Image Data Generation with GAN, VAE, and Sampling

Aditya Akula
College of Computing
Georgia Institute of Technology
akula@gatech.edu

Eshani Chauk
College of Computing
Georgia Institute of Technology
eshani@gatech.edu

Harsha Karanth
College of Computing
Georgia Institute of Technology
hkaranth3@gatech.edu

Shiva Ramaswami
College of Computing
Georgia Institute of Technology
sramaswami30@gatech.edu

1 Introduction

The goal of our project is to compare the effectiveness of multiple methods of synthetic data generation in neural network training. This process is particularly relevant for cases where training data is limited, such as in medical settings with rare diseases. In particular, we will be experimenting with modifications to the inference process of the Generative Adversarial Network (GAN) and the Variational Autoencoder (VAE), which are widely used for synthetic data generation.

The GAN is a network that uses two separate neural networks, and trains them to perform two separate tasks. The generator is trained to output synthetic examples that are meant to mimic the input dataset. The discriminator is trained to classify an input as synthetically generated data or real data from the dataset. In practice, after training we use the generator to create synthetic data; however, recent research shows that a combination of using MCMC sampling and the results from the discriminator can help improve the sampling quality of the generator. So we will compare the original GAN to a network with this added idea.

The VAE is a network that is trained to compress an input into a smaller representation, and then recreate the input. It does this using an encoder-decoder architecture, where the encoder is trained to transform the input into a compressed latent space, and the decoder is trained to transform input from the latent space back into the original input. Once training is finished, probabilistic samples from the latent space can be passed into the decoder to generate completely new synthetic data. Our project will test whether sampling methods that use the geometry of the latent space outperform standard MCMC sampling from the dataset's latent space (Gaussian prior for latent space), creating a synthetic dataset that leads to better performance.

2 Related Works

2.1 A Probe Towards Understanding GAN and VAE Models

Mi, Shen, and Zhang [1] provide comparisons between Generative Adversarial Networks (GAN) and Variational Autoencoders (VAE). Both models have been used to approximate the distributions of datasets, but their approaches greatly differ. In the GAN structure, the generator NN creates images from noise vectors, and the discriminator tries to classify real vs. generated images. The modified Wasserstein GAN (WGAN) solves the training difficulty and mode collapse problems with a new loss function that leverages the Wasserstein distance between the data distributions. However, GAN and WGAN still do not produce realistic images, which is why VAEs were developed. VAEs use two

Submitted to Conference on Neural Information Processing Systems (NeurIPS). Do not distribute.

NNs to minimize the KL divergence between generative and real distributions. Based on the authors' experiments, VAE and WGAN typically tended to produce higher-quality images. The new proposed model in this paper is VAE-GAN, which adds a discriminator on top of the generated image from the encoder-decoder to generate better images. It can produce images similar to WGAN but has the advantage of using the latent distribution from an input image.

2.2 Metropolis-Hastings Generative Adversarial Networks

In the paper titled "Metropolis-Hastings Generative Adversarial Networks", the "MH-GAN" is introduced to use the discriminator D to aid in data generation. This is an alternative of a traditional Generative Adversarial Network (GAN), which uses the discriminator in training but only uses the generator G during inference. The modified algorithm samples from the generator G, evaluates the sample using the discriminator D, and uses Metropolis-Hastings sampling to create a sample G' that is more representative of the dataset than data straight from the generator in some cases. Because D is trained specifically to differentiate between elements of the dataset ("real data") and generated elements ("fake data"), D can be used to evaluate the output of our generator during inference to generate samples closer to the dataset. This may be practical when our generator G isn't able to reach a "perfect" level, because the discriminator D should still be able to determine the quality G's outputs at a high accuracy. Through experimentation and testing of this method, results in the paper show that the MH-GAN is able to predict the distribution of the true data better than just a GAN when the generator doesn't fully capture the data distribution.

2.3 Data Augmentation with Variational Autoencoders and Manifold Sampling

In "Data Augmentation with Variational Autoencoders and Manifold Sampling" [5], the concept of manifold sampling — using the geometry of the latent space to inform the sampling process is introduced. This is an iteration on the default process of data generation with a VAE, which typically assumes the latent space follows a Gaussian distribution. This modified algorithm combines the process of normal sampling of random noise with the latent space's exponential map to generate samples from the latent space that can be decoded into images. In experiments, this process was able to significantly improve performance on a test set, especially in cases with limited training data.

