IE 534: Deep Learning, Fall 2018

Homework 6: GANs on CIFAR10

Lecturer: Justin Sirignano *Submitted by: Rachneet Kaur, Oct 29th, 2018*

Implementation

The objective is to **train a discriminator and generator** pair to generate artificial data similar to the 32×32 dimensional coloured image from the corresponding one of the 10 mutually exclusive classes namely {airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck} in CIFAR10 data set. The CIFAR10 data set has 50000 coloured training images and 10000 images in the test set of dimensions $3 \times 32 \times 32$.

Model Architecture

I implemented a class *discriminator* to define the architecture for the discriminator model as mentioned in notes with Layer Normalization and Leaky ReLU implementations. The input image is $3 \times 32 \times 32$. The output for the discriminator model is a score determining whether the input is a real image from the data set or a fake image coming from the generator and an auxiliary classifier corresponding to the class label for the input image predicted by the discriminator architecture. My architecture has 8 convolution layers and 2 fully connected layers, layer normalization, max pooling and LeakyReLU activation.

- The code for training only the discriminator model is given in *OnlyDiscriminator.py*

I implemented a class *generator* to define the architecture for the generator model as mentioned in notes with batch Normalization and ReLU activation. The input image is a random noise vector and the output for the generator is the $3 \times 32 \times 32$ generated fake image resembling the corresponding the CIFAR10 image classes. My architecture has 5 convolution layers, 3 transposed convolution layers and one fully connected layer with batch normalization, ReLU activation and tanh activation for the last convolution layer.

- The code for training both the discriminator and generator model is given in *DiscriminatorAndGenerator.py*

```
Files already downloaded and verified
In epoch  0  the accuracy of the training set = 98.71995192307692
In epoch  1  the accuracy of the training set = 98.81610576923077
In epoch  2  the accuracy of the training set = 98.84415064102564
In epoch  3  the accuracy of the training set = 98.8877203525641
In epoch  4  the accuracy of the training set = 98.92988782051282
Accuracy on the test set =  87.00553797468355
```

Figure 1: Testing accuracy for the only discriminator trained model on 90 epochs

The generator model requires one forward and backward pass for both the generator and the discriminator. The discriminator model requires a forward pass through the generator and two forward and backward passes for the discriminator.
Hence,

- For implementation in Python, I used the Layer Normalization and LeakyReLU activation function for Discriminator.
- I trained the discriminator without generator model for 90 epochs.
- I used the Batch Normalization and ReLU activation function for Generator.
- I trained the discriminator with generator model for 200 epochs with 196 features.
- I used the batch size = 128.
- I used data augmentation techniques in the training set.
- I used ADAM optimizer to minimize the cross entropy loss function.
- As in Wasserstien GAN implementation, gradient penalty was used in the discriminator loss function during optimization.
- As in Auxiliary Classifier GAN, some information about the class labels is provided to the generator to generate images from a specific class.

A snapshot of the training and test accuracy's for the discriminator model trained without the generator after training for 90 epochs is given in Figure 1.

Note that the accuracy on the training set after 90 epochs reaches approx. 98.92, but clearly since the test accuracy is 87.005, we can notice that the model **overfit** the training set.

```
86 150 0.12 -1.83 -1.43 0.83 0.22 71.19
86 200 0.12 -1.83 -1.43 0.83 0.22 71.21
86 250 0.12 -1.84 -1.43 0.83 0.22 71.23
86 300 0.12 -1.84 -1.43 0.83 0.22 71.24
86 350 0.12 -1.84 -1.43 0.83 0.22 71.26

In epoch  86  the accuracy of the training set = 71.2730483716475

In epoch  86 the testing accuracy is  80.3995253164557
Time to train this epoch of model is: 67973.55388188362
87 0 0.12 -1.84 -1.44 0.83 0.22 71.27
87 50 0.12 -1.84 -1.44 0.82 0.22 71.29
87 100 0.12 -1.84 -1.44 0.82 0.22 71.31
87 150 0.12 -1.84 -1.44 0.82 0.22 71.33
87 200 0.12 -1.84 -1.44 0.82 0.22 71.35
87 250 0.12 -1.84 -1.44 0.82 0.22 71.36
87 300 0.12 -1.84 -1.44 0.82 0.22 71.38
87 350 0.12 -1.84 -1.44 0.82 0.22 71.40

In epoch  87  the accuracy of the training set = 71.40800280448718

In epoch  87 the testing accuracy is  80.82476265822785
Time to train this epoch of model is: 68755.0385465622
88 0 0.12 -1.84 -1.44 0.82 0.22 71.41
88 50 0.12 -1.84 -1.44 0.82 0.22 71.42
88 100 0.12 -1.85 -1.44 0.82 0.22 71.44
88 150 0.12 -1.85 -1.44 0.82 0.22 71.46
88 200 0.12 -1.85 -1.44 0.82 0.22 71.47
88 250 0.12 -1.85 -1.45 0.82 0.22 71.49
88 300 0.12 -1.85 -1.45 0.82 0.22 71.51
88 350 0.12 -1.85 -1.45 0.82 0.22 71.52

In epoch  88  the accuracy of the training set = 71.53828147507923

In epoch  88 the testing accuracy is  81.14121835443038
Time to train this epoch of model is: 69536.57416725159
```

