Question 4 : CycleGAN For Face Image to Sketch And Vice Versa Translation *

*

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 based on input sketches and vice versa. The goal was to map faces to sketches and sketches to faces. This task is implemented using the concepts and architecture introduced in the original cGAN paper, where two generators are used one for generating faces from sketches and the other for generating sketches from faces. Each generator is accompanied by its discriminator whose objective is to guide the generators towards improvement.

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 CycleGAN consists of two generator networks and two discriminator networks:

D. Generators

1) GeneratorSketchToPhoto: Converts sketches to faces. It comprises 5 convolutional layers for downsampling and 4 transposed convolutional layers for upsampling with ReLU activations and a Tanh activation at the output.

2) GeneratorPhotoToSketch: Converts photo images to sketches, following a similar architecture to GeneratorSketchToPhoto but optimized for grayscale output.

E. Discriminators

1) DiscriminatorSketch: Distinguishes between real sketches and generated sketches using 4 convolutional layers with LeakyReLU activations.

2) DiscriminatorPhoto: Similar to DiscriminatorSketch but focuses on differentiating real photos from generated ones. The training process include adversarial training, cycle consistency loss, and identity loss to ensure the model learns the transformations effectively.

F. Flask Deployment

1) app.py : this file contains

- Both generator models created with pytorch.
- Loading trained models in pth and making it compatible to CPU.
- Functionalities of both the upload options and live camera inputs.

the generators and involves loading of pth files of both and incorporates real time camera input and uploading files and all the things in a

2) Index.html:

- Front end code for user friendly interface.

III. Results

A. Outputs

Refer to image 1

B. Training Loss

Refer to image 2 *Epoch [91/100], Step [62/63], Loss G: 6.0000, Loss D X: 0.0106, Loss D Y: 0.0042*

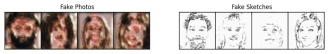

Fig. 1. outputs

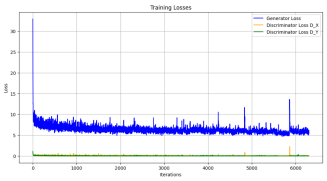

Fig. 2. Training loss

1) Front End: Refer to 3 and 4

IV. Discussion

*1) **Analysis of Results**:* The results doesn't show perfect improvements as it depends on a number of factors like learning rates and no of epochs.

*2) **Challenges Faced**:*

- **Memory Issues**: Training on the full dataset caused memory overflows that is why i used a chunk of 500 images at a time with a batch size of 16.
- **Hyperparameter Tuning**: Finding the right balance of learning rates was very hard and the results are not perfect due to this.

Conclusion

The task of making the cycleGAN from the original paper able to turn sketches to faces and faces to sketches was completed but the results could be further optimized by finding the right set of hyperparameters to find the convergence.

Prompts

- Create cycleGANs following the original paper.
- Preprocess and load the data into chunks.
- Give me training loop for cycleGANs.
- Give me code to visualize losses and results during training.
- Why is the model not converging? optimize it.
- Write report for this code.

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] Zhu, J.-Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. *IEEE International Conference on Computer Vision (ICCV)*, 2017, 2242-2251. URL: https://arxiv.org/abs/1703.10593

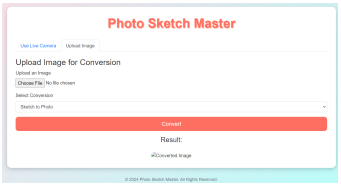

Fig. 3. Front end with upload option

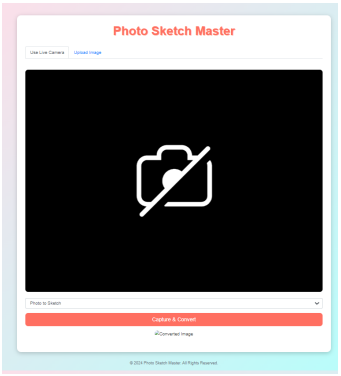

Fig. 4. Live Camera option