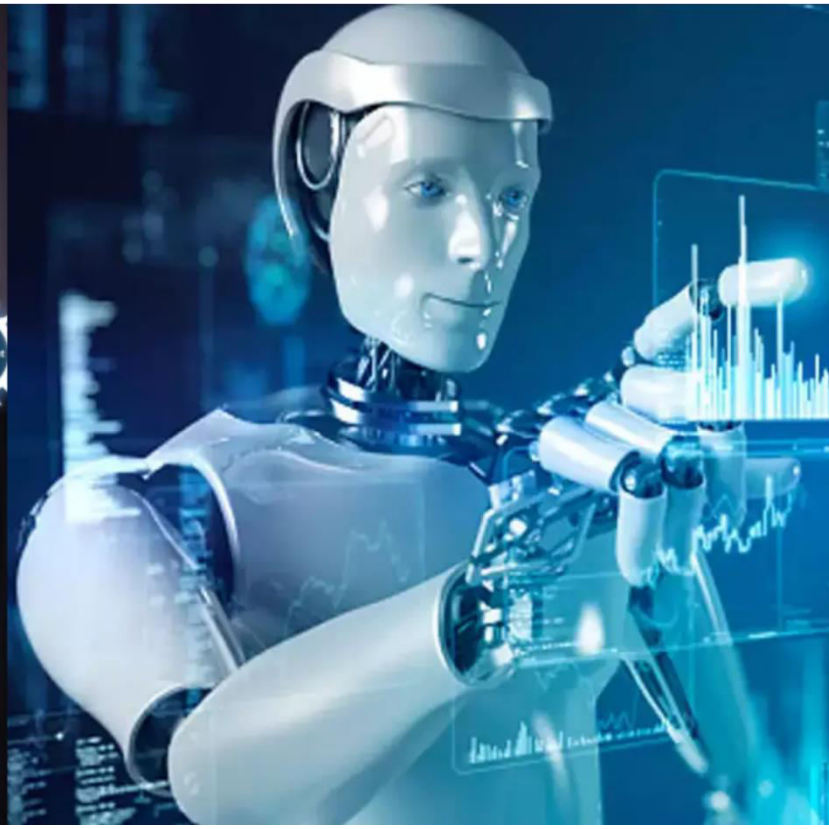
IJIRSET
International Journal of Innovative Research in Science | Engineering | Technology

ISSN(O): 2319-8753
ISSN(P): 2347-6710

International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.699

Volume 14, Issue 3, March 2025

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 3, March 2025

|DOI: 10.15680/IJIRSET.2025.1403164|

Image to Image Translation using Cyclegans

Mrs. Kudaravalli Deepika[1,a)], Nihith Yelchuri[2,b)], Ravipati Sahithya[3,c)], P Nagur Khan Sohail[4,d)], Syamala Vishnu Vardhan Reddy[5,e)], Challa Srishanth[6,f]

Assistant Professor, Department of CSE (Artificial Intelligence and Machine Learning), Vasireddy Venkatadri Institute of Technology, Nambur, Andhra Pradesh, India[1]

B. Tech Student, Department of CSE (Artificial Intelligence and Machine Learning), Vasireddy Venkatadri Institute of Technology, Nambur, Andhra Pradesh, India[2,3,4,5,6]

ABSTRACT: This is an advanced idea, designed for the image-to-image translation which is the most significant task in Computer Vision which enables the transformation of images from one domain to another domain with out losing any kind of characteristics. Whereas normal supervised approach requires a large data set and also requires more time for generating images and cross-checking images in traditional method where as by using Cycle GAN model various image translation tasks which includes medical imaging and other forms of translation. And this model is evaluated using key metrics of SSIM, PSNR and FID which results in high range and quality images. Overall, this research shows how efficient and flexible of Cycle GAN for various kinds of images.

I. INTRODUCTION

This System is to generate image-to-image translation using Cycle GANs, this is a fundamental problem in computer vision domain which involves many steps. It has many kinds of applications which are medical imaging, sketches etc.
First of all, user will upload the image in the input column as jpg or jpeg format into the system and then the process will begin. That uploaded image will be undergone into the generator G and the main goal is to translate this image into different kinds of domains while maintaining consistency. Generator has 3 different layers called G1,G2,G3 where as the process moves from one to another, Now after receiving the image from the user generator will try to transform the image as resultant image if that doesn't not matches he sends it to the discriminator D, then discriminator D will cross verify whether that the image us fake or real and then if it doesn't match with the final image discriminator D will classify the images into 2 groups real and fake and if it doesn't matches it is sent to the fake module, and this fake image is generated by the Generator G which supposed to be real from the target domain but in later evaluation by the discriminator D will be done and classified.

