re]

Jan 28 · 8 min read

How do GANs intuitively work?

GANs or Generative Adversarial Networks are a kind of neural networks that is composed of 2 separate deep neural networks competing each other: the **generator** and the **discriminator**.

Their goal is to generate data points that are magically similar to some of the data points in the training set.

GAN is a really powerful idea. Even Yann LeCun (one of the fathers of Deep Learning) is saying that it's the coolest idea of Machine Learning in the last 20 years.

Currently, people use GAN to generate various things. It can generate realistic images, 3D-models, videos, and a lot more.

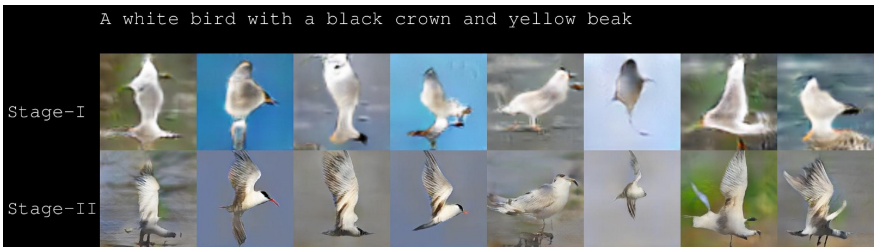
Why do we want the machine to generate data?

It's a essential for the machine to be intelligent. If it can generate, it would understand.

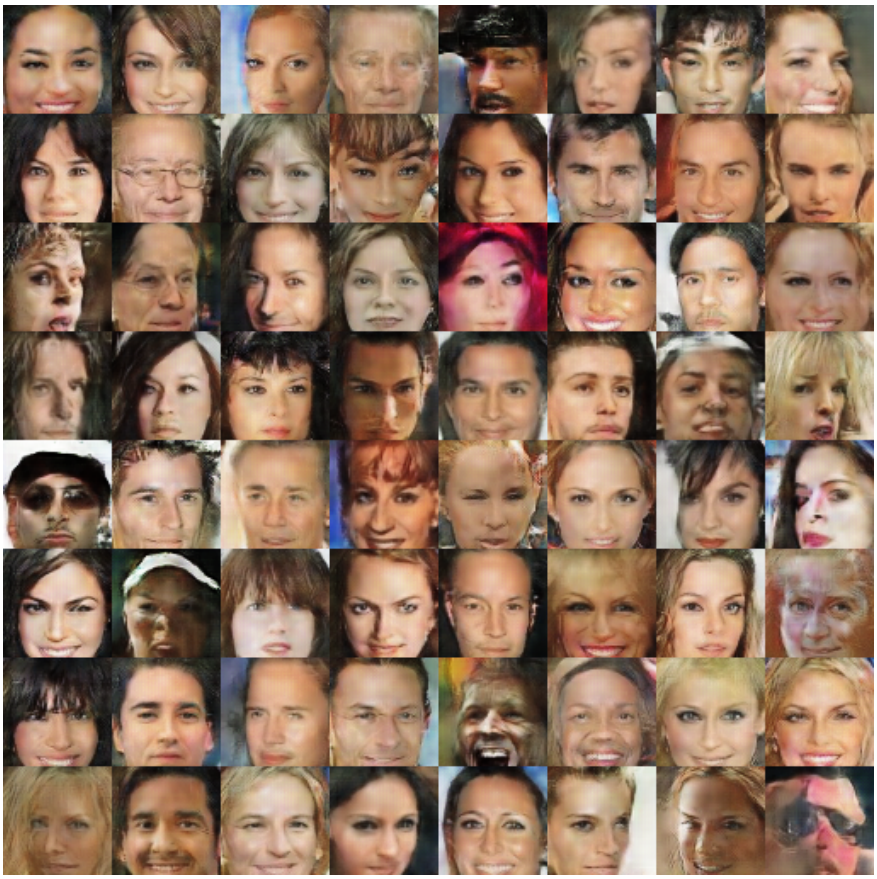
> *"What I cannot create, I do not understand."—Richard Feynman*

Example of things it can generate

Generating digit images similar to MNIST dataset that even HUMANS cannot distinguish from the real images.



