Generative adversarial networks, or GANs, are an unsupervised machine learning model that are used to generate data that imitates true data from random inputs. The model consists of two adversarial neural networks, a generator and a discriminator, that partake in a minimax game where the discriminator is trying to maximize the following objective function and the generator is trying to minimize it (Gandhi, 2018).

$$\mathbb{E}_{x\sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z\sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

In this objective function, $D_{\theta d}$ represents the discriminator's output for data while $G_{\theta g}$ represents the generator's output for data. In order for the discriminator to maximize the objective function, $D_{\theta d}$ would correctly classify all real data as genuine (value of 1) and all generated data as fake (value of 0). To minimize the objective function, the generator wants $D_{\theta d}$ to incorrectly classify all generated data as real (value of 1) (Gandhi, 2018). Thus, a GAN can be trained by alternating between trying to improve the discriminator's ability to classify data as real or fake and trying to improve the generator's ability to create fake outputs that appear real.

For this project, a GAN was developed in order to create fake images of cats' faces using data from a publicly available dataset containing approximately 16,000 64x64 pixel images of cats' faces (Ghosh, 2019). Since GANs are an unsupervised learning model, there does not need to be any classification of the data as each image in the dataset is labeled as real. The GAN was developed using Pytorch. The discriminator consists of four convolutional layers each with batch normalization and a leaky ReLU activation function with a negative slope of 0.2 and a fifth convolutional layer with a sigmoid activation function. On the other hand, the generator consists of four transposed convolutional layers each with batch normalization and a ReLU activation function and a fifth transposed convolutional layer with a hyperbolic tangent activation function (Rao, 2021).

As previously mentioned, the two neural networks perform directly opposite actions. The discriminator takes a 64x64 pixel image with three feature maps and increases the number of feature maps while decreasing the size of each feature map by using a stride of two. Finally, the last convolutional layer reduces the number of output feature maps to one with a size of

one-by-one. This last value is then used to classify the original input image as real or fake. Conversely, the generator takes in random noise values and using transposed convolutional layers, converts each noise value into three output feature maps of size 64x64, the original dimensions of images in the dataset. Since the two networks are working adversarially, it makes intuitive sense that the networks would work in opposite manners as explained above.

In order to gauge results of the network, there are noise tensors set at the beginning of run time that are input to the generator after each epoch of training to see the resulting images. This provides a visual representation of the learning inside the network and assures proper functionality of the network. As for quantitative results, the loss for both the discriminator and generator have been recorded. Between these two data records, one can detect the learning and see possible areas for improvement in the future development of this GAN.

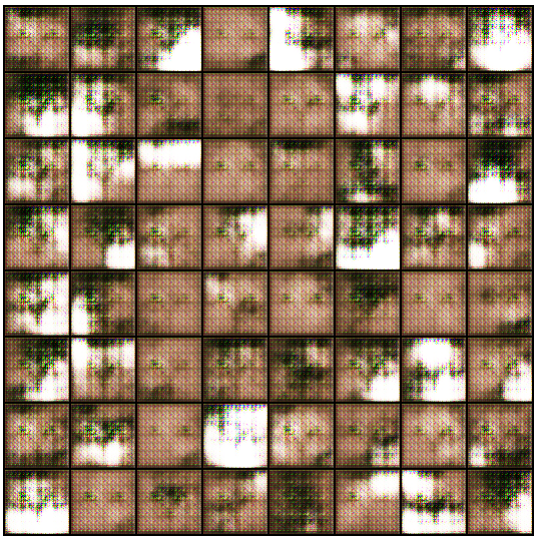
For this project, the GAN was trained over the entirety of the 15,737 cat images in the dataset for 24 epochs using mini-batch sizes of 128. With each epoch taking approximately twenty minutes, the model trained for around eight hours. Below are tables of the loss for the two networks for each epoch:

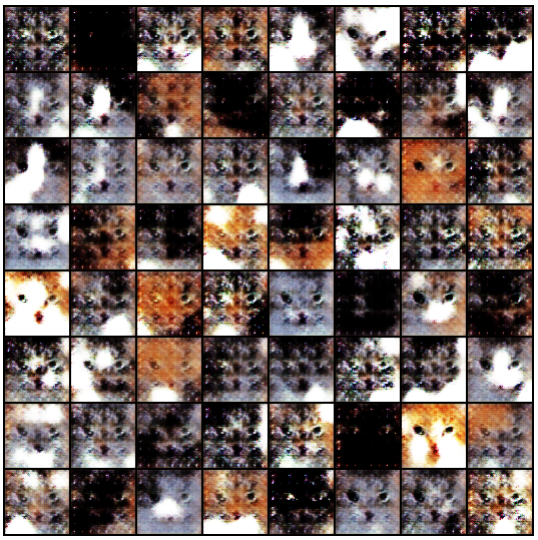
Epoch	Discriminator Loss	Generator Loss
1	110.514	638.390
2	131.743	479.310
3	105.886	477.438
4	101.760	541.392
5	87.434	585.805
6	86.375	593.712
7	96.993	633.227
8	93.374	609.369
9	79.677	621.225
10	105.565	630.287
11	87.627	636.327

