Exploring the Impact of Model Pruning on GANs

ECE 591
Chinmaya Srivatsa, Akshay Khanna

NC STATE

Motivation

The rising prominence of Generative Adversarial Networks (GANs) has transformed AI, particularly in image generation and data synthesis. However, their computational demands limit their practical deployment. Our motivation lies in exploring how model pruning techniques can significantly reduce these demands while maintaining performance, thereby enabling more efficient and deployable GAN models.

NC STATE

Abstract

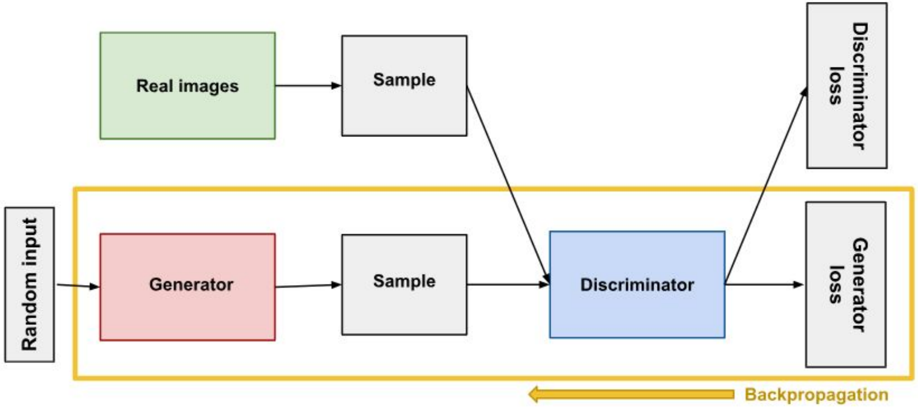
- Project Focus:
  - Investigating Model Pruning on DCGANs.
  - Explore trade-offs between model size, computational efficiency, and image quality.

- Dataset and Adaptability:
  - Initial Training: CIFAR-10 dataset.
  - Adaptability: Fashion-MNIST

- Objective and Scope:
  - Analyze Sensitivity and Percentage Pruning
  - Analyze Sparsity vs Image quality

Generative Adversarial Networks

GANs (Generative Adversarial Networks) contain two networks, the generator and discriminator, collaborating adversarially to generate data.



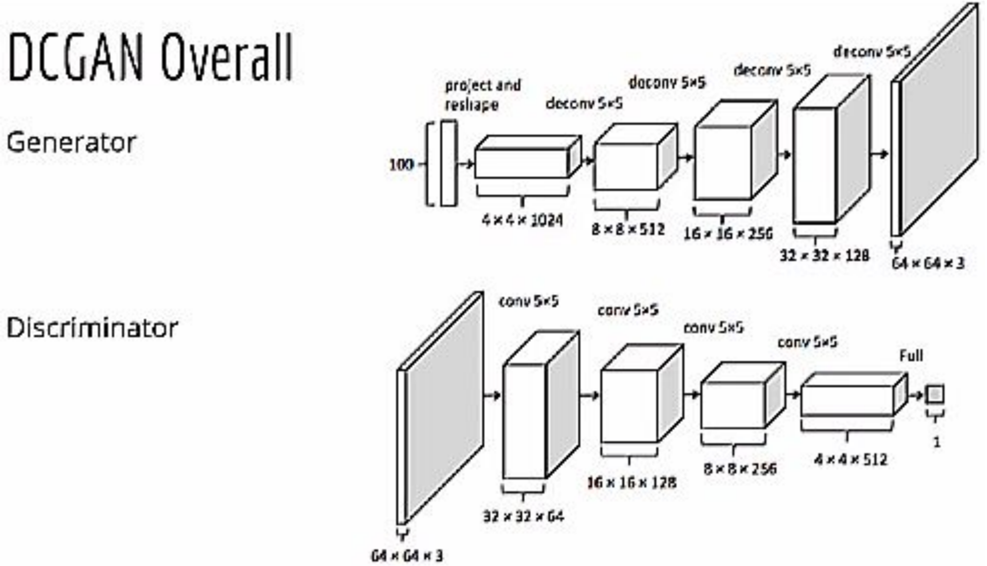
- **Generator Function:** Generates synthetic data resembling real data by learning from the provided dataset.
- **Discriminator Function:** Evaluates generated data by discerning between real and fake, providing feedback to the generator for improvement. Adversarial Training:
- **Generator** and **Discriminator** engage in a competitive learning process, improving each other's capabilities.

NC STATE

DC Generative Adversarial Networks

A DCGAN is a direct extension of the GAN described above, except that it explicitly uses convolutional layers in the discriminator and generator

- DCGANs utilize convolutional layers in both the generator and discriminator networks, enabling efficient handling of image data.
- The **generator** upsamples noise or random inputs into images through a series of **transposed** convolutional layers.
- The **discriminator** acts as a convolutional neural network classifying between real and generated images.
- Generation of increasingly complex and realistic images over training iterations.



NC STATE

Pruning

- Pruning:
  - **Reduction of Model Size:** Reduces the size of a neural network by eliminating weights/parameters
  - **Sparsity Introduction:** Introduces sparsity into the network, making it more computationally efficient.
  - **Resource Efficiency:** Faster training, lower memory footprint, and easier deployment on resource-constrained devices.
  - **Relearning of Parameters:** Doesn't involve retraining the network from scratch

NC STATE

Datasets

- CIFAR-10:
  - Contains 60,000 32x32 color images across 10 classes (such as airplanes, cars, birds, cats, etc.).
  - Widely used for image classification tasks and for benchmarking machine learning algorithms.

- Fashion MNIST:
  - Consists of 28x28 grayscale images of fashion items (like clothes, shoes, bags) across 10 categories.
  - Used as a drop-in replacement for the original MNIST dataset, serving as a more challenging benchmark for image classification tasks.

NC STATE

DCGAN - Discriminator

```
Discriminator(
  (main): Sequential(
    (0): Conv2d(1, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): LeakyReLU(negative_slope=0.2, inplace=True)
    (2): Conv2d(64, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (3): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (4): LeakyReLU(negative_slope=0.2, inplace=True)
    (5): Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (6): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (7): LeakyReLU(negative_slope=0.2, inplace=True)
    (8): Conv2d(256, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (9): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (10): LeakyReLU(negative_slope=0.2, inplace=True)
    (11): Conv2d(512, 1, kernel_size=(4, 4), stride=(1, 1), bias=False)
    (12): Sigmoid()
  )
)
```

