FAM-GANs

The Next Evolution of the Creative Adversarial Network

By Chris Garces, Yogitha Mariyappa, Durand Sinclair & Sereen Yaseen

Abstract **2**

The FAM-GAN Framework **2**

The Problem with Generative Adversarial Networks 2

Creative Adversarial Networks 2

The Problem with Creative Adversarial Networks 3

The Solution: FAM-GANs 4

Technical Challenges **4**

Convergence to a new style of sufficient quality 4

Converging Around One Picture 6

Commercial Applications **6**

Copyright Issues 7

Reference List **8**

Appendices **9**

Appendix 1: Summary of Effort 9

Appendix 2: Other Applications Explored 9

Appendix 3: Summary of 5 GAN Papers 9

Abstract

We propose a new type of Generative Adversarial Network framework called a FAM-GAN. It is designed to generate a family of artworks of a different style than its dataset, yet similar enough to itself to be recognisable as a style. The original dataset consists of art with labelled styles. The FAM-GAN attempts to do three things: First, the Discriminator should be fooled into thinking that generated art comes from the original dataset, like a standard GAN. Second, the Discriminator should be confused as to which style-label to put on the art (indicating it's invented a new style), like a Creative GAN. Third, the new art should be similar to itself, which is what makes FAM-GANs different to the Creative GAN. As human beings prefer the novelty of new art styles, a FAM-GAN has multiple commercial applications, including in paintings, computer games, and even the invention of new fantasy characters for the movie industry.

The FAM-GAN Framework

The Problem with Generative Adversarial Networks

Up until now, the best technology for inventing new art with a computer has been the Generative Adversarial Network (GAN), where a Generator attempts to produce pictures that a Discriminator cannot distinguish from a dataset of real pictures. The strength of a GAN is that it produces pictures so realistic that humans cannot distinguish between real ones and fake ones. But its weakness is that the creativity is convergent. That is, if we have a dataset of dogs, then the Generator could only ever produce dogs. If, for example, the Generator produced a picture of a cat, the Discriminator would reject it as a fake. This limits the ability of GANs to be truly creative. It is the musical equivalent of inventing a robot cover-band, rather than a composer.

Creative Adversarial Networks

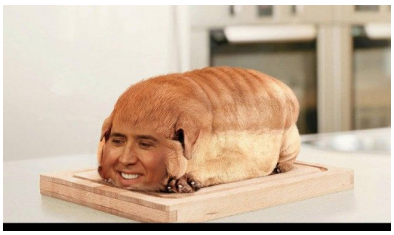
Creative Adversarial Networks (CANs) [Elgammal et al, 2017] got around this problem by feeding in a labelled dataset to the GAN and posing two optimisation challenges. First, fool the Discriminator into thinking the generated image was from the original dataset. Second, confuse the Discriminator so that it doesn't know which style label to put on the art. This mixture of convergence and divergence produces a new style. For example, if you fed in a dataset with pictures of dogs and muffins, the Generator would produce something new: dog-muffins.

Dog-Muffins

The Problem with Creative Adversarial Networks

The problem with Creative Adversarial Networks is the heterogeneity of the new style.

If you are only mixing two styles together, then there wouldn't be a problem. There is only one new style you can create out of dogs and muffins, and that's dog-muffins. However, what happens if you have three styles? For example, if you had a dataset containing pictures of dogs, muffins, and Nicholas Cage, your new "style" would have wildly different types of pictures: dog-muffins, dog-Nicholas-Cages, and Nicholas-Cage-muffins.

		
Dog-Muffins	Dog-Nicholas-Cages	Nicholas-Cage-Muffins

While creative, these images are too different from each other to be considered a single style. To create artworks that have their own oeuvre, we need the artworks to be similar to each other, (in the same way that, say, Police Academy V is similar to Police Academy VII and Police Academy IV), yet different from the original style, (say, actually funny comedy).

			
Police Academy V	Police Academy VII	Police Academy IV	Actually funny comedy

But how do you get the system to create art that's similar to itself, even while it's different from the original styles in the dataset?

The Solution: FAM-GANs

The solution is to add a third challenge to the Discriminator - ask it if subsequently generated samples are of the same style as the first piece that fooled the Discriminator into thinking it was art. But we should only set this challenge once the Generator has already learned to create sufficiently good artwork of a new style.

