Ch 15 GANs

Goal:

generate new samples that are statistically indistinguishable from a set of real training data.

A single new sample is generated by

1. (i) choosing a latent variable from a simple base distribution (e.g., a standard normal) and then
2. (ii) passing this data through a network $x = g(z, \theta)$ with parameters θ. This network is known as the generator. During the learning process, the goal is to find parameters θ so that the samples look “similar” to the real data

Similarity can be defined in many ways, but the GAN uses the principle that the sam- ples should be indistinguishable from the true data.

How. Illustration of training

- a second network $f(\cdot, \phi)$ called the discriminator is introduced.

The goal of this network is to classify its input as being a real example or a generated sample. The discriminator provides a signal that can be used to improve the generation process.

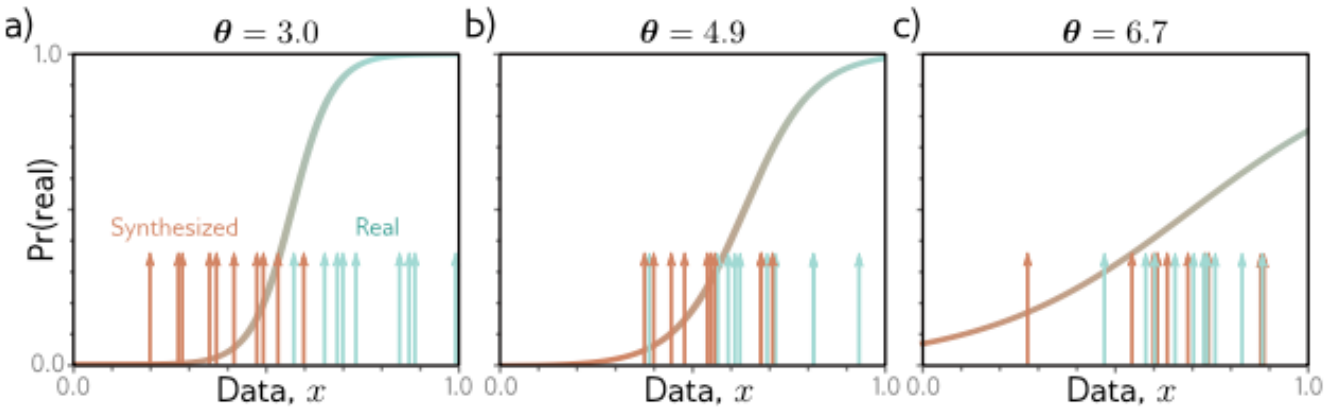


Figure: principle behind this scheme. We start with

- a training set S^m of real 1D examples.
- A different batch of ten of these examples is shown in each panel (cyan arrows).

To create a batch of samples {x∗j }, we use the simple generator:

$$x_j = g(z_j, \theta) = z_j + \theta, \tag{1}$$

where latent variables {zj} are drawn from a standard normal distribution, and the parameter θ translates the generated samples along the x-axis (figure 15.1).

- At initialization θ = 3.0, and the generated samples (orange arrows) lie to the left of the true samples (cyan arrows). The discriminator is trained to distinguish the generated samples from the real examples (the sigmoid curve indicates the probability that a data point is real).
- During training, the generator parameters θ are manipulated to increase the probability that its samples are classified as real. In this case, this means increasing θ so that the samples move rightwards where the sigmoid curve is higher.

We alternate between retraining the discriminator and updating the generator.

- Figure show two iterations of this process. It gradually becomes harder to classify the data, and so the impetus to change θ becomes weaker (i.e., the sigmoid becomes flatter).
- At the end of the process, there is no way to distinguish the two sets of data.
- the discriminator, which now has chance performance, is discarded and we are left with a generator that makes plausible samples.

