University of Burgundy

Assignment II

Style transfer with unpaired collections of images for classification

Kimberly Grace Sevillano Colina

---

1 Introduction

This assignment explores image classification and style transfer using unpaired images from two datasets: the Street View House Numbers (SVHN) and MNIST. We will leverage CycleGAN1 concepts to perform style transfer between these datasets. While SVHN provides a complex set of real-world number images, MNIST offers simpler handwritten digits. Both datasets are available in Pytorch's dataset module. The task will deepen our understanding of GAN training and its application in improving classification tasks through style transfer.

2 Part I - Environment Setup & Inputs

2.1 Environment Setup

To set up the conda environment for working on the local machine, follow the steps below:

1. Create a new conda environment: `conda create -n ML_styleTransfer`
2. Activate the environment: `conda activate ML_styleTransfer`
3. Install Jupyter Notebook: `conda install jupyter`
4. Navigate to the workspace directory of the course and run Jupyter: `jupyter-notebook`

3 Part II - ResNet-18 Backbone for Classification

3.1 Questions

3.1.1 Implement a classifier of digits 0 to 9 that has as encoder backbone the ResNet-18 model.

Below is the Python code for loading a pre-trained ResNet-18 model, removing its last fully connected layer, and replacing it with a new one for 10 classes:

```
# Load pre-trained ResNet-18 model
resnet18_pretrained = models.resnet18(pretrained=True)
# Remove the last fully connected layer (fc) and replace it with a new one for 10 classes
num_features = resnet18_pretrained.fc.in_features
resnet18_pretrained.fc = nn.Linear(num_features, 10)
```

3.1.2 Make the required changes either in the image loaders or in the network model such as that your model can process either images from MNIST and SVHN without changes. Please justify your choices.

To handle both MNIST and SVHN datasets without changing the model or loaders, I made the following adjustments:

- **Image Resizing and Grayscale Conversion :**

  Since MNIST images are 28x28 pixels and in grayscale, and SVHN images are 32x32 pixels and in color, the transform pipeline includes resizing and grayscale conversion. This ensures all images are of the same size (224x224, suitable for ResNet) and have three channels (grayscale images are converted to have three channels to mimic color images).

- **Justification :**

  1. *Resizing :* ResNet models are typically trained on ImageNet, which has images of size 224x224 pixels. Resizing MNIST and SVHN to this size allows the use of ResNet without structural changes.
  2. *Grayscale Conversion :* Since MNIST is in grayscale and SVHN in color, converting MNIST images to 3-channel grayscale allows using the same network architecture designed for color images.

This approach simplifies the training pipeline by avoiding the need for separate models or complex loader logic for each dataset. It's a practical compromise between dataset consistency and model adaptability.

3.1.3 Please check and show the histogram of the classes of your cropped dataset. Are your datasets balanced?

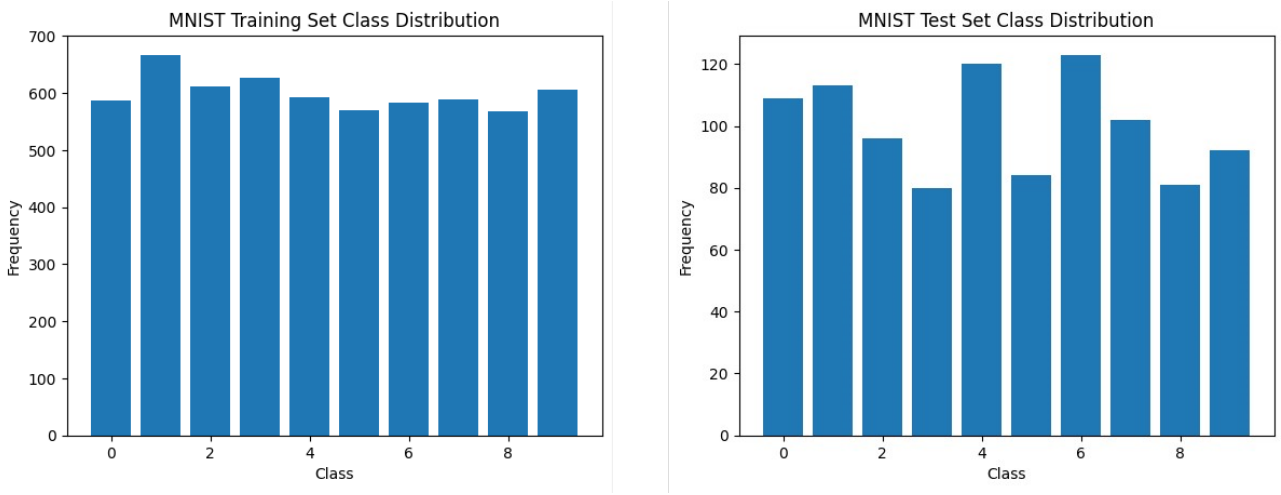


Figure 1: MNIST dataset plotting.

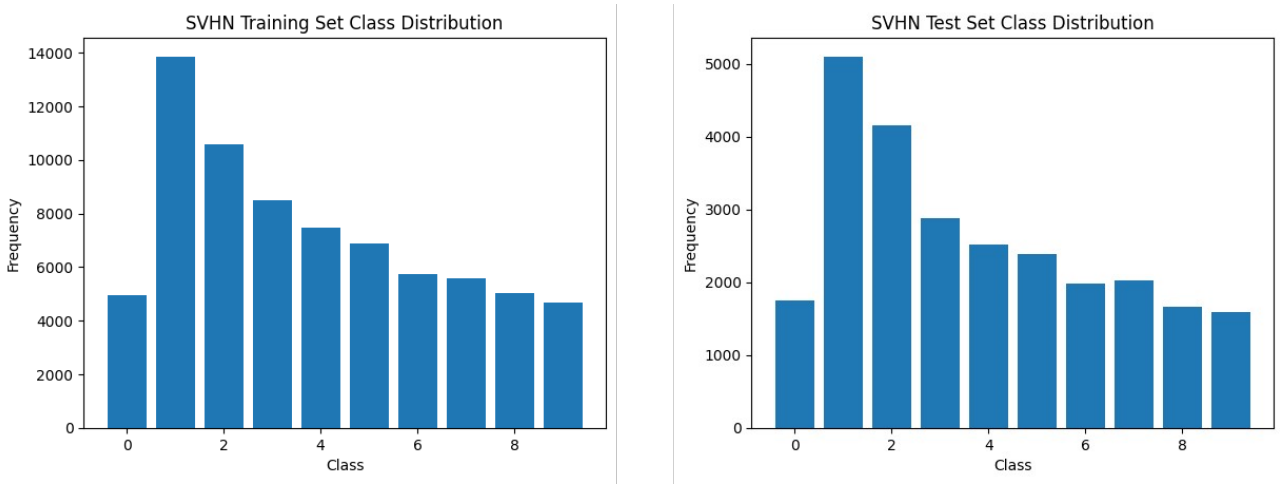


Figure 2: SVHN dataset plotting.

