CMSC 25040: Homework 4 Write-Up

Kameel Khabaz

March 3, 2023

The objective of this homework was to train a Generative Adversarial Network (GAN) for generating images in the Fashion-MNIST dataset.

1 Model Design

The model design is based off of the Deep Convolutional Generative Adversarial Network (DCGAN) architecture.

The discriminator contains several convolutional blocks. Each convolutional block consists of a 2D convolutional layer, a 2D batch normalization layer, and a leaky ReLU layer (negative slope of 0.2).

The discriminator is designed to take advantage of the training data's structure that contains class classification information. As such, it outputs a 1×1 binary classification for a real/fake image in addition to a linear output 1×10 multi-class classification for the item type. The discriminator has the following structure:

1. 2D convolution that increases the number of channels to the image width with a kernel of 4, a stride of 2, and a padding of 1. This is followed by a leaky ReLU layer (negative slope is 0.2 for all leaky ReLUs).

2. Convolutional block doubling the number of channels with a kernel of 4, stride of 2, and padding of 1.

3. Convolutional block doubling the number of channels with a kernel of 3, stride of 2, and a padding of 1.

4. 2D convolution that doubles the number of channels with a kernel of 4, stride of 2, and a padding of 1.

5. The network now splits. One branch obtains a binary classification and the other branch obtains a multi-class classification of the item category.

    - For real/fake binary classification, we use a 2D convolutional layer that decreases the number of channels to 1 and that has a kernel of 2, a stride of 1, and a padding of 0. A sigmoid function is then applied to the output.

- For multi-class classification, a flattening layer is used and is then followed by a linear layer projecting the data into a 10-dimensional output.

The generator model creates a 28×28 image from a linear input vector, represented as a 1×1 image with 100 channels. It consists of a series of decoder blocks borrowed from my image segmentation model in the previous assignment. A decoder block here consists of a 2D convolution transpose layer, a batch normalization layer, and a leaky ReLU layer (with a negative slope of 0.2).

The model is designed as follows:

1. A decoder block that increases the number of channels to 8 times the image size with a kernel size of 2, stride of 1, and a padding of 0. The image output size is 2×2.

2. A decoder block that decreases the number of channels to 4 times the image size with a kernel size of 4, stride of 2, and padding of 1. The image output size is 4×4.

3. A decoder block that decreases the number of channels to 2 times the image size with a kernel size of 3, stride of 2, and padding of 1. The image output size is 7×7.

4. A decoder block that decreases the number of channels to the image size with a kernel size of 4, stride of 2, and padding of 1. The image output size is 14×14.

5. A 2d convolution transpose layer that decreases the number of channels to 1 (for a grayscale image) and has a kernel of 4, stride of 2, and padding of 1. This makes the final image size of 28×28. The output is normalized using a sigmoid activation function.

Weights for the convolutional layers are initialized to a normal distribution with a mean of 0 and a standard deviation of 0.02. Weights for the batch normalization layers are initialized to a normal distribution with a mean of 1 and a standard deviation of 0.02 (and the bias is initialized to 0).

I use a binary cross entropy loss function for discriminating between real and generated images and a cross entropy loss function for the categorical multi-class classification. I also use an Adam optimizer with learning rates $\beta_1 = 0.5$ (which I observed worked better than the default value of 0.9) and $\beta_2 = 0.999$ (the default value). The learning rate is 0.0002.

2 Model Training

When training the model, I first update the discriminator network. A real image is fed through the discriminator network, and the loss is calculated as the sum of the binary classification loss and the multi-class classification loss (corresponding to the image's category in the Fashion-MNIST dataset). This is because we want the discriminator to label real images as such and to classify them correctly.

