FOGGY IMAGE ENHANCEMENT

A Project Report

Submitted by

DHANUSH KUMAR [CB.EN.U4CSE18432]
KRISHNA SHARMA S [CB.EN.U4CSE18434]
MAMIDELA ADITYA SAI [CB.EN.U4CSE18436]
C MONISHVER [CB.EN.U4CSE18440]

Under the guidance of
Suchithra M.,
(Asst. Professor, Department of Computer Science & Engineering)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

AMRITA VISHWA VIDYAPEETHAM

Amrita Nagar PO, Coimbatore - 641 112, Tamilnadu

November 2021

ABSTRACT

Fog is a typical atmospheric phenomenon in which dust, smoke, and other particles which greatly reduces the quality and visibility of images, thus making difficulty in further perception and understanding. Therefore, foggy image enhancement has many practical applications and is of great value for academic fields and industries. Single image defogging is a challenging problem in the field of computer vision. Existing methods towards this problem rely on learning-based approaches that require paired foggy and fog-free training example images. Moreover, foggy images have more irregular pixels than images captured in normal weather (clear images). So, it is difficult to work with foggy images.

In this paper, we propose a GAN (Generative Adversarial Network) to remove fog and enhance foggy images. The approach is to use two sets of generators and discriminators such that the output of the first GAN is given as input to the second GAN. This will help in automatic training of image-to-image translation from two different domains. Thus, we propose an effective way to make foggy image clear.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

Fog is a typical atmospheric phenomenon in which dust, smoke, and other particles which greatly reduces the quality and visibility of images, thus making difficulty in further perception and understanding. Foggy image enhancement aims to clear the fog from foggy images. Foggy image enhancement has many practical applications and is of great value for academic fields and industries.

1.1 Problem Definition

To enhance foggy images using GAN and then apply semantic segmentation on enhanced images.

Images are seriously degraded under hazy and foggy weather, which will affect the detection, tracking, and recognition of targets. Usually, the state-of-the-art frameworks provide solutions to primarily undemanding backgrounds. The problem statement focuses on enhancing these foggy images and do semantic segmentation so that the image becomes clearer and can be used in various fields.

Chapter 2

LITERATURE SURVEY

2.1 FISS GAN: A Generative Adversarial Network for Foggy Image Semantic Segmentation (2021)

This paper proposed a generative adversarial network (GAN) for foggy image semantic segmentation (FISS GAN), which contains two parts: an edge GAN and a semantic segmentation GAN. however, the result was not accurate for images with limited textures.

2.2 HardGAN: A Haze-Aware Representation Distillation GAN for Single Image Dehazing (2020)

This approach uses Haze-Aware Representation Distillation (HARD) layer and normalization layer for their GAN.

2.3 HIDeGan: A Hyperspectral-guided Image Dehazing GAN (2020)

HIDeGAN uses 2 GANs , to estimate Hyperspectral Image(HSI) and then convert to RGB channel. It is evaluated on RESIDE dataset and D-Hazy dataset.

2.4 Single Image Haze Removal Using Conditional Wasserstein Generative Adversarial Networks (2019)

This approach restores a clear image from a haze-affected image using a Wasserstein generative adversarial network. Here, quantitave metrics on images from D-Hazy dataset showed better contrast gain and lower saturated pixels in other methods.

2.5 Benchmarking Single Image Dehazing and Beyond (2018)

The paper presents a comprehensive study and evaluation of existing single image dehazing algorithms, using a new large-scale benchmark consisting of both synthetic and real-world hazy images, called REalistic Single Image DEhazing (RESIDE). Algorithms were evaluated on PSNR and SSNM. Subjective metrics correlated with human perception may lead to different results.

2.6 DehazeGAN: When Image Dehazing Meets Differential Programming (2018)

This approach uses GAN as well as differential programming for dehazing. Running Time was found to be greater than AOD-Net. However it was still better than most methods it was compared to.

2.7 Single Image Dehazing via Conditional Generative Adversarial Network (2018)

This paper presents haze removal using a conditional generative adversarial network (cGAN), where the clear image is estimated by an end-to-end trainable neural network. The proposed model doesn't work well with night hazy images or light hazy images.

2.8 A New Fast Method for Foggy Image Enhancement (2016)

The proposed method developes an image atmospheric model which is based on the Koschmieder's theory of the atmospheric vision. For some regions, colours got altered by the algorithm.

Chapter 3

PROPOSED SYSTEM

3.1 System Analysis

3.1.1 System requirement analysis

Hardware details

- **CPU Model Name:** Intel(R) Xeon(R) CPU @ 2.30GHz
- **RAM:**13GB
- **GPU:** Tesla K80 , 12GB GDDR5 VRAM

Python libraries used

- cv2
- glob
- keras
- keras_contrib
- math
- matplotplib
- numpy
- os
- skimage

3.1.2 Module details of the system

Dataset Preprocessing

Objective:

To preprocess the image dataset for it to be used in the GAN model training.

Explanation:

In our dataset, we have clear images and corresponding foggy images. We resize the clear images to 256x256 pixels to standardize the input for our model. The cycleGAN does not need paired images. So, we give the images to the GAN without pairing them.

GAN Model Design

Objective:

To train a generative adversarial network for removing fog from a foggy/hazy image.

Explanation:

Generative adversarial networks (GANs) are algorithmic architectures that use two neural networks(Generator and Discriminator), pitting one against the other (thus the "adversarial") in order to generate new, synthetic instances of data that can pass for real data. The Generator tries to fool the discriminator by generating fake fog-free image, and the discriminator has to identify the whether the image generated by the generator is real or fake. We train until both the networks reach an equilibrium and the output is a fog-free image close to the ground truth.