Generating bird images from text caption: StackGAN



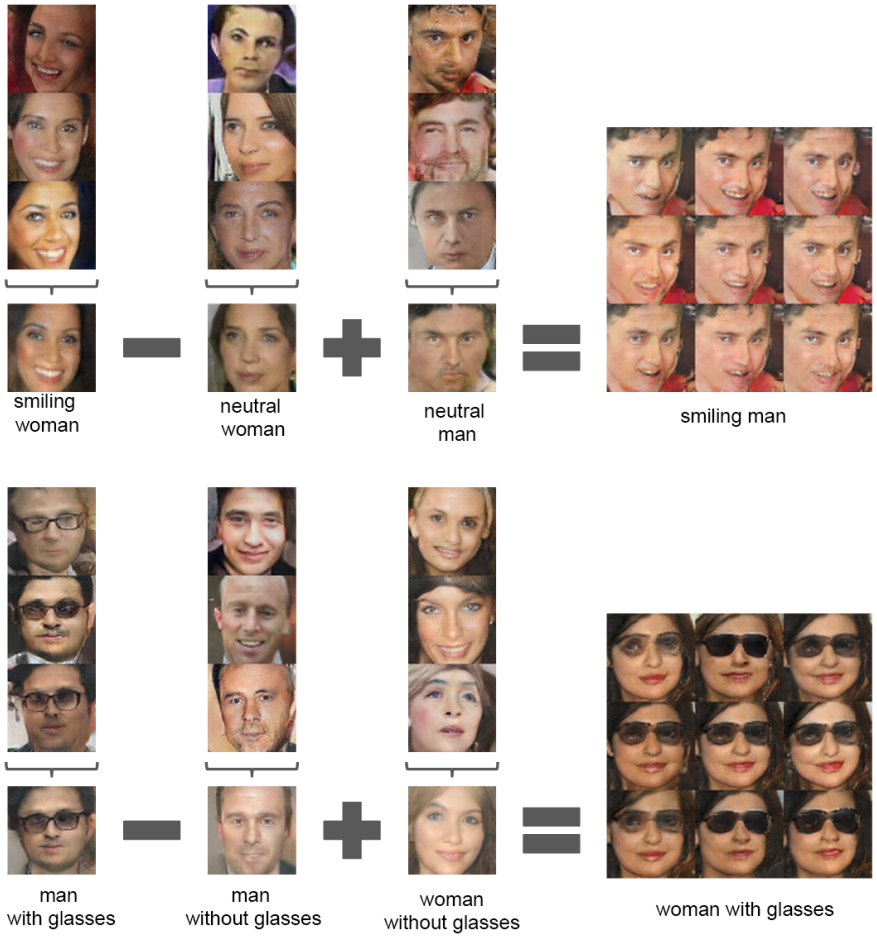
Generating faces using DCGAN

Generating Anime characters using DCGAN

Predicting future frames from a still image

Generating 3D-models

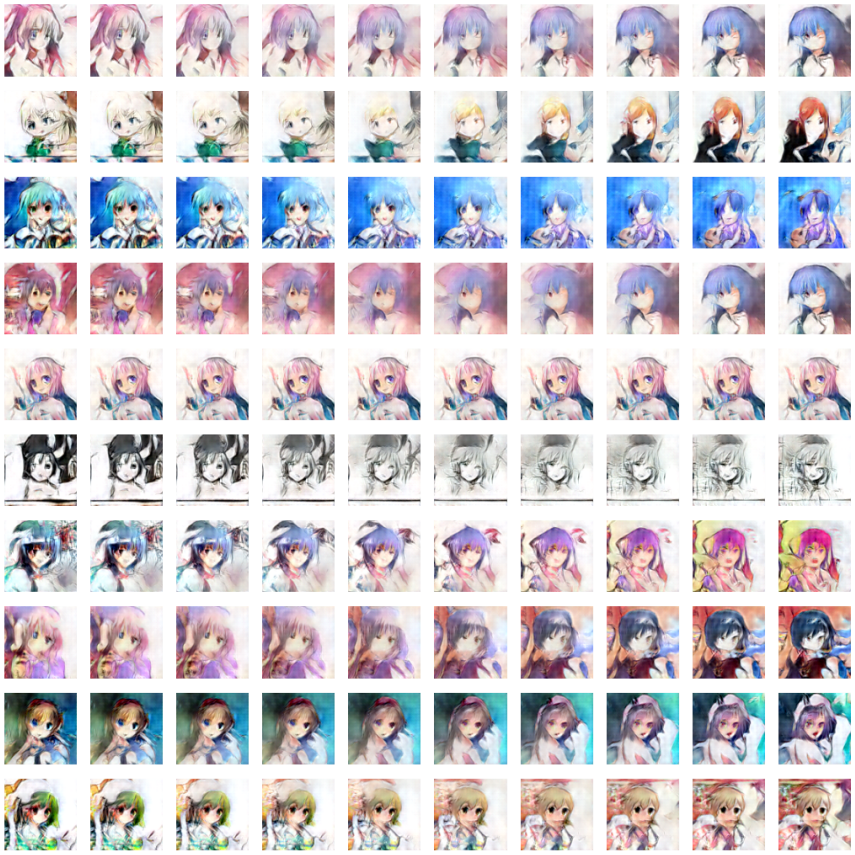
Beside from generating things, you can do arithmetic on abstract ideas like removing glasses from a face!