Figure 2: Testing accuracy for the discriminator with generator trained model on 200 epochs

A snapshot of the training and test accuracy's for the discriminator model trained with the generator after training for 200 epochs is given in Figure 2.

Note that the accuracy on the training set after 200 epochs reaches approx. 71.5, but clearly since the test accuracy is 81.14, we can notice that the model performs better on the test set. Also the accuracy is worse than the only discriminator model.

- The code for generating the visualization results based on the trained models is given in *Visualization.py*

Results

Part 1: Training the models

Discriminator without the generator

The code for training the discriminator without generator model is given in *OnlyDiscriminator.py*
The results for the discriminator trained model without the generator are:

- The accuracy on the training set with only discriminator trained model = 98.92%
- The accuracy on the test set with only discriminator trained model = 87.005 %
- The estimated time to train the model for 90 epochs is about 2 hours.

Discriminator with the generator

The code for training both the discriminator and generator model is given in *DiscriminatorAndGenerator.py*
The results for the discriminator trained model with the generator are:

- The accuracy on the training set of discriminator trained model with generator = 71.5%
- The accuracy on the test set of discriminator trained model with generator = 81.14 %
- The estimated time to train the model for 200 epochs is about 45 hours.

Looking at generated images for discriminator trained with the generator

Below are the generated images by the generator trained along with a discriminator at Epoch 0 (Figure 3), Epoch 20 (Figure 4), Epoch 50 (Figure 5), Epoch 70 (Figure 6), Epoch 100 (Figure 7) and Epoch 120 (Figure 8)

Part 2: Visualization Results

The code for generating the visualization results based on the trained models is given in *Visualization.py*

Figure 3: Generated Images at **Epoch 0**

Perturb Real Images

- Classification accuracy for only discriminator trained model on a batch of real images is 92.1875%
- Classification accuracy for only discriminator trained model on a batch of fake images is 17.96875%

Figure 9 is a batch of real images.
Figure 10 is the batch of gradients to perturb the real images and Figure 11 is the batch of the perturbed images based on the real images and the gradients.