Image Enhancement

Objective:

To enhance the image generated by the GAN architecture, so that semantic segmentation could be done more accurately.

Explanation:

The image we get from the GAN architecture may not be large enough for it to be used for applications like semantic segmentations. So we are change the dimensions using pyramid upscaling which will ensure that edge information is retained as much as possible.

Performance Evaluation

Objective:

To evaluate the quality of the image generated by the GAN model.

Explanation:

The output of the GAN will be evaluated based on metrics like Peak signal-to-noise ratio (PSNR) and Structural Similarity Index (SSIM).

Semantic segmentation

Objective:

Use pre-trained models to do semantic segmentation for the enhanced fog-free images.

Explanation:

The goal of semantic image segmentation is to label each pixel of an image with a corresponding class of what is being represented. We propose to use pre-trained models for this, and explore different architectures that are already existing that suits well for our foggy images.

3.2 System Design

3.2.1 Flow diagram of the system

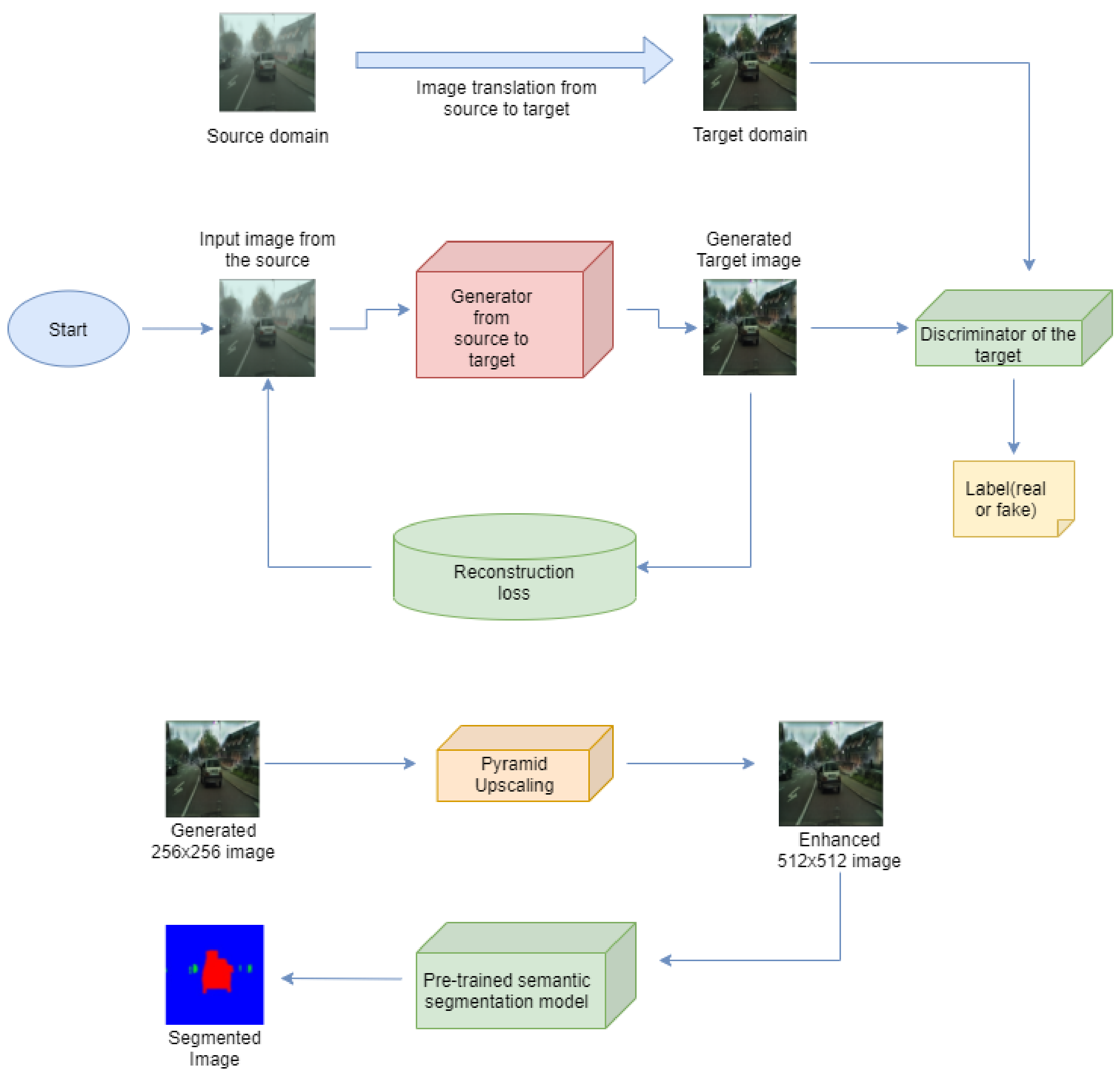


Figure 3.1: Flow Diagram

3.2.2 Architecture diagram

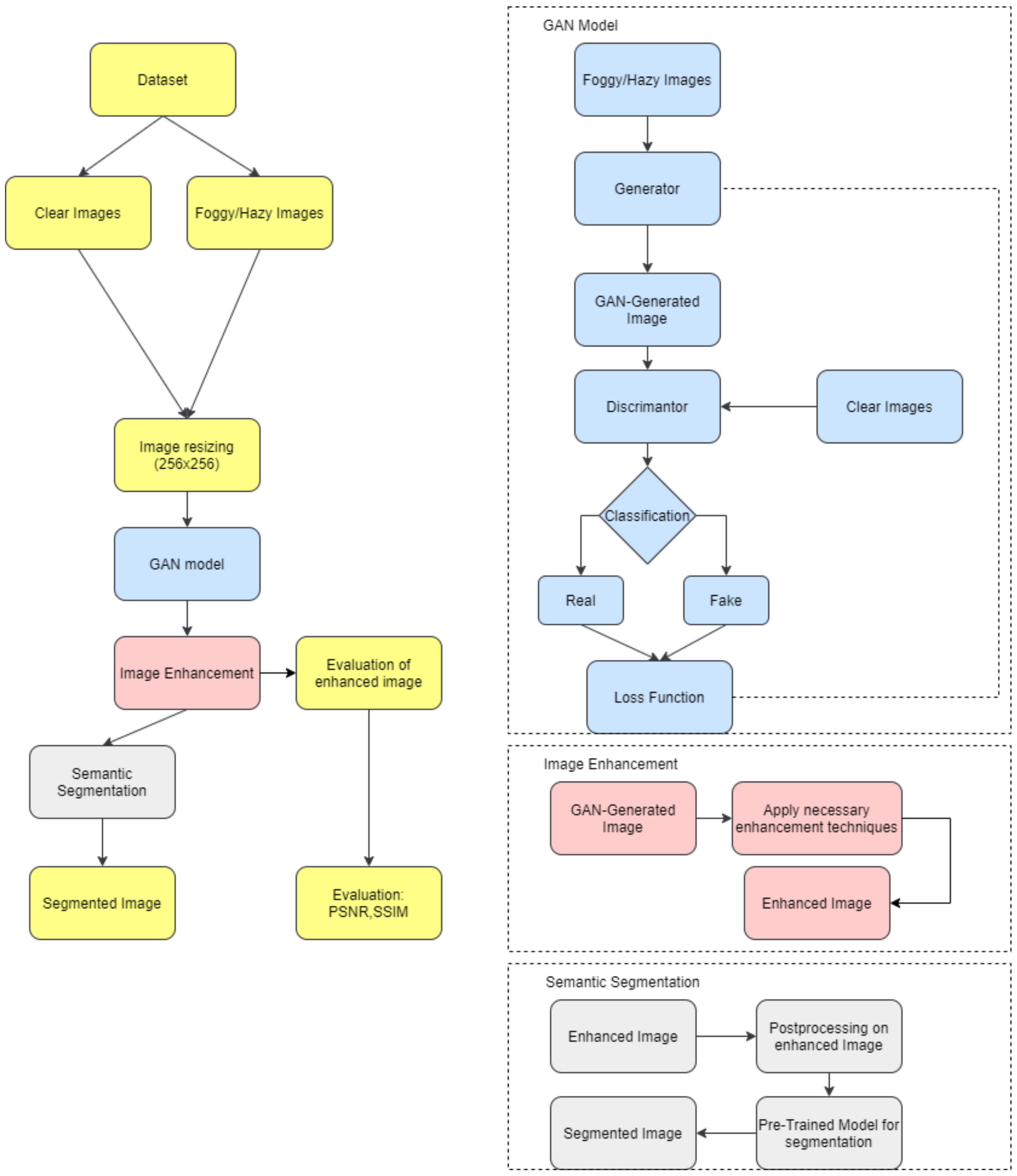


Figure 3.2: Architecture Diagram

Chapter 4

IMPLEMENTATION AND TESTING

4.0.1 CycleGAN

The CycleGAN is a technique that involves the automatic training of image-to-image translation models without paired examples. It can be used to train models for image-to-image translation using unpaired collection of images from two different domains.

In CycleGAN we treat the problem as an image reconstruction problem. We first take an image input (x) and using the generator G to convert into the reconstructed image. Then we reverse this process from reconstructed image to original image using a generator F. The most important feature of this cycleGAN is that it can do this image translation on an unpaired image where there is no relation exists between the input image and output image.

Like all the adversarial network, CycleGAN also has two parts Generator and Discriminator, the job of generator to produce the samples from the desired distribution and the job of discriminator is to figure out the sample is from actual distribution (real) or from the one that are generated by generator (fake).

It works mainly on the concept of Cycle Consistency where the image output from one generator could be used as input to second generator. Likewise, the output of second generator can be given as input to the first generator and the result should ideally match the input given to second generator.

Each generator model is optimized via the combination of four outputs with four loss functions:

- Adversarial loss (L2 or mean squared error).
- Identity loss (L1 or mean absolute error).
- Forward cycle loss (L1 or mean absolute error).
- Backward cycle loss (L1 or mean absolute error).

