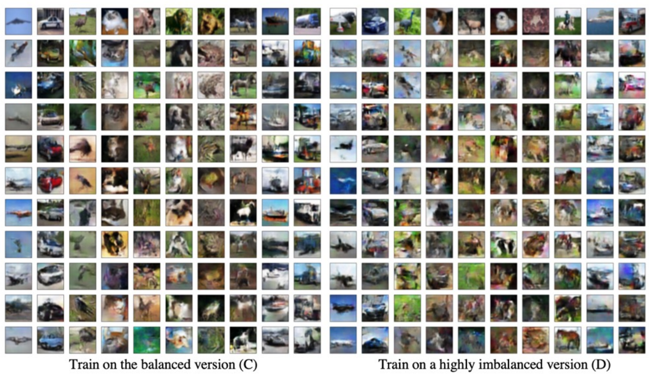
Train on the balanced version (C) Train on a highly imbalanced version (D)

Fig. 7 Representative samples generated in the *CIFAR-10*. The order of these images follows Equation (16).

Table 1 Class weight of *MNIST Fashion* (balanced & imbalanced)

	T-shirt	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Boot
A	4231	4165	4199	4211	4185	4217	4189	4241	4175	4187
B	4166	73	139	210	287	370	422	387	545	651

Table 2 Class weight of *CIFAR-10* (balanced & imbalanced)

	Airplane	Automobile	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck
C	3527	3523	3500	3458	3563	3455	3535	3509	3453	3476
D	3490	71	130	221	269	349	435	485	572	628

Table 3 Class weight of *Cells* dataset

	normal	ring	schizont	trophozoite
	(type 0)	(type 1)	(type 2)	(type 3)
Train	5600	292	106	887
Test	1400	73	27	222

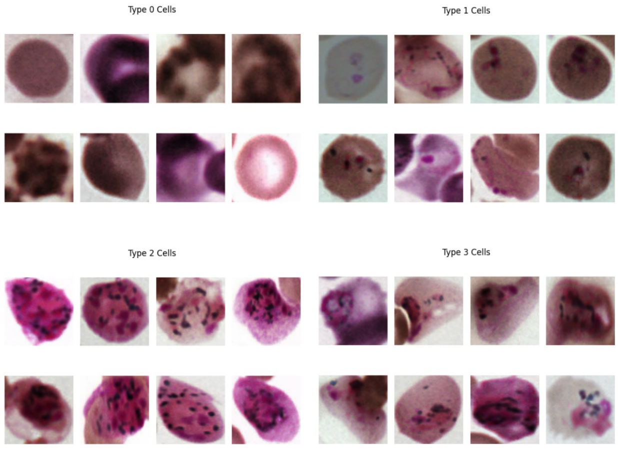

Fig. 8 Real images per class of *Cells* dataset

cells Figure 8. It means they look similar but some different in specific features. Visually, it is hard to distinguish some type 2 cells with type 3 cells.

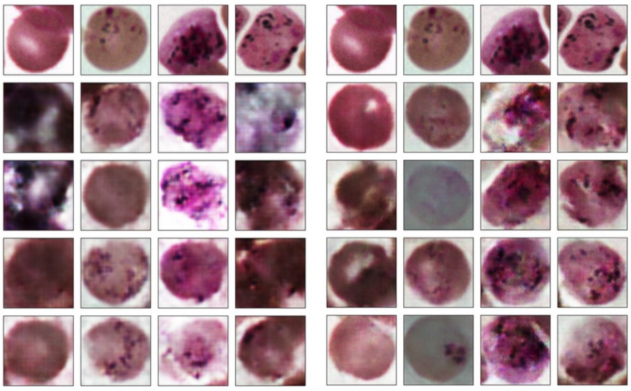

Fig. 9 Generated images by BAGAN (left) and BAGAN-GP (right). The order of these images follows Equation (16).

In Figure 9 , we observe BAGAN is trying to improve the minority-class generation by sacrificing the quality of majority class. It is exactly the objective of BAGAN, but we are not satisfied on this result. With our BAGAN-GP, all types of cells are generated in high quality. In the section 5, we will quantitatively analyze the performance of our model.