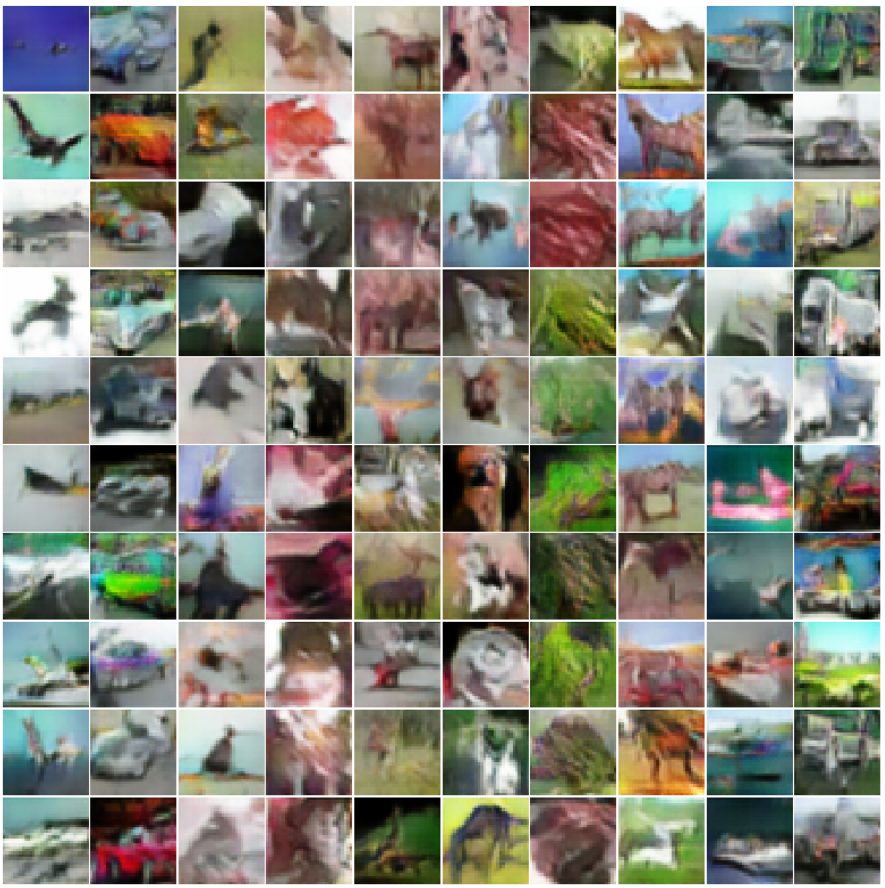

Figure 4: Generated Images at **Epoch 20**

Synthetic Images Maximizing Classification Output

Figure 12 denotes the synthetic images maximizing class output for discriminator trained without the generator.

Figure 13 denotes the synthetic images maximizing class output for discriminator trained with the generator.

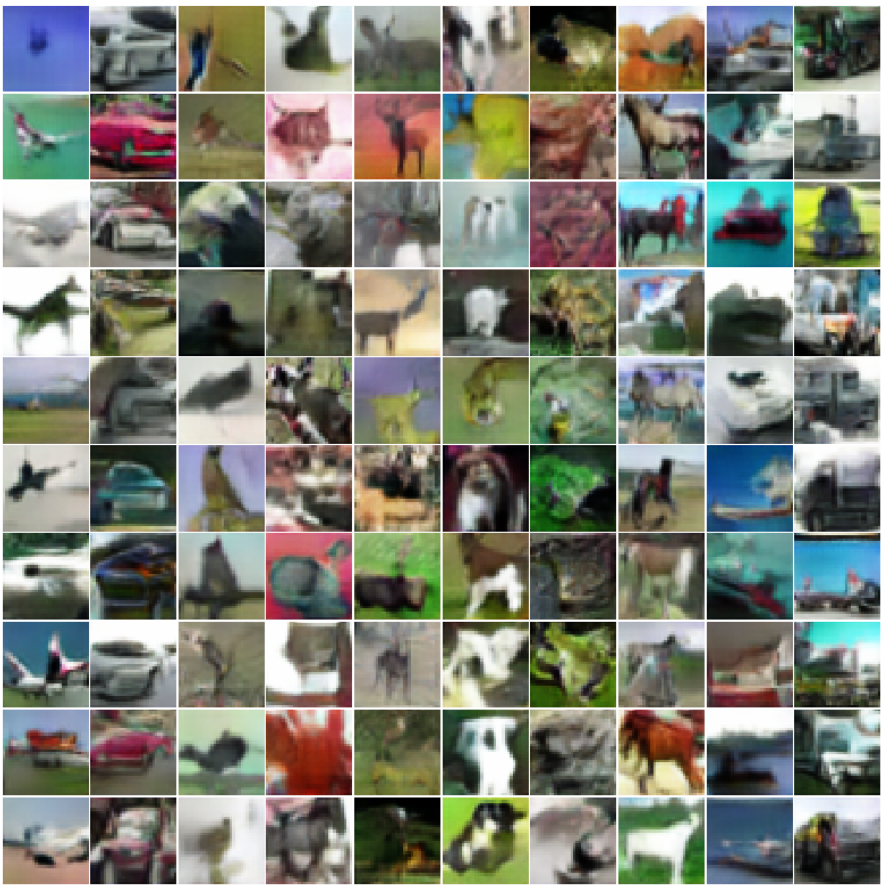

Figure 5: Generated Images at **Epoch 50**

Synthetic Features Maximizing Features at Various Layers

Figure 14 and 15 denotes the synthetic images maximizing layer 4 and layer 8 features for discriminator trained without the generator respectively.

Figure 16 and 17 denotes the synthetic images maximizing layer 4 and layer 8 features for discriminator trained with the generator respectively.