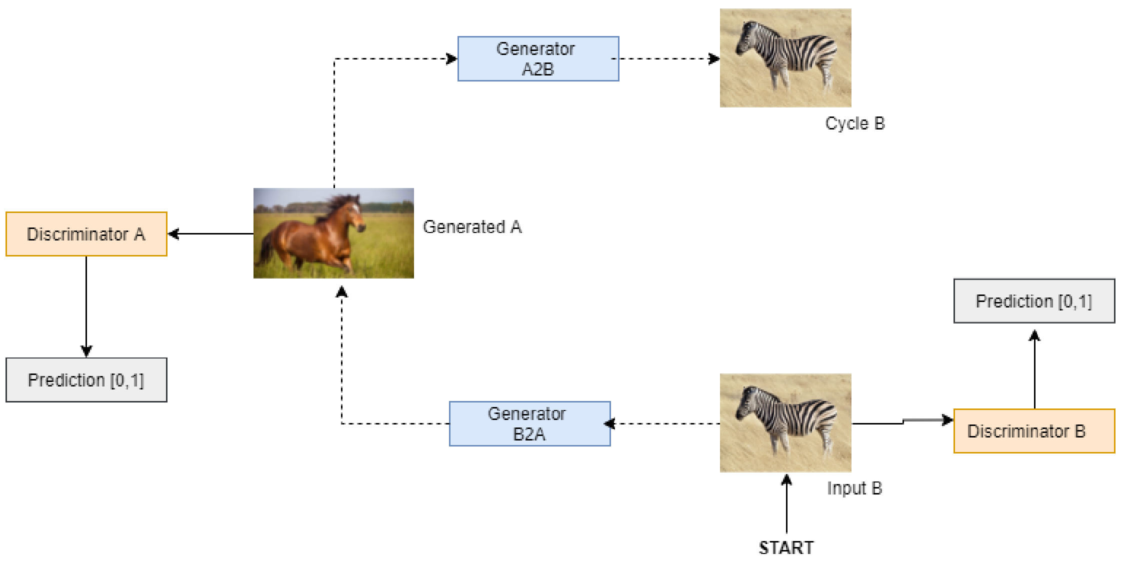


Figure 4.1: CycleGAN architecture

The diagram shows the CycleGAN architecture for converting an image with horse to one with zebra. In our case, we need to convert Foggy image to Clear image. The CycleGAN discussed also has a similar architecture.

Preprocessing Images

Foggy images were obtained from the ’FoggyCityscapes’ dataset and clear images were obtained from ’Cityscapes’ Dataset. Images in the ’FoggyCityscapes’ datasets have fog of varying intensities - 0.05, 0.1 and 0.2. In order to train the model to clear a reasonable intensity of fog, images with fog intensity of 0.1 were considered from training.

For the training set, 438 images were taken and they were resized into 256x256 pixels. This is because the input to the CycleGAN must be 256x256 pixels.

For testing the performance of the GAN model, 77 images from the validation set were taken and resized to 256x256 pixels.

Setting up environment for training

Google Colaboratoy (Colab) was used to train the model. Google Colaboratory allows us to create and run Jupyter Notebook files on a virtual machine with the option to run code on CPU, GPU or TPU (Tensor Processing Unit).

All images used for training and validation were uploaded to the drive.

The runtime for training was set to ‘GPU’ and then, the Google Drive was mounted on Colab.

Steps for building GAN

The GAN consists of generator and discriminator. The structure of discriiminator is simpler than the generator. The define_discriminator() function implements the 70×70 PatchGAN discriminator model as per the design of the model in the paper. The model takes a 256×256 sized image as input and outputs a patch of predictions. The model is optimized using least squares loss (L2) implemented as mean squared error, and a weighting it used so that updates to the model have half (0.5) the usual effect. This makes it slower than the generator.

The generator is an encoder-decoder model architecture. The model takes a source image (e.g. Foggy photo) and generates a target image (e.g. Clear photo). It does this by first downsampling or encoding the input image down to a bottleneck layer, then interpreting the encoding with a number of ResNet layers that use skip connections, followed by a series of layers that upsample or decode the representation to the size of the output image. The ResNet blocks are comprised of two 3×3 CNN layers where the input to the block is concatenated to the output of the block, channel-wise.

A composite model has two inputs for the real photos from Domain-A and Domain-B, and four outputs for the discriminator output, identity generated image, forward cycle generated image, and backward cycle generated image.

Only the weights of the first or main generator model are updated for the composite model and this is done via the weighted sum of all loss functions. The cycle loss is given more weight (10-times) than the adversarial loss as described in the paper, and the identity loss is always used with a weighting half that of the cycle loss (5-times). This is seen in the function define_composite_model().

The model is trained with the training set and is saved frequently (every 2 epochs).

Testing the model

To test the model and evaluate it with respect to PSNR and SSIM, we define a function to calculate PSNR taking the generated image a nd ground truth image as input. We also use scikit-image library's find_ssim() function in a similar manner to find SSIM.

Images from the validation set of 'FoggyCityscapes' are given as input along with corresponding clear images from 'Cityscapes' dataset and SSIM and PSNR is noted.

4.0.2 Semantic Segmentation

The goal of semantic image segmentation is to label each pixel of an image with a corresponding class of what is being represented. The dataset used was CamVid dataset.

Environment

The models were trained using pre trained weights on Google Colboratory (Colab) with runtime set to 'GPU'.

Training

Pre-trained models from segmentation-models library have been used for semantic segmentation to attain the results. The models were trained with different architectures and backbones to find out which model performs well in our dataset. Augmentation using albumentation was used for increasing the diversity of data available for training models, without actually collecting new data. The models trained were mostly based on Unet architecture.