NC STATE

DCGAN - Generator

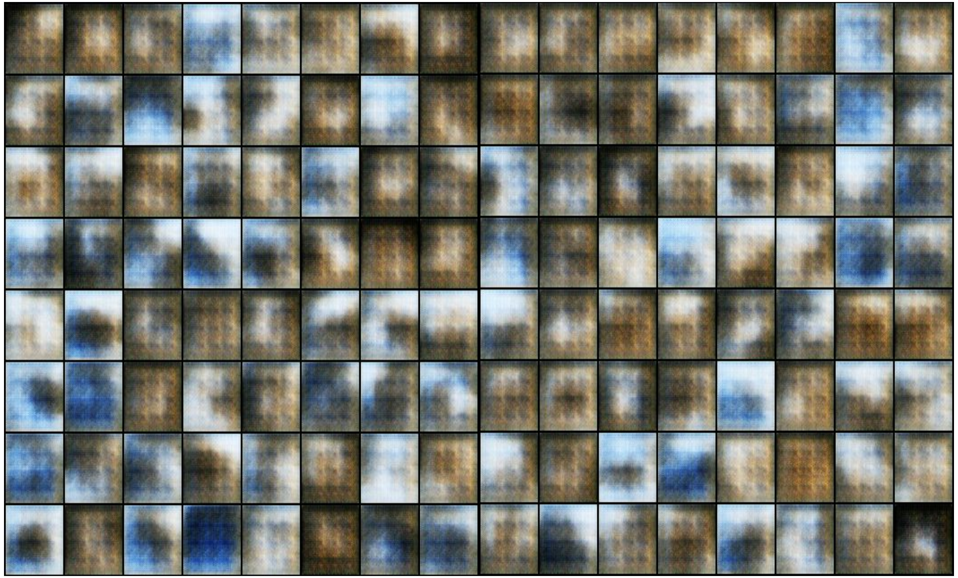
```
Generator(
  (main): Sequential(
    (0): ConvTranspose2d(100, 512, kernel_size=(4, 4), stride=(1, 1), bias=False)
    (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU(inplace=True)
    (3): ConvTranspose2d(512, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (4): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (5): ReLU(inplace=True)
    (6): ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (7): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (8): ReLU(inplace=True)
    (9): ConvTranspose2d(128, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (10): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (11): ReLU(inplace=True)
    (12): ConvTranspose2d(64, 1, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (13): Tanh()
  )
)
```

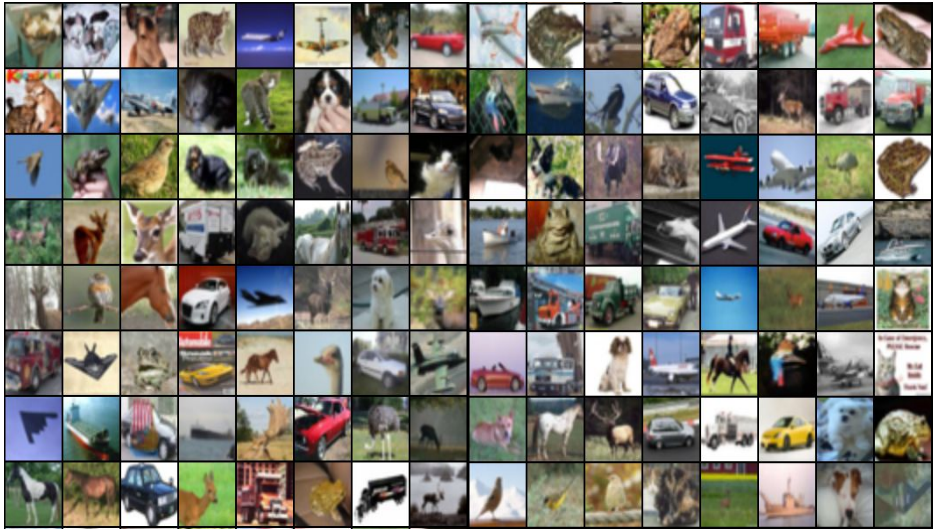
DCGAN - Training (CIFAR10)

- Epochs - 30
- Discriminator Learning Rate - 0.0002
- Generator Learning Rate - 0.0003
- Optimizer - Adam
- Betas - 0.5, 0.999

NC STATE

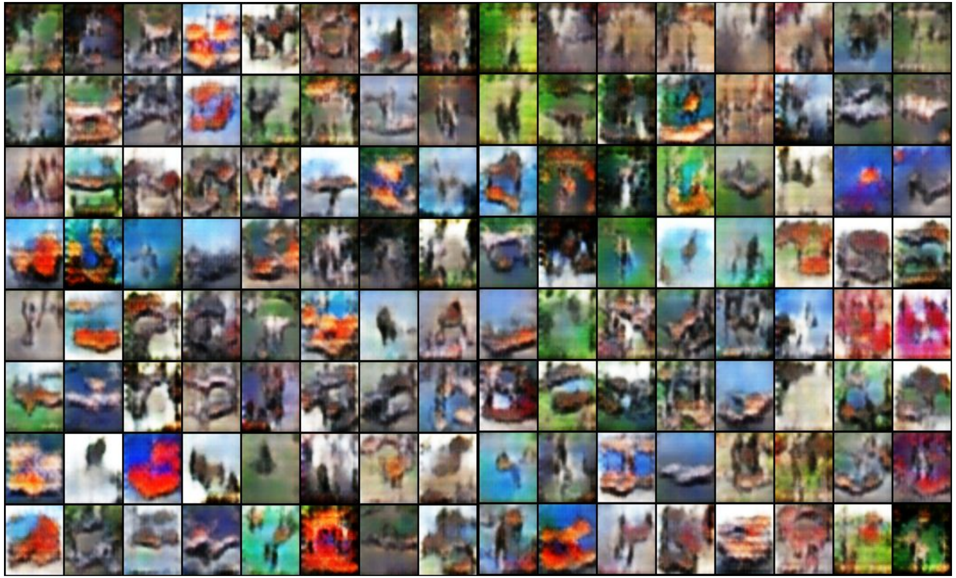
Generated v/s Real Images for 0 epoch(CIFAR-10)

