School of Computing

FACULTY OF ENGINEERING AND
PHYSICAL SCIENCES

UNIVERSITY OF LEEDS

Final Report

Single Image Dehazing Software Using U-Net Architecture

Sandra Ana Maria Guran

Submitted in accordance with the requirements for the degree of
BSc Computer Science

2023/24

COMP3931 Individual Project

The candidate confirms that the following have been submitted*:*

Items	**Format**	**Recipient(s) and Date**
Final Report	*PDF file*	*Uploaded to Minerva (08/05/24)*
Link to online code repository	*URL*	*Sent to supervisor and assessor (08/05/24)*

The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.

(Signature of student) Sandra Guran

Summary

Dehazing, or defogging, is the process of transforming hazy/foggy images or videos by removing fine atmospheric particles like dust, smoke, and haze. Such software is crucial for enhancing visibility in applications like autonomous driving and surveillance systems, allowing for better image segmentation and object detection. These methods are categorised under computer vision techniques.

This report outlines the necessity for image dehazing, reviews older methods, and discusses new approaches utilising neural networks, particularly encoder-decoder architectures. The developed software employs an encoder-decoder method based on the U-Net architecture. Throughout the project, the U-Net structure is enhanced with various improvements. These enhancements include additional convolutional layers, ranging from 64 to 1024 filters, as well as Batch Normalisation, Kernel Initialization, Squeeze and Excitation blocks, adding Bottleneck and Dropout layers to prevent overfitting. Throughout testing, three different activation functions are evaluated, LeakyReLU, BeReLU and ReLU, with the last one ultimately being chosen for the final model. Additionally, four loss architectures are tested, with the best performing one being a combination of perceptual loss using VGG19 architecture and a mean squared error (MSE) loss function. Output layer activation functions, such as tanh and sigmoid, are also tested, with the latter being selected for the final model.

Efficiency evaluation of the dehazing algorithm incorporates both visual parameters and technical metrics. Peak Signal-to-Noise Ratio (PSNR) measures pixel-wise accuracy, while the Structural Similarity Index Measure (SSIM) assesses similarity to the ground truth image on a scale from 0 to 1, with 1 representing a perfect match. These parameters are assessed and compared for each enhancement incorporated into the encoder-decoder method.

The model undergoes training on two distinct datasets: Foggy Cityscapes [32], which comprises street images, and Indoors RESIDE [34], containing images with lighter fog and more defined objects. Each dataset includes clear images, corresponding ground truth images, and a series of three to ten variations of hazy images with different fog intensities. By pairing each clear image with its hazy counterpart, the model learns to transform hazy images into clear ones. The results are then compared with those obtained from other prominent software and encoder-decoder projects, showcasing the efficiency and enhancements achieved throughout the process.

Acknowledgements

I would like to thank my supervisor, Dr. Duygu Sarikaya, for guiding and helping me throughout the project. She has supported me with great advice and shared her extensive expertise in computer vision.

I would also like to thank my assessor, Professor Zheng Wang, for his valuable feedback during the progress meetings, as well as his insightful questions and tips on writing an effective and impressive report.

Finally, I want to thank my family and friends for their emotional support and life advice, which were crucial in succeeding in such a long-term project, and for their understanding of my tantrums.

Table of Contents

Chapter 1
Introduction and Background Research

1.1 Introduction

1.1.1 Problem Statement

Autonomous vehicles, also known as self-driving cars, represent the pinnacle of automotive technology advancements. These vehicles utilise sensors, cameras, radars, and artificial intelligence (AI) to navigate and assess the environment around them. Supervision cameras require clear images for tasks such as further face recognition and movement detection (used in CCTV, etc.). As these vehicles heavily depend on accurately perceiving their surroundings to make decisions, the quality and clarity of the visual information received from the cameras are crucial. This is where the importance of dehazing images becomes apparent.

In the industry [1], the predominant sensors employed include Light Detection and Ranging (LiDAR), standard cameras, and infrared (IR) cameras, with LiDAR playing a critical role due to its sensitivity to various weather phenomena like rainfall, fog, and snow. Based on the findings of Rasshofer and Gresser, 2005 [2], the necessity for weather condition algorithms is underscored by statistics and evidence, such as that shown in Figure 1.1, which demonstrates how fog can significantly impair the operation of LiDAR and cameras, thereby affecting the movement and procedural execution of autonomous vehicles. Furthermore, beyond mere visibility, the clarity of images is essential for the effective segmentation and detection of objects, a prerequisite for the autonomous navigation of vehicles. To this end, the convolutional neural network (CNN) method I propose aims to enhance this aspect by effectively removing fog from images, ensuring the production of clear images crucial for both accurate perception and the reliable functioning of these vehicles.

	Short Range Radar	Long Range Radar	LiDAR	Camera	IR Camera
Operation in Rain	++	+	O	O	O
Operation in Fog or Snow	++	++	-	-	O
Operation if Dirt on Sensor	++	++	++	- -	- -
++: Ideally suited, +: Good performance, O: Possible, drawbacks, -: Bad, - -: Impossible					

Figure 1.1: Common advantages and limitations of automotive sensors across various weather conditions.[1]

1.1.2 Aims & Objectives

- To demonstrate the utility of image dehazing algorithms and provide a comprehensive understanding of image dehazing methodologies, with a focus on encoder-decoder techniques.
- To develop an end-to-end convolutional neural network algorithm using U-Net architecture for single image dehazing, prioritising time efficiency, user-friendliness, and achieving favourable visual and technical outcomes.
- To document the iterative improvements throughout the development process and assess the proposed network's performance qualitatively and quantitatively.

Evaluation will include visual representation and two key metrics: PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index Measure), ensuring assessment of both perceptual and statistical accuracy in the dehazing results.

1.1.3 Project Deliverables

This project delivers a comprehensive report and an effective single image dehazing software. The report aims to provide insights into various dehazing techniques, enhancing understanding among researchers and practitioners in computer vision. The project focuses on explaining and testing the development process of an encoder-decoder method using U-Net architecture for single image dehazing. It prioritises factors like time efficiency and user-friendliness to offer a practical solution, focusing on accuracy improvements for practical deployments.

The project also offers a software repository containing all versions of the code, facilitating understanding of the evolution of the final encoder-decoder model. It includes the final trained model on the dataset, and its weights and two main files for dehazing images. One allows for direct output dehazing of hazy images uploaded, while the other provides options for uploading hazy images with ground truth images to also obtain PSNR and SSIM evaluations. The code structures that generate the evaluation graphs, tables, and figures are available in the repository as reference for future work improvements and opportunities as well as reusability.

1.2 Atmospheric Scattering Model - Mathematical Representation

McCartney [5] initially proposed the atmospheric scattering model to mathematically represent a hazy image, which was subsequently refined by Narasimhan and Nayar [6], [7]. Written as:

$$I(x) = J(x)t(x) + \alpha(1 - t(x)), \quad (1)$$

In this equation, I(x) denotes the observed hazy image, J(x) represents the true scene, t(x) signifies the medium transmission, indicating the fraction of light that reaches the camera without scattering, α stands for the global atmospheric light, and x indexes the pixels.

$$\text{The transmission } t(x) = e^{-\beta d(x)}, \quad (2)$$

The transmission t(x) is influenced by both the distance d(x) from the camera to the scene point and the atmosphere's scattering coefficient β, with t(x) nearing zero as d(x) increases indefinitely. In practice, α is determined not from the impractical infinite distance but from the maximum intensity within regions where t(x) falls below a threshold, highlighting the importance of accurately estimating the medium transmission map for effective haze removal.

1.3 Image Dehazing in Computer Vision

Computer vision [87], an area within artificial intelligence (AI) uses machine learning and neural networks to empower computers and systems in extracting valuable insights from digital images, videos, and other visual data. This facilitates their ability to provide recommendations or take necessary actions upon detecting defects or issues.

Computer vision and machine learning techniques are crucial for image dehazing, improving the visibility and clarity of images taken in challenging weather conditions like fog, haze, or smog. These techniques are vital for a range of applications, including autonomous driving, outdoor surveillance, and aerial photography. Initial methods for fog elimination strive to estimate the medium transmission and atmospheric light for haze removal. This section explores two methods: the fundamental approach in traditional computer vision methods for image dehazing: Dark Channel Prior (DCP) [4], and an early CNN method DehazeNet [3], which still focuses on capturing the Transmission Map Refinement; and modern full CNN methods with focus on autoencoder-decoder architectures for image denoising or segmentation.

1.4 Traditional Methods

1.4.1 Atmospheric Light Estimation: Dark Channel Prior (DCP)

The Dark Channel Prior (DCP) [4] method, introduced in 2011 by Kaiming He et al., stands as a cornerstone in the field of image dehazing. By leveraging statistical observations, DCP identifies regions with low pixel intensities, known as the "dark channel," which typically indicate shadowed areas obscured by haze in outdoor images. Through a series of steps, DCP estimates atmospheric light, computes the transmission map to gauge light scattering, and refines this map using soft matting techniques. Finally, utilising the refined transmission map and atmospheric light estimates, DCP reconstructs the haze-free image, unveiling the true colours and details of the scene.

While DCP offers a robust statistical approach to image dehazing, it does encounter limitations under certain conditions. For instance, it may struggle to accurately estimate transmission for objects closely resembling atmospheric light or lacking distinct shadows, such as white marble. Additionally, conventional haze imaging models, upon which DCP relies, may overlook complex atmospheric phenomena like the sun's influence on the sky or the bluish tint of distant horizons. To overcome these challenges, ongoing research seeks to explore advanced models capable of capturing these intricate environmental effects more accurately, thus enhancing the efficacy of haze removal techniques.

There are still papers that use DCP in the process of dehazing, for example DA dahazing, 2020, [88] accompanied by training both the image translation and dehazing network simultaneously in an end-to-end manner, incorporates the properties of the clear image, such as the dark channel prior and image gradient smoothing, to enhance domain adaptivity and improve the effectiveness of image dehazing.

1.5 Deep learning methods

1.5.1 Convolutional Neural Network

The advancement of deep learning methods has notably impacted image dehazing, particularly through the adoption of convolutional neural networks (CNNs) and generative adversarial networks (GANs). Early methods [3], [8] employed CNNs to forecast the transmission map and then recover the haze-free image based on established atmospheric scattering models, such as DehazeNet [3], as detailed below.

More recent efforts have transitioned to designing comprehensive end-to-end solutions based on either CNNs [9,10,11] or GANs [12,13]. These advanced models aim to directly achieve image dehazing without the explicit need to estimate the transmission map. Particularly, the adoption of encoder-decoder architectures has become prevalent among these methods, allowing for the extraction of multi-scale features essential for dehazing.

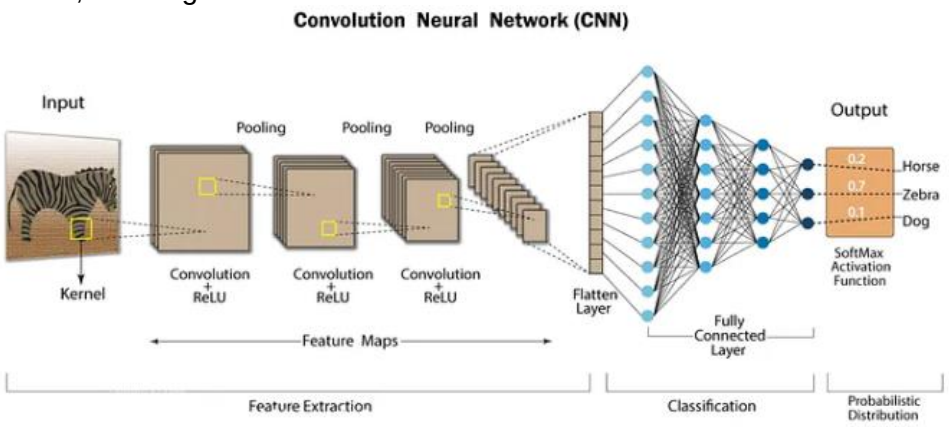


Figure 1.2. An example of CNN Architecture for object classification [14]

As neural networks become more complex, a notable challenge emerges, the potential loss of spatial contextual information, essential for preserving the accuracy of feature representations. Consequently, despite their progress, many advanced dehazing methods based on deep learning still encounter difficulties in effectively addressing noticeable artifacts.

