REALISTIC FACE IMAGE GENERATION BASED ON GENERATIVE ADVERSARIAL NETWORK

TING ZHANG[1], WEN-HONG TIAN[1], TING-YING ZHENG[1], ZU-NING LI[1], XUE-MEI DU[1], FAN LI[1]

[1]School of Information and Software Engineering,
University of Electronic Science and Technology of China, Chengdu 611731
E-MAIL: 285173686@qq.com, tian_wenhong@uestc.edu.cn

Abstract:

Using a computer to generate images with realistic images is a new direction in current computer vision research. This paper designs an image generation model based on the Generative Adversarial Network (GAN). This paper creates a model – a discriminator network and a generator network by eliminating the fully connected layer in the traditional network and applying batch normalization and deconvolution operations. This paper also uses a hyper-parameter to measure the diversity and quality of the generated image. The experimental results of the model on the CelebA dataset show that the model has excellent performance in face image generation.

Keywords:

Generative Adversarial Network; Face image generation; Hyper-parameter

1. Introduction

There has been a surge of interest in the field of computer vision over the past few years, most notably in the area of generate images automatically with real visual effects by computers. With the increasing use of deep learning in the field of computer vision, automatic image generation based on depth models has received more and more attention. By using the powerful learning ability of the depth model, the inherent distribution law in the data can be efficiently mined to generate images with similar distribution. These generated images can be applied to different scenes such as automatic synthesis of images, face image prediction of different ages, and obtaining artistic pictures. In addition, high-quality generated images can also be used to expand the amount of image data in the dataset, which can alleviate the need for a -large number of training samples during deep learning model training, so that practical applications such as face recognition have higher accuracy.

In the current research on the automatic generation of images, the idea of using GAN is widely adopted. The discriminator and the generator will eventually get balance by their confrontation to capture the distribution of data.

We obtain the model by using Deep Convolutional Neural Network and Deep Transpose Convolutional Neural Network respectively as a discriminator and generator. We create an image generation model based on GAN, tests it on the CelebA dataset, and compares the model with the results of Deep Convolution Generative Adversarial Networks (DCGAN) and Wasserstein Generative Adversarial Networks (WGAN). The experimental results show that our model has better performance on image generation.

2. Related Work

For deep learning algorithms, in order to understand the input data, they need to learn to create the data. The most promising approach is to use a generation model that learns to discover the rules of the data and find the best distribution to represent it. In addition, by learning to generate models, we can even draw samples that are not in the training set but follow the same distribution.

As a new generation model framework, the Generative Adversarial Network [1] proposed in 2014 can generate a composite image that is better than the previous generation model, and has since become one of the most popular research fields. The Generative Adversarial Network includes two neural networks, a generator and a discriminator, wherein the generator attempts to generate a real sample to spoof the discriminator, and the discriminator attempts to distinguish between the real sample and the generated sample from the generator.

Generating images is by far the most widely studied area of GAN. The main methods of GAN in image generation are hierarchical methods, iterative methods and direct methods. The algorithm under the hierarchical approach uses two generators and two discriminators in its model, where different generators have different purposes, and the relationship between the two generators can be parallel or in series, such as Secure Steganography based on

GAN (SS-GAN) [4]. The iterative methods are different from the hierarchical methods. First, the models generate images from coarse to fine, and each generator regenerates the details of the results. When using the same structure in the generator, the iterative method can use weight sharing between the generators, and the hierarchical methods usually cannot do this. The classic types is StackGAN [5]. The direct methods follow the principle of using a generator and a discriminator in its model, and the structure of the generator and discriminator is straightforward, with no branches. Many of the GAN models fall into this category, such as DCGAN [3], Spectral Normalization (SN-GAN) [7]. Among them, DCGAN [3] is one of classic, since many later models use its structure, and many GAN model network designs like this, such as cGANs [8]. This approach is relatively straightforward to design and implement compared to layered and iterative methods, and generally yields good results.

3. Method

GAN inspires from the two-player game in game theory. The two players in the GAN model are respectively composed of a generative model and a discriminative model. Generate model G to capture the distribution of sample data, and generate a sample similar to real training data with noise z obeying a certain distribution (uniform distribution, Gaussian distribution, etc.). The goal is to generate samples as good as real samples; the discriminant model D is one the classifier that estimates the probability of deriving the samples from the training data (rather than the generated data). If the sample is from real training data, D outputs a large probability; otherwise, D outputs a small probability.

3.1. Network Architecture

In this section, we will introduce the network architecture used.