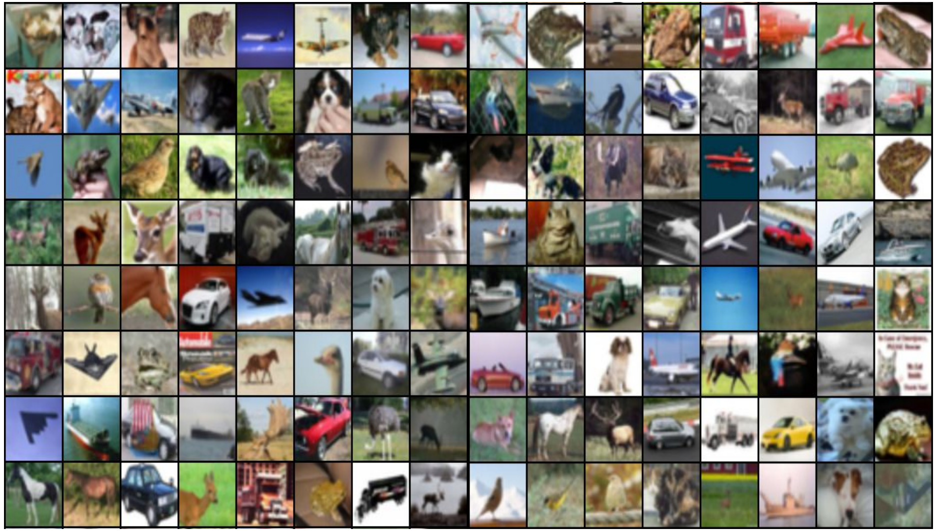




NC STATE

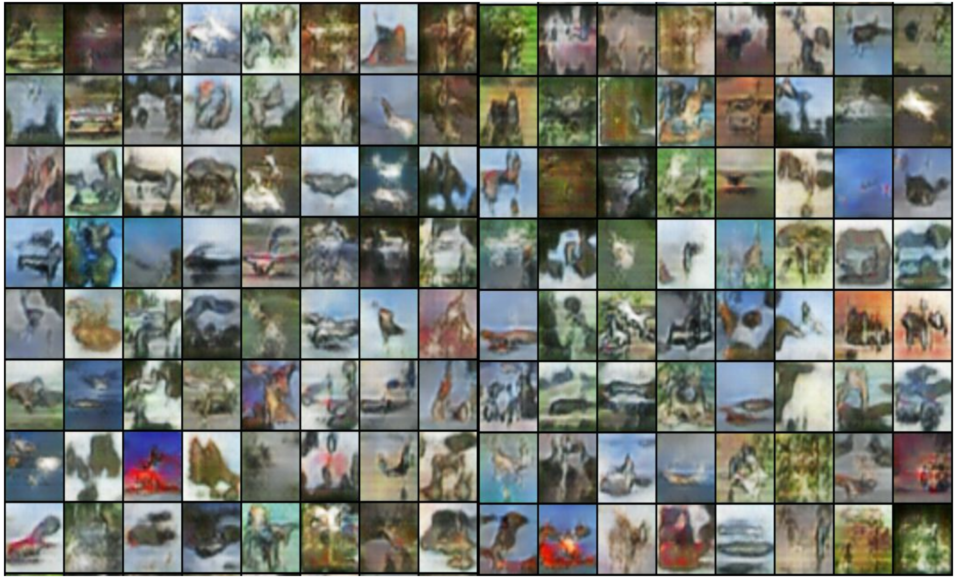
Generated v/s Real Images for 5 epoch(CIFAR-10)

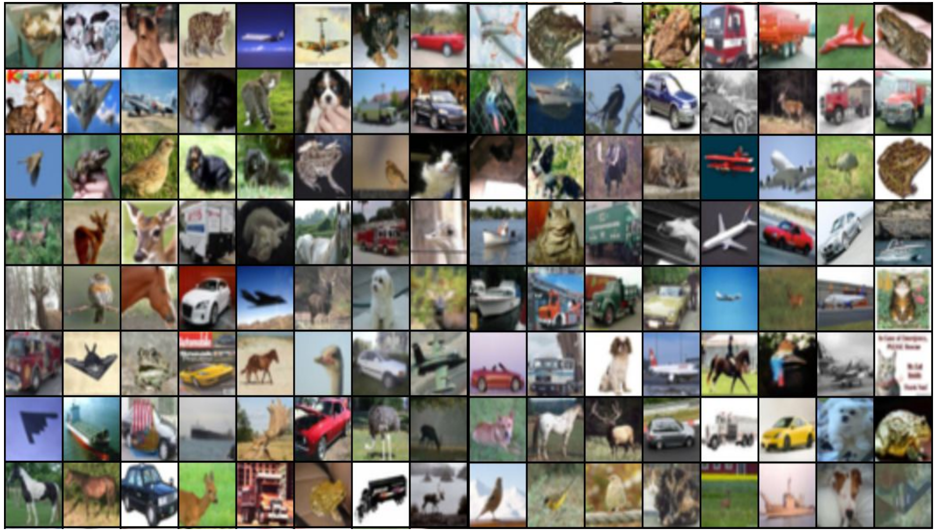




NC STATE

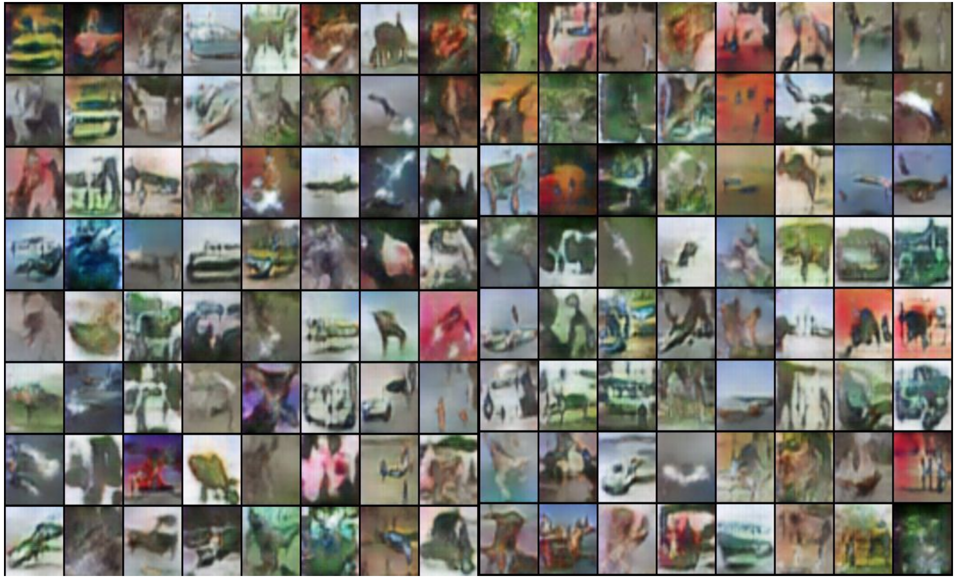
Generated v/s Real Images for 10 epoch(CIFAR-10)

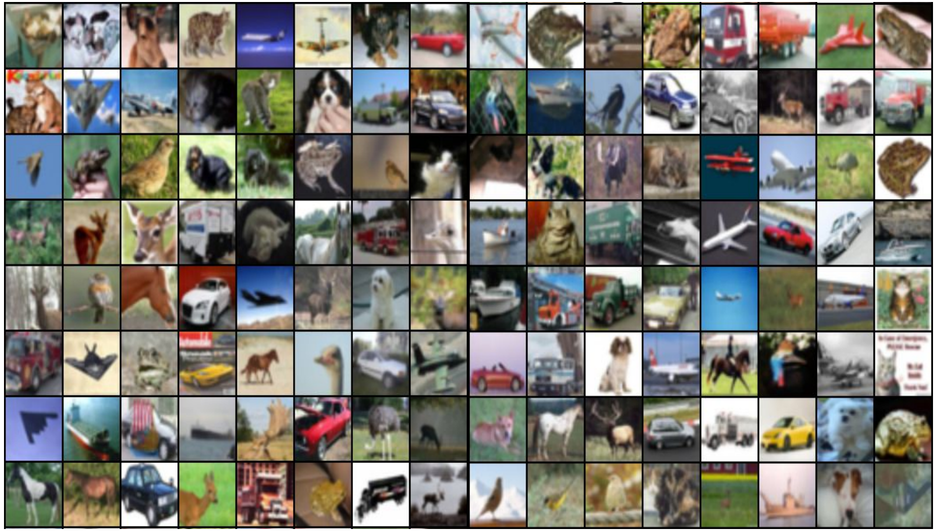




NC STATE

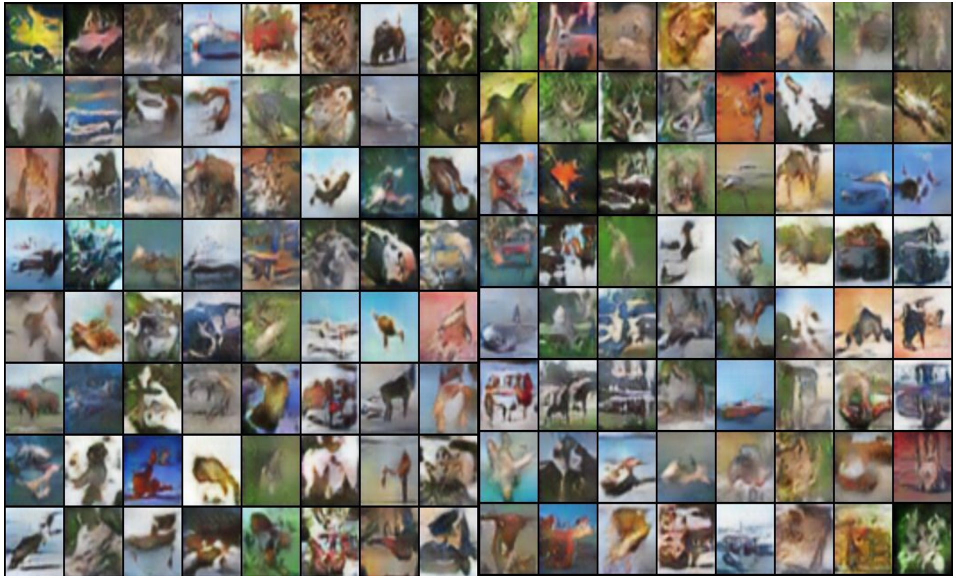
Generated v/s Real Images for 15 epoch(CIFAR-10)