If the image is sent to the real module that means it is typically matched with the images in the dataset which are used for the training by the discriminator D. Basically Discriminator is of 2 types D1, D2 which represents D1 means decision D1 and D2 means decision D2, These disclinations receive both real and fake images from the generator G to make the decision whether the image should come under real category or fake category. After all the process of classification is done by the discriminators the output of these decisions helps to calculate loss function which is used for training. And also, there is an unchanged state which is when the image is transformed from domain A to domain B and then Back to A, it will remain in unchanged state. If needed by the discriminator the image can the reconstructed even after passing it to the generator.

II. LITERATURE SURVEY

There are several strategies that come out in existing studies to enhance image to image translation using cycle GANs and we inherited some of the properties for these papers which are pre published some of them are
Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks (CycleGAN) – 2020
Here, They Introduced cycle consistency loss and making able to unpaired image-to-image translation. Here even there are features like style transfer, season change and domain adaption.
Image-to-Image Translation with Conditional Adversarial Networks (Pix2Pix) – 2018
Proposed conditional GANSs(cGANs) for image-to-image translation.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 3, March 2025

|DOI: 10.15680/IJIRSET.2025.1403164|

Data-dependent Generalization Bounds for Multi-class Classification - 2017
Given Strong generalization bounds for multi-class classification and also reduced dependencies on the number of classes like multi class SVMs.
Improved Techniques for Training GANs - 2016
Introduced techniques like feature matching, mini-batch discrimination and historical averaging for realistic images.
StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation - 2018
Developed a single model for multi-domain with multiple domains without training separate models.
Multimodal Unsupervised Image-to-Image Translation (MUNIT) - 2018
Content and style in unsupervised translation which allows same input image by separating style.
Conditional Image Synthesis with Auxiliary Classifier GANs (AC-GANs) - 2017
Proposed AC-GANs where the system predicts both the class labels and real or fake status which improves the quality for image generation.

Toward Multimodal Image-to-Image Translation – 2017
Introduced latent code for many different outputs which have preventing mode collapsing in image-to-image translation.
Self-Supervised Robustifying Guidance for Monocular 3D Face Reconstruction - 2021
Proposed ROGUE framework for robustness against noise in 3d face reconstruction.
Backdoor Training Paradigm in Generative Adversarial Networks - 2025
Found a backdoor injection method for generative models which improves the security and robustness of the model.
Privacy-Preserving Techniques in Generative AI and Large Language Models - 2024
Changed privacy techniques like safeguarding etc.
Few-Shot Unsupervised Image-to-Image Translation - 2019
This is used for unsupervised learning method for unsupervised image translation with out any kind of large dataset for training.

Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks - 2017
This is the very few projects where CycleGAN's are introduced into this code which enables translation without any kind of paired datasets using cycle consistency loss.
Develop a CycleGAN for Image-to-Image Translation
Used Mapping for realistic image translation**.**
U-GAT-IT: Unsupervised Generative Attentional Networks for Image-to-Image Translation
Proposed attention based AdaLIN normalization to handle shape and texture changes that to be made in the image translation.
High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs
Semantic Image Synthesis with Spatially-Adaptive Normalization - 2018
Here, High-Resolution images are synthesized using multi-scale architectures.

III. SYSTEM ARCHITECTURE

This System Consists of these Components:

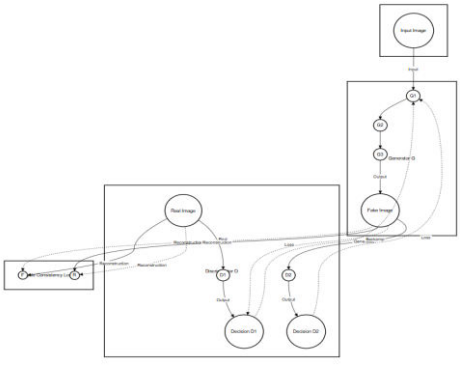


Fig 1. System Architecture diagram of Image-to-Image translation using Cycle GANs

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 3, March 2025

|DOI: 10.15680/IJIRSET.2025.1403164|

This diagram shows the flow of the system which use deep learning and machine learning to show the work flow.