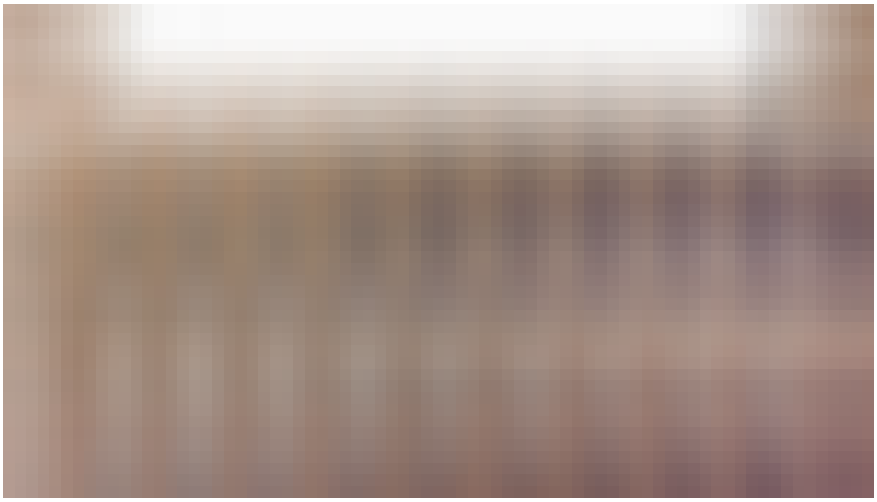


Arithmetic on faces: DCGAN_code

What about interpolations?

Given 2 images, let it generate images that are in between.





There are a lot more stuff you can do.

Impressive, isn't it?

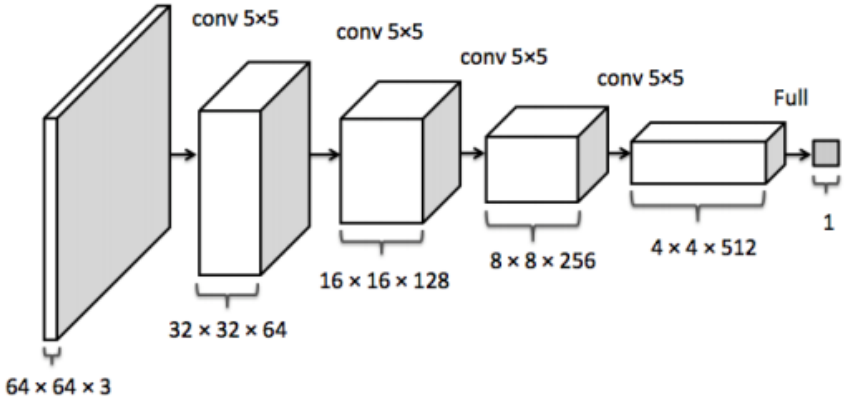
Idea of GAN

To give an example, let's assume that we want our GAN to generate images that look like people faces in the training set like CelebA dataset.

