A Review on GANS, Style Based Generator and Progressive Growth Methodology

Neha Joshi[1], Parag Chaudhari[2], Shubhankar Pawar[3], Pooja Shitole[4]

[1-4]Students, Dept. of Computer Science and Engineering, MIT School of Engineering, MIT ADT University, Pune, Maharashtra, India

---***---

Abstract - *Generative Adversarial Networks, or GANs, are an approach to generative modeling using deep learning methods and is one of the hot topics these days. It was first designed in 2014, and has been widely studied since then. GANs include automatically discovering and learning the similarity in the input data in a way that the model can be trained to generate the original dataset. In this paper we try to provide a review on Generative Adversarial Networks methods and it's typical applications in image processing and computer vision. We will talk about Progressive growth, a methodology for GANs, dealing with the progressive growth of the network and style-based generator architecture, in which the generated data is influenced in a way that it shows only certain specific traits of the image data.*

Keyword: Deep learning, Generative Adversarial Networks, Generative Modelling, Generator, discriminator.

1. INTRODUCTION

Generative Adversarial Networks has become a very popular topic recently. Yann LeCunn, VP and Chief AI scientist, Facebook, a legend in artificial intelligence, said, "GANS, and the variations that are now being proposed is the most interesting idea in the last 10 years in ML, in my opinion". There is a large number of papers being published related to GANs every year. According to google scholar, there are about 21,400 papers related to GANs published since 2020, that makes around 1900 papers published per month and around 64 papers everyday.

GANs are the smart way to train a generative model by putting the problem as a supervised learning problem with two sub-models, a generator and a discriminator, which are implemented using neural networks. A Generator is trained on true examples to generate a new example which are similar to the true ones. A discriminator is basically a binary classifier, which classifies the examples generated by the generator as real or fake. The optimization of GANs terminates at a saddle point that is a minimum for the generator and a maximum for the discriminator, i.e., the two sub modes compete with each other until they reach Nash equilibrium [1] where the generator starts creating images so realistic that it can fool the discriminator and humans as well.