Training Loss for discriminator

- Input Labeled dataset of Natural images $S^N = (x_i, 1)$ and generated images $S^G = (x_j = g(z_j), 0)$ labelled real and rake.
- Train discriminator with classification loss to distinguish:

$$\min_{\phi} \widehat{L}_S(f(x_i, \phi), y_i), \qquad x_i = g(z_j, \theta), S = S^N \cap S^G \tag{2}$$

Training loss for generator

Generator wants to confuse discriminator. So given $f(x_i, \phi)$ do

$$\max_{\theta} \widehat{L}_S(f(x_i, \phi), y_i), \qquad x_i = g(z_j, \theta), S = S^N \cap S^G \tag{3}$$

Combined training

- can train by alternating ascent / descent for each of this.
- Combined loss is

$$\max_{\theta} \min_{\phi} \widehat{L}_S(f(x_i, \phi), y_i), \qquad x_i = g(z_j, \theta), S = S^N \cap S^G \tag{4}$$

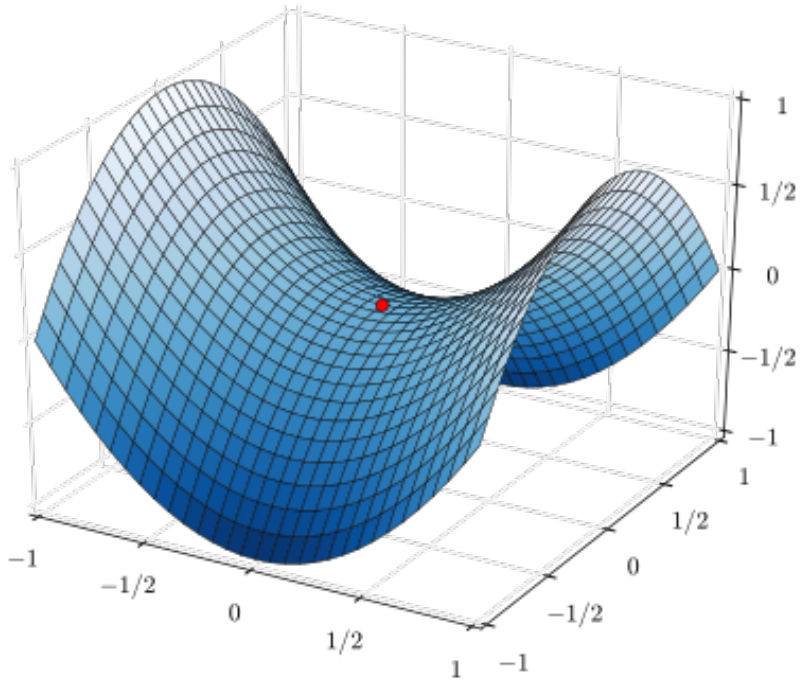
or

$$\max_{\theta} \min_{\phi} \sum_i \ell(f(x_i, \phi), y_i = 1) + \sum_i \ell(f(g(z_i), \phi), y_i = 0) \tag{5}$$

Minimax game

Even if the models are linear, we now have two sets of parameters, and the goal is to find a *saddle point*

