GANs – General notes – John Flynn

Quote from Deep Learning book (section 20.10.2) that explains the difficulties of generator networks compared to deep supervised learning.

> Approaches based on differentiable generator networks are motivated by the success of gradient descent applied to differentiable feedforward networks for classification. In the context of supervised learning, deep feedforward networks trained with gradient-based learning seem practically guaranteed to succeed given enough hidden units and enough training data. Can this same recipe for success transfer to generative modeling?
>
> Generative modeling seems to be more difficult than classification or regression because the learning process requires optimizing intractable criteria. In the context of differentiable generator nets, the criteria are intractable because the data does not specify both the inputs z and the outputs x of the generator net. **In the case of supervised learning, both the inputs x and the outputs y were given, and the optimization procedure needs only to learn how to produce the specified mapping. In the case of generative modeling, the learning procedure needs to determine how to arrange z space in a useful way and additionally how to map from z to x.**
>
> Dosovitskiy et al. (2015) studied a simplified problem, where the correspondence between z and x is given. Specifically, the training data is computer-rendered imagery of chairs. The latent variables z are parameters given to the rendering engine describing the choice of which chair model to use, the position of the chair, and other configuration details that affect the rendering of the image. Using this synthetically generated data, a convolutional network is able to learn to map z descriptions of the content of an image to x approximations of rendered images. This suggests that contemporary differentiable generator networks have sufficient model capacity to be good generative models, and that contemporary optimization algorithms have the ability to fit them. **The difficulty lies in determining how to train generator networks when the value of z for each x is not fixed and known ahead of each time.**
>
> The following sections describe several approaches to training differentiable generator nets given only training samples of x.

Variational autoencoder disadvantages. Goodfellow et al. (2016):

- Samples tend to be blurry (since its optimisation is similar to mean squared error).
- Tend to use only a small subset of z-space.

Manifold learning/interpolation of the z-space.

Quote from Deep Learning book (section 20.10.4) on the difficulties of finding equilibria in a minimax game:

> Note that the equilibria for a minimax game are not local minima of v. Instead, they are points that are simultaneously minima for both players' costs. **This means that they are saddle points of v that are local minima with respect to the first player's parameters and local maxima with respect to the second player's parameters. It is possible for the two players to take turns increasing then decreasing v forever, rather than landing exactly on the saddle point where neither player is capable of reducing its cost**. It is not known to what extent this non-convergence problem affects GANs.

Same section on better performing formulation of the GAN game

> In realistic experiments, the best-performing formulation of the GAN game is a different formulation that is neither zero-sum nor equivalent to maximum likelihood, introduced by Goodfellow et al. (2014c) with a heuristic motivation. In this best-performing formulation, the generator aims to increase the log probability that the discriminator makes a mistake, rather than aiming to decrease the log probability that the discriminator makes the correct prediction. This reformulation is motivated solely by the observation that it causes the derivative of the generator's cost function with respect to the discriminator's logits to remain large even in the situation where the discriminator confidently rejects all generator samples.

Proof of optimal GAN discriminator:

https://www.desmos.com/calculator/odj1obavmt

Image on generative networks from OpenAI blog:

https://blog.openai.com/generative-models/

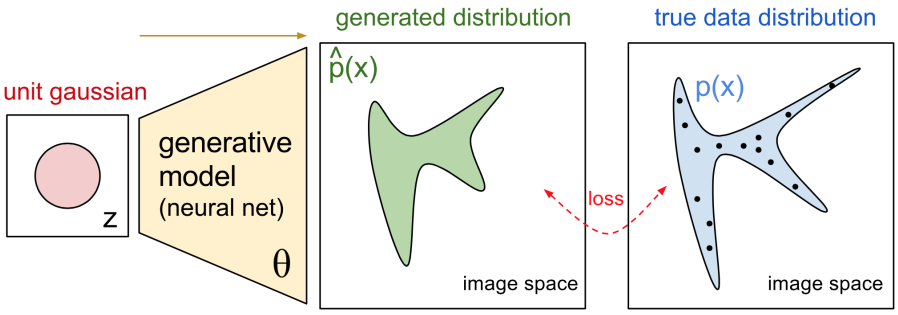

Figure 1: Generative model diagram from OpenAI.