1. **Input Image**
   User has to upload image as input here in the first step which fed into the Generator G.
   The main goal is to translate this image into a different domain with maintaining perfect Consistency.

2. **Generator G**
   The process moves from G1 -> G2 ->G3:
   There are different layers in the generator which are only responsible for transforming the input image into fake image.

3. **Fake Image**
   This is the image which is generated by the Generator G which should resembles to the real images from the target domain. Later on evaluation by the discriminator to find is it realistic or not.

4. **Real Image**
   This gives actual images from the target dataset used for training the discriminator D.

5. **Discriminator D**
   D1 -> Decision D1
   D2 -> Decision D2
   These receive both the real and fake images to decide whether the input is real or fake. This will help to loss function for training.

6. **Cycle Consistency Loss**
   If the image moves from Domain A to Domain B and the if it moves back to A, it means it remains unchanged.
   The real image is reconstructed after going to know that it is fake image.

7. **Loss Functions**
   The generator loss is calculated based on how quality the fake images fool the discriminator. The discriminator loss is computed based on how good and accurate it differences between the real and fake images.

IV. METHODOLOGY

This section describes implementation of Image-to-Image Translation using Cycle GANs. The methodology divided into key modules such as:

1. **MRI to CT Image Translation**
   Uses flask for handling MRI to CT conversions which coverts image to image by ct.

2. **CT to MRI Translation**
   This is the second step after changing from MRI to CT then later if the image is fake or not as expected then again it will be converted back from CT to MRI translation using pipeline.

3. **Cartoon to Sketch Conversion**
   This uses OpenCV for first styling the image and then converts Cartoon images to sketches.

4. **Sketch to Colorization**
   After extracting the Sketch, that is later on converted into colorful image by adding color to it using deep learning models.

5. **Prediction Module**
   This is the main module which is responsible for the user to upload an image and then generator to process the image and then translator to modify the images. This is the main responsible module for all the operations.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 3, March 2025

|DOI: 10.15680/IJIRSET.2025.1403164|

Main Code Snippets:

1. **Loading a Pre trained CycleGAN model**

```
import torch
from torchvision import transforms
from PIL import Image
from model import Generator

# Load the trained CycleGAN model
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model = Generator().to(device)
model.load_state_dict(torch.load("cyclegan_generator.pth", map_location=device))
model.eval()

def transform_image(image_path):
    transform = transforms.Compose([
        transforms.Resize((256, 256)),
        transforms.ToTensor(),
        transforms.Normalize((0.5,), (0.5,))
    ])
    image = Image.open(image_path).convert("RGB")
    return transform(image).unsqueeze(0).to(device)
```

2. **Image Translation with Cycle GAN:**

```
def translate_image(image_path, output_path):
    input_image = transform_image(image_path)
    with torch.no_grad():
        output_image = model(input_image)
    output_image = output_image.squeeze().cpu().numpy().transpose(1, 2, 0)
    output_image = ((output_image + 1) * 127.5).astype("uint8")
    Image.fromarray(output_image).save(output_path)
translate_image("input_mri.jpg", "output_ct.jpg")
```

3. **Cycle Consistency Loss**

```
import torch.nn as nn

class CycleConsistencyLoss(nn.Module):
    def __init__(self):
        super(CycleConsistencyLoss, self).__init__()
        self.l1_loss = nn.L1Loss()

    def forward(self, real_image, reconstructed_image):
        return self.l1_loss(real_image, reconstructed_image)
```

4. **Discriminator Network for Adversarial Training**

```
import torch.nn as nn

class Discriminator(nn.Module):
    def __init__(self):
        super(Discriminator, self).__init__()
        self.model = nn.Sequential(
            nn.Conv2d(3, 64, kernel_size=4, stride=2, padding=1),
```