2.4 Data Augmentation Generative Adversarial Networks

In the paper "Data Augmentation Generative Adversarial Networks," the authors introduce a new GAN architecture that generates new synthetic data from images without depending on the class of the image. They called this network a Data Augmentation Generative Adversarial Network (DAGAN). The main goal when augmenting data is to ensure that the new image still has the same class label as the original image. In a DCGAN, a random Gaussian vector and an encoded representation of the input image is sent to the generator (decoder). This component will generate an image, and the discriminator will accept a distribution of real images and a distribution with some real images and the generated image. The discriminator will decide what is the real distribution and fake distribution, and loss functions are used to calculate how accurate the predictions are and update the weights of the discriminator and generator. Because the discriminator receives the input images with the distributions and the discriminator is predicting realness with distributions and not only images, the model creates images that are similar to the source image but not the same.

3 Proposed Method

Specifically, we will compare results from four different data augmentation methods: VAE with sampling from a normal prior, VAE with manifold-aware sampling, standard GAN, and Metropolis-Hastings GAN. The first VAE model samples from the VAE's latent space using a normal prior, while the second VAE model utilizes differential geometry to more accurately map the latent space and samples from the resulting exponential map, theoretically resulting in more representative generated

samples. Similarly, the third model directly samples images solely from the GAN's generator, while the fourth model uses the discriminator's output values on samples created by the generator along with the Metropolis-Hastings algorithm to create more representative samples. In this section, we will discuss the synthetic models in more depth, and in the next section, we will discuss our experiments with the datasets.

3.1 Variational Autoencoder Method

We simultaneously train a Variational Autoencoder along with a corresponding Decoder network to encode samples as probability distributions into a latent space, sample from those distributions, and then reconstruct the samples into the original image. After training is complete, we can sample from the latent space by drawing randomly from the latent space and passing it into the decoder, giving us synthetic images. We will experiment with different methods of drawing randomly. We enforce a Gaussian prior on the latent space during training, so the default method is just to sample from this Gaussian, but more geometry-aware methods should (in theory) work better, as they are more likely to generate points similar to ones that appeared in the training set.

Specifically, we will compare standard sampling to the results of a manifold-aware Riemannian Random Walk. This works by using the manifold's exponential map, which intuitively corresponds to shooting a certain distance along the manifold's geodesic, measuring volume with $\sqrt{\det G(z)}$, where G defines an inner product on the manifold. This quantity is lower in areas of low density and higher in areas of high density, so we can shoot along these geodesics and then use Metropolis-Hastings sampling with an acceptance/rejection rate defined by

$$\alpha(z, z') = \min\left(1, \frac{\sqrt{\det G^{-1}(z')}}{\sqrt{\det G^{-1}(z)}}\right)$$

, so the random walk is more incentivized to sample from areas that resemble the data distribution in the latent space.

3.2 Metropolis-Hastings GAN Method

The generator distribution is denoted by pG, and the discriminator distribution in respect to the generator is called pD. The generator distribution, pG, does not always converge to the real image data distribution (due to data or resource constraints) and can result in an imperfect generator. The discriminator distribution, pD, is closer to the real distribution, and we can obtain a new generator G' that models the data distribution more accurately by sampling from the pD distribution. We use the MH-GAN approach mentioned in related works and the algorithm mentioned above to sample the generator output using the discriminator. We will first initialize the best generator sample to a real sample so that we can check after the sampling procedure if the best generator sample is still a real sample. If it is, we restart the sampling procedure because we did not accept a sample from the generator. We will sample K times and will accept the new sample or keep a previous sample sequentially. During a sampling process, we generate one sample using the generator and draw a number from the uniform distribution between 0 and 1. The acceptance probability is equal to

$$p_g \,/\, p_d = \min\left(\frac{D(x)^{-1}-1}{D(x')^{-1}-1}, 1\right)$$

D(x) represents the discriminator score of the best sample, and D(x') represents the discriminator score of the new sample. If the ratio is greater than one, then the new sample is better than the old sample, and the acceptance probability will be equal to 1 because of the min function. The number that is getting compared to the acceptance probability is between 0 and 1 so this sample will always get accepted. After K samples, we will find the best generator sample or we restart the process. This process is described in Figure 1 and 2.

4 Experiments and Results

In this section, we will discuss the training and image generation process for all the models such as MH-GAN, GAN, VAE, and VAE with manifold sampling. To avoid these models from making hybrid classes from the cifar dataset, we trained each model on one specific class. Because training each model on each individual class will take a substantial amount of time, our benchmark dataset only consisted of dog and cat images from the CIFAR-10 dataset. We created a standard CNN classifier that trained and gave predictions on the pure cifar dataset with dogs and cats and 4 other cifar datasets with synthetic data from MH-GAN, normal GAN, VAE with manifold-aware sampling, and normal VAE. Each synthetic dataset contained 50% pure cifar images and 50% from the assoicated model. We then graphed the accuracy and loss of the classifier on each dataset to discover what dataset performed the best.