Technical Challenges

There are two technical challenges to overcome to make this work.

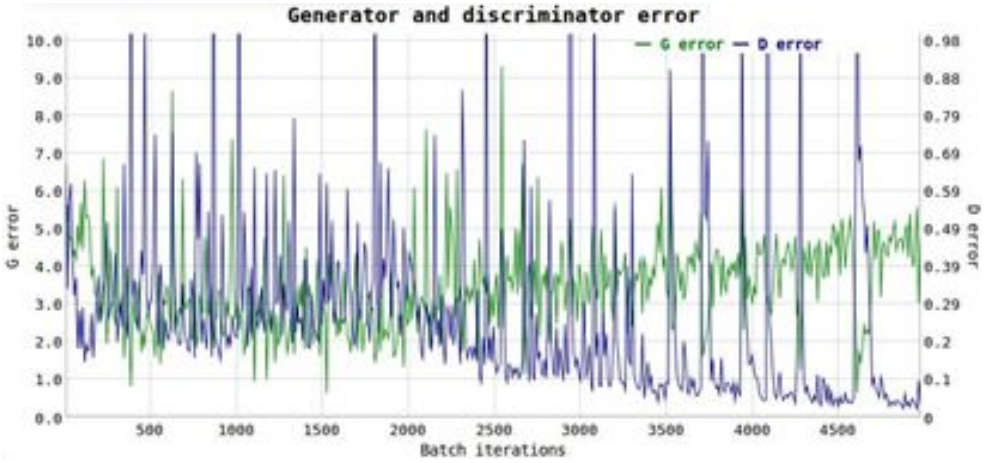
- **Convergence to a new style of sufficient quality** - We propose using Wasserstein distance (also known as Earth Mover Distance), to measure both the quality of our new art and the divergence from the old style. When Wasserstein's distance is small enough, the third signal can be sent and we can start optimising around it.
- **Converging around one picture** - Once we have generated our first picture of sufficient quality, we can start to converge around it. But this poses a different challenge, - you don't want all the new art to look exactly the same as each other. GANs usually have a whole dataset to converge around, allowing more variation. We propose that this problem can be solved with a supervised approach (conditional GANs) or an unsupervised approach (setting a boundary within the image manifold).

Convergence to a new style of sufficient quality

From a mathematical perspective, a GAN improves the quality of its generations by converting one statistical distribution to another. There are several ways to do this – in fact, Gibbs & Su [2002] outlined ten different methods. But in the last three years, a couple have come to the fore:

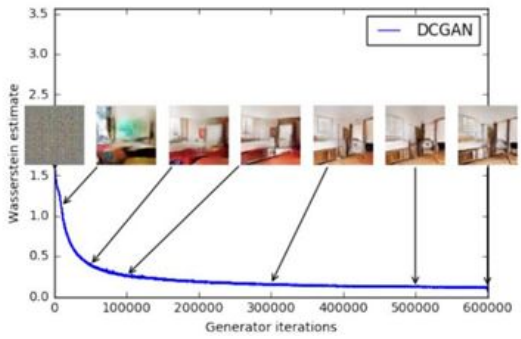
- **Information difference** – We could use a measure of how much new *information* is found in one probability distribution compared to the other, and minimise that difference.

You calculate the amount of information in a distribution by finding out how many ways you can rearrange the data and still see the same pattern [Seranno, 2017]. For instance, a completely black picture has little information, because you can rearrange the pixels in any way you want and it would look the same, whereas a photo of a face has more information because rearranging it would lead to a different picture. Goodfellow [2014] used Jensen-Shannon divergence, which compares how much new information there is in one distribution compared to the other. However, this technique has two problems. Firstly, mode collapse, where the samples generated look too similar to each other. And secondly, information gain metrics have a problem with symmetry, which means that if you try to tune both the Generator and Discriminator at the same time, they start to diverge. This in turn means that there is a low correlation between how much you tune your parameters and the quality of your images. Eventually they drift away from each other, as shown in the picture below.



