CycleGAN Implementation for Person Face Sketches: Image-to-Image Translation

SYED MUHAMMAD MURTAZA KAZMI
Department of Computer Science
National University of Computer and Emerging Sciences
Email: i210685@nu.edu.pk

***Abstract*—This paper presents the implementation of a CycleGAN model for translating person face sketches to real face images and vice versa. Utilizing a custom dataset of person face sketches and corresponding photos, the model achieves end-to-end image-to-image translation. The implementation addresses challenges such as memory constraints and model training stability, and includes a user-friendly interface for real-time image conversion. Results demonstrate the model's ability to generate accurate facial structures, although challenges remain in accurately capturing facial colors. The developed interface facilitates easy interaction with the model, enabling both sketch-to-photo and photo-to-sketch translations. The source code is available on GitHub for reproducibility and further development.**

***Index Terms*—CycleGAN, Image-to-Image Translation, Face Sketches, Deep Learning, Generative Adversarial Networks, PyTorch, Flask, Computer Vision, GitHub**

I. Introduction

Image-to-image translation has garnered significant attention in the field of computer vision, enabling the transformation of images from one domain to another with remarkable accuracy and realism. Traditional methods often require paired datasets, which are not always feasible to obtain. CycleGAN, introduced by [1], overcomes this limitation by enabling unpaired image-to-image translation through the use of cycle-consistency loss. This paper focuses on implementing CycleGAN for translating person face sketches to real face images and vice versa. The primary objective is to develop a robust model capable of generating high-fidelity images that preserve facial structures and details. Additionally, a user interface is developed to facilitate real-time image conversions, enhancing the practical applicability of the model in various applications such as digital art, forensics, and entertainment. The source code for this implementation is publicly available on GitHub [5], ensuring transparency and enabling reproducibility.

II. Methodology

This section details the dataset used, preprocessing steps, model architecture, training strategy, and implementation specifics.

A. Dataset

The dataset employed for this project consists of person face sketches and corresponding real face photos. The dataset is divided into training and validation sets, each containing separate directories for photos and sketches. The dataset structure is as follows:

TABLE I: Dataset Structure

Directory	**Contents**
`root/train/photos`	Real face images for training
`root/train/sketches`	Face sketches for training
`root/val/photos`	Real face images for validation
`root/val/sketches`	Face sketches for validation

Each image in the dataset is resized to 256x256 pixels to maintain consistency and facilitate efficient training. The dataset size is sufficient to train the CycleGAN model effectively, with diverse representations of facial features, lighting conditions, and artistic styles in the sketches.

B. Preprocessing

Effective preprocessing is crucial for the performance and stability of the CycleGAN model. The following preprocessing steps are applied to each image:

- **Resizing**: Images are resized using bicubic interpolation to ensure uniformity across the dataset.
- **Random Cropping**: A random crop of 256x256 pixels is extracted from the resized image to introduce spatial variability.
- **Horizontal Flipping**: Images are horizontally flipped with a 50% probability to augment the dataset and introduce invariance to horizontal orientations.
- **Normalization**: Pixel values are scaled to the range [-1, 1] using mean and standard deviation normalization. This scaling is essential for stabilizing GAN training and ensuring that the network operates within a suitable numerical range.

The preprocessing pipeline is implemented using PyTorch's `transforms` module, enabling efficient and reproducible transformations.

C. Model Architecture

The CycleGAN architecture comprises two generator networks (*Generator A2B* and *Generator B2A*) and two discriminator networks (*Discriminator A* and *Discriminator B*).

1) Generators: Each generator follows a ResNet-based architecture with the following components:

- **Initial Convolution Layer**: A 7x7 convolutional layer with reflection padding, followed by instance normalization and ReLU activation.
- **Downsampling Layers**: Two convolutional layers with stride 2 for downsampling, each followed by instance normalization and ReLU activation.
- **Residual Blocks**: Nine residual blocks, each consisting of two 3x3 convolutional layers with reflection padding, instance normalization, and a skip connection.
- **Upsampling Layers**: Two transposed convolutional layers with stride 2 for upsampling, each followed by instance normalization and ReLU activation.
- **Output Layer**: A final 7x7 convolutional layer with reflection padding and Tanh activation to generate the output image.

