What are GANS? video transcript

One of my favorite machine learning algorithms is Generative Adversarial Networks, or GAN. It pits two AI models off against each other, hence the "adversarial" part.

Now, most machine learning models are used to generate a prediction. So, we start with some input training data, and we feed that into our model. A model then makes a prediction in the form of output. And we can compare the predicted output with the expected output from the training data set. And then based upon that expected output and the actual predicted output, we can figure out how we should update our model to create better outputs. That is an example of supervised learning.

A GAN is an example of unsupervised learning, it effectively supervises itself, and it consists of two sub-models. So, we have a generator sub-model, and we have a discriminator sub-model.

Now, the generator's job is to create fake input or fake samples. And the discriminator's job is to take a given sample and figure out if it is a fake sample or if it's a real sample from the domain. And therein lies the adversarial nature of this. We have a generator creating fake samples and sending them to a discriminator. The discriminator is taking a look at a given sample and figuring out, "Is this a fake sample from the generator? Or is this a real sample from the domain set?"

Now, this sort of scenario is often applied in image generation. There are images all over the internet of generators that have been used to create fake 3D models, fake faces, fake cats, and so forth.So, this really works by the generator iterating through a number of different cycles of creating samples, updating its model and so forth until it can create a sample that is so convincing that it can fool a discriminator and also fool us humans as well.

So, let's take an example of how this works with, let's say, a flower. So, we are going to train a generator to create really convincing fake flowers, and the way that we start by doing this is we need to, first of all, train our discriminator model to recognize what a picture of a flower looks like. So, our domain is lots of pictures of flowers, and we will be feeding this into the discriminator model and telling it to look at all of the attributes that make up those flower images. Take a look at the colors, the shading, the shapes, and so forth. And when our discriminator gets good at recognizing real flowers, then we'll feed in some shapes that are not flowers at all and make sure that it can discriminate those as being not-flowers.

Now, this whole time our generator here was frozen, it wasn't doing anything. But when our discriminator gets good enough at recognizing things from our domain, then we apply our generator to start creating fake versions of those things. So, a generator is going to take a random input vector, and it is going to use that to create its own fake flower.

Now, this fake flower image is sent to the discriminator, and now the discriminator has a decision to make: is that image of a flower the real thing from the domain, or is it a fake from the generator?

Now, the answer is revealed to both the generator and the discriminator. The flower was fake and based upon that, the generator and discriminator will change their behavior. This is a zero-sum game, there's always a winner and a loser. The winner gets to remain blissfully unchanged. Their model doesn't change at all, whereas the loser has to update their model.

So if the discriminator successfully spotted that this flower was a fake image, the lead discriminator remains unchanged. But the generator will need to change its model to generate better fakes.Whereas if the reverse is true and the generator is creating something that fools the discriminator, the discriminator model will need to be updated itself in order to better be able to tell where we have a fake sample coming in, so it's fooled less easily.

And that's basically how these things work. And we go through many, many iterations of this until the generator gets so good that the discriminator can no longer pick out its fakes. And there we have built a very successful generator to do whatever it is we wanted it to do.

Now, often in terms of images, the generator and the discriminator are implemented as CNNs.
These are Convolutional Neural Networks. CNN's are a great way of recognizing patterns in image data and entering into sort of the area of object identification. We have a whole separate video on CNNs, but they're a great way to really implement the generator and discriminator in this scenario.

But the whole process of a GAN isn't just to create really good fake flowers or fake cat images for the internet. You can apply it to all sorts of use cases.

So take, for example, video frame prediction. If we fit in a particular frame of video from a camera, we can use a GAN to predict what the next frame in this sequence will look like. This is a great way to be able to predict what's going to happen in the immediate future and might be used, for example, in a surveillance system. If we can figure out what is likely to happen next, we can take some action based upon that.

There's also other things you can do, like image enhancement. So, if we have a kind of a low-resolution image, we can use a GAN to create a much higher resolution version of the image by figuring out what each individual pixel is and then creating a higher resolution version of that.

And we can even go as far as using this for things that are not related to images at all, like encryption. But we can create a secure encryption algorithm that can be decrypted and encrypted by the sender and receiver, but cannot be easily intercepted, again by going through these GAN iterations to create a really good generator.

So that's GAN. It's the battle of the bots where you can take your young, impressionable and unchanged generator and turn it into a master of forgery.

If you have any questions, please drop us a line below. And if you want to see more videos like this in the future, please like and subscribe.