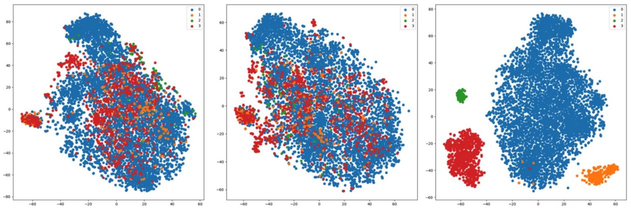

Fig. 10 Two-dimensional t-SNE plot of the encoded latent vectors. Left: Encoder of BAGAN. Middle: Encoder of the improved BAGAN-GP (ours). Right: Encoder + Embedding (ours).

In practice, BAGAN is unstable to train on some imbalanced datasets, especially the medical images datasets, e.g. *Cells* dataset in our experiment. The encoder of the original BAGAN cannot translate the input images into dispersed groups of latent vectors Figure 10. Then, the labeled latent vectors are generated by the distribution of these undivided latent vectors. Thus, the later GAN model will fail to generate images in different classes due to the misleading labeled latent vectors. With our improved autoencoder, we observe that BAGAN becomes stable in training and it is not sensitive to the GAN architecture and hyperparameters.

At the feature-level cognition of ResNet-50 Figure 11, the generated samples can be regarded as effective augmented images. Furthermore, we observe the generated images manifold are equally distributed around the real images manifold. It means, for each class, our generator is not creating one or few modes of images. In other words, the generator comprehensively learns the real data distribution and does not suffer the problem of mode collapse.

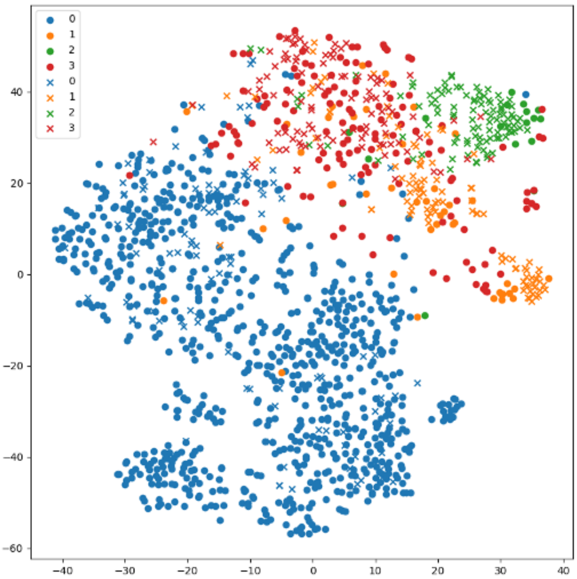

Fig. 11 Comparing the real samples (o) and generated samples (x) by the feature layer output via ResNet-50.

5 Evaluation

– Metric: Fréchet Inception Distance.

There are two common metrics to evaluate the quality of the generated images: Inception Score (IS) [25] and Fréchet Inception Distance (FID) [26]. Both of these two measurements are based on the Inception V3 network, which is pretrained on ImageNet dataset. IS is derived from the classification logits while FID is derived from the feature layer. IS only measures the distance between the generated sample distribution and the ImageNet distribution, whereas FID calculates the feature-level distance between the generated sample distribution and the real sample distribution. In this work, our objective datasets, medical image datasets, are quite different from ImageNet dataset. Therefore, we adopt FID as the evaluation metric. Fréchet Distance is defined as:

$$FID = \|\mu_r - \mu_g\|^2 + Tr\left(\Sigma_r + \Sigma_g - 2\left(\Sigma_r \Sigma_g\right)^{1/2}\right)$$

where μ_r is the mean of the real features, μ_g is the mean of the generated features, Σ_r is the covariance matrix of the real features, Σ_g is the covariance matrix of the generated features.

– FID on *Cells*. Table 4

All FID scores are calculated by the real samples from validation set and the target samples. For comparison, we introduce two baseline FID scores: the reconstructed samples by autoencoder and the real samples from training set. The FID of reconstructed samples is regarded as a lower baseline and the FID of real samples is regarded as an upper baseline. The quality of target samples is higher when its FID is lower.