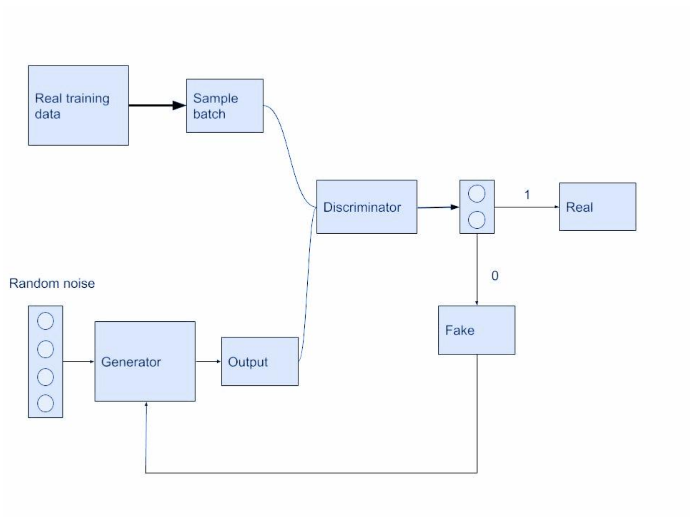


Fig. 1 Working of generators and discriminator of GANs

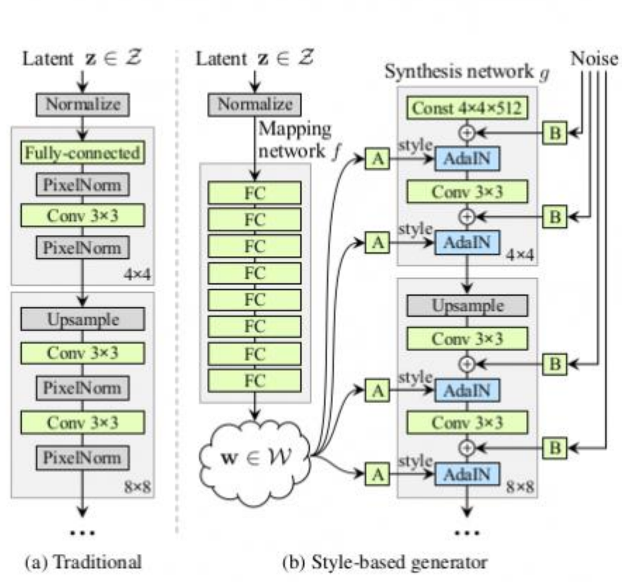
We will now discuss the methodology used and added to the generative adversarial networks to make it more useful and to increase its efficiency.

2. STYLE BASED GENERATOR

Since GAN's inception, plenty of improvements have been proposed which made it a state-of-the-art method to generate synthetic data including synthetic images, etc. However most of those improvements are made on the discriminator part of the model which doesn't improve the generation ability of the generator. This also implies that there wasn't much specialization in the generator part of the model which causes the shortage of control over the generator of GANs.

Motivated by style transfer technique [2], they redesigned the generator architecture in such a way that it opens many unique ways to control the process of image synthesis. Generator start from a learned constant input and adjusts the "style" of the image at each convolution layer based on the latent variables, therefore directly controlling the strength of image features at different scales. There are many parameters that could be changed by the generator like background, foreground and elegance or for human faces, features like hair colour, eyes colour etc, can be altered during the generation of images.

Style based generation architecture proposes to bring changes in the generator which can allow it to generate



the realistic high-quality images.

Fig. 2. In traditional GANs(a), the latent code is given to the networks through the input layer directly, in style based generator (b), the input is mapped to a latent space W which then connects to the network through adaptive instance normalization. The mapping network here has 8 layers and the synthesis network g is 18 layers deep with gaussian noise added after each convolution layer. The output from the last layer is converted to RGB using another 1 × 1 convolution. [3]

In traditional GANs, the latent code is provided to the generator by the input layer of the feed forward network (Fig 2a), while in a style based generator, we depart from the traditional way and the latent code is stored in a latent space (Fig 2b) which controls the generator through adaptive instance normalisation(AdaIN) after each convolutional layer of the network w(Fig 2b) is specialised to style y = (y_s, y_b) by learned affine transformation which then controls AdaIn.

The AdaIN network is defined by[3]

$$\mathrm{AdaIN}(\mathbf{x}_i, \mathbf{y}) = \mathbf{y}_{s,i}\frac{\mathbf{x}_i - \mu(\mathbf{x}_i)}{\sigma(\mathbf{x}_i)} + \mathbf{y}_{b,i}, \tag{1}$$

where x_i is the feature maps which are normalized separately and then scaled and biased using the scalar components from y.

Finally, a generator with the direct intentions to generate stochastic details by adding noises, is provided. The Gaussian noises are fed to each layer of the synthesis network. The noise image is sent to each feature maps and then added to the output of the corresponding convolution(Fig 2b)

2.1. Properties of the style-based generator

The quality of images generated are better than those in traditional GANs. The architecture can control the image synthesis by modifying the style. The mapping network and affine transformations can draw samples for each style from a learned distribution, and the synthesis network can generate a new image based on a collection of styles. The effects of each style are localized in the network, i.e., aspects of each image formed can be affected by modifying the specific subset of the style.

1. Style Mixing: Mixing regularization is introduced to encourage the style to localize, where two random latent codes are used instead of one during training, to generate a given percentage of images. When generating such an image, we simply switch from one latent code to another at a randomly selected point in the synthesis network, this process is referred to as style mixing. Mixing regularization improves the localization when enabled during the training.

2. Stochastic variation: Stochastic means having a pattern that cannot be determined precisely, even human portraits have many stochastic features and they are represented randomly as long as they don't affect our perception. The style based generator applies stochastic variation to the various layers by addition of random noises after each convolution in the style based generator(fig 2b) which makes the image more realistic and precise. There is a fresh set of noises for each layer in the synthesis network.

3. Separation of global effects from stochasticity: In the style-based generator, the entire image is affected by the style, not just the stochastic features, because complete feature maps are scaled and biased with the same values. Therefore, global effects such as lighting or background style can also be controlled reasonably. Also, as the noise is added independently to each pixel and it is ideally suited for controlling stochastic variation.

Based on all the research done in the field of GANs, it is noticed that with better and more realistic image quality, the style based generators are better than the traditional generative approach.

3. PROGRESSIVE GROWING OF GANs

The main idea of progressive growth is a training methodology for GANs, to grow both the generator and discriminator progressively, that is training them together, starting from a low resolution, and then adding new layers to the network which speeds up the training and also stabilizes it, allowing us to produce more realistic images.

It takes really small images and increasingly adds blocks of layers that increase the output size of the generator model and the input size of the discriminator model until the desired image size is achieved. This approach has proven effective at generating high-quality fake faces that look very realistic. Progressive growing GAN is a firm

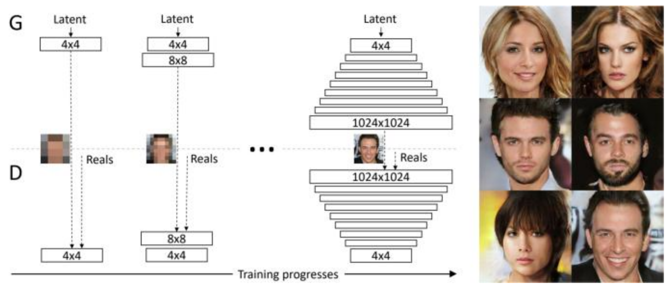


Fig. 3 Training starts with the generator(G) and discriminator(D), which have 4 × 4 pixels. The training is advanced by adding more layers to G and D. On the right we can the images generated using progressive growing at 1024 × 1024. [5]

approach for the training of GANs model to generate large high-quality images that involves increasing the size of the model during training. Traditional GANs can capture only a subset of the variation found in training data, so "minibatch discrimination" [5] was suggested as a solution. So feature statistics were not only calculated from the images but for the mini batches too. By this the mini batch and training images both showed the same statistics. Progressive GANs neither use learnable parameters nor new hyperparameters. It first computes the standard deviation for each feature in each spatial location over the minibatch and then averages these estimates over all features and spatial locations to arrive at a single value. They copy the value and add it to all the spatial locations and over the minibatch as well, creating one additional feature map. This layer could be inserted anywhere in the discriminator, but it is best to insert it towards the end[5].

Another way to deal with the variation problem is by unrolling the discriminator, this is done to regularise it's update and a "repelling regularizer" is used that adds up a new loss term to the generator. All this solutions can increase the variation even more.

CONCLUSION

This paper presents a brief review on GANs and it's algorithms. We discussed how traditional generative adversarial networks are improvised by introducing style based generators and progressive growth methodology and giving our readers a gist about GANs.

REFERENCES

[1] L. J. Ratliff, S. A. Burden, and S. S. Sastry, 'Characterization and computation of local nash equilibria in continuous games', in Annual Allerton Conference on Communication, Control, and Computing, pp. 917–924, 2013.

[2] X. Huang and S. J. Belongie. 'Arbitrary style transfer in real-time with adaptive instance normalization'. CoRR, abs/1703.06868, 2017.

[3] Tero Karras, Samuli Laine, Timo Aila, 'A Style-Based Generator Architecture for Generative Adversarial Networks', arXiv:1812.04948, 2019

[4]J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A review on generative adversarial networks: Algorithms, theory, and applications," arXiv preprint arXiv:2001.06937, 2020.

[5]Tero Karras, Timo Aila, Samuli Laine, Jaakko Lehtinen, "Progressive Growing of GANs for Improved Quality, Stability, and Variation". arXiv:1710.10196, 2017