1.5.2 Transmission Map Refinement DehazeNet

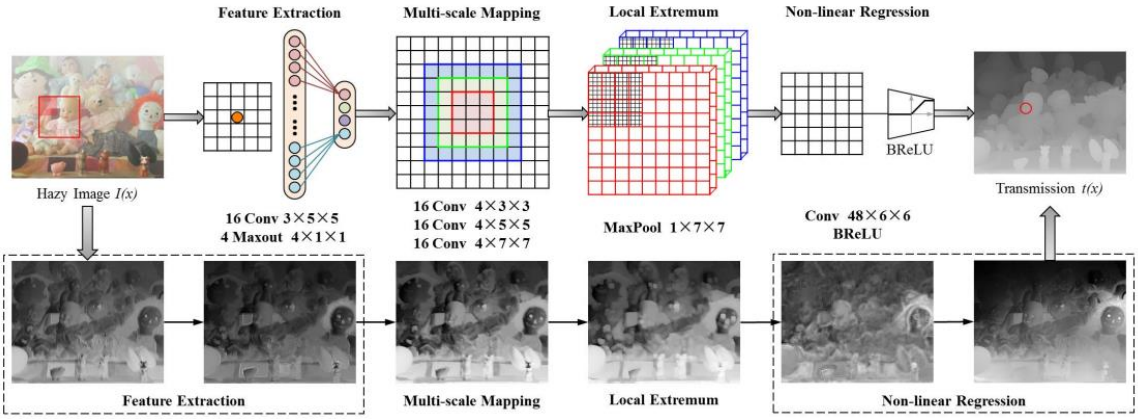


Figure 1.3. Illustrates the architecture of DehazeNet[3]. DehazeNet is composed of four consecutive operations, including feature extraction, multi-scale mapping, local extremum detection, and non-linear regression. These operations are implemented using three convolutional layers, a max-pooling layer, a Maxout unit, and a BReLU activation function[3].

DehazeNet [3], introduced in 2016, represents an early application of Convolutional Neural Networks (CNNs) for image dehazing. Unlike conventional methods relying on handcrafted features, DehazeNet adopts a data-driven approach, directly learning the mapping between hazy images and their corresponding transmission maps from data. Employing a patch-based strategy, DehazeNet processes small regions of the hazy image to locally predict the transmission map, enabling the capture of local variations in haze density influenced by factors such as object distance and atmospheric particle concentration. Once estimated, these transmission maps undergo refinement and aggregation to produce a comprehensive map for the entire image.

The transmission map, denoting the fraction of light reaching the camera without scattering, is a pivotal component of DehazeNet's architecture. Comprising three convolutional layers, a max-pooling operation, a Maxout unit, and a Bilateral Rectified Linear Unit (BReLU) activation function, the CNN architecture effectively learns to estimate transmission maps. Notably, the BReLU activation function, an extension of the Rectified Linear Unit (ReLU), plays a crucial role in achieving accurate image restoration, highlighting its significance in DehazeNet's performance.

1.5.3 Encoder-Decoder Architectures

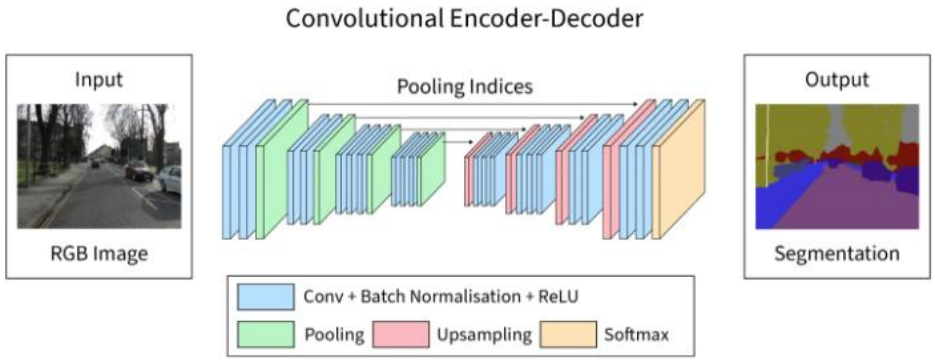


Figure 1.4. An example of CNN Encoder-Decoder Architecture for Image Segmentation, also known as SegNet(2016)[15]

Encoder-decoder architectures[16, 17, 18] are important in deep learning for jobs where you need to change one type of data into another, this includes making noisy images clear, cutting out parts of images, or translating languages. These architectures excel when there's a disparity between the input data and the desired output but maintain a meaningful connection between them.

Compared to traditional techniques that require separate estimations of the transmission map and atmospheric light, a process prone to inaccuracies due to unfamiliar datasets or incorrect interpretations of lighting and image shades, these methods could yield erroneous denoising outcomes.

This project focuses on image denoising undertaking architectures of image segmentation, particularly using the U-Net architecture, inspired by pioneering work in various encoder-decoder applications like SegNet[90] using VGG16 for pixel-wise semantic segmentation; FCN[91] for transforming fully connected layers into convolutions for semantic segmentation; DeepLab[92] and its matrix convolutions for detailed feature capture; and PSPNet [93] which uses a pyramid pooling module to enhance pixel-wise predictions.

Notable developments in image dehazing include U-Net variants like: DRGNet[44], which incorporates residual blocks and the SK fusion module; DSEU[23], which integrates Squeeze-and-Excitation (SE) blocks and parallelized dilated convolution blocks for enhanced attention and detail capture; improved U-Net [43] employs Group Normalisation instead of Batch Normalisation to address the issue of insufficient batch size caused by hardware limitations, it also introduces FReLU activation into the U-Net block, enabling the capture of complex visual layouts with standard convolutions; ST-UNet [89] is designed for VHR image processing and utilises parallel processing and an optimised data structure to maximise computing resources for faster processing.

Other notable encoder-decoder models for image dehazing include: GMAN-net[50], which avoids colour darkening and excessive sharpening; Y-net[77], which uses the W-SSIM loss function for high-quality image restoration; W U-net[76], which utilises wavelet transforms for feature extraction and chromatic adaptation transform to enhance images; and AOD-Net [9] effectively combines a K-estimation module using five convolutional layers with a clean image generation module, utilising element-wise operations to efficiently dehaze images within a singular, integrated framework, demonstrating its applicability in enhancing image clarity across various deep learning applications.

Encoder-decoder architectures like U-Net have revolutionised tasks such as image dehazing and segmentation by adeptly transforming complex inputs into detailed outputs. Innovations in these models, including advanced blocks and optimization techniques, significantly enhance their effectiveness and adaptability, making them vital in advancing image processing technologies.

Chapter 2
Methods

2.1 Sprints

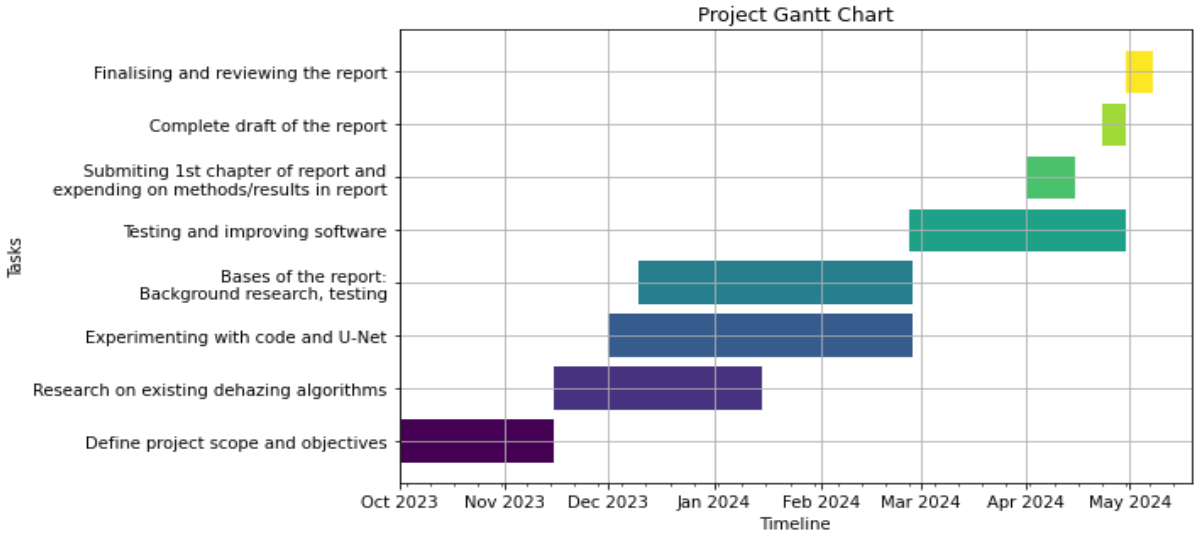


Figure 2.1. A representation of the project timeline.

As seen on the timeline of the Gantt chart above, extensive background research was essential to the development of this Single Image Dehazing project, totalling approximately four full months. This phase was crucial as we were not familiar with the domain of computer vision/data training. The decision to focus on dehazing software was further inspired by the discovery of a well detailed and extensive dataset, Foggy Cityscapes. Testing and improving the U-Net algorithm posed a challenge, with another three and a half months spent on development, especially testing and comparing results, including enhancing the U-Net architecture and comparing a total of nine loss architecture functions.

At the end of the timeline, the final month was dedicated to rigorously drafting and writing the report, in accordance with references from professional papers studied throughout the year; approximately 50 single image dehazing papers were consulted for this project.

2.2 Development tools

The programming language chosen for this project is Python 3, a well known choice for projects involving machine learning and CNN architecture algorithms. Most papers referenced in this report utilise Python or Matlab for their execution. Python is an excellent tool that supports open source machine learning platforms like TensorFlow[24] and PyTorch[25], which are libraries extensively used in computer vision.

2.2.1 Setup

The code was developed in the initial stages using Microsoft's free, open-source code editor, "VS Code". I tested existing papers and conducted background research using this for ease of working with multiple files and testing different trained models with the dataset chosen for the project for comparing dehazing methods: DehazeNet, FogRemoval [27], personal initial architectures, and additional peers' dehazing/defogging algorithms. The first basis of the U-Net architecture was also developed in VS Code and trained as the

architecture was still at a low level of training, and the laptop's GPU(Intel(R) UHD Graphics), although still slow with hours of training, could successfully train our model.

After establishing the basis of the architecture and needing to use a larger amount of images for training and more epochs, the laptop used was averaging days of work to train the model. The platform used further was Google Collab for its availability of connecting with Google Drive and training the model on a large amount of data (9 GB of images to train on) and for the performance offered by the online, cloud supported GPUs. To run the software, V100 GPU, T4 GPU, and High RAM were used for maximum efficiency. The code was developed in Jupyter Notebooks, necessitating the use of Google Colab. Additionally, the tables and graphs made to represent the efficiency of the software were also created in Jupyter Notebooks.

2.2.2 Version Control

The version control tool used for this project is Github, using the university setup repository[26]. Github is a great tool for keeping track of your progress and different versions of code. It helped me greatly when testing different subparts of the algorithms to keep track of the best one and reuse old code or techniques that approved higher accuracy results. Having experience in using GitHub also facilitated the choice of using this system version.

2.2.3 Libraries

TensorFlow is a well known open source deep learning framework developed by Google. It is a tool used in the manipulation of mathematical expressions over numerical tensors[28]. In the image dehazing field, the work on PyTorch or TensorFlow is divided. The structure I have followed is more user friendly using TensorFlow and is much more widespread in software using a U-Net architecture in various types of image segmentation problems. It is important to keep in mind that TensorFlow is not a single library, but a well documented platform. The library we used in the project from TensorFlow is Keras[29], a deep learning API for Python. Keras is used in deep learning models, which allows the use of a wide range of different workflows beneficial for updating code and testing different scenarios to find the most efficient method.

Most importantly, the base of the architecture utilises the following extensions from tensorflow.keras: {layers, models, optimizers, losses, applications, regularizers}. These are crucial in an image encoder-decoder architecture, facilitating the foundation of the structure. The Numpy library is used in the custom made loss function. The OpenCV[31] and Matplotlib[30] libraries are used for handling image processing and displaying data/images.