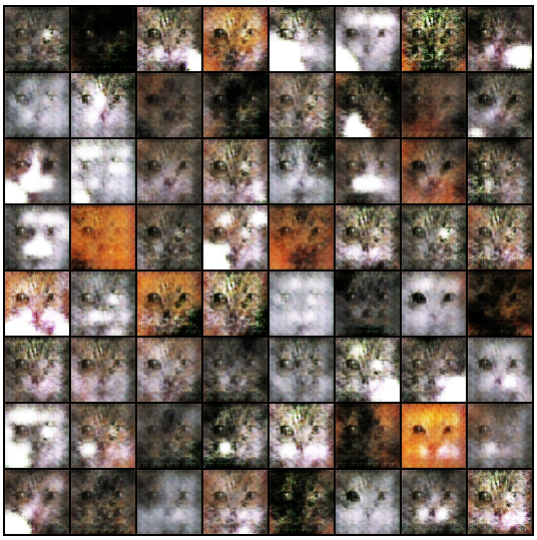
Epoch	Discriminator Loss	Generator Loss
12	68.275	662.541
13	104.161	666.865
14	77.761	641.402
15	98.203	646.037
16	86.136	660.390
17	83.454	673.978
18	78.819	736.572
19	76.196	732.100
20	72.117	700.044
21	74.171	756.558
22	63.353	777.212
23	76.892	793.240

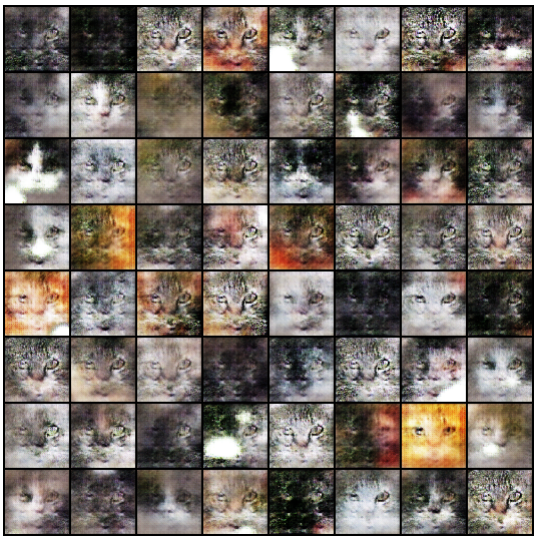
The data shows the loss for the discriminator generally decreasing while the loss of the generator generally increases. This result shows the discriminator is improving at determining when images are real and when they are generated. Since the generator is trained by feeding improper labels to discriminator with generated images, the increase in loss can be interpreted as the discriminator improving its ability to separate real from fake images. If the model was trained for more epochs, the generator loss would be expected to decrease as the generator would create fake images that deceive the discriminator.

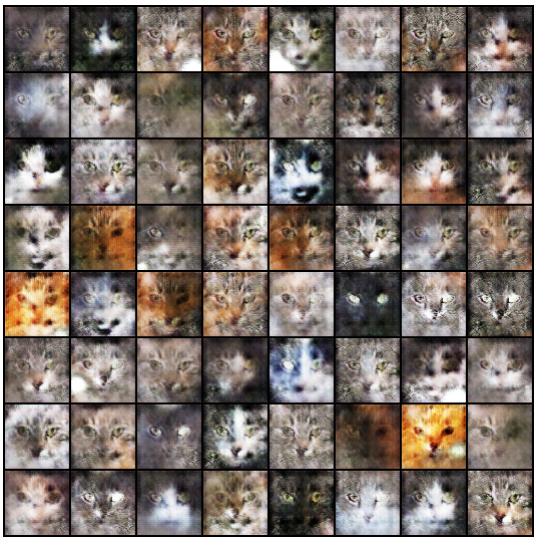
Viewing the images output by the generator after each epoch is an easier way to see the results of the adversarial learning in the model. Below are five graphs showing the output of the generator on the same 64 random noise tensors throughout the course of the model's training (Rao, 2021).











These images clearly show improved image generation of cats' faces. First, the model seems to learn the silhouette of the cat's head, then major features such as the eyes and nose, and finally attempts to learn the textures of fur and different fur patterns. Although it cannot easily be determined what the generator or discriminator is learning, by producing images after each epoch, one can clearly identify areas of advancement as well as determine possible updates to hyperparameters for future development.

In retrospect, there are multiple adjustments to multiple hyperparameters that should improve the results of the model. First, a learning rate of 0.001 for both the discriminator and generator proved to be too great. Examining the change in loss for both the discriminator and generator shows a large deal of fluctuation. Although some variation in the direction of change is expected due to the opposing learning by the networks, the large levels of change display a learning rate that is too large causing instance of divergence from the optimal solution. Again, this is clearly displayed in the images produced after each epoch as the resolution of the photo become visibly worse after an additional epoch. Another update to the generator would be to increase the size of the random noise tensors passed as inputs. The increase in size should allow for more features to be learned from the true dataset. Lastly, the number of epochs needs to be increased to obtain images with higher resolution which could deceive a human discriminator. These simple improvements should produce loss values and output images which follow a more proper learning path.

Gandhi, R. (2018, May 14). *Generative Adversarial Networks - Explained*. Medium. Retrieved April 24, 2022, from https://towardsdatascience.com/generative-adversarial-networks-explained-34472718707a

Ghosh, S. (2019, February 4). *Cats faces 64x64 (for generative models)*. Kaggle. Retrieved April 24, 2022, from https://www.kaggle.com/datasets/spandan2/cats-faces-64x64-for-generative-models

Rao, A. (2021, January 1). *Training Generative Adversarial Networks (GANs) in PyTorch*. Jovian. Retrieved April 24, 2022, from https://jovian.ai/aakashns/06b-anime-dcgan