The MNIST dataset is balanced, while the SVHN dataset is not. To address this, I preprocessed the SVHN dataset to achieve balance, as illustrated in the graph below.

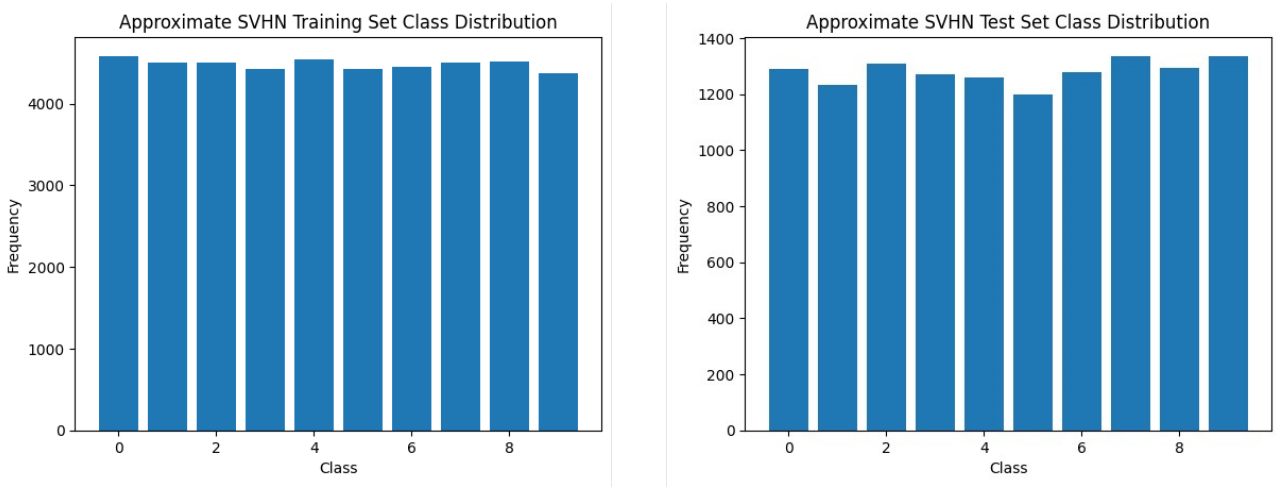


Figure 3: Plotting of SVHN balanced dataset.

3.1.4 Define two versions of your model, one that reuses the pre-trained weights of ResNet-18 and another trained from scratch. What are the classification accuracies on MNIST of both models?

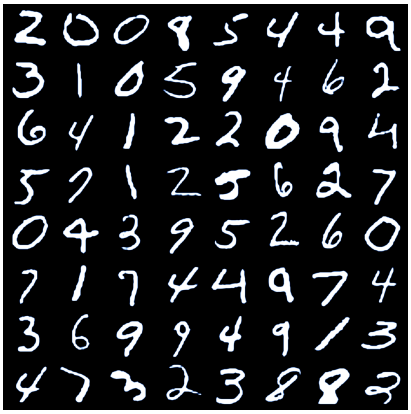


Figure 4: Images of the pre-trained MNIST dataset.

Figure 5: MNIST Dataset: Scratch Training Images.

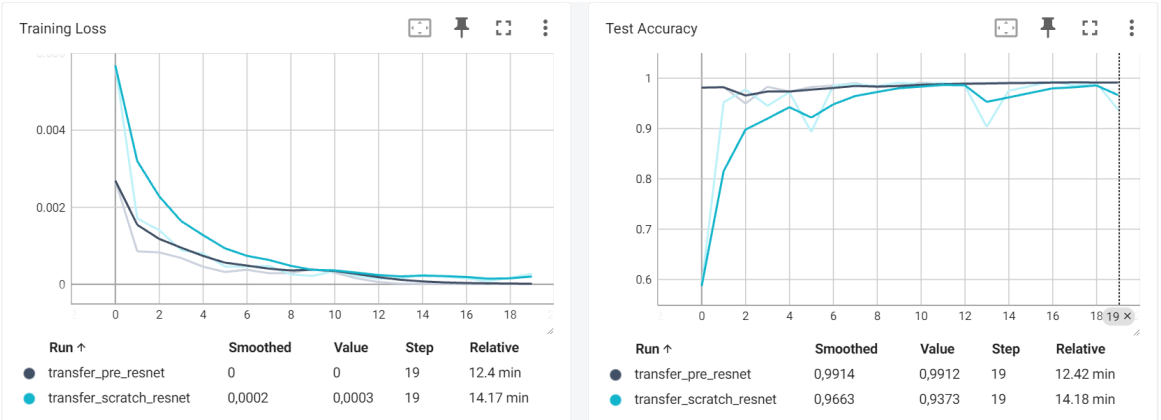


Figure 6: Comparison of loss-on-training and test accuracy plots between pre-trained and scratch-trained MNIST databases.

Comparison of Accuracy on MNIST:

Pre-trained model: Achieves a maximum accuracy of 0.9912. Model trained from scratch: Achieves a maximum accuracy of 0.9932.