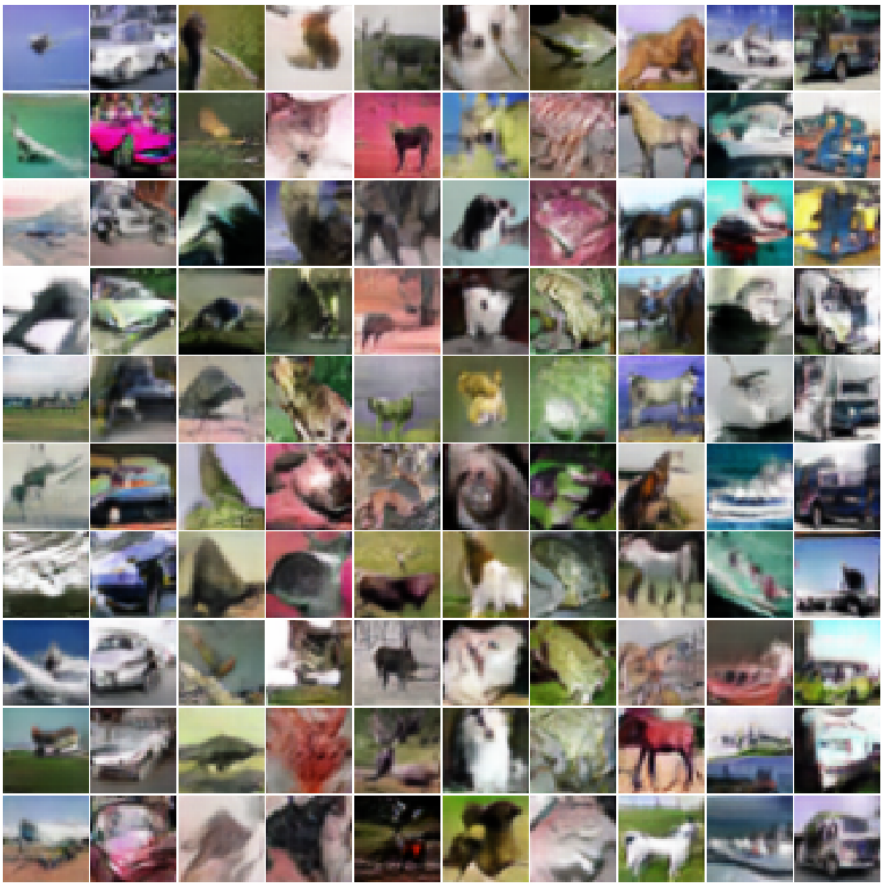

Figure 6: Generated Images at **Epoch 70**

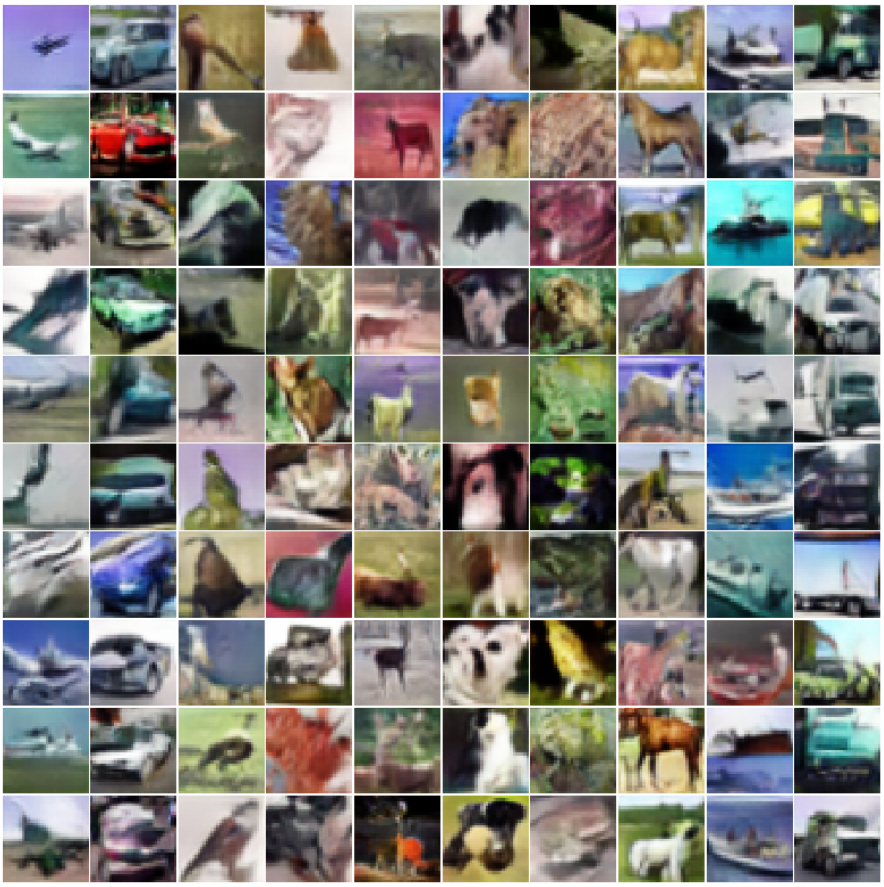

Figure 7: Generated Images at **Epoch 100**

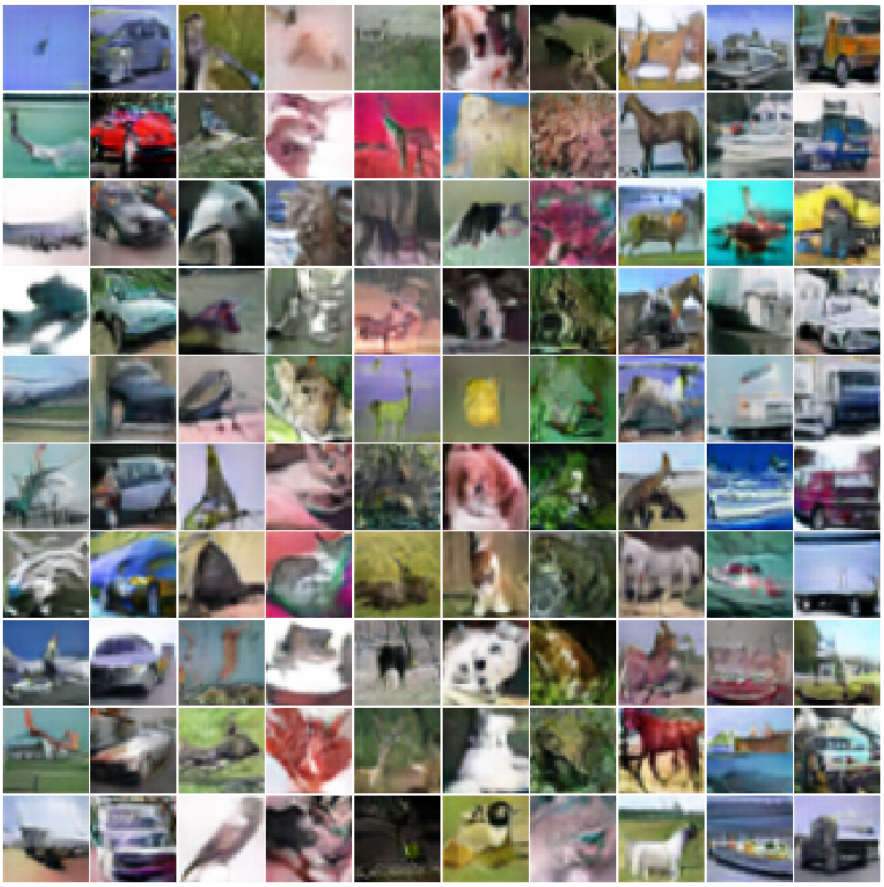