4.1 VAE Results

As previously mentioned, we split the CIFAR-10 dataset by class to aid in training time and results. We wanted to test the quality of images generated in each class and avoid hybrid images, which is why we chose this approach. We utilized early stopping to train the VAE and the Manifold-Aware VAE. We started with 1000 epochs, but both models stabilized and stopped around 450 epochs for both cat and dog training datasets. This allowed for the model itself to determine the stopping point, and we experimented with different early stopping points, learning rates, batch sizes, and regularization. The most optimal combination we found was 1000 epochs with 50 early stopping epochs, batch size of 200, and learning rate of 0.001.

We observed that adding the manifold-aware sampling to the VAE increased the quality of images in the cat and dog classes. Of the 100 samples we generated, there were more images in the dog class that we could perceive as images of dogs. We noticed the same trend for the cat class images as well. These are only human, qualitative observations though, so we also utilized a more qualitative measure to help us understand if the MCMC manifold-aware sampling was an improvement for generating images with VAEs. 100 samples of generated images for cats and dogs from VAE and manifold-aware VAE are shown in Figures 10, 11, 12, and 13.

For the CNN-based classifier, we compared three datasets: pure CIFAR-10 data, 50% CIFAR with 50% VAE images, and 50% CIFAR with 50% Manifold-Aware VAE images. This is because we wanted to see if adding the synthetic images from the VAE experiments would increase the classifier's accuracy. We focused on the accuracy and loss per epoch of each of these datasets run through the classifier, which is shown in Figure 9. In terms of accuracy, the pure CIFAR dataset (only cats and dogs) reached 87%, while the combined CIFAR/VAE images produced an accuracy of 92% and the combined CIFAR/manifold-VAE reached 93%. For loss, the pure CIFAR dataset reached a loss of 0.371, while the combined CIFAR/VAE datset reached 0.191 and the combined CIFAR/manifold-VAE dataset reached a loss of 0.202.

This clearly shows that adding the synthetic images to the CIFAR dataset for the cat and dog classes showed an impressive 5% increase in accuracy. This tells us that the images generated by the VAE and manifold-VAE models are more likely to be classified into their correct classes since they are better representations of the respective class, meaning that the VAE models have generated better images of cats and dogs. Between VAE and manifold-VAE, the latter shows slight improvements in both accuracy and loss. This means that the manifold-VAE produces better quality and more representative images. With some more hyperparameter tuning and enhancing the Riemmanian random walk based sampling, there is great potential to improve the images generated by manifold-aware VAE. This VAE modification is especially helpful in scenarios where data is limited because it is able to rely on the latent space structure of the training data to generate high-quality and representative images.

4.2 MH-GAN Results

In our initial experiments, we implemented the MH-GAN and trained a model on the entire CIFAR-10 dataset to observe its performance in augmenting the dataset. After running training for 40 epochs, we saw significant improvement in sampled images (from observation of images and inception score) that used MH sampling with the discriminator. In our final experiments, we extended our analysis by running two more experiments – one specifically for cats and one for dogs. This time, we were able to see even more improvements as we ran each experiment for 100+ epochs.

Once again, we observed that the discriminator was useful to disregard some lower quality images because discriminator loss was significantly smaller than generator loss during training. As the generator learned, we observed the output images as well as the inception score, a metric that describes the diversity of the model outputs, to decide when to stop training and which epoch iterations to take the GAN outputs from. As shown in Figure 8, inception score for each epoch isn't linearly decreasing. As the generator learns, the diversity and accuracy of generated images when compared to the original dataset goes up and down. During this second training iteration with only images from one class, we saw a higher inception score as opposed to the original training iteration with the full dataset. This could have been because of the full CIFAR-10 GAN generating hybrid-class images and having a difficult time generating class specific images, whereas the dog-specific and cat-specific GANs learned how to generate diverse and accurate images of the given class.

When running the standardized classifier on the pure CIFAR-10 dataset, the CIFAR-10 dataset with 50% GAN images, and CIFAR-10 dataset with 50% MH-GAN images, we focused on the classifer's accuracy and loss per epoch.

For the accuracy metric, we found that the model had an accuracy of 87% for the pure CIFAR dataset, 92% for the GAN generated images + CIFAR dataset, and 92% for the MH-GAN generated images + CIFAR dataset when training the model for 20 epochs. For the loss metric, the model ended with a loss of .371 for the pure CIFAR dataset, .215 for the GAN generated images + CIFAR dataset, and .227 for the MH-GAN generated images + CIFAR dataset. The graphs for the accuracy and loss are in the appendix (Figure 7).