Our generator's architecture might look something like this:

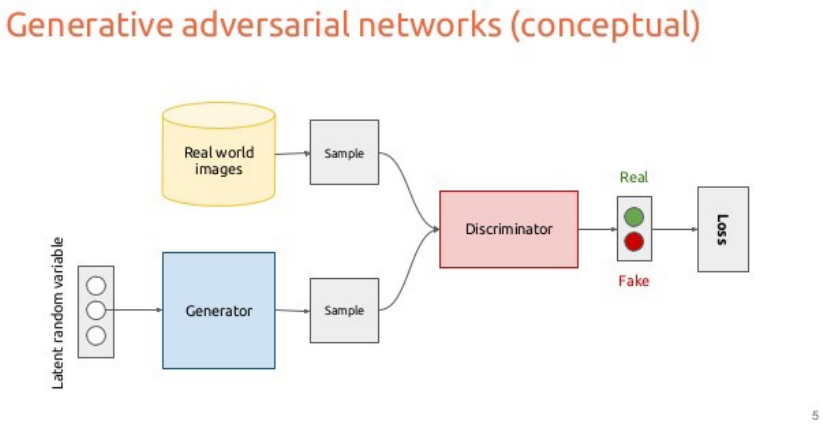
A DCGAN generator, input is a random normal vector "Code" that passes through de-convolution stacks and output an image. (Exercise: Try to explain why the code vector is randomized every-time during training, give your thought as a response is appreciated)

And the discriminator:



The discriminator takes an image as input, passes through convolution stacks and output a probability (sigmoid value) telling whether or not the image is real.

And our whole GAN architecture will be this:



GAN architecture

"The generator will try to generate fake images that fool the discriminator into thinking that they're real. And the discriminator will try to distinguish between a real and a generated image as best as it could when an image is fed."

They both get stronger together until the discriminator cannot distinguish between the real and the generated images anymore. It could do nothing more than guessing with a probability of 0.5 for both choices because the generator generates really realistic face images.

From the official GAN paper:

> *The generative model can be thought of as analogous to a team of counterfeiters, trying to produce fake currency and use it without detection, while the discriminative model is analogous to the police, trying to detect the counterfeit currency. Competition in this game drives both teams to improve their methods until the counterfeits are indistiguishable from the genuine articles.*

In the ideal optimal state, the generator will know how to generate realistic face images and the discriminator will know what faces are composed of.

The optimal generator

Intuitively, the **Code** vector that I shown earlier in the generator will represent things that are abstract. For example, if the **Code** vector has 100 dimensions, **there might be a dimension that represents "face age" or "gender" automatically.**

Why would it learn such representation? Because knowing people ages and their gender helps you draw their face more properly!

The optimal discriminator

When given an image, the discriminator must be looking for components of the face to be able to distinguish correctly.

Intuitively, some of the discriminator's hidden neurons will be excited when it sees things like eyes, mouths, hair, etc. These features are good for other purposes later like classification!

How to train

We train both the generator and the discriminator to make them stronger together and avoid making one network significantly stronger than the other by taking turn.

Why train both networks together by taking turn instead of training each to be strong separately?

If one network is too strong, the other won't be able to keep up and you will be left with two dumb networks. One network would think that it's smart without knowing that it's competing with a dumb opponent. And the one that thinks it's smart will likely over-fit to the dumb one.

Training the discriminator

Just give it an image from the training set and another one generated image from the generator. It should output 0 if it gets fed a generated image. And it should output 1 if it gets fed a real image.

Technical stuff: A cross entropy loss optimizer will do. Piece of cake!

Training the generator

The generator must try to make the discriminator outputs 1 (real) when given its generated image.

Now, this is an interesting part.

Suppose that the generator generated one image and the discriminator thought that this image has 0.40 probability of being a real image, how should the generator tweak its generated image to increase that probability to, say 0.41?

The answer is:

In order to train the generator, the discriminator have to tell the generator how to tweak that generated image to be more realistic.

The generator must ask for suggestions from the discriminator!

Intuitively, the discriminator tells **how much to tweak each pixel** in order to make the image a little bit more realistic.

Technically, you do that by back-propagating the gradients of the discriminator's output with respect to the generated image. That way, you will have gradients tensor that have the same shape as the image.

If you add these gradients to the generated image, it will make the image more realistic (in the discriminator's eyes).

But we are not just going to add these gradients to the image.

Instead, we will back-propagate these image gradients further into all the weights that made up the generator so that the generator learns how to generate this new image.

> Let me repeat this, to generate a good stuff, you have to present your work to the teacher and receive feedback!

> It would be cruel if the discriminator does not help the generator, because the generator here is doing a tougher job. It generates stuff!

And that's how the generator is trained.

You keep taking turn training them like this until you reach an equilibrium.

. . .

