Question 3: Conditional GAN for Sketch to Real Face Generation *

*

Areeba Fatah
AI (artificial intelligence)
FAST, NUCES)
Islamabad, Pakistan
i210349@nu.edu.pk

I. Introduction

In this question, a Conditional Generative Adversarial Network (cGAN) was implemented to generate realistic face images conditioned on input sketches. The goal was to map face sketches to real faces. This task uses the original cGAN paper, where the generator learns to create realistic faces based on an input condition i.e. sketches and the discriminator evaluates the realism of the generated images in comparison to real ones.

II. Methodology

A. Dataset

The **Person Face Sketches** dataset was used, cons isting of paired images where each sketch has a corresponding real face image. This dataset was already split into train and test sets, but the focus here was on the train set.I have also used the test dataset to test the quality of images.

B. Preprocessing

A dataset class was created to load and preprocess the data in an organized way.

1) **Loading images**: Sketches are loaded as grayscale images and real photos as RGB images.
2) **Resizing images**: Images are resized to 256x256 pixels.
3) **Normalizing**:
   - Sketches are normalized to the range i.e. tanh range i.e. mean: 0.5 and std: 0.5.
   - Photos are normalized with mean and std both equal to 0.5.
4) **Batch Loader**: A pytorch's Data Loader is used for shuffling and slicing the dataset into batches of size 16. Since the dataset is very large so at a time only 1000 images were used.

C. Model Architecture

The architecture of the cGAN comprises two components: the generator and the discriminator.

D. Generator

The generator is designed to transform 1-channel sketch images into 3-channel realistic face images. It consists a series of convolutional layers followed by transposed convolutional layers with activation functions and batch normalization to ensure stable training of cGAN. The generator's goal is to create realistic images from the sketches.

E. Discriminator

The discriminator is structured to take two inputs: the sketch and a real or generated image of face. It concatenates these inputs to create a 4-channel input. The discriminator is trained to produce a single output value that shows the similarity of the paired inputs to distinguish between real and generated images.

F. Training Procedure

The model was trained using binary cross-entropy loss for both the generator and discriminator. An Adam optimizer was used for updating the weights of both of the networks. Rest of the training process was the same as GAN. The training was conducted over a series of epochs (100 at a time), during which the generator improved its ability to generate realistic images.

III. Results

A. Outputs

The following are the outputs of generators during training. Inputs: Refer to Image1 Results: Refer to Image2

Fig. 1. Sketches

B. Training loss

Epoch [41/50] — D loss: 0.5234721302986145 — G loss: 1.2039703130722046 Refer to Image 3

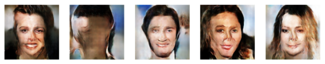

Fig. 2. Results

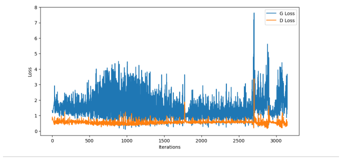

Fig. 3. Training loss

C. Testing Loss

Refer to Image 4

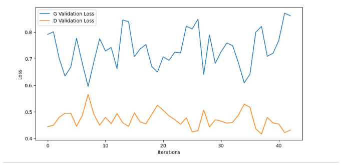

Fig. 4. Testing Loss

D. Testing Output

Refer to Image 5

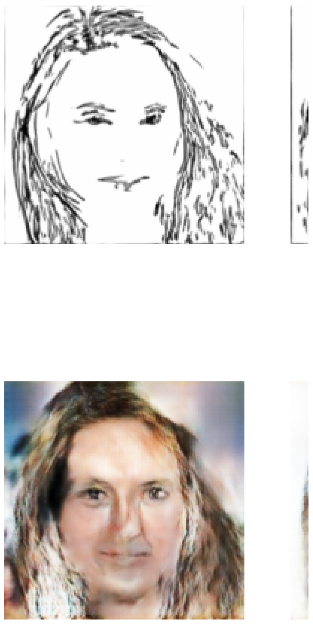

Fig. 5. output sketches

IV. Discussion

cGAN generated realistic results.Challenges faced during the implementation included:

- **Lack of Convergence**: The number of epochs, learning rate everything played a crucial role to optimize the performance otherwise due to lack of convergence the results would have been bad.
- **Detail Representation**: At first, the generated images lacked finer details such as facial features but with training these details improved significantly creating somewhat realistic images. But results could have been more better with better convergence as you can see the hair looks messy and the image lacks clarity.

Conclusion

The Conditional GAN was able to convert sketches into realistic face images. but further improvement is required.The model's capability highly relies on the hyperparameters and the way of training. Determining the correct number of epochs is also very important that's why the results were better but not perfect. Further analysis and resources are required for improving the results.

Prompts

- Code for conditional GAN
- How to design cGAN training loop
- Give me code to design training loop and plot results including loss functions.
- Give me report for this task as mentioned in the format.
- Why is the training curve not smooth make it smooth.
- Do the same for testing data.
- Preprocess the data at the link for training with GAN.
- Create data loaders for GAN training and testing.

References

[1] How to concatenate images in PyTorch for conditional GAN input? *PyTorch Documentation*. Accessed: 2024-10-21. URL: https://pytorch.org/docs/stable/torch.htmltorch.cat

[2] Best practices for monitoring GAN training and preventing mode collapse? *Machine Learning Mastery*. Accessed: 2024-10-21. URL: https://machinelearningmastery.com/how-to-prevent-mode-collapse-in-gans/

[3] How to generate high-quality plots for GAN loss? *Matplotlib Documentation*. Accessed: 2024-10-21. URL: https://matplotlib.org/stable/gallery/lines$_{b}ars_{a}nd_{m}arkers/plot_{l}egend.html$