2.3 Dataset

The datasets for training and validating the model for this software include both Foggy Cityscapes[32] and the hazy RESIDE[34] dataset to offer diversity in image models and different intensities of fog, as well as datasets that are based outdoors and indoors.

Foggy Cityscapes is a synthetically generated dataset based on the Cityscapes[33] dataset, which is used for various projects, including image segmentation and object detection, primarily in autonomous vehicles. It offers a variety of outdoor street view images where fog is predominantly present under challenging weather conditions. It has three levels of fog intensity: beta = 0.01, beta = 0.02, and beta = 0.005. The following subsets were chosen: Strasbourg, Hanover, Hamburg, Aachen, with a total number of 5939 pairs of hazy and clear images, with dimensions 2048X1024.

Figure 2.2. FC clear and foggy example.

A second dataset used for training and validation of the model is RESIDE. This dataset is employed particularly for its ITS (Indoor Training Set) images, which provide additional diversity to the model. The subsets chosen include the validation set, with a total of 10,000 hazy images(309.6MB) and 1,000 clear images(2.4GB), with dimensions 541x407. Each image features 10 different fog intensities, randomly generated, varying from beta = 0.9 to beta = 0.7.

Figure 2.3. ITS clear and foggy example.

The model is trained on the two datasets separately and on the combination of both of them to offer a variety in training and to observe the results on a larger amount of data.

To accurately compare the proposed method with other methods in the industry, the testing and results are conducted on both datasets, across the different subsets of images. This approach tests the efficiency of the model on new, unseen images, as well as on images the software has already been trained on. The majority of papers utilising CNN architecture train and test their model on the RESIDE dataset, with prominent examples detailed in these papers: DCP[4], MSCNN[8], DehazeNet[3], AOD-Net[9], GMAN[50], DRNNet[51]. Results and specifics of the testing process will be discussed in the following chapter.

2.4 Pre-Processing data - Distributing the dataset

For testing this algorithm two different datasets were used to offer visual diversity: Foggy Cityscapes (FC) and Indoor RESIDE (IR). To offer visual diversity in the model training, one dataset contains only street images, FC, while the other, IR, exclusively features indoor images of objects. This diversity helps the model adapt and facilitates the reproduction of future images it will be tested on. The images from these datasets are downscaled to 256x256 pixels for consistency, reduced overfitting and resource management. For each clear image in the FC dataset, three different fog intensities (0.01, 0.02, 0.005) are assigned; while for each clear image in the IR dataset they get assigned with ten randomised fog intensities. After pairing clear images with their hazy counterparts, a total of 2949 pairs of images from FC and 2990 pairs from IR were collected, making a total of 5939 pairs. The data is then shuffled and divided into 80% for the training dataset and 20% for the validation dataset. The complete dataset is now ready to be trained on with the previously generated model.

2.5 U-Net CNN Architecture

2.5.1 Basic U-Net Architecture

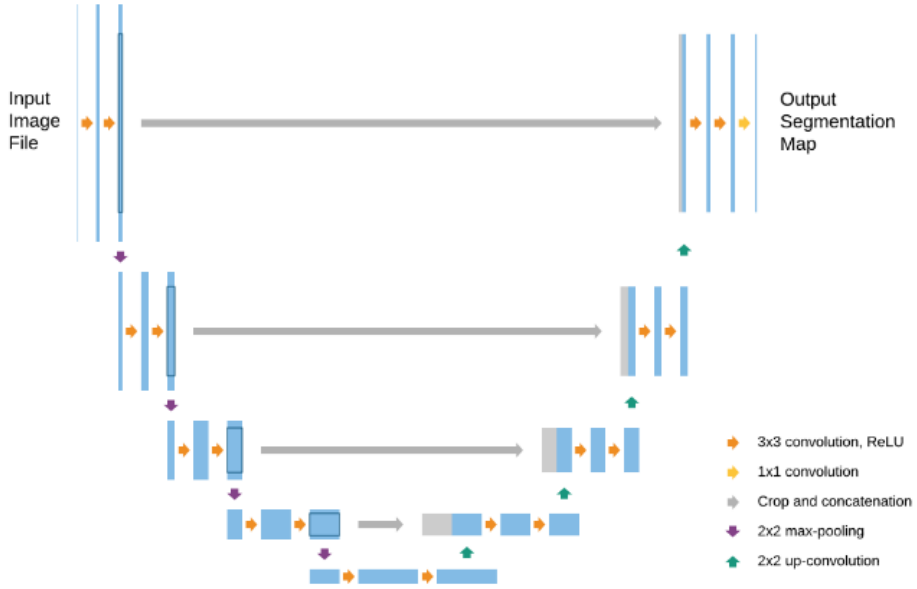


Figure 2.4. The base of a U-Net architecture[36]. The arrows signify the various processes, while the blue rectangles indicate the feature map at each layer, and the grey rectangles denote the trimmed feature maps from the contracting path.

This type of network architecture shown in Figure 2.4. forms the base of an U-Net architecture[36]. Since its creation in 2015 [35], U-Net is best known and widely used in the biomedical image segmentation field, with numerous papers employing this structure[36, 37, 38, 39]. U-Net is especially effective because it generates precise segmentation maps even with a relatively small number of training samples. Additionally, U-Net trains more quickly compared to many other segmentation models due to its ability to learn from contextual information[36]. Its versatility makes it an excellent tool that can be adapted for various image-to-image translation tasks, including image dehazing; some other papers that this project took inspiration from[23, 42, 43, 44].

The chosen architecture is user-friendly and offers the potential for enhancement. Throughout the development process, numerous additional features were incorporated and tested to refine the model.

To explain the base architecture of the U-Net [40, 41]; it can be divided into two parts: the contracting path, first half of the U-shaped architecture, and the second part as the expansive path. The first half of the U-shaped network is a normal CNN architecture which uses two successive **3x3 convolutions**. These convolutions are used to extract features from the image. It is followed by an **activation function**, most used in industry being ReLU[57] or LeakyRelu[62, 78]. This is used after each convolution to introduce non-linearity into the model, helping it learn more complex patterns. **Max-Pooling Layers** follow the convolutions and serve to reduce the spatial dimensions of the feature maps. This down-sampling helps in increasing the receptive field of the convolutions and reduces the computational load for subsequent layers. As the U-Net is an encoder-decoder architecture, this first part consists of the encoder part and it is primarily responsible for capturing the context in the image.

The second part of the U-shaped architecture, known as the decoder, is responsible for enabling precise localization using the context gathered by the contracting path. This part

reconstructs the segmentation map from the encoded features. For its structure 2x2 **up-convolution (transposed convolution)** is used to reverse what the initial contracting path has done. It is followed by **cropping and concatenation,** the feature map that has been enlarged is fused with the appropriately trimmed feature map from the contracting path. Trimming is required since the dimensions of the feature map after up-convolution may differ from the expected size due to the loss of edge pixels during each convolution. This process is vital as it merges the contextual, high-level features from the down-sampling path with the detailed localization features from the up-sampling path. To match the first half of the U-shaped structure two successive **3x3 convolutions** are applied, followed by the **activation function** (ReLu/LeakyReLU). These serve to refine the features post upsampling and concatenation. Final **1x1 convolution**, known as the output layer, is the last convolutional step used to map the deep feature maps to the desired number of output channels.

By combining these properties, U-Net architectures are efficient for the task of image dehazing, potentially resulting in clear, dehazed images while keeping the details and structures in the given picture.

2.5.2 Mathematical energy function representation

$$E = \sum w(x)\log\left(p_{k(x)}(x)\right), \quad (1)$$

Where pk represents the per-pixel SoftMax function[45] utilised on the ultimate feature map.

$$p_k(x) = \frac{e^{a_k(x)}}{\sum_{k'=1}^{K} e^{a_k(x)'}} \quad (2)$$

And ak(x) signifies the activation within channel k.

The SoftMax function operates on a per-pixel basis across activation levels that denote distinct categories, a standard approach in semantic segmentation tasks. In such scenarios, the network is trained to convert intricate inputs, such as medical images - in our case hazy images, into multichannel outputs. Each output channel indicates the probability of a particular class's presence at every pixel. This design enables the network to deliver precise, location-specific inferences regarding the image's elements in a spatially consistent way.

Chapter 3
Results and Evaluation

3.0 Introduction

3.0.1 Structure and timeline summary

In this chapter, the evolution of improvements made to the dehazing software, built on the basis of a U-Net architecture model, is discussed. Multiple methods are tested, and the reasoning for choosing them is explained, with the best one selected for the final method. A summary of the base U-Net architecture improvements selected along the way: using batch normalisation and kernel initialization, two additional convolutional layers are added to the encoder and decoder, a squeeze-and-excite block is included, and dropout and a bottleneck are implemented, various activation functions were tested (with ReLU chosen), 4 loss architectures were evaluated (with VGG19 chosen), and tests were conducted on two separate and combined datasets.

The best method is selected based on visual results (on 7 different fog intensity images) and with technical parameters (PSNR and SSIM evaluations). It will be observed that some improvements benefit the measurement values, while others enhance the visual image quality. It will also be noted that higher PSNR and SSIM results do not necessarily mean that the image is reconstructed correctly from a visual standpoint. Therefore, maintaining a balance in testing both aspects is essential to the project. Although all tested methods offer room for improvement or development of personalised activation and loss functions, which will be further explained in Chapter 4. Discussion, under Future Work.

3.0.2 Evaluation parameters

PSNR (Peak Signal-to-Noise Ratio)[52] is generally used as a measure of image quality compared to an original image, and a higher PSNR indicates better quality, the lowest results after training the model should be around 15 for a reasonable image output. As the beta value increases, indicating more fog, the quality of dehazing decreases, as shown by the lower PSNR and SSIM scores Table 0.1. SSIM (Structural Similarity Index)[52] measures the similarity between two images. The SSIM values should start from 0.4 for a decent image output. A value closer to 1 would indicate perfect similarity.

There is a visible correlation between the observed visual quality in the stitched images and the numerical metrics provided in the table, validating the effectiveness of these metrics in assessing the performance of the dehazing algorithms. The variation in results across different images significantly impacts how the model performs; even with visually poor quality results (Figure 3.0), the highest PSNR detected is very high, 16.1, suggesting good image quality of architecture chosen. PSNR and SSIM must be analysed together as a significant discrepancy can lead to misleading conclusions. It is important to test for a variety of photos and reach to achieve similar values, as this proves the efficiency of the model.

3.0.2.1 Visual results

For the initial visual testing of this project five clear pictures were chosen with seven different fog intensities. As stated above, besides the technical values, which are crucial but also challenging to use to determine the performance of your model, the image quality can be deceptive. In Figure 3.0, technical results are high for some images, but the image quality is unintelligible, so both visual and technical parameters must be checked. The dataset used for these initial tests is limited and lacks image diversity, so it is important to include a variety

of images from both datasets and a range of fog distributions, these are included at the end of testing.

We observe that the RESIDE images Fig. 3.0.1 has a more evenly distributed fog level, all buildings, even in fog are distinguishable, while the Foggy Cityscapes images exhibit a more concentrated level of fog in specific areas (such as the sky which becomes fully white). At first glance, the RESIDE images are considered to have more fog in terms of quantity, but for the software, the Foggy Cityscapes images are much more challenging as the fog is extremely dense in certain parts, making it impossible to see through even with the naked eye.

Figure 3.0.1 Representation of foggy sectioning for FC and IR images

3.0.2.2.1 Industry results

Method	DCP [4]	BCCR [46]	GRM [47]	CAP [48]	NLD [49]	DehazeNet [3]	MSCNN [8]	AOD-Net [9]	GMAN [50]	DRHNet[51]
PSNR	18.870000	17.870000	20.440000	21.310000	18.530000	22.680000	20.010000	21.010000	27.940000	31.390000
SSIM	0.794000	0.770000	0.823000	0.824000	0.702000	0.833000	0.791000	0.837000	0.897000	0.974000

Table 0.1 Quantitative Dehazing Evaluation: Average PSNR and SSIM on 500 Indoor Images in SOTS Dataset.

