Evaluation of Spectral Normalization for GANs Using Inception Score

Eysteinn Gunnlaugsson, Egill Vignisson, Charles Hamesse

Group 16

Abstract. We experiment various Deep Convolutional Generative Adversarial Networks (DCGANs) for image generation. We start with a vanilla DCGAN and gradually add features that are expected to improve learning in terms of speed, stability and quality of generation. We evaluate our models using the inception score and discuss its relevance on cifar10 and a dataset that we collected ourselves consisting of reptile images. Finally, we discuss our findings and challenges and draw conclusions on how the features we implemented help the training of GANs.

Keywords: Generative models, generative adversarial networks, image generation

1 Introduction

Motivate the problem you are trying to solve, attempt to make an intuitive description of the problem and also formally define the problem. (1-2 pages including title, authors and abstract)

The purpose of this project is to investigate the performance of the different types of Generative Adversarial Networks (GANs) [**?**] for image generation as well as possible improvement options. The project was originally defined based on varying levels of priority where everything assigned priority 1 was promised to be completed.

- Implement Deep Convolutional Generative Adversial Network with original loss [**?**] (priority 1)
- Implement the inception score metric [**?**] (priority 1)
- Implement Spectral Normalization (priority 1)
- Evaluate all our GANs on our reptile dataset (priority 1)
- Implement other losses (LSGAN, WGAN) [**?**] (priority 2)
- Evaluate GANs on CIFAR-100 (priority 2)
- Implement mini-batch discrimination and or other improvements [**?**]. (priority 3)

In order to evaluate the performance of these GANs, the evaluation metric known as the inception score, as described in [**?**], will be implemented. Furthermore a data set consisting of roughly 30K animal pictures (mostly reptiles), was fetched from the Flickr API while other well known options such as a subset of CIFAR-100 or ImageNet were thought of as possible replacements in the case of unsatisfactory results.

2 Background

We present the framework of GANs starting with the original paper by Ian Goodfellow [**?**] then describe more recent advances in a chronological order, as depicted in Figure 1.

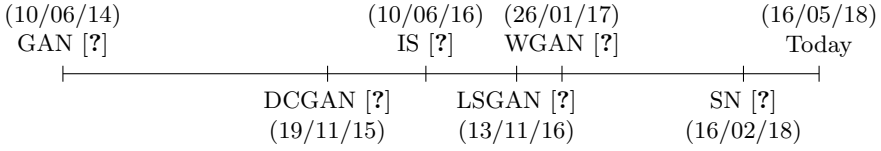

Fig. 1: Major contributions in the field of Generative Adversarial Networks.

2.1 Generative Adversarial Networks

Generative Adversarial Networks (GANs) were first introduced in [**?**]. They are a class of generative models trained in a adversarial manner by opposing two networks: a generative network G that learns the data distribution and a discriminative network D that tries and estimates the probability that a given sample came from the real training data rather than a generation from G.

The objective for G is to maximize the probability of D making a mistake, and the objective for D is to minimize that same probability. This framework corresponds to a minimax two-player game. In the space of arbitrary functions G and D, a unique equilibrium solution exists, with G recovering the real data distribution and D equal to 0.5 everywhere, meaning it's unable to distinguish the real images from the ones generated by G. Since G and D are (de-)convolutional networks, both can be trained using available backpropagation techniques.

To learn the distribution p_g over the real data $\boldsymbol{x}$, G starts from sampling input variables $\boldsymbol{z}$ from a distribution of our choice $p_z(\boldsymbol{z})$, then maps the input variables $\boldsymbol{z}$ to space $G(\boldsymbol{z};\theta_g)$ that should, after training, resemble the training data space. The discriminator, D, maps images to a boolean $D(\boldsymbol{x};\theta_d)$ indicating whether images are from training data or generated from G. The orignal minimax objective for GANs is defined as:

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

2.2 Deep Convolutional Generative Adversarial Networks

Building upon Goodfellow's work [**?**], Radford et al. apply the GAN framework to computer vision, bridging the gap between the success of Convolutional Neural Networks (CNNs)for supervised learning and GANs for unsupervised learning [**?**]. Training on various image datasets, authors show that the adversarial pair learns a hierarchy of features in both the generator and discriminator.