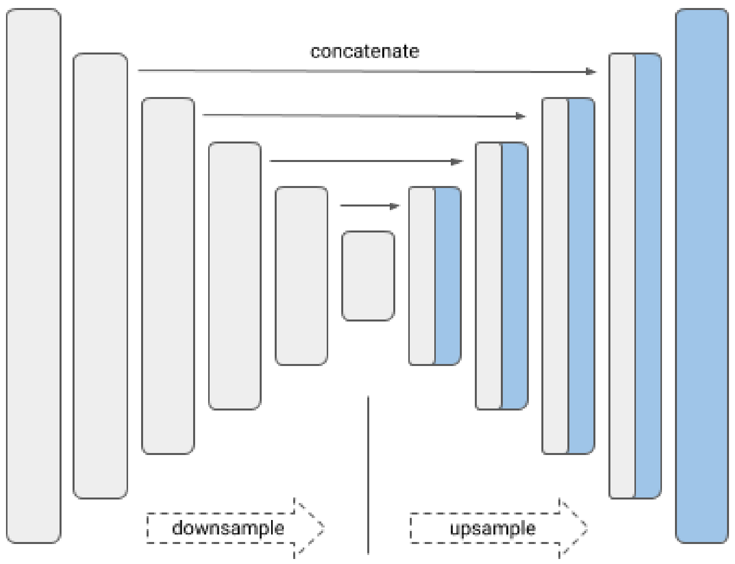


Figure 4.2: UNet architecture

The backbones included VGG, ResNet, ResNeXt, EfficientNet. Best results were found with UNet and EfficientNet.

Testing the model

The models were mainly tested based on IOU (Intersection-Over-Union) and F1 scores.

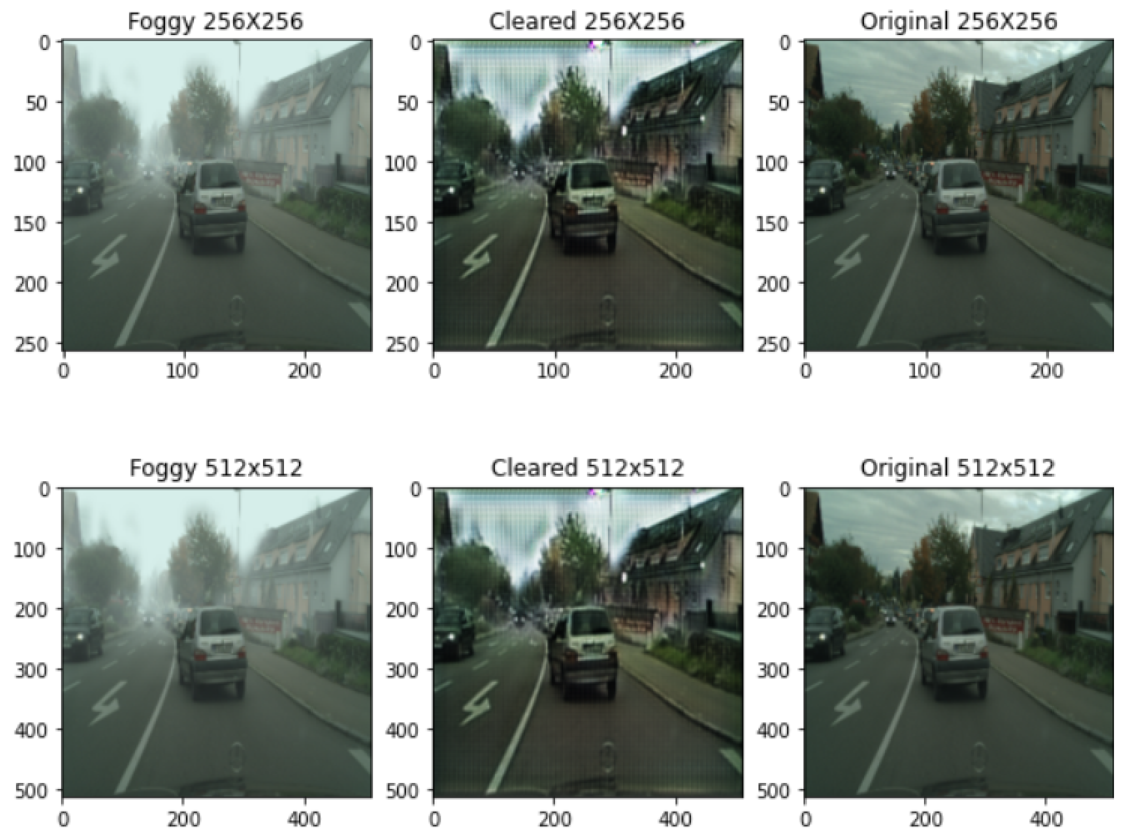
Chapter 5

RESULTS AND DISCUSSION

5.0.1 CycleGAN Results

The CycleGAN was able to clear fog from the foggy images and give images which closely resembled the ground truth. The original output from CycleGAN is a 256x256 pixel image. It is then resized to 512x512 using pyramid upscaling and its PSNR (Peak Signal to Noise Ratio) and SSIM (Structural Similarity Index) were recorded. The PSNR and SSIM values were found to be slightly better when the image was resized using pyramid upscaling as opposed to Linear interpolation. This is because pyramid upsacling retains edge information better than linear interpolation while upscaling.