Ultimately, this displays that adding synthetic images does improve model accuracy and that adding MH-GAN generated images to the dataset can reach the same accuracy as adding normal GAN generated images to the dataset. Additionally, MH-GAN is more efficient than GAN because it is able to reject bad samples. As a result, it is able to output more quality images or converge faster than a normal GAN. Because the MH-GAN dataset achieves the same accuracy as the GAN dataset and the training time for MH-GAN is faster, we found that using MH-GAN is an effective GAN for synthetic data generation and can be used as an alternative when data and/or compute resources are restricted.

5 Overall Dataset Comparison and Conclusion

Overall, adding synthetic images improved the accuracy of the classifier. The classifier had a 5-10% increase in accuracy and a 50% decrease in loss when synthetic images where added. When comparing the performance of the classifier on the synthetic datasets, the model performed slightly better on the VAE generated datasets than the GAN generated datasets. The classifier was able to reach an accuracy of 92.7 with the VAE with manifold-aware sampling images and around 92% accuracy with the MH-GAN images. Additionally, the classifier started at a higher accuracy of 70% with the VAE generated images when training started while the classifier started with an accuracy of 60% with GAN generated images. An explanation for why the VAE with manifold-aware sampling performed better slightly better than the other models is that the random walk sampling was more effective in sampling the latent space than the other models, especially when the latent space wasn't super well mapped to start with, especially since we trained for a relatively short time. In conclusion, when data is limited, synthetic data generation is a valid method to increasing model performance.

References

[1] L. Mi, M. Shen, and J. Zhang, A Probe Towards Understanding GAN and VAE Models. 2018.

[2] R. Turner, J. Hung, E. Frank, Y. Saatci, and J. Yosinski, Metropolis-Hastings Generative Adversarial Networks. 2019.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.

[4]Antoniou, A., Storkey, A., & Edwards, H. (2018). Data Augmentation Generative Adversarial Networks.

[5] Clément Chadebec, Stéphanie Allassonnière, "Data Augmentation with Variational Autoencoders and Manifold Sampling," 2021.

Appendix

Algorithm 1 MH-GAN

Input: generator G, calibrated disc. D, real samples
Assign random real sample $\mathbf{x}_0$ to $\mathbf{x}$
for $k = 1$ **to** K **do**
  Draw $\mathbf{x}'$ from G
  Draw U from Uniform$(0, 1)$
  if $U \leq (D(\mathbf{x})^{-1} - 1)/(D(\mathbf{x}')^{-1} - 1)$ **then**
    $\mathbf{x} \leftarrow \mathbf{x}'$
  end if
end for
If $\mathbf{x}$ is still real sample $\mathbf{x}_0$ restart with draw from G as $\mathbf{x}_0$
Output: sample $\mathbf{x}$ from G'

Figure 1: MH-GAN Pseudocode (Turner et al, 2018)

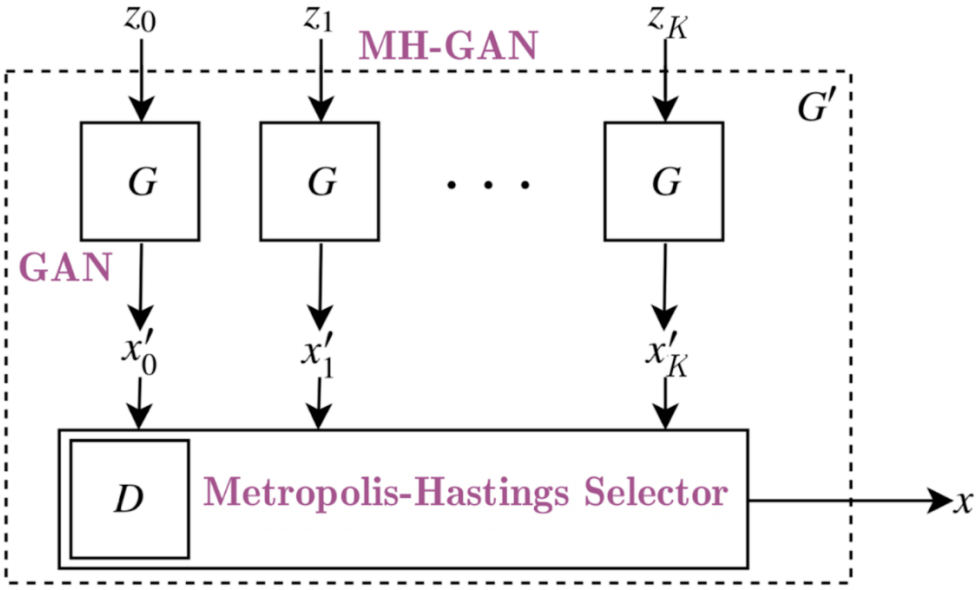


Figure 2: MH-Selector purpose visualization. It uses K generator samples and then outputs the best generator sample using the Discriminator. (Turner et al, 2018)