In the *Cells* dataset, BAGAN can only generate poor samples. Its performance is only better than autoencoder. As we construct our BAGAN-GP model, we first build a cDRAGAN model Equations (12) and (13) and combine cDRAGAN with BAGAN framework to get our final model. We need to demonstrate the combined model is better than the previous independent models. cDRAGAN can generate majority-class images with high quality and ignore the minority, which is the drawbacks of non-BAGAN. When we apply autoencoder initialization to cDRAGAN and keep the same loss function, the BAGAN-GP (v1) can further improve the quality of the majority but there is no improvement on the minority.

Note on BAGAN-GP. (v1): using real labels for generated images Equations (12) and (13) . (v2): feeding balancing labels in generator at training Equations (14) and (15) . (v3): replacing BAGAN original encoder by our encoder. (100/200): the training epochs. 100 epochs for 1800s, and 200 epochs for 3600s.

Comparing BAGAN-GP (v1) with BAGAN-GP (v2), there is a negative effect on the majority-class generation when we apply balanced training to generator, which is analogous to BAGAN. However, the improvement on minority-class generation is significant while the negative effect on majority-class generation is small. If our purpose is generating minority-class images, it is recommended to use balanced training

Table 4 FID: Compare with real samples (validation set)

	Type0	Type1	Type2	Type3
	(1400 samples)	(73 samples)	(27 samples)	(222 samples)
Rec. samples (Autoencoder)	132.715	247.174	322.567	240.519
BAGAN	197.961	213.705	278.755	184.903
cDRAGAN	90.981	184.440	233.512	155.564
BAGAN-GP (v1)	**77.831**	211.698	227.366	168.240
BAGAN-GP (v2)	97.445	152.986	213.864	141.798
BAGAN-GP (v3, 100)	100.151	**143.703**	**195.926**	**112.875**
BAGAN-GP (v3, 200)	**88.562**	**147.994**	**194.544**	**115.881**
Real samples (Train)	20.498	93.721	127.392	58.048

(v2). Otherwise, we can omit the balanced training step to generate highest quality images of the majority class. Many traditional GANs will fail to converge with a long training time. Thanks to the gradient penalty term, our BAGAN-GP is stable during a long training period. We observe the longer training on BAGAN-GP, the better overall performance it will achieve.

Although BAGAN-GP is stable with less hyperparameter tuning, here we give some suggestions to build a better BAGAN-GP for future work. In our experiments, we observe it is not recommended to set a high latent dimension and a complex embedding model. Besides, we suggest the discriminator does not need to inherit the weights from the pretrained encoder. The potential reason is the pretrained encoder is not powerful without the embedding part.

6 Conclusion

In this work, we proposed a new architecture of BAGAN with gradient penalty in loss function. With gradient penalty term, we have a more stable BAGAN in training. For the autoencoder initialization, we proposed a supervised autoencoder with an intermediate embedding model to learn the label information directly, which helps to encode the similar but different-class images dispersedly.

We compared the improved BAGAN-GP against the original BAGAN. From the dispersion of labeled latent vectors to the quality of generated images, our model has stronger performance than the original BAGAN. Besides, our model can handle minority class generation in a wide range of datasets, including medical image datasets.

- Future work

We observe our model can generate images in different classes unambiguously. If we can transfer the class knowledge from generative models to classification models, we believe it will significantly improve the performance of classifiers on imbalanced datasets.

We only use the plain dataset to train the GAN model in this work. In practice, we can apply data augmentation in the step of GAN training, there will be a further improvement on the final results.

There are many research topics dealing with the scarcity of data, such as data augmentation, few-shot and zero-shot learning. We hope our work can broaden the ideas in these topics.