2.3 Inception Score

The Inception Score (IS) is first presented in [**?**]. This work (originating from OpenAI, whose team includes author of the original GAN paper Ian Goodfellow) presents a variety of new architectural features and training procedures meant to improve the training of GANs. Amongst other things, authors make the observation that GANs lack an objective function, which makes it difficult to compare performance of different models. In the context of image generation, an intuitive performance metric can be obtained by human annotators assessing the quality of the generated images. However this method isn't as scalable as one would wish, for obvious reasons. The inception score is proposed as an alternative, and shown to correlate well with human evaluation.

We describe the method briefly. By applying the Inception model [**?**] to all generated images, we get the conditional label distribution $p(y|\boldsymbol{x})$. Images that contain meaningful objects should have a conditional label distribution $p(y|\boldsymbol{x})$ with low entropy. Moreover, we expect the model to generate varied images, so the marginal $\int p(y|\boldsymbol{x} = G(z))dz$ should have high entropy. By combining these two requirements, the proposed metric becomes: $\exp(\mathbb{E}_{\boldsymbol{x}}\mathrm{KL}(p(y|\boldsymbol{x})||p(y)))$, where results are exponentiated so that values are easier to compare.

In [**?**] the Inception score is defined as:

$$IS(G) = exp\big(\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{data}}(\boldsymbol{x})} D_{KL}(p(y|\boldsymbol{x})||p(y))\big). \tag{2}$$

Here D_{KL} is known as the Kullback Libler divergence and it measures how one probability distribution diverges from another. $p(y|\boldsymbol{x})$ is the conditional label distribution while $p(y)$ is the marginalized label distribution, i.e. $p(y) = \int_{\boldsymbol{x}} (p(y|\boldsymbol{x})p_g(\boldsymbol{x})d\boldsymbol{x}$. In order to calculate the score we replace $p(y)$ with the empirical marginal class distribution $\hat{p}(y) = \sum_{i=1}^{N} p(y|\boldsymbol{x}^i)$ and estimate the expectation using the basic Monte Carlo estimator yielding:

$$IS(G) = exp\big(\sum_{i=1}^{N} D_{KL}(p(y|\boldsymbol{x}^i)||\hat{p}(y))\big). \tag{3}$$

2.4 Least Squares Generative Adversarial Networks

Viewing the discriminator as a classifier, regular GANs adopt the sigmoid cross entropy loss function. As stated in Section **??**, when updating the generator, this loss function will cause the problem of vanishing gradients for the samples that are on the correct side of the decision boundary, but are still far from the real data. To remedy this problem, we propose the Least Squares Generative Adversarial Networks (LSGANs). Suppose we use the a-b coding scheme for the discriminator, where a and b are the labels for fake data and real data, respectively. Then the objective functions for LSGANs can be defined as follows:

$$\begin{aligned}\min_D V_{\text{LSGAN}}(D) =& \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\big[(D(\boldsymbol{x})-b)^2\big] + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\big[(D(G(\boldsymbol{z}))-a)^2\big] \\ \min_G V_{\text{LSGAN}}(G) =& \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\big[(D(G(\boldsymbol{z}))-c)^2\big],\end{aligned} \tag{4}$$

where c denotes the value that G wants D to believe for fake data.

Benefits of LSGANs The benefits of LSGANs can be derived from two aspects. First, unlike regular GANs, which cause almost no loss for samples that lie in a long way on the correct side of the decision boundary (Figure **??**(b)), LSGANs will penalize those samples even though they are correctly classified (Figure **??**(c)). When we update the generator, the parameters of the discriminator are fixed, i.e., the decision boundary is fixed. As a result, the penalization will make the generator to generate samples toward the decision boundary. On the other hand, the decision boundary should go across the manifold of real data for a successful GANs learning. Otherwise, the learning process will be saturated. Thus moving the generated samples toward the decision boundary leads to making them be closer to the manifold of real data.

Second, penalizing the samples lying a long way to the decision boundary can generate more gradients when updating the generator, which in turn relieves the problem of vanishing gradients. This allows LSGANs to perform more stable during the learning process. This benefit can also be derived from another perspective: as shown in Figure **??**, the least squares loss function is flat only at one point, while the sigmoid cross entropy loss function will saturate when x is relatively large.

2.5 WGAN

2.6 Spectral Normalization

3 Approach

Describe the final approach you are take for this problem. For instance, here you would describe the details of the network's architecture. What training parameters and techniques you have used. The computational complexity of your model. And similar questions. To help explain your approach please make figures to accompany your text description. (1-3 pages)

3.1 Data Collection

Images were collected using Flickr's API. Due to varying amounts of quality pictures of objects that would be interesting to base the image generation on, a collection that included mostly reptiles with a fair amount of arachnid's thrown into the mix was ultimately settled upon. In total the data set is made up of approximately 30k color images all re-sized to dimensions of 108x108x3 before training was conducted.