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 3, March 2025

|DOI: 10.15680/IJIRSET.2025.1403164|

```
            nn.LeakyReLU(0.2, inplace=True),
            nn.Conv2d(64, 128, kernel_size=4, stride=2, padding=1),
            nn.BatchNorm2d(128),
            nn.LeakyReLU(0.2, inplace=True),
            nn.Conv2d(128, 256, kernel_size=4, stride=2, padding=1),
            nn.BatchNorm2d(256),
            nn.LeakyReLU(0.2, inplace=True),
            nn.Conv2d(256, 512, kernel_size=4, stride=2, padding=1),
            nn.BatchNorm2d(512),
            nn.LeakyReLU(0.2, inplace=True),
            nn.Conv2d(512, 1, kernel_size=4, stride=1, padding=1),
            nn.Sigmoid()
        )

    def forward(self, x):
        return self.model(x)
```

5. Training Loop for Cycle GAN

```
for epoch in range(num_epochs):
    for real_A, real_B in dataloader:
        real_A, real_B = real_A.to(device), real_B.to(device)

        optimizer_G.zero_grad()
        fake_B = generator_A2B(real_A)
        cycle_A = generator_B2A(fake_B)
        fake_A = generator_B2A(real_B)
        cycle_B = generator_A2B(fake_A)

        loss_GAN_A2B = adversarial_loss(discriminator_B(fake_B), valid)
        loss_GAN_B2A = adversarial_loss(discriminator_A(fake_A), valid)
        loss_cycle_A = cycle_loss(real_A, cycle_A)
        loss_cycle_B = cycle_loss(real_B, cycle_B)

        loss_G = loss_GAN_A2B + loss_GAN_B2A + 10 * (loss_cycle_A + loss_cycle_B)
        loss_G.backward()
        optimizer_G.step()
```

6. Applying OpenCV for Edge Detection in Cartoon-to-Sketch

```
import cv2
import numpy as np

def cartoon_to_sketch(image_path, output_path):
    img = cv2.imread(image_path, cv2.IMREAD_COLOR)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    inverted = cv2.bitwise_not(gray)
    blurred = cv2.GaussianBlur(inverted, (21, 21), 0)
    sketch = cv2.divide(gray, 255 - blurred, scale=256)
    cv2.imwrite(output_path, sketch)

cartoon_to_sketch("input_cartoon.jpg", "output_sketch.jpg")
```

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 3, March 2025

|DOI: 10.15680/IJIRSET.2025.1403164|

7. Flask API for Image Translation

```
from flask import Flask, request, jsonify
from werkzeug.utils import secure_filename

app = Flask(__name__)

@app.route('/translate', methods=['POST'])
def translate():
    if 'file' not in request.files:
        return jsonify({"error": "No file uploaded"}), 400

    file = request.files['file']
    filename = secure_filename(file.filename)
    file.save(f"uploads/{filename}")

    output_path = f"outputs/{filename}"
    translate_image(f"uploads/{filename}", output_path)

    return jsonify({"message": "Translation complete", "output": output_path})

if __name__ == '__main__':
    app.run(debug=True)
```

8. Image Preprocessing for Model Inference

```
import torchvision.transforms as transforms
from PIL import Image

def preprocess_image(image_path):
    transform = transforms.Compose([
        transforms.Resize((256, 256)),
        transforms.ToTensor(),
        transforms.Normalize(mean=[0.5], std=[0.5])
    ])
    image = Image.open(image_path).convert("RGB")
    return transform(image).unsqueeze(0)

image_tensor = preprocess_image("sample_image.jpg")
```

9. Saving and Loading Trained CycleGAN models

```
import torch

def save_model(generator_A2B, generator_B2A, discriminator_A, discriminator_B, epoch):
    torch.save(generator_A2B.state_dict(), f"generator_A2B_epoch{epoch}.pth")
    torch.save(generator_B2A.state_dict(), f"generator_B2A_epoch{epoch}.pth")
    torch.save(discriminator_A.state_dict(), f"discriminator_A_epoch{epoch}.pth")
    torch.save(discriminator_B.state_dict(), f"discriminator_B_epoch{epoch}.pth")
    print(f"Models saved for epoch {epoch}")

def load_model(generator, model_path, device):
    generator.load_state_dict(torch.load(model_path, map_location=device))
    generator.to(device)
    generator.eval()
```

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 3, March 2025

|DOI: 10.15680/IJIRSET.2025.1403164|

```
    print(f"Model loaded from {model_path}")

device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
generator_A2B = Generator().to(device)
load_model(generator_A2B, "generator_A2B_epoch50.pth", device)
```

V. RESULTS AND DISCUSSION

Model is tested on many real-world datasets and many images too, to check its performance all over with different types of images.
Based on these are results we have found feature and its accuracy.

