Deep Convolution GAN
Conditional Self-Attention Wasserstein GAN

Project 1
CSE676: Deep Learning(Fall 2019)

Monica Vashu Kherajani
Person #50290424
mkheraja@buffalo.edu

Abstract

The task of generating images from a particular distribution by learning the distribution is performed by generative adversarial networks(GAN). The scope of this project is limited to implementing two models of GAN, viz. the Deep Convolution GAN (DCGAN), Conditional Self-Attention Wasserstein GAN (CSAWGAN) for generating images from the CIFAR-10 dataset distribution.

1 Problem Statment

The objective of this project is to implement DCGAN and understand its short-comings and hence make improvements by adding on the top of DCGAN various concepts like wasserstein/ hinge loss, spectral normalization, conditional GAN and self-attention layers.

2 DCGAN

Deep Convolution GAN uses various properties of convolutional neural networks and modifies some of them as follows:
- It uses strided convolutions instead of max-pooling layer.
- It uses transposed convolution for upsampling.
- It eliminates the fully connected layers.
- It uses batch normalization.

2.1 Problems

However, DCGANs exhibits limitations while modelling long term dependencies for image generation tasks. The problem with DCGANs exists because model relies heavily on convolution to model the dependencies across different image regions. Since convolution operator has a local receptive field, long ranged dependencies can only be processed after passing through several convolutional layers. This could prevent learning about long-term dependencies due to:

2.1.1 Mode Collapse

If a generator produces an especially plausible output, the generator may learn to produce only that output. In fact, the generator is always trying to find the one output that seems most plausible to the discriminator.

If the generator starts producing the same output (or a small set of outputs) over and over again, each iteration of generator over-optimizes for a particular discriminator, and the discriminator never

manages to learn its way out of the trap. As a result the generators rotate through a small set of output types. This form of GAN failure is called mode collapse.

2.1.2 Trade off between representational capacity and computational efficiency

Increasing the size of the convolution kernels can increase the representational capacity of the network but doing so also loses the computational and statistical efficiency obtained by using local convolutional structure.

2.2 Code comments

2.2.1 Architecture and Parameters

The parameters have been borrowed from the DCGAN paper.

1. Generator CNN
   - Uses fours convolutional layers with kernel size as 4 and strides as 2.
   - Conv2DTranspose is used for upscaling.
   - All the convolutional layers use relu activation.
   - The last layer uses tanh activation.
   - Uses the learning rate as 0.0002 and β_1 as 0.5
2. Discriminator CNN
   - Uses fours convolutional layers with kernel size as 4 and strides as 2.
   - Conv2D is used for downscaling.
   - All the convolutional layers use leakyRelu activation.
   - The last layer uses sigmoid activation to generate the final classification result for the input image.
   - Uses the learning rate as 0.0002 and β_1 as 0.5

2.3 Experiments

2.3.1 Results

The best results are obtained from the above mentioned settings. Figure 1 shows the variation of Frechet Inception Distance (FID) with the number of epochs.

3 CSAWGAN

3.1 Solution to Mode Collapse

Wasserstein/ hinge loss: It prevents mode collapse by letting you train the discriminator to optimality even with vanishing gradients.

3.2 Spectral Normalization(SN)

Spectral normalization is added to the layers of the discriminator to stabilize its training. Applying SN helps to maintain the Lipschitz constant of the convolutional filters constraint.

3.3 Experience Replay

Due to the drawbacks of the other available methods to handle exploding and vanishing gradients like increased computational cost in case of gradient penalty and no guarantee of model convergence and lesser intuitive approach in case of weight clipping, using experience replay we can show previously generated samples to the discriminator and hence prevent the generator from easily fooling the discriminator.