Figure 8: Generated Images at **Epoch 120**

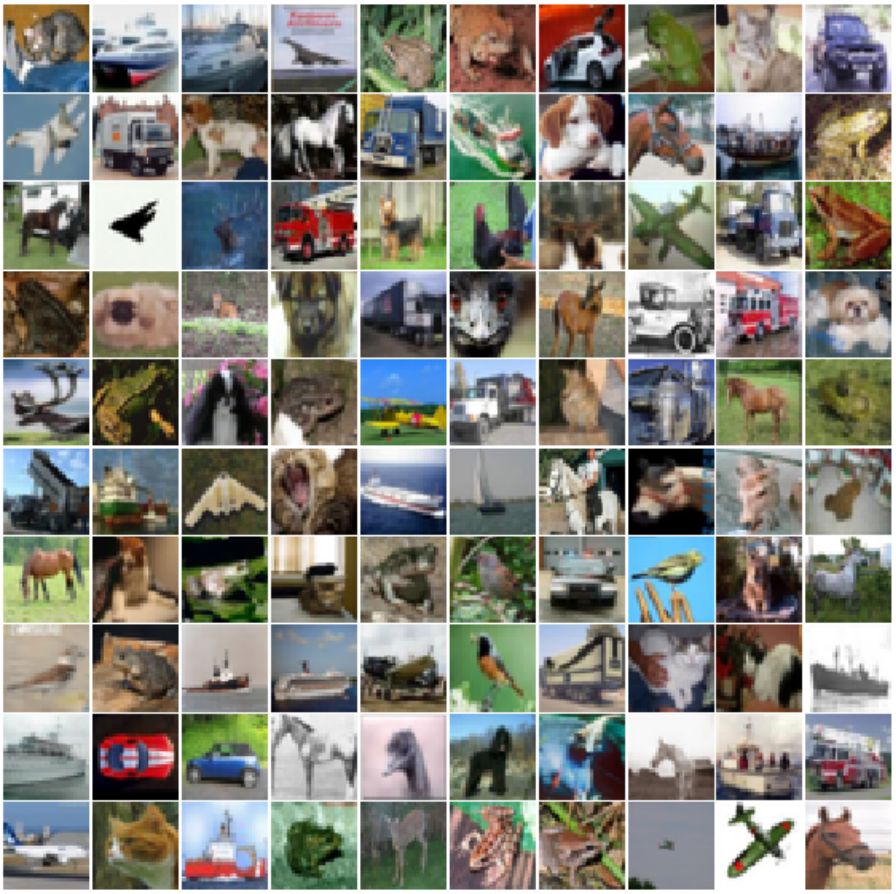

Figure 9: Real Images

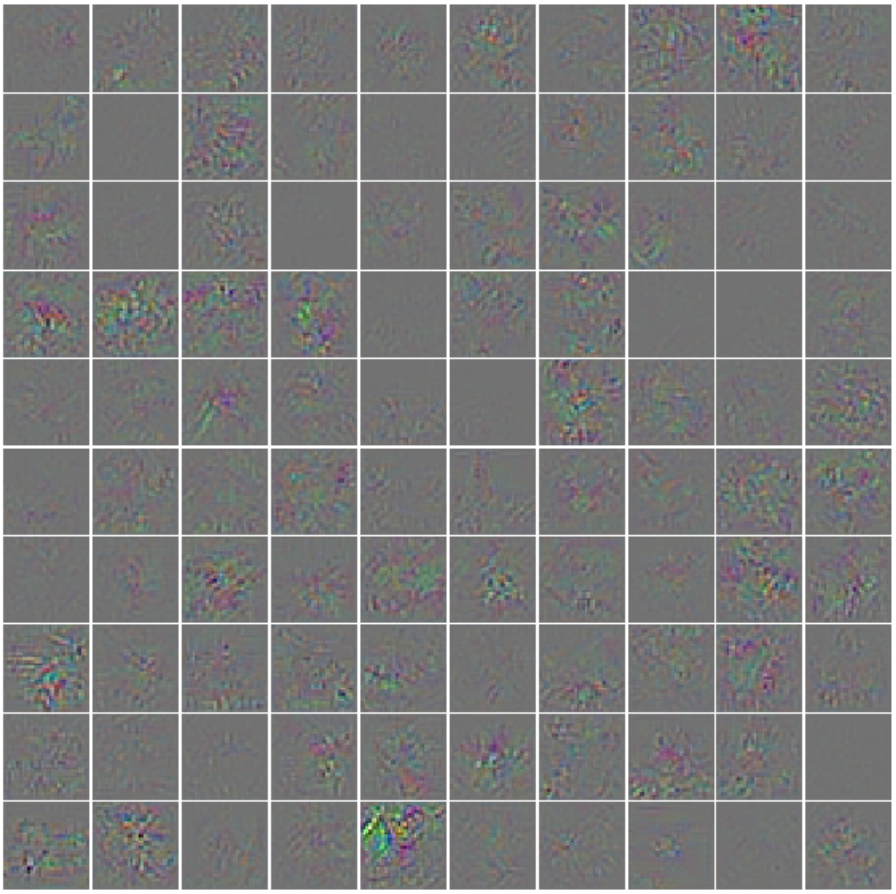

Figure 10: Gradients

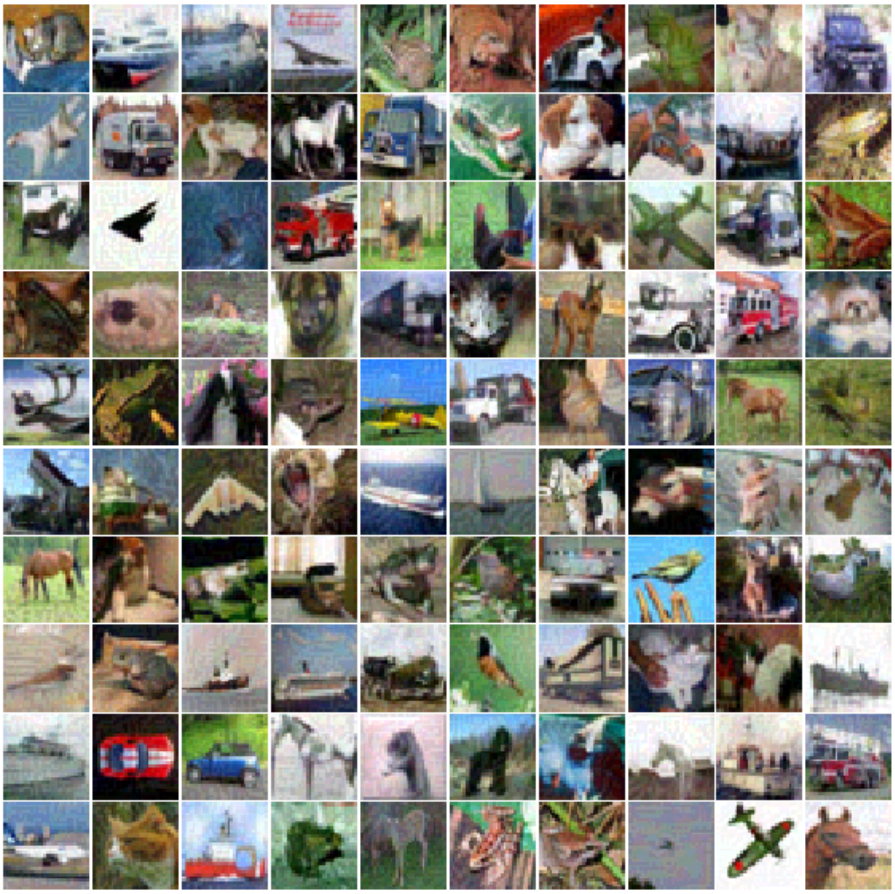