Then, the generator creates a fake image from noise, and the fake image is fed through the discriminator. The discriminator loss is similarly calculated as the sum of the binary classification

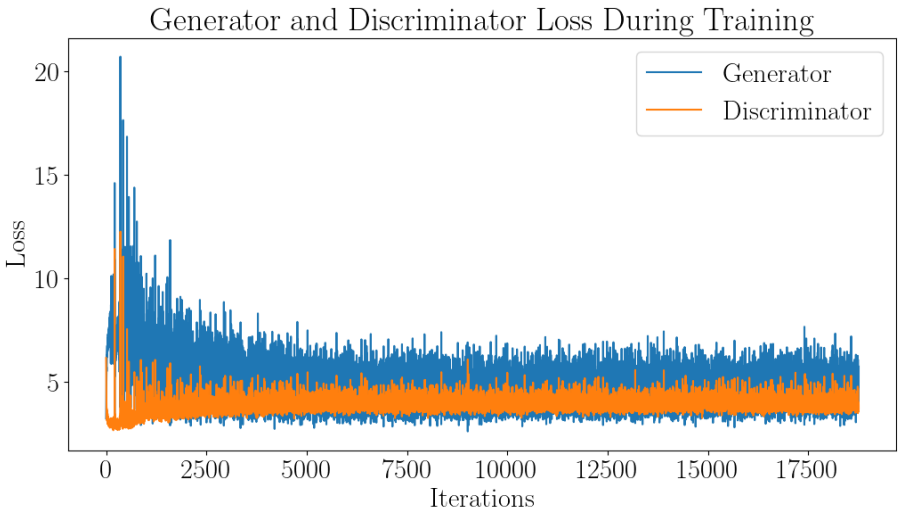

Figure 1: Generator and discriminator losses during training

loss and the multi-class classification loss. In this case, the ground truth class label for a fake image is a randomly drawn integer from a uniform distribution of 0 to 9 because we want the generator to produce random and uncertain class classifications for fake images in addition to labeling them as fake.

Then, we update the generator network, in which the loss is calculated from the discriminator's classification for an image that is generated from random noise. The generator loss is equal to the sum of the binary classification loss and multi-class classification loss (with the random ground truth class value).

3 Model Results

The generator and discriminator losses are shown in Figure 1. The cyclic nature of the losses demonstrates the adversarial nature of the GAN and is an indicator of successful training of the model.

A comparison of fake generated images at the end of model training and real images from the Fashion-MNIST dataset are shown in Figure 2. As shown, the generated images are visually very similar to the real images, except for a slightly lower level of detail and lower quality for the items with more complex shapes (such as the high-heeled shoes).

Finally, Figure 3 shows images of the generated outputs at different epochs of training. As expected, there is clear improvement in the quality of generated images throughout the training phase, indicating that the model is becoming better and better at generating realistic images.