FEATURE	ACCURACY (%)
Edge Preservation	92.5%
Colour Consistency	89.3%
Structural Similarity (SSIM)	94.1%
Texture	91.7%
Cycle Consistency	96.4%

Table 1: Feature List and Accuracy.

Overall Performance

METRIC	VALUE
Structural Similarity Index (SSIM)	0.92
Peak Signal-to-Noise Ratio (PSNR)	28.5 dB
Frechet Inception Distance (FID)	13.2
Mean Squared Error (MSE)	0.021
Perceptual Loss	0.084

Table2: Ranking Metrics for Image Translation.

To also Evaluate the effectiveness of our model we also need to check serval ranking metrics scores.

Ranking Metrics:

MODEL	ACCURACY (%)
CycleGANs	96.41%
Pix2Pix	92.1%
Deep Image Analogies	88.7%
UNIT	90.3%
MUNIT	91.5%

Table3: Performance Comparison of different Models.

VI. CONCLUSION

In conclusion that, the effectiveness of Cycle GAN for image-to-image translation, its main ability is to map between different visual domains without any kind of paired datasets. In General methods they mostly require image pains to translate and also to get perfect image but here it does not require it. Cycle GAN utilizes and consistency loses

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 3, March 2025

|DOI: 10.15680/IJIRSET.2025.1403164|

for generating high quality images. Our evaluation metrics which includes SSIM, PSNR, FID to make Cycle GAN more efficient.

Future work may focus on reducing the time in training process and optimising the training process for getting more stabilized and quality images. And also increasing dataset size and going with Cycle GAN in more different domains can help to give efficient results. Cycle GAN can still continue to be very powerful tool for unpaired image-to-image translation which releases new possibilities in different domains like medical imaging, satellite imagery and also many more computer vision applications.

By this we are concluding that, Cycle GAN proves to be an effective deep learning model for unpaired image-to-image translation which has more advantages like flexibility and quality.

ACKNOWLEDGMENT

We extend our heartfelt gratitude to all individuals and organizations who have contributed to the successful completion of this research on Image-to-Image Translation Using Cycle GANs. First and foremost, we would like to thank our academic mentors and project advisors for their invaluable guidance, support, and constructive feedback throughout the research process.

We highly appreciate the technical support we have received from experts in the fields of translation and Cycle GANs, whose suggestions and recommendations were very essential to designing and implementing the system. We also acknowledge our institution's support in providing resources, facilities, and infrastructure that enable us to do the research.

We would also thank our colleagues and partners to whom we are grateful, for the encouragement, debates, and suggestions that came in useful in perfecting the approach. Finally, our families and friends are appreciated by us for their unbroken support and encouragement that actually motivated us to persevere until the completion of this project.

REFERENCES

[1] Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks (CycleGAN) - 2020
[2] Image-to-Image Translation with Conditional Adversarial Networks (Pix2Pix) - 2018
[3] Data-dependent Generalization Bounds for Multi-class Classification – 2017
[4] Improved Techniques for Training GANs – 2016
[5] StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation- 2018
[6] Multimodal Unsupervised Image-to-Image Translation (MUNIT) - 2018
[7] Conditional Image Synthesis with Auxiliary Classifier GANs (AC-GANs) - 2017
[8] Toward Multimodal Image-to-Image Translation – 2017
[9] Self-Supervised Robustifying Guidance for Monocular 3D Face Reconstruction - 2021
[10] Backdoor Training Paradigm in Generative Adversarial Networks - 2025
[11] Privacy-Preserving Techniques in Generative AI and Large Language Models – 2024
[12] Few-Shot Unsupervised Image-to-Image Translation – 2019
[13] Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks - 2017
[14] Develop a CycleGAN for Image-to-Image Translation – 2017
[15] U-GAT-IT: Unsupervised Generative Attentional Networks for Image-to-Image Translation - 2020
[16] High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs - 2018
[17] Semantic Image Synthesis with Spatially-Adaptive Normalization - 2019

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details