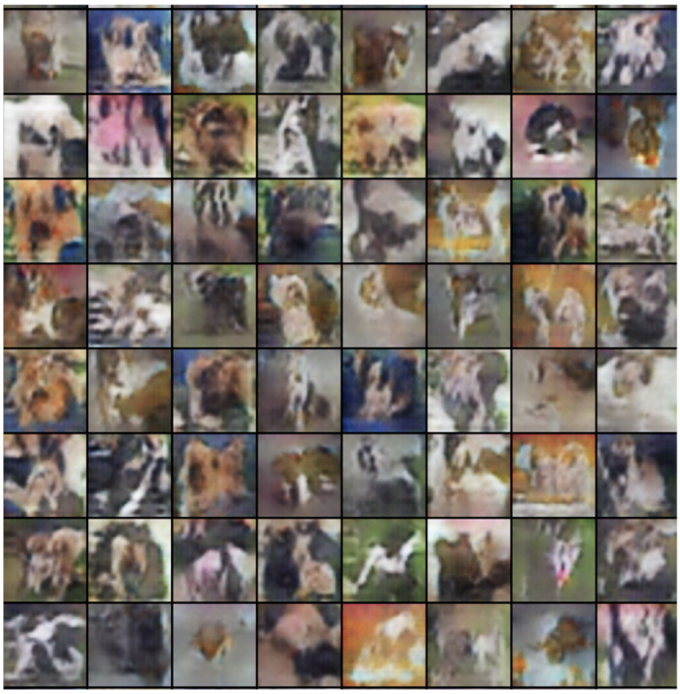


Figure 3: 64 dog images generated by MH-GAN.

Figure 4: 64 cat images generated by MH-GAN.

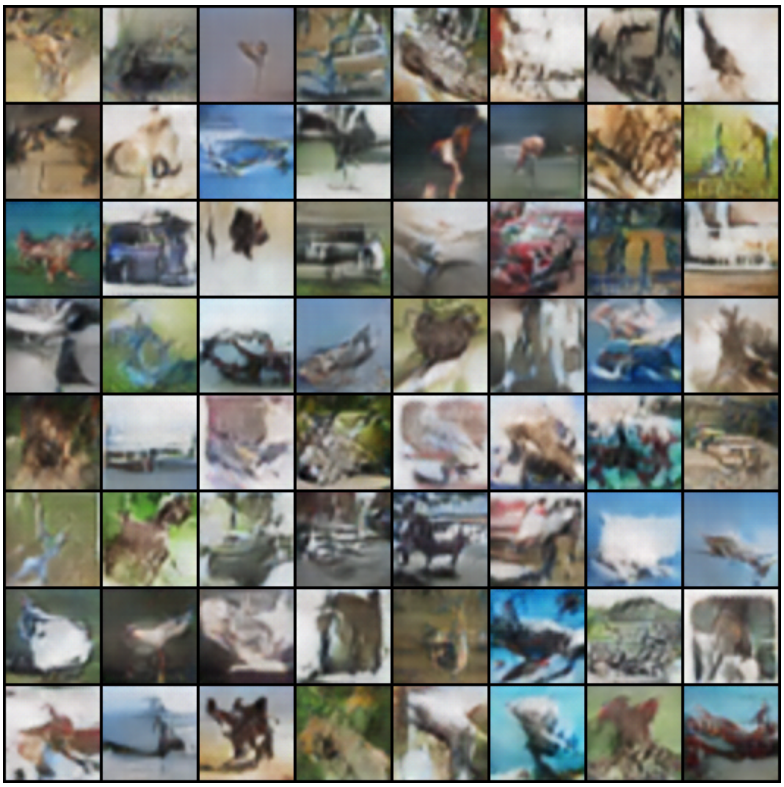


Figure 5: 64 images spanning all classes generated by normal GAN.