Figure 11: Altered Images

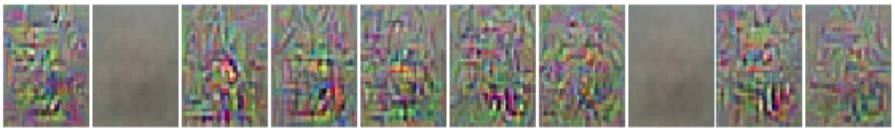

Figure 12: Max class for discriminator trained without generator

Figure 13: Max class for discriminator trained with generator

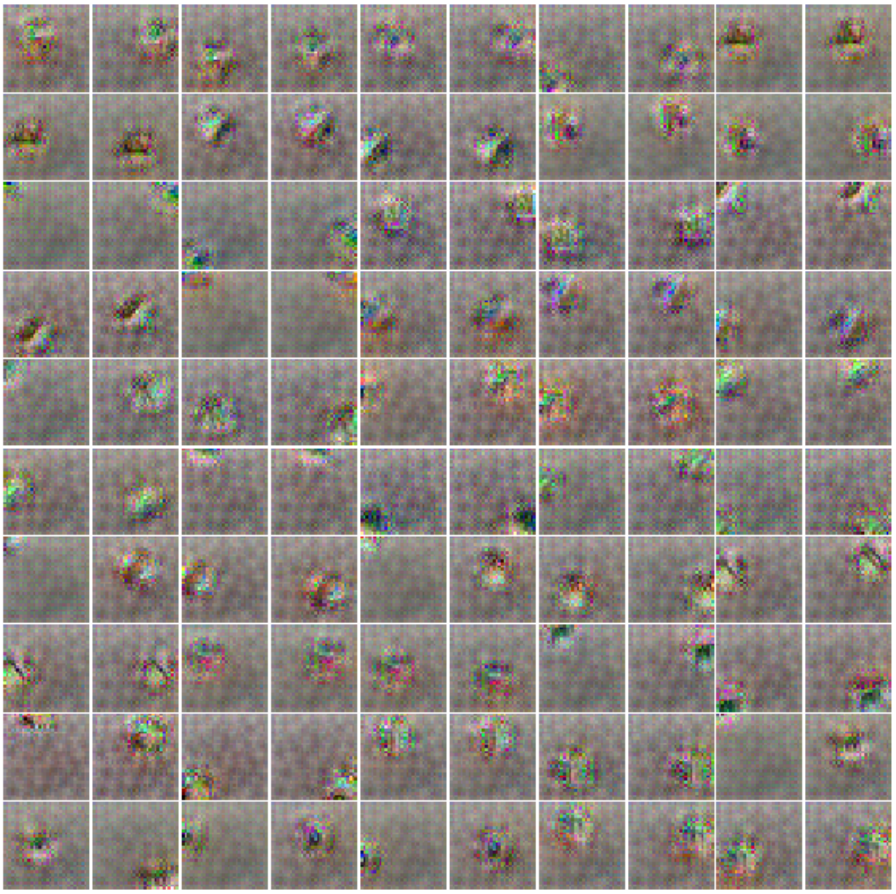

Figure 14: Max features at layer 4 for discriminator without generator

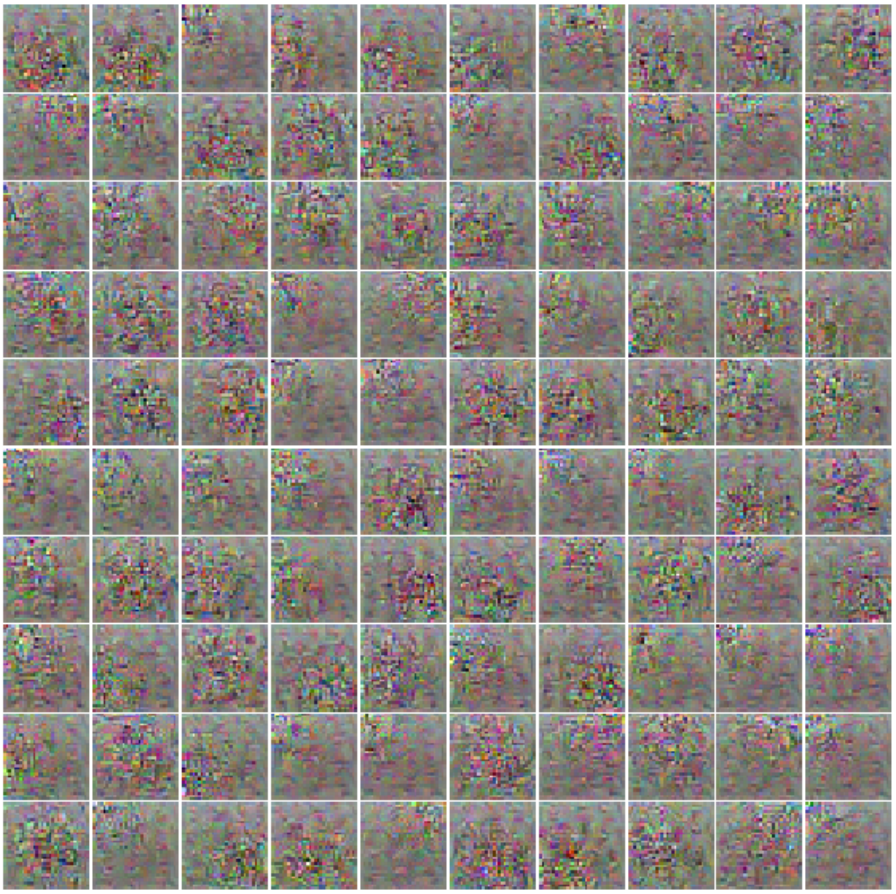

Figure 15: Max features at layer 8 for discriminator without generator