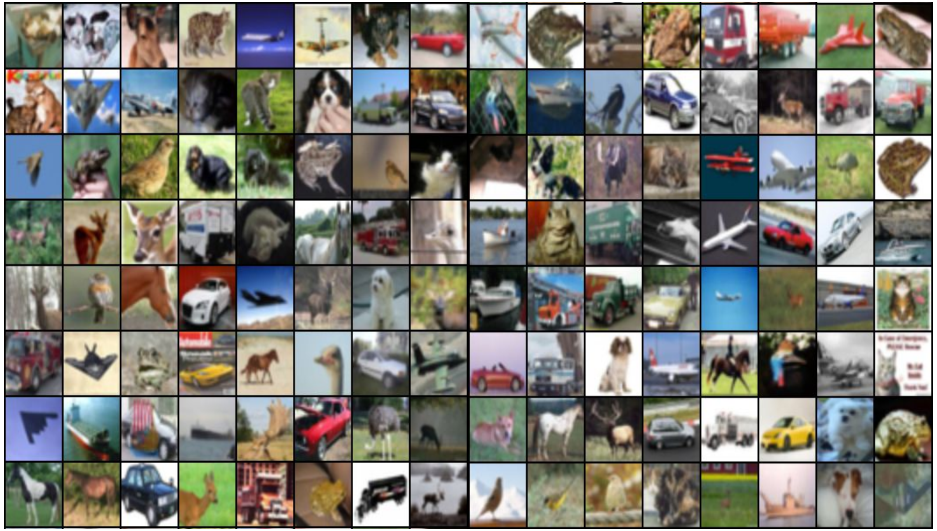




NC STATE

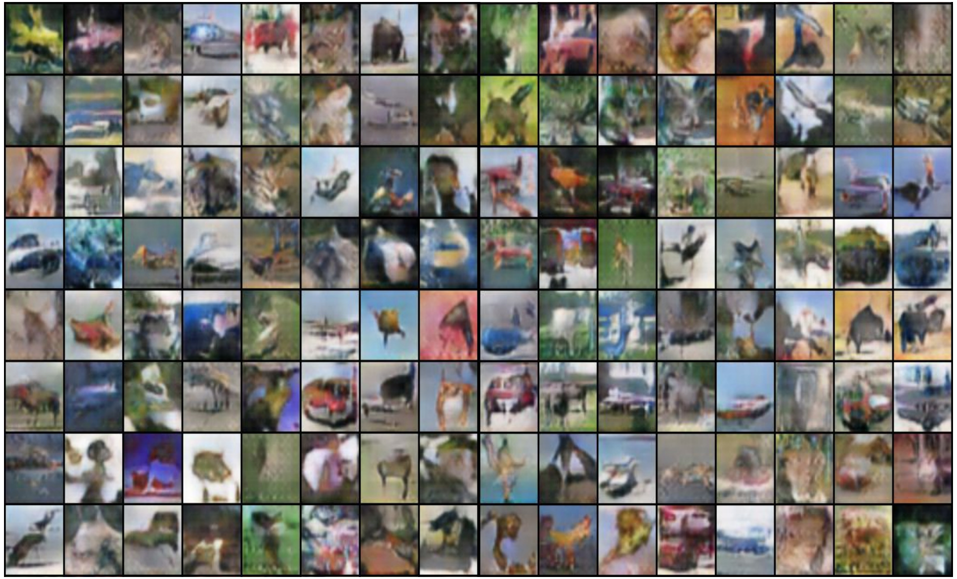
Generated v/s Real Images for 25 epoch(CIFAR-10)

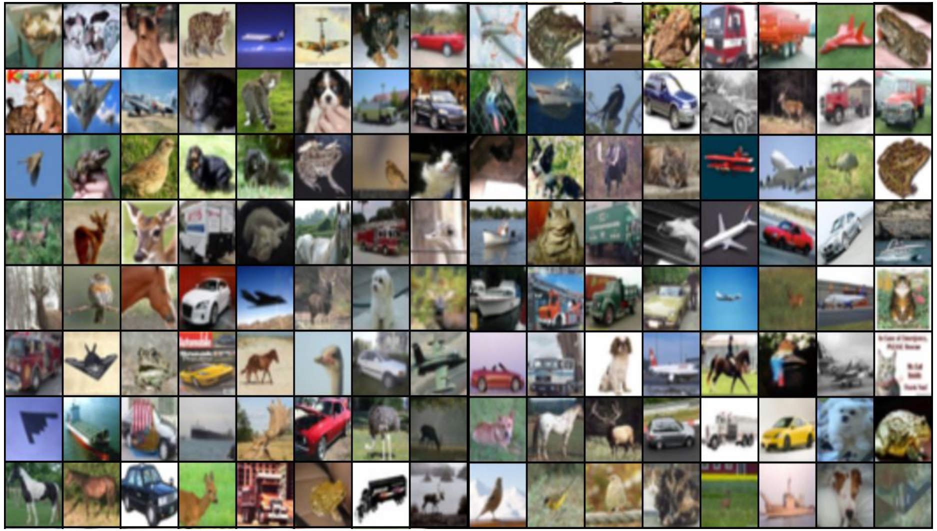




NC STATE

Generated v/s Real Images for 30 epoch(CIFAR-10)





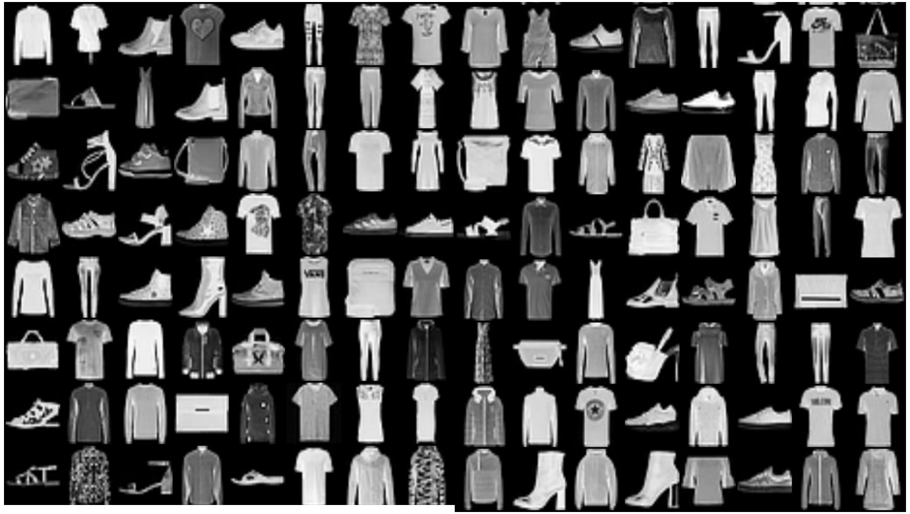
NC STATE

DCGAN - Training (Fashion-MNIST)