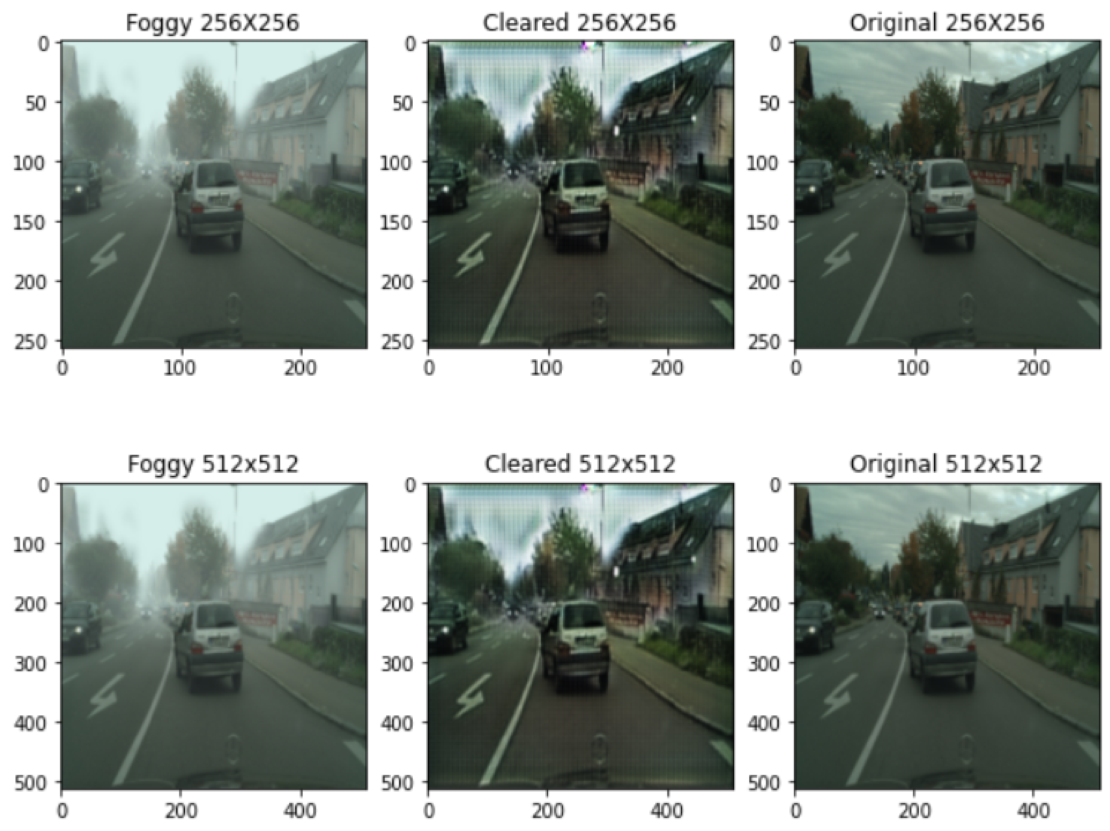
Resizing using cv2 library's inbuilt resize() function



```
PSNR: 67.77739607001513
SSIM: 0.8415428632538455
```

Figure 5.1: Results using cv2.resize()

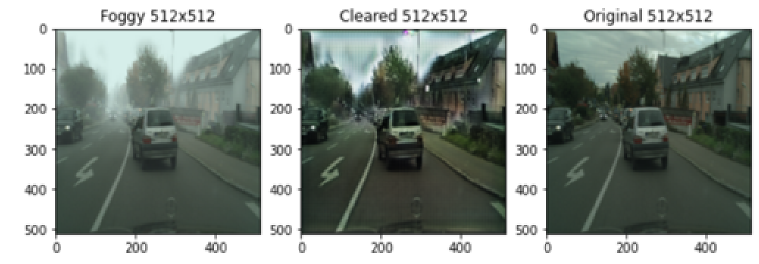
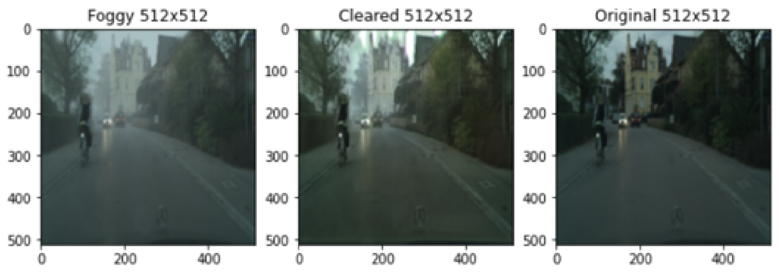
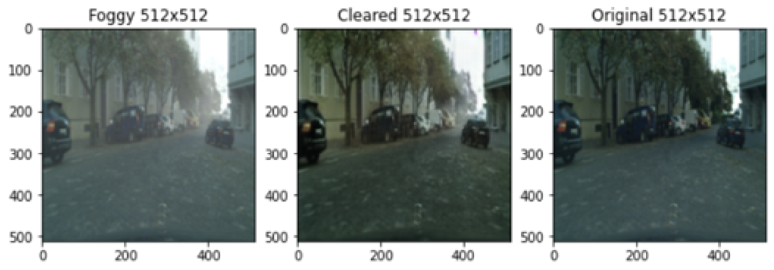
Resizing using pyramid upscaling



```
PSNR: 67.89535258504083
SSIM: 0.8545020665989869
```

Figure 5.2: Results using Pyramid Upscaling

Below are the results on a few images from the validation set.

Image	PSNR	SSIM
	67.895	0.854
	66.695	0.891
	68.031	0.912

Summary of results

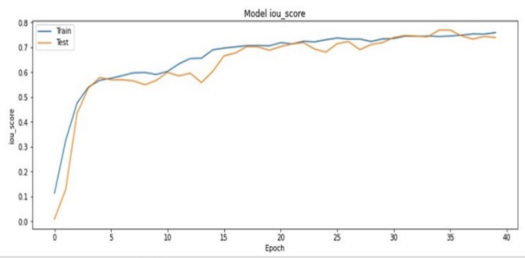
	PSNR	SSIM
Original Output	68.18	0.859
Linear Interpolation	68.574	0.894
Pyramid Upscaling	68.666	0.902