These are different industry recognised results. Which are tested on the Outdoor RESIDE dataset, our tests are also made on this plus the FC dataset. In Table 0.1 we observe the results. These are accompanied by visual representation as well for the set of images please check Appendix C.1. The results are based on the trained model and how well it performs on a small dataset of images 500, 10 images.

3.0.3 Basic U-Net Architecture

3.0.3.1 Observation of standardised testing

For consistent and efficient testing, the initial models are trained on the Foggy Cityscapes (FG) dataset, which includes 2949 pairs of images. The ReLU activation function is the standard choice until further evaluations in section 3.6, due to its prevalence in existing papers. Similarly the MSE Loss Function will be used until other loss functions and architectures are explored in section 3.7 for simplicity. The final convolutional layer initially uses the commonly preferred sigmoid activation, but additional activations will be tested later for consistency with other research addressing similar problems.

3.0.1 Outline

The build_unet() function in the train file defines a U-Net model structured into two main parts: the encoder and the decoder.The encoder uses two successive 3x3 convolutions ranging from 64 to 256 filters, each followed by a ReLU activation function to introduce non-linearity. Max-Pooling layers are employed after convolutions to reduce the spatial dimensions of the feature maps; in the decoder part, the model utilises 2x2 Up-convolutions (Transposed Convolutions) to upsample the feature maps, gradually restoring their spatial dimensions to match those of the input image. This is followed by Cropping and

Concatenation, where the upsampled feature maps are combined with the corresponding cropped feature maps from the encoder via skip connections. The decoder ends with two successive 3x3 convolutions and a final 1x1 sigmoid convolution that maps the deep feature maps to the desired number of output channels, effectively reconstructing the image.

The entire network utilises the Mean Squared Error (MSE) loss function during training. This function calculates the pixel-wise discrepancies between the predicted output images and the actual ground truth images. By minimising this error, the model learns to improve its predictions, gradually enhancing its ability to accurately reconstruct the dehazed images.

3.0.2 Results and Observations

Visual results:

Figure 3.0. Representation of 3_0BasicUNet model. For bigger/clearer images please check Appendix C.3.0.a/b.

Visually the image is reconstructed but the layers are still lacking, the outline of the image is distinctable but the model struggles with reconstructing colour. As seen even from these low quality reconstruction the images with less fog appear more clear in the reconstruction.

Performance Metrics results:

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	**15.31**	0.4793
hamburg_beta_0.01.png	14.78	0.4598
hamburg_beta_0.02.png	14.23	0.4471
0182_0.8_0.16.jpg	13.35	**0.2909**
0100_0.9_0.12.jpg	11.21	**0.4864**
0001_0.8_0.2.jpg	10.86	0.3646
0391_1_0.08.jpg	**8.68**	0.4594

Legend:

Highest Value

Lowest Value

Table 0.1 Representation of metrics results for 3_0BasicUNet model.

The PSNR values range from about 8.68 to 15.31, which shows a significant variation in image restoration quality, based on the intensity of the fog in each image, which is not a good sign, the trained model is unstable. SSIM measures the similarity between two images, here the SSIM values range from 0.2909 to 0.4864, which is a reasonable result but a closer number to 1.0 is better.

3.1 Improving Convolutional Layers

3.1.1 Outline

To improve the convolutional layers in Conv2D() for better image learning we add **batch normalisation** and **kernel initialization**. These are techniques that help in stabilising and accelerating the training of deep neural networks.

Batch normalisation[46] simplifies training by standardising layer inputs, creating more robust learning, faster convergence due to stable distribution of activations, and

facilitating the use of higher learning rates. Additionally, it introduces a mild regularising effect similar to dropout, mitigating overfitting.

For kernel initialization**,** “He_Normal” initialization[47], is an initialisation method specifically designed for layers that have ReLU activation functions and it is the most common in image dehazing[48]. Addresses the vanishing/exploding gradients Problem, it helps in preventing the gradients from becoming too small or too large, which can cause the training to stall or explode. Improves convergence , good initialization methods lead to quicker convergence by providing an appropriate starting point for the optimization process. This initialization is effective for networks with many layers, which is common in modern neural networks.

3.1.2 Results and Observations

Visual results:

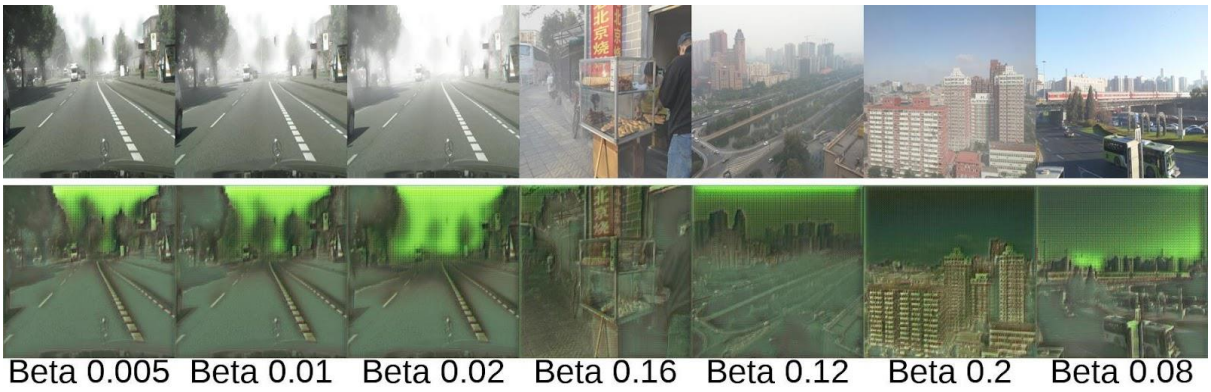


Figure 3.1. Representation of 3_1ImprovedConvBlock model. For bigger/clearer images please check Appendix C.3.1.a/b.

The images show a clearer definition compared to previous ones, though there is a green hue overall because the model is not yet fully capable of accurately recreating the colour of the ground truth image. Although the difference in image quality is not substantial, the outline of objects is sharper and more well defined. Additionally, the model clearly identifies the fog, and the efforts to recreate a more accurate image are visibly improving. In the first three dehazed images, even trees further away are outlined, indicating that the model is making progress.

Performance Metrics results:

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	**14.71**	**0.4798**
hamburg_beta_0.01.png	14.24	0.4422
hamburg_beta_0.02.png	13.45	0.3911
0182_0.8_0.16.jpg	12.21	0.3906
0100_0.9_0.12.jpg	9.9	0.3939
0001_0.8_0.2.jpg	10.78	0.4438
0391_1_0.08.jpg	**8.82**	**0.3102**

Legend:

Highest Value

Lowest Value

Table 1.1 Representation of metrics results for 3_1ImprovedConvBlock model.

The performance metrics have not significantly changed compared to Table 0.1, in fact, they have slightly decreased. This approach is not substantially enhancing the model's performance metrics but is instead refining what it was already doing, to offer a visually better image, which is also needed for image dehazing.

3.2 Improving the Encoder and Decoder

3.2.1 Outline

To enhance the encoder and decoder, **two additional layers** with 512 and 1024 filters were added into the contracting path, enhancing the model's capability to capture more detailed and complex features at different scales, enabling a more in depth analysis. The layer with 1024 filters is the deepest, equipped with the most filters. Corresponding **upsampling layers** were added to match the additional downsampling layers, ensuring the symmetry of the network, which is crucial in the U-Net architecture. Each upsampling layer in the expansive path is **concatenated** with its corresponding layer from the contracting path, providing the local information necessary for precise localization.

3.2.2 Results and Observations

Visual results:

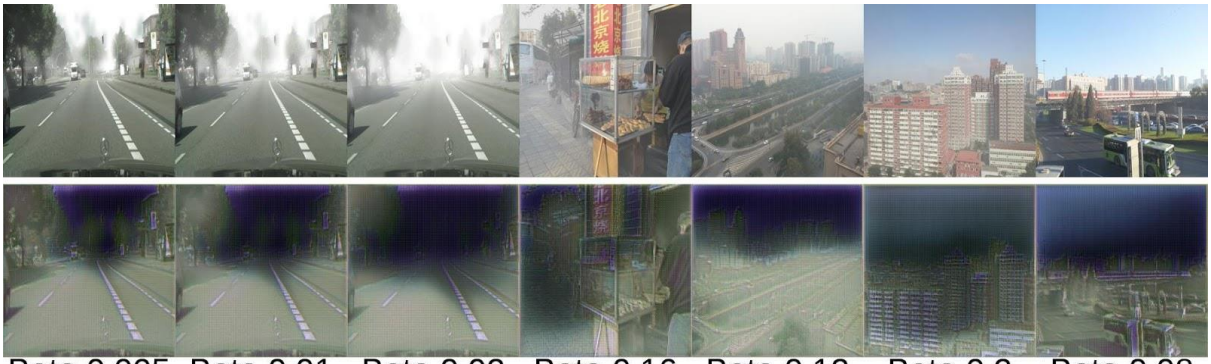


Figure 3.2. Representation of 3_2ImprovedEncoder model. For bigger/clearer images please check Appendix C.3.2.a/b.

With two additional layers added, the CNN model can develop deeper and recreate images more effectively. From the dehazed results, it is already apparent that the model can distinguish between the sky (the more hazy/bright areas) and the streets/buildings (the clearer parts of the image). These layers contribute to greater uniformity, softening the sharpness and outlines seen in previous examples and making the images more consistent. The colour hue has also shifted to a more neutral tone, closer to the ground truth images, compared to the bright neon green of previous examples; other colour shades such as red, yellow, and purple are now more visible, beyond the general blue/grey hue.

Performance Metrics results:

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	**13.67**	**0.457**
hamburg_beta_0.01.png	13.08	0.4235
hamburg_beta_0.02.png	12.76	0.3938
0182_0.8_0.16.jpg	11.26	**0.3239**
0100_0.9_0.12.jpg	7.67	0.3588
0001_0.8_0.2.jpg	9.28	0.3372
0391_1_0.08.jpg	**6.75**	0.3422

Legend:

Highest Value

Lowest Value

Table 2.1 Representation of metrics results for 3_2ImprovedEncoder model.

In terms of performance metrics, the values have slightly decreased, which is understandable as the composition of the image is beginning to change. The image now appears darker and might initially seem worse than before, but it contains more details captured by the model. With the impact on image composition, darker images with more details result in reduced PSNR or SSIM scores, as these metrics may not fully appreciate the complexity or darker tones, suggesting a perceived quality decline. It is important to remember that both visual and technical metrics are crucial in image dehazing. While the enhancements in the CNN architecture allow for a richer and more detailed extraction of features, which is beneficial for visual quality and realism, they might temporarily lead to a

drop in performance metrics as these metrics may not fully align with the improved visual output or might be indicating a need for further model optimisation and tuning.

3.3 Squeeze and Excitation (SE) blocks

3.3.1 Outline

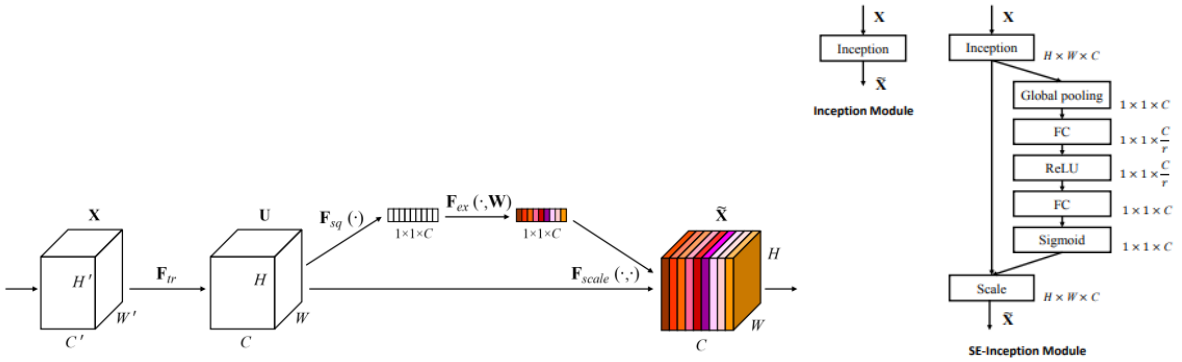


Figure. 3.3.1 A Squeeze and Excitation block. [53]
Figure. 3.3.1 The diagram on the left illustrates the original Inception module, while the diagram on the right depicts the SE-Inception module. [53]

The Squeeze and Excitation (SE) block[53], reviewed by the University of Oxford, adaptively recalibrates channel wise feature responses by explicitly modelling interdependencies between channels. This SE module is composed of two parts: squeezing, which compresses the global spatial information of the feature map, and excitation, which uses this information to emphasise informative features while suppressing less useful ones. This mechanism allows the network to achieve higher accuracy and efficiency in recognizing images [54]. Squeeze-and-Excitation (SE) modules are integrated into U-Net architectures to weight the importance of input feature maps from skip connections. Throughout the network, the ReLU activation function is employed in all layers except for the last, where a Sigmoid activation is utilised [55].