- Epochs - 15
- Discriminator Learning Rate - 0.0001
- Generator Learning Rate - 0.0003
- Optimizer - Adam
- Betas - 0.5, 0.999

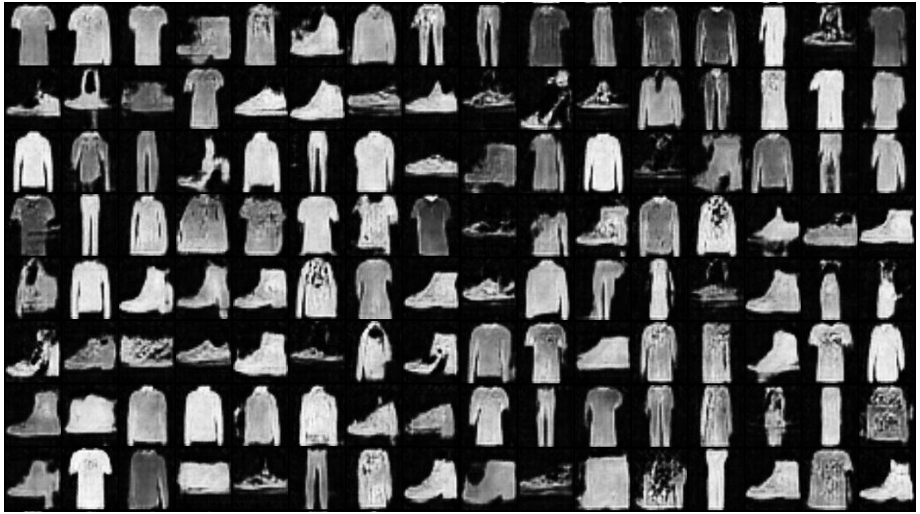
NC STATE

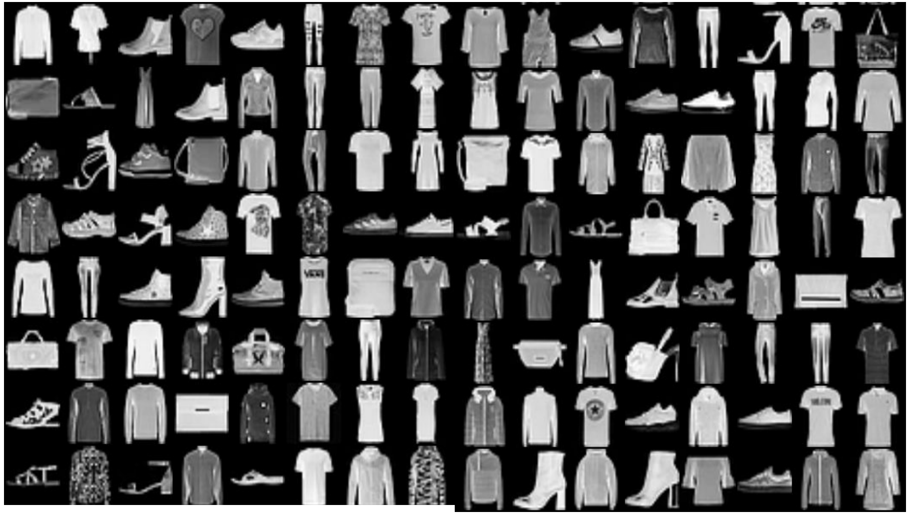
Generated v/s Real Images for 0 epoch(Fashion-MNIST)



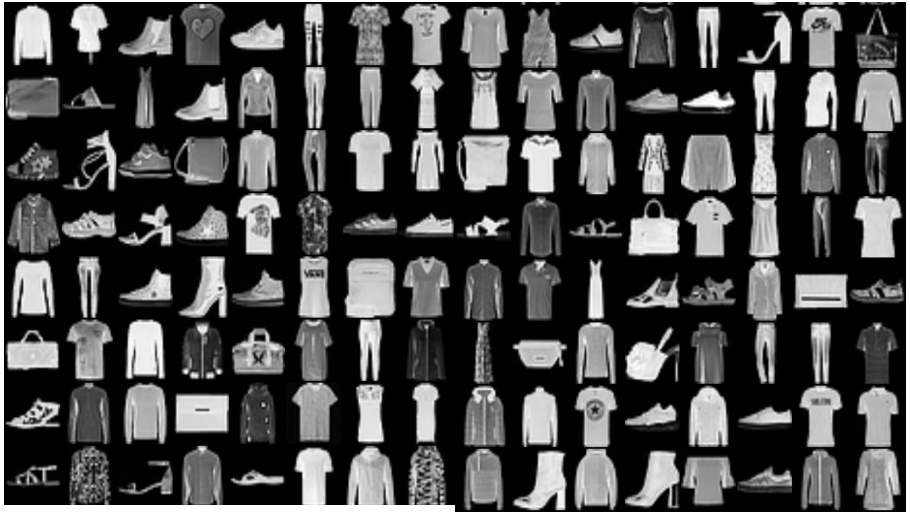
NC STATE

Generated v/s Real Images for 5 epoch(Fashion-MNIST)



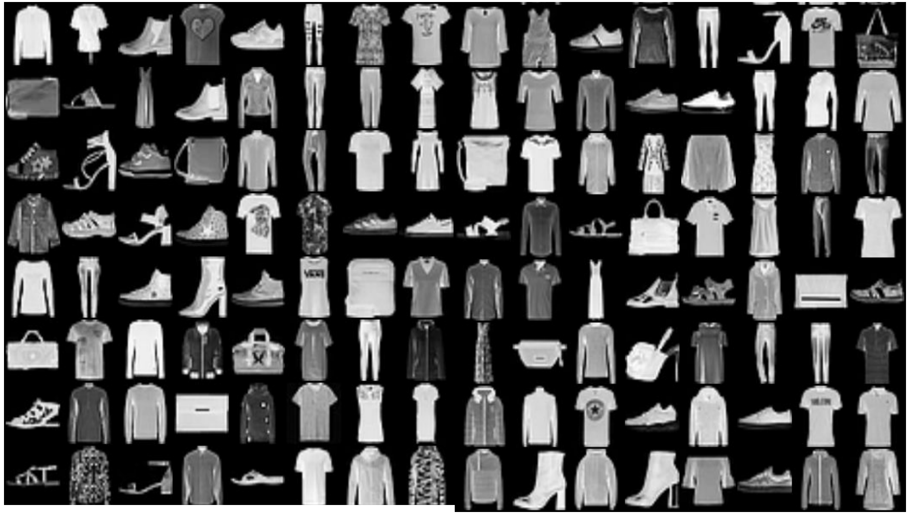


Generated v/s Real Images for 10 epoch(Fashion-MNIST)



NC STATE

Generated v/s Real Images for 15 epoch(Fashion-MNIST)