Intuition in plain (maybe) English

If you are confused, here is the intuition as a conversation of the two networks trying to learn in the **early dumb state**:

> *G: I have an image of a face, is it realistic enough compared to all the face images you have seen in the past?*

> *D: It's quite realistic but also looks like it's generated. (yielding 0.4 probability for real image) I am not quite sure but I guess that what you gave me is probably a generated image.*

> *G: That's right! It's an image I generated. How do I tweak it to look more realistic?*

> *D: Let me think for a sec. (Actually back-propagating in the head) I think you should add the eyes into your images. Face images often have eyes in them.*

> *(Technically speaking: I think you should increase pixel number 0 by 1, decrease pixel number 1 by 5, …, increase pixel number 4095 by 8)*

> *G: Roger that. (Back-propagating those gradients to all the weights)*

Dumbness

The above is a dumb conversation. Both of them are dumb. The discriminator is not even sure whether or not faces should have eyes. It even says that the generated image without eyes looks realistic! (A smart discriminator should certainly says no to that image because a face image should certainly have eyes in it!)

. . .

After training for awhile they will be come smarter until they reach an optimal state where they are really clever.

And here is the intuition as a conversation of the two networks trying to learn in the **optimal clever state**:

> *G: I have an image of a face, is it realistic enough compared to all the face images you have seen in the past?*

> *D: It's really realistic. (yielding 0.8 confidence probability for real image) But I don't have a clue whether or not this is a real or a generated image. Because apparently, you did so damn good at generating realistic images. But if I had to predict, you gave me a real image.*

> *G: It's an image I generated. I know that it's already realistic but I want more. How do I tweak it to look more realistic?*

> *D: Let me think for a sec. (Actually back-propagating in the head) I think your face image has every components I think you should have. It's really realistic for me. Your image certainly have eyes, mouth, ears, hair and that face is of a young boy. I don't think I have anything to suggest. But if you really want something, go and remove beard from that young boy.*

> *(Technically speaking: I think you should increase pixel number 0 by 6, decrease pixel number 1 by 7, …, increase pixel number 4095 by 2)*

> *G: Roger that. (Back-propagating those gradients to all the weights)*

Cleverness

After they are clever, the generator will generate realistic images. And the discriminator cannot distinguish these cleverly generated images anymore.

They both understand concepts like beard, eyes, mouths, hairs, young faces without any supervision from humans.

You have reached an equilibrium.

If you continue teaching the generator to make an image more real, you will likely over-fit into the thought of the discriminator like thinking that a boy should have no beard at all. That's a thought that the discriminator has developed but it might be incorrect. It's like an opinion from a teacher that you should not trust too much. If you keep training, you will gain nothing.

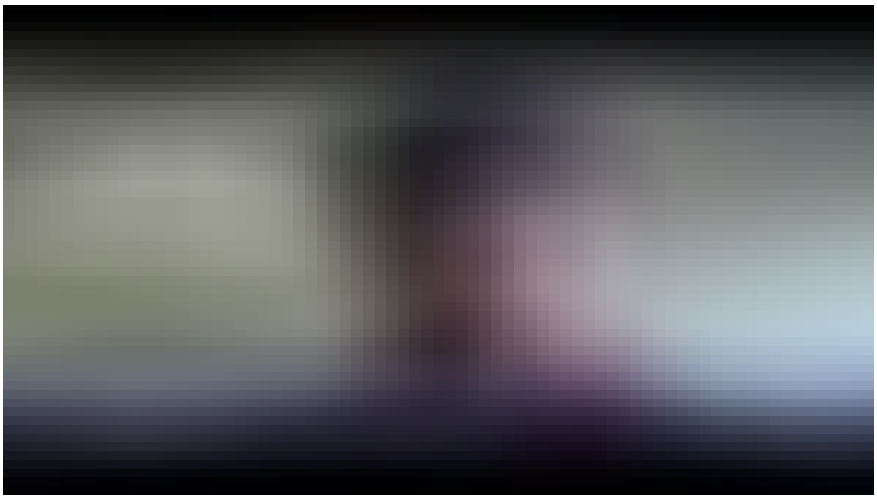
Conclusion

Both networks aren't fighting each other, they have to work cooperatively to achieve their joint goal. The discriminator has to teach the generator by giving tweak suggestions to the generated data during the entire training process. While it also learns to be a better teacher overtime.

They both get stronger together and hopefully reach an equilibrium.

Bonus

If you want to learn directly from the GAN inventor, watch this video by Ian Goodfellow:



Turns out, the adversarial network is not so adversarial after all.

PS. If you want to learn more about machine learning, I have a list dedicated for that.