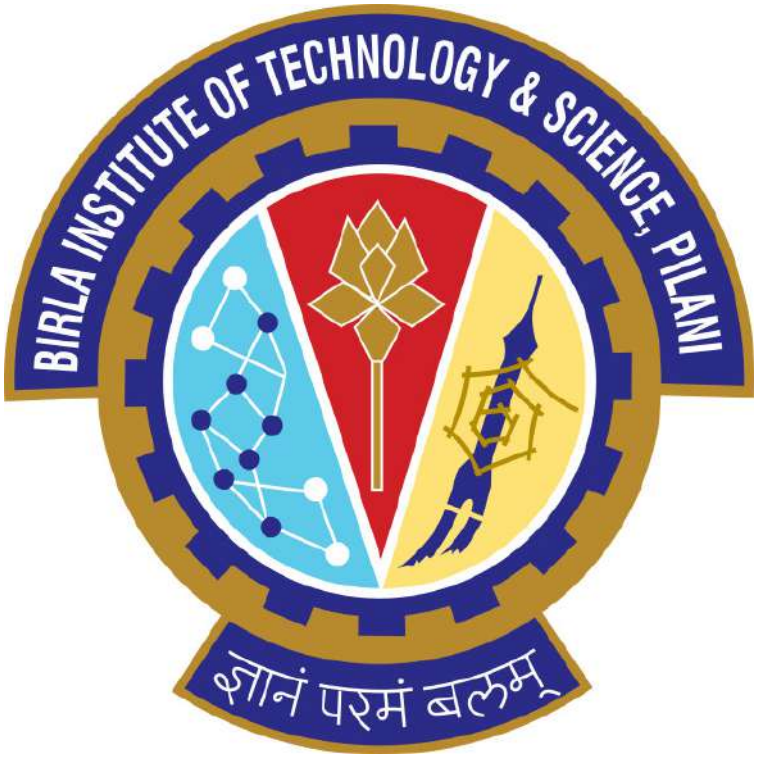
CSF425 Deep Learning Project: Dehazing of Images using Conditional GANs

Yash Pandey 2021A7PS0661P
Vikram Komperla 2021A4PS1427P
Adhvik Ramesh 2021AAPS2465P

April 26, 2024



BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
March, 2024

Contents

1 Introduction

The removal of haze from images is a crucial task in computer vision with wide-ranging applications, from enhancing visibility in surveillance footage to improving the quality of outdoor photography. In this project, we address the challenge of dehazing images using Generative Adversarial Networks (GANs). GANs have demonstrated remarkable capabilities in generating realistic images by pitting a generator against a discriminator in a competitive learning framework.

The goal of our project is to develop a robust dehazing model capable of transforming hazy images into clear, visually appealing counterparts. Leveraging a training dataset comprising 7619 image pairs of hazy and ground truth images, along with a validation dataset of around 1000 images, we explore the potential of GANs in this context.

Our approach involves employing conditional GANs, where both the generator and discriminator are conditioned on the hazy input image. This setup allows the generator to learn the mapping from hazy images to their corresponding clear versions, guided by the feedback from the discriminator.

To augment the training dataset and improve the generalization capability of our models, we apply various augmentation techniques to the provided hazy images. These techniques include adding haze, random occlusion, fog, and shadows, aimed at simulating diverse real-world conditions.

Furthermore, we experiment with intermediate representations such as transmission maps and semantic maps as additional conditioning inputs to the GAN. These representations capture crucial information about the haze distribution and scene semantics, potentially aiding the dehazing process.

2 Overview

The following section covers the model architectures we have experimented with and the loss functions that we have used in different combinations, along with a summary of the augmentations we have used.

2.1 Model Architectures

We have approached the problem of dehazing images by using conditional GANs. In this setup, the generator and discriminator receive the following inputs:

1. Generator: Hazy Image as input and generated image as output
2. Discriminator: Hazy Image as conditional input and Ground Truth Image or Generated Image as input that needs to be classified

The architectures used are as follows: Model 1's architecture and the use of different transforms for clean and hazy images are inspired by [6].

1. Model 1 (U-Net Generator with PatchGAN discriminator)
   (a) **Generator**: The generator comprises an encoder-decoder architecture, featuring a series of encoder and decoder blocks. The input hazy image undergoes downsampling and feature extraction through multiple down-sampling blocks in the encoder section, capturing hierarchical features. Bottleneck layers distill the essential representation of the input. The decoder section upsamples the features and progressively reconstructs the dehazed image, with skip connections facilitating the fusion of low-level and high-level features for effective reconstruction. Dropout layers are integrated to enhance generalization and prevent overfitting during training.
   (b) **Discriminator**: The discriminator takes pairs of input images, consisting of a hazy image and its corresponding ground truth clear image, and processes them through a series of convolutional layers. Initially, the input images are concatenated and passed through an initial convolutional layer with LeakyReLU activation. Subsequently, the concatenated feature maps are further processed by a series of CNN blocks, each comprising a convolutional layer, followed by batch normalization and LeakyReLU activation. These blocks progressively extract and abstract features from the input images. Finally, a convolutional layer with a kernel size of 4 and a stride of 1 produces a single-channel output, representing the discriminator's prediction of whether the input images are real or generated.

2. Model 2 (Simpler U-Net generator with PatchGAN discriminator)

   (a) **Generator**: The generator architecture is based on a Multi-Scale U-Net design, where the input hazy image undergoes multi-resolution processing through a series of downsampling and upsampling blocks. Each block in the encoder section captures hierarchical features at different scales, while the decoder section progressively reconstructs the dehazed image using skip connections to fuse low-level and high-level features for more effective restoration.

   (b) **Discriminator**: The discriminator follows a PatchGAN architecture, taking pairs of input images and processing them through multiple convolutional layers. It evaluates the realism of image patches rather than the entire image, enabling more localized and detailed feedback to the generator. The discriminator's output indicates the likelihood of each patch being real or generated.

3. Model 3 (Baseline CNN Generator with W-Net Discriminator)

   (a) **Generator**: The generator features a baseline CNN architecture with a series of downsampling and upsampling blocks. It transforms the input hazy image into a dehazed image by iteratively enhancing features and spatial resolution. Each block comprises convolutional layers followed by batch normalization and activation functions to capture and amplify image features.

   (b) **Discriminator**: The discriminator adopts a W-Net design, processing pairs of input images through convolutional layers to evaluate their realism. Unlike traditional discriminators, the W-Net discriminator considers both the entire image and local patches, enabling more comprehensive feedback to the generator. This architecture encourages the generator to produce realistic and visually pleasing dehazed images.

4. Model 4: Another simple U-Net Generator with symmetric skip connections and PatchGAN Discriminator

   (a) **Generator:** The simple U-Net with skip connections generator architecture is inspired by the U-Net design, tailored for image translation tasks. It features a combination of encoding and decoding layers to transform input hazy images into dehazed versions.

   - **Encoding Layers:** The encoding layers consist of convolutional blocks (down_conv) responsible for feature extraction and downsampling. These blocks progressively reduce the spatial dimensions of the input while increasing the number of feature maps to capture hierarchical features.
   - **Decoding Layers:** The decoding layers comprise transposed convolutional blocks (up_trans), followed by convolutional blocks (up_conv) for upsampling and feature refinement. Skip connections are utilized to fuse low-level and high-level features from the encoding and decoding pathways, aiding accurate image reconstruction.
   - **Output Layer:** The final output layer consists of a convolutional operation (out) to generate the dehazed image with three channels corresponding to RGB color channels.

5. Model 5: Complex Generator with UNet-like Architecture and PatchGAN discriminator

   (a) **Generator:** The enhanced generator exhibits a UNet-like architecture just like model 4 but a more complex version with more encoding and decoding layers similar to the one used in [2], combining encoding and decoding pathways to transform hazy images into dehazed versions. Notable features include:

   - **Encoding Layers:** The encoding layers consist of a series of convolutional blocks (down_conv) responsible for feature extraction and downsampling. These blocks progressively reduce the spatial dimensions of the input while capturing hierarchical features.
   - **Decoding Layers:** The decoding layers comprise transposed convolutional blocks (up_trans), followed by convolutional blocks (up_conv) for upsampling and feature refinement. Skip connections are employed to fuse low-level and high-level features from the encoding and decoding pathways, aiding in accurate image reconstruction.
   - **Output Layer:** The final output layer consists of a convolutional operation (out), producing the dehazed image with three channels corresponding to RGB color channels.

2.2 Loss Functions

The various loss functions that we have combined in various ways for our experiments are as follows: The conditions used for the WGAN loss such as using RMSprop optimiser with a learning rate of 0.00005, clamping of discriminator parameters, and the range for doing so, along with the formulas for calculation of losses for generator and discriminator are based on research in [1] and described in:

1. **Adversarial Loss (Min-Max Loss):** This loss function aims to train the generator to generate images that are indistinguishable from real images by the discriminator. It is based on the minimax game between the generator and discriminator, where the generator tries to minimize the probability of the discriminator correctly classifying generated images as fake.

2. **Wasserstein Loss:** The Wasserstein loss, also known as the Earth-Mover's distance, is used in Wasserstein Generative Adversarial Networks (WGANs). It measures the discrepancy between the distributions of real and generated samples. Unlike traditional GANs, WGANs use a critic instead of a discriminator, and the Wasserstein loss provides a more stable training process by avoiding problems like mode collapse.

3. **L1 Loss:** The L1 loss, also known as the mean absolute error (MAE), measures the average absolute difference between corresponding elements of the generated and target images. It encourages the generator to produce images that are close to the ground truth images in terms of pixel-wise intensity.

4. **Perceptual Loss:** Perceptual loss, also referred to as content loss, captures high-level image features extracted from pre-trained deep neural networks, such as VGG or ResNet. It measures the perceptual similarity between the generated and target images based on feature representations at multiple layers of the neural network. Perceptual loss is effective in preserving image content and structure.

5. **L2 Loss:** The L2 loss, also known as the mean squared error (MSE) loss, measures the average squared difference between corresponding elements of the generated and target images. It encourages the generator to produce images that closely match the ground truth images in terms of pixel-wise intensity and is effective in promoting pixel-wise similarity.

It's important to note that although perceptual loss is effective, it was seen to be very computationally expensive with the exact imitation of [2] taking 60+ seconds per batch to train, leading us to omit it from our experiments to avoid excessive computational overhead. Instead, we have used L2 Mean-square-error loss in its place in the following formula to combine adversarial, l_1 loss, and perceptual loss and used the values of $\alpha = 1$, $\beta = 150$, $\gamma = 150$ based on the research done in [2].

$$\mathcal{L} = \alpha L_A + \beta L_P + \gamma L_T$$

Figure 1: Combination of adversarial, perceptual and l1 loss in [1]

where:

- L_A is the adversarial loss,
- L_T is the L1 regularisation gradient prior,
- L_P is the perceptual loss,
- α, β, γ are the weights assigned to each loss term.

2.3 Data Augmentation

We have applied data augmentation on top of the provided dataset by adding a low-haze image and a medium-haze image corresponding to every clean image to triple the size of the dataset. The idea for using images with a varied level of haze came from [4] who used a dataset with 5 levels of hazy images, and the code for adding haze was inspired from [5], which we modified to add haze in varying intensity.

2.4 Chosen Model & Loss: Model-1 (U-Net CGAN) with L1, L2, Adversarial and Regularisation Loss (Experiment - 2)

Upon analyzing the experimental results, several observations emerge, shedding light on the performance of different models in reducing haze from images. Model-1, which employed a combination of L1, L2, regularisation, and adversarial losses, showcased the most promising results compared to other variants. With a mean PSNR of 69.66 and a mean SSIM of 0.7978, it demonstrated superior image quality and structural similarity with ground truth images. The median PSNR and SSIM values of 69.72 and 0.8100, respectively, further affirm the consistent performance of Model-1 across the dataset. Notably, Model-1 effectively addressed the challenge of haze removal, generating clean images with remarkable clarity and fidelity. The inclusion of diverse loss functions facilitated comprehensive learning, enabling the model to capture both global and local image features effectively.

Model	ConditionalGAN	Schedulers	Loss Function	Mean PSNR	Mean SSIM
Model 1	Yes	No	L1, L2, Reg, Adv.	21.53	0.7069
Model 2	Yes	Yes	L1, Adv.	19.27	0.5725
Model 3	No	Yes	With Schedulers	12.44	0.1386
Model 3	No	No	L1, Adv.	12.66	0.0421
Model 4	Yes	Yes	L1, Adv.	16.09	0.5493
Model 5	Yes	Yes	L1, Adv.	22.89	0.884

Table 1: Comparison of Mean PSNR and Mean SSIM for different GAN Models

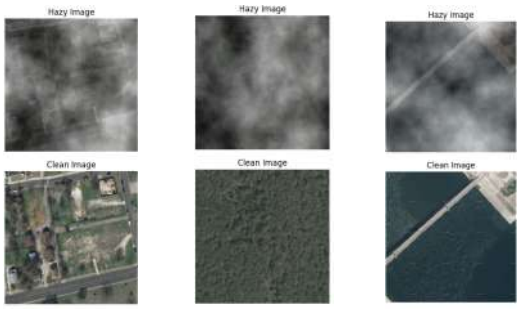
Model-2, which utilized L1 and adversarial losses, also exhibited commendable performance, albeit slightly inferior to Model-1. However, Models 3 and 4, with variations in loss functions and architectures, demonstrated comparatively lower mean PSNR and SSIM values. Moreover, despite Model-5's superior performance, it's architecture is much heavier than Model-1 and consequently, Model-1 was selected as the preferred choice for its superior performance with the comprehensive loss function combination from [2].

3 Augmentation of Hazy Images

3.1 Reasons for Augmenting

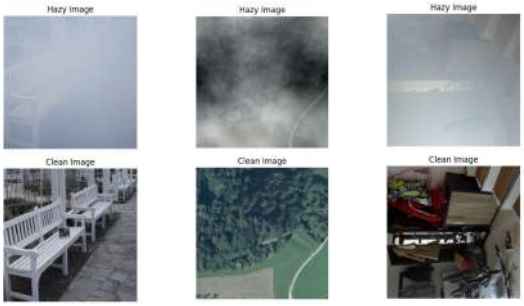
From the dataset, we observed the following points because of which we decided to proceed with image augmentation:

1. Varying amount of haze in provided hazy versions of image. A few images had an extreme amount of haze such that the ground truth image essentially lost all valuable information. The generator cannot make very good constructions from such images and wouldn't be able to improve much based on the discriminator's feedback
2. Low amount of images as compared to implementations in research paper. [4] used over 60 thousand images for their experiments. Te provided dataset only has 7000 images and hence more augmented images would help the generator learn better.



(a) (b) (c)

Figure 2: Overly hazed images versus corresponding clean images in dataset (Part 1)



(a) (b) (c)

Figure 3: Overly hazed images versus corresponding clean images in dataset (Part 2)

3.2 Augmentation Experiments

We experimented with the following types of augmentations applied to ground truth images provided in the dataset. The code for adding fog, shadows to images was inspired from [5].

1. Adding Haze
2. Random Occlusion
3. Adding Fog
4. Adding Shadows

3.3 Chosen Augmentation: Varying Haze

However, to cohere with the hazy images provided in the dataset, we applied the adding haze functions with the following varying levels of haze. This is the augmentation that was used for our dataset, inspired from [4] who used a dataset with 5 different levels of haze.

1. Low Haze Image
2. Medium Haze Image
3. High Haze Image

Figure 4: Data Augmentation: Adding haze to clean images in varying intensity

However, to constrain the size of the dataset, we used only 1 low haze image and 1 medium haze image generated from each ground truth image. The total size of the augmented dataset was hence $7619 + 7619 * 2 = 22,857$

4 Intermediate Representations

Based on research papers (no paper in particular), we experimented with the following intermediate representations as possible conditioning input to the GAN. The code for semantic map generation using a pre-trained deeplap model was inspired from [3]:

4.1 Transmission Maps

Utilizing transmission maps as a representation of the haze in an image, we wished to capture the extent to which light is attenuated as it travels through the image. These maps provide crucial information about the haze distribution and help in modeling the dehazing process effectively. The following two methods were used to generate transmission maps from hazy images:

4.1.1 Dark Channel Prior Method:

1. Compute the dark channel of the hazy image by taking the minimum value across color channels.
2. Estimate the atmospheric light as the maximum intensity in the dark channel.
3. Normalize the dark channel by dividing it by the atmospheric light.
4. Compute the transmission map using the normalized dark channel and a parameter (commonly denoted as omega).

Figure 5: Transmission maps created from hazy images: Observed to reveal considerable information

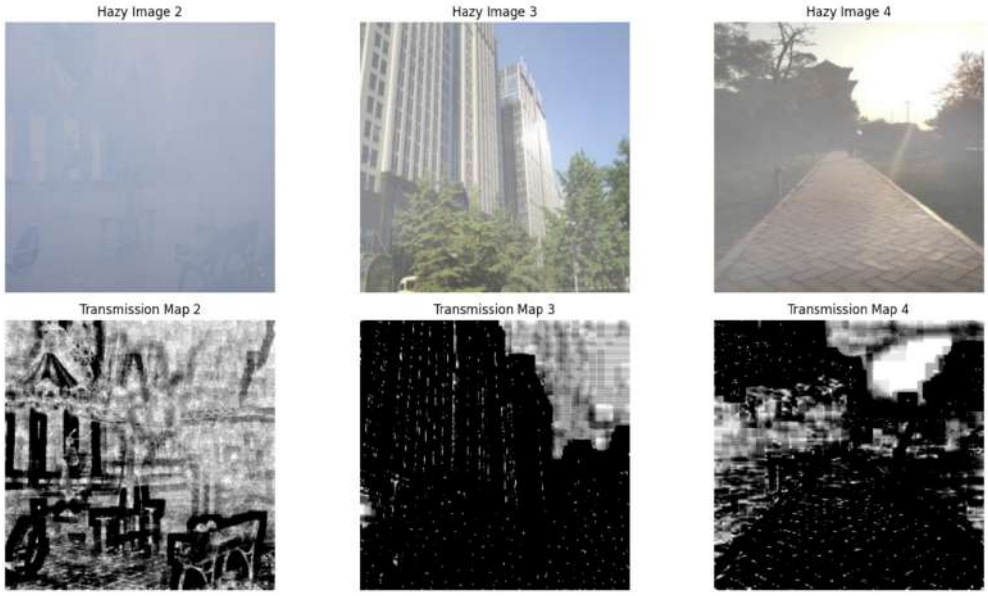


Figure 6: Transmission maps created from hazy images: Observed to reveal considerable information

4.1.2 Guided Filter Method:

This method was inspired from []. The guided filter of the OpenCV library was used in the final generating step.

1. Compute the dark channel of the hazy image.
2. Estimate the atmospheric light as the maximum intensity in the dark channel.
3. Normalize the dark channel.
4. Compute the transmission map using the atmospheric light and omega parameter.
5. Apply guided filter for refinement

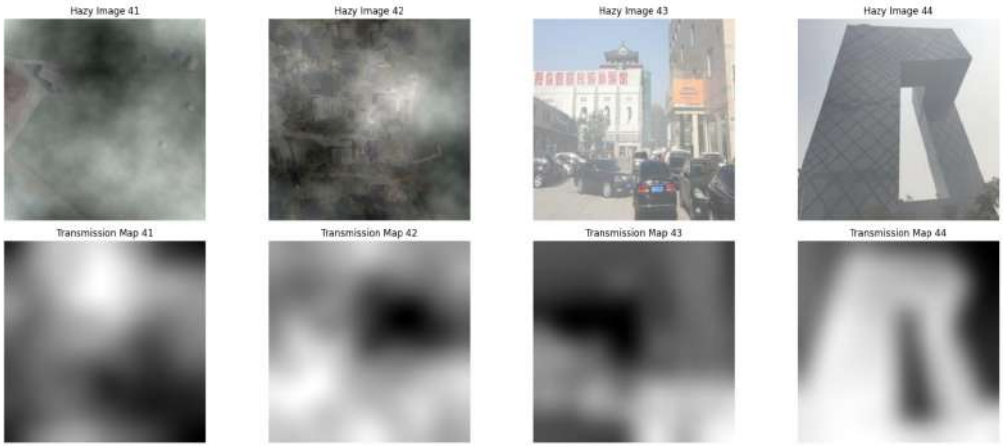


Figure 7: Transmission maps using guided filters created from hazy images: Observed to reveal considerable information

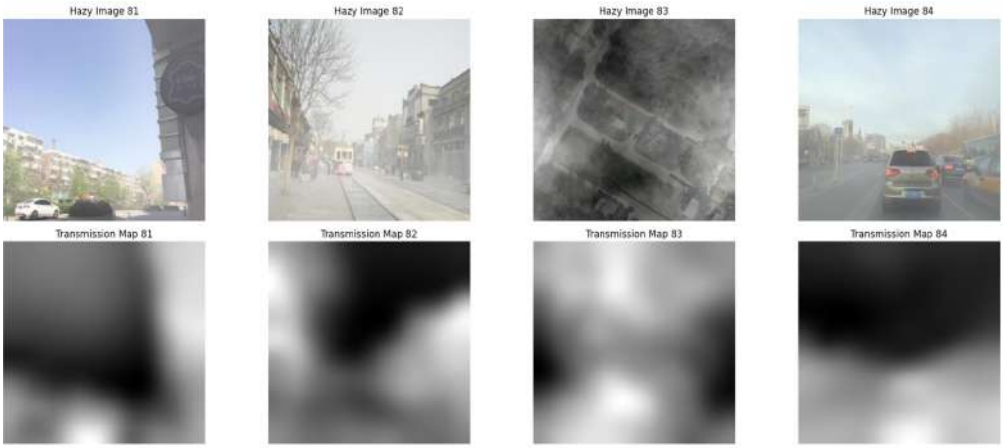


Figure 8: Transmission maps using guided filters created from hazy images: Observed to reveal considerable information

4.2 Semantic maps

were generated using the DeepLabV3 model featuring a ResNet101 backbone, pretrained on the COCO dataset, specifically designed for semantic segmentation tasks. Leveraging atrous convolution, this architecture efficiently captures multi-scale contextual information. By integrating both local and global context, the model accurately assigns class labels to individual pixels in the input image, resulting in the creation of high-resolution semantic segmentation maps. However, a majority of the maps for the hazy image in the dataset were monocolor.

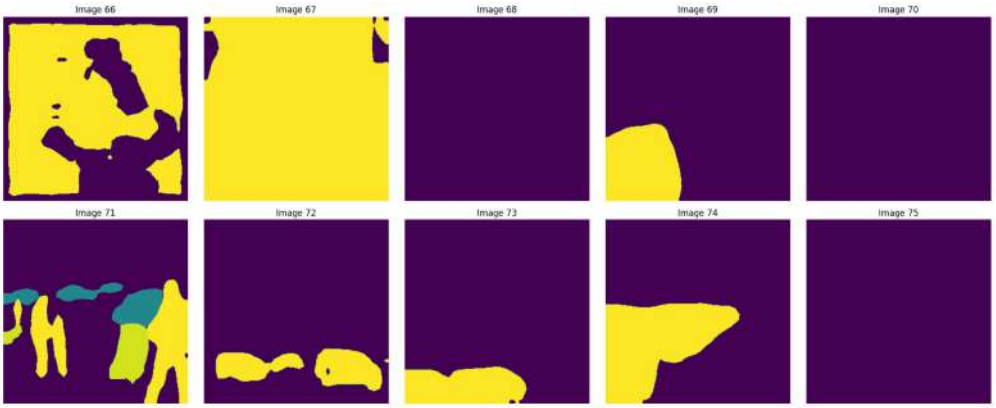


Figure 9: Generated semantic maps from hazy imagess

The below sections describe various important experiments conducted by us. We have worked with a variety of model architectures under different setups with different loss functions and learning rate schedulers as described below.

5 Experiment 1: Model-1 with L1 Loss and Adversarial Loss

For this experiment, we utilize Model-1 architecture, which consists of an encoder-decoder generator and a discriminator network. The generator incorporates an encoder-decoder architecture, featuring a series of encoder and decoder blocks. The input hazy image undergoes downsampling and feature extraction through multiple down-sampling blocks in the encoder section, capturing hierarchical features. Bottleneck layers distill the essential representation of the input. The decoder section upsamples the features and progressively reconstructs the dehazed image, with skip connections facilitating the fusion of low-level and high-level features for effective reconstruction. Dropout layers are integrated to enhance generalization and prevent overfitting during training.

The discriminator takes pairs of input images, consisting of a hazy image and its corresponding ground truth clear image, and processes them through a series of convolutional layers. Initially, the input images are concatenated and passed through an initial convolutional layer with LeakyReLU activation. Subsequently, the concatenated feature maps are further processed by a series of CNN blocks, each comprising a convolutional layer followed by batch normalization and LeakyReLU activation. These blocks progressively extract and abstract features from the input images. Finally, a convolutional layer with a kernel size of 4 and a stride of 1 produces a single-channel output, representing the discriminator's prediction of whether the input images are real or generated.

5.1 Loss Functions

The experiment utilizes the following loss functions:

- **Binary Cross-Entropy (BCE) with Logits Loss:**
  - Formula:
$$\text{BCE\_with\_Logits}(x,y) = \frac{1}{N}\sum_i [y_i \cdot \log(\sigma(x_i)) + (1-y_i)\cdot \log(1-\sigma(x_i))]$$
  - Combines sigmoid activation and binary cross-entropy.
  - Measures the difference between predicted probabilities and target labels.
  - Guides the discriminator to distinguish between real and generated images effectively.
- **L1 Loss:**
  - Formula:
$$\text{L1 Loss}(x,y) = \frac{1}{N}\sum_i |x_i - y_i|$$
  - Also known as Mean Absolute Error (MAE).
  - Calculates absolute pixel-wise differences between generated and ground truth images.
  - Encourages the generator to produce outputs resembling ground truth images in intensity values.

The total loss was calculated as follows:

- **Total Generator Loss:**

$$\text{Total Generator Loss} = \lambda_{adv} \times \text{BCE\_with\_Logits}(G(\text{hazy\_imgs}), \text{real\_labels})$$
$$+\lambda_{res} \times \text{L1 Loss}(G(\text{hazy\_imgs}), \text{clean\_imgs}) + \lambda_{per} \times \text{L2 Loss}(G(\text{hazy\_imgs}), \text{clean\_imgs})$$
$$+\lambda_{reg} \times \left(\sum_i \left|\frac{\partial G}{\partial x_i}\right| + \sum_i \left|\frac{\partial G}{\partial y_i}\right|\right)$$

- **Total Discriminator Loss:**

$$\text{Total Discriminator Loss} = \text{BCE\_with\_Logits}(\text{D(hazy\_imgs, clean\_imgs)}, \text{real\_labels})$$
$$+\text{BCE\_with\_Logits}(\text{D(hazy\_imgs, G(hazy\_imgs))}, \text{fake\_labels})$$

5.2 Optimizers

The optimization process employs the following optimizer:

- **Adam Optimizer:**
  - Combines advantages of AdaGrad and RMSProp optimizers.
  - Maintains adaptive learning rates for each parameter.
  - Uses exponentially decaying averages of past gradients and squared gradients.
  - Facilitates faster convergence and better generalization.

5.3 Learning Rate Schedulers

To dynamically adjust learning rates during training, the experiment incorporates the following scheduler:

- **StepLR Scheduler:**
  - Reduces learning rate by a factor of γ after a fixed number of epochs (step size).
  - Allows optimization process to stabilize and prevents overshooting of minima.
  - Facilitates better convergence and avoids oscillations during training.

5.4 Results & Analysis

The Conditional GAN (Model-1) trained with both L1 loss and adversarial loss, along with schedulers for both generator and discriminator, achieved promising results, especially for only being trained for 2 epochs. We restricted number of epochs to 2 to be in terms with the training not being too heavy as the model architecture is complex. We generated dehazed images from hazy images in the val dataset using our trained model-1 and observed that with images with a realistic amount of haze, the model performed extremely well, generating the clean image nearly perfectly. However, for overly hazed images, it's generated images were distorted in those paritcular areas. The results are shown below.

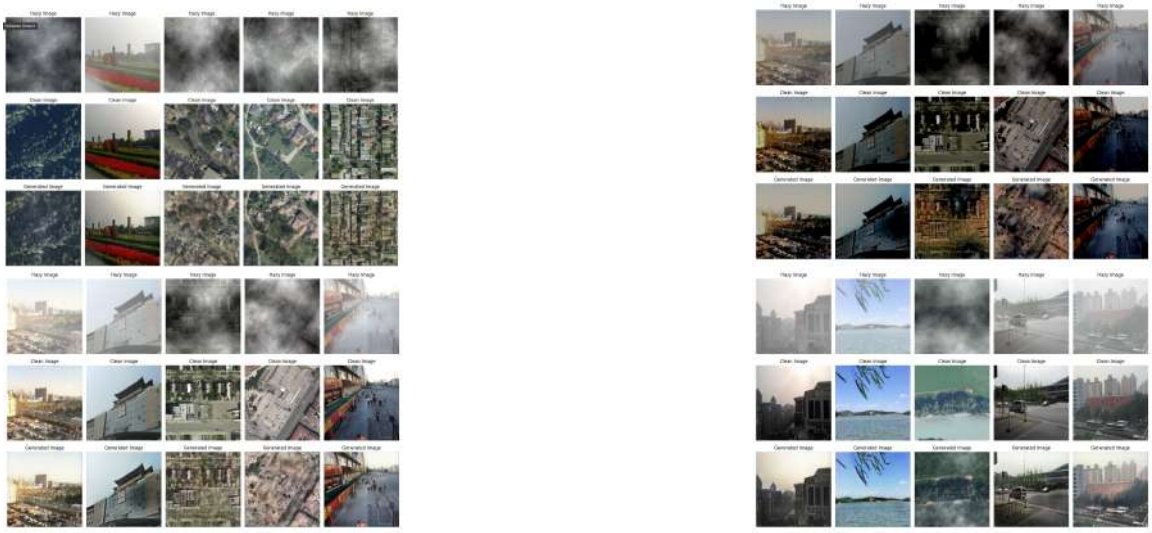


Figure 10: Results from conditional GAN with L1 loss

The mean PSNR and SSIM values were measured at 18.97 and 0.5441, respectively, indicating a high average quality and structural similarity between the generated dehazed images and the ground truth images. Similarly, the median PSNR and SSIM values of 18.95 and 0.5409, respectively, suggest a consistent performance across the dataset. Table 2 summarizes the quantitative evaluation metrics for Model-1.

	Mean	Median	Max
PSNR	18.97	18.95	26.592
SSIM	0.5441	0.5409	0.89

Table 2: Performance Metrics for Model 1 with All Losses on Validation Dataset

6 Experiment 2: Model 1 with Combination of L1 Loss, Adversarial Loss, L2 Loss, & Regularization Loss

In this experiment, we trained a conditional Generative Adversarial Network (cGAN) based on the Model 1 architecture. The network comprises an encoder-decoder generator and a discriminator. The generator aims to transform hazy images into clear images, while the discriminator learns to distinguish between real (clear) and generated (fake) images.

6.1 Setup

The setup for training Model 1 is as follows:

- **Generator:** The generator architecture features an encoder-decoder design, consisting of multiple encoder and decoder blocks. The encoder section downsamples the input hazy image and extracts hierarchical features, while the decoder section upsamples the features to reconstruct the dehazed image. Skip connections are incorporated to facilitate feature fusion between different levels of abstraction. Dropout layers are included for regularization.
- **Discriminator:** The discriminator processes pairs of input images (hazy and clear) through convolutional layers. It evaluates the realism of the images and provides feedback to the generator. The discriminator architecture involves an initial convolutional layer followed by a series of CNN blocks, each comprising convolutional layers, batch normalization, and LeakyReLU activation.
- **Loss Functions:**
  - **Adversarial Loss:** The adversarial loss ($\mathcal{L}_{\text{adv}}$) is computed using binary cross-entropy loss with logits (BCEWithLogitsLoss), measuring the probability that the generated images are real or fake.
  - **L2 Loss:** The L2 loss ($\mathcal{L}_{\text{per}}$) is calculated using the mean squared error (MSE) loss between pixel values of generated and ground truth images, promoting pixel-wise similarity.
  - **L1 Loss:** The L1 loss ($\mathcal{L}_{\text{L1}}$) measures the absolute pixel-wise difference between the generated and ground truth images, encouraging high-fidelity reconstruction.
  - **Regularization Loss:** The regularization loss ($\mathcal{L}_{\text{reg}}$) enforces smoothness in the generated images by penalizing sharp changes in pixel values. It is computed as the sum of absolute differences between adjacent pixels in the generated image.
- **Optimizers:** We used the Adam optimizer for both the generator and the discriminator. The learning rate was set to 0.0002, and the momentum parameters (β_1, β_2) were set to $(0.5, 0.999)$.
- **Learning Rate Schedulers:** Learning rate schedulers were employed for both the generator and the discriminator. The learning rate was decayed by a factor of γ every n epochs to facilitate convergence.

6.2 Loss Function

The total loss for the generator ($\mathcal{L}_{\text{G}}$) is computed as the weighted sum of individual losses:

$$\mathcal{L}_{\text{G}} = \lambda_{\text{adv}} \times \mathcal{L}_{\text{adv}} + \lambda_{\text{res}} \times \mathcal{L}_{\text{L1}} + \lambda_{\text{per}} \times \mathcal{L}_{\text{per}} + \lambda_{\text{reg}} \times \mathcal{L}_{\text{reg}}$$

where λ_{adv}, λ_{res}, λ_{per}, and λ_{reg} are hyperparameters controlling the contribution of each loss term.
The discriminator loss ($\mathcal{L}_{\text{D}}$) is calculated as the average of the binary cross-entropy losses for real and fake images:

$$\mathcal{L}_{\text{D}} = \frac{\mathcal{L}_{\text{D\_real}} + \mathcal{L}_{\text{D\_fake}}}{2}$$

where $\mathcal{L}_{\text{D\_real}}$ and $\mathcal{L}_{\text{D\_fake}}$ are the binary cross-entropy losses for real and fake images, respectively.

6.3 Results

The model was trained for 10 epochs and the below figures show some samples taken through the training process and samples after training was finished. The model is seen to clearly perform very well, especially on images with a normal amount of haze. However, it does struggle with recognising areas with a very high amount of haze and often pads such areas with grass, perhaps due to the large amount of data in the dataset with green pastures.

Figure 11: Results from conditional GAN with a combination of L1, Adversarial, L2 and Regularisation Loss on Validation Dataset

Analyzing the results reveals encouraging performance metrics, summarized in the table below:

	Mean	Median	Max
PSNR	21.53	21.59	31.31
SSIM	0.7069	0.7101	0.9475

Table 3: Performance Metrics for Model 2 on Validation Dataset

These metrics affirm the model's efficacy in image restoration, with PSNR values averaging at 21.53 and SSIM scores reaching 0.7978 on average. Notably, the model achieves a maximum PSNR of 31.31 and a maximum SSIM of 0.9547, underscoring its capability to faithfully reconstruct images while preserving their structural integrity and details. Despite encountering challenges in extreme hazy images, it performs very well on images with a medium or low amount of haze, as seen in the figures below.

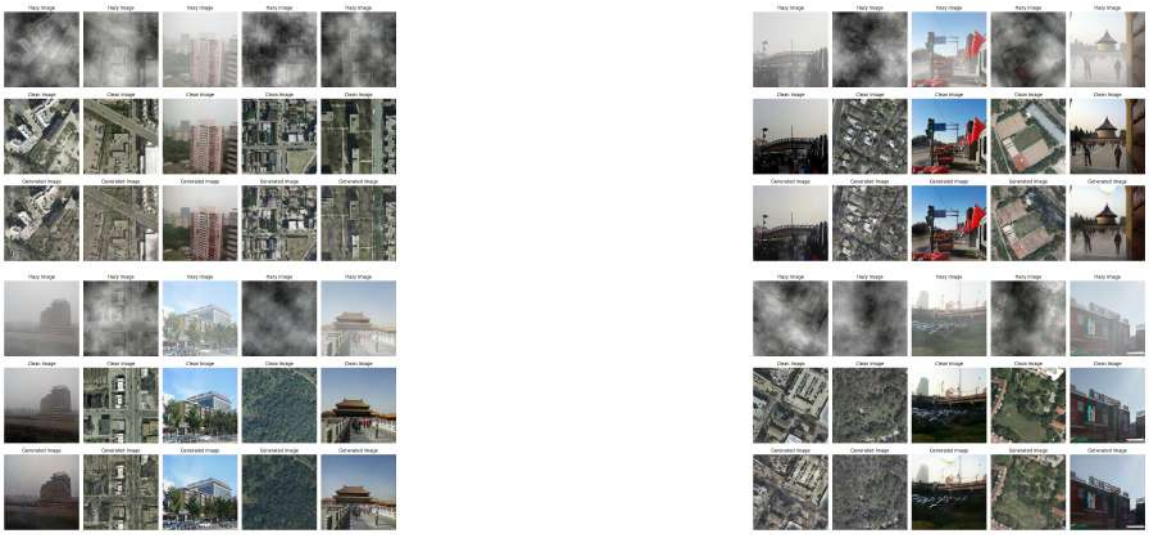


Figure 12: Results from our chosen model, experiment-2, Model-1 with a combination of L1, L2, Adversarial and Regularisation loss

We monitored the total generator loss and the discriminator loss throughout the training process, storing their values every batch and plotting the graph of epoch (divided based on number of batches) versus generator and discriminator loss every 20 batches. The final graph was 13 which illustrates that we were able to achieve a converging loss for the generator through the above combination of losses allowing stability to the GAN's generation, something which can't be achieved using just the min-max loss.

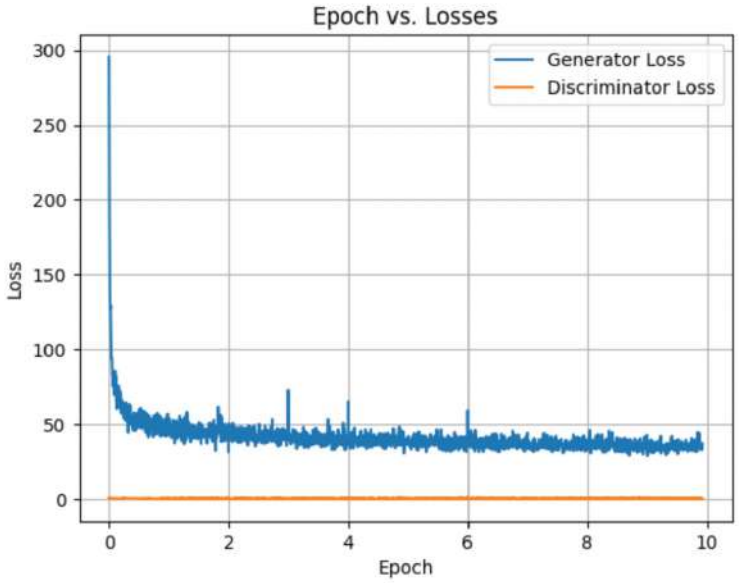


Figure 13: Generator Loss & Discriminator Loss versus Epoch (divided into batches) for Model1 using L1, L2, Regularisation and Min-Max Loss

7 Experiment 3: Model-2 with L1 Loss and Adversarial Loss

For this experiment, we adopt the same setup as Experiment 1, with the only variation being the utilization of Model-2 architecture. Model-2 employs a Multi-Scale UNet-like architecture, comprising encoder and decoder blocks for feature extraction and reconstruction, respectively. The discriminator network remains unchanged, processing pairs of hazy and ground truth images to distinguish between real and generated images.

The experiment employs the same loss functions, optimizers, and learning rate schedulers as Experiment 1, facilitating a direct comparison of the performance between Model-1 and Model-2 architectures under similar conditions.

Refer to Experiment 1 for detailed descriptions of the setup, loss functions, optimizers, and learning rate schedulers.

7.1 Results

The results of the Conditional GAN (Model-2) experiment demonstrate promising performance in dehazing images, with a mean PSNR of 19.27 and a median PSNR of 19.20. Additionally, the mean SSIM value of 0.5725 and median SSIM value of 0.5628 indicate a high level of structural similarity between the generated and ground truth images. These metrics suggest that Model-2, equipped with both L1 loss and adversarial loss, effectively enhances image quality by reducing haze while preserving image details and structures. The slightly higher PSNR and SSIM values compared to Experiment 1 may indicate that the Multi-Scale UNet-like architecture of Model-2 better captures and reconstructs image features, leading to improved dehazing performance. Table **??** summarizes the quantitative evaluation metrics for Model-3.

	Mean	Median	Max
PSNR	19.27	19.20	27.32
SSIM	0.5725	0.5628	0.5779

Table 4: Performance Metrics for Model 2 on Validation Dataset

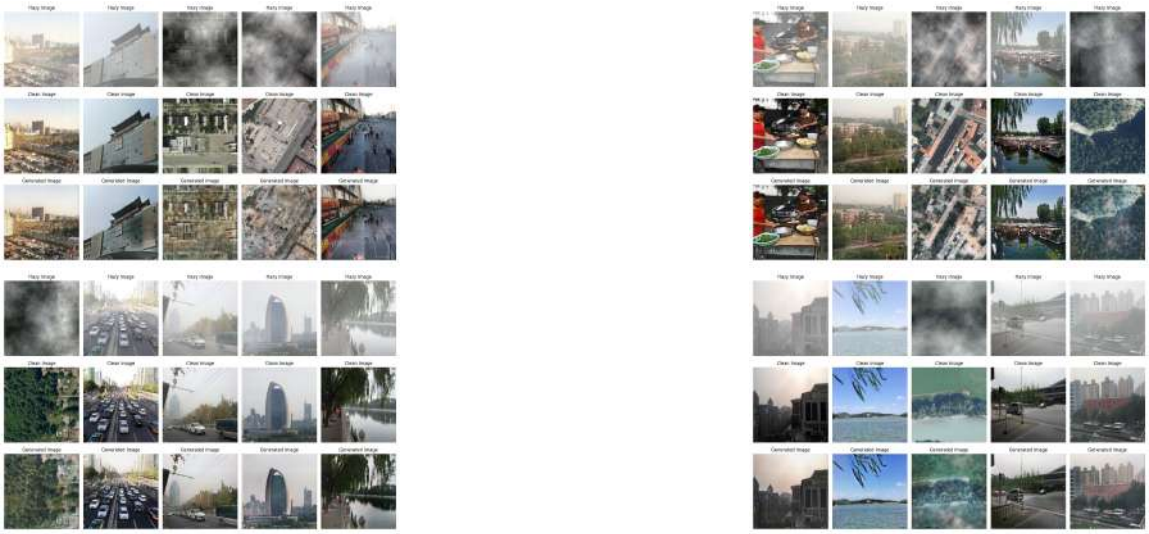


Figure 14: Results from conditional GAN with L1 loss on Validation Dataset

8 Experiment 4: Model-4 with L1 Loss and Adversarial Loss

For this experiment, we adopt the same setup as Experiment 1, with the only variation being the utilization of Model-4 architecture. Model-4 employs a simplified U-Net-like architecture, featuring down-sampling and up-sampling blocks for feature extraction and reconstruction, respectively. The discriminator network remains unchanged, processing pairs of hazy and ground truth images to distinguish between real and generated images.

The experiment employs the same loss functions, optimizers, and learning rate schedulers as Experiment 1, facilitating a direct comparison of the performance between Model-1 and Model-4 architectures under similar conditions.

Please refer to Experiment 1 for detailed descriptions of the setup, loss functions, optimizers, and learning rate schedulers.

8.1 Experiment - 4: Results & Analysis

Model 4, employing an Enhanced Multi-Scale Generator with PatchGAN Discriminator, exhibits moderate performance in reducing haze in images. With a mean PSNR of 16.09 and median PSNR of 15.90, along with a mean SSIM of 0.5493 and median SSIM of 0.5313 5, the model shows satisfactory but not exceptional results. Despite the U-Net inspired architecture, featuring encoding and decoding layers for feature extraction and refinement, Model 4 struggles with preserving fine details and textures consistently across different samples. Further optimization and refinement of the architecture, along with fine-tuning of training parameters, may be required to enhance its performance.

	Mean	Median	Max
PSNR	16.09	15.90	24.23
SSIM	0.5493	0.5313	0.9056

Table 5: Performance Metrics for Model 4 on Validation Dataset

The below figure 15 displays a subset of the results of model-4's dehazing on validation dataset.

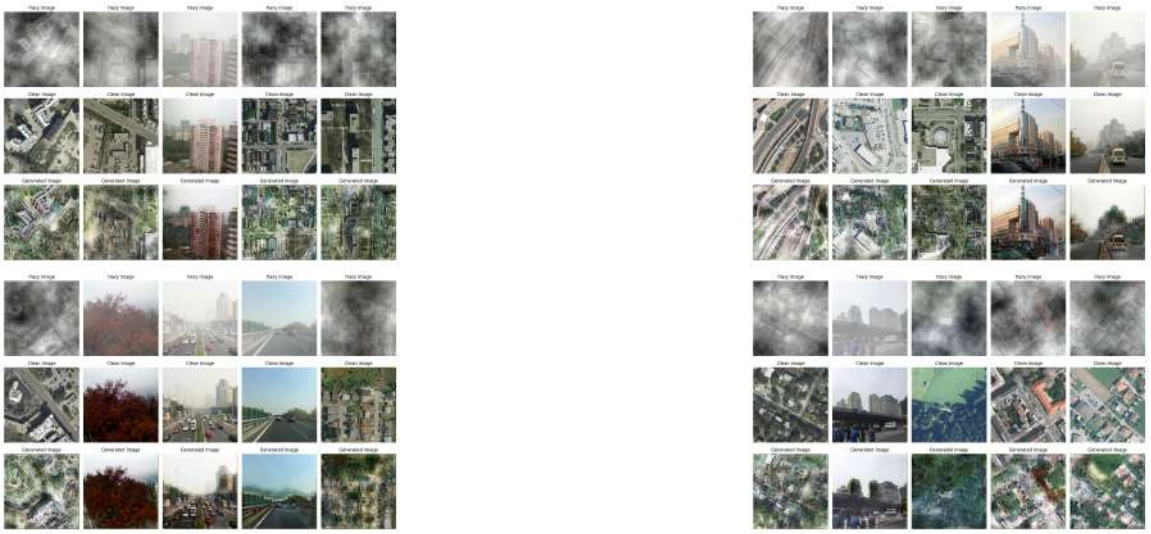


Figure 15: Results from Model-4 on validation dataset

9 Experiment 5: Model - 5 - Variation of U-Net Generator (Complex Version) with PatchGAN Discriminator, Adversarial Loss & L1 loss

This experiment has a setup identical to experiment-1. The discriminator loss is adversarial loss and the generator loss is the sum of adversarial loss and L1 loss (MAE) on fake images, real images. We used the step learning rate scheduler for both the generator and the discriminator. Moreover, we used the Adam's optimiser for both the generator and discriminator. The model was trained for 6 epochs.

9.1 Experiment - 5: Results & Analysis

The results obtained for Model 5, the "Complex Generator with UNet-like Architecture and PatchGAN discriminator" demonstrate promising performance in image dehazing. The mean PSNR (Peak Signal-to-Noise Ratio) achieved is 22.89 dB, indicating high fidelity in the reconstructed images compared to the ground truth clear images. The median PSNR of 23.10 dB suggests consistent quality across the dataset, with a maximum PSNR of 32.18 dB, indicating exceptional performance on certain images. Additionally, the mean SSIM (Structural Similarity Index) of 0.8625 reflects the preservation of structural information between the generated and ground truth images, while the median SSIM of 0.8782 reinforces the overall quality consistency. Below is a table summarizing the PSNR and SSIM statistics for Model 5:

	Mean	Median	Max
PSNR	22.89	23.10	32.18
SSIM	0.8840	0.8963	0.9615

Table 6: Performance Metrics for Model 5 on Validation Dataset

The below figures display a few of the generated images by model 5 from hazy images in the validation dataset,

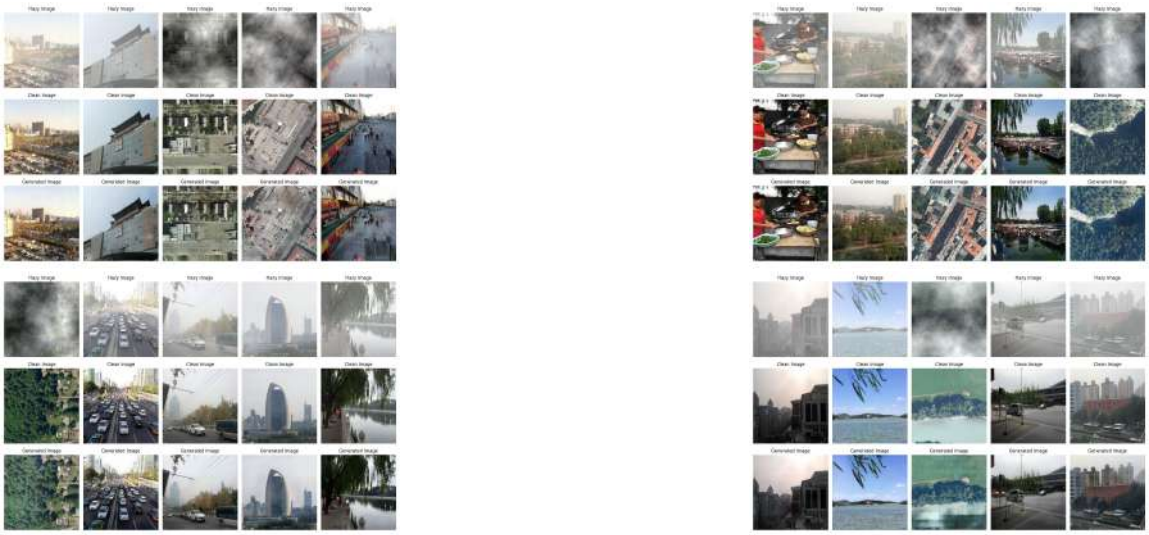


Figure 16: Results from Model-5 using a combination of L1loss and Adversarial loss

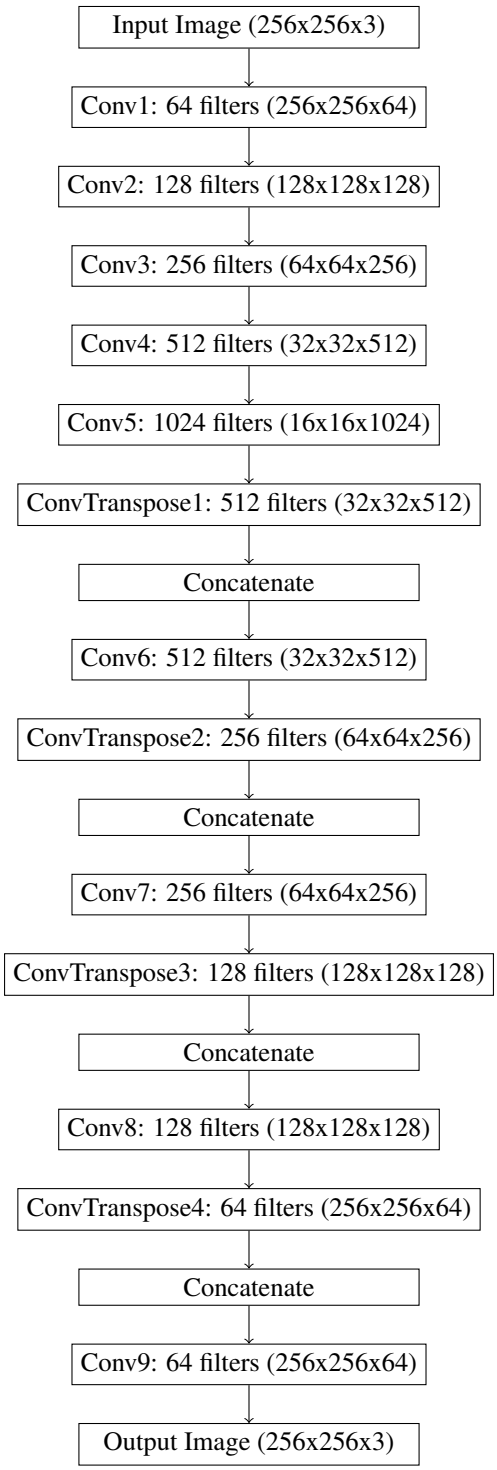


Figure 17: Flowchart of the Model 5 Generator(Complex U-like architecture)

10 Comparison With Non-Conditional GANs: Experiments with Model-3

We also trained and tested a non-conditional GAN architecture where the discriminator takes only the clean image as input. The generator and discriminator for this GAN were kept relatively simple as compared to the conditional GAN but we discovered a huge gap in performance. The model, model-3 was tested in the following setups.

1. Setup - 1
    - Loss Functions: Ony Min-Max Adversarial Loss
    - Schedulers: Step learning rate schedulers for both the generator and discriminator
2. Setup - 2
    - Loss Functions: Adversarial Loss & L1 Loss for the generator and only adversarial loss for the discriminator combined as $L_G = L_{Adv} + L_{L1}$, $L_D = L_{Adv}$
    - Schedulers: Step learning rate schedulers for both the generator and discriminator
3. Setup - 3
    - Loss Functions: Adversarial Loss & L1 Loss for the generator and only adversarial loss for the discriminator combined as $L_G = L_{Adv} + L_{L1}$, $L_D = L_{Adv}$
    - Schedulers: No learning rate schedulers
4. Setup - 4
    - Loss Functions: Wasserstein GAN loss and training based on [1]
    - Schedulers: Step learning rate schedulers for both the generator and discriminator

Based on the experimental results, we observe significant differences in the performance metrics among the three models. Model-1, a Conditional GAN with L1 loss and adversarial loss, achieves a mean PSNR of 21.53 and a mean SSIM of 0.6622. Model-2, another Conditional GAN with similar loss functions, underperforms compared to Model-1 slightly with a mean PSNR of 19.27 and a mean SSIM of 0.685. On the other hand, Model-3, a Normal GAN, shows lower performance with a mean PSNR of 12.44 and a mean SSIM of 0.3011 when schedulers are utilized. However, when schedulers are not employed, Model-3 exhibits a slight improvement in mean PSNR to 12.66, yet the mean SSIM decreases to 0.1586.

Model	**Loss Function**	**Mean PSNR**	**Median PSNR**	**Mean SSIM**	**Median SSIM**
Model 1	L1, L2, Adv., Reg.	21.53	21.59	0.7069	0.7101
Model 2	L1, Adv	19.27	19.20	0.5725	0.5628
Model 3	L1, Adv (W/schedulers)	12.44	12.45	0.1386	0.1231
Model 3	L1, Adv (WO/schedulers)	12.66	12.83	0.0421	0.0631
Model 4	L1, Adv.	16.09	15.90	0.5493	0.5313
Model 5	L1, Adv.	22.89	23.10	0.7916	0.8174

Table 7: Overall Comparison of Performance Metrics on Validation Dataset for Different Models

11 Experiment - 6: Conditional GAN with Wasserstein Loss

To find out the performance of conditional GANs with Wasserstein loss, we trained model-1 using the Wasserstein training process described in [1]. The model was trained for 4 epochs with RMSprop optimisers using a learning rate of 0.0002 and stepLR schedulers for the generator and discriminator. However, the results were not very promising and due to a lack of time we did not experiment on this further as we had already obtained good results with experiment-2 and experiment-5.

11.1 Experiment - 6: Results

The below figure displays a few results from this extra experiment. It depicts a huge performance gap with experiment-2 depicting the strength of the combined loss functions used in the experiment [2].

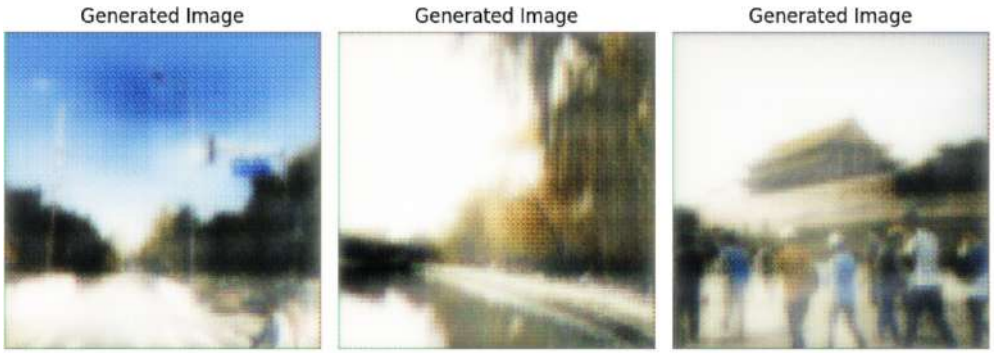


Figure 18: Generated Images from a CGAN with Wasserstein loss: depiciting a considerable performance gap with our chosen model (Experiment-2)

12 Conclusion

In our deep learning project focused on image dehazing, we conducted a thorough investigation into different models and loss functions to improve visibility and clarity in hazy images. Through extensive experimentation, we found that conditional generative adversarial networks (cGANs) consistently outperformed traditional GANs, showcasing the importance of incorporating contextual information into the dehazing process. Specifically, cGANs benefited from the additional guidance provided by hazy images, resulting in more realistic and visually appealing dehazed images. This underscores the effectiveness of leveraging conditional information to enhance the performance of generative models in image dehazing tasks.

Moreover, our exploration of various combinations of loss functions revealed that integrating adversarial loss, perceptual loss, and L1 loss yielded the most promising results. By combining these losses, we were able to effectively reduce haze while preserving crucial image details, resulting in significant improvements in quantitative metrics such as Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM). This multi-loss approach demonstrates the importance of leveraging diverse optimization objectives to address the complex challenges inherent in image dehazing. Overall, our findings contribute to advancing the state-of-the-art in image dehazing techniques and lay the groundwork for further research and applications in this domain

References

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223, 2017.

Runde Li, Jinshan Pan, Zechao Li, and Jinhui Tang. Single image dehazing via conditional generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8202–8211, 2018.

Satya Mallick. DeepLabV3: Ultimate Guide, 2024.

Aditya Mehta, Harsh Sinha, Murari Mandal, and Pratik Narang. Domain-aware unsupervised hyperspectral reconstruction for aerial image dehazing. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 413–422, 2021.

Zachary Mueller. Image Augmentation: Make it Rain, Make it Snow - How to Modify a Photo with Machine Learning, 2019.

Aquib Nisar. Pix2pix conditional gans. `https://github.com/AquibPy/Pix2Pix-Conditional-GANs`, 2021. 2024-04-17.