**GAN models**

Even though VAEs manage to generate data that are, to some extent, similar to the original data, they are not the most accurate generation method, and the generated images can often be blurry [93]. One solution and a natural next step in the tale of generative models, is the model Goodfellow et al. (2014) proposed called generative adversarial network (GAN) [7]. GANs work is trained with an “unusual” technique that avoids the typical process of maximizing a log likelihood, which involves various approximations and is often intractable [93]. GANs don’t require specified likelihood functions or Markov chains either [92]. Apart from the fact that both models utilize a latent space to generate new samples, there are no clear similarities between autoencoders and GANs. However, some GANs use an encoder-decoder architecture as the generator [8]. This architecture is able to learn the latent space representation of the original data, reconstruct the original data, and generate new samples. But what is a GAN, what is a generator, and how does it work?

Simply put, a GAN model contains two separate neural networks: A generator, which as you may have guessed, generates new data, and a discriminator which are supposed to distinguish between real and generated data [7]. Goodfellow et al. [7], who first proposed the model, described it as the generator being a counterfeiter trying to fake currency and use it without being caught, while the discriminator is the police, trying to detect the fake currency. Personally, I love the following metaphor of a GAN model:

*«Consider the dynamics of interactions between an art forger and a curator. The forger is set on tricking the art curator to accept his forgeries as real to be able to turn a profit. The goal of the art curator is to be able to tell the forgeries from the real art to be able to turn a profit. We can think of the interactions between the forger and the curator as a game between two players, where each player seeks to maximize their own profits at the expense of the other.»* (nadli)

Here, the forger represents the generator who tries to “fool” the curator which represents the discriminator. The generator tries to create data that imitates real data as good as possible, trying to reduce (minimize) the discriminators’ ability to correctly distinguish between real and generated data. On the other hand, the discriminator is trying to tell the real and generated data apart, by assigning a probability of being real, and thus want to maximize the number of correct classifications real and generated data. [7, 18, nadli]

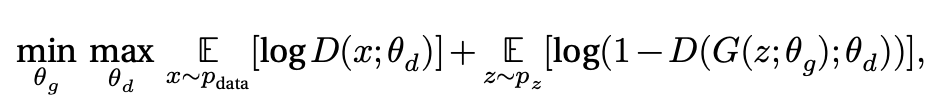


Figure 3.nadli

The two networks work against each other in a minmax game that is described with the art forger and curator. Each network is participating in the minmax game trying to optimize its own objective function. This game theory concept, where one player (network) wants to minimize an objective, while the other want to maximize it, is used to model the adversarial relationship between the generator and the discriminator, which provides the impression of a competitive relationship between the two networks. (18,93, nadli)

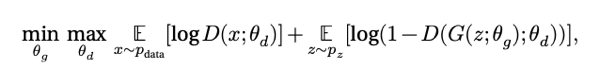
****

Figure 2. nadli

**A math equations and numbers

Description automatically generated with medium confidence**

Figure 1. goodfellow [7]

Solving the described minmax problem is a hard affair. The two networks must alternate training, trying to improve towards its objective. This alternating training method continuous until convergence or a stopping criterion is met. The perfect solution to this alternating training minmax game is called a *Nash Equilibrium*, which is defined as a point where no player can improve in terms of its objective. Nash equilibrium comes from game theory and describes the solution in a two player noncooperative competition where none of the players can improve by changing their strategy [93]. That means to get the best trained GAN, the model must achieve a Nash equilibrium, which means neither the generator nor the discriminator can improve in terms of their objective. In practice, that means the discriminator predicts close to 0.5 chance of being real for both real data and generated data since the generated data is almost equal to the real data [7, 8].

Achieving such an equilibrium when training can be extremely challenging, since it requires both networks to converge, without any of them overfitting to early and “winning” the game alone. Trying to find such a solution can lead oscillations, early convergence (overfitting), and other unstable behaviors that GANs are famous for, because they are, indeed, famous for being hard to train [7, 8].