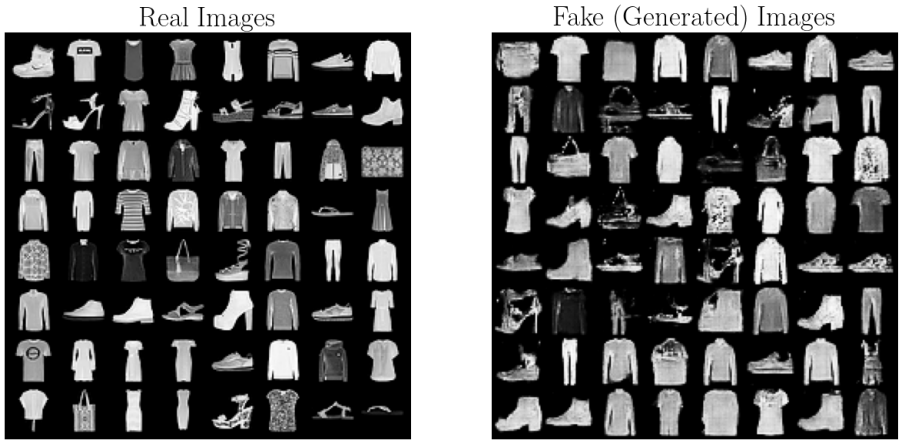

Figure 2: Comparison of real vs. generated images

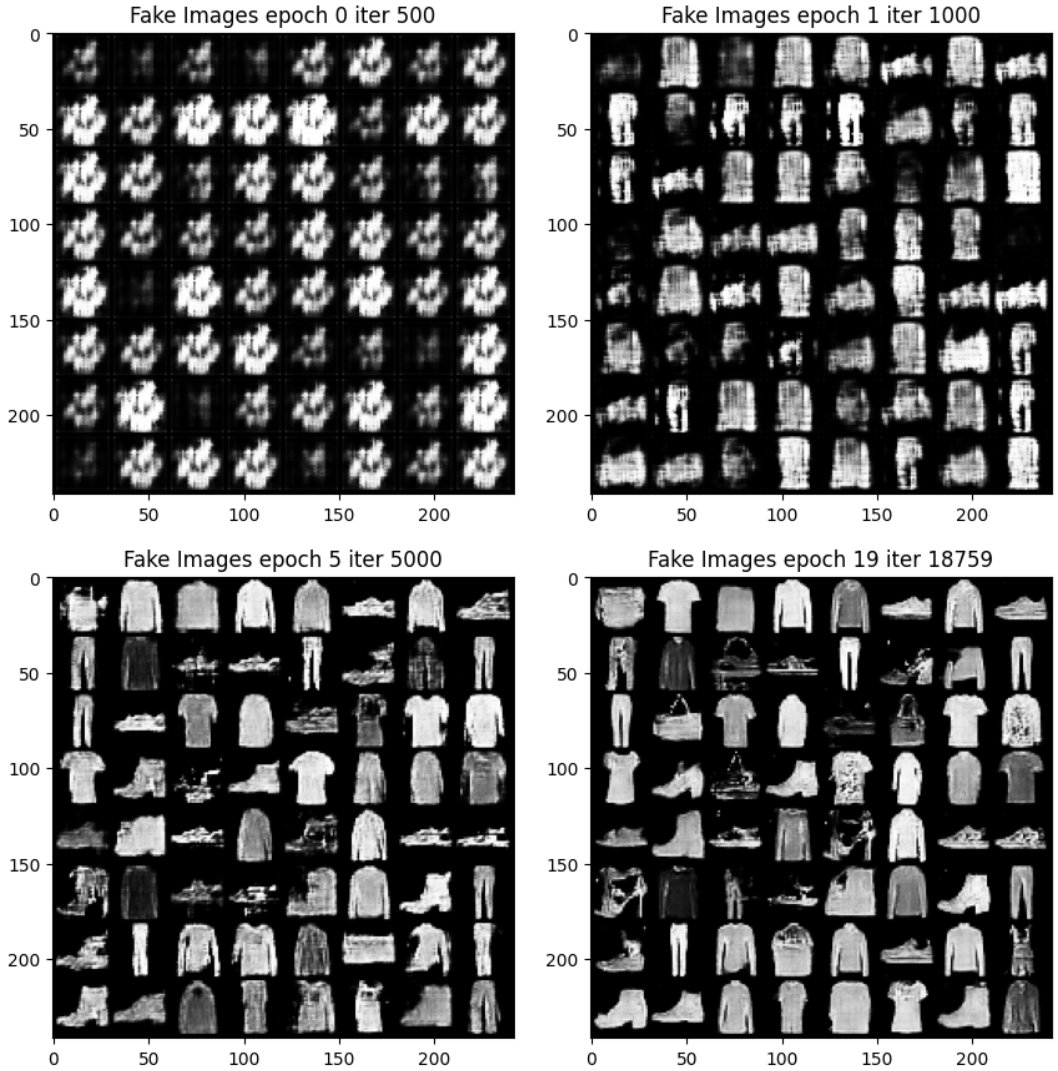

Figure 3: Generated images during training

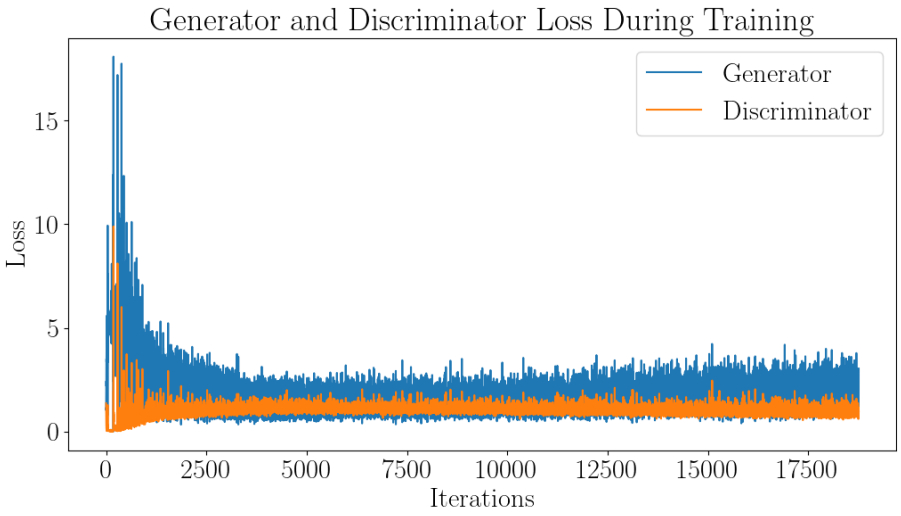

Figure 4: Generator and discriminator losses during training (no multi-class classification)

To investigate the impact of the multi-class classification loss component on the generated images, I trained a second version of the model with that component removed (so the only loss comes from the binary real/fake classification). The source code for this model is found in the `gan_noclass.ipynb` file, while the `gan.ipynb` file has the source code for the full model (with the multi-class classification according to category labels).

For this modified model, the generator and discriminator losses throughout the training period are shown in Figure 4. A comparison of real vs. fake images is shown in Figure 