3.2 Frame of Reference

In order to measure the performance of each implementation a frame of reference had to be established. As the golden standard, that every implementation would be compared to, the inception score of the actual data set was calculated resulting in a score of **1.5295098 ± 0.06244182**.

4 Experiments

In this section, you should present the results you achieved with various experiments. The results can be presented in tables, plots, etc.

The purpose of the project is to analyze the nature and effectiveness of different GAN architectures as well as different improvement options. The different models but in order to do so a frame of reference was needed

4.1 Inception Score Considerations

We present a challenge related to the computation and evaluation of the inception score. Most authors evaluate the inception score on 50K GAN-generated images, as recommend the authors of the original paper [**?**]. By running a few preliminary experiments, we quickly realize that on top of the actual training of the network, sampling and computing the inception score are also resource-intensive tasks, and sampling 50K images is simply not possible with the time or resources available for this project.

Now, the number of images considered for evaluating the inception score has an impact on this score, as shows Table 1. This is due to the fact that the inception score not only evaluates the content of a given image but also the distribution of categories amongst the whole set of images resulting from the split. In other words, the score is sensitive to the number of images divided by the number of splits.

Images	Splits	Inception score
256	5	8.13 +- 0.41
512	5	8.04 +- 0.54
1024	5	9.79 +- 0.36

Images	Splits	Inception score
256	10	6.72 +- 0.55
512	10	7.92 +- 0.56
1024	10	8.95 +- 0.44

Table 1: Inception score for various number of samples of the cifar10 dataset.

We choose to stick with 1024 generated images and 5 splits for all of our experiments. With this configuration, we have a target inception score of 9.79. As expected, this is below the claimed inception score of the whole cifar10 dataset, 11.24 [**?**]. Thus we won't reach state of the art results in terms of inception score,

but this isn't an issue since our purpose is to compare various improvements of GAN networks, which isn't affected by this choice. Other considerations on the inception score are explained in [**?**].

4.2 DCGAN

We implement our baseline model in this section. The DCGAN is evaluated on cifar10. We present the evolution of the inception score in Figure 10 and both losses in Figure 3

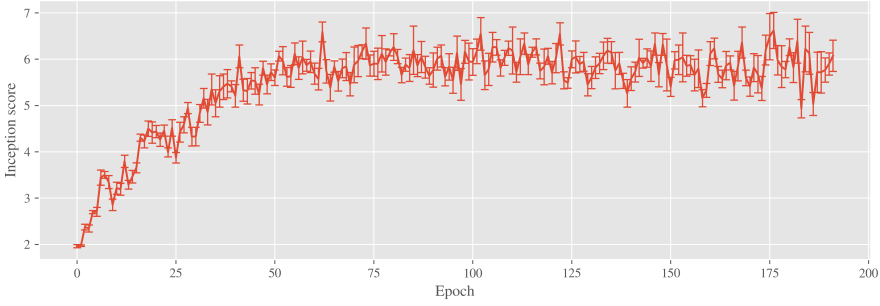

Fig. 2: DCGAN - Inception score, training on cifar10 over 190 epochs

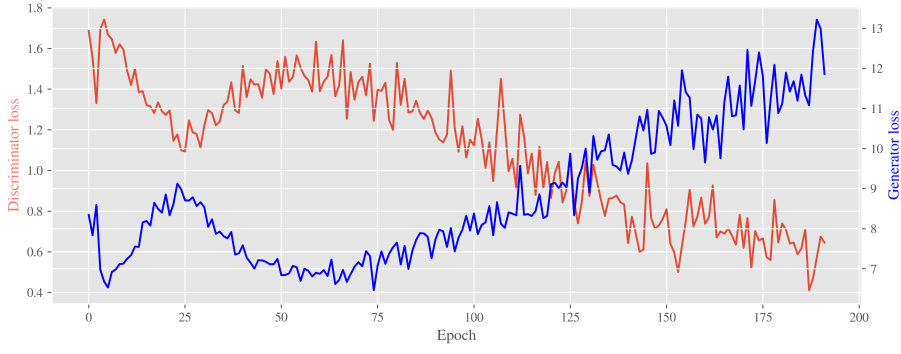

Fig. 3: DCGAN - Losses training on cifar10 over 190 epochs.

Losses can hardly be interpreted when treating with GANs, since the generator and discriminator are in a situation of competition where an improvement on the one leads to a deterioration on the other.