Goodfellow and Salimans et al. [23] proposed an explanation for the unstable training of GANs. They explained that since the solution to the optimization problem is a saddle point, typical gradient descent techniques for optimizing training and updating parameters of both networks are unsuitable for the task. [23]. GANs are hard to train since getting to networks to parallelly converge is difficult. Both networks need to learn in a reasonable similar speed, not overfitting to early or taking to large jumps. They need to be synchronized in some way.

There are multiple factors and reasons within the model architecture and the training dynamics that makes it difficult to train. GANs are sensitive to changes in the input data related to data quality and how the input data is distributed. If the input data is not balanced it can affect the performance of the GAN quite drastically. Additionally, complex input data will be difficult to learn. GANs are also sensitive to hyperparameters and small changes in hyperparameters can affect the results, performance, and stability of the GAN. Vanishing gradients is also an issue when training GANs, since the gradients from the discriminator that are passed to the generator might be extremely low, which makes the update unnoticeable, and the model might not learn anything or converges very slow. Training two networks in an adversarial manner creates convergence and stability difficulties, where one network outperforms and dominates the other.

As explained earlier, the loss function of both networks uses the discriminator output, and since the discriminator quickly converges, there might not be a reliable gradient pathway for updating the generator. This occurs since when the generator is imperfect in the beginning of training, the discriminator will perform well since it will be unproblematic to confidently separate and distinguish real and fake images [93]. A proposed solutions to this are to train the discriminator x times and then train the generator one time. This prevents the generator to overfit or collapse into generating very similar samples, which is known as “mode collapse” [7], [23]. Mode collapse is a common issue when training GANs and means that the model has only learned to generate a subset (mode) of the original input, this leads to limited diversity in the generated data.

There are multiple challenges when training a GAN, and to deal with these issues require careful tuning and experimentation of the model architecture, hyperparameters, and regularization techniques. It is obvious that training the GAN correctly is the main disadvantage of GANs, but managing to get a stable trained GAN that converges offers access to the advantages of GANs. The main advantages of GANs are that it can produce very sharp and clear images. Additionally, GAN offers very fast sampling that does not need additional computation or Markov chains [93].

GANs have become tremendously popular and been the state-of-the-art and go-to for generative tasks for some time now. The influence of GANs have been huge, especially after Radford et al. (2015) extended the model with convolutional layers in a deep convolutional GAN (DCGAN). The DCGAN improved the already decent results of the GAN and made it a popular choice by proposing some structural changes.

**Evaluation**

Evaluating GANs, like other generative models, are a challenge. Generated images don’t have gold labels, which makes it hard to compare to something, since traditional evaluation metrics don’t provide dependable information regarding the quality of the model and generated samples. Common metrics used for GANs are Frechet Inception Distance (FID) and Inception score (IS), but they have limitations like any other evaluation metric, therefore the evaluation of GANs should be application specific.

**Encoder-decoder**

In the GAN that Goodfellow et al (2014) proposed, the generator consisted of a single neural network. However, as paper 2, 3 and Other sources with encoder-decoder GAN suggests, applying an encoder-decoder architecture similar to the autoencoder encoder-decoder architecture as the generator produce better results. The encoder-decoder structure encoder the original data into the latent space, where the original generator usually would sample noise from, then the encoder tries to reconstruct the original image learning the structure and patterns of the original data. Using the encoder-decoder architecture makes the decoder learn the patterns and structure of the original patterns better, which requires a reconstruction loss to measure how well the encoder-decoder architecture manages to reconstruct the original data.

To generate new images in such an architecture is the same as in the original generator architecture, which basically only is a decoder. That means generating a new image only require sampling noise similar to the latent space size, then passing the noise through the decoder.

The only difference when applying an encoder-decoder architecture as the generator in a GAN is that it requires a reconstruction loss to calculate the similarity/differences between the reconstructed and original images so that the encoder and decoder can be updated appropriately. However, the reconstruction loss is aggregated with the original generator loss from which involves how competent the generator (decoder) is in terms of generating samples that fool the discriminator. In that sense, the generator with an encoder has two main objective: generating new samples that fool the discriminator, and reconstruct the original images.