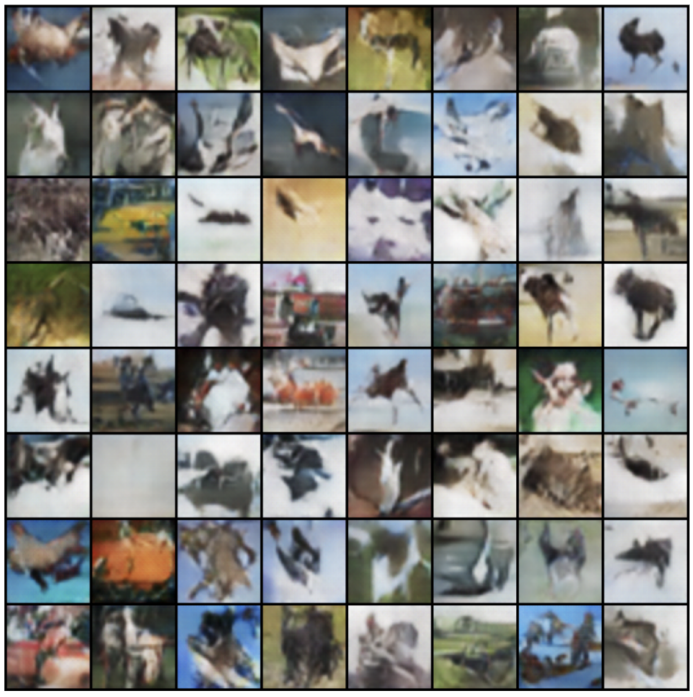
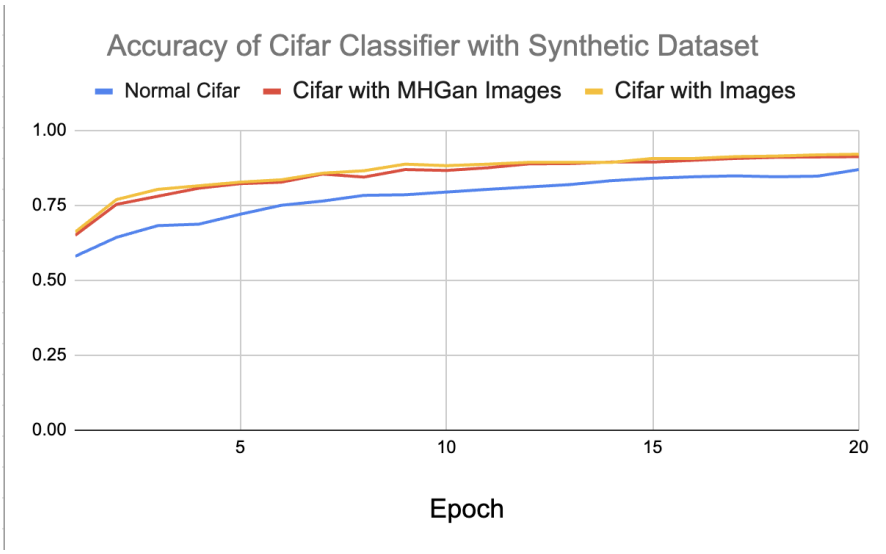


Figure 6: 64 images spanning all classes generated by MH-GAN.



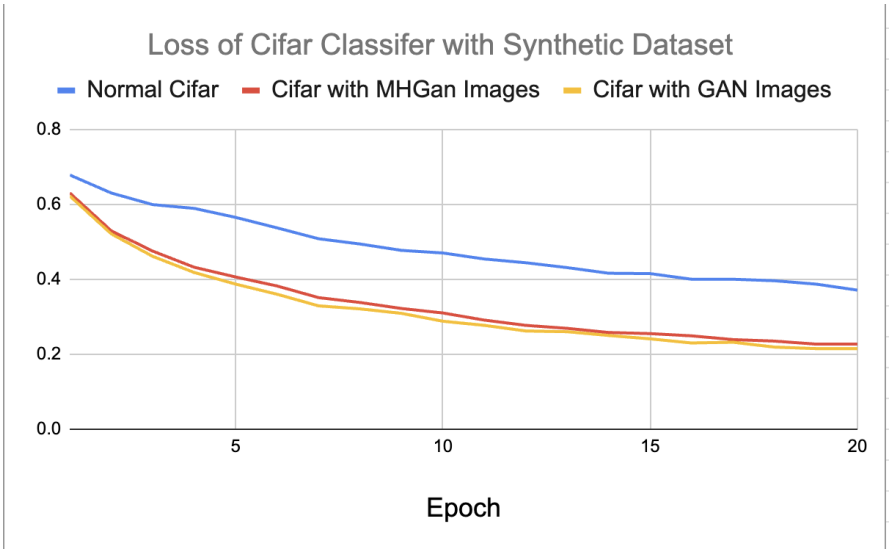


Figure 7: Accuracy and Loss graphs for cifar classifer on a pure cifar dataset, a cifar dataset with MH-GAN Images, and a cifar dataset with GAN Images.

