Generative Adversarial Nets Write up

qdc597

March 2024

Introduction and Concept

Generative adversarial networks are a framework in which two machine learning models gets trained simultaneously. The two models is usually referred to as the generator and the discriminator. The role of the generator is the generate samples mimicking the training data, and it is the discriminators role to judge whether a certain sample comes from the training data distribution or was generated by the generator. In this way the generator attempts to fool the discriminator by learning the training data distribution while the discriminator learns how to accurately predict the origin of a given sample. This corresponds to a min-max game between the generator and the discriminator. Formally

$$min_G max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})}[log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[log(1 - D(G(\boldsymbol{z}))]$$

Where $D(\boldsymbol{x})$ is a scalar function denoting the probability of the sample originating from the training distribution[1]. A visualisation is shown in figure 1.

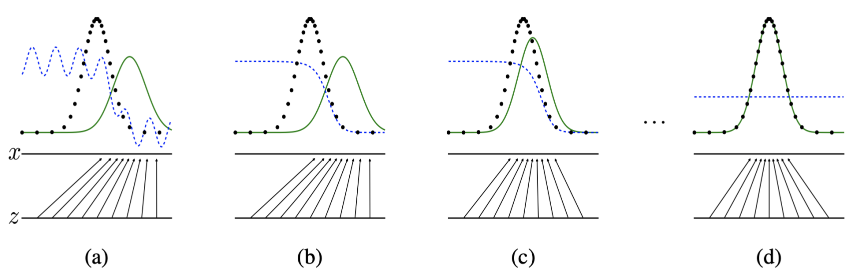


Figure 1: Visualisation of procedure from the article[1]. Distribution of the discriminator is the blue dotted line, training data distribution in black dots, and the generator distribution as the green solid line. The arrows from the z line to the x line represent the mapping learned by the generator.

Procedure

In figure 2 the outline of the training procedure is shown.

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  • Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.

  • Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.

  • Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  • Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.

  • Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

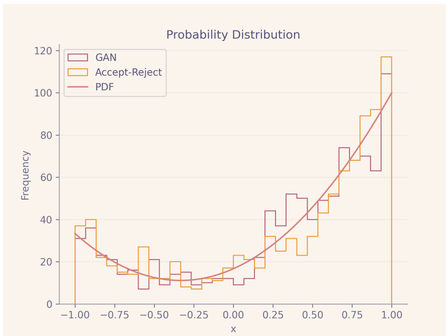
Figure 2: The algorithm presented by Goodfellow[1].

Theoretical Results: The short version

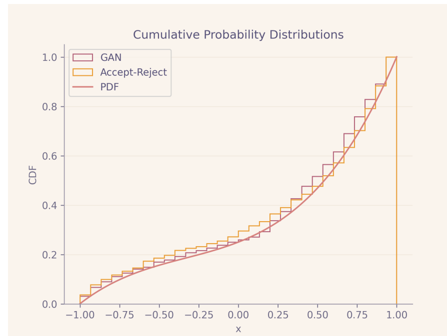
In the article Goodfellow derives some convergence guarantee given a non-parametric setting, the following is a very brief conclusion of the result. If the generator and the discriminator have infinite capacity and training time. There exist a global optimum for the combined distribution $V(D, G)$ where the distribution of the generator aligns with the training data[1].

My Application

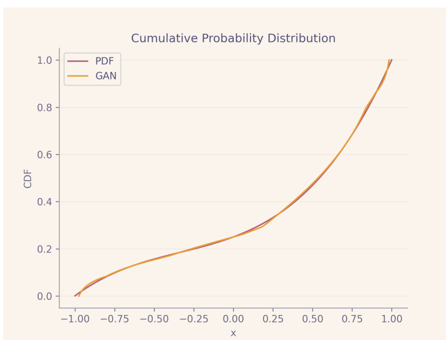
Instead of showing the authors result, I did decide to attempt to implement the models myself. The result is shown in 3. Personally the generated distribution passes the eye test, but when applying the KS-test it is apparent that it hasn't quite matched the target distribution, but it does proof the concept.



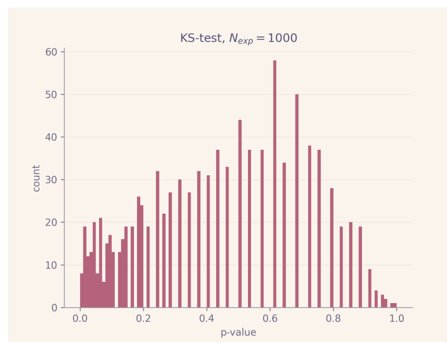
(a) A comparison of samples generated with Accept-Reject method and the GAN model



(b) Cumulative distribution of the sample from 3a.



(c) Cumulative distribution of the resulting GAN and the training PDF.



(d) KS-Test on 1000 iteration. As seen it is not quite the same distribution.

Figure 3: Overview of own application.

Conclusion

Even though the model did not match the training distribution perfectly there is still a lot promises that it could be a viable way of approximating unknown distributions. The process is very finicky, there is a ton of adjustable training parameters to manage, and on top of that there is the models to take into account. Despite of this, even with limited amount of experience in training ML models, the procedure is powerful enough to produce a reasonable result.

References

[1] Ian J. Goodfellow et al. "Generative Adversarial Nets". In: (2014). DOI: `https://arxiv.org/abs/1406.2661`.