- **Wasserstein's distance** – Rather than minimise the information difference, the Wasserstein distance (aka earthmover's distance) attempts to minimise all differences between two distributions, regardless of whether they contain information or not. This is the technique proposed in Arjovsky et al's [2017] paper on Wasserstein GANs. It is a more reliable cost function as it does not have the same problems with symmetry, which means the picture quality improves as your loss gets smaller. However, it is still prone to mode collapse.
- **Improved Wasserstein's Distance** – This approach, outlined by Gulrajani et al [2017], solves the problem of mode collapse by not allowing the Discriminator to get too good. The better the Discriminator gets, the less variation it will allow in generated samples, so to avoid mode collapse you need to limit the Discriminator's effectiveness. Arjovsky's paper attempted to do this by weight clipping, but Gulrajani got better results by penalising the norm of the gradient instead.

In summary, by using a better optimisation method (Wasserstein's distance) and penalising the norm of the gradient rather than using weight clipping, improved Wasserstein distance gives a metric that we can use to measure the quality of the image as the GAN improves it.



This means we can tell our system to keep optimising until the generated image reaches a certain level of quality, before declaring we have a new style. When we have our first picture of the new style, we can converge around that.

Converging Around One Picture

There are a two techniques we can use to make the FAM GAN converge around a single style.

- **Supervised Learning with Conditional GANs** - In this approach, based on Mirza [2014], each picture in the original dataset would have multiple descriptive labels attached, and before running the GAN, we would choose which labels we'd want to appear in our new style. For example, if we are creating cartoon characters, and wanted the GAN to generate ones with blue eyes, we would choose to optimise towards this label. The advantage of this approach is that it gives the user creative input into what is generated. The disadvantages are that it involves a lot of hand-labelling of the dataset, and the results will depend on the taxonomy of labels used.
- **Unsupervised Learning with iGAN** - In this approach, the computer would randomly choose a new style and create multiple artworks of that style. It would do this by drawing a (multi-dimensional) circle around the first generated artwork that the discriminator considered art but could not label, and would make sure the style of subsequent generated pictures fell within that range. We could use Wasserstein's distance to define the boundaries of this range. This approach is similar to Zhu et al's [2016] paper on image manipulation, in which drawing shapes manipulates the boundaries of the image manifold, allowing the shape to be filled with an image of a particular style.

Commercial Applications

There are several commercial applications that can be produced using a FAM-GAN.

- **Paintings** - If you start with a dataset of paintings (by scraping Google images for example), the FAM-GAN can create a new style. If you then hooked up your computer to a CNC machine like Spinelli did [2009], your computer could paint a batch of paintings of the new style onto actual canvases. If you displayed those paintings at an art gallery, and challenged the public to see if they could tell the difference between human art and AI art, it could generate great publicity. Brophy [2010] finds that a typical price for a

single painting is $2,200, or $6 per square inch. However, paintings in a unique style created by AI, could arguably fetch much more.

- **Computer game assets** - These are the background elements in a computer game world that make it feel more realistic, like trees, benches, barrels etc. Computer game assets are sold on game asset websites, like gamedevmarket.net. Each individual piece costs $2 - $10 [GameDevMarket.net, 2017], and the customers tend to be game developers who want to save time. Bouanani [2015] argues that the key to selling into this market is create assets that look different from the ones already available, which is exactly what a FAM GAN is designed to do. With an unsupervised FAM-GAN, you could flood the market with slightly different objects for game developers. However, it may be difficult to find a quality dataset to generate from, firstly because they are less available online than pictures, and secondly because we will need multiple versions of each object to generate new ones. Eg, we need multiple existing barrels to generate barrels with a new style - we can't put barrels in with other objects like trees. One solution is to purchase several asset collections which sell at a discount, and then use them to generate our own variations, which we can sell online one at a time.
- **Fantasy character generation** - This is potentially the most lucrative idea, as you can build a whole franchise around fantasy characters, like hobbits from Lord of the Rings or the Na'vi from Avatar. As an example of the monetary value, the Pokemon franchise had income of $1.5 billion a year in 2013 [Berman, 2014]. There is plenty of fantasy art on the internet which we can use in our dataset, including on Google Images and Pinterest. We could also use a high quality database of faces, like the Face Recognition dataset [face-rec.org, 2017]. By using FAM-GAN's tendency to drift from the original styles, we should end up with new and unusual creatures, which is perfect when trying to generate characters for a fantasy universe.