5, and generated images throughout training are shown in Figure 6. At first glance, the results for the two models seem similar in that the fake images appear fairly realistic. However, if one looks closely at the results in Figure 5, it becomes apparent that the model trained without the mutli-class classification often produces generated images that have a dark black circle in the middle of the image. I do not see this artifact in the images for the generator trained using the multi-class classification loss in Figure 2. So, adding that second component to the loss function seems to improve the quality of the generated images.

4 References

[1] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks." arXiv, Jan. 07, 2016. Accessed: Mar. 02, 2023. [Online]. Available: http://arxiv.org/abs/1511.06434

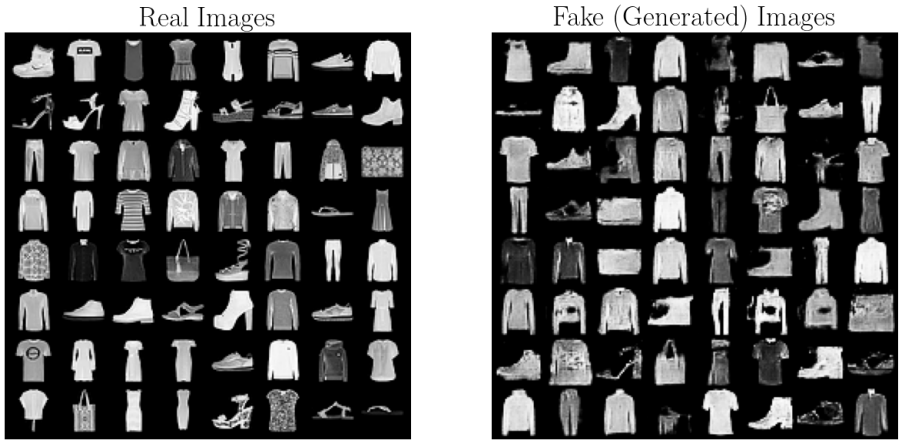

Figure 5: Comparison of real vs. generated images (no multi-class classification)