NC STATE

Before Pruning

```
Layer id        Type            Parameter       Non-zero parameter      Sparsity(\%)
1024
1               Convolutional   1024            1024                    0.000000
131072
2               Convolutional   131072          131072                  0.000000
3               BatchNorm       N/A             N/A                     N/A
524288
4               Convolutional   524288          524288                  0.000000
5               BatchNorm       N/A             N/A                     N/A
2097152
6               Convolutional   2097152         2097152                 0.000000
7               BatchNorm       N/A             N/A                     N/A
8192
8               Convolutional   8192            8192                    0.000000
Total nonzero parameters: 2761728
Total parameters: 2761728
Total sparsity: 0.000000
0.0
```

Before Pruning

```
Layer id        Type            Parameter       Non-zero parameter      Sparsity(\%)
819200
1               Convolutional   819200          819200                  0.000000
2               BatchNorm       N/A             N/A                     N/A
3               ReLU            N/A             N/A                     N/A
2097152
4               Convolutional   2097152         2097152                 0.000000
5               BatchNorm       N/A             N/A                     N/A
6               ReLU            N/A             N/A                     N/A
524288
7               Convolutional   524288          524288                  0.000000
8               BatchNorm       N/A             N/A                     N/A
9               ReLU            N/A             N/A                     N/A
131072
10              Convolutional   131072          131072                  0.000000
11              BatchNorm       N/A             N/A                     N/A
12              ReLU            N/A             N/A                     N/A
1024
13              Convolutional   1024            1024                    0.000000
Total nonzero parameters: 3572736
Total parameters: 3572736
Total sparsity: 0.000000
0.0
```

NC STATE

DCGAN architecture modified for pruning

```
Discriminator(
  (main): Sequential(
    (0): PrunedConv(
      (conv): Conv2d(1, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    )
    (1): LeakyReLU(negative_slope=0.2, inplace=True)
    (2): PrunedConv(
      (conv): Conv2d(64, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    )
    (3): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (4): LeakyReLU(negative_slope=0.2, inplace=True)
    (5): PrunedConv(
      (conv): Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    )
    (6): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (7): LeakyReLU(negative_slope=0.2, inplace=True)
    (8): PrunedConv(
      (conv): Conv2d(256, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    )
    (9): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (10): LeakyReLU(negative_slope=0.2, inplace=True)
    (11): PrunedConv(
      (conv): Conv2d(512, 1, kernel_size=(4, 4), stride=(1, 1), bias=False)
    )
    (12): Sigmoid()
  )
)
```

NC STATE

DCGAN architecture modified for pruning

```
Generator(
  (main): Sequential(
    (0): PrunedConvTrans(
      (conv): ConvTranspose2d(100, 512, kernel_size=(4, 4), stride=(1, 1), bias=False)
    )
    (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU(inplace=True)
    (3): PrunedConvTrans(
      (conv): ConvTranspose2d(512, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    )
    (4): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (5): ReLU(inplace=True)
    (6): PrunedConvTrans(
      (conv): ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    )
    (7): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (8): ReLU(inplace=True)
    (9): PrunedConvTrans(
      (conv): ConvTranspose2d(128, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    )
    (10): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (11): ReLU(inplace=True)
    (12): PrunedConvTrans(
      (conv): ConvTranspose2d(64, 1, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    )
    (13): Tanh()
  )
)
```

NC STATE

Sensitivity Pruning

```
Layer id        Type            Parameter       Non-zero parameter      Sparsity(\%)
1024
1               Convolutional   1024            443                     0.567383
131072
2               Convolutional   131072          58897                   0.550652
3               BatchNorm       N/A             N/A                     N/A
524288
4               Convolutional   524288          237451                  0.547098
5               BatchNorm       N/A             N/A                     N/A
2097152
6               Convolutional   2097152         947728                  0.548088
7               BatchNorm       N/A             N/A                     N/A
8192
8               Convolutional   8192            3787                    0.537720
Total nonzero parameters: 1248306
Total parameters: 2761728
Total sparsity: 0.547998
0.5479982098164627
```

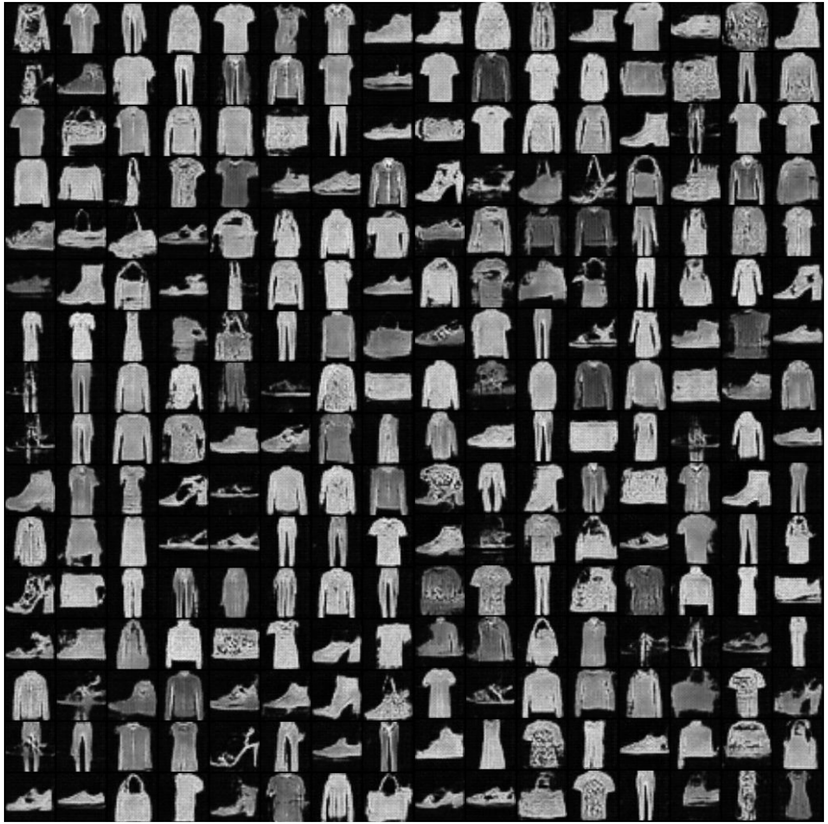
NC STATE

Sensitivity Pruning

```
Layer id        Type            Parameter       Non-zero parameter      Sparsity(\%)
819200
1               Convolutional   819200          335638                  0.590286
2               BatchNorm       N/A             N/A                     N/A
3               ReLU            N/A             N/A                     N/A
2097152
4               Convolutional   2097152         936944                  0.553230
5               BatchNorm       N/A             N/A                     N/A
6               ReLU            N/A             N/A                     N/A
524288
7               Convolutional   524288          224538                  0.571728
8               BatchNorm       N/A             N/A                     N/A
9               ReLU            N/A             N/A                     N/A
131072
10              Convolutional   131072          50450                   0.615097
11              BatchNorm       N/A             N/A                     N/A
12              ReLU            N/A             N/A                     N/A
1024
13              Convolutional   1024            396                     0.613281
Total nonzero parameters: 1547966
Total parameters: 3572736
Total sparsity: 0.566728
0.5667281321653769
```

Generated Images Before Sensitivity Pruning

Generated Images After Sensitivity Pruning



NC STATE

Sensitivity Pruning

	UQI	SSIM	MSSSIM
Mean	0.922090	1.0+0.0j	1.0
Median	0.930414	1.0+0.0j	1.0
Mode	0.731951	1.0+0.0j	(1.0, 1.0)
Max	0.979795	1.0+0.0j	1.0
Min	0.731951	1.0+0.0j	1.0
Variance	0.001521	0.0+0.0j	0.0
Std	0.038994	0.0+0.0j	0.0

s=0.75,
Sparsity=0.5667

Different degrees of Sensitivity Pruning

S = 0.25, 0.5, 0.75, 1.0, 1.25
Sparsity = 0.208, 0.401, 0.566, 0.698, 0.798

	UQI	SSIM	MSSSIM
Mean	0.995695	1.0+0.0j	1.0
Median	0.995861	1.0+0.0j	1.0
Mode	0.983032	1.0+0.0j	(1.0, 1.0)
Max	0.999576	1.0+0.0j	1.0
Min	0.983032	1.0+0.0j	1.0
Variance	0.000006	0.0+0.0j	0.0
Std	0.002495	0.0+0.0j	0.0

	UQI	SSIM	MSSSIM
Mean	0.967296	1.0+0.0j	1.0
Median	0.969074	1.0+0.0j	1.0
Mode	0.896409	1.0+0.0j	(1.0, 1.0)
Max	0.995776	1.0+0.0j	1.0
Min	0.896409	1.0+0.0j	1.0
Variance	0.000339	0.0+0.0j	0.0
Std	0.018416	0.0+0.0j	0.0

	UQI	SSIM	MSSSIM
Mean	0.920053	1.0+0.0j	1.0
Median	0.929079	1.0+0.0j	1.0
Mode	0.683314	1.0+0.0j	(1.0, 1.0)
Max	0.987823	1.0+0.0j	1.0
Min	0.683314	1.0+0.0j	1.0
Variance	0.001951	0.0+0.0j	0.0
Std	0.044171	0.0+0.0j	0.0

	UQI	SSIM	MSSSIM
Mean	0.824658	1.0+0.0j	1.0
Median	0.829120	1.0+0.0j	1.0
Mode	0.509161	1.0+0.0j	(1.0, 1.0)
Max	0.956412	1.0+0.0j	1.0
Min	0.509161	1.0+0.0j	1.0
Variance	0.004359	0.0+0.0j	0.0
Std	0.066026	0.0+0.0j	0.0

	UQI	SSIM	MSSSIM
Mean	0.666122	1.0+0.0j	1.0
Median	0.656245	1.0+0.0j	1.0
Mode	0.313283	1.0+0.0j	(1.0, 1.0)
Max	0.929446	1.0+0.0j	1.0
Min	0.313283	1.0+0.0j	1.0
Variance	0.012498	0.0+0.0j	0.0
Std	0.111793	0.0+0.0j	0.0

NC STATE

Percentage Pruning

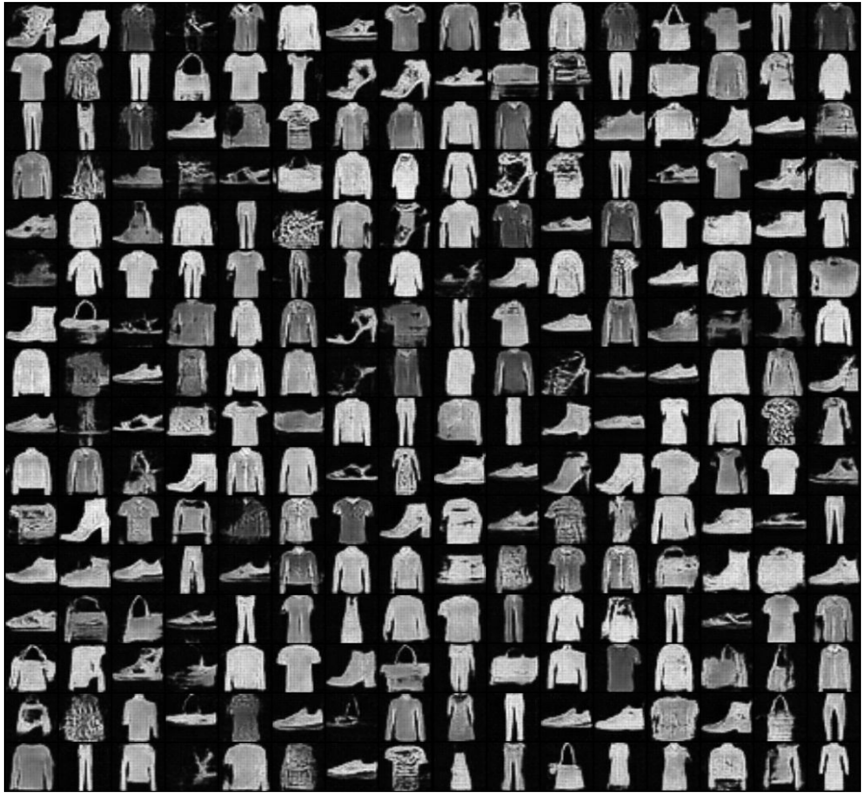
```
Layer id        Type            Parameter       Non-zero parameter      Sparsity(\%)
819200
1               Convolutional   819200          409600                  0.500000
2               BatchNorm       N/A             N/A                     N/A
3               ReLU            N/A             N/A                     N/A
2097152
4               Convolutional   2097152         1048576                 0.500000
5               BatchNorm       N/A             N/A                     N/A
6               ReLU            N/A             N/A                     N/A
524288
7               Convolutional   524288          262144                  0.500000
8               BatchNorm       N/A             N/A                     N/A
9               ReLU            N/A             N/A                     N/A
131072
10              Convolutional   131072          65536                   0.500000
11              BatchNorm       N/A             N/A                     N/A
12              ReLU            N/A             N/A                     N/A
1024
13              Convolutional   1024            512                     0.500000
Total nonzero parameters: 1786368
Total parameters: 3572736
Total sparsity: 0.500000
0.5
```

Percentage Pruning

```
Layer id        Type            Parameter       Non-zero parameter      Sparsity(\%)
1024
1               Convolutional   1024            512                     0.500000
131072
2               Convolutional   131072          65536                   0.500000
3               BatchNorm       N/A             N/A                     N/A
524288
4               Convolutional   524288          262144                  0.500000
5               BatchNorm       N/A             N/A                     N/A
2097152
6               Convolutional   2097152         1048576                 0.500000
7               BatchNorm       N/A             N/A                     N/A
8192
8               Convolutional   8192            4096                    0.500000
Total nonzero parameters: 1380864
Total parameters: 2761728
Total sparsity: 0.500000
0.5
```

Generated Images Before Percentage Pruning

Generated Images After Percentage Pruning



Percentage Pruning

	UQI	SSIM	MSSSIM
Mean	0.950989	1.0+0.0j	1.0
Median	0.954953	1.0+0.0j	1.0
Mode	0.859114	1.0+0.0j	(1.0, 1.0)
Max	0.993308	1.0+0.0j	1.0
Min	0.859114	1.0+0.0j	1.0
Variance	0.000772	0.0+0.0j	0.0
Std	0.027780	0.0+0.0j	0.0

q=50
Sparsity=0.50

NC STATE

Different degrees of Percentage Pruning

	UQI	SSIM	MSSSIM
Mean	0.996848	1.0+0.0j	1.0
Median	0.997203	1.0+0.0j	1.0
Mode	0.988448	1.0+0.0j	(1.0, 1.0)
Max	0.999628	1.0+0.0j	1.0
Min	0.988448	1.0+0.0j	1.0
Variance	0.000004	0.0+0.0j	0.0
Std	0.002015	0.0+0.0j	0.0