2) Discriminators: Each discriminator follows a PatchGAN architecture, which classifies each N×N patch in an image as real or fake. The components include:

- **Convolutional Layers**: A series of convolutional layers with increasing feature maps and stride 2, each followed by instance normalization and LeakyReLU activation.
- **Output Layer**: A final convolutional layer that outputs a feature map indicating the probability of each patch being real.

D. Training Strategy

The model is trained end-to-end for 50 epochs using the following strategies:

1) Loss Functions: Three primary loss functions are employed:

- **Adversarial Loss**: Uses Mean Squared Error (MSELoss) to encourage generators to produce images that the discriminators classify as real.
- **Cycle-Consistency Loss**: Utilizes L1Loss to ensure that translating an image to the other domain and back results in the original image.
- **Identity Loss**: Also uses L1Loss to preserve color composition when translating images that are already in the target domain.

2) Optimizers and Learning Rate Schedulers: Adam optimizers are used for both generators and discriminators with a learning rate of 0.0002 and beta parameters (0.5, 0.999). Learning rate schedulers gradually decrease the learning rate after a set number of epochs to facilitate convergence.

3) Replay Buffers: Replay buffers store previously generated fake images to stabilize discriminator training by preventing rapid oscillations. This technique ensures that the discriminators receive a diverse set of fake samples over time.

E. Implementation Details

The implementation is carried out using PyTorch, leveraging GPU acceleration for efficient computation. Key implementation details include:

- **Batch Size**: Set to 4 to balance between memory constraints and training stability.
- **Data Loading**: Utilizes PyTorch's `DataLoader` with multiple workers and pin memory to optimize data transfer to the GPU.
- **Checkpointing**: Model weights are saved after every epoch to allow resumption from the last checkpoint in case of interruptions.
- **Memory Management**: Addresses memory issues by splitting the dataset into smaller batches and reducing the number of training samples when necessary.

The training loop includes alternating updates between generators and discriminators, ensuring balanced training dynamics. Additionally, to facilitate reproducibility and further development, the complete source code is made available on GitHub [5].

III. Results

This section presents the qualitative results of the CycleGAN model, including sample image translations and the functionality of the user interface.

A. Image Translation Examples

The CycleGAN model successfully translates sketches to photos and photos to sketches. Below are sample translations demonstrating the model's capability to capture and reproduce facial structures.

