Generative Adversarial Networks (GANs) and Deep Reinforcement Learning

Challenges with GANs

Usually, we would want our GAN to produce a range of outputs. We would expect, for example, another face for every random input to the face generator that we design. Instead, through subsequent training, the network learns to model a particular distribution of data, which gives us a monotonous output which is give in the slide. In the process of training, the generator is always trying to find the one output that seems most plausible to the discriminator. Because of that, the discriminator's best strategy is always to reject the output of the generator. But if the next generation of discriminator gets stuck in a local minimum and doesn't find its way out by getting its weights even more optimised, it'd get easy for the next generator iteration to find the most plausible output for the current discriminator.

This phenomenon happens when the discriminator performs significantly better than the generator. Either the updates to the discriminator are inaccurate, or they disappear. One of the proposed reasons for this is that the generator gets heavily penalised, which leads to saturation in the value post-activation function, and the eventual gradient vanishing.

Since there are two networks being trained at the same time, the problem of GAN convergence was one of the earliest, and quite possibly one of the most challenging problems since it was created. The utopian situation where both networks stabilise and produce a consistent result is hard to achieve in most cases. One explanation for this problem is that as the generator gets better with next epochs, the discriminator performs worse because the discriminator can't easily tell the difference between a real and a fake one. If the generator succeeds all the time, the discriminator has a 50% accuracy, like that of flipping a coin. This poses a threat to the convergence of the GAN.