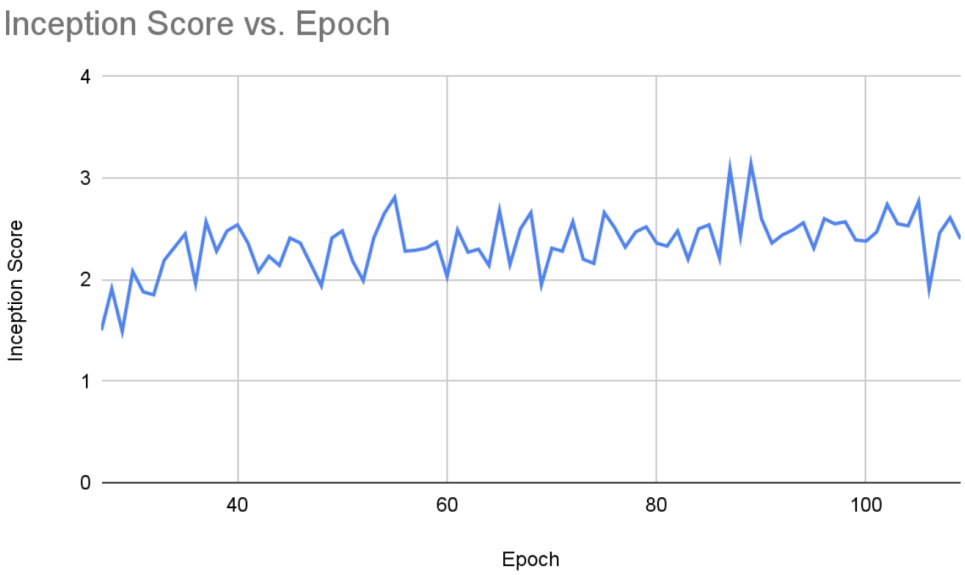
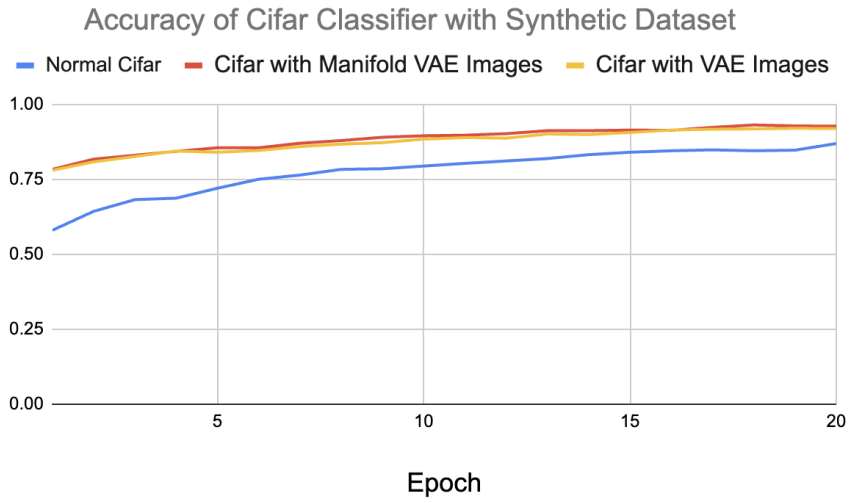


Figure 8: Inception Score for each epoch during MH-GAN training.



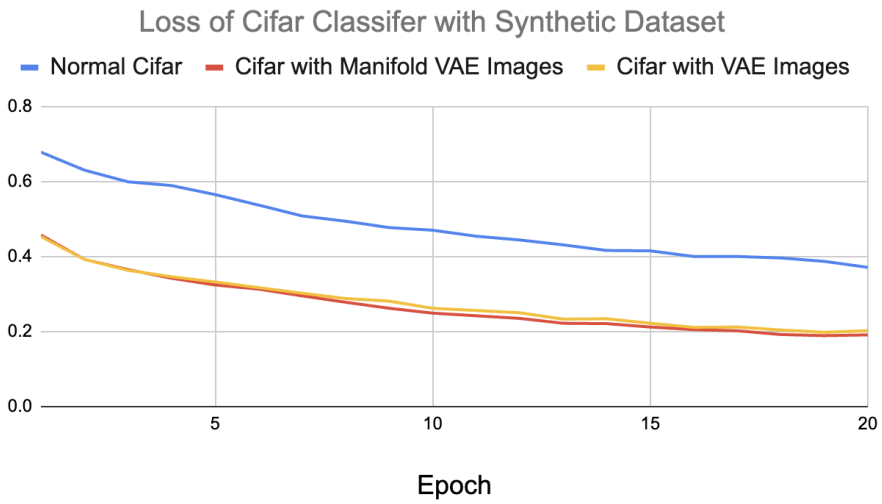


Figure 9: Accuracy and Loss graphs for cifar classifer on a pure cifar dataset, a cifar dataset with Manifold-VAE Images, and a cifar dataset with VAE Images.