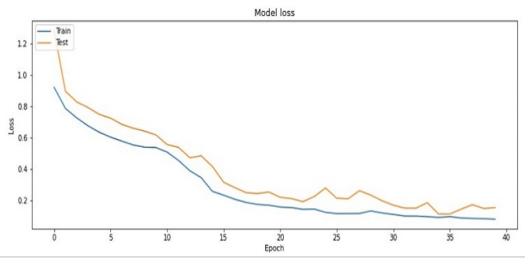
5.0.2 Semantic Segmentation results

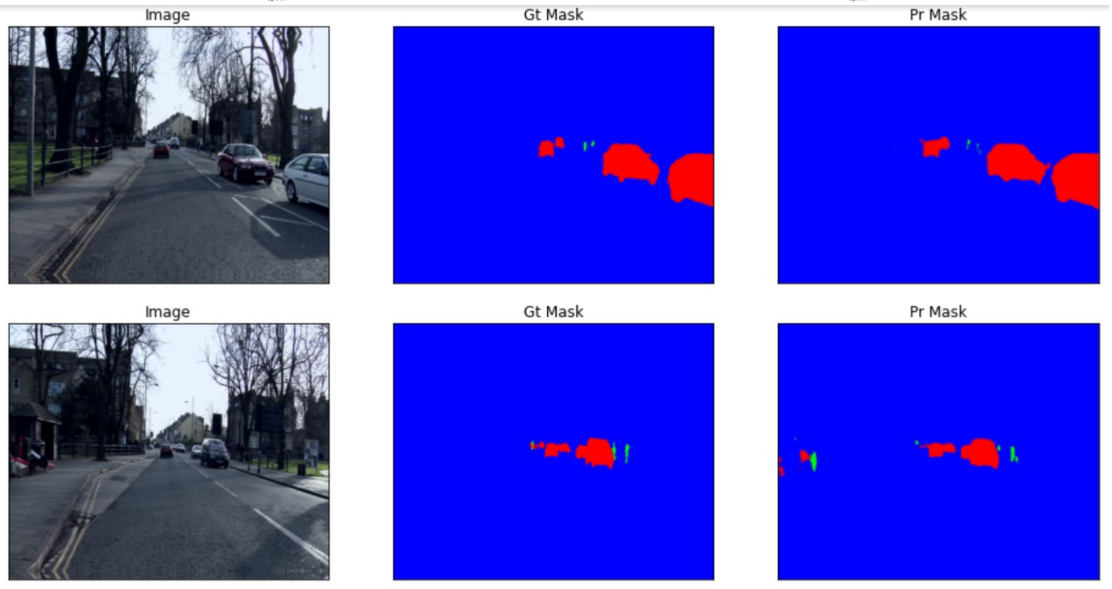
We have used pre-trained models for semantic segmentation to attain the results. We trained our models with different architectures and backbones to find out which model performs well in our dataset. Below is the comparison table for different models on IOU and F1 scores.

Results for semantic segmentation of images from CamVid dataset for two classes - pedestrian and cars are shown below.

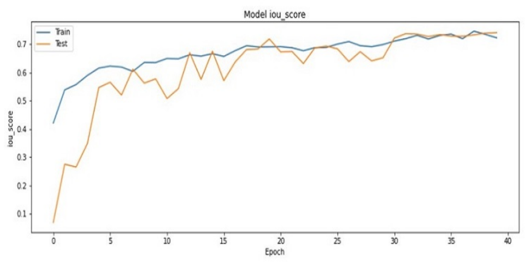
UNet+effiecentnetb3

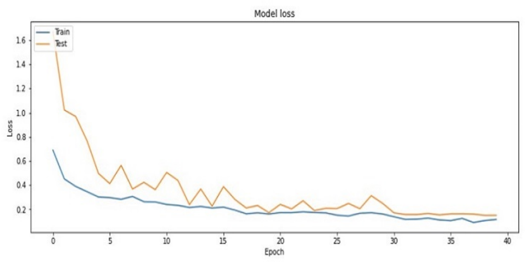


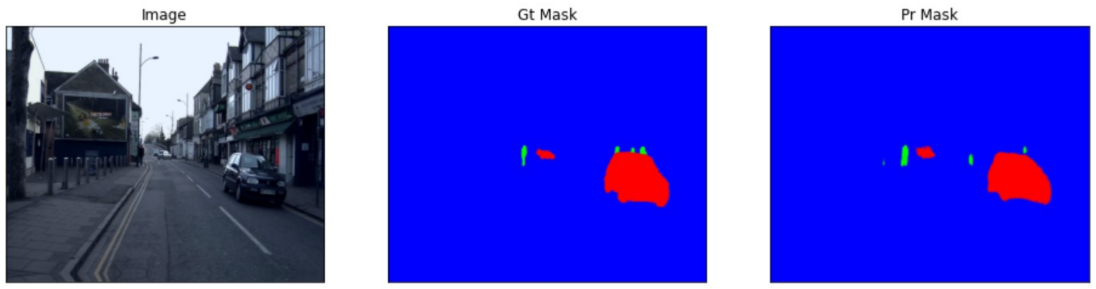




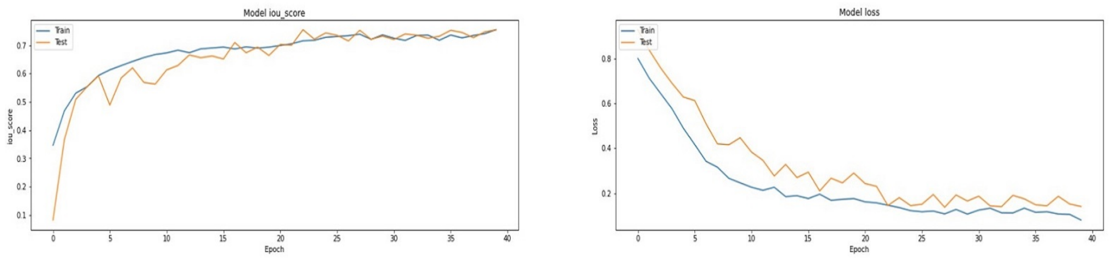
Unet+efficientnetb3(LR:0.001)