https://en.wikipedia.org/wiki/Saddle_point

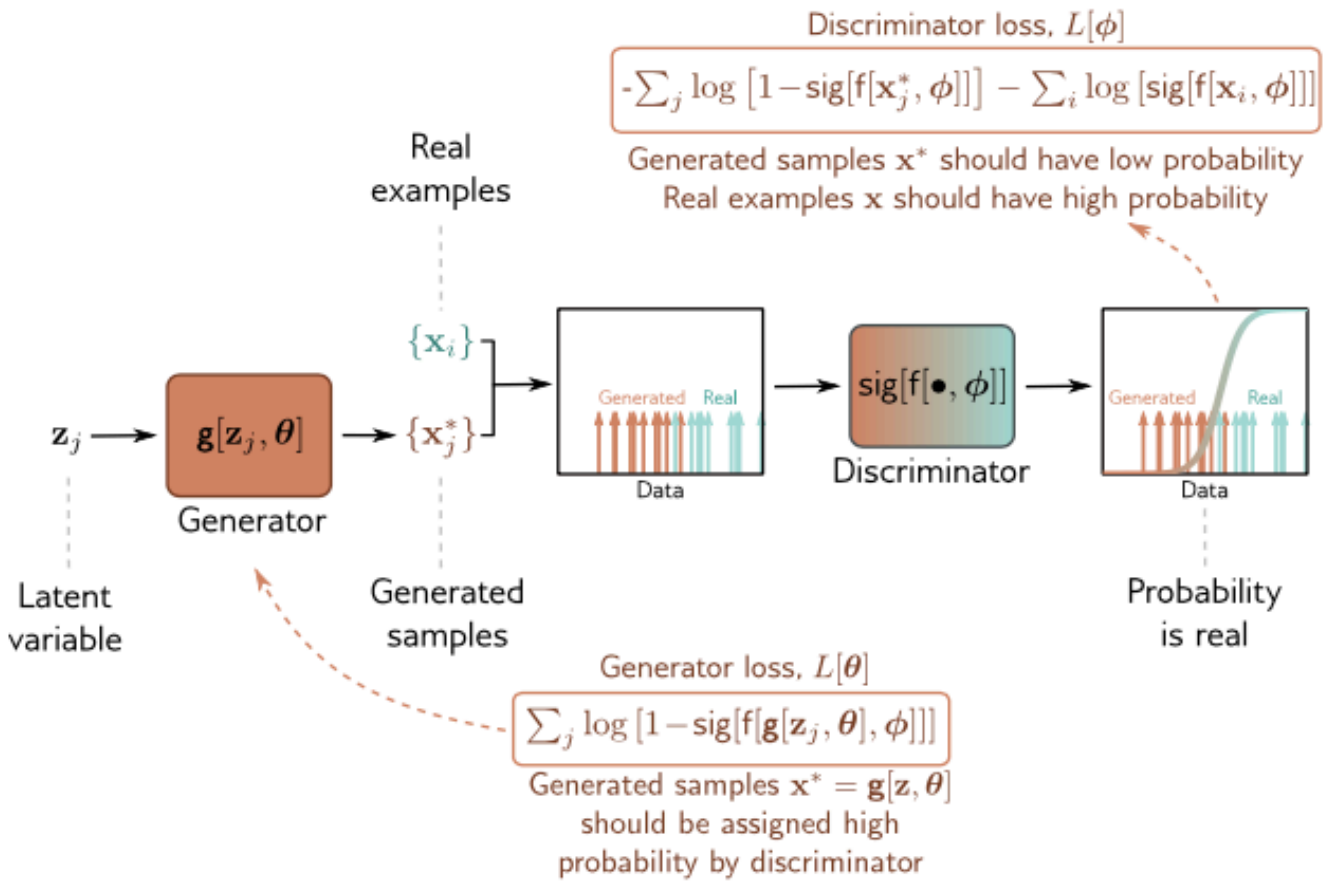


$$\max_{\theta} \min_{\phi} L(\theta, \phi) = -\theta^2 + \phi^2 \tag{6}$$

Saddle point is $(0, 0)$

Trickier to train! Unstable

.



Difficulty training GANs

Theoretically, the GAN is fairly straightforward. However, GANs are notoriously difficult to train. For example, to get the DCGAN to train reliably it was necessary to (i) use strided convolutions for upsampling and downsampling; (ii) use BatchNorm in both generator and discriminator except in the last and first layers, respectively; (iii) use the leaky ReLU activation function in the discriminator; and (iv) use the Adam optimizer but with a lower momentum coefficient than usual. This is unusual; most deep learning models are relatively robust to such choices.

Failures of the GAN loss

no coverage.

- Generator only cares about sample quality (i.e. samples are hard to discriminate). but no reason to cover the support of the distribution.
- So, e.g. can memorize half the dataset, and get zero loss.

How to force coverage? Need to say for each real samples, there will also be a similar generated sample. (Invertible model)

$$\forall x_i \in S, \exists z \implies g(z) = x_i \tag{7}$$

this condition is not apparent in the classification loss.

vanishing gradients

- same as for classifiers. If start out too bad, gradients are zero. No gradient signal.

Conditional Generation

GANs produce realistic images don't' specify their attributes: we can't choose the hair color, ethnicity, or age of face samples without training separate GANs for each combi- nation of characteristics. Conditional generation models provide us with this control.

conditional GAN?

The conditional GAN passes a vector c of attributes to both the generator and discrimi- nator, which are now written as $g[z, c, \theta]$ and $f[x, c, \varphi]$, respectively. The generator aims to transform the latent variable z into a data sample x with the correct attribute c. The discriminator's goal is to distinguish between (i) the generated sample with the target attribute or (ii) a real example with the real attribute

Auxiliary classifier GAN

1. The auxiliary classifier GAN or ACGAN simplifies conditional generation by requiring that the classifier correctly predicts the attribute (figure 15.13b). For a discrete attribute with C categories, the discriminator takes the real/synthesized image as input, and has C+1 outputs; the first is passed through a sigmoid function and predicts if the sample is generated or real. The remaining outputs are passed through a softmax function to predict the probability that the data belongs to each of the C classes. Networks trained with this method can synthesize multiple classes from ImageNet