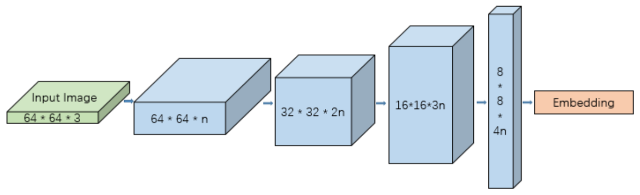

Fig.1 Discriminator Architecture

Discriminator D: We use an auto-encoder with a depth encoder and decoder. We use two convolutions per layer. At each subsampled, the convolution filter is linearly increased, achieve subsampling with stride of two, and do upsampling by nearest neighbors. The discriminator receives the three-channel image, obtains the feature map after multiple convolution operations, maps it to a scalar value after deforming it into a vector, and uses the Sigmoid function to determine the loss function value of the discriminator, as shown in Fig.1.

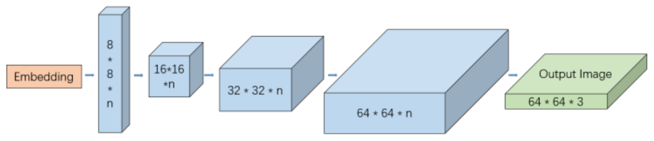

Fig.2 Generator Architecture

Generator G: We use the same architecture as the discriminator decoder, but with different weights. The first layer receives the random noise signal vector for the fully connected layer, and is multiplied by the weight matrix to be transform into a three-dimensional tensor; the remaining layers are deconvolution operation layers, and 2×2 strides are used in the deconvolution operation. The Relu activation function is used, the last deconvolution layer uses the tanh activation function to generate a three-channel image, the structure of which shows in Fig.2.

3.2. Hyper-parameter

Ideally, when the image generated by G is the same as the real image, the result is the best.

However, there is a problem at this time: if the discriminator cannot discriminate between the generated sample and the real sample, then their error distribution and their expected errors will be the same. To solve this problem, we introduce a hyper-parameter γ that can help the generator and the discriminator balance the assignment tasks so that the two sides are balanced and the above problem does not occur.

Thus, the discriminator has two competing goals: self-encoding the real image and distinguishing the generated image from the real image. This hyper-parameter γ balances these two goals. The low γ value will result in poor image diversity, because the discriminator is too concerned about self-encoding the real image, so the best value of γ is between [0, 1].

3.3. Loss Function

Wasserstein Distance. We use a self-encoder as a classifier to match the loss distribution of the self-encoder by the loss based on the Wasserstein distance.

The Wasserstein distance is also called the Earth-Mover (EM) distance and is defined as follows：

$$W\left(P_r, P_g\right) = \inf_{\gamma\sim\pi\prod(P_N P_e)} E_{(x,y)\sim\gamma}[\|x-y\|] \qquad (1)$$

- $\left(P_r, P_g\right)$ is a set of all possible joint distributions of P_r and P_g. Conversely, the edge distribution of each of $\left(P_r, P_g\right)$ is P_r and P_g. For each possible joint distribution γ, a real sample x and a generated sample y can be obtained from the middle sample (x, y) obeying the γ distribution;
- The Wasserstein distance is smooth, and it provides a meaningful gradient when the parameters are optimized using the gradient descent method.
- Calculate the distance of the pair of samples $\|x-y\|$; we can obtain the expected value of the distance of the sample under the joint distribution γ. The lower bound of this mean value in all possible joint distributions. It is the auto-encoder.

G Loss and D Loss. According to the principle of GAN confrontation, the goal of D is to enlarge the distance between the two distributions, that is, to maximize Wasserstein distance: $W\left(P_r, P_g\right)$, and G to minimize it. The loss functions show as follows:

$$\begin{cases} L_D = k_t \times L\left(G(z_D)\right) & for\ \theta_D \\ L_G = L\left(G(z_G)\right) & for\ \theta_G \\ k_{t+1} = \gamma[L(x)\times L_G] + k_t & for\ step\ t \end{cases} \qquad (2)$$

4. Experiments

In this section, we will introduce the train and test results of our two models (64×64 and 128×128) on the celebA dataset, and we will compare the results with the classic DCGAN and WGAN architecture. The specific experimental procedures and results are as follows.

4.1. Dataset

This paper conducted training and testing of the model on the CelebA dataset. The CelebA Face Dataset [2] is the open data of The Chinese University of Hong Kong, which contains 202,599 face images of 10,177 celebrity identities, all of which have been feature-marked, which is a very useful dataset for face-related experiments. The tag identity_CelebA.txt file is the label corresponding to the face, and the list_attr_celeba.txt file is the feature tag in each face map. For example, whether the person wears glasses. We convert the txt file into a csv file, and distinguish the face image according to the csv file. Each folder represents a person, and thus obtains a single picture dataset.

4.2. Parameters Setting

We trained two models, one is 64×64, and the other is 128×128, set both initial learning rate as 1e-4. When we trained the model, we found the results are the best when the learning rate decay is 0.9 every 2000 iter.

4.3. Results

The 64×64 model's result shows as Fig.3. The 128×128 model's result is very impressive, we can see set of teeth, as shown in Fig.4.

