Generative Dog Images Using a Variety of GANs

Chengxuan Lin, Electronic Information,
Junhao Lin, Electronic Information,
Qian Hu, Electronic Information,
Yuhang Lu, Electronic Information,
Liangwei Shi, Electronic Information

Abstract

Generation methods (especially GANs) are currently being used for data expansion in different places and have great potential; they can learn to mimic any distribution of data across any domain: photographs, drawings, music, and prose. Recently, an idea about using GANs to generate pictures of dogs attracted our interest. In this paper, we first introduce the five known GANs architectures in detail, and train these five different GAN under the dataset provided by Kaggle, compare the results of the generated dog images, and put these generated images into the article. Finally, it is found that the effect of BigGAN model is the best.

Introduction

With the step-by-step learning and understanding of deep learning in the classroom, we are more eager to use the knowledge learned in the classroom to explore more practical applications of deep learning. After searching through Kaggle for competitions and datasets, we found a competition called "Generative Dog Images", hoping to combine the knowledge we learned in the classroom, use the data set in the competition, and use the Generative Adversarial Network (GAN) [1] to generate dog images.

Compared with common face generation tasks, such as "CelebFaces Attribute", a dataset of celebrity face attributes, the human face is always facing the camera, and there is little difference in facial features between humans. The data set used in this article includes 20,580 pictures of 120 kinds of dogs, and each dog has a large difference in color, size, and appearance. The biggest challenge is that our target is not only the face of the dog, but also the body shape of the dog. This requires our GAN network to have strong learning and generation capabilities.

In the past few years, GAN has become more and more popular because of its great potential in simulating data distribution. Gan was first described in the paper Generative Adversarial Nets, proposed by Ian Goodfellow and other researchers at the University of Montreal in 2014. GAN understands the world through neural networks and creates new images like never before. It has two components: the generator used to create the image and the discriminator used to evaluate the image. The job of the generator is to generate "fake" images that look like real samples. The job of the discriminator is to determine whether the image is a real training image or a fake image. In the training process, the discriminator first learns the difference between real samples and fake images. Then the generator uses the currently trained discriminator to learn and create fake images. Next, the generated fake images and real samples are input to the discriminator, making the discriminator a better detective to identify more difficult real and fake samples. And the generator keeps trying to make the discriminator unable to judge the authenticity by generating better and better forgeries. Continue to iterate this process until the discriminator has only a 50% probability of judging the authenticity of the fake image, which means that the fake image from the generator is indistinguishable from the real sample [1].

Our research focuses on using generative adversarial networks to generate images of dogs. We have also tried various new technologies to improve the quality of the generated images.

Related works

Generative methods that produce novel samples from high-dimensional data distributions, such as images, are finding widespread use, for example in image-to-image translation , and image in painting. Generative models based on deep learning are common, but GANs are among the most successful generative models (especially in terms of their ability to generate realistic high resolution images). GANs are a framework to produce a model distribution that mimics a given target distribution, and it consists of a generator that produces the model distribution and a discriminator that distinguishes the model distribution from the target. The concept is to consecutively train the model distribution and the discriminator in turn, with the goal of reducing the difference between the model distribution and the target distribution measured by the best discriminator possible at each step of the training. GANs have been applied to many specific tasks, like text-to-image translation [2], image-to-image translation [3], and image enhancement [4]. Despite the great successes GANs have achieved, improving the quality of generated images is still a challenge.There are a lot of works have been proposed to improve the quality of images for GANs. Alecet al [4]. proposed deep convolu-

tional generative adversarial networks(DCGANs), that utilizes deep convolutional layers in the generator and learns a hierarchy of representations from object parts to scenes in both the generator and discriminator.Mao et al [6]. proposed the Least Squares Generative Adversarial Networks (LSGANs) which adopt the least squares loss function for the discriminator. LSGANs not only are able to generate higher quality images than regular GANs, but also LSGANs perform more stable during the learning process. The stability of learning process is another critical issue for GANs. Many works have been proposed to address this problem. Qi proposed the Loss-Sensitive GAN whose loss function is based on the assumption that real samples should have smaller losses than fake samples and proved that this loss function has non-vanishing gradient almost everywhere [5]. Takeru et al [11]. proposed a novel weight normalization technique called spectral normalization to stabilize the training of the discriminator. We experimentally confirmed that spectrally normalized GANs (SN-GANs) is capable of generating images of better or equal quality. Zhang et al.PROGAN proposed by Tero utilizes a deeper architecture and the model speeds the training up and greatly stabilizes it, and produce high resolution images. The diversity of image generation is the most challenging problem for GANs. It is very difficult for GANs to produce realistic diverse images such as natural images [11]. In terms of architecture-variant GANs, only SAGAN [12] and BigGAN [13] address such kind of issue. Benefiting from self-attention mechanism, CNNs in SAGAN and BigGAN can process large receptive field which overcomes the components shitting problems in generated images. This enables such type of GANs are able to produce diverse images.

