12 Generative Adversarial Networks

Back in Chapter 3, we introduced the idea of deep learning models that can create novel and unique pieces of visual imagery—images that we might even be able to call *art*. In this chapter, we combine the high-level theory from Chapter 3 with the convolutional networks from Chapter 10, the Keras `Model` class from Chapter 11, and a couple of new layer types, enabling you to code up a generative adversarial network (GAN) that outputs images in the style of sketches hand drawn by humans.

Essential GAN Theory

At its highest level, a GAN involves two deep learning networks pitted against each other in an *adversarial* relationship. As depicted by the trilobites in Figure 3.4, one network is a *generator* that produces forgeries of images, and the other is a *discriminator* that attempts to distinguish the generator's fakes from the real thing. Moving from trilobites to slightly more-technical schematic sketches, the generator is tasked with receiving a random noise input and turning this into a fake image, as shown on the left in Figure 12.1. The discriminator—a binary classifier of real versus fake images—is shown in Figure 12.1 on the right. (The schematics in this figure are highly simplified for illustrative purposes, but we'll go into more detail shortly.) Over several rounds of training, the generator becomes better at producing more-convincing forgeries, and so too the discriminator improves its capacity for detecting the fakes. As training continues, the two models battle it out, trying to outdo one another, and, in so doing, both models become more and more specialized at their respective tasks. Eventually this adversarial interplay can culminate in the generator producing fakes that are convincing not only to the discriminator network but also to the human eye.

Training a GAN consists of two opposing (*adversarial!*) processes:

1. *Discriminator training*: As mapped out in Figure 12.2, in this process the generator produces fake images—that is, it performs inference only[1]—while the discriminator learns to tell the fake images from real ones.

1. Inference is forward propagation alone. It does not involve model training (via, e.g., backpropagation).

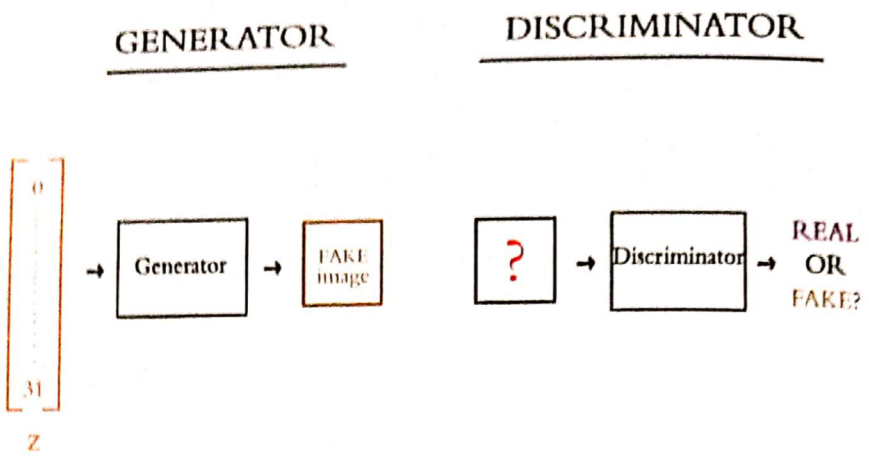

Figure 12.1 Highly simplified schematic diagrams of the two models that make up a typical GAN: the generator (left) and the discriminator (right)

2. *Generator training*: As depicted in Figure 12.3, in this process the discriminator judges fake images produced by the generator. Here, it is the *discriminator* that performs inference only, whereas it's the *generator* that uses this information to *learn*—in this case, to learn how to better fool the discriminator into classifying fake images as real ones.

Thus, in each of these two processes, one of the models creates its output (either a fake image or a prediction of whether the image is fake) but is not trained, and the other model uses that output to learn to perform its task better.

During the overall process of training a GAN, discriminator training alternates with generator training. Let's dive into both training processes in a bit more detail, starting with discriminator training (see Figure 12.2):

- The generator produces fake images (by inference; shown in black) that are mixed in with batches of real images and fed into the discriminator for training.
- The discriminator outputs a prediction ($\hat{y}$) that corresponds to the likelihood that the image is real.
- The cross-entropy cost is evaluated for the discriminator's $\hat{y}$ predictions relative to the true y labels.
- Via backpropagation tuning the discriminator's parameters (shown in green), the cost is minimized in order to train the model to better distinguish real images from fake ones.

Note well that during discriminator training, it is *only* the discriminator network that is learning; the generator network is not involved in the backpropagation, so it doesn't learn anything.

Now let's turn our focus to the process that discriminator training alternates with: the training of the generator (shown in Figure 12.3):

- The generator receives a random noise vector z as input[2] and produces a fake image as an output.

2. This random noise vector z corresponds to the *latent-space* vector introduced in Chapter 3 (see Figure 3.4), and *it is unrelated* to the z variable that has been used since Figure 6.8 to represent $w \cdot x + b$. We cover this in more detail later on in this chapter.