3.3.2 Results and Observations

Visual results:

Figure 3.3.3. Representation of 3_3SEBlocks model. For bigger/clearer images please check Appendix C.3.3.a/b.

With the help of the SE block, we observe significant visual improvements, the dehazed images are clearer and closer to the original colour hue. Notable signs of haze removal are especially visible in the first three images.
Performance Metrics results:

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	**14.12**	**0.4837**
hamburg_beta_0.01.png	13.65	0.4669
hamburg_beta_0.02.png	13.08	0.4419
0182_0.8_0.16.jpg	12.87	**0.2986**
0100_0.9_0.12.jpg	11.5	0.4773
0001_0.8_0.2.jpg	13.2	0.3928
0391_1_0.08.jpg	**10.14**	0.4818

Legend:

Highest Value

Lowest Value

Table 3.1 Representation of metrics results for 3_3SEBlocks model.

For the first time we also see improvements in performance metrics, demonstrating uniformity in results. Compared to Table 2.1, where the lowest PSNR result was 6.75, we now have a lowest result of 10.14, which represents a substantial improvement. The SSIM results are still lacking so there is still a lot of improvement to be made.

3.4 Dropout and Bottleneck

3.4.1 Outline

The combination of the bottleneck [83] and dropout [82] layers is critical in U-Net for dehazing images. By forcing the network to learn robust features that generalise well across different images, it helps ensure that the dehazing process can effectively clear up images across varied scenes and fog densities, not just the ones seen during training. This leads to a more versatile model capable of handling the complex variations typically found in real-world scenarios where visibility is compromised by haze or fog.

In the U-Net architecture, after the fourth convolutional block (conv4), a Dropout layer with a rate of 0.5 is applied (drop4). This means that half of the units in conv4 are randomly dropped during each training iteration. This is followed by a max pooling layer (pool4), which reduces the spatial dimensions of the input, effectively summarising the features extracted so far.

The network then reaches the bottleneck phase, where another convolutional block (conv5) processes these summarised features. A second dropout layer (drop5) is also applied here with the same dropout rate, enhancing the network's ability to generalise by preventing overfitting. This is particularly crucial in the bottleneck, the deepest part of the network, where the risk of learning overly complex patterns that do not generalise well is the highest.

3.4.2 Results and Observations

Visual results:

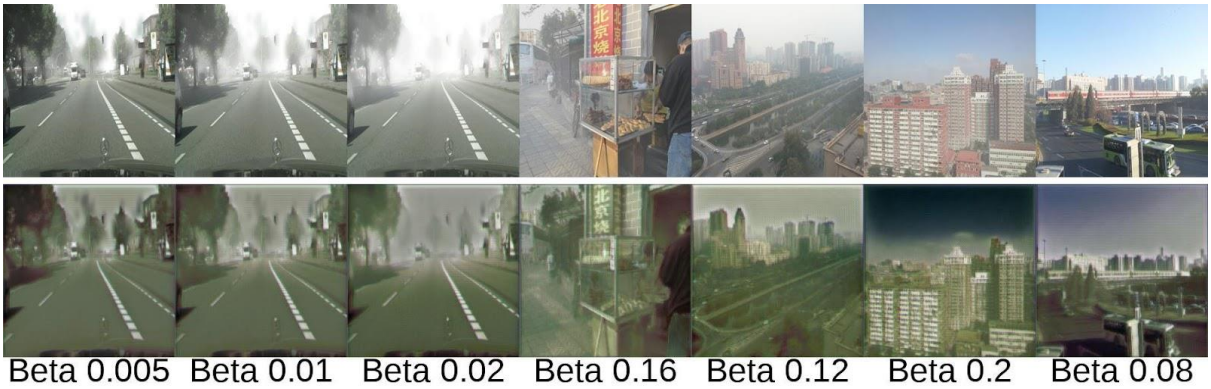


Figure 3. 4. Representation of 3_4Dropout_Bottleneck model. For bigger/clearer images please check Appendix C.3.4.a/b.

There has been a significant improvement in image quality and edge detection, with haze effectively removed. Colours are now closer to those in the ground truth images, and signs of fog removal are already evident in all images, especially for beta of 0.005.

Performance Metrics results:

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	13.23	0.4733
hamburg_beta_0.01.png	12.56	0.4639
hamburg_beta_0.02.png	11.91	0.4503
0182_0.8_0.16.jpg	12.31	**0.2857**
0100_0.9_0.12.jpg	11.61	0.4608
0001_0.8_0.2.jpg	**13.65**	0.4505
0391_1_0.08.jpg	**10.9**	**0.5184**

Legend:

Highest Value

Lowest Value

Table 4.1 Representation of metrics results for 3_4Dropout_Bottleneck model.

Surprising results show that for the image with typically the lowest SSIM score, has now increased to the highest at 0.52, indicating a great improvement in image similarity. Although the average PSNR has slightly decreased, this is a normal part of the enhancement process. At this stage, it's crucial for the model to learn extensively from training and consistently recreate improved image quality.

3.5 Activation Functions

3.5.1 Outline

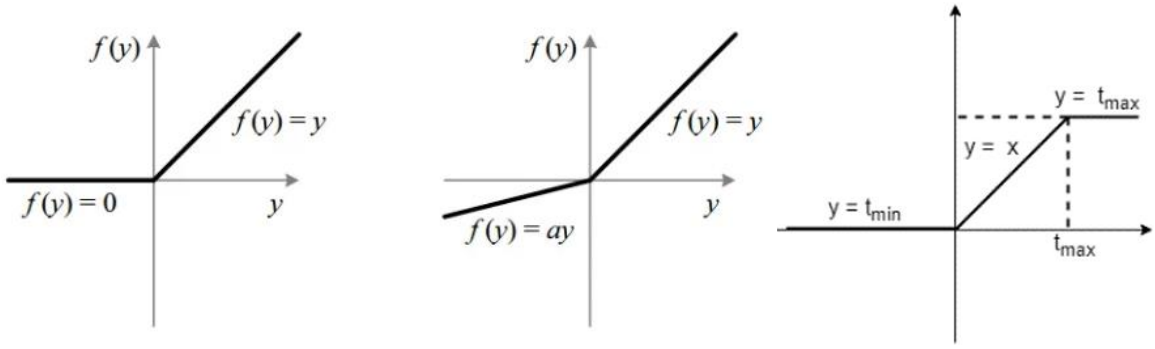


Fig : ReLU v/s Leaky ReLU

(c) BReLU Activation Function

Figure 3.5.1 Representation of different activation functions[56].

The most commonly used activation function in image dehazing is ReLU (Rectified Linear Unit)[60, 61], prized for its efficiency and ability to facilitate faster convergence during training. ReLU operates by passing positive values unchanged while setting negative values to zero, helping to highlight features critical for distinguishing between hazy and clear image regions. This function speeds up the learning process without being overly demanding on resources.

LeakyReLU [62, 78] and BReLU [3, 9, 58, 59] are variations of ReLU that address some of its shortcomings. LeakyReLU allows a small gradient when the unit is inactive (input less than zero), thus preventing neurons from dying, a problem in standard ReLU where neurons stop learning entirely. On the other hand, BReLU limits the maximum output of the activation function, preventing runaway activations and ensuring that outputs remain within a controlled range, which can be particularly useful for normalising outputs in dehazing tasks[57].

Custom activation functions are also explored in image dehazing to address specific challenges unique to the task, such as varying haze densities across images. These activations are tailored to better manage the statistical properties of haze and adapt to the

nuances of image recovery under varied environmental conditions. Some papers that use their own activation functions[51, 63].

3.5.2 Results and Observations

Visual results:

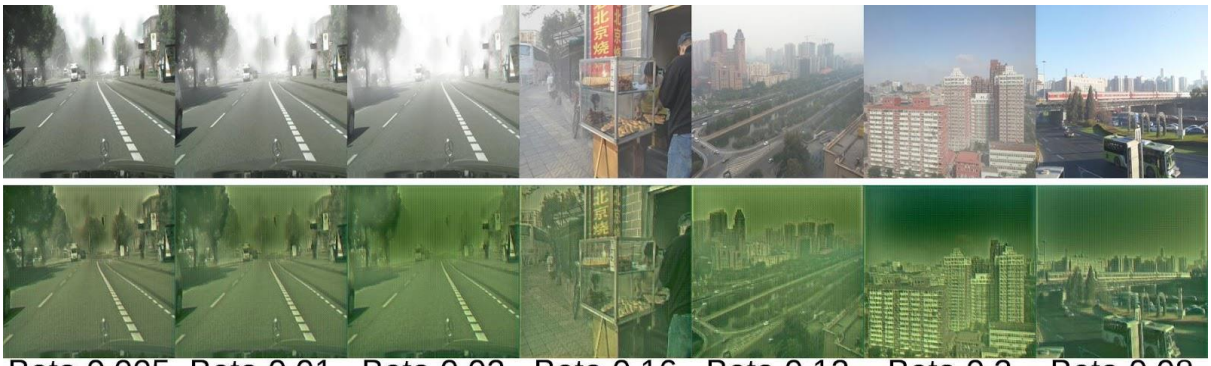


Figure 3.5.2. Representation of 3_5ActivFun_LeakyR model. For bigger/clearer images please check Appendix C.3.5.a/b.

Figure 3.5.3. Representation of 3_5ActivFun_BReLu model.
Performance Metrics results:

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	**14.53**	**0.5159**
hamburg_beta_0.01.png	14.05	0.4897
hamburg_beta_0.02.png	13.65	0.4608
0182_0.8_0.16.jpg	12.94	**0.3734**
0100_0.9_0.12.jpg	10.78	0.4949
0001_0.8_0.2.jpg	12.92	0.4883
0391_1_0.08.jpg	**9.33**	0.4897

Legend:
Highest Value
Lowest Value

Table 5.1 Representation of metrics results for 3_5ActivFun_LeakyR model.

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	**13.04**	**0.4411**
hamburg_beta_0.01.png	12.97	0.4318
hamburg_beta_0.02.png	12.98	0.4174
0182_0.8_0.16.jpg	11.74	**0.252**
0100_0.9_0.12.jpg	10.17	0.4119
0001_0.8_0.2.jpg	13.0	0.4106
0391_1_0.08.jpg	**8.82**	0.4234

Legend:
Highest Value
Lowest Value

Table 5.2 Representation of metrics results for 3_5ActivFun_BReLu model.

3.5.3 Choose Best Method

The results showcased in Figure 3.5.2, which employ the ReLU activation function, are significantly superior both visually and in terms of performance metrics when compared to those using LeakyReLU and BReLU. Despite their poorer outcomes, LeakyReLU and

BReLU provide opportunities for extensive testing and customization. These functions suggest that the model struggled to accurately learn from the ground truth images, as indicated by the unusual hues in the output images and the similar numerical results for both functions. Given its stability and effectiveness in dehazing tasks, the ReLU function will be continued throughout the remainder of the project.

3.6 Loss Architectures

3.6.1 Outline

For the loss function, both VGG16 and VGG19 are convolutional neural network architectures proposed by the Visual Geometry Group (VGG) at the University of Oxford[94], and they are great for image dehazing . During neural network training, the primary objective is to minimise the loss function, which quantifies the disparity between predicted outputs and ground truth labels. A lower loss indicates that the model's predictions closely match the true values[69]. Throughout training epochs, the loss ideally diminishes consistently, reflecting the model's enhanced performance and improved predictive capabilities.