	UQI	SSIM	MSSSIM
Mean	0.970894	1.0+0.0j	1.0
Median	0.973217	1.0+0.0j	1.0
Mode	0.869624	1.0+0.0j	(1.0, 1.0)
Max	0.995376	1.0+0.0j	1.0
Min	0.869624	1.0+0.0j	1.0
Variance	0.000306	0.0+0.0j	0.0
Std	0.017494	0.0+0.0j	0.0

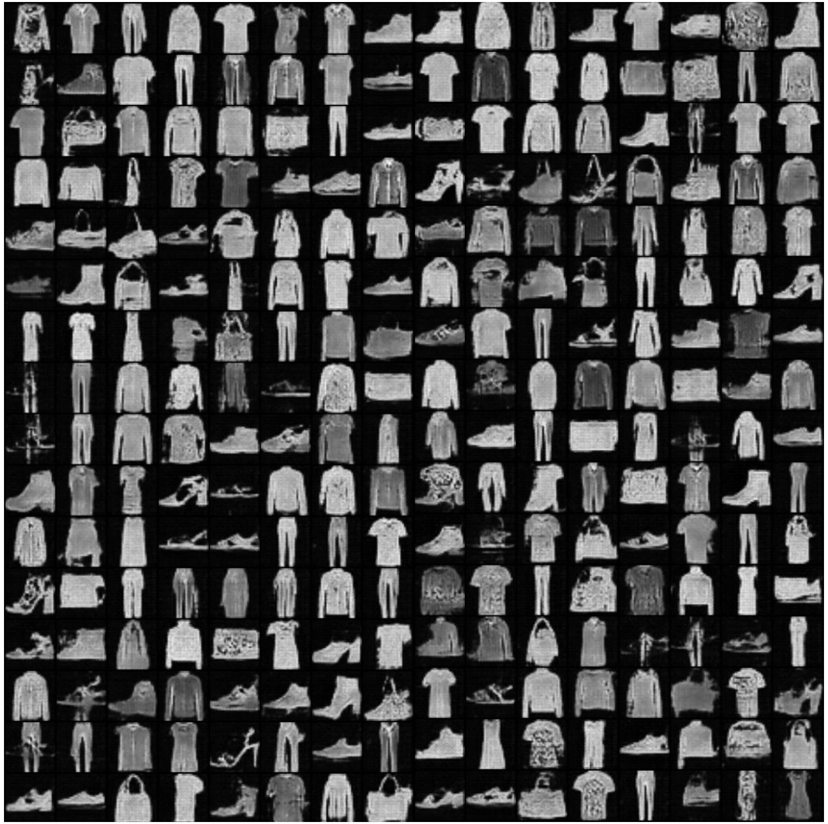
q=20, 40, 60, 80
Sparsity=0.20, 0.40, 0.60, 0.80
Respectively

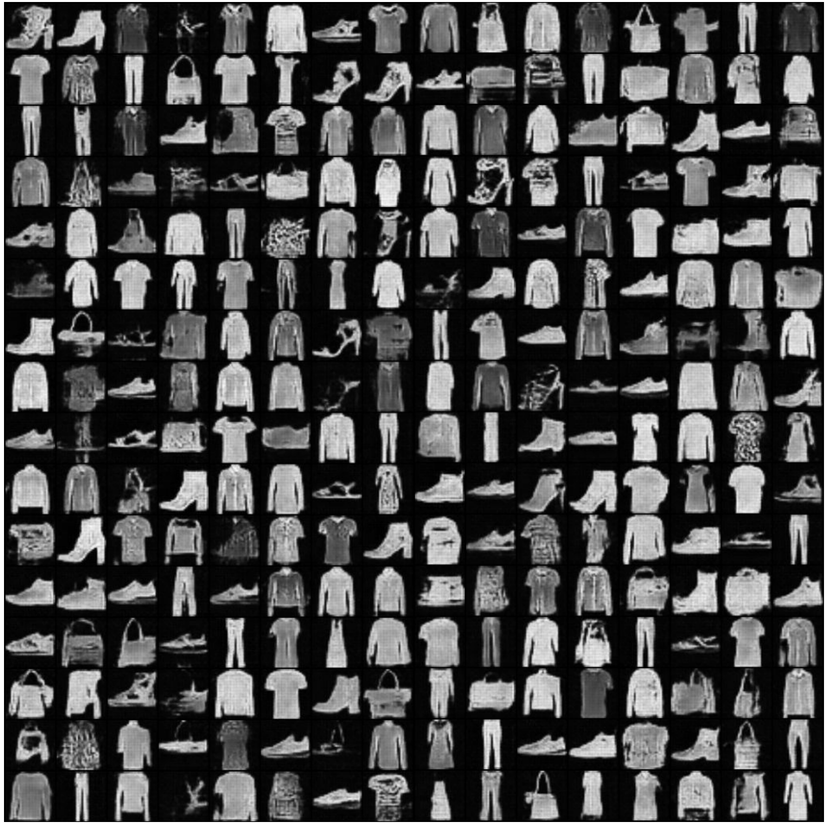
	UQI	SSIM	MSSSIM
Mean	0.919971	1.0+0.0j	1.0
Median	0.928008	1.0+0.0j	1.0
Mode	0.692482	1.0+0.0j	(1.0, 1.0)
Max	0.985625	1.0+0.0j	1.0
Min	0.692482	1.0+0.0j	1.0
Variance	0.001971	0.0+0.0j	0.0
Std	0.044400	0.0+0.0j	0.0

	UQI	SSIM	MSSSIM
Mean	0.704175	1.0+0.0j	1.0
Median	0.707740	1.0+0.0j	1.0
Mode	0.398456	1.0+0.0j	(1.0, 1.0)
Max	0.925480	1.0+0.0j	1.0
Min	0.398456	1.0+0.0j	1.0
Variance	0.010979	0.0+0.0j	0.0
Std	0.104783	0.0+0.0j	0.0

NC STATE

Sensitivity vs Percentage Pruning





Conclusion

- From our experiments we observe that for a standard DCGAN architecture trained on a relatively simple dataset, we can successfully introduce sparsity while preserving the quality of the generated Images.
- We observe the effects of sensitivity and percentage pruning at varying degrees of sparsity.
- We notice that while there is a noticeable decrease in UQI, the SSIM and MSSSIM remain strong, alluding to the fact that generated images preserve their structural integrity despite the introduction of varying degrees of sparsity in the generator

NC STATE

Challenges Faced

- We needed to arrive at the right learning rates for the discriminator and generator independently to ensure synchronization
- Training DCGANs required us to baby sit the model during training, since we encountered mode collapse with several hyperparameter settings and kernel sizes
- Limited GPU resources prevented us from exploring architectures beyond the standard DCGAN, and hence also bigger and more complex datasets
- FID and IS scores were not possible to compute effectively since the standard DCGAN architecture was modified to handle grayscale images and the pretrained architectures required to compute FID and IS scores do not handle this

Future Work

- Different and better architectures of GANs can be experimented on for analyzing the effects of model pruning
- This would also allow the application of GANs on bigger and more complex datasets
- The effects of model compression techniques (not limited to pruning) such as quantization and huffman encoding could also be explored in the context of GANs

NC STATE

THANK YOU

NC STATE