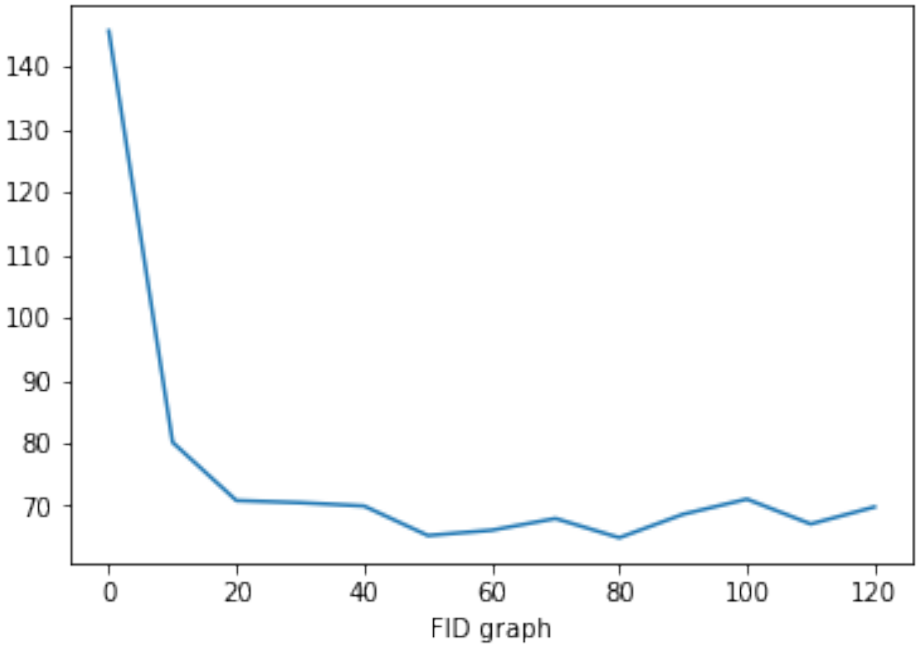

Figure 1: FID vs. Epochs

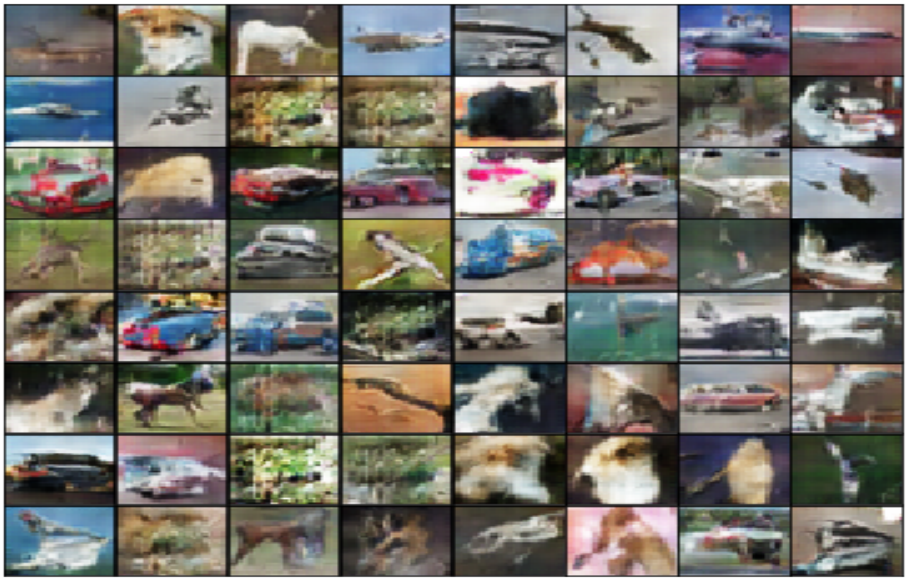

Figure 2: DCGAN final grid

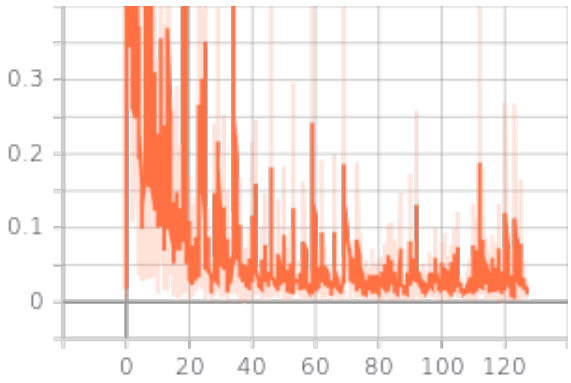

Figure 3: Discriminator loss on fake images for DCGAN

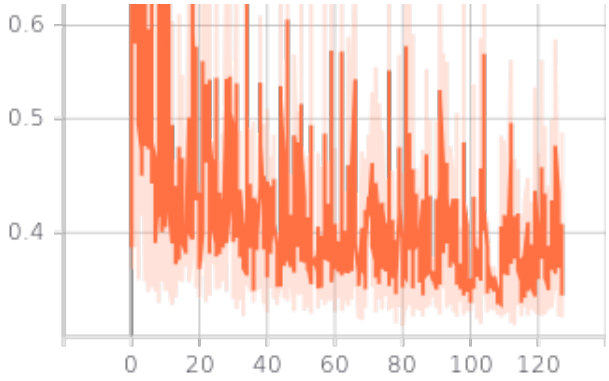

Figure 4: Discriminator loss on Real images for DCGAN

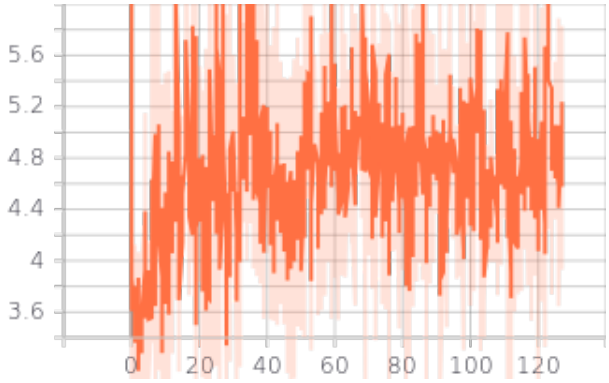

Figure 5: GAN loss for DCGAN

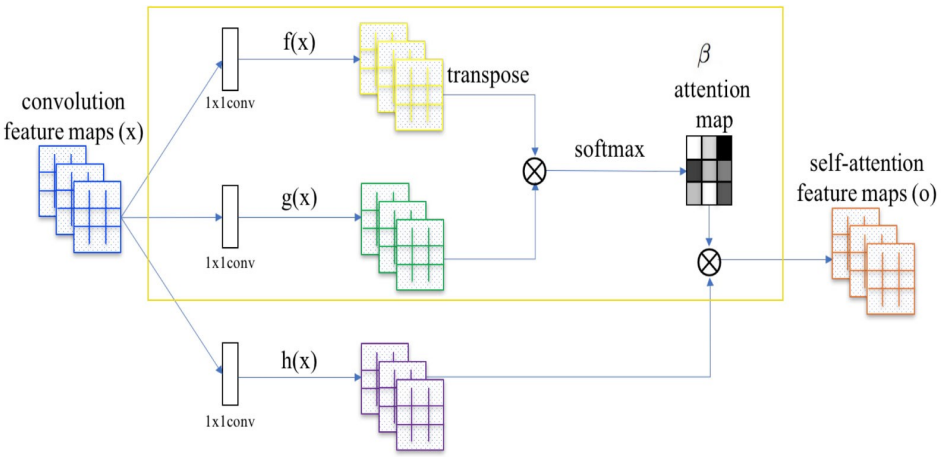

Figure 6: Self Attention

3.4 Self-Attention

Convolution operator has a local receptive field, long ranged dependencies can only be processed after passing through several convolutional layers. This could prevent learning about long-term dependencies as it increases the computation cost while increasing the representational capacity of the network. Self attention better balances between the ability to model long-range dependencies and the computational and statistical efficiency. The self-attention composes of

- Compute the attention map β
- Compute the self-attention output o

$$f(x) = W_f x$$

$$g(x) = W_g x$$

$$h(x_i) = W_h x_i$$

f(x) and g(x) are used to compute the attention map β, which has the following equation:

$$\beta_{j,i} = exps_{ij} / \sum i = 1^N exp(s_{ij})$$

and s_{ij} is the convolution of f(x) and g(x): $s_{ij} = f(x_i)^T g(x_j)$

Self attention is applied both to generator and discriminator CNN.

3.5 Code comments

3.5.1 Architecture

1. Generator CNN
   - Uses fours convolutional layers with kernel size as 4 and strides as 2 with spectral normalization.
   - ConvSN2DTranspose is used for upscaling.
   - All the convolutional layers use relu activation.
   - The last layer uses tanh activation.
   - Uses the Two time scale rule, learning rate as 0.0001 , β_1 as 0 and β_2 as 0.5
2. Discriminator CNN
   - Uses fours convolutional layers with kernel size as 4 and strides as 2 with spectral normalization.