3.1.5 What are the classification accuracies on SVHN of both models?

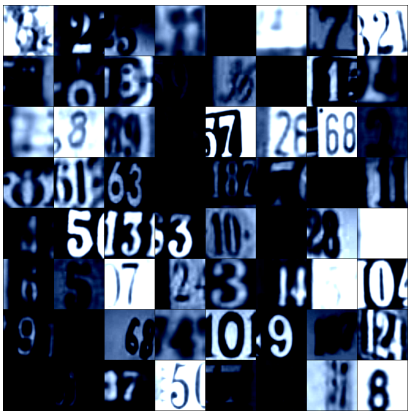


Figure 7: Images of the pre-trained SVHN dataset.

Figure 8: SVHN Dataset: Scratch Training Images.

Figure 9: Comparison of training loss and test accuracy plots between pre-trained and scratch-trained SVHN databases.

Comparison of Accuracy on SVHN:

Pre-trained model: Achieves a maximum accuracy of 0.9565. Model trained from scratch: Achieves a maximum accuracy of 0.9499.

3.1.6 Which dataset is easier to be classified and what is the configuration you would keep?

MNIST appears to be easier to classify than SVHN, as both the pre-trained and scratch-trained models achieved higher accuracies on the MNIST dataset.

- For MNIST, the model trained from scratch shows a slight improvement in accuracy compared to the pre-trained model. However, the difference is minimal and might not justify the additional training cost, especially considering the training time (14.17 minutes for scratch vs. 12.4 minutes for pre-trained).
- For SVHN, the pre-trained model outperforms the scratch-trained model in terms of accuracy, though by a small margin. However, the training time is significantly longer for the pre-trained model (2.962 hours) compared to the scratch-trained model (2.723 hours).

In summary, using pre-trained models seems to be a good choice in terms of balancing accuracy and training time, especially for SVHN. For MNIST, the scratch-trained model has a slight edge in accuracy, but one needs to consider whether the additional training time is worth it. In a production or research scenario where time and resources are crucial factors, the pre-trained model might be preferred due to its time efficiency.

3.1.7 Do you observe any underfitting or overfitting in the obtained training loss and metrics curves?

For both the MNIST and SVHN datasets, there are no clear indicators of underfitting or overfitting. Both models have learned the respective datasets effectively, with the pre-trained models showing a slight edge in terms of starting from a better point and achieving slightly higher test accuracy. The training process appears to be stable and well-tuned for both datasets.

4 Part III - Image-to-Image Translation

I will now design the CycleGAN model and evaluate the performance of my model for style transfer (also known as image-to-image translation), changing images from MNIST to look colored, similar to those in SVHN. I will also test different ways to evaluate the performance of my model.

Figure 10: Domain A (real colored samples).

Figure 11: Domain B (generated samples).

First 64 image samples of colorized MNIST data and data generated by CycleGAN.

Please compute the FID distance between two sets of images in the following scenarios:
- Domain A: real colored MNIST and domain B: generated colored MNIST
- Domain A: Original MNIST (first 1000 images of the test set) and domain B: generated colored MNIST
- Domain A: Original MNIST and domain B: SVHN, both first 500 images of the test set.

4.1 Questions

4.1.1 Which one presents the smallest FID and highest FID? Comment and analyse the results from what you would expect.

Evaluating the FID scores provided insightful comparisons between different sets of images:

1. *Real Colored MNIST (created by get_rgb) vs. CycleGAN Generated Colored MNIST (FID: 47.91):* Among the comparisons, this FID score was the lowest, indicating a closer alignment between CycleGAN's output and the Real Colored MNIST. It suggests that CycleGAN effectively captured the distribution of the colored dataset, reflecting in generated images that closely resemble the real ones in terms of statistical properties. The low FID score underlines the model's success in preserving essential features such as digit shapes and colors.

2. *Original MNIST (first 1000 images of the test set) vs. Generated Colored MNIST (FID: 160.60):*

   This score, while higher, was expected given the comparison between grayscale and colored images. The significant FID increase can be attributed to the fundamental difference in color spaces, as the CycleGAN introduces color to the originally grayscale MNIST images, adding variations that weren't present in the original set. This comparison highlights the impact of colorization on the FID score, emphasizing the role of color features captured by the Inception model in FID calculations.

3. *MNIST (test set, first 500) vs. SVHN (training set, first 500) (FID: 220.21):*

   The highest FID score in these comparisons reflects the stark contrast between MNIST's simple grayscale digits and SVHN's colorful, complex backgrounds. The substantial difference can be attributed to the intrinsic disparities in context, texture, and style between the datasets. This score illustrates the challenges in directly comparing datasets with fundamentally different characteristics, even when they share a common theme like digit representation.

In summary, the FID scores here reflect not just differences in the quality of the images generated by the models compared to real datasets, but also inherent differences in the data being compared. The results suggest that the models are relatively effective in capturing the distribution of the target datasets, though inherent differences in the source and target data can lead to higher FID values.

Note! In this part I want to emphasize that it took me much more time to try to have adequate FID results for each of the comparisons of the datasets (many times I got negative results, complex, too high, or by bad practice in the logic), many times it was because the normalization was not good, or did not fit with the inception model, so you can see that I constantly check that detail, besides having to clean my memory constantly.

4.1.2 Train your CycleGAN model to perform style transfer between the following domains:

- Domain A: Original MNIST (first 1000 images of the test set) and domain B: first 1000 images of real colored MNIST