4.3 SN-DCGAN

We implement the spectral normalization layer and use on the discriminator of the DCGAN used in the previous section. We present the evolution of the inception score and losses in Figures 4 and 5, respectively.

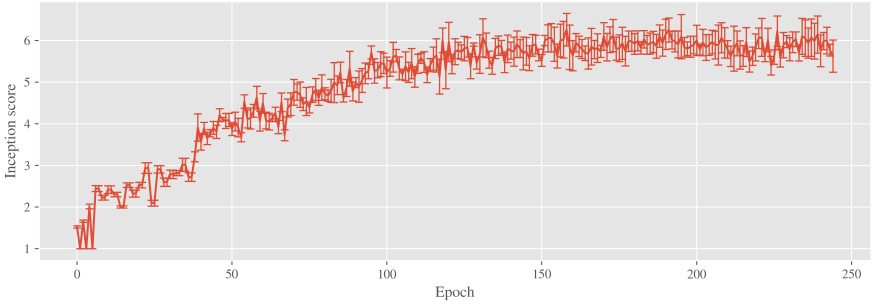

Fig. 4: SN-DCGAN - Inception score, training on cifar10 over 250 epochs

We don't notice any significant improvement on the mean of the inception score after convergence. However, the average standard deviation is reduced compared to that of the DCGAN without spectral normalization, depicted in Figure 10. In Figure 3, we observe that both losses are much smoother with

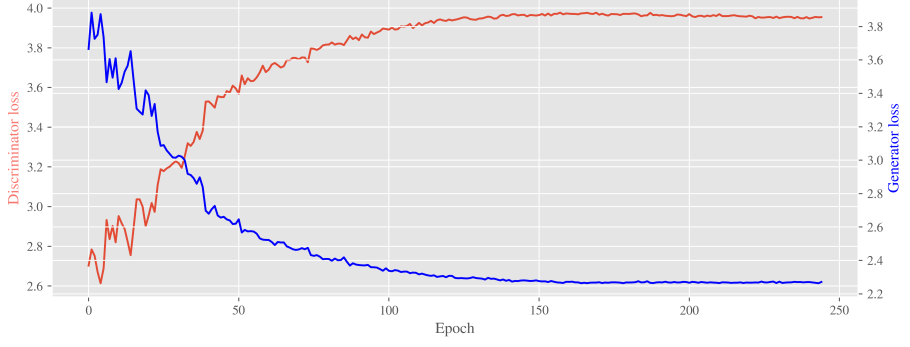

Fig. 5: SN-DCGAN - Losses training on cifar10 over 250 epochs.

spectral normalization. This is the desired result; the original intent of the paper on spectral normalization was to provide a solution to stabilize the training of GANs [**?**].

4.4 W-DCGAN

We present the evolution of the inception score and losses in Figures 6 and 7, respectively.

Fig. 6: W-DCGAN - Inception score, training on cifar10 over 250 epochs

Fig. 7: W-DCGAN - Losses training on cifar10 over 250 epochs.

4.5 W-SN-DCGAN

We present the evolution of the inception score and losses in Figures 8 and 9, respectively.

Fig. 8: W-SN-DCGAN - Inception score, training on cifar10 over 250 epochs

4.6 Performance Comparison

5 Conclusions

Explain what conclusions you can draw from these set of experiments? The set of experiments and results reported here should justify some of the design choices described in the previous sections. (3-6 pages)

Fig. 9: W-SN-DCGAN - Losses training on cifar10 over 250 epochs.

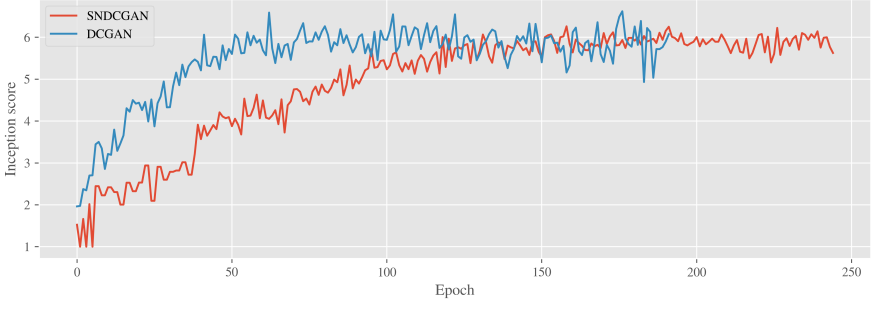

Fig. 10: All evaluated models - Inception score, training on cifar10