Proposed Method

During the experiment, we found out that BigGAN has achieved satisfactory performance, so we decided to have a close look at it.

Increase the model

Simply increase the batch size can achieve a better performance and verify this. The BigGAN first increases the batch size from 256 to 2048 to cover more patterns, which not only provide more accurate gradients but also reduce the number of iterations of training. And BigGAN also increases the channels from 64 to 96 because increasing the number of channels has achieved great results on both ImageNet and JFT-300M.

Batch	Ch.	Param (M)	Shared	Skip-z	Ortho.	Itr $\times 10^3$	FID	IS
256	64	81.5	SA-GAN Baseline			1000	18.65	52.52
512	64	81.5	✗	✗	✗	1000	15.30	58.77(±1.18)
1024	64	81.5	✗	✗	✗	1000	14.88	63.03(±1.42)
2048	64	81.5	✗	✗	✗	732	12.39	76.85(±3.83)
2048	96	173.5	✗	✗	✗	295(±18)	9.54(±0.62)	92.98(±4.27)
2048	96	160.6	✓	✗	✗	185(±11)	9.18(±0.13)	94.94(±1.32)
2048	96	158.3	✓	✓	✗	152(±7)	8.73(±0.45)	98.76(±2.84)
2048	96	158.3	✓	✓	✓	165(±13)	8.51(±0.32)	99.31(±2.10)
2048	64	71.3	✓	✓	✓	371(±7)	10.48(±0.10)	86.90(±0.61)

Figure 1: Performance of BigGAN with different Batch size and channels

It can be seen that when the batch size is increased to 8 times, the IS in the generation performance has increased by 46%. The article speculates that this because covering more patterns in each batch provides a better gradient for generating and discriminating the two networks. Increasing the batch size will bring about training a better performance model in less time, but also reduce the stability of the model in training. In experiments, the number of channels in each layer of the article has also been increased. When the channel increases by 50%, it is about twice the number of parameters in the two models. This will result in a further 21% increase in IS. Because the increase in model capacity relative to the complexity of the data set. Interestingly, the article found in experiments that blindly increasing the network depth will not bring better results, but there will be a certain drop in generation performance.

The conditional label embedded under BatchNorm layer will bring a lot of parameter increases. BigGAN uses a shared embedding instead of setting each embedding separately for one layer. This embedding is linearly projected to the bias and weight of each layer, which reduces computational and memory costs and increases the training speed.

BigGAN has made improvements one the embedding of the prior distribution z. Common GANs directly embed z as input into the generation network, while BigGAN sends the noise vector z to multiple layers of generator instead of just the initial layer. The latent space z can directly affect the characteristics of different resolution and hierarchical levels. The conditional generation of BigGAN is achieved by dividing z into a block of each resolution and connecting each block to the condition vector c, so that the performance improves about 4% and the training speed increases by 18% [13]. As shown in the picture, the noise vector z is divided into multiple blocks by split and then connected with the condition label to send to each layer of the generation network. For each residual block of the generation network, it can be further expanded into the right picture structure. The block of the noise vector z and the condition label are sent to the BatchNorm layer after concatenate operation under the residual block.

Truncation technique

In addition, the truncation technique is also used, which is to cut off the sampling of z by sampling from the prior distribution and setting a threshold, where the values outsides the range are resampled to fall within the range. This threshold can be determined based on the IS and FID generated quality indicators. Through experiments, we can know that by setting the threshold, the quality of the generation will get better and better as the threshold decreases. However, due to the decrease of the threshold and the narrowing of the sampling range, the orientation of generation will become singular and the generation will be diverse the problem of lack of diversity. Often IS can reflect the quality of image generation, while FID will pay more attention to the diversity of generation.