Copyright Issues

The Berne Convention is an international treaty that protects copyright, both of the original work, and works derived from it. Article 2 section 3 defines a derivation to include an adaptation. If our GAN starts with a dataset of copyrighted material, it could be argued that the final result is an adaptation, and using a FAM GAN violates the copyright-holder's rights.

There are two arguments against this. Firstly, the FAM GAN dataset contains multiple styles, each comprising multiple works of art. This would make the connection between a particular work of art in the dataset and the final product to be incidental. And secondly, a FAM-GAN is designed to produce a work unlike that in the existing corpus, and to establish its own style.

However, imagine if the original dataset only contained one picture. The output would surely be a derived work that violated copyright. On the other hand, if the original dataset contained thousands of pictures, then it wouldn't. So there is a cutoff point somewhere, below which one cannot use copyrighted material in a FAM-GAN. It is hard to determine where that cutoff point is.

Reference List

Arjovsky, M., Chintala, S., & Bottou, L., 2017, “Wasserstein GAN”, arXiv:1701.07875v2 [stat.ML] 9 Mar 2017, accessed 15/11/2017

Berman, N., 2016, “How much is the entire Pokemon franchise worth?”, *Money Inc*, http://moneyinc.com/pokemon-franchise/, accessed 10/11/2017

Bouanani, O, 2015, “How to fund your games by creating and selling game assets”, *EnvatoTuts*, https://gamedevelopment.tutsplus.com/articles/how-to-fund-your-games-by-creating-and-selling-game-assets--cms-24380, accessed 17/11/2017

Brophy, M., 2010, “How to price your original artworks”, http://mariabrophy.com/business-of-art/how-to-price-your-original-artworks.html, accessed 17/11/2017

Elgammal, A., Liu, B., Elhoseiny, M., Mazzone, M., 2017, “CAN: Creative Adversarial Networks, Generating ‘art’ by learning about styles and deviating from style norms”, arXiv:1706.07068, [cs.AI] 23 Jun 2017, accessed 10/11/2017

GameDevMarket.net, n.d., https://www.gamedevmarket.net/ accessed 17/11/2017

Gibbs, A.L., & Su, F.E., 2002, “On Choosing and Bounding Probability Metrics”, arXiv:math/0209021v1 [math.PR] 3 Sep 2002, accessed 15/11/2017

Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y., 2014, “Generative Adversarial Networks”, arXiv:1406.2661v1 [stat.ML] 10 Jun 2014, accessed 15/11/2017

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A., 2017, “Improved training of Wasserstein GANs”, arXiv:1704.00028v2 [cs.LG] 29 May 2017, accessed 15/11/2017

Mirza, M., & Osindero, S, 2014, “Conditional Generative Adversarial Nets”, arXiv:1411.1774v1 [cs.LG] 6 Nov 2014, accessed 15/11/2017

Serrano, L., 2017, “Shannon Entropy, Information Gain, and Picking Balls from Buckets”, *Medium,* https://medium.com/udacity/shannon-entropy-information-gain-and-picking-balls-from-buckets-5810d35d54b4 accessed 15/11/2017

Spinelli, A., 2009, “my first cnc painting machine”, https://www.youtube.com/watch?v=J0YDP4QsQus&t=22s accessed 15/11/2017

Statista.com, 2017, “Value of the global video games market 2011 to 2020”, https://www.statista.com/statistics/246888/value-of-the-global-video-game-market/, accessed 10/11/2017

Zhu, J.Y., Krahenbul, P., Schechtman, E., & Efros, A.A., 2016, “Generative Visual Manipulation on the Natural Image Manifold”, arXiv:1609.03552v2 [cs.CV], 25 Sep 2016, accessed 15/11/2017

Appendices

Appendix 1: Summary of Effort

Here is a summary of the work we've done to produce this paper

- Attempted to invent a new type of GAN.
- Explored the maths behind how GANs converge styles, both using entropy/information metrics like Jensen Shannon, and more useful metrics like earthmover's distance.
- Explored the maths behind how an image is transformed into multiple normal distribution of colour or brightness intensity.