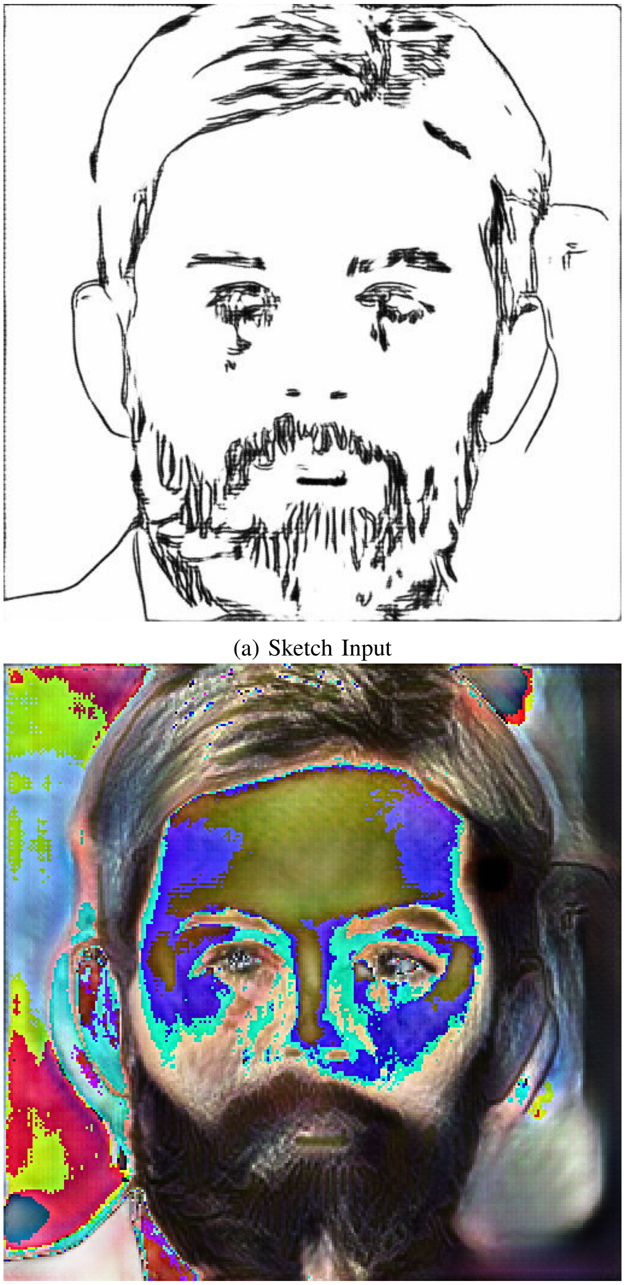
1) Sketch to Photo Translation: As illustrated in Figure 1, the model effectively translates sketches into realistic face images, accurately capturing the facial structure and key features such as eyes, nose, and mouth. However, the generated images often lack accurate facial colors, resulting in grayscale or desaturated outputs. This limitation indicates that while the model preserves structural integrity, color fidelity requires further enhancement.

2) Photo to Sketch Translation: Figure 2 showcases the model's ability to convert real face images into sketches. The translated sketches maintain the essential facial features and overall shape, demonstrating the model's proficiency in capturing the nuances of facial structures. The sketches exhibit clear outlines and shading, characteristic of hand-drawn sketches, validating the effectiveness of the CycleGAN architecture in this translation direction.

B. User Interface

A Flask-based web interface was developed to facilitate real-time image conversions. The interface allows users to:

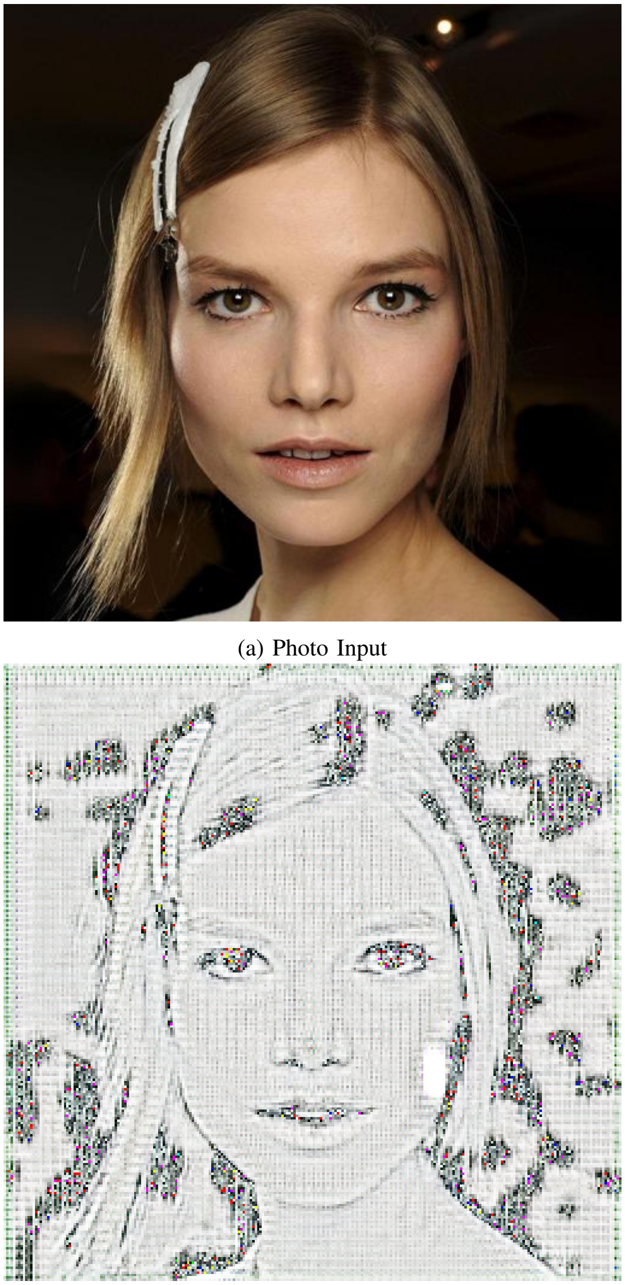
- **Upload Images**: Users can upload a photo or sketch from their device.
- **Live Camera Input**: Users can capture images using their device's camera.
- **Select Translation Direction**: Users can choose between sketch-to-photo or photo-to-sketch translation.
- **View Results**: The translated image is displayed alongside the original input.