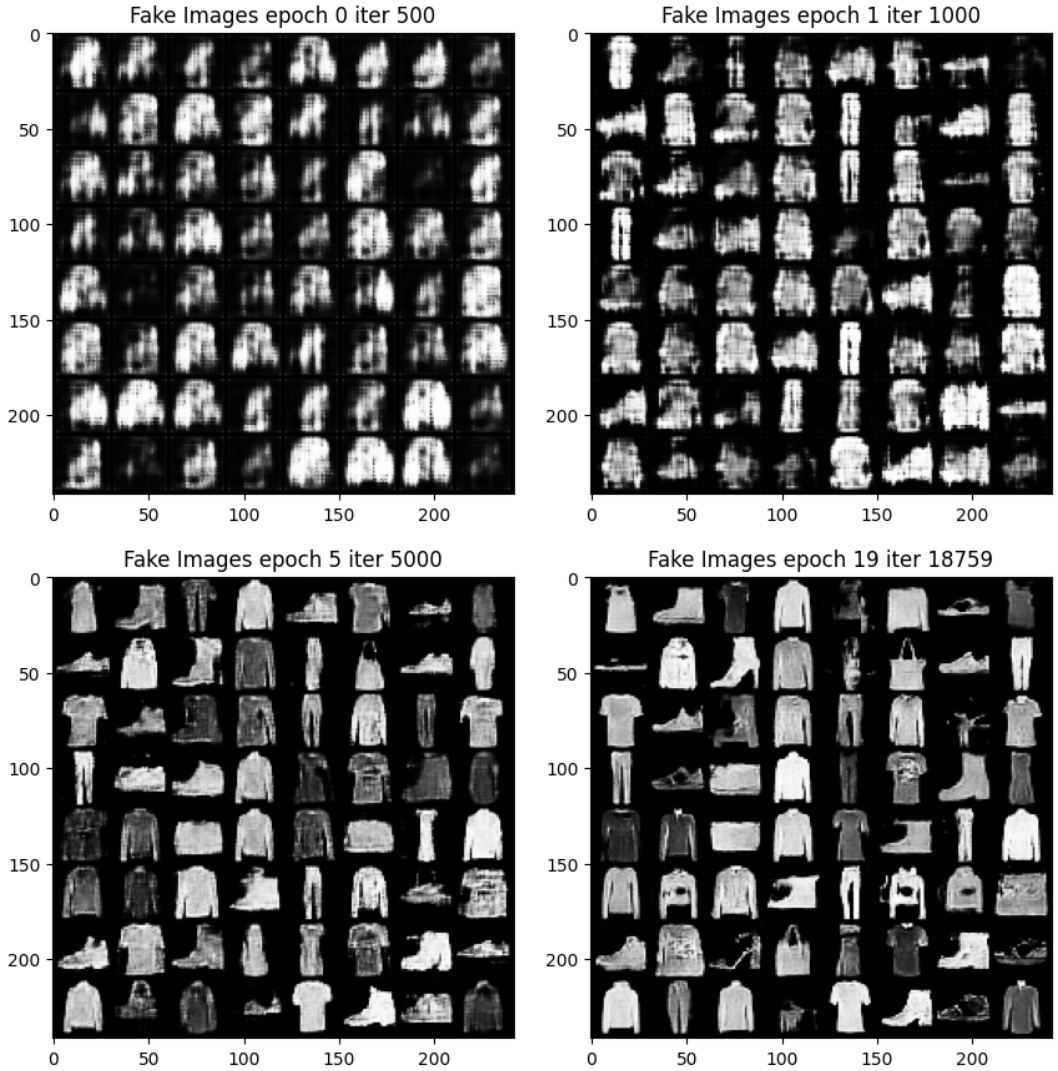

Figure 6: Generated images during training (no multi-class classification)