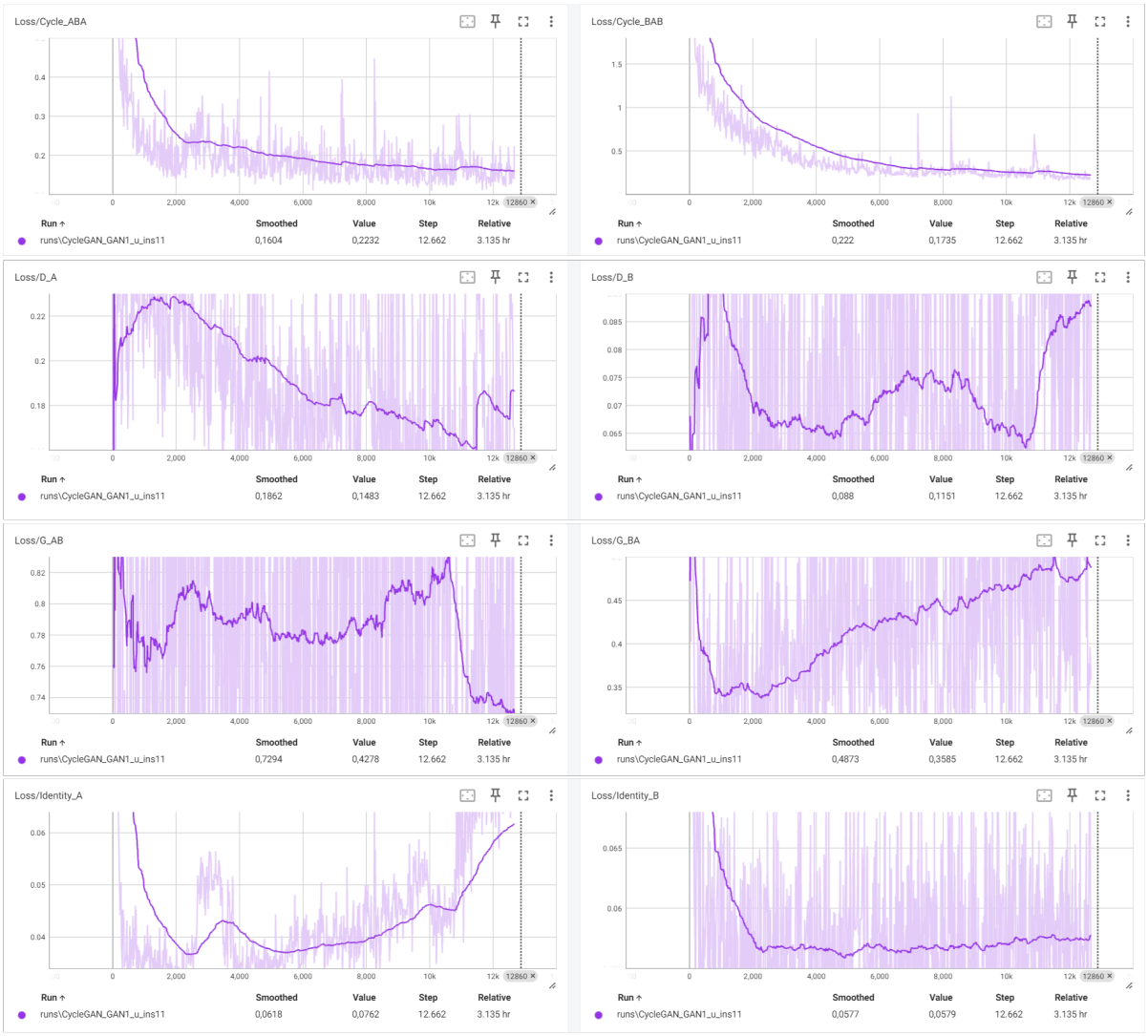


Figure 12: CycleGAN Training: Original MNIST Test Set and Real Colored MNIST (First 1000 Images).

I want to highlight in this section that I tested several more extensive and deeper UNET architectures, changing optimizers, and so on, which, according to the literature, were much more efficient.

However, due to the lack of memory, my training was not completed, even when I decreased the batch size. For this reason, I opted for a simpler model.

Although my results are not good enough, one way to improve them was to use InstanceNorm layers (see Figure 12, purple curves) instead of BatchNorm (see Figure 12, green curves). This is because, according to the literature, InstanceNorm layers perform better in Style Transfer models, as can be seen in the TensorBoard plots. Additionally, I want to add that InstanceNorm is preferable over BatchNorm in CycleGAN models for Style Transfer because InstanceNorm normalizes the data in each image independently, which allows the preservation of the independence of each image in the dataset. This is crucial for Style Transfer tasks, where the unique relationship between the content and style of each individual image is more important than the common features of the dataset as a whole.