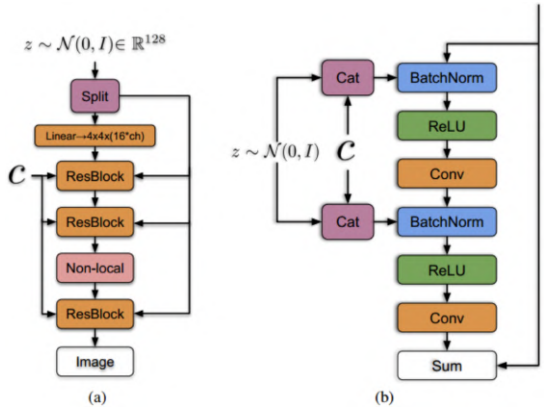

Figure 2: (a)A typical architectural layout for *G*; details are in the following tables. (b)A Residual Block in *G*. *c* is concatenated with a chunk of *z* and projected to the BatchNorm gains and biases.

Figure 3: The effects of increasing truncation. From left to right, threshold = 2, 1.5, 1, 0.5, 0.04.

Figure 4: Saturation artifacts from applying truncation to a poorly conditioned model.

As the cutoff threshold decreases, the quality of the generation is improving, but the generation is also approaching simplification. Therefore, according to the generation requirements of the experiment, we need to weigh quality and diversity. Often the decline of the threshold will cause IS to rise all the way, but FID will get better first and then get worse all the way. There are also some larger models that are not suitable for truncation, which will produce saturation artifacts when embedding truncation noise. As shown in Figure (b) above, in order to counteract this situation, the article enforces truncation by adjusting the generator to smooth Adaptability so that the entire space of z will be mapped to good output samples. To this end, the article adopts orthogonal regularization, which directly enforces the orthogonality condition.

$$R_\beta(W) = \beta||W^T W - I||_F^2$$

The original orthogonal regularization constraint is too strong, because it restricts the norm of the matrix column vector to 1. Therefore, a variant is used.

$$R_\beta(W) = \beta||W^T W \odot (1 - I)||_F^2$$

Training on large-scale data

In exploring the stability of the model, the article monitors a series of weights, gradients, and loss statistics during training to find indicators that may indicate the beginning of training collapse. The experiment found that the first three singular values of each weight matrix $\sigma 0$, $\sigma 1$, $\sigma 2$ are the most useful. In order to solve the training collapse on generator, the singular value $\sigma 0$ is adjusted appropriately to offset the effect of spectral explosion. First, the maximum singular value of each weight is directly regularized. Second, use partial singular value decomposition to truncate the largest eigenvalue.

$$W = W - max(0, \sigma_0 - \sigma_{clamp}) v_0 u_0^T$$

Like the generator G, the behavior is revealed by analyzing the spectrum of the weight matrix of the discriminator D, and then the training process is stabilized by adding some additional constraints. By observing the $\sigma 0$ of D, it is found that, unlike G, the spectrum of D has a lot of noise. The singular value will grow throughout the training process, and will jump when it crashes instead of exploding. The following penalty terms are used in the experiment to explicitly regularize the change of the Jacobian matrix of D.

$$R_1 := \frac{\gamma}{2} E_{p_{D(x)}}[||\bigtriangledown D(x)||_F^2]$$

Experiment

Our task is to generate pictures of dogs with about 20K different pictures of dogs given. The data we used is Stanford Dogs Dataset built from ImageNet. But, one of 20580 images doesn't include, so we have 20579, which consist of 120 dog breeds, from 148 to 252 photos per breed, with 75% quantile equal to 186 photos per breed.The interesting data facts we have found are:

- There are pictures with more than one dog (even with 3 dogs);
- There are pictures with the dog (-s) and person (people);
- There are pictures with more than one person (even with 4 people);
- There are pictures where dogs occupy less than 1/5 of the picture;
- There are pictures with text (magazine covers, from dog shows, memes and pictures with text);
- Even wild predators included, e.g. African wild dog or Dingo, but not wolves.

Let us visualize 9 random pictures of a given dataset.

