Movie Reviews dataset from Kaggle to identify the sentiments.

a. Implemented text preprocessing steps.

Applied various preprocessing techniques, such as tokenization, removing stop words, stripping HTML, converting to lowercase, and lemmatization/stemming.

b. Utilize a combination of Word2Vec and Sequential models: RNN, LSTM, GRU, and BiLSTM.

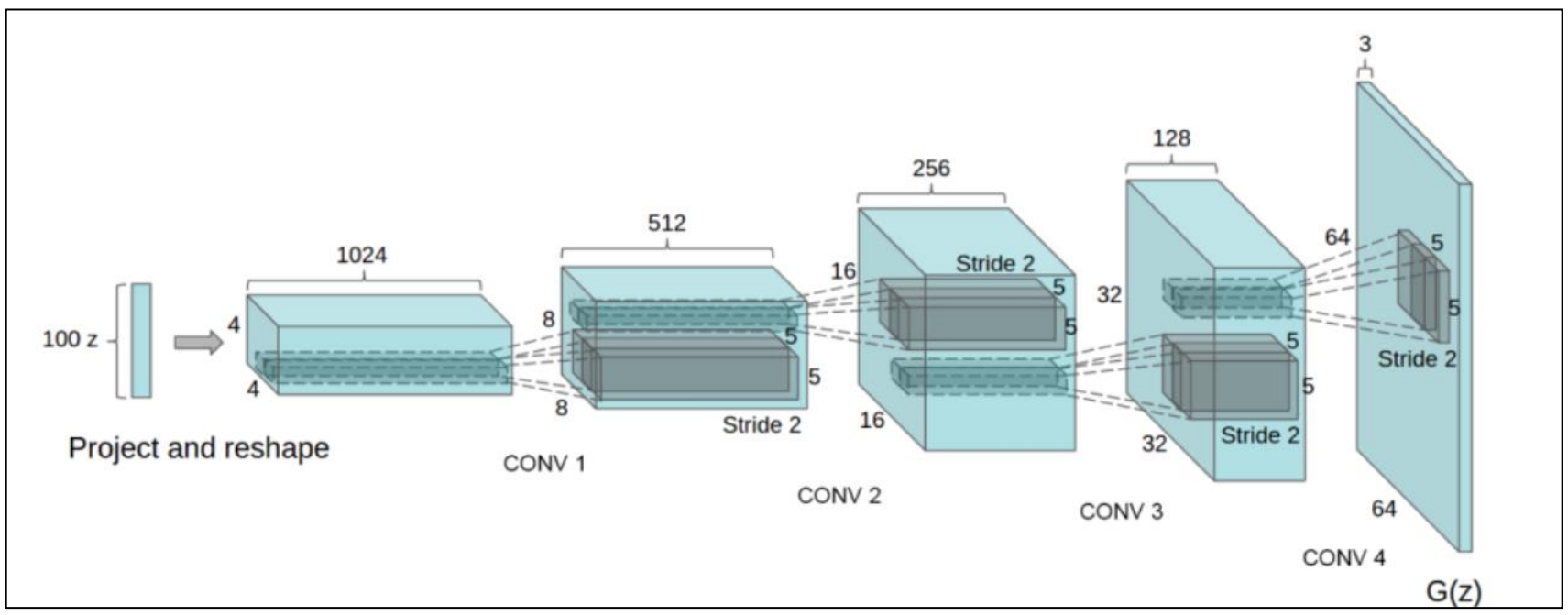
Findings -

- **BiLSTM with Word2Vec -> best with 71.5% accuracy in just 9 epochs**
- **LSTM with Word2Vec -> second-best at 69.5% accuracy but required more training (23 epochs)**
- **GRU with Word2Vec -> moderate results (54.1% accuracy)**
- **Basic RNN with Word2Vec -> performed worst at 50.5% accuracy**
- **Complex models (BiLSTM, LSTM) outperformed simpler ones (RNN)**

Apply Deep Convolutional Generative Adversarial Networks (DCGAN).

Number of epochs - 1000.

The architecture of the generator.



Discriminator architecture with minimum of 3 convolutional layers.

I have used "PathMNIST" dataset.

Generator Architecture

```
Generator(
  (project): Linear(in_features=100, out_features=16384, bias=True)
  (conv1): Sequential(
    (0): ConvTranspose2d(1024, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.2, inplace=True)
  )
  (conv2): Sequential(
    (0): ConvTranspose2d(512, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.2, inplace=True)
  )
  (conv3): Sequential(
    (0): ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.2, inplace=True)
  )
  (conv4): Sequential(
    (0): ConvTranspose2d(128, 3, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): Tanh()
  )
)
```

Discriminator Architecture

```
Discriminator(
  (conv1): Sequential(
    (0): Conv2d(3, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): LeakyReLU(negative_slope=0.2, inplace=True)
    (2): Dropout2d(p=0.2, inplace=False)
  )
  (conv2): Sequential(
    (0): Conv2d(64, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.2, inplace=True)
    (3): Dropout2d(p=0.2, inplace=False)
  )
  (conv3): Sequential(
    (0): Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.2, inplace=True)
    (3): Dropout2d(p=0.2, inplace=False)
  )
  (conv4): Sequential(
    (0): Conv2d(256, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.2, inplace=True)
    (3): Dropout2d(p=0.2, inplace=False)
  )
  (final): Conv2d(512, 1, kernel_size=(4, 4), stride=(1, 1), bias=False)
  (attention): Sequential(
    (0): Conv2d(256, 1, kernel_size=(1, 1), stride=(1, 1), bias=False)
    (1): Sigmoid()
  )
)
```

```
Starting training...
Epoch 0/1000: 100%|| 704/704 [00:44<00:00, 15.73it/s, D_loss=-1.61, G_loss=3.48]
Epoch 1/1000: 100%|| 704/704 [00:43<00:00, 16.29it/s, D_loss=-.245, G_loss=-.134]
Epoch 2/1000: 100%|| 704/704 [00:43<00:00, 16.31it/s, D_loss=1.62, G_loss=4.79]
Epoch 3/1000: 100%|| 704/704 [00:43<00:00, 16.27it/s, D_loss=-.67, G_loss=7.47]
Epoch 4/1000: 100%|| 704/704 [00:43<00:00, 16.26it/s, D_loss=-2.59, G_loss=4.5]
Epoch 5/1000: 100%|| 704/704 [00:43<00:00, 16.27it/s, D_loss=0.779, G_loss=3.57]
Epoch 6/1000: 100%|| 704/704 [00:43<00:00, 16.24it/s, D_loss=-2.24, G_loss=12.2]
Epoch 7/1000: 100%|| 704/704 [00:43<00:00, 16.25it/s, D_loss=-.117, G_loss=9.14]
Epoch 8/1000: 100%|| 704/704 [00:43<00:00, 16.12it/s, D_loss=-.53, G_loss=14.1]
Epoch 9/1000: 100%|| 704/704 [00:43<00:00, 16.23it/s, D_loss=0.782, G_loss=13.7]
Epoch 10/1000: 100%|| 704/704 [00:43<00:00, 16.26it/s, D_loss=-2.84, G_loss=9.32]
Epoch 11/1000: 100%|| 704/704 [00:43<00:00, 16.28it/s, D_loss=-1.06, G_loss=8.82]
Epoch 12/1000: 100%|| 704/704 [00:44<00:00, 15.96it/s, D_loss=-.617, G_loss=14.9]
Epoch 13/1000: 100%|| 704/704 [00:45<00:00, 15.46it/s, D_loss=1.37, G_loss=15]
Epoch 14/1000: 100%|| 704/704 [00:44<00:00, 15.65it/s, D_loss=-.11, G_loss=16.7]
Epoch 15/1000: 100%|| 704/704 [00:45<00:00, 15.60it/s, D_loss=-1.41, G_loss=17.1]
Epoch 16/1000: 100%|| 704/704 [00:45<00:00, 15.53it/s, D_loss=-.966, G_loss=13.6]
Epoch 17/1000: 100%|| 704/704 [00:45<00:00, 15.61it/s, D_loss=1.3, G_loss=15.5]
Epoch 18/1000: 100%|| 704/704 [00:45<00:00, 15.62it/s, D_loss=-.942, G_loss=15.1]
Epoch 19/1000: 100%|| 704/704 [00:45<00:00, 15.64it/s, D_loss=-.68, G_loss=13.2]
Epoch 20/1000: 100%|| 704/704 [00:45<00:00, 15.52it/s, D_loss=-4.58, G_loss=12.8]
Epoch 21/1000: 100%|| 704/704 [00:44<00:00, 15.87it/s, D_loss=-.356, G_loss=13.3]
Epoch 22/1000: 100%|| 704/704 [00:43<00:00, 16.27it/s, D_loss=-.621, G_loss=16.7]
Epoch 23/1000: 100%|| 704/704 [00:43<00:00, 16.25it/s, D_loss=2.09, G_loss=13.9]
Epoch 24/1000: 100%|| 704/704 [00:43<00:00, 16.29it/s, D_loss=-3.54, G_loss=18.2]
Epoch 25/1000: 100%|| 704/704 [00:43<00:00, 16.28it/s, D_loss=0.135, G_loss=17.7]
Epoch 26/1000: 100%|| 704/704 [00:43<00:00, 16.22it/s, D_loss=2.51, G_loss=16.3]
Epoch 27/1000: 100%|| 704/704 [00:43<00:00, 16.30it/s, D_loss=-1.09, G_loss=17.1]
Epoch 28/1000: 100%|| 704/704 [00:43<00:00, 16.27it/s, D_loss=0.675, G_loss=21.5]
Epoch 29/1000: 100%|| 704/704 [00:43<00:00, 16.29it/s, D_loss=0.418, G_loss=19.1]
Epoch 30/1000: 100%|| 704/704 [00:43<00:00, 16.29it/s, D_loss=-1.5, G_loss=18.2]
Epoch 31/1000: 100%|| 704/704 [00:43<00:00, 16.24it/s, D_loss=0.98, G_loss=17.2]
Epoch 32/1000: 100%|| 704/704 [00:43<00:00, 16.26it/s, D_loss=0.0404, G_loss=17.1]
Epoch 33/1000: 100%|| 704/704 [00:43<00:00, 16.28it/s, D_loss=-1.61, G_loss=18.6]
Epoch 34/1000: 100%|| 704/704 [00:43<00:00, 16.18it/s, D_loss=-.0903, G_loss=19.4]
Epoch 35/1000: 100%|| 704/704 [00:44<00:00, 15.71it/s, D_loss=0.261, G_loss=18]
Epoch 36/1000: 100%|| 704/704 [00:44<00:00, 15.98it/s, D_loss=-2.27, G_loss=22.2]
Epoch 37/1000: 100%|| 704/704 [00:43<00:00, 16.00it/s, D_loss=1.31, G_loss=20.5]
```

FID score after every 100 epochs.

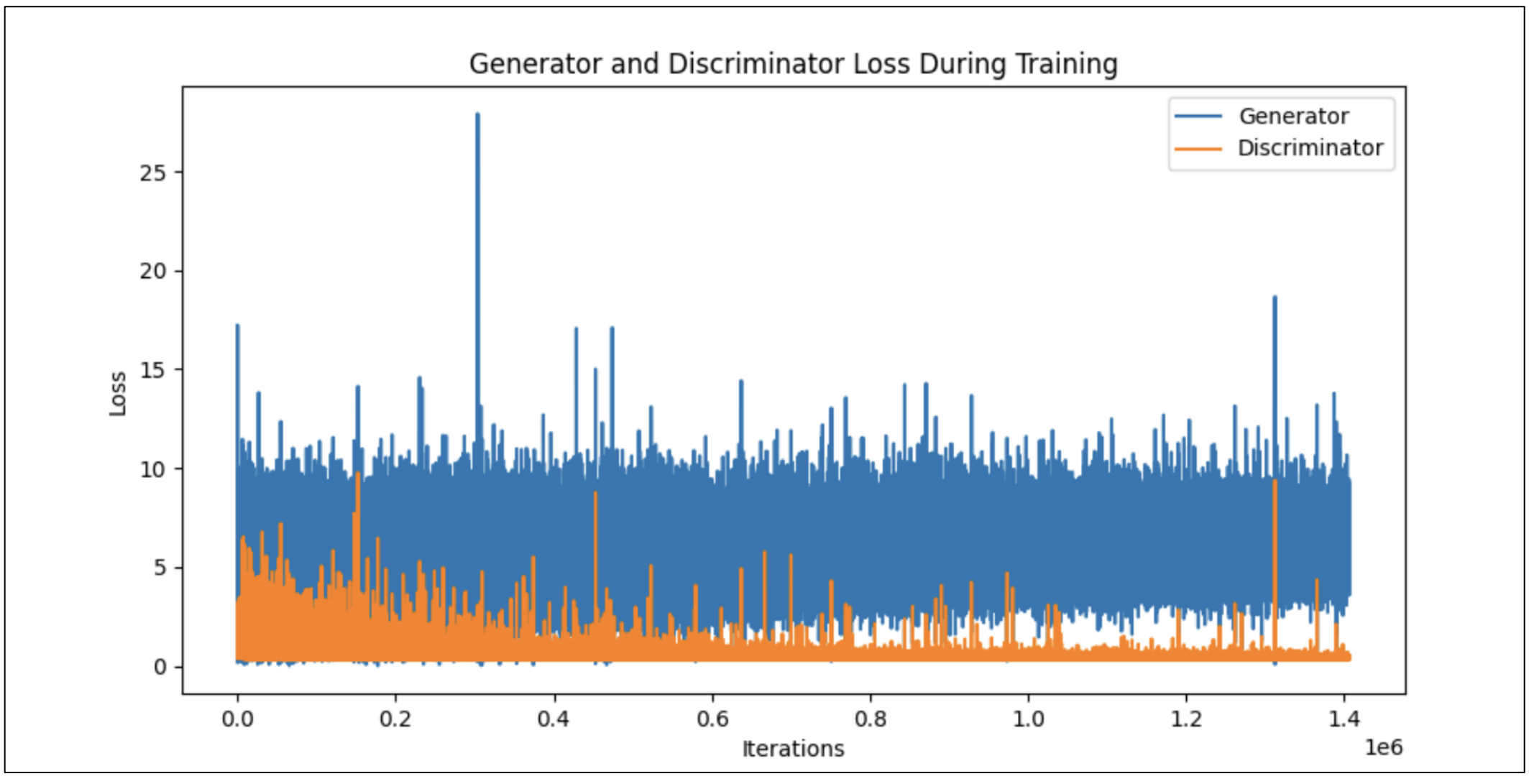
```
Epoch 95/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.27it/s, D_loss=-1.49, G_loss=21.2]
Epoch 96/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.26it/s, D_loss=2.6, G_loss=20.3]
Epoch 97/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.28it/s, D_loss=-.437, G_loss=22.8]
Epoch 98/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.26it/s, D_loss=2.9, G_loss=21.2]
Epoch 99/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.24it/s, D_loss=2.13, G_loss=23.1]
Epoch 100/1000: 100%|███████████████████| 704/704 [00:43<00:00, 16.25it/s, D_loss=-2.19, G_loss=23.9]
Calculating FID score for epoch 100...
Calculating FID score...
Epoch 100: FID Score = 225.2728
New best model saved with FID: 225.2728
Epoch 101/1000: 100%|███████████████████| 704/704 [00:44<00:00, 15.96it/s, D_loss=1.15, G_loss=23.7]
Epoch 102/1000: 100%|███████████████████| 704/704 [00:43<00:00, 16.19it/s, D_loss=-.803, G_loss=23.2]
Epoch 103/1000: 100%|███████████████████| 704/704 [00:43<00:00, 16.28it/s, D_loss=4.78, G_loss=24.9]
```

```
Epoch 987/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.30it/s, D_loss=-15.2, G_loss=-57.8]
Epoch 988/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.30it/s, D_loss=-5.25, G_loss=-46.9]
Epoch 989/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.30it/s, D_loss=-605, G_loss=-50.1]
Epoch 990/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.29it/s, D_loss=3.8e+4, G_loss=-65.7]
Epoch 991/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.29it/s, D_loss=17.7, G_loss=-643]
Epoch 992/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.29it/s, D_loss=6.83e+3, G_loss=-62.3]
Epoch 993/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.25it/s, D_loss=-6.04e+3, G_loss=-71]
Epoch 994/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.27it/s, D_loss=5.79e+3, G_loss=64.2]
Epoch 995/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.32it/s, D_loss=2.62e+3, G_loss=-55.9]
Epoch 996/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.26it/s, D_loss=-74.9, G_loss=-60.4]
Epoch 997/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.27it/s, D_loss=17.4, G_loss=-9.78]
Epoch 998/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.29it/s, D_loss=-8.5, G_loss=-167]
Epoch 999/1000: 100%|████████████████████| 704/704 [00:43<00:00, 16.28it/s, D_loss=39.4, G_loss=-71.2]
Calculating FID score for epoch 999...
Calculating FID score...
Epoch 999: FID Score = 220.9184
```

Final FID Score is = 220.91

Note: I tried fine tuning and ran all 1000 epochs thrice I was able to reduce the FID score from 784 -> 340 -> 220.

The learning curve including generator and discriminator loss and discussed presence of mode collapse issue



Learning Curve Analysis

Generator Loss:

- **Overall Trend**: Stable range between 5-15 for most of the training, with few spikes up to 28.
- **Initial Phase**: Higher variance in the early iterations (0-0.2M)
- **Stability**: After the initial phase -> Loss stabilizes with consistent oscillation patterns
- **Spikes**: Spikes around 0.3M, 0.45M and 1.25M iterations -> indicate temporary instability in training
- **End Behavior**: Generator continues to learn throughout training without signs of convergence to a single value

Discriminator Loss:

- **Overall Trend**: Gradual decrease and stabilization
- **Initial Phase**: Higher values (4-7) with many fluctuations
- **Middle Phase**: Gradually decreases and becomes more stable
- **Final Phase**: After approx. 0.8M iterations, the discriminator loss stabilizes at lower values (around 1-2)
- **Occasional Spikes**: Similar to the generator -> occasional spikes around 0.45M and 1.25M iterations

Mode Collapse Issue

Mode collapse is when a GAN learns to generate only a small subset of possible outputs rather than the full range of diverse examples present in the training data.

Basically, when the generator forgets how to generate certain types of samples and instead focuses on producing a few types that can consistently fool the discriminator.

Following observations shows Mode Collapse:

1. **Sustained Generator Oscillations:** Generator loss fluctuating throughout training rather than flattening to a constant value -> This behavior is expected mode collapse.
2. **Healthy Loss Range**: Range of loss values are 5-15 and are not converging to zero -> indicates still learning diverse patterns.
3. **Discriminator Stabilization:** Discriminator loss gradually decreases and stabilizes at non-zero values -> shows that it is still able to differentiate between real and fake samples.

So basically,

- sharp spikes in generator loss indicate moments -> where the generator briefly struggled.
- very low and stable discriminator loss in the later stages -> indicate the discriminator becoming too confident
  -> precedes mode collapse.

This shows that generator continues to challenge the discriminator throughout training and the discriminator continues to provide good feedback.

- **Calculated FID score: used minimum 1000 real and 1000 generated images. Determined mode collapse based on the FID score.**

So I calculated FID score after each 100 epochs. My FID scores were like –

Epochs	FID Score
100	225.2728
200	228.1769
300	187.1703
400	287.9670
500	247.7154
600	261.9076
700	306.7651
800	243.8183

900	240.5490
999	220.9184

Lower FID scores = Better -> Indicates high quality and diversity in the generated images

Observations –

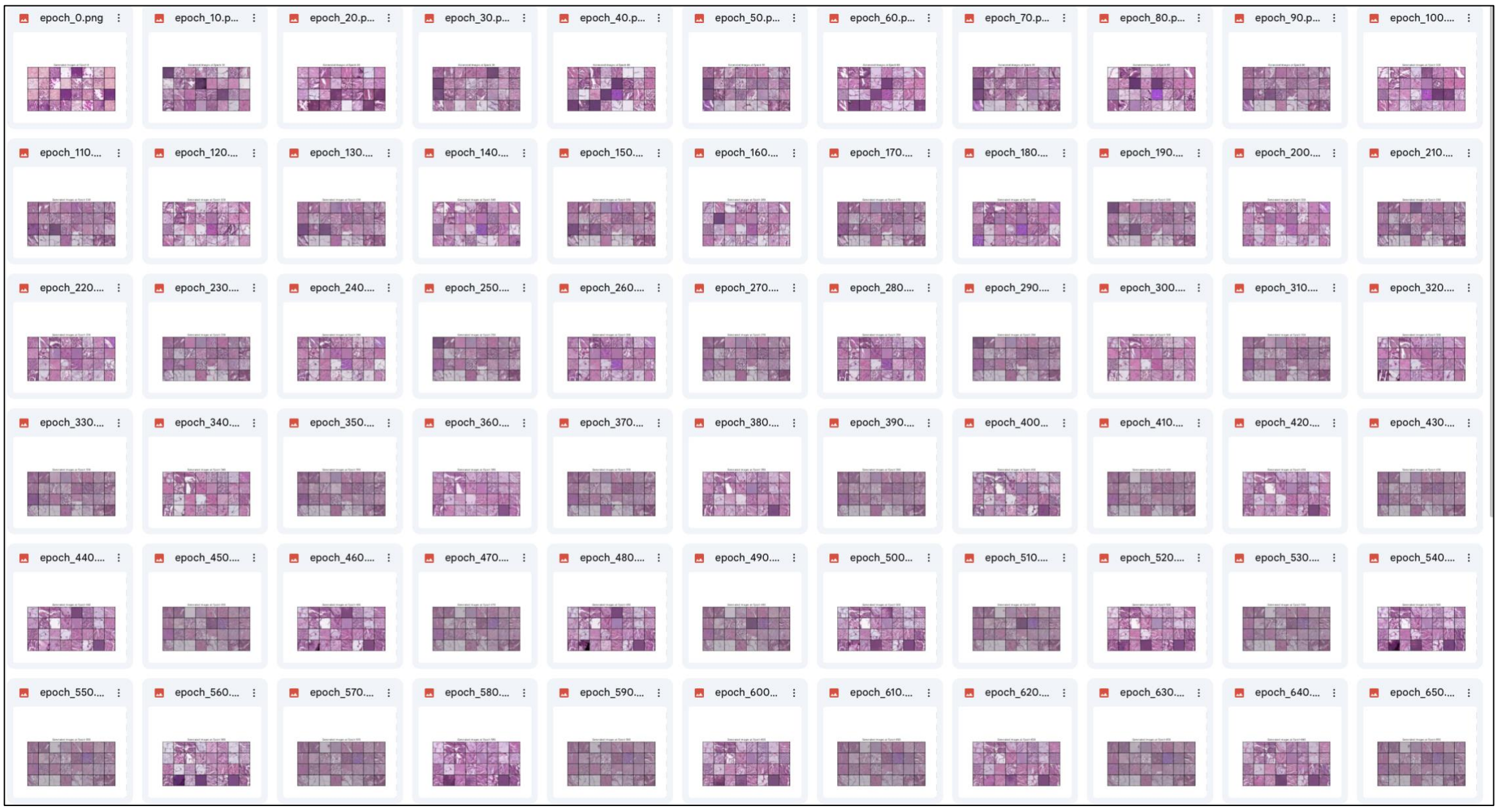
- Lowest FID score of 187.17 at epoch 300
- Then increased significantly to 287.97 at epoch 400
- Worst performance with an FID of 306.77
- From epoch 700 to 999, there is a gradual improvement in FID scores-> decreased from 306.77 to 220.92
- Scores fluctuate throughout training -> did not show downward trend

Mode Collapse based on FID score:

1. All FID scores are above 180 -> which is high compared.
2. Large variations in FID scores (from 187 to 307) -> unstable training dynamics -> mode collapse issue.
3. Sharp increase in FID after epoch 300 -> suggests the generator might have started focusing on a narrower range of patterns -> maybe discarding some modes it had previously learned.
4. FID improved in the final epochs BUT it never returned to the best value (187.17) -> model has not fully recovered its earlier diversity.

• After completing the training process, generated 32 images and mentioned whether there is any issue of mode collapse in terms nature of generated images and generator and discriminator loss curve.

I saved the generated images.



Mode Collapse based on Images

1. **Repeated patterns**:
   - Specific white/light regions -> consistently appear in the same positions across different epochs
   - Dark purple regions -> maintain similar patterns across epochs
   - Cell boundaries and tissue interfaces -> show repetitive patterns
2. **Limited Texture Variety**:
   - Struggles to generate the -> full spectrum of tissue textures present in PathMNIST

    - Granular cellular structures -> similar across multiple generated samples
3. **Pattern Consistency Across Epochs**:
    - Images from epochs 300-700 -> striking similarities in their overall composition
    - Alternating pattern of light and dark regions is consistent
    - Model seems stuck -> generated a limited subset of possible PathMNIST patterns

After looking at the loss curve and all the generated images it is pretty evident that there is mode collapse issue.