Appendix 2: Other Applications Explored

1. Plant health detection using phone app
2. Autonomous Garden with NDVI plant health monitoring and pest detection
3. Self-navigation robot with short voice command
4. Shoe design generation
5. Cartoon character generator
6. Age progression
7. Unsupervised image colorization
8. Landmark recognition

Appendix 3: Summary of 5 GAN Papers

Generative Adversarial Networks

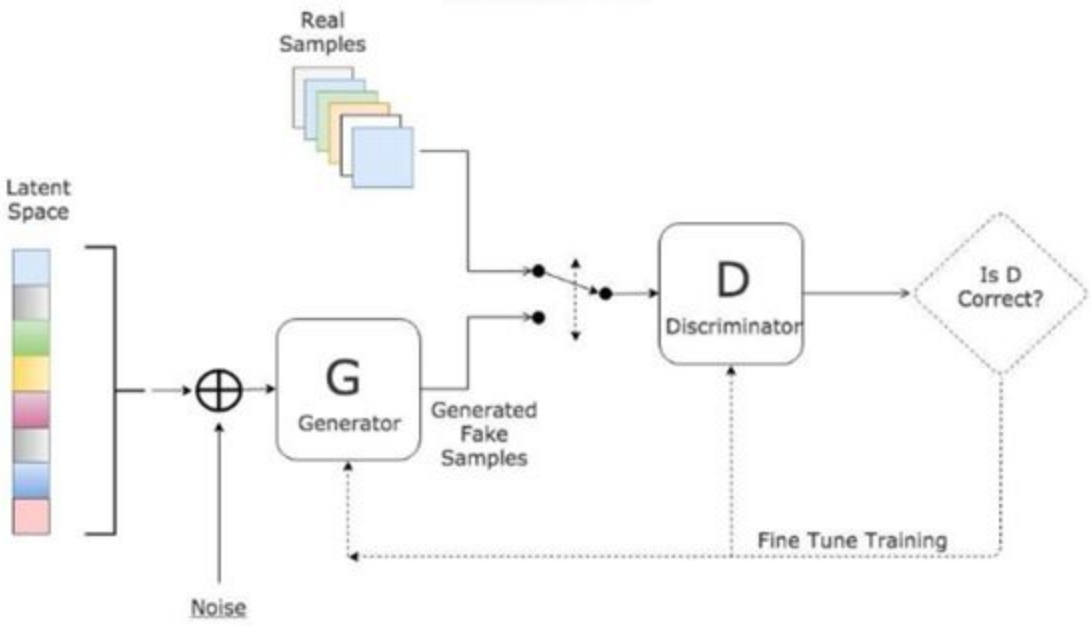
Generative Adversarial networks are an architecture for neural networks pioneered by Ian Goodfellow et al. in 2014. The main idea behind GAN is to have two independent networks, a Generator and Discriminator), contesting with each other in a min-max.

A Generator(G) is tasked to generate fake images, from random noise, carefully crafted to look almost real. A discriminator(D) is trained on legitimate images from training data as well as the fake images from G. The task of the discriminator is to classify the real images from the fake, by producing a probability of each image being real.

The training for the network follows a two-step process in which,

- The discriminator is optimized in order to increase the likelihood of giving a high probability to the real data and a low probability to the fake data.
- The generator is then optimized in order to increase the probability of the fake data being rated highly, through backpropagation.

The GAN slowly converges to produce data that is as real as the network is capable of modeling.