(a) Sketch Input

(b) Photo Output

Fig. 1: Sketch to Photo Translation



(a) Photo Input

(b) Sketch Output

Fig. 2: Photo to Sketch Translation

The interface is designed to be intuitive and responsive, providing immediate feedback to user inputs. Below is a screenshot of the user interface:

IV. Discussion

This section delves into the model's performance improvements over time, challenges encountered during implementation and training, and potential avenues for future enhancements.

A. Model Performance Improvement

Throughout the training process, the CycleGAN model exhibited consistent improvements in image translation quality. Early epochs produced rudimentary translations with discernible facial structures but significant artifacts and color inconsistencies. As training progressed, the generators became adept at producing more refined images with accurate facial features and reduced artifacts. The cycle-consistency and identity losses played a pivotal role in enforcing the preservation of essential facial characteristics, thereby enhancing the overall realism of the generated images.

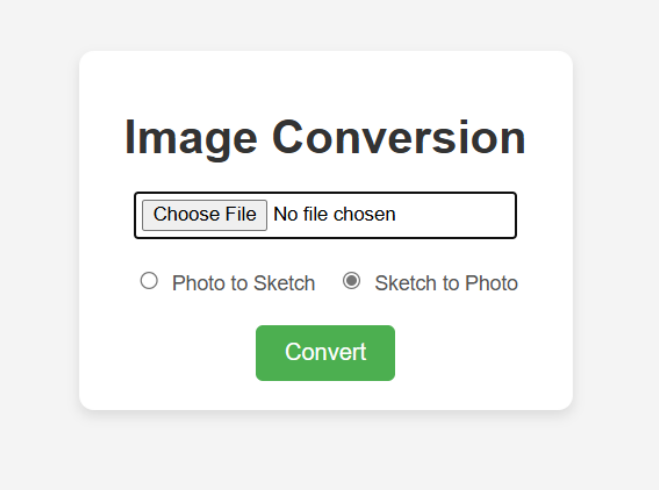


Image Conversion

No file chosen

○ Photo to Sketch ● Sketch to Photo

Fig. 3: Flask-Based User Interface for Real-Time Image Conversion

B. Challenges Encountered

Several challenges were encountered during the implementation and training phases:

1) Memory Constraints: Training GANs, particularly CycleGANs, is computationally intensive and memory-demanding. The initial attempts to load the entire dataset at once led to memory overflows, necessitating the splitting of the dataset into smaller batches. Additionally, reducing the batch size was considered to mitigate memory usage, albeit at the expense of training stability and speed.