A. Until now we have used a simple mean squared error (MSE) loss [69, 70] for image dehazing involving comparing the pixel-wise differences between the predicted dehazed image and the ground truth clear image. This loss function calculates the average squared difference between corresponding pixel intensities, penalising larger discrepancies more heavily, thereby guiding the dehazing algorithm to minimise overall image distortion and produce clearer results. There are a lot of methods that use this as it is very effective and has a very low loss but it is not good at capturing the in depth information during the training. The best results utilising this loss function are visible in Figure 3. 4. representation of 3_4 model.

B. This architecture employs VGG16[66], a convolutional neural network pre-trained on the ImageNet dataset [67], for perceptual loss computation in image dehazing tasks. The VGG16 model is utilised to extract high-level features from both the ground truth image (y_true) and the predicted image (y_pred) [68, 69]. It consists of 16 weight layers, including 13 convolutional layers and 3 fully connected layers. By comparing these feature representations, the perceptual loss function measures the perceptual difference between the two images. This approach is effective for image dehazing because it encourages the generated images to not only match the ground truth at a pixel level but also to capture meaningful high-level features, such as textures, edges, and structures. By incorporating perceptual loss alongside traditional mean squared error (MSE) loss, the model can better preserve important visual characteristics during the dehazing process, leading to more visually appealing results.

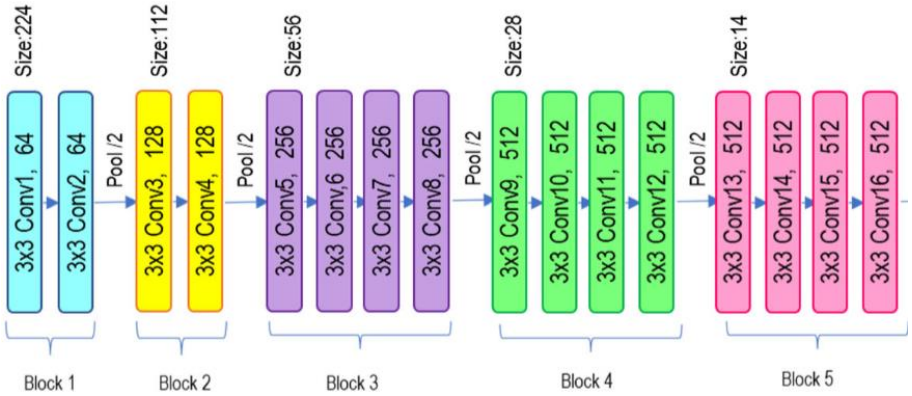


Figure 3.5.1 VGG19 block architecture [72]

C.This VGG19 architecture[66, 71] is a pre-trained convolutional neural network (CNN), that computes perceptual loss for image dehazing. It has 19 weight layers, with 16 convolutional layers and 3 fully connected layers, as shown in Figure 3.5.1. VGG19 is

chosen for its ability to extract high-level features from images effectively. The network is configured to exclude its top layers, which are typically used for classification tasks, and instead focuses on intermediate feature maps. By comparing the feature representations of the true and predicted images at multiple layers, the perceptual loss function measures the discrepancy between their visual characteristics. This approach leverages the insight that perceptual similarity between images can be better captured by comparing feature representations rather than pixel-wise differences alone. But as seen in Figure 3.5.3 it is not able to recreate the correct colour hue as well as MSE so we decide to combine the two.

D. This architecture combines both Mean Squared Error (MSE) loss and perceptual loss, VGG19, for image dehazing tasks, used from DSEU [23]. By comparing the feature representations at a specific layer (in this case, 'block2_conv2') of VGG19, the perceptual loss measures the perceptual difference between the dehaze image prediction and the ground truth. By combining MSE loss, which focuses on pixel-level accuracy, with perceptual loss, which emphasises perceptual similarity, the custom loss function encourages the model to produce dehazed images that are both visually pleasing and faithful to the ground truth. This combination aims to improve the overall quality of dehazed images by addressing both low-level and high-level image features, proven in Figure 3.5.4.

3.6.2 Results and Observations

Visual results:
A. MSE Loss, check Figure 3. 4. for comparison.
B. VGG16 Architecture with MSE Loss

Figure 3.5.2. Representation of 3_6VGG16LossMSE model using VGG16 Architecture. For bigger/clearer images please check Appendix C.3.6.1.a/b.
C. VGG19 Architecture

Figure 3.5.3. Representation of 3_6VGG19Loss model using VGG19 Architecture. For bigger/clearer images please check Appendix C.3.6.2.a/b.

D. VGG19 Architecture with MSE Loss

Figure 3.5.4. Representation of 3_6VGG19LossMSE model using VGG19 Architecture with MSE Loss. For bigger/clearer images please check Appendix C.3.6.3.a/b.

Performance Metrics results:

A. MSE Loss, check Table 4.1 for comparison.

B. VGG16 Architecture with MSE Loss

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	13.34	0.5038
hamburg_beta_0.01.png	12.68	0.485
hamburg_beta_0.02.png	11.94	0.461
0182_0.8_0.16.jpg	12.64	**0.3596**
0100_0.9_0.12.jpg	11.87	0.5178
0001_0.8_0.2.jpg	**13.76**	0.4932
0391_1_0.08.jpg	**10.87**	**0.5513**

Legend:

Highest Value

Lowest Value

Table 6.1 Representation of metrics results for 3_6VGG16LossMSE model.

C. VGG19 Architecture

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	12.74	0.4769
hamburg_beta_0.01.png	12.3	0.4657
hamburg_beta_0.02.png	11.96	0.447
0182_0.8_0.16.jpg	12.02	**0.3228**
0100_0.9_0.12.jpg	11.15	0.4594
0001_0.8_0.2.jpg	**13.78**	0.4674
0391_1_0.08.jpg	**10.4**	**0.4884**

Legend:

Highest Value

Lowest Value

Table 6.2 Representation of metrics results for 3_6VGG19Loss model.

D. VGG19 Architecture with MSE Loss

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	12.81	0.5217
hamburg_beta_0.01.png	11.96	0.4938
hamburg_beta_0.02.png	**11.1**	0.4554
0182_0.8_0.16.jpg	12.25	**0.4033**
0100_0.9_0.12.jpg	12.11	0.5197
0001_0.8_0.2.jpg	**14.5**	0.5192
0391_1_0.08.jpg	11.66	**0.5426**

Legend:

Highest Value

Lowest Value

Table 6.3 Representation of metrics results for 3_6VGG19LossMSE model.

3.6.3 Choose Best Method

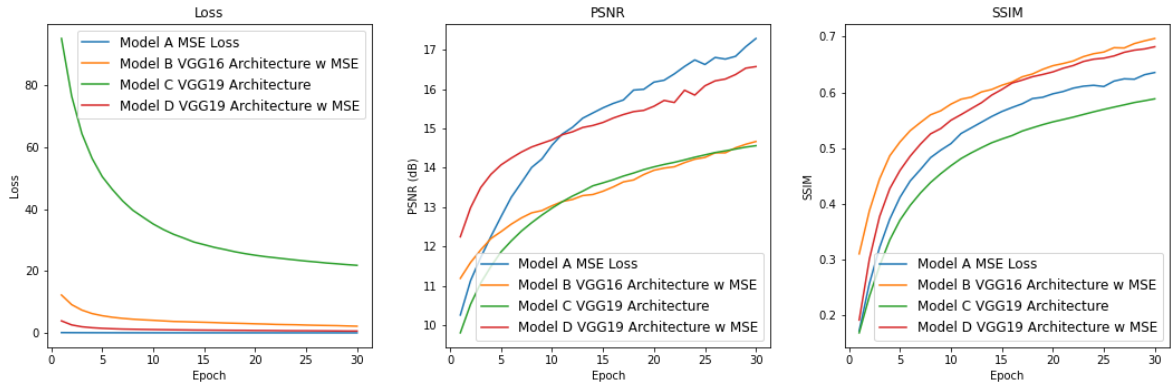


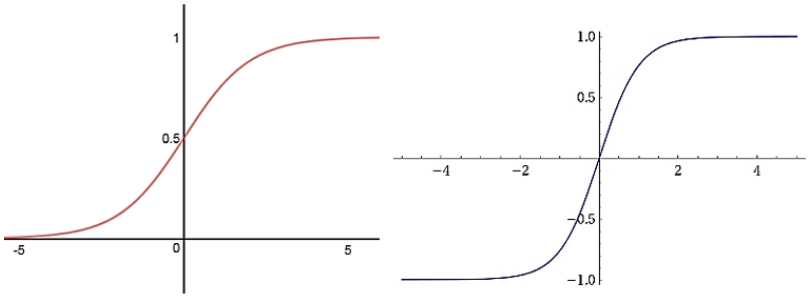
Table 6.4 Comparison of Loss, PSNR, and SSIM Across Different Loss Models.

The four models, A through D, exhibit varying degrees of performance in terms of loss, PSNR, and SSIM across different architectures and loss functions. Model A, utilising MSE loss, demonstrates steady improvement over epochs, although its performance plateaus after around 20 epochs. Model B, employing the VGG16 architecture with MSE loss, shows significant enhancements compared to Model A, suggesting that the VGG16 architecture contributes to better convergence. Model C, utilising the VGG19 architecture, achieves lower loss and higher PSNR and SSIM compared to the previous models, showcasing consistent improvement throughout training. Finally, Model D, which combines the VGG19 architecture with MSE loss, emerges as the top performer, achieving the lowest loss, highest PSNR, and highest SSIM scores overall.

Overall, Model D appears to be the best-performing model, offering the most significant improvements in terms of image quality metrics. From the visual results, we see that the model using VGG19 with MSE loss is also closest to the ground truth image. The additional layers in VGG19 enable it to capture more complex patterns compared to VGG16, leading to better performance on certain tasks. However, this comes at the cost of increased computational complexity and longer training times. The choice between VGG16 and VGG19 often depends on the specific requirements of the task and the available computational resources, but in this case VGG19 outperforms all the other models.

3.7 Output Layer Activation Functions

3.7.1 Outline



(e) Sigmoid Activation Function (f) Tanh Activation Function

Figure 3.7.1 Output Layer Activation Functions[64]

A. In the context of image dehazing, the choice of activation function [73] in the output layer plays a crucial role in determining the characteristics of the final output image.

Employing the sigmoid [74] activation function in the output layer confines the output values to the range [0, 1]. This activation function is commonly used for binary classification tasks or when the output is required to be normalised between 0 and 1. In the context of image dehazing, using sigmoid activation can be useful for tasks where the output needs to be interpreted as probabilities or pixel intensities scaled between the minimum and maximum values of the original image. However, it may not capture the full range of pixel intensities as effectively as tanh, potentially leading to less expressive output images, with papers employing this function [44, 64].

B. Using the tanh [75] activation function in the output layer typically scales the output values to the range [-1, 1]. This can be beneficial for image dehazing tasks because it allows the model to produce pixel values that are well-distributed across the entire dynamic range of the image. The tanh function is symmetric around zero, which means that it can effectively capture both positive and negative changes in pixel intensity. This can help in enhancing the contrast and overall clarity of the dehaze image, with papers employing this function [23].

3.7.2 Results and Observations

Visual results:
A. Results using Sigmoid activation function in Figure 3. 4.
B. Tanh activation function

Figure 7.1. Representation of 3_6VGG19LossMSETanh model. For bigger/clearer images please check Appendix C.3.7.a/b.

Performance Metrics results:
A. Results using Sigmoid activation function in Table. 4.1
B. Tanh activation function

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	9.69	0.1618
hamburg_beta_0.01.png	9.88	0.2158
hamburg_beta_0.02.png	**10.05**	**0.2212**
0182_0.8_0.16.jpg	8.21	**0.1224**
0100_0.9_0.12.jpg	6.89	0.203
0001_0.8_0.2.jpg	6.19	0.1623
0391_1_0.08.jpg	**5.78**	0.1669

Legend:
Highest Value
Lowest Value

Table 7.1 Representation of metrics results for 3_6VGG19LossMSETanh model.