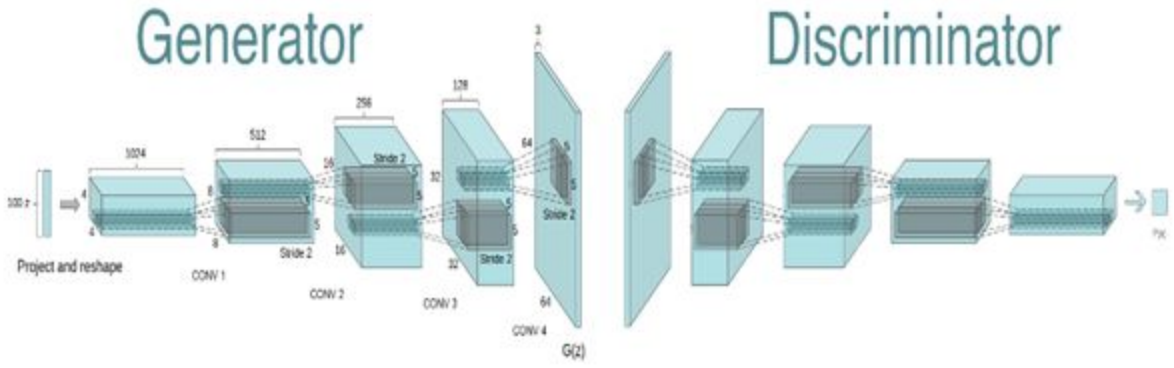


The biggest challenge that Vanilla GANs have is that training is quite unstable, often resulting in a generators that produce nonsensical outputs

Deep Convolutional Generative Adversarial Networks

GANs are known to be unstable to train, often resulting in Generators that produce nonsensical outputs. In January 2016, Alec radford, Luke Metz and Soumith Chintala, introduced *Deep Convolutional Generative Adversarial Networks* in their paper (Radford, Metz and Chintala, 2016). The authors describe the combination of some deep learning techniques as key for stabilizing training in GANs. Some of these techniques include,

- **Avoid fully hidden connected layers and pooling layers** – Emphasis was given to the usage of strided convolutions for both Generator and Discriminator, increasing and decreasing the spatial dimensions of the feature map instead of using classical pooling functions. Image below adapted from the DCGAN paper. Note the non-existence of fully connected and pooling layers.



- **Batch Normalization** - Normalizing the input of each, layer to have zero mean and unit variance in all layers, was found stabilize learning.
- **Activation functions** – In order to speed up training, Rectified Linear Unit(ReLU) and Leaky Rectified Linear Unit (Leaky ReLU) were used on layers of Generator and Discriminator respectively.

Improved DCGANs

New architectural features and training procedures on DCGANs were introduced in March 2017(Salimans et al., 2017). When used for semi-supervised classification and these techniques were found to generate images that are of better high-resolution.

- **Feature matching**: Typically, a Generator is trained to maximize the output probability of the Discriminator. The authors propose a new objective function that requires the Generator to generate data matching the statistics of the real data. So the job of the Discriminator is only to specify which are the statistics worth matching, by training it to find those features that are most discriminative of real data versus data generated by the current model.
- **Minibatch Discrimination** : Allows Discriminator to look at multiple samples in combination rather than single examples, so as to know if it is getting the same (or similar) images more easily. This can be achieved by giving the Discriminator features that tell it about the distance of each image to other images in the same batch.
- **Historical averaging**: When updating the parameters, also take into account their past values. They add a penalty term that punishes weights which are rather far away from their historical average values.
- **One-sided label smoothing**: Simply make your Discriminator target output from [0=fake image, 1=real image] to [0=fake image, 0.9=real image]. This helps model training.
- **Virtual batch normalization**: Avoid dependency of data on the same batch by using statistics collected on a reference batch. It is computationally expensive, and hence only used on the Generator.

The result,when trained on images from imagenet is as shown below. We can notice that Improved DCGANs generate better quality images in comparison to DCGANs.

Conditional GAN

The paper on cGAN was published by Mehdi Mirza and Simon Osindero in November 2014. Figure A.1 shows the framework of cGAN. The conditional information that describes some characteristics of the dataset is feed into both the Generator and the Discriminator. The conditional dataset needs to be labeled. cGAN will learn to exploit this samples and generate new images with variations of the conditional information.

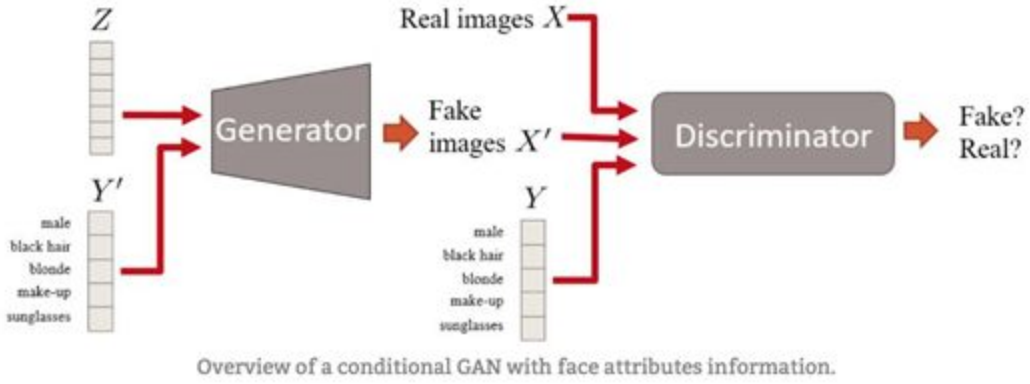


Figure A.1 cGAN framework.