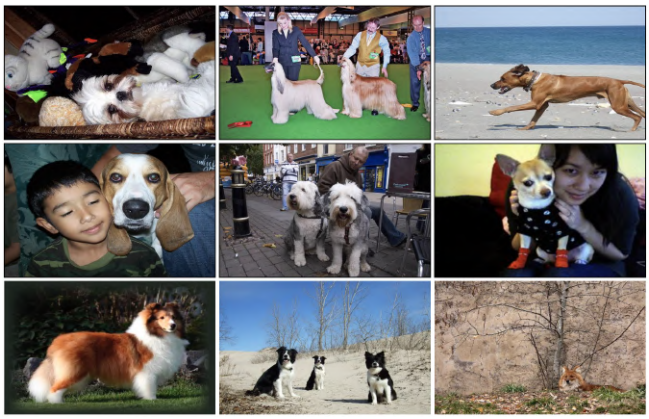

Figure 5: 9 random pictures of the given dataset

Evaluation

The models we used are evaluated on MiFID (Memorization-informed Fréchet Inception Distance), which is a modification from Fréchet Inception Distance (FID). The smaller MiFID is, the better your generated images are. FID and Inception Score (IS) are both commonly used in recent publications as the standard for evaluation methods of GANs.In FID, we use the Inception network to extract features from an intermediate layer. Then we model the data distribution for these features using a multivariate Gaussian distribution with mean μ and covariance Σ. The FID between the real images r and generated images g is computed as:

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

where T_r sums up all the diagonal elements. FID is calculated by computing the Fréchet distance between two Gaussians fitted to feature representations of the Inception network.

In addition to FID, we take training sample memorization into account. The memorization distance is defined as the minimum cosine distance of all training samples in the feature space, averaged across all user generated image samples. This distance is thresholded, and it's assigned to 1.0 if the distance exceeds a pre-defined epsilon.

In mathematical form:

$$d_{ij} = 1 - cos(f_{gi}, f_{rj}) = 1 - \frac{f_{gi} \cdot f_{rj}}{|f_{gi}||f_{rj}|}$$

where f_g and f_r represent the generated/real images in feature space (defined in pre-trained networks); and f_{gi} and f_{rj} represent the i^{th} and j^{th} vectors of f_g and f_r, respectively.

$$d = \frac{1}{N}\sum_j \min_j d_{ij}$$

defines the minimum distance of a certain generated image (i) across all real images ((j), then averaged across all the generated images.

$$d_{thr} = \begin{cases} d, & if\ d < \epsilon \\ 1, & otherwise \end{cases}$$

defines the threshold of the weight only applies when the (d) is below a certain empirically determined threshold. Finally, this memorization term is applied to the FID:

$$MiFID = FID \cdot \frac{1}{d_{thr}}$$

GAN experiments Firstly, we used vanilla GAN to train and generate the dog pictures. But the generative images was so ambiguous and nauseating that we barely could tell there is dog on the picture. The generative dog images shown in figure 6. In order to generate the adorable puppy, we tried six different variant GAN: CGAN, DCGAN, RalSGAN, WGAN, and BIGGAN. Alongside using MiFID to test these models, we also present the generated dog images from these models to show how good these models are. The dog images that generated by these models are shown in the following figures:

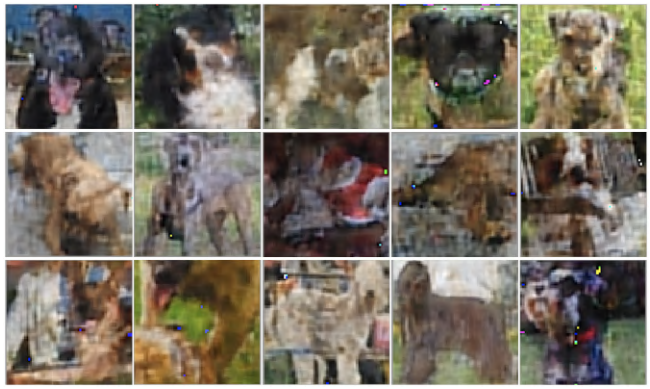

Figure 6: Generative dog images using vanilla GAN

Figure 7: Generative dog images using Conditional GAN

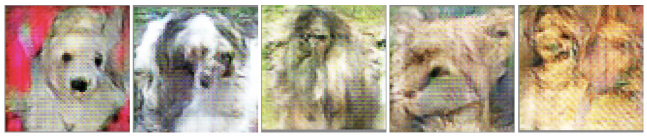

Figure 8: Generative dog images using Deep Convolutional GAN

Figure 9: Generative dog images using RaLSGAN

Figure 10: Generative dog images using WGAN

Figure 11: Generative dog images using BIGGAN

In the course of the experiment, we found that Conditional GAN can also generate pictures of endearing dogs, however, the resolution of the generated pictures is not enough, and the pictures generated by other network locks have very unsatisfied effects that cannot be called Dogs. We think this is related to the complexity of the data set used. The following table 1 shows how we use MiFID to evaluate the performance of each model. As you can see BigGAN can reach the 82.30464, which is the smallest MiFID among these models.

DATASET	METHOD	MiFID
Stanford Dogs Dataset	GAN	136.42730
	CGAN	113.78694
	DCGAN	187.42699
	RaLSGAN	143.65150
	WGAN	146.48429
	BigGAN	**82.30464**

Table 1: Use MiFID to evaluate the performance of each model

We also encounter mode collapse problem [14] when we tried to train our BigGAN longer. As you can notice that the dog images in figure12 are qualified when epoch = 70. However, the mode collapse problem raised when epoch = 120.

Figure 12: The generative dog images when epoch = 70.

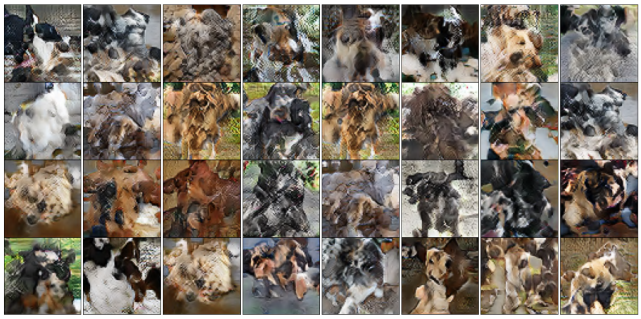

Figure 13: The mode collapse problem raised when epoch = 120.

Conclusion

In this paper, we used 6 kinds of gan networks to try to complete our task of "generating dog images", and got different experimental results. The biggan network is emphatically introduced because its performance in our experiments is particularly prominent, which is also consistent with the existing research results. However, the capabilities of the gan network are not limited to this. In the future, we will also try to use the gan network to complete more challenges and try to improve it.

References

[1] Goodfellow, I., et al. (2014). Generative adversarial nets. Advances in neural information processing systems.

[2] S Reed, Z Akata, X Yan, L Logeswaran arXiv preprint arXiv, and 2016. Generative adversarial text to image synthesis. jmlr.org, 2016.

[3] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Finegrained text to image generation with attentional generative adversarial networks. arXiv preprint, 2017.

[4] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 5967–5976. IEEE, 2017.

[5] Yifan Jiang, Xinyu Gong, Ding Liu, Yu Cheng, Chen Fang, Xiaohui Shen, Jianchao Yang, Pan Zhou, and Zhangyang Wang. Enlightengan: Deep light enhancement without paired supervision. arXiv preprint arXiv:1906.06972, 2019.

[6] Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv preprint arXiv:1511.06434 (2015).

[7] Mao, Xudong, et al. "Least squares generative adversarial networks." Proceedings of the IEEE international conference on computer vision. 2017.

[8] Qi, Guo-Jun. "Loss-sensitive generative adversarial networks on lipschitz densities." International Journal of Computer Vision 128.5 (2020): 1118-1140

[9] Miyato, Takeru, et al. "Spectral normalization for generative adversarial networks." arXiv preprint arXiv:1802.05957 (2018).

[10] Karras, Tero, et al. "Progressive Growing of GANs for Improved Quality, Stability, and Variation." International Conference on Learning Representations. 2018.

[11] Wang, Zhengwei, Qi She, and Tomas E. Ward. "Generative adversarial networks in computer vision: A survey and taxonomy." arXiv preprint arXiv:1906.01529 (2019).

[12] Zhang, Han, et al. "Self-attention generative adversarial networks." International conference on machine learning. PMLR, 2019.

[13] Brock, Andrew, Jeff Donahue, and Karen Simonyan. "Large scale gan training for high fidelity natural image synthesis." arXiv preprint arXiv:1809.11096 (2018).

[14] Salimans T, Goodfellow I, Zaremba W, et al. Improved techniques for training gans[J]. Advances in neural information processing systems, 2016, 29: 2234-2242.