2) Color Fidelity in Sketch to Photo Translation: One of the prominent challenges was achieving accurate color reproduction in the sketch-to-photo translation. The generated images often lacked vibrant colors, resulting in grayscale or desaturated outputs. This issue suggests that while the model effectively captures structural details, it struggles with color information. Possible reasons include:

- **Insufficient Color Data**: The dataset may not provide enough color variation or may be biased towards certain color palettes.
- **Loss Function Limitations**: The current loss functions prioritize structural consistency over color accuracy.
- **Model Capacity**: The generator may require additional capacity or specialized layers to handle colorization effectively.

3) Training Stability: Maintaining a balance between generator and discriminator training is critical for GANs. Instances of mode collapse, where the generator produces limited variations of images, were observed. Implementing replay buffers and carefully tuning hyperparameters helped mitigate these issues but did not entirely eliminate them.

C. Future Work

To address the identified challenges and enhance the model's performance, the following avenues are proposed for future work:

1) Enhanced Colorization Techniques: Integrating advanced colorization methods or specialized loss functions focused on color accuracy can improve the realism of sketch-to-photo translations. Techniques such as perceptual loss or color histogram matching may be beneficial.

2) Data Augmentation and Expansion: Expanding the dataset to include a more diverse range of colors, lighting conditions, and artistic styles can provide the model with richer information, aiding in better color reproduction and overall image quality.

3) Architectural Improvements: Exploring more sophisticated architectures, such as incorporating attention mechanisms or multi-scale generators, can enhance the model's ability to capture intricate facial details and color nuances.

4) Higher-Resolution Training: Training the model on higher-resolution images can improve the quality of generated images, allowing for finer details and more accurate color representations.

5) Advanced Training Strategies: Implementing techniques such as spectral normalization, gradient penalty, or adaptive learning rates can further stabilize training and prevent issues like mode collapse.

V. Conclusion

This paper presents a comprehensive implementation of the CycleGAN model for translating person face sketches to real face images and vice versa. The model successfully captures and preserves facial structures, demonstrating the efficacy of CycleGAN in unpaired image-to-image translation tasks. Despite challenges related to memory constraints and color fidelity, the project achieved significant milestones, including the development of a user-friendly interface for real-time image conversion. The availability of the source code on GitHub [5] ensures transparency and facilitates further development and research. Future enhancements are anticipated to address current limitations, thereby improving the model's performance and broadening its applicability in various domains such as digital art, forensics, and entertainment.

VI. Prompts

In the context of this project, ”Prompts” refer to the user inputs provided through the interface for image translation. Users interact with the system by:

- **Selecting Translation Direction**: Choosing whether to convert a sketch to a photo or a photo to a sketch.
- **Uploading Images**: Providing an image file from their device for translation.
- **Using Live Camera Input**: Capturing a real-time image using their device's camera for immediate translation.

These prompts facilitate seamless interaction with the CycleGAN model, allowing users to visualize and utilize the image translation capabilities effectively. The source code repository on GitHub [5] includes detailed instructions and examples to help users deploy and interact with the interface.

VII. References

References

[1] J. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks,” *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2223–2232.

[2] A. Paszke *et al.*, “PyTorch: An Imperative Style, High-Performance Deep Learning Library,” *Advances in Neural Information Processing Systems*, 2019.

[3] Pallets Projects, “Flask Documentation,” [Online]. Available: https://flask.palletsprojects.com/

[4] PyTorch, “Security in PyTorch,” [Online]. Available: https://github.com/pytorch/pytorch/blob/main/SECURITY.md#untrusted-models

[5] S. M. K.47, “CycleGAN Implementation for Person Face Sketches,” GitHub. [Online]. Available: https://github.com/smmk47/CycleGAN-Face-Sketch

[6] J. Zhu, “CycleGAN: Unpaired Image-to-Image Translation with Generative Adversarial Networks,” GitHub. [Online]. Available: https://github.com/junyanz/CycleGAN

[7] P. Oliva and A. Torralba, “Unbiased Look at Dataset Bias,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 1526–1534.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative Adversarial Networks,” *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.