IMAGE TO PENCIL SKETCH USING CYCLE GAN

TEAM MEMBERS

P.Deekshith.	**cb.sc.u4aie24039**
S.Dheeraj	**cb.sc.u4aie24050**
S.V.Dhiraj	**cb.sc.u4aie24052**
S.Sathwik	**cb.sc.u4aie24051**

INTRODUCTION

- CycleGAN (Cycle-Consistent Generative Adversarial Network) is a type of GAN designed for image-to-image translation without requiring paired training data.
- It consists of two generator networks and two discriminator networks.
- The network learns to map between two domains (e.g., photos to sketches and vice versa).

OBJECTIVES

- Build a CycleGAN model to transform real images into pencil sketches.
- Use unpaired datasets for training, eliminating the need for matched image-sketch pairs.
- Focus on generating high-quality pencil sketches with realistic shading and details.
- Optimize model performance for fast sketch generation.
- Test and evaluate the model on various image types (e.g., faces, landscapes).
- Compare results with other image-to-sketch methods to showcase CycleGAN's effectiveness.

METHODOLOGY

Step 1: Collecting the Data

We need two sets of images:

- Real-world images (natural photos of people, objects, or scenery).
- Pencil sketch images (hand-drawn or software-generated sketches).
- These images should be of the same size (e.g., 256×256 pixels).
- More diverse images = Better training.

Step 2: Preprocessing the Data

Before we feed images into the CycleGAN model, we prepare them properly:

- Resizing images to a fixed size (e.g., 256×256 pixels).
- Normalizing pixel values (changing them from 0-255 to 0-1 or -1 to 1 for better training).
- Data Augmentation (applying small changes like rotation, flipping, cropping to make the model learn better).

Step 3: Building the Generator (Creates Pencil Sketches)

- The Generator is a deep learning model that converts a normal image into a pencil sketch.
- Takes a normal image as input.
- Passes it through Convolutional layers (to extract features like edges, textures).
- Uses ResNet Blocks to retain details while modifying the image.
- Upsamples (increases the size back) to generate a pencil sketch.
- Outputs a transformed image that looks like a pencil sketch.

Step 4: Building the Discriminator (Checks if Sketch is Real or Fake)

- The Discriminator is a deep learning model that checks whether a sketch is real (hand-drawn) or fake (generated by AI).
- Takes an image (either a real pencil sketch or a generated one).
- Passes it through multiple Convolutional Layers (extracts important details).
- Uses PatchGAN (checks if small parts of the image are real or fake).
- Outputs a probability score:

Close to 1 → The image is real.
Close to 0 → The image is fake.

Step 5: CycleGAN Training Process

Since we don’t have paired images, we use two Generators and two Discriminators.

Training Process in Steps

- Generator A converts a real image into a pencil sketch.
- Discriminator A checks whether the sketch is real or fake.
- Generator B tries to convert the sketch back into a real image (cycle consistency).
- Discriminator B checks if the reconstructed image looks like a real photo.
- Cycle loss is applied: If the generated image cannot be converted back properly
- Both Generators and Discriminators are updated to improve performance over time.

Step 6: Loss Functions (Making AI Learn Properly)

- Adversarial Loss (GAN Loss) → Helps Generators create realistic images.
- Cycle Consistency Loss → Ensures that an image converted into a sketch can be converted back.

Step 7: Training the CycleGAN Model

- Use Google Colab or a GPU-powered system to speed up training.
- Load the preprocessed dataset and divide it into training and testing sets.
- Train the CycleGAN model using Keras and TensorFlow.

Step 8: Testing and Evaluating the Model

- Input real images into the trained model and check the generated sketches.
- Compare generated sketches with real hand-drawn sketches.
- Use metrics like Structural Similarity Index Measure (SSIM)

METRICS

SSIM measures the similarity between two images based on luminance, contrast, and structure. Unlike traditional pixel-wise metrics (e.g., MSE), SSIM focuses on the way human eyes perceive image quality.

SSIM values range from -1 to 1:

- 1 → Perfectly similar images.
- 0 → No similarity.
- -1 → Completely different images.

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

- μx and μy: Mean (average) intensity values of images x and y.
- σx^2 and σy^2: Variances of images x and y, which measure the spread of intensity values.
- σxy: Covariance between x and y, which indicates how much the intensity values of the two images change together.
- C1 and C2: Small constants to stabilize the division and prevent instability when the denominator is close to zero.
- Our model achieved an SSIM of 0.56

ADVERSIAL LOSS FUNCTION

FOR DISCRIMINATOR LOSS:

$$L_D = -\mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] - \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

FOR GENERATOR LOSS:

$$L_G = -\mathbb{E}_{z \sim p_z} [\log D(G(z))]$$

Where:

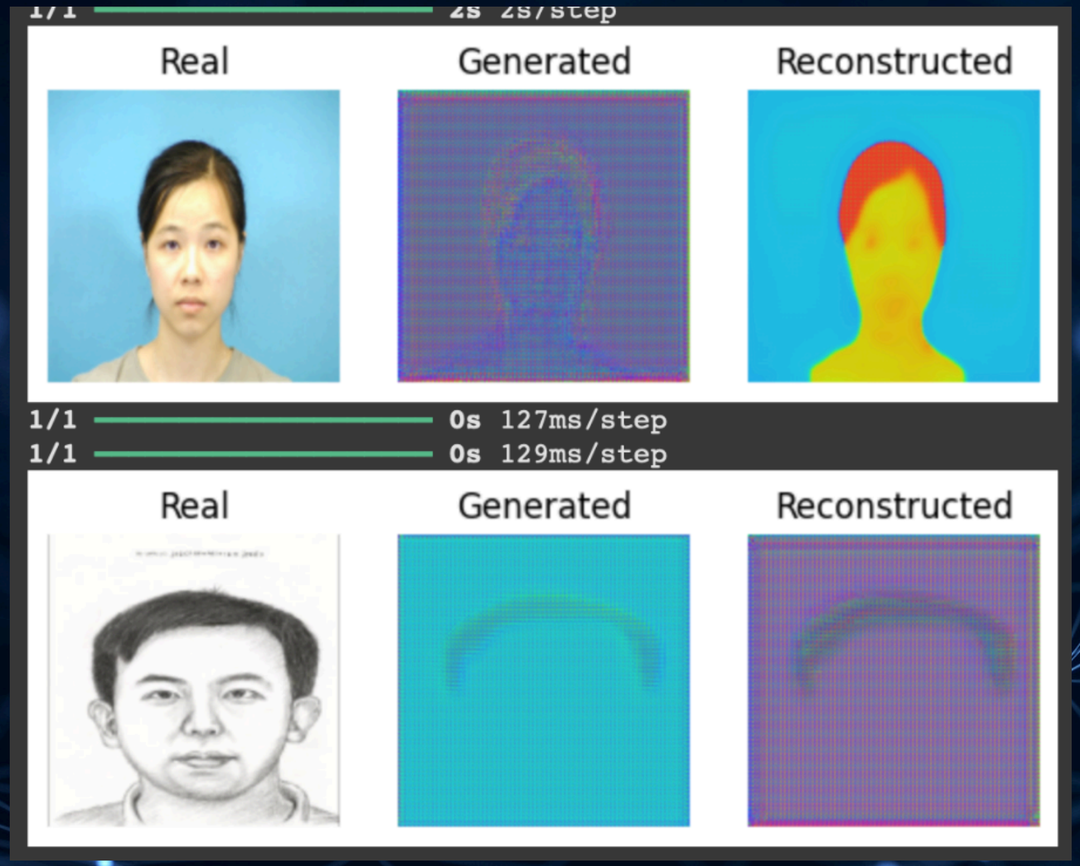
- **D(x) is the Discriminator's prediction for a real image x (the probability that x is real).**
- **G(z) is the output of the Generator, i.e., the fake image generated from random noise z.**
- **pdata(x) is the distribution of real images in the training set.**
- **p z(z) is the distribution of random noise input to the Generator.**

CYCLE CONSISTENCY LOSS FUNCTION

$$L_{\text{cycle}} = \mathbb{E}_{x \sim p_{\text{data}}(x)} \left[\| G_B(G_A(x)) - x \|_1 \right] + \mathbb{E}_{y \sim p_{\text{data}}(y)} \left[\| G_A(G_B(y)) - y \|_1 \right]$$

- x is an image from domain X (e.g., a photo).
- y is an image from domain Y (e.g., a sketch).
- GA(x) is the image generated by mapping x(from domain X) to domain Y.
- GB(y) is the image generated by mapping y (from domain Y) to domain X.
- GB(GA(x)) is the result of mapping x to domain Y and then back to domain X.
- GA(GB(y)) is the result of mapping y to domain X and then back to domain Y.
- | . |_1 is the L1 norm (mean absolute error), which computes the pixel-wise absolute difference between the images. This is used to measure how similar the original and reconstructed images are.

RESULTS



- High-Quality Sketches: The model successfully converts real images into realistic pencil sketches with fine details.
- Bidirectional Image Translation:CycleGAN effectively learns to translate images to sketches and back to the original image.
- Preserved Structural Details: The model maintains key features like edges, contours, and shading patterns in the generated sketches.

TIME LINE

As of now we did-

- Data Collection & Preprocessing Research & Dataset Gathering
- Training Implement CycleGAN architecture,Start training CycleGAN with a small dataset subset

Later we will work on-

Improve training stability by:

- Adjusting for sharper sketches
- Test model on real-world images outside the dataset


THANK
YOU