Figure 10. 100 images of cats generated by VAE.

Figure 11. 100 images of cats generated by manifold-aware VAE.

Figure 12. 100 images of dogs generated by VAE.

Figure 13. 100 images of dogs generated by manifold-aware VAE.

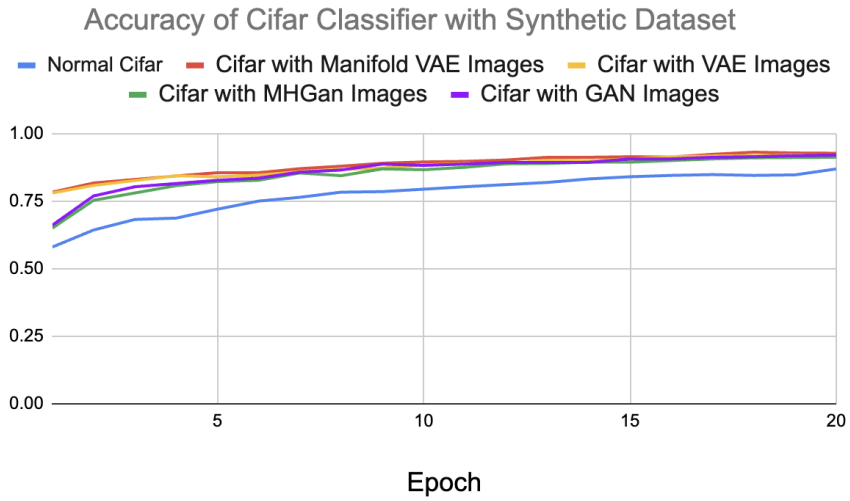


Figure 14. Accuracy graph for cifar classifier on a pure cifar dataset, a cifar dataset with Manifold-VAE Images, a cifar dataset with VAE Images, a cifar dataset with MH-GAN Images, and a cifar dataset with GAN Images.

Old Plan of Activities

Week	Tasks	Key Dates
10/1-10/7	- Research topics and ideas - Collect datasets - Write proposal	- Proposal due 10/4 5pm
10/8-10/14	- Implement GAN for data augmentation - Experiment with MCMC in GAN to enhance training	
10/15-10/21	- Implement VAE for data augmentation - Train classification model with synthetic data from both GAN and VAE	
10/22-10/28	- Start writing midway report (intro, background, related work) - Finish implementations of data augmentation techniques	
10/29-11/4	- Write methodology and experiments section of midway report - Understand how to use MH or Gibbs sampling with image data	
11/5-11/11	- Write conclusion - Refine and submit midway report - Start implementing sampling-based data augmentation	- Midway report due 11/6 5pm
11/12-11/18	- Implement data augmentation based on MCMC sampling with GAN and VAE	
11/18-11/25	- Prepare presentation for class	
11/26-12/2	- Practice and refine presentation	- Presentation 11/29-12/4
12/3-12/11	- Finish final experiments for data augmentation - Write final report (elaborate methods, experiments, conclusion)	- Final report due 12/11 5pm

Revised Plan of Activities

Week	Tasks	Key Dates
10/1-10/7	- Research topics and ideas - Collect datasets - Write proposal	- Proposal due 10/4 5pm
10/8-10/14	- Implement PyTorch DCGAN for data augmentation - Implement VAE for data augmentation	
10/15-10/21	- Understand how to use MH sampling with image data for GAN	
10/22-10/28	- Implement MH methods to add to MH-GAN	
10/29-11/4	- Write methodology and experiments section of midway report - Generate synthetic images from MH-GAN and add results to report	
11/5-11/11	- Refine and submit midway report - Understand how sampling-based data augmentation works with VAE	- Midway report due 11/6 5pm
11/12-11/18	- Implement MCMC sampling with VAE - Generate synthetic images from modified VAE	
11/18-11/25	- Train classification model with synthetic data from both GAN and VAE (testing method)	
11/26-12/2	- Prepare and practice presentation	- Presentation 11/29-12/4
12/3-12/11	- Finish final experiments for data augmentation - Write final report (elaborate methods, experiments, conclusion)	- Final report due 12/11 5pm