Wasserstein GAN

GANs are hard to train because they always have problems with convergence. Its loss function does not correlate with the quality of the generated images. Consider the chart below showing the training performance of the DCGAN, as batch iteration increases, the Discriminator error improves but there are instances of spikes which suggest instability. It is also observed that the Generator error increased which construe that it didn't learn from the training iteration. But this GAN correctly generates images of handwritten digits using MNIST dataset. This interpretability and instability of loss function are being addressed by Wasserstein GANs.

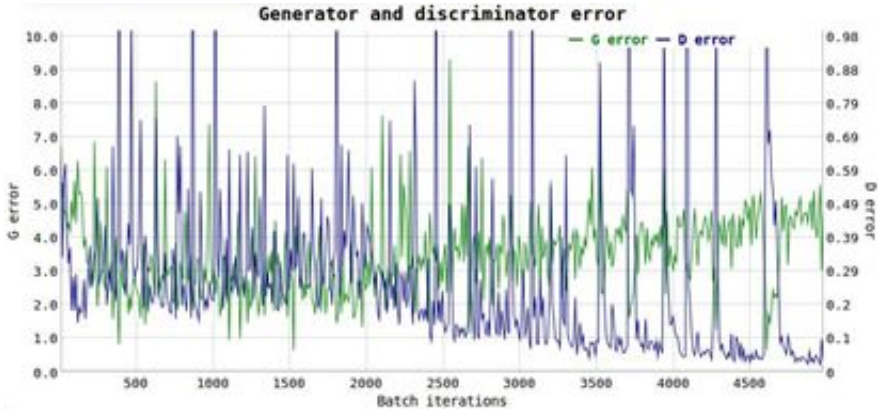


Figure A.2. DCGAN training error performance.

The paper on Wasserstein GAN was published by Martin Arjovsky, Soumith Chintala, and L´eon Bottou in January 2017. It replaced the loss function of the Generator and Discriminator with Wasserstein Distance or Earth Mover Distance (EMD). EMD is a method to measure the dissimilarities between two multi-dimensional distribution. It informally visualised the two distributions as a pile of dirt and calculate the minimum total work to transform one pile into the other pile, hence term Earth Mover Distance. The figure below shows the result of replacing a DCGAN's loss function with Wasserstein Distance. Here, we can see a much stable Wasserstein estimate without the spikes of the previous chart, and image quality improves as the estimate decreases.

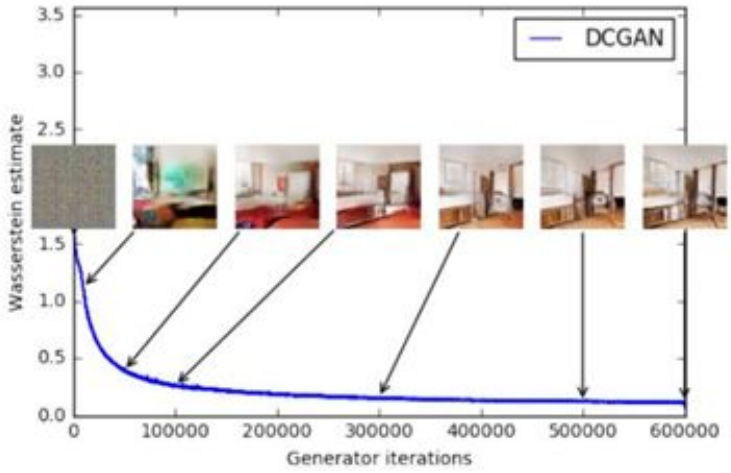


Figure A.3. Wasserstein estimate's correlation with image quality