TRAINING THE DISCRIMINATOR

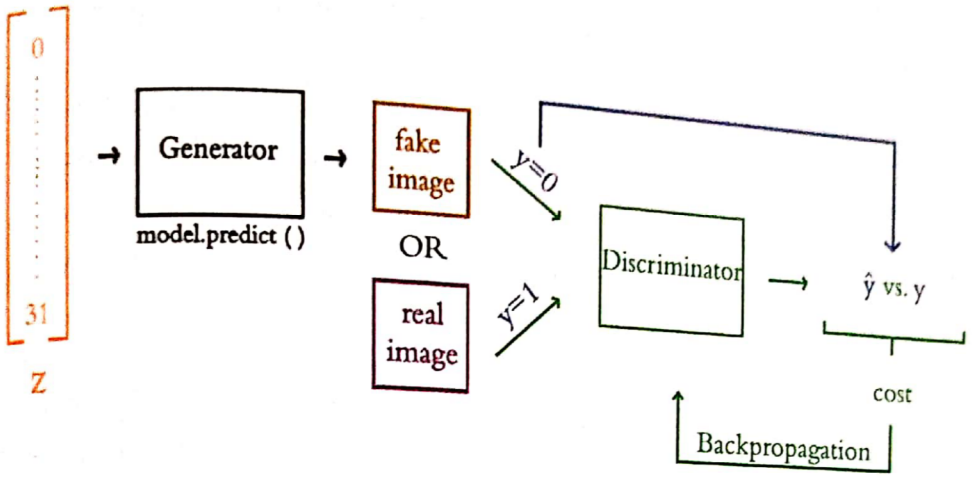

Figure 12.2 This is an outline of the discriminator training loop. Forward propagation through the generator produces fake images. These are mixed into batches with real images from the dataset and, together with their labels, are used to train the discriminator. Learning paths are shown in green, while non-learning paths are shown in black and the blue arrow calls attention to the image labels, *y*.

TRAINING THE GENERATOR

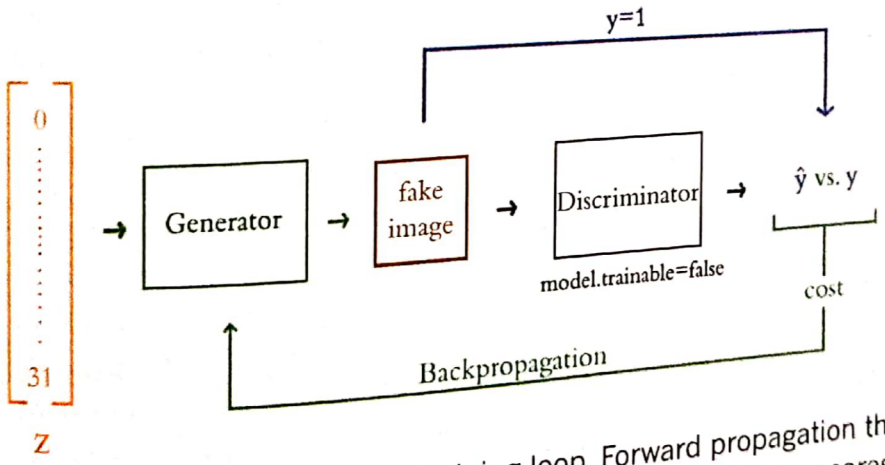

Figure 12.3 An outline of the generator training loop. Forward propagation through the generator produces fake images, and inference with the discriminator scores these images. The generator is improved through backpropagation. As in Figure 12.2, learning paths are shown in green, and non-learning paths are shown in black. The blue arrow calls attention to the relationship between the image and its label *y* which, in the case of generator training, is always equal to 1.

- The fake images produced by the generator are fed directly into the discriminator as inputs. Crucially to this process, we lie to the discriminator and label all of these fake images *as real* ($y = 1$).
- The discriminator (by inference; shown in black) outputs $\hat{y}$ predictions as to whether a given input image is real or fake.
- Cross-entropy cost here is used to tune the parameters of the generator network (shown in green). More specifically, the generator learns how convincing its fake images are to the discriminator network. By minimizing this cost, the generator will learn to produce forgeries that the discriminator mistakenly labels as real—forgeries that may even appear to be real to the human eye.

So, during generator training, it is *only* the generator network that is learning. Later in this chapter, we show you how to freeze the discriminator's parameters so that backpropagation can tune the generator's parameters without influencing the discriminator in any way.

At the onset of GAN training, the generator has no idea yet what it's supposed to be making, so—being fed random noise as inputs—the generator produces images of random noise as outputs. These poor-quality fakes contrast starkly with the real images—which contain combinations of features that blend to form actual images—and therefore the discriminator initially has no trouble at all learning to distinguish real from fake. As the generator trains, however, it gradually learns how to replicate some of the structure of the real images. Eventually, the generator becomes crafty enough to fool the discriminator, and thus in turn the discriminator learns more-complex and nuanced features from the real images such that outwitting the discriminator becomes trickier. Back and forth, alternating between generator training and discriminator training in this way, the generator learns to forge ever-more-convincing images. At some point, the two adversarial models arrive at a stalemate: They reach the limits of their architectures, and learning stalls on both sides.[3]

At the conclusion of training, the discriminator is discarded and the generator is our final product. We can feed in random noise, and it will output images that match the style of the images the adversarial network was trained on. In this sense, the generative capacity of GANs could be considered *creative*. If provided with a large training dataset of photos of celebrity faces, a GAN can produce convincing photos of "celebrities" that have never existed. As in Figure 3.4, by passing specific z values into this generator, we would be specifying particular coordinates within the GAN's latent space, enabling us to output a celebrity face with whatever attributes we desire—such as a particular age, gender, or type of eyeglasses. In the GAN that you'll train in this chapter, you'll use a training dataset consisting of sketches hand drawn by humans, so our GAN will learn to produce novel drawings—ones that no human mind has conceived of before. Hold tight for that section, where we discuss the specific architectures of the generator and discriminator in more detail. First, though, we describe how to download and process these sketch data.

3. More-complex generator and discriminator networks would learn more-complex features and produce more-realistic images. However, in some cases we don't *need* that complexity, and of course these models would be harder to train.