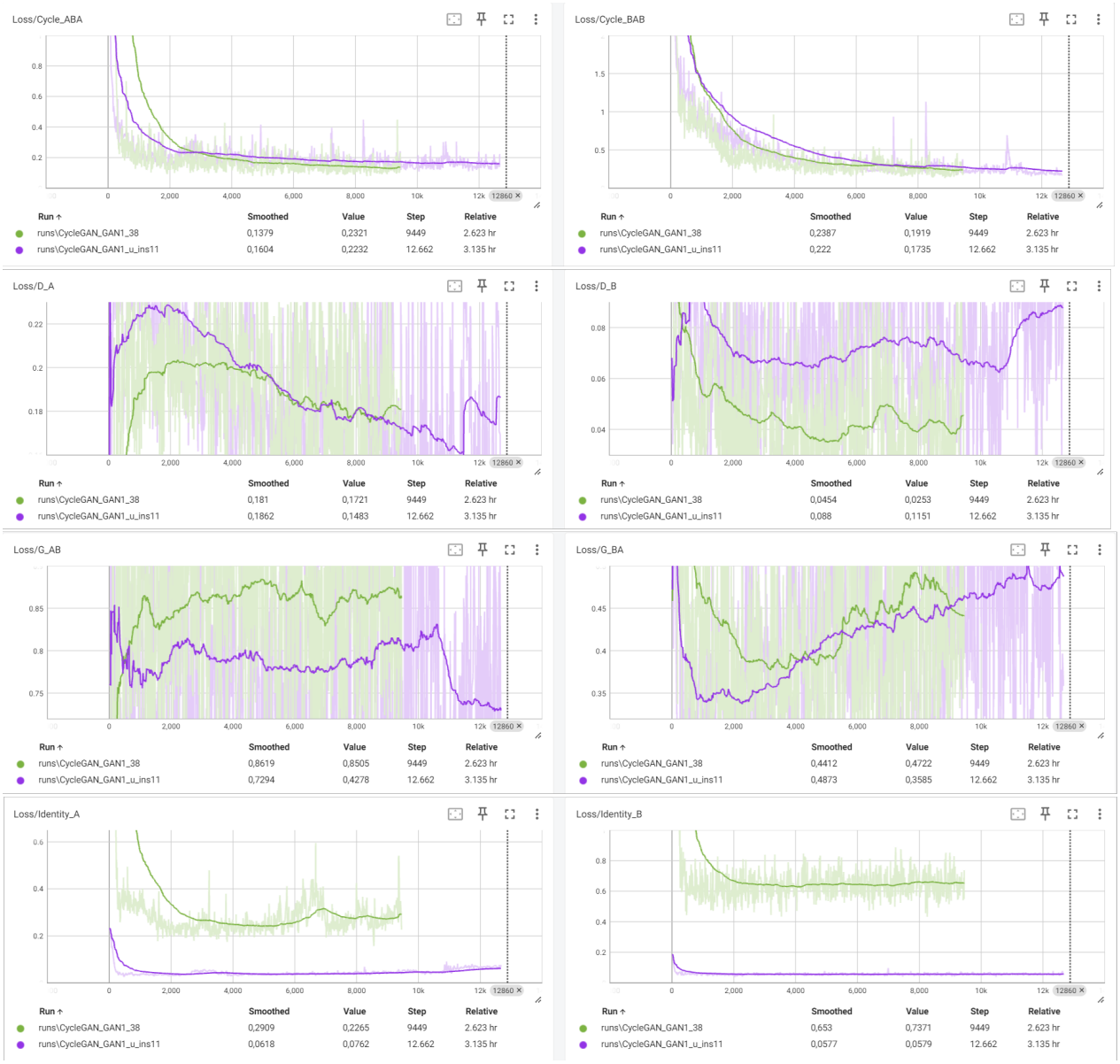


Figure 13: Loss Comparison in CycleGAN Training: BatchNorm vs. InstanceNorm on Original MNIST Test Set and Real Colored MNIST (First 1000 Images)

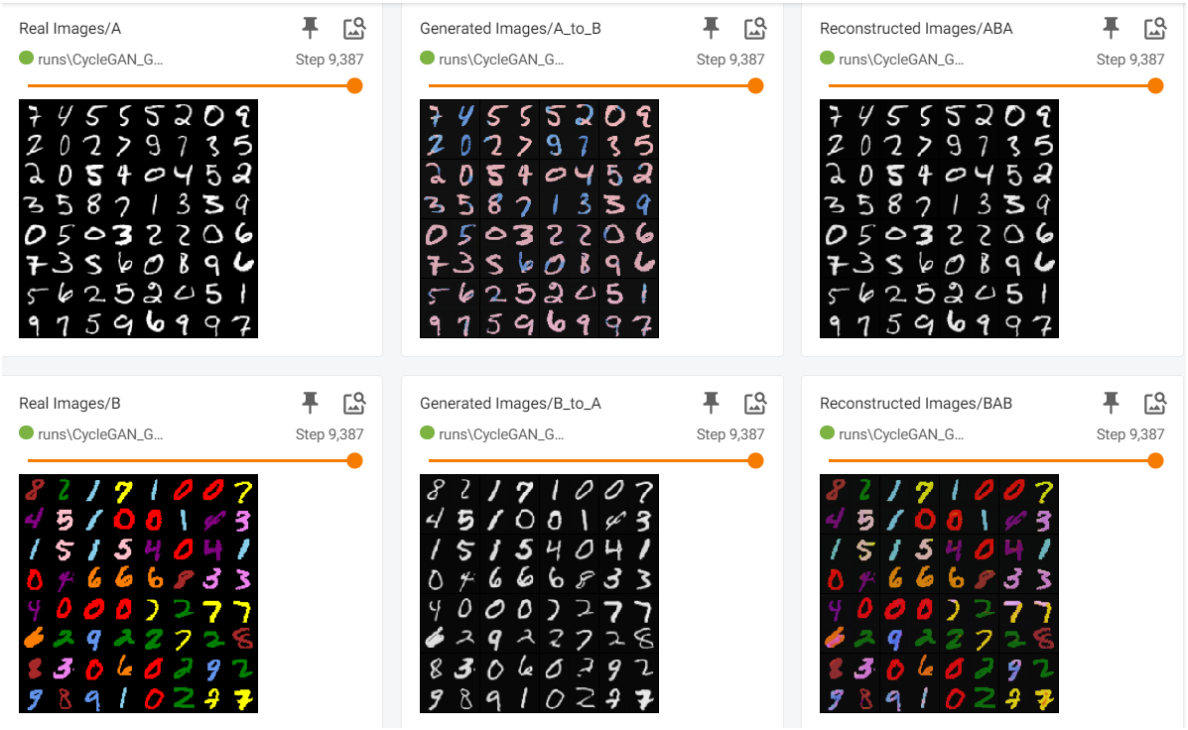


Figure 14: Images generated by CycleGAN training with BatchNorm layers.