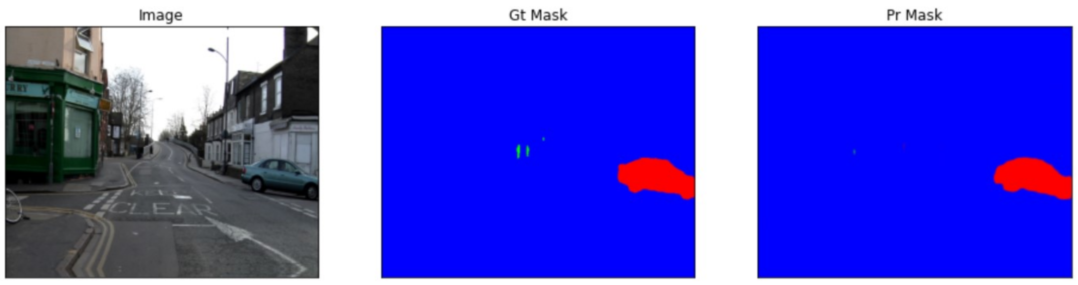






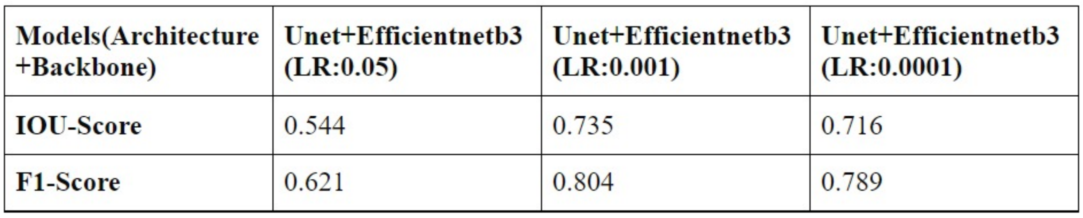
Unet+resnext50





Summary of Semantic Segmentation results

The learning rates were also tweaked and the best results were obtained for Learning Rate = 0.001 as shown in the table below:



Models(Architecture +Backbone)	Unet+Efficientnetb3 (LR:0.05)	Unet+Efficientnetb3 (LR:0.001)	Unet+Efficientnetb3 (LR:0.0001)
IOU-Score	0.544	0.735	0.716
F1-Score	0.621	0.804	0.789

Chapter 6

CONCLUSION

- Images acquired by a visual system are seriously degraded under hazy and foggy weather, which will affect the detection, tracking, and recognition of targets.
- We propose a solution using Generative Adversarial Network to generate fog-free images with great accuracy and then do semantic segmentation on the enhanced fog-free image.
- The approach using CycleGAN and pyramid upscaling has given us very good results and semantic segmentation using pre trained models has also worked out quite well.

Chapter 7

FUTURE ENHANCEMENT

- Semantic Segmentation with different architecture has been done. We could also try different backbones and see if the results improve.
- At present, only two classes are considered for semantic segmentation. In the future, more classes like Road, buildings could also be considered.

REFERENCES

1. Abbaspour, M. J., Yazdi, M., and Masnadi-shirazi, M. (2016). "A new fast method for foggy image enhancement." *2016 24th Iranian Conference on Electrical Engineering (ICEE)*. 1855–1859.

2. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. (2016). "The cityscapes dataset for semantic urban scene understanding." *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

3. Deng, Q., Huang, Z., Tsai, C.-C., and Lin, C.-W. (2020). "Hardgan: A haze-aware representation distillation gan for single image dehazing." *ECCV*.

4. Ebenezer, J. P., Das, B., and Mukhopadhyay, S. (2019). "Single image haze removal using conditional wasserstein generative adversarial networks." *2019 27th European Signal Processing Conference (EUSIPCO)*. 1–5.

5. Engin, D., Genc, A., and Ekenel, H. K. (2018). "Cycle-dehaze: Enhanced cyclegan for single image dehazing." *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 938–9388.

6. Li, B., Ren, W., Fu, D., Tao, D., Feng, D., Zeng, W., and Wang, Z. (2019). "Benchmarking single-image dehazing and beyond." *IEEE Transactions on Image Processing*, 28(1), 492–505.

7. Li, R., Pan, J., Li, Z., and Tang, J. (2018). "Single image dehazing via conditional generative adversarial network." *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8202–8211.

8. Liu, K., Ye, Z., Guo, H., Cao, D., Chen, L., and Wang, F.-Y. (2021). "Fiss gan: A generative adversarial network for foggy image semantic segmentation." *IEEE/CAA Journal of Automatica Sinica*, 8(8), 1428–1439.

9. Mehta, A., Sinha, H., Narang, P., and Mandal, M. (2020). "Hidegan: A hyperspectral-guided image dehazing gan." *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 846–856.

10. Sakaridis, C., Dai, D., Hecker, S., and Van Gool, L. (2018). "Model adaptation with synthetic and real data for semantic dense foggy scene understanding." *European Conference on Computer Vision (ECCV)*. 707–724.

11. Zhu, H., Peng, X., Chandrasekhar, V., Li, L., and Lim, J.-H. (2018). "Dehazegan: When image dehazing meets differential programming. 1234–1240.