3.7.3 Choose Best Method

The images produced using the sigmoid activation function are much closer to the ground truth image, with significantly sharper definition of edges in buildings, streets, and other details. The performance results also exhibit notable disparities, with the highest PSNR/SSIM values recorded at 14.5/0.54 for the sigmoid activation, while the highest results when using the tanh function are 10.05/0.22. Notably, the SSIM metric appears markedly lower when employing the tanh activation function which is the indicator that the “dehazed image” is extremely different from the original. The images using the Sigmoid activation

function are much closer to the ground truth image and the definition of edges in building, streets and all details is much more defined.

3.8 Training on combined datasets

Until now for convenience all steps have been trained on the Foggy Cityscapes dataset but for a better performance we are going to train the model on another dataset the indoors RESIDE and show each result with the model trained individually on each dataset and on the combined dataset to compare the image quality performance of the results.
A. Results are shown in Figure 3. 4. and table 4.1.
B. Reside dataset training

Figure 3.8.1. Representation of 3_6VGG19LossMSERes model. For bigger/clearer images please check Appendix C.3.8.1.a/b.

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	13.63	0.5983
hamburg_beta_0.01.png	11.53	0.5576
hamburg_beta_0.02.png	**9.57**	0.5068
0182_0.8_0.16.jpg	14.06	**0.4612**
0100_0.9_0.12.jpg	15.35	0.6381
0001_0.8_0.2.jpg	**15.64**	0.5786
0391_1_0.08.jpg	15.61	**0.6748**

Legend:
Highest Value
Lowest Value

Table 8.1 Representation of metrics results for 3_6VGG19LossMSERes model.
C. For combined dataset training

Figure 3.8.2. Representation of 3_6VGG19LossMSEResFog model. For bigger/clearer images please check Appendix C.3.8.2.a/b.

Image Name	PSNR	SSIM
hamburg_beta_0.005.png	13.32	0.5335
hamburg_beta_0.01.png	11.82	0.5085
hamburg_beta_0.02.png	**10.15**	0.4763
0182_0.8_0.16.jpg	13.42	**0.4286**
0100_0.9_0.12.jpg	14.21	0.5787
0001_0.8_0.2.jpg	**15.43**	0.5689
0391_1_0.08.jpg	13.64	**0.6206**

Legend:
Highest Value
Lowest Value

Table 8.2 Representation of metrics results for 3_6VGG19LossMSEResFog model.

The final results, Figure 3.8.2 and Table 8.2, as expected, demonstrate that utilising the combined dataset shows superior outcomes for both types of images tested, along with enhanced performance. The combined dataset offers greater diversity and quantity of images, aiding the model in training on a more extensive dataset. While this increases training time, it significantly benefits the model's output without affecting the processing time for dehazing new images. The visual results are notably clear, with hues closely resembling those of the ground truth image, and the dehazing process being visibly effective. Performance metrics also indicate significant improvement compared to previous attempts, with consistency observed across images with different beta values. The highest recorded PSNR/SSIM values of 15.43/0.62 exceed average expectations, showcasing the model's strong performance.In Appendix C.4, additional images are provided to assess the effectiveness of the dehazing process and evaluate the performance of the model.

Chapter 4
Discussion

4.1 Conclusion - Final U-Net Architecture

4.1.1 Outline of final architecture

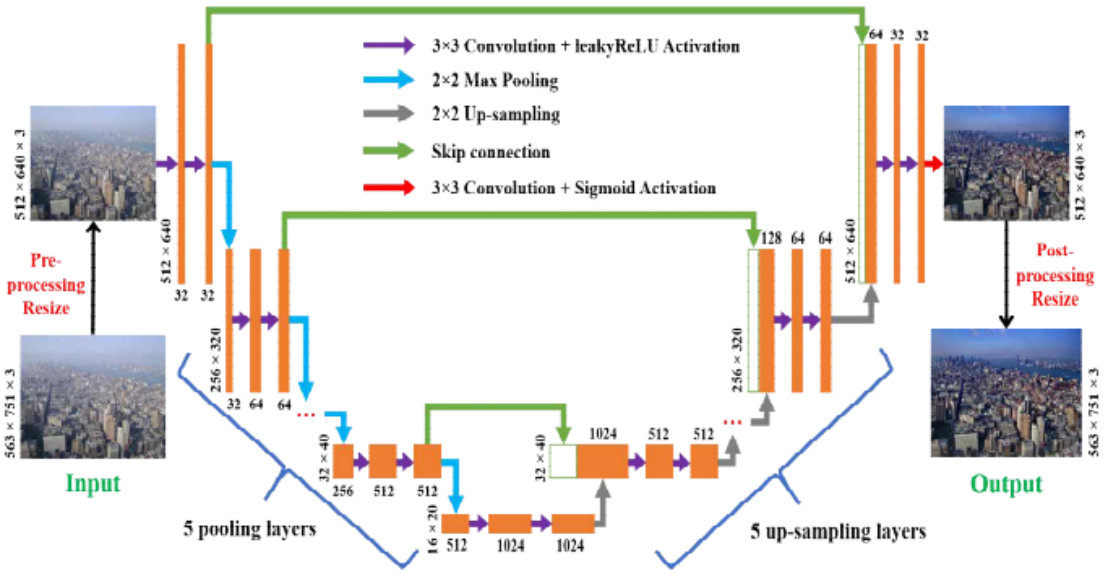


Figure 4.1 A close representation of the final five layer improved U-Net Architecture, but instead of the LeakyRelu activation, our method uses the Relu activation function. [78]

The final architecture of the U-Net model architecture incorporates an encoder-decoder structure where the encoder section utilises sequential 3x3 convolutions, scaling from 64 to 256 filters, paired with ReLU activations and Max-Pooling to compress spatial dimensions. This is followed by a decoder that employs 2x2 transposed convolutions to progressively upscale the feature maps. These upscaled maps are then merged with corresponding layers from the encoder through skip connections, which help preserve local image features and enhance detail in the reconstructed output.

In terms of advanced features, the model includes Batch Normalisation and Kernel Initialization to stabilise training and mitigate issues related to gradient propagation. Additional convolutional layers with 512 and 1024 filters allow the network to capture more complex image patterns, essential for effective image dehazing. Squeeze and Excitation blocks adjust channel-wise feature responses, focusing on the most salient features, while Bottleneck and Dropout layers work to prevent overfitting by promoting generalisation. The model employs two loss functions: Mean Squared Error (MSE) for pixel-wise accuracy and a perceptual loss using VGG19 architecture to ensure the outputs not only are accurate but also possess qualities that align closely with human visual perception, thereby enhancing the overall effectiveness in a broad range of dehazing tasks. While the model is trained on two varied datasets, Foggy Cityscapes and Indoors RESIDE.

4.4.2 Final Observations

Final best performing results of the architecture above can be seen in Figure 3.8.2 and Table 8.2. For the purpose of more testing additional dehazed images are added in Appendix C.

Method	DCP [4]	BCCR [46]	GRM [47]	CAP [48]	NLD [49]	DehazeNet [3]	MSCNN [8]	AOD-Net [9]	GMAN [50]	DRHNet[51]	Our Method
PSNR	18.870000	17.870000	20.440000	21.310000	18.530000	22.660000	20.010000	21.010000	27.940000	31.390000	15.430000
SSIM	0.794000	0.770000	0.823000	0.824000	0.702000	0.833000	0.791000	0.837000	0.897000	0.974000	0.620000

Table 4.1. Results on Outdoors RESIDE. Industry, professional papers. Comparison to Our Method

Method	DCP[4]	DEA-Net[85]	DehazeFormer[85]	DFF-Net[86]	AOD-Net[9]	MixDehazeNet	PriorNet[84]	Our Method
PSNR	14.099000	12.956600	10.867700	11.865500	15.125400	13.001200	17.275600	15.430000
SSIM	0.758800	0.732600	0.635500	0.738700	0.752500	0.733100	0.832900	0.620000

Table 4.2.Results on Haze4K. Simpler papers. Comparison to Our Method

The performance of "Our Method" when compared to other dehazing methods across two datasets shows mixed results. In the first dataset, "Our Method" records a PSNR (Peak Signal-to-Noise Ratio) of 15.43 and an SSIM (Structural Similarity Index) of 0.62. These figures are significantly lower than the best performing methods in the dataset, such as GMAN and DRHNet, which achieve PSNR values of 27.94 and 31.39, respectively, and SSIM scores of 0.897 and 0.974. This indicates a notable gap in performance, with "Our Method" struggling to match the effectiveness of these advanced techniques in restoring image quality.

In the second dataset, "Our Method" again records a PSNR of 15.43 and an SSIM of 0.62, placing it in the middle tier among the compared methods. It outperforms several methods such as DehazeFormer and DFF-Net, which show lower PSNRs of 10.8677 and 11.8655, and SSIMs of 0.6355 and 0.7387, respectively. However, it falls short of reaching the performance of PriorNet, which leads with a PSNR of 17.2756 and an SSIM of 0.8329. This variability in performance across different datasets suggests that while "Our Method" is competitive, it may not consistently deliver the highest dehazing quality compared to more specialised or advanced approaches.

Our proposed method has achieved its objectives and performs above average. Compared to simpler methods with similar architecture and structure, our model significantly outperforms many of them. By leveraging these properties, we have successfully met our goals. The U-Net architecture, utilised in our approach, is well-suited for the task of image dehazing. It produces clearer images with well-preserved details and structures, which are crucial for subsequent image analysis or enhancing visual quality for human viewing. This confirms the effectiveness of our method in addressing the challenges of image dehazing.

4.2 Limitations

4.2.1 Hardware Costs

The model was trained using Google Collab Plus with hardware acceleration on a cloud based V100 GPU. Due to the complexity of the model, personal devices with limited RAM were insufficient for training. As the dataset size increases, training becomes more challenging and time-consuming, necessitating a powerful and high-quality device. However, using this model in automated cars should pose no issues.

4.2.2 Dataset Size

Despite having 5939 pairs of images(9 GB), the dataset size is relatively small compared to its potential(40 GB). Memory limitations and hardware capabilities constrained the amount of data used for training. If the complete datasets of both Foggy Cityscapes and Indoor RESIDE were utilised, the model's performance would improve significantly. Unfortunately, this is constrained by space and memory limitations.

4.2.3 Software Performance

Compared to professional papers, the model may still lack high-performance results, but it remains competent in producing fast and clear images. As mentioned earlier, there are hardware, physical, and space limitations that could enhance training the model. Additionally, accessing real-life data from autonomous cars to assess the model's performance and testing image object detection on both dehazed and hazy images would provide valuable insights into the software's effectiveness.

4.3 Ideas for future work

4.3.1 Improving the Model Architecture

There is considerable potential for enhancing the project by implementing custom loss and activation functions. While using standardised functions aids in understanding CNNs, custom functions offer the flexibility to tailor the model to specific needs, especially in tasks like dehazing where image quality and fog intensity vary significantly. Custom functions allow for fine-tuning according to specific requirements. In the context of dehazing, where factors like image quality and fog intensity vary widely, custom functions offer the adaptability needed to address these nuances effectively. Drawing insights from existing papers that have demonstrated the efficacy of customised functions, many have demonstrated the effectiveness of customised functions[51, 63], further experimentation and refinement of the model architecture can lead to breakthroughs in dehazing performance. Papers that add their own additional implementations [76, 77, 78].

4.3.2 Increasing Dataset

Expanding the dataset size is crucial for improving the model's ability to generalise and perform well across diverse scenarios. The current dataset is constrained by memory limitations and hardware capabilities, underscoring the need for augmentation, as mentioned above the size of the current dataset. By incorporating a larger and more diverse dataset, the model can better capture the intricacies of hazy scenes and learn robust representations that generalise well. This expanded dataset can encompass a wide range of environmental conditions, including varying levels of haze intensity, lighting conditions, and scene complexities, thereby enabling the model to adapt and perform optimally in real-world settings.