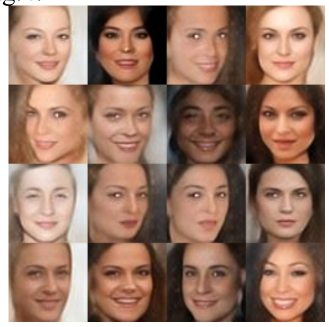

Fig.3 The 64×64 model generated image

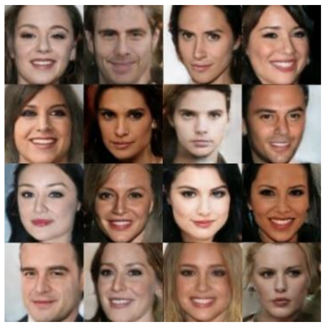

Fig.4 The 128×128 model generated image

We also trained the classic DCGAN and WGAN model, as shown in Fig.5, our model has a better performance in generating face images.

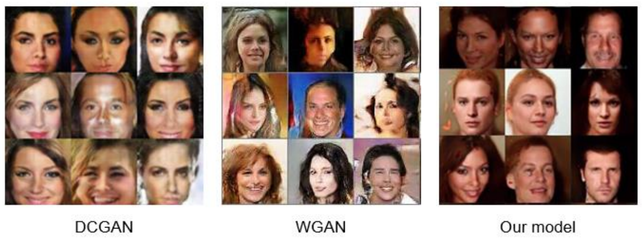

Fig.5 Comparison of the results of the three models

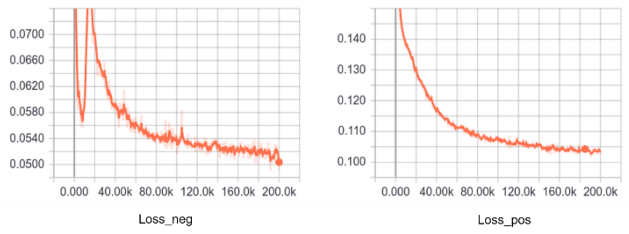

Fig.6 The loss curve

We also use tensorboard to visualize the change of the loss value in the training. The left graph of Fig.6 shows the generated data's loss value through the self-encoder, and the right graph shows that the real data's loss value through the self-encoder. We can see from the figure that the loss curve tends to be stable eventually and the network model converges.

4.4. Test and Verify

High-quality generated images can be used to augment the training image data in the dataset, which can alleviate the need for a large number of training samples during deep learning model training.

Table 1 The results of Face recognition

	Can't detect face	Can detect faces but can't identify who
DCGAN	281/1000	0/1000
WGAN	123/1000	0/1000
Our Model	19/1000	0/1000

As shown in Table 1, in order to verify the realism of the generated face image, we use the open source project on GitHub called Face_recognition[1], which based on the deep learning model dlib, to identify the face images generated by three models (DCGAN, WGAN and our model). The results show that the face image generated by our model has the highest recognition rate, up to 98.10%.

5. Conclusions

This paper proposes a new network architecture based on the Generative Adversarial Network, and designs the image generation model based on the TensorFlow deep learning framework. In the experiment, we used the CelebA dataset to train DCGAN, WGAN, and our model. We also use the open source project on GitHub called Face_Recognition[1] to detect and recognize the faces generated by the three models. The results show that our model has a better performance in face image generation.

Acknowledgements

This work is supported by the Sichuan Science and Technology Program (2018FZ0101, 2018GZDZX0039, 2019ZDZX0009, 2018GZDZX0031) and the Chengdu Science and Technology Program (2018-YF05-01138-GX).

References

[1] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, X. Bing, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, "Generative adversarial nets," in Advances in Neural Information Processing Systems 27, Curran Associates, Inc., 2014, pp. 2672--2680.
[2] Z. Liu, P. Luo, X. Wang and X. Tang, "Deep Learning Face Attributes in the Wild," in The IEEE International Conference on Computer Vision(ICCV), 2015.
[3] A. Radford, L. Metz and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," Computer Science, 2015.
[4] H. Shi, J. Dong, W. Wang, Y. Qian and X. Zhang, "SSGAN: Secure Steganography Based on Generative Adversarial Networks," in Advances in Multimedia Information Processing -- PCM 2017.
[5] 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 The IEEE International Conference on Computer Vision (ICCV), 2017.
[6] M. A. Bottou, S. Chintala and Leon, "Wasserstein Generative Adversarial Networks," in Proceedings of the 34th International Conference on Machine Learning, 2017.
[7] T. Miyato, T. Kataoka, M. Koyama and Y. Yoshida, "Spectral Normalization for Generative Adversarial Networks," CoRR, 2018.
[8] T. Miyato and M. Koyama, "cGANs with Projection Discriminator," CoRR, 2018.

[1] The code for Face_recognition is publicly at: https://github.com/ageitgey/face_recognition