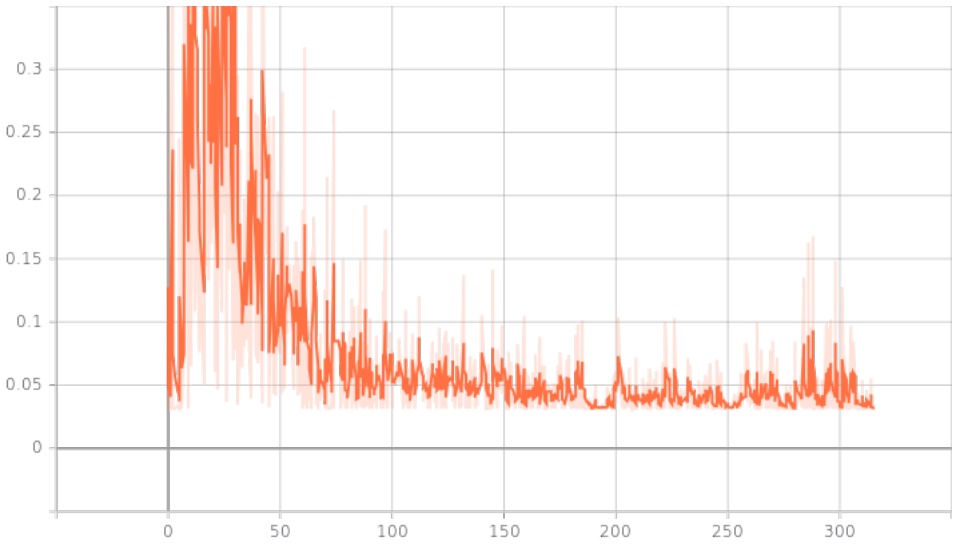

Figure 7: Discriminator loss on fake images for CSAWGAN

- ConvSN2D is used for downscaling.
- All the convolutional layers use leakyRelu activation.
- The last layer uses linear activation to generate the final classification result for the input image.
- Uses the learning rate as 0.0004, β_1 as 0.5 and β_2 as 0.5

3.6 Experiments

3.6.1 Parameters

The parameters have been borrowed from the SAGAN paper.

3.6.2 Results

The following indicates the best FID results obtained after training the models:

- DCGAN = 78.087
- CSAWGAN = 147.449

4 References

- Medium blog on DCGAN by Jonathan Hui
- Google developers GAN problems
- Medium post on training DCGAN on CIFAR10 by Utkarsh Desai