4.3.3 Testing on Autonomous Cars

Assessing the performance of the dehazing software in real-world scenarios, particularly on autonomous cars, is essential for validating its practical utility. Applying the software to images captured by autonomous vehicles allows for an evaluation of its impact on critical tasks such as image segmentation and object detection. By analysing improvements in tasks like human detection and traffic light recognition, insights can be gained into the software's effectiveness in enhancing overall situational awareness and safety. This real-world testing, although ambitious, provides valuable feedback for refining the software and optimising its performance for deployment in autonomous driving systems [79].

4.3.4 Expanding on Video Dehazing

Enabling real-time video dehazing [80, 81] capabilities presents an exciting opportunity to extend the application of the algorithm to dynamic environments. By seamlessly processing continuous video recordings from moving vehicles, the algorithm can enhance visibility and clarity in real-time, enhancing the safety and performance of autonomous driving systems. Overcoming the challenges associated with real-time processing, such as optimising computational efficiency and minimising latency, requires leveraging insights gained from previous projects and adopting innovative approaches. The development of a practical and effective solution for video dehazing opens up new avenues for enhancing visual perception and situational awareness in autonomous vehicles, ultimately contributing to safer and more efficient transportation systems.

List of References

[1] Song, R., Wetherall, R.J., Maskell, S., and Ralph, J.F. 2020. Weather Effects on Obstacle Detection for Autonomous Cars. Advances in Vehicle Engineering. 2(1), pp. 5-6.

[2] Rasshofer, R.H., and Gresser, K. 2005. Automotive radar and lidar systems for next generation driver assistance functions. Advances in Radio Science. 3(B.4), pp. 205-209.

[3] Cai, B., Xu, X., Jia, K., Qing, C., and Tao, D. 2016. Dehazenet: An end-to-end system for single image haze removal. IEEE Transactions on Image Processing. 25, pp. 5187–5198.

[4] He, K., Sun, J., and Tang, X. 2011. Single image haze removal using dark channel prior. IEEE Transactions on Pattern Analysis and Machine Intelligence. 33(12), pp. 2341–2353.

[5] McCartney, E.J. 1976. Optics of the atmosphere: scattering by molecules and particles. New York: John Wiley and Sons, Inc.

[6] Nayar, S.K., and Narasimhan, S.G. 1999. Vision in bad weather. Proceedings of the IEEE International Conference on Computer Vision. 2, pp. 820–827.

[7] Narasimhan, S.G., and Nayar, S.K. 2003. Contrast restoration of weather degraded images. IEEE Transactions on Pattern Analysis and Machine Intelligence. 25(6), pp. 713–724.

[8] Ren, W., Liu, S., Zhang, H., Pan, J., Cao, X., and Yang, M-H. 2016. Single image dehazing via multi-scale convolutional neural networks. Proceedings of the European Conference on Computer Vision (ECCV). Springer, pp. 154–169.

[9] Li, B., Peng, X., Wang, Z., Xu, J., and Feng, D. 2017. Aod-net: All-in-one dehazing network. Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 4770–4778.

[10] Ren, W., Ma, L., Zhang, J., Pan, J., Cao, X., Liu, W., and Yang, M-H. 2018. Gated fusion network for single image dehazing. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3253–3261.

[11] Liu, Z., Xiao, B., Alrabeiah, M., Wang, K., and Chen, J. 2019. Single image dehazing with a generic model-agnostic convolutional neural network. IEEE Signal Processing Letters. 26(6), pp. 833–837.

[12] Li, R., Pan, J., Li, Z., and Tang, J. 2018. Single image dehazing via conditional generative adversarial network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8202–8211.

[13] Qu, Y., Chen, Y., Huang, J., and Xie, Y. 2019. Enhanced pix2pix dehazing network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8160–8168.

[14] Shahriar, N. 2023. What is Convolutional Neural Network — CNN (Deep Learning). [Online]. [Accessed 11/04/2024]. Available from: https://nafizshahriar.medium.com/what-is-convolutional-neural-network-cnn-deep-learning-b3921bdd82d5.

[15] Sivakumar, D. 2023. Putting Encoder - Decoder Together. [Online]. [Accessed 11/04/2024]. Available from: https://www.scaler.com/topics/nlp/encoder-and-decoder/.

[16] Leal-Taixé, L., Prof. 2021. ADL4CV:DV - Encoder-decoder. [Online]. [Accessed 04/03/2024]. Available from: https://www.youtube.com/watch?v=aV1jWHIT2iM.

[17] Rosebrock, A. 2020. Putting Encoder - Decoder Together. [Online]. [Accessed 04/03/2024]. Available from: https://pyimagesearch.com/2020/02/24/denoising-autoencoders-with-keras-tensorflow-and-deep-learning/.

[18] Chollet, F. 2021. Deep Learning with Python 2nd Edition. pp. 393-411.

[20] Li, B., Ren, W., Fu, D., Tao, D., Feng, D., Zeng, W., and Wang, Z. 2018. Benchmarking single-image dehazing and beyond. IEEE Transactions on Image Processing. 28(1), pp. 492-505.

[21] Li, B., Peng, X., Wang, Z., Xu, J., and Feng, D. 2017. Aod-net: All-in-one dehazing network. In Proceedings of the IEEE International Conference on Computer Vision, pp. 4770-4778.

[22] Li, R., Pan, J., Li, Z., and Tang, J. 2018. Single image dehazing via conditional generative adversarial network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8202-8211.

[23] Lee, Y.-W., Wong, L.-K., and See, J. 2020. Image Dehazing With Contextualized Attentive U-NET. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Abu Dhabi, United Arab Emirates, pp. 1068-1072.

[24] Abadi, M. et al. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. [Online]. [Accessed 19/04/2024]. Available from: https://www.tensorflow.org/.

[25] Paszke, A. et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Advances in Neural Information Processing Systems 32. Curran Associates, Inc., pp. 8024–8035. [Online]. [Accessed 19/04/2024]. Available from: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[26] Final Year Project-sc21samg. [Online]. Available at: https://github.com/uol-feps-soc-comp3931-2324-classroom/final-year-project-sc21samg.

[27] Jin, Y., Yan, W., Yang, W., and Tan, R.T. 2022. Structure Representation Network and Uncertainty Feedback Learning for Dense Non-Uniform Fog Removal. In: Proceedings of the Asian Conference on Computer Vision (ACCV), December 2022, pp.2041-2058.

[28] Chollet, F. 2021. Deep Learning with Python 2nd Edition. pp. 68-71.

[29] Chollet, F. et al. 2015. Keras. [Online]. [Accessed 19/04/2024]. Available from: https://github.com/fchollet/keras.

[30] Hunter, J.D. 2007. Matplotlib: A 2D graphics environment. Computing in Science & Engineering. 9(3), pp.90-95. DOI: 10.1109/MCSE.2007.55.

[31] Culjak, I., Abram, D., Pribanic, T., Dzapo, H., and Cifrek, M. 2012. A brief introduction to OpenCV. In: Proceedings of the 35th International Convention MIPRO, May 2012, pp. 1725–1730.

[32] Sakaridis, C., Dai, D., and Van Gool, L. 2018. Semantic Foggy Scene Understanding with Synthetic Data. International Journal of Computer Vision. 126(9), pp. 973-992. [Online]. [Accessed 02/11/2023]. Available at: https://doi.org/10.1007/s11263-018-1072-8; https://people.ee.ethz.ch/~csakarid/SFSU_synthetic/; https://www.cityscapes-dataset.com/downloads/.

[33] 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. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [Online]. [Accessed 02/11/2023]. Available at: https://www.cityscapes-dataset.com/citation/.

[34] 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), pp. 492-505. [Online]. [Accessed 02/11/2023]. Available at: https://sites.google.com/view/reside-dehaze-datasets/reside-standard?authuser=3D0.

[35] Ronneberger, O., Fischer, P., and Brox, T. 2015. U-net: Convolutional networks for biomedical image segmentation. pp. 234–241.

[36] Siddique, N., Paheding, S., Elkin, C.P., and Devabhaktuni, V. 2021. U-Net and Its Variants for Medical Image Segmentation: A Review of Theory and Applications. IEEE Access. 9, pp. 82031-82057. [Online]. [Accessed 20/04/2024]. Available at: https://doi.org/10.1109/ACCESS.2021.3086020.

[37] Ronneberger, O., Fischer, P., and Brox, T. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. Springer International Publishing. [Online]. [Accessed 27 April 2024] Available at: https://doi.org/10.1007/978-3-319-24574-4_28.

[38] Half-UNet: A Simplified U-Net Architecture for Medical Image Segmentation. 2022. [pdf]. [Online]. [Accessed 27 April 2024]. Available at: https://www.researchgate.net/publication/361186968_Half-UNet_A_Simplified_U-Net_Architecture_for_Medical_Image_Segmentation.

[39] Weng, Y., Zhou, T., Li, Y., and Qiu, X. 2019. NAS-UNet: Neural architecture search for medical image segmentation. IEEE Access. 7, pp. 44247-44257. [Online]. [Accessed 27 April 2024]. Available at: https://doi.org/10.1109/ACCESS.2019.2941278.

[40] Ronneberger, O., Fischer, P., and Brox, T. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. [eprint] arXiv:1505.04597. [Online]. [Accessed 27 April 2024]. Available at: https://arxiv.org/abs/1505.04597.

[41] Dhumak, S. 2021. Image Segmentation with U-Net. [Online]. [Accessed 30 April 2024]. GitHub. Available at: https://github.com/sushantdhumak/Image-Segmentation-with-U-Net/tree/main?tab=readme.

[42] Lee, Y.-W., Wong, L.-K., and See, J. 2020. Image Dehazing With Contextualized Attentive U-NET. In: 2020 IEEE International Conference on Image Processing (ICIP), Abu Dhabi, United Arab Emirates. IEEE, pp.1068-1072. [Online]. [Accessed 27 April 2024]. Available at: https://doi.org/10.1109/ICIP40778.2020.9190725.

[43] Ge, W., Lin, Y., Wang, Z., Wang, G., and Tan, S. 2021. An Improved U-Net Architecture for Image Dehazing. IEICE TRANSACTIONS on Information and Systems. E104-D(12), pp.2218-2225. [Online]. [Accessed 27 April 2024]. Available at: https://doi.org/10.1587/transinf.2021EDP7043.

[44] Wang, Z., Jia, J., Lyu, P., and Min, J. 2023. Efficient Dehazing with Recursive Gated Convolution in U-Net: A Novel Approach for Image Dehazing. J. Imaging. 9(9), 183. [Online]. [Accessed 27 April 2024]. Available at: https://doi.org/10.3390/jimaging9090183.

[45] Goodfellow, I., Bengio, Y., and Courville, A. 2016. Deep Learning. Cambridge: MIT Press, pp.179-184.

[46] Meng, G., Wang, Y., Duan, J., Xiang, S., and Pan, C. 2013. Efficient image dehazing with boundary constraint and contextual regularization. In: Proceedings of the IEEE International Conference on Computer Vision, Dec. 2013, pp. 617–624.

[47] Chen, C., Do, N., and Wang, J. 2016. Robust image and video dehazing with visual artifact suppression via gradient residual minimization. In: Proceedings of the European Conference on Computer Vision, 2016, pp. 576–591.

[48] Zhu, Q., Mai, J., and Shao, L. 2015. A fast single image haze removal algorithm using color attenuation prior. IEEE Transactions on Image Processing. 24(11), pp. 3522–3533, Nov. 2015.

[49] Berman, D. and Avidan, S. 2016. Non-local image dehazing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Jun. 2016, pp. 1674–1682.

[50] Liu, Z., Xiao, B., Alrabeiah, M., Wang, K., and Chen, J. 2019. Single image dehazing with a generic model-agnostic convolutional neural network. IEEE Signal Processing Letters. 26(6), pp. 833–837, Jun. 2019.

[51] Wang, C., Li, Z., Wu, J., Fan, H., Xiao, G., and Zhang, H. 2020. Deep Residual Haze Network for Image Dehazing and Deraining. IEEE Access. 8, pp. 9488-9500, 2020.

[52] Wang, Z., Bovik, A., Sheikh, H., and Simoncelli, E. 2004. Image quality assessment: From error visibility to structural similarity. IEEE Transactions on Image Processing. 13(4), pp. 600–612.

[53] Hu, J., Shen, L., and Sun, G. 2018. Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7132–7141

[54] