UNIVERSITÀ DI PISA

DEHAZING IMAGES USING GENERATIVE ADVERSARIAL NETWORKS

ARTIFICIAL INTELLIGENCE AND DATA ENGINEERING DII

SYMBOLIC AND EVOLUTIONARY ARTIFICIAL INTELLIGENCE

Di Donato Mattia

Giorgi Matteo

IN COLLABORATION WITH FLYSIGHT

Contents

1 Fog Generation Techniques **1**

1.1 Dataset problem 1

1.2 Atmospheric diffusion model 2

1.3 Synthetic dataset generation 3

2 Dataset **5**

2.1 M3FD 5

2.2 RESIDE SOTS 6

3 Pix2PixHD **7**

3.1 Introduction 7

3.2 Pix2PixHD structure 9

4 Models **11**

4.1 Single GAN model 11

4.1.1 G1 training 12

4.1.2 G1 and G2 training 12

4.2 Classifier and GANs 12

4.2.1 Classifier 13

4.2.2 Gans 14

5 Results **15**

5.1 Qualitative Evaluation 15
5.2 Classifier comparison 16
5.3 Model Comparison 17
5.3.1 Reference metrics 17
5.3.2 No-reference metrics 18
5.3.3 Model Comparison Indoor 18
5.3.4 Model Comparison Outdoor 18
5.4 Training and Execution times 19

6 Conclusions **21**

List of Figures

1.1 Atmospheric Scattering Model 2

1.2 Depth map 4

1.3 Original image 4

1.4 Low-fog 4

1.5 Medium-fog 4

1.6 High-fog 4

2.1 M3FD scene 6

2.2 Reside SOTS indoor scene 6

2.3 Reside SOTS outdoor scene 6

3.1 Network architecture of our generator. We first train a residual network G1 on lower resolution images. Then, another residual network G2 is appended to G1 and the two networks are trained jointly on high resolution images. Specifically, the input to the residual blocks in G2 is the element-wise sum of the feature map from G2 and the last feature map from G1. 9

4.1 G1 and G1+G2 Models 11

4.2 Classifier with GANs 13

5.1 Results 15

5.2 VGG16 and InceptionV3 results 16

Abstract

Haze significantly impairs the efficacy of machine vision systems. Its presence obscures scene visibility, thereby hindering various machine vision tasks, including object detection, target tracking, and semantic segmentation. Consequently, image dehazing plays a pivotal role in computer vision. By mitigating haze effects, noise is typically diminished, edges become clearer, and the signal-to-noise ratio improves, thereby enhancing the overall performance and resilience of machine vision systems. Given these factors, the demand for image dehazing spans across numerous fields, including security surveillance, autonomous driving, weather analysis, and geographic information systems.

With advancements in image processing technology, the approach to image dehazing is undergoing significant evolution. Initially, the problem of image dehazing was approached as one of image enhancement utilizing, for example, techniques such as contrast enhancement. Subsequently, researchers turned to image restoration methods grounded in the atmospheric scattering model to address the challenge of image dehazing. Given the complexity of the image dehazing task, researchers have increasingly integrated multiple methods to enhance performance and in the recent years, propelled by the rapid advancements in deep learning technology, a surge in deep learning-based dehazing algorithms has been observed. Compared to conventional methods, these deep learning-based approaches have demonstrated significant improvements in dehazing effectiveness and robustness.

However, haze removal encounters numerous challenges. One of the major obstacles in image dehazing research lies in the difficulty of acquiring paired clear and blurred images. Therefore, continuous improvement of image defogging datasets is critical for defogging research. In fact, during the initial stages, much of the research on image dehazing was based on single hazy images. Without the availability of clean images coupled with hazy images, objectively evaluating the accuracy of dehazing algorithms was also a considerable challenge.

CHAPTER 1

Fog Generation Techniques

1.1 Dataset problem

In effectively tackling the problem of fog removal to date, one of the best ways is dictated by the use of Neural Networks. To effectively train Neural Networks and obtain satisfactory results, it is essential to use large and high-quality datasets, both in terms of resolution and in terms of scenario characteristics. Different types of datasets can be used in this context.

Real datasets include scenes captured in different weather conditions, providing the benefit of lifelike images. However, a notable drawback is the lack of correspondence between images with and without fog due to atmospheric differences, not only in terms of fog but also in factors such as lighting and contrast, as these photos are taken at different times. Furthermore, obtaining a large and diverse set of real images can be difficult.

Alternatively, **artificial datasets** generated with the use of fog machines can be used. While these datasets are typically smaller, they offer an advantage in terms of image matching. Each scene includes images with and without fog, maintaining constant atmospheric characteristics. However, the downside is the loss of realism in simulated fog.

Finally, **synthetic datasets** created by manipulating images provide a compelling solution. This technique allows the generation of a large dataset with matching between images with and without fog. However, the downside is that the generated fog may not replicate real-world fog conditions, as it is produced through image manipulation techniques such as atmospheric dispersion modeling.

We decided to use synthetic datasets that allow us to obtain large amounts of data to train our

Neural Networks. This decision is based on the ability to generate a significant number of images with and without fog, ensuring that our Neural Network can clearly learn the fog removal process.

1.2 Atmospheric diffusion model

The Atmospheric Scattering Model (ATSM) has long been the cornerstone of image dehazing techniques. Typically, images captured outdoors often suffer from poor visibility, reduced contrast and color variations due to the presence of haze. This atmospheric phenomenon, caused by aerosols such as dust, fog and smoke, introduces complicated and non-linear noise, posing a formidable challenge for image restoration and enhancement. Furthermore, when light passes through the atmosphere, both scattering and absorption occur. This process is wavelength dependent, as shorter wavelengths (e.g. blue light) are scattered more strongly than longer wavelengths (e.g. red light). As a result, hazy scenes exhibit a color shift towards shorter wavelengths, often manifesting as bluish tones.

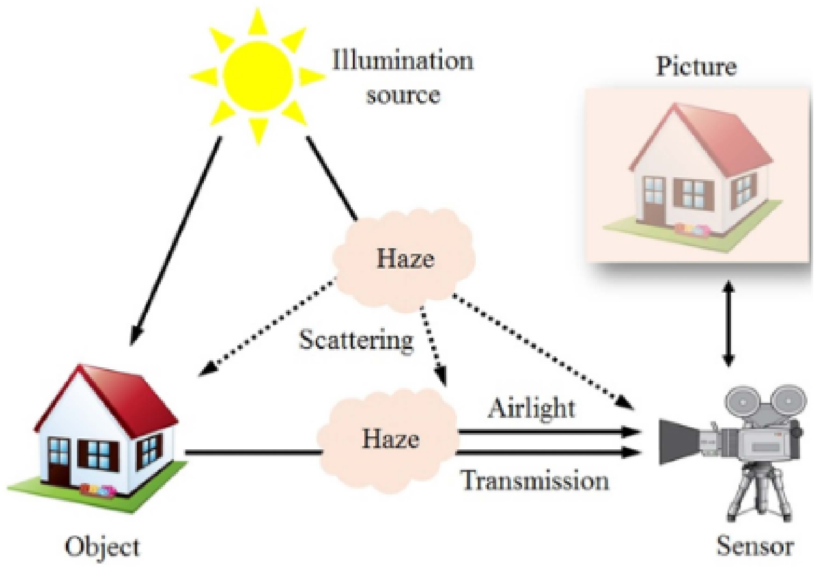

Figure 1.1: Atmospheric Scattering Model

Atmospheric Scattering Model provides a theoretical framework for understanding the generation of hazy images. According to this model, the observed hazy image I is the result of the haze-free scene radiance J attenuated by the transmission matrix t and added with the global atmospheric light A as showed in the following formula.

$$I(x,y) = J(x,y) \cdot t(x,y) + A \cdot (1 - t(x,y))$$

where $I(x,y)$ is the hazed scene, $J(x,y)$ is the original scene, A is atmospheric light and the transmission matrix is $t(x,y) = e^{-\beta \cdot d(x,y)}$.

Examining the formula, if $A = 0$, *airlight scatter*, the portion on the right disappears, leaving only the *attenuation* part on the left. This is solely controlled by the transmission map, thus influenced by the β parameter and depth maps. Conversely, if $\beta = 0$, there is no scattering effect, resulting in the original image.

State-of-the-art dehazing methods leverage the Atmospheric Scattering Model to estimate critical parameters such as the atmospheric light *A* and the transmission matrix *t(x,y)*. These parameters, often estimated through physically grounded or data-driven approaches, are essential for accurately restoring clear and high-quality images from hazy observations.

1.3 Synthetic dataset generation

The objective we aim to achieve is to use this model not merely for directly removing fog from images but rather for creating a synthetic dataset. In this process, we will start with haze-free images to generate images with added fog. To accomplish this task, it is crucial to employ *depth maps* that enable the creation of a transmission map, furthermore is important to estimate the values of *A* (atmospheric light) and β as accurately as possible, ensuring the generation of synthetic fog effects that closely resemble real-world conditions.

For each image in the original dataset, we employed corresponding depth maps, black and white images where pixel values denote subject distances, with values close to 0 indicating near subjects and values near 255 representing distant subjects. These depth maps facilitated the computation of the transmission map in the scattering formula.

Manipulating the parameters A and β, we produced different level of haze. A represents the image's brightness, ranging from 0 to 255, the minimum and maximum pixel values. On the other hand, β controls atmospheric light scattering, essentially determining the dispersion of light, higher values result in a more pronounced and dense fog effect.

In our case, we used A values of $\{120,180,220\}$, along with corresponding β values of

$\{0.004,0.006,0.008\}$. This yielded three types of images: low, medium, and high levels of fog.

Figure 1.2: Depth map

Figure 1.3: Original image

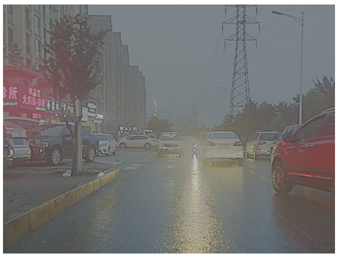

Figure 1.4: Low-fog

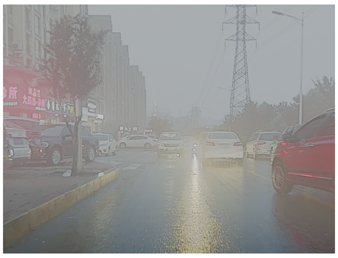

Figure 1.5: Medium-fog

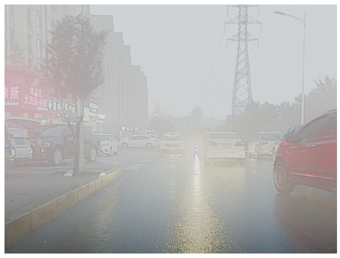

Figure 1.6: High-fog

CHAPTER 2

Dataset

In the previous chapter, fundamental concepts related to the challenge of removing fog from images were discussed. Among these, one of the main issues lies in the limited availability of image pairs containing both foggy and fog-free scenes, representing a wide range of contexts. These image pairs are crucial for the training phase of the models. In this regard our work has been divided into several phases, some of which involved the use of specific datasets. Initially, to generate a large and synthetic dataset used to train our model, we worked with the M3FD dataset. Subsequently, in the final phase, we evaluated the model generated using the RESIDE SOTS dataset.

2.1 M3FD

The M3FD dataset consists of 4200 images extracted from frames and videos of various outdoor environments without fog. Each image is accompanied by a corresponding depthmap. We used this dataset to create a new set of data, completing the scenes with a fog filter to pair them with the corresponding fog-free images. This was possible by utilizing information about the depth of the scenes in the depthmap and using an atmospheric scattering model. In this way, we provided the GAN model with a large and complete dataset that includes both foggy and fog-free images.

Figure 2.1: M3FD scene

2.2 RESIDE SOTS

RESIDE is a comprehensive benchmark for single image dehazing, comprising a large set of synthetic data for training and two sets designed for objective and subjective quality evaluations. The RESIDE SOTS (Synthetic Objective Testing Set) is a part of this dataset. It includes 1000 synthetic images of indoor and outdoor environments, generated using the same process as the training data. We used this dataset to evaluate the performance of our model on a dataset widely recognized as a benchmark for comparing the performance of various dehazing algorithms.

Figure 2.2: Reside SOTS indoor scene

Figure 2.3: Reside SOTS outdoor scene

CHAPTER 3

Pix2PixHD

3.1 Introduction

Over the years, rendering photo-realistic images using traditional graphics techniques has posed significant challenges. These challenges stem from the intricate simulation required for geometry, materials, and lighting, which often result in time-consuming and expensive processes for building and editing virtual environments. Despite the advancements in graphics algorithms, creating and manipulating realistic images remains cumbersome, as every aspect of the world must be explicitly modeled.

Recognizing these limitations, researchers have sought alternative approaches to generating high-resolution images with realistic textures. One such innovative solution is the utilization of Generative Adversarial Networks (GANs). GANs aim to model the natural distribution of images by ensuring that generated samples are indistinguishable from real images. They have enabled a wide number of applications, including image generation, representation learning, image manipulation, and object detection.

The approach to fog removal using a Generative Adversarial Network (**GAN**) is based on optimizing the generator to produce a fog-free image from a foggy input. This process involves the discriminator, a network that compares the generated fog-free image with the real one, to assess whether satisfactory fog removal results has been achieved.

Specifically:

- **The generator:** It is a neural network that takes a foggy image as input and aims to generate a corresponding fog-free image. During the training process, the generator learns to identify the main characteristics of fog in order to effectively remove them from the image. A possible loss function for this network could be the mean squared error:

$$L_{\text{MSE}}(G) = \frac{1}{N}\sum_{i=1}^{N}(I_{\text{target}} - G(I_{\text{foggy}}))^2$$

I_{target} represents the target image without fog.
$G(I_{foggy})$ is the image generated by the generator given foggy input.
N is the total number of samples.

- **The discriminator:** It is a network that receives the fog-free image generated by the generator as input and compares it with real, fog-free images. Its task is to distinguish between synthetically generated images and real ones. A possible loss function for this network could be the binary cross-entropy:

$$L_{\text{BCE}}(D) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_{\text{real}} \cdot \log(D(I_{\text{target}})) + (1 - y_{\text{real}}) \cdot \log(D(G(I_{\text{foggy}})))\right]$$

y_{real} is the real label: (1) for target images and (0) for generated images.
$D(I_{target})$ is the output of the discriminator for a fog-free image.
$D(G(I_{foggy}))$ is the output of the discriminator for a generated image.

- **Adversarial Training:** The generator and the discriminator are trained adversarially, where the generator tries to produce synthetic fog-free images that are as similar as possible to real images, while the discriminator seeks to increasingly distinguish between synthetic and real images.

- **The ultimate goal of the training process** is to ensure that the generator produces synthetic fog-free images that are so realistic that the discriminator cannot distinguish between synthetic and real images.

However, traditional methods, such as GANs have faced challenges in generating high-resolution images due to training instability. To address this issue, recent advancements have introduced novel techniques, such as robust adversarial learning objectives and multi-scale architectures for generators and discriminators. These enhancements have significantly stabilized GAN training on high-resolution images, resulting in superior image quality and more realistic textures compared to previous approaches.

3.2 Pix2PixHD structure

Building upon the challenges addressed in traditional graphics techniques and the advancements made in Generative Adversarial Networks (GANs), Nvidia introduced the Pix2PixHD framework, a sophisticated solution tailored for image-to-image translation tasks. Unlike its predecessor, Pix2Pix, which faced limitations in generating high-resolution images, Pix2PixHD offers significant enhancements to both photorealism and resolution.

At its core, Pix2PixHD maintains the conditional GAN architecture, comprising a generator (G) and a discriminator (D). The generator's objective is to translate semantic label maps into realistic images, while the discriminator distinguishes between real and synthesized images, operating within a supervised setting. However, unlike the original Pix2Pix framework, Pix2PixHD employs a more advanced approach to address the challenges of generating high-resolution images. The key innovations introduced in Pix2PixHD include:

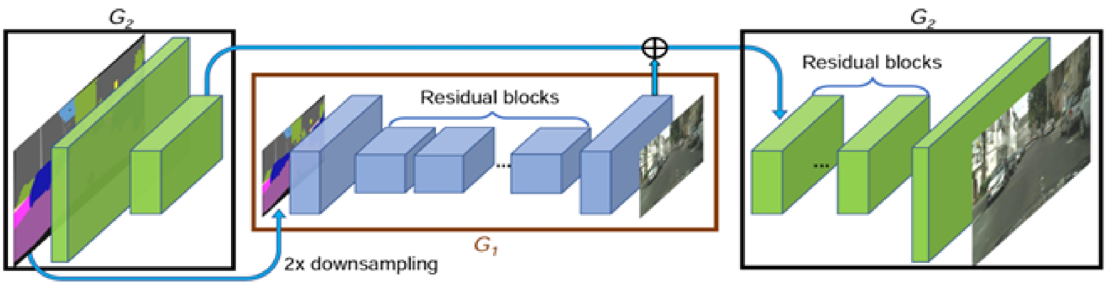

Figure 3.1: Network architecture of our generator. We first train a residual network G1 on lower resolution images. Then, another residual network G2 is appended to G1 and the two networks are trained jointly on high resolution images. Specifically, the input to the residual blocks in G2 is the element-wise sum of the feature map from G2 and the last feature map from G1.

1. **Coarse-to-fine generator**: One notable improvement is the division of the generator into two sub-networks: a global generator (G1) and a local enhancer (G2). The global generator operates at a lower resolution, while the local enhancer refines the output at a higher resolution. By sequentially passing the semantic label maps through these networks, Pix2PixHD can synthesize images with finer details.

2. **Multi-scale discriminators**: To effectively handle high-resolution images, Pix2PixHD utilizes three discriminators operating at different scales. Each discriminator is trained to differentiate between real and synthesized images at its respective scale. This multi-scale approach enables the model to capture both global context and fine details, enhancing overall image quality. We will refer to these discriminators as D1, D2, and D3. Specifically, we downsample both

the real and synthesized high-resolution images by factors of 2 and 4, respectively, to create an image pyramid of 3 scales. The discriminators D1, D2, and D3 are then trained to differentiate real and synthesized images at the 3 different scales, respectively. Although the discriminators have an identical architecture.

3. **Improved adversarial loss**: Pix2PixHD elevates the GAN loss function by integrating a feature matching loss grounded in the intermediate representations of the discriminator. This mechanism prompts the generator to generate images possessing natural statistics across various scales, thereby fostering stability in the training process and amplifying the overall quality of synthesized images. By incorporating this feature matching loss, the training becomes more stable, compelling the generator to produce images with natural statistics across multiple scales. This is achieved by extracting features from various layers of the discriminator and aligning these intermediate representations between real and synthesized images.

These advancements substantially enhance the efficacy of the Pix2Pix framework, empowering it to synthesize high-resolution images imbued with heightened photorealism and intricate details.

CHAPTER 4

Models

Two distinct models have been developed for image dehazing: a GAN model and a more complex model that incorporates a fog classifier along with three GANs. For both models, the *M3FD* dataset was used for training, while the *RESIDE SOTS* dataset was used for evaluation. These models require powerful GPUs and high RAM, and indeed, the *Colab Pro* platform was utilized for training the GANs.

4.1 Single GAN model

One-third of the dataset was used for training by randomly sampling 4200 images, while maintaining balanced classes. At inference time, the generator part of the GAN is utilized by provid-

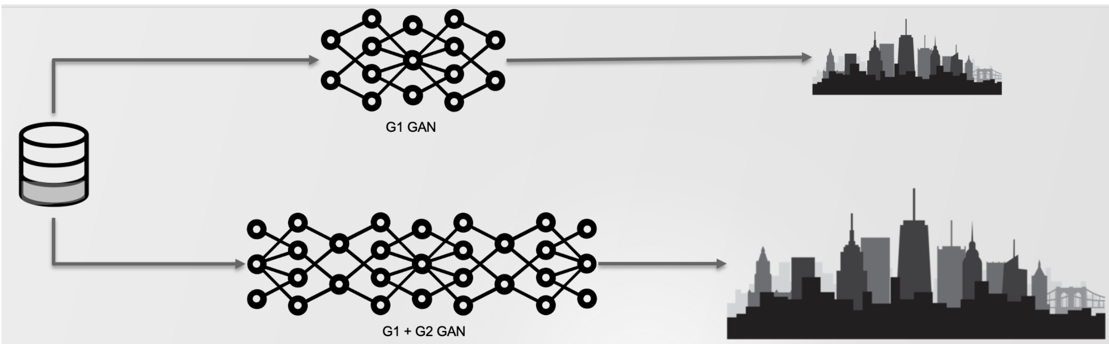

Figure 4.1: G1 and G1+G2 Models

ing an image with haze as input and obtaining a synthetic image without haze as output.

4.1.1 G1 training

The training of Pix2PixHD was carried out using images of size 512 pixels and a batch size of 4, continuing up to the 100th epoch, taking an average of 40 minutes for each epoch.

```
#training on low resolution(512) only G1 (global net)
!python train.py --continue_train --label_nc 0 --no_instance
    --name nebbia --dataroot ./datasets/nebbia --
   resize_or_crop crop --fineSize 512 --batchSize 4
```

4.1.2 G1 and G2 training

Starting from the model trained only with the G1, we continued the training by introducing the G2. This allows us to obtain higher-resolution images, thus more defined. In this case, we used images with a size of 1024 pixels and a batch size of 2. The training of this network was only possible through the use of the *A100 GPU* machine model, which has 40 GB of dedicated memory. The training of this model continued for 10 epochs, after which the Colab resources were revoked, and the model saving did not occur correctly, rendering it unusable.

```
#training on high resolution (1024) G1 and G2 (local+global)
    starting from epoch 100 of G1
!python train.py --netG local --continue_train --
   load_pretrain checkpoints/nebbia/ --label_nc 0 --
   no_instance --name nebbia --dataroot ./datasets/nebbia --
   resize_or_crop crop --fineSize 1024 --batchSize 2
```

4.2 Classifier and GANs

At inference time, we first classify the image into 4 categories, and subsequently, the GAN associated with the specific fog level is applied to the input image to generate a more accurate haze-free image.

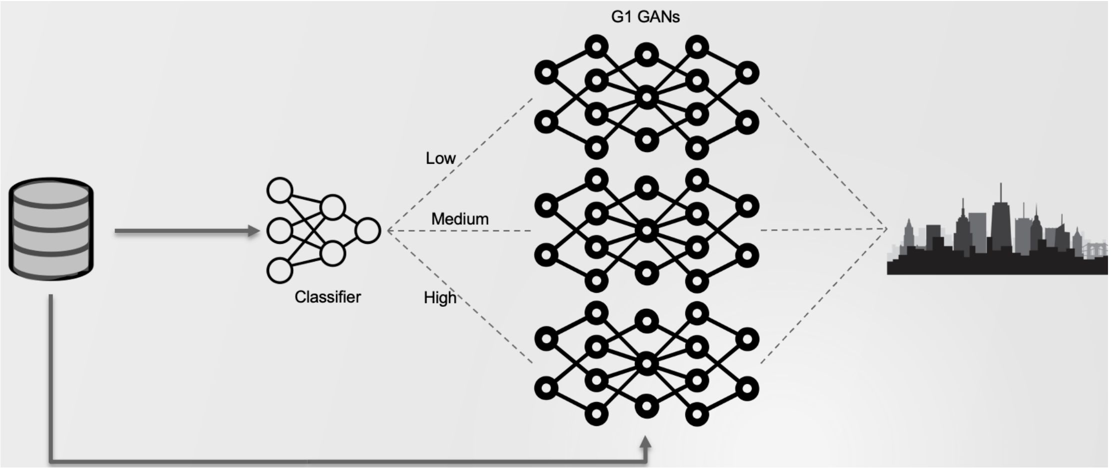

Figure 4.2: Classifier with GANs

4.2.1 Classifier

We used two pre-trained models (VGG16 and InceptionV3). Using the transfer learning technique, we loaded the model weights, set them as non-trainable, and added layers at the end of the network to adapt it to our classification task. The task involved classifying images of haze into 4 classes: no-fog, low-fog, medium-fog, and high-fog.

These classifiers were trained on the entire dataset (16,800 images), partitioned into a 60% training set, 20% validation set, and 20% test set. The input image size was set to 512 pixels, with a batch size of 16.

VGG16 model structure

```
inputs = keras.Input(shape=(image_size, image_size, 3))
x = base_model(inputs)
x = layers.Flatten()(x)
x = layers.Dense(512, activation="relu")(x)
x = layers.Dropout(0.3)(x)
x = layers.Dense(256, activation="relu")(x)
x = layers.Dropout(0.5)(x)
outputs = layers.Dense(4, activation="softmax")(x) # 4
    classi
model = keras.Model(inputs=inputs, outputs=outputs)
# freeze the base model
```

```
for layer in base_model.layers:
    layer.trainable = False
```

InceptionV3 model structure

```
model = Sequential()
model.add(base_model)
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(4, activation='softmax'))
# freeze the layers of the pretrained model
for layer in base_model.layers:
    layer.trainable = False
```

4.2.2 Gans

For each fog level, except for no-fog, a GAN was created by training only G1 up to the 50th epoch. Then the entire dataset was used for training with an input size of 512 pixels and a batch of 4.

CHAPTER 5

Results

5.1 Qualitative Evaluation

The results obtained can be evaluated qualitatively and from this perspective, they have been more than satisfactory. In Figure 5.1, it is evident how we managed to achieve a good-quality output from an image affected by a considerable level of fog. Comparing the output with the original image, it can be noted that using the general GAN model results in greater sharpness in the colors and edges of the various objects in the scene. This result is further improved using the Specialized GAN model, which achieves an even higher image quality. However, there re-

Figure 5.1: Results

mains a noticeable difference compared to the original, mainly due to the limited computational resources available during the training phase. Nevertheless, it still represents a significant result, which suggests further improvements in the case of greater computational resources.

5.2 Classifier comparison

We conducted more technical tests for the selected classifiers, which are the convolutional neural networks VGG16 and InceptionV3. We used the test portion of the M3FD dataset and extracted these confusion matrices as shown in Figure 4.3. The results obtained are excellent for both networks, as evidenced by an almost perfect diagonal in the matrix. However, we performed

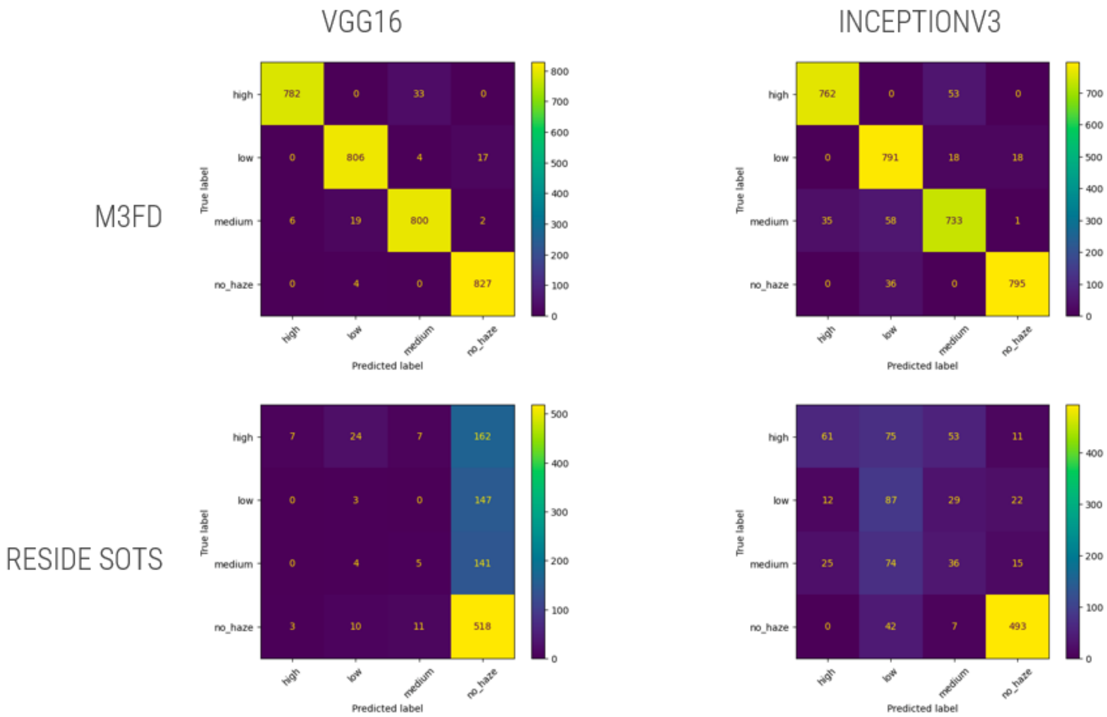

Figure 5.2: VGG16 and InceptionV3 results

the same tests using the RESIDE SOTS dataset, and unfortunately, the results were poor. This is probably due to the initial choice of parameters, which were too different from those used to generate the RESIDE synthetic dataset. A closer analysis of the two confusion matrices reveals that all images in the dataset were classified as "no fog", confirming that our selection of parameters A and β, which were much different and higher than those used to generate the RESIDE SOTS, negatively affected the results, distancing them from optimality.

5.3 Model Comparison

To compare the results obtained by various models, we considered several metrics for image quality evaluation. Specifically, we utilized both reference metrics, which compare the original image to the generated one, and no-reference metrics, which are only applied to the generated image.

5.3.1 Reference metrics

1. MSE (Mean Squared Error) measures the average discrepancy between the original and generated images. It calculates the squared difference between corresponding pixel values and computes the mean of these values. A lower MSE indicates better image similarity, but it doesn't consider human perception of the image.

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(I_{\text{original}}(i) - I_{\text{generated}}(i))^2$$

   $I_{\text{original}}(i)$: Pixel value of the original image at position i.
   $I_{\text{generated}}(i)$: Pixel value of the generated image at position i.
   N: Total number of pixels in the images.

2. SSIM (Structural Similarity Index) evaluates image similarity considering luminance, contrast, and structure. It measures differences in luminance, contrast, and local structure between images. SSIM returns a value between -1 and 1, where 1 indicates perfect similarity, 0 means complete dissimilarity, and positive values below 1 indicate some degree of similarity. SSIM is widely used as it accounts for human perception.

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

   x, y: Local windows of the two images for comparison of the same size.
   μ_x, μ_y: Mean intensity of x and y.
   σ_{xy}: Covariance of x and y.
   σ_x, σ_y: Variance of x and y.
   C_1, C_2: Small constants to stabilize the division.

3. PSNR (Peak Signal-to-Noise Ratio) assesses image quality in terms of error compared to the original. It calculates the ratio of the maximum signal peak to the Mean Squared Error. PSNR is expressed in decibels (dB), where higher values (typically above 30 dB)

indicate good image quality with few discrepancies from the original, and lower values suggest significant differences.

$$\text{PSNR} = 20 \cdot \log_{10}\left(\frac{\text{MAX}}{\sqrt{\text{MSE}}}\right)$$

MAX: Maximum possible pixel value (e.g., 255 for 8-bit RGB images).

5.3.2 No-reference metrics

1. PIQE (Perceptual Image Quality Evaluator) estimates perceived image quality without prior knowledge of reference images. It considers features like contrast, brightness, sharpness, and visual artifacts to generate a numerical score reflecting perceived image quality.

2. NIQE (Natural Image Quality Evaluator) assesses perceived quality of natural images without using a reference image. Based on statistics of natural images, NIQE focuses on how the image differs from statistical characteristics of considered natural images.

5.3.3 Model Comparison Indoor

	MSE	**SSIM**	**PSNR**	**PIQE**	**NIQE**
GAN 50	104.7894	0.7211	15.9983	27.8453	15.5920
GAN 100	105.0702	0.7416	16.5626	29.8438	15.9985
Classifier + 3GAN	106.4946	0.7053	15.3621	33.5978	14.9868

Table 5.1: Indoor Comparison

5.3.4 Model Comparison Outdoor

	MSE	**SSIM**	**PSNR**	**PIQE**	**NIQE**
GAN 50	104.7441	0.7536	17.2679	14.4238	14.8355
GAN 100	101.7645	0.7848	18.5019	12.8818	14.7754
Classifier + 3GAN	102.7947	0.7613	16.5595	10.6168	13.0812

Table 5.2: Outdoor Comparison

From the emerged results, we can conclude that:

- Low PSNR values indicate poor image quality, likely attributed to image resizing.
- The model based on the 3 GANs generates more natural images according to PIQE and NIQE metrics, benefiting from training on the entire dataset.
- The highest SSIM ratings are achieved with the extensively trained model.

5.4 Training and Execution times

Training Phase:

Model	**Time (s)**
VGG16	390s/epoch $\times$ 29 epochs = 189 minutes
InceptionV3	335s/epoch $\times$ 16 epochs = 90 minutes
GAN100	40min/epoch $\times$ 100 epochs = 67 hours
3GAN50	40min/epoch $\times$ 50 epochs $\times$ 3 classes = 100 hours

Table 5.3: Execution times for the training phase

Inference Phase:

Task	**Time (s)**
Load classifier	9s
Classify image	45ms
Load GAN	14s
Evaluate single image	1.26s

Table 5.4: Execution times for the inference phase

As previously discussed in the Models chapter, the training times of GANs are exceedingly high, and they also require a powerful GPU. Due to the absence of a dedicated powerful GPU, we relied on Colab Pro, which, despite being a paid service, still has limitations on usage in terms of time, after which resources are not immediately reassigned. Regarding inference times, they naturally depend on the hardware on which they are executed. Beyond the loading times of GANs and classifiers, performed only once, the inference time for a single image in the case of a single GAN is 1.26 seconds, while in the case of a classifier + GAN, the inference time is slightly higher. However, this remains extremely high for critical applications as outlined in the abstract, such as autonomous driving or visual support for vehicles. It is imperative to implement a more efficient and low-level version tailored to the hardware used in the vehicle.

CHAPTER 6

Conclusions

To further enhance the performance of our model, we have identified several potential areas for improvement.

- A critical aspect of the project is the generation of the synthetic dataset, particularly the selection of parameters A and β to achieve more realistic fog effects while simultaneously improving performance on the RESIDE SOTS test dataset for both the GAN and classifiers.
- It is important to consider that these images in the M3FD dataset are frames extracted from video clips, so it might be necessary to control the randomization of the training set selection.
- Regarding the training process, we might reconsider the option of training the model using the g1 method first and then the g1 + g2 method to assess whether this leads to significant performance improvements.

Integrating these strategies could contribute to enhancing our model and addressing the challenges identified during the evaluation phase.

For further questions or information you can also access the project repository on GitHub: https://github.com/mattiadido95/Haze-Fog-suppression.git.