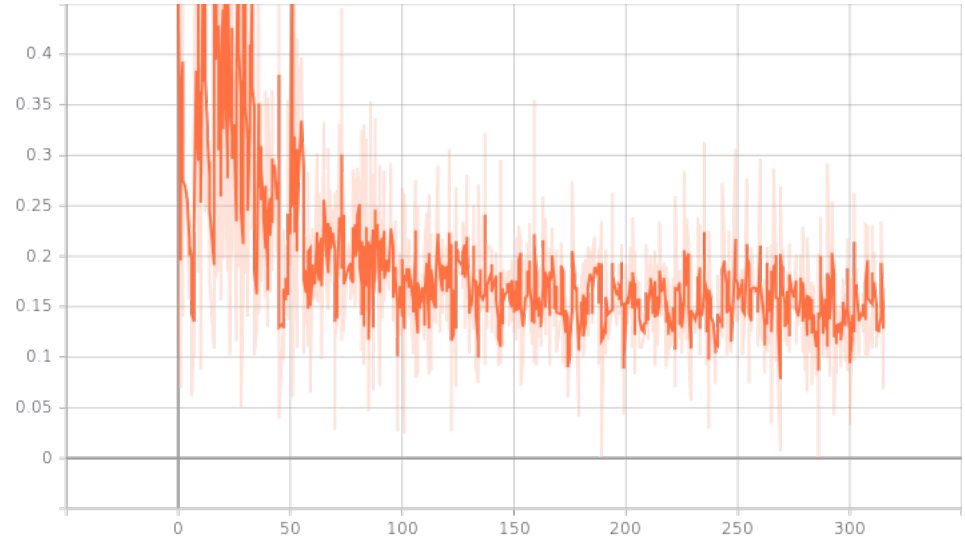

Figure 8: Discriminator loss on Real images for CSAWGAN

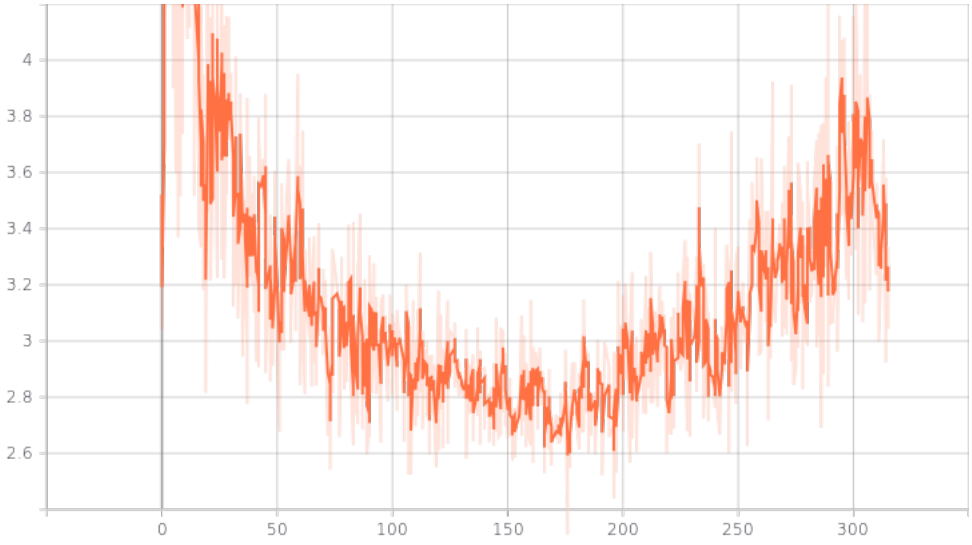

Figure 9: GAN loss for CSAWGAN

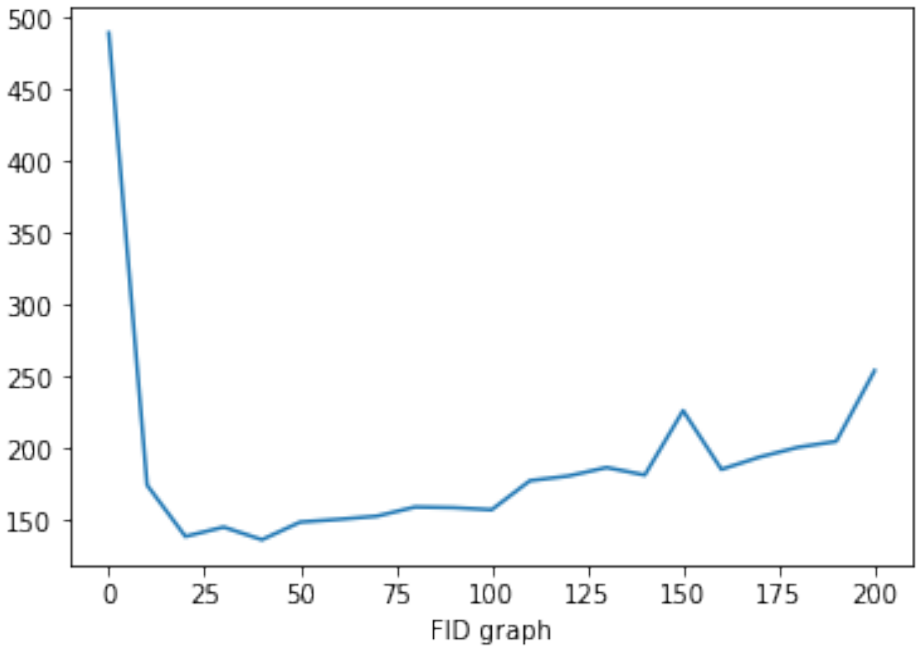

Figure 10: FID vs. Epochs

Figure 11: Self Attention final grid