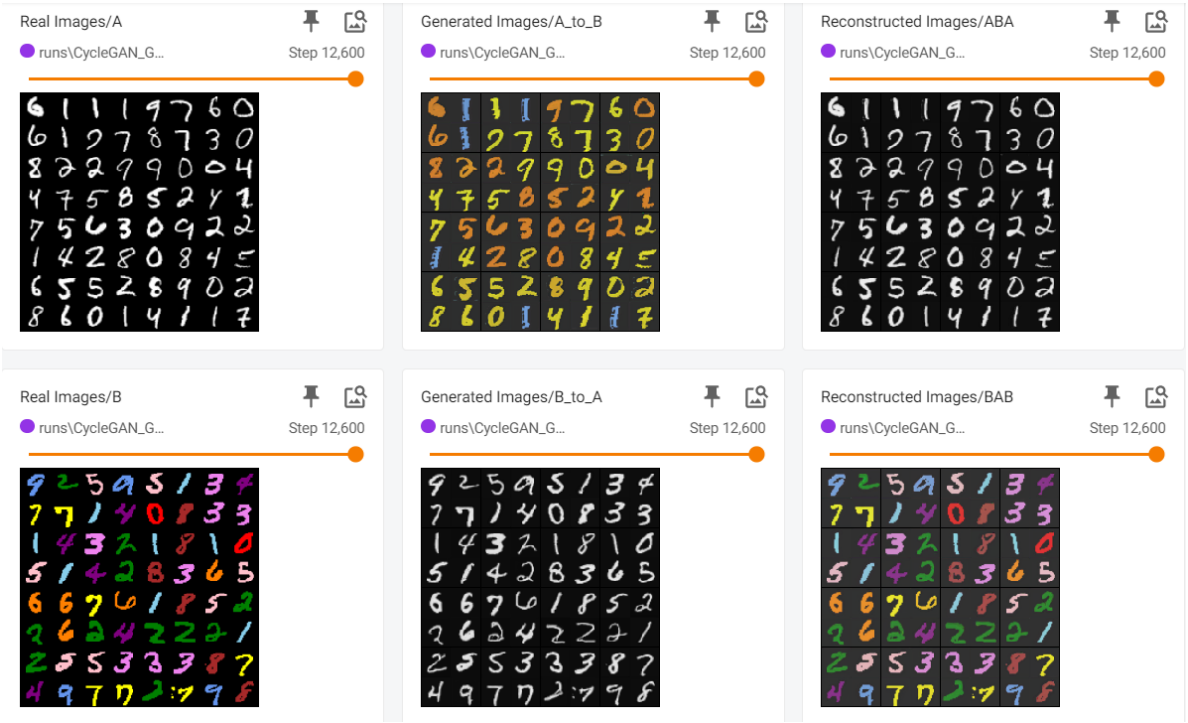


Figure 15: Images generated by CycleGAN training with InstanceNorm layers.

As you can observe, the images generated by CycleGAN with BatchNorm layers (see Figure 14) can distinguish only two colors in the MNIST numbers, in contrast to those generated by CycleGAN with InstanceNorm layers (see Figure 15), which can distinguish three colors.

- Domain A: Original MNIST (first 500 images of the test set) and domain B: first 500 images of training set of SVHN

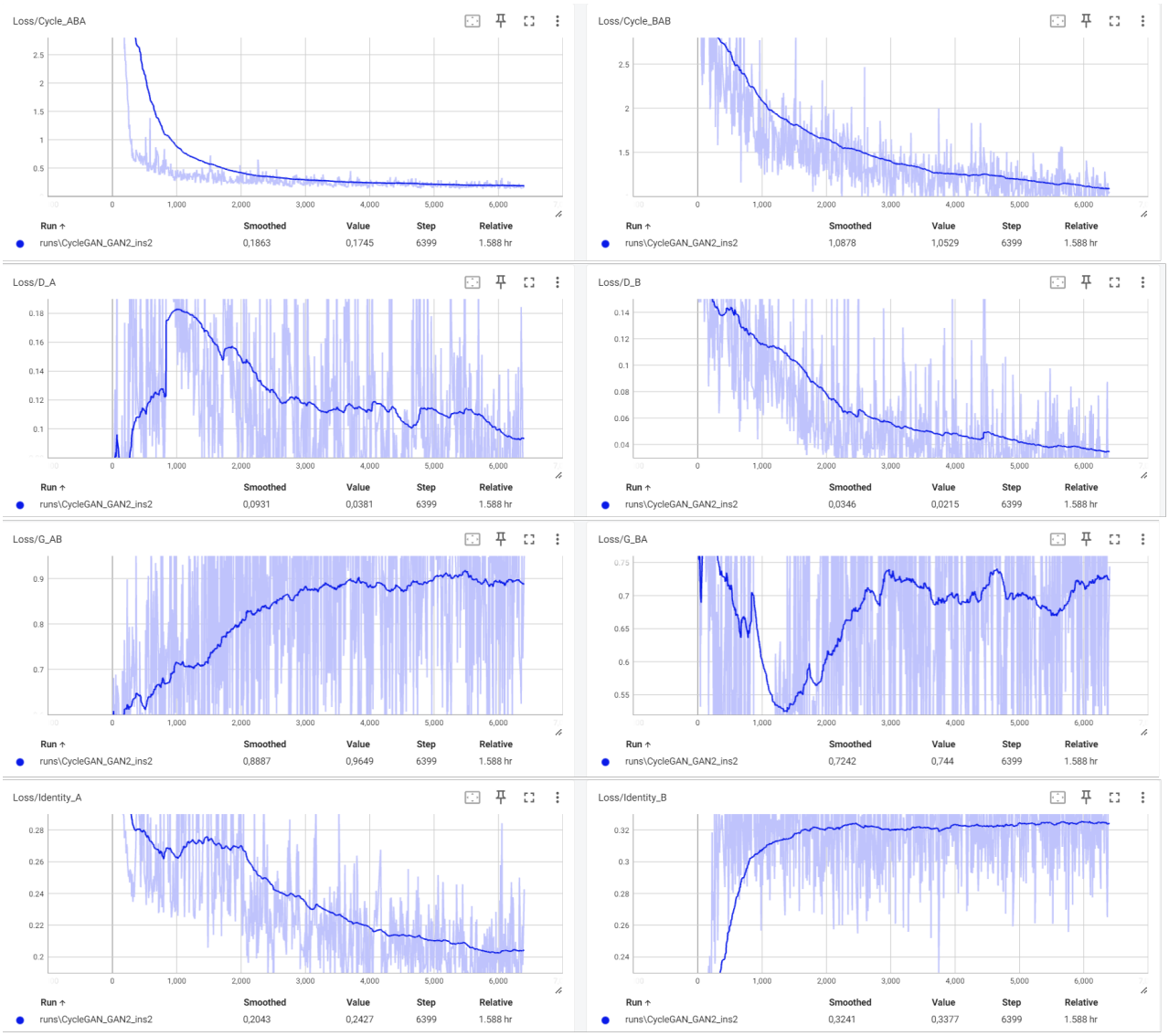


Figure 16: CycleGAN Training: Original MNIST Test Set and SVHN Test Set (First 500 Images).

This time, when I trained my CycleGAN model between the original MNIST (first 500 images from the test set) and the first 500 images from the SVHN training set for style transfer, the results, in my opinion, were not good enough, as you can see in the images. This is especially true for the G_BA generator, where the numbers are almost unrecognizable. Using a larger batch size could improve the training because it provides a more stable and accurate gradient estimation in each update, which can lead to more effective learning. Also, a more robust UNET architecture might better capture the complexities of the data, resulting in more accurate style transfer and visually improved results.

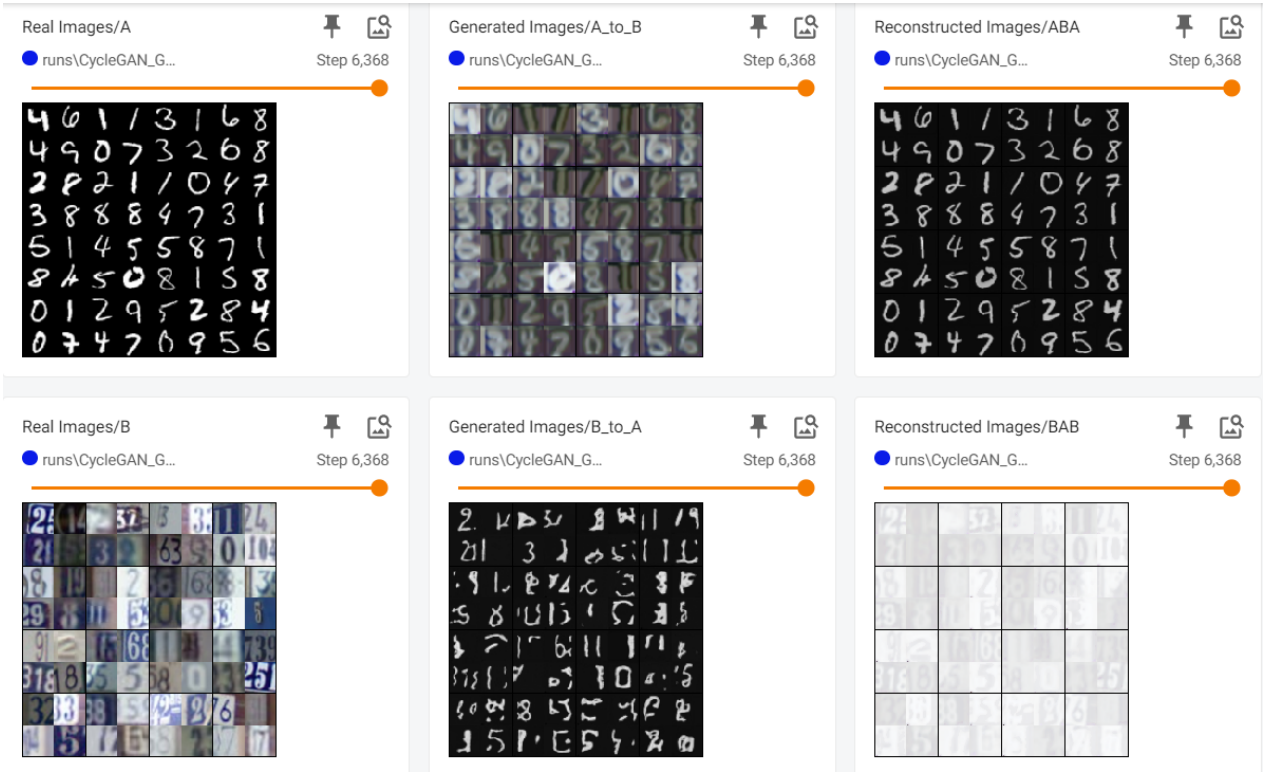


Figure 17: CycleGAN generated images for MNIST and SVHN.

4.1.3 Please provide the generators and discriminators loss curves. Please show the 64 inference results of your style transfer for the two transferring scenarios. Tip: for the plot you can use the following code (or the one previously provided in the Assignment I):

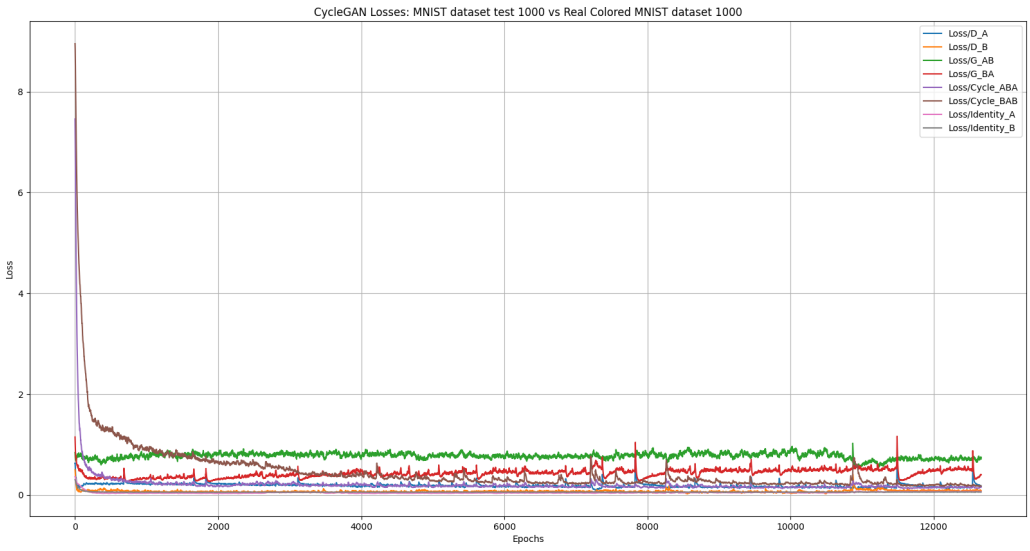


Figure 18: Generator, discriminator, cycle and identity loss curves of the first CycleGan model (MNIST and Real Colored MNIST).

(a) Inferred image of the generator G_AB (b) Inferred image of the generator G_BA

Figure 19: Inferred images by CycleGAN I (MNIST and Real Colored MNIST)