References

1. J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, “ImageNet: A Large-Scale Hierarchical Image Database,” 2009.
2. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.
3. G. Mariani, F. Scheidegger, R. Istrate, C. Bekas, and C. Malossi, “Bagan: Data augmentation with balancing gan,” 2018. [Online]. Available: arXiv:1803.09655
4. J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, “A review on generative adversarial networks: Algorithms, theory, and applications,” 2020. [Online]. Available: arXiv:2001.06937
5. I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, “Improved training of wasserstein gans,” *Advances in neural information processing systems*, pp. 5767–5777, 2017.
6. M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein gan,” 2017. [Online]. Available: arXiv:1701.07875
7. J. Zhao, M. Mathieu, and Y. Lecun, “Energy-based generative adversarial network,” 2016. [Online]. Available: arXiv:1609.03126
8. D. Berthelot, T. Schumm, and L. Metz, “Began: Boundary equilibrium generative adversarial networks,” 2017. [Online]. Available: arXiv: 1703.10717
9. G.-J. Qi, “Loss-Sensitive Generative Adversarial Networks on Lipschitz Densities,” *International Journal of Computer Vision*, vol. 128, no. 5, pp. 1118–1140, 2020. [Online]. Available: https://dx.doi.org/10.1007/s11263-019-01265-2
10. N. Kodali, J. Abernethy, J. Hays, and Z. Kira, “How to train your DRAGAN,” 2017. [Online]. Available: arXiv:1705.07215
11. J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.
12. A. Almahairi, S. Rajeswar, A. Sordoni, P. Bachman, and A. Courville, “Augmented cyclegan: Learning many-to-many mappings from unpaired data,” 2018. [Online]. Available: arXiv:1802.10151
13. T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of gans for improved quality, stability, and variation,” 2017. [Online]. Available:

arXiv:1710.10196

14. H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, “Self-attention generative adversarial networks,” 2018. [Online]. Available: arXiv: 1805.08318
15. T. Shaham, T. Rott, T. Dekel, . Michaeli, J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, “Singan: Learning a generative model from a single natural image,” *A review on generative adversarial networks: Algorithms, theory, and applications*, pp. 4570–4580, 2019.
16. A. Odena, C. Olah, and J. Shlens, “Conditional image synthesis with auxiliary classifier gans,” *International conference on machine learning*, pp. 2642–2651, 2017.
17. S.-W. Huang, C.-T. Lin, S.-P. Chen, Y.-Y. Wu, P.-H. Hsu, and S.-H. Lai, “Auggan: Cross domain adaptation with gan-based data augmentation,” *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 718–731, 2018.
18. A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton, “Veegan: Reducing mode collapse in gans using implicit variational learning,” *Advances in Neural Information Processing Systems*, pp. 3308–3318, 2017.
19. C. Shorten and T. M. Khoshgoftaar, “A survey on Image Data Augmentation for Deep Learning,” *Journal of Big Data*, vol. 6, no. 1, pp. 60–60, 2019. [Online]. Available: https://dx.doi.org/10.1186/s40537-019-0197-0
20. Y. Luo and B.-L. Lu, “EEG data augmentation for emotion recognition using a conditional wasserstein GAN,” *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2535–2538, 2018.
21. S. Kazeminia, C. Baur, A. Kuijper, N. B. V. Ginneken, S. Navab, A. Albarqouni, and Mukhopadhyay, “GANs for medical image analysis,” 2018. [Online]. Available: arXiv: 1809.06222
22. H.-C. Shin, N. A. Tenenholtz, J. K. Rogers, C. G. Schwarz, M. L. Senjem, J. L. Gunter, K. P. Andriole, and M. Michalski, “Medical image synthesis for data augmentation and anonymization using generative adversarial networks,” in *International workshop on simulation and synthesis in medical imaging*. Springer, 2018, pp. 1–11.
23. C. Han, K. Murao, T. Noguchi, Y. Kawata, F. Uchiyama, L. Rundo, H. Nakayama, and S. Satoh, “Learning more with less: Conditional PGGAN-based data augmentation for brain metastases detection using highly-rough annotation on MR images,” *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 119–127, 2019.
24. W. Fedus, M. Rosca, B. Lakshminarayanan, A. M. Dai, S. Mohamed, and I. Goodfellow, “Many paths to equilibrium: GANs do not need to decrease a divergence at every step,” 2017. [Online]. Available: arXiv:1710.08446
25. S. Barratt and R. Sharma, “A note on the inception score.” 2018. [Online]. Available: arXiv:1801.01973
26. D. C. Dowson and B. V. Landau, “The Fréchet distance between multivariate normal distributions,” *Journal of Multivariate Analysis*, vol. 12, no. 3, pp. 450–455, 1982. [Online]. Available: https://dx.doi.org/10.1016/0047-259x(82)90077-x