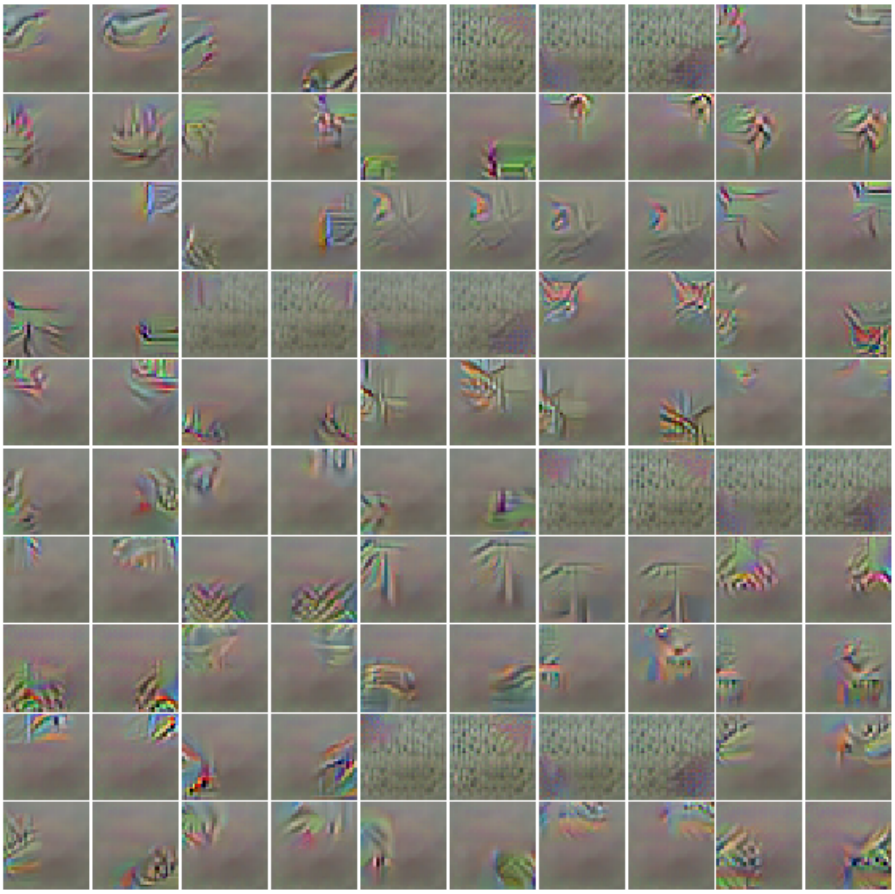

Figure 16: Max features at layer 4 for discriminator with generator

Figure 17: Max features at layer 8 for discriminator with generator