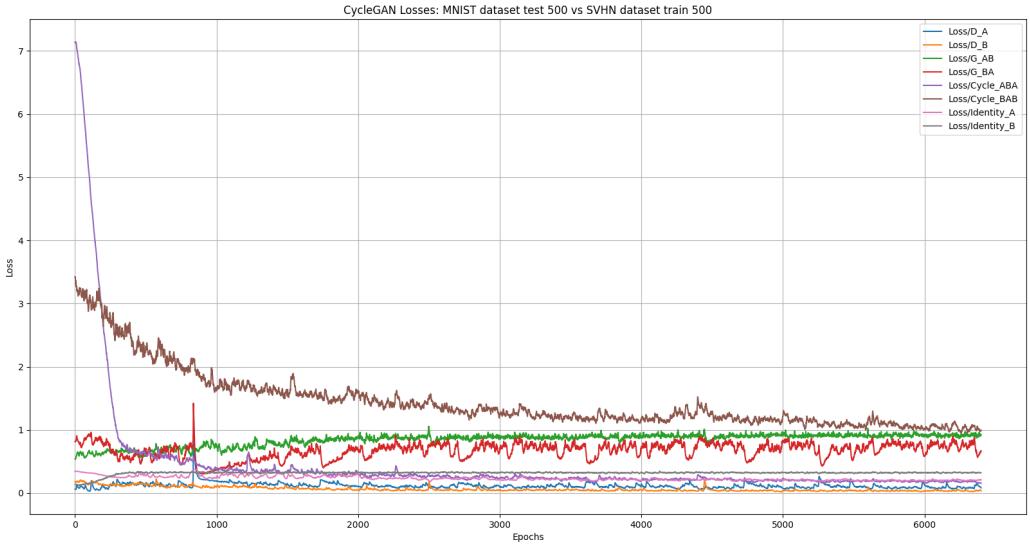
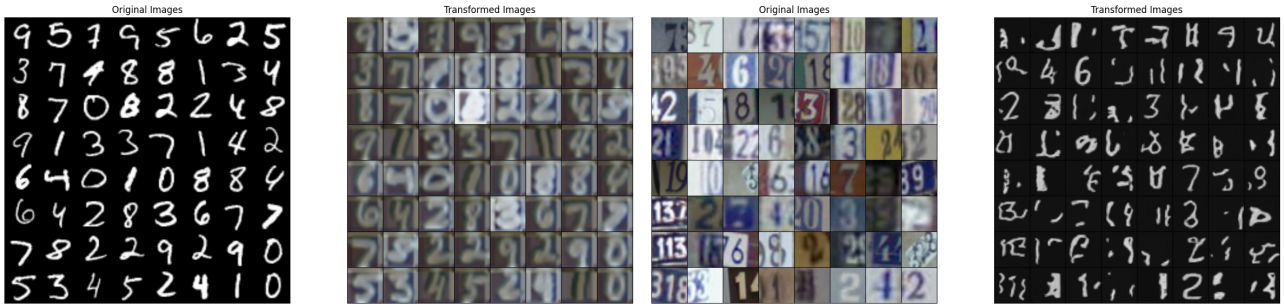


Figure 20: Generator, discriminator, cycle and identity loss curves of the second CycleGan model (MNIST and SVHN).



(a) Inferred image of the generator G_AB (b) Inferred image of the generator G_BA

Figure 21: Inferred images by CycleGAN II (MNIST and SVHN)

4.1.4 Please compute the FID distance and comment the results between the generated and real samples:

- Domain A: real colored MNIST and domain B: generated colored from your Cycle-GAN

Figure 22: Comparison of Domain A and Domain B, first scenario

Observing the first scenario, I noted an FID of 105.999, indicating a moderate similarity between the real colored MNIST images and those my Cycle-GAN generated. This score suggests to me that although my model has managed to capture fundamental aspects of the real dataset, there is still room for improvement in perfectly mimicking its distribution. I realize my Cycle-GAN has successfully learned to colorize and replicate MNIST digits to some extent; however, the difference in the FID score highlights the areas where the generated images diverge from the real ones in terms of detail and color accuracy.

- Domain A: first 500 images of training set of original MNIST and domain B: first 500 images of training set of SVHN

Figure 23: Comparison of Domain A and Domain B, second scenario.

In the second scenario, the significantly high FID of 224.015 highlights a substantial disparity between the original MNIST and SVHN datasets. This vast difference was expected due to the inherent differences between both domains: MNIST is grayscale with simple backgrounds, while SVHN is in color with more complex scenes. This high FID underscores the challenge of comparing these fundamentally different datasets, evidencing the broad diversity in visual appearance and composition between MNIST and SVHN. This observation reminds me of the importance of considering the unique characteristics of each dataset when evaluating transformations generated by models like Cycle-GAN.

- Domain A: first 500 images of training set of SVHN and domain B: generated SVHN numbers with your CycleGAN (500 first samples of MNIST training set)

Figure 24: Comparison of Domain A and Domain B, third scenario.

Finally, in the third scenario, with an FID of 202.081, I see a notable but slightly lower disparity between real SVHN images and SVHN-like numbers generated from MNIST through my Cycle-GAN. This result indicates that my model has somewhat managed to adapt the MNIST dataset to resemble the complexity and color scheme of SVHN, although there are still significant differences in texture, background complexity, and digit styles. This FID signals a successful transformation in some aspects but also points out limitations in fully capturing SVHN's diverse and rich visual content.

5 Part IV [Bonus Points] - Data Augmentation for Classification

In addressing the challenge of improving classifier accuracy with limited annotated samples, I plan to integrate style transfer as a key strategy for data augmentation. My intent is to creatively blend the visual styles of MNIST and SVHN datasets to produce a richly varied set of training images. This approach hinges on the belief that by introducing a wider spectrum of visual features through stylized images, I can significantly enhance the model's ability to generalize across different visual domains.

To ensure seamless integration into the training process, all images, both original and stylized, will be meticulously normalized and resized to meet the ResNet-18 model's input specifications. Given the model's architecture, which is initially tailored for a broad range of image recognition tasks, I foresee modifying its structure to better suit digit classification. This modification will likely involve altering the model's final layer to output ten classes, aligning with the digits 0 to 9.

For the training regimen, I envision leveraging the combined dataset of original and augmented images. This blend not only promises to enrich the model's exposure to varied patterns but also challenges it to discern and learn the essential features for accurate digit classification. The use of categorical cross-entropy for loss calculation, coupled with the Adam optimizer, is anticipated to further refine the learning process.

Balancing the dataset to prevent class bias and meticulously tuning the learning rate and epoch count based on performance feedback will be critical. Through this planned methodology, I'm optimistic about significantly boosting the classifiers' performance, making them more adept at recognizing digits from both the MNIST and SVHN datasets with greater accuracy. This forward-looking strategy, grounded in the augmentation of training data via style transfer and strategic model adaptation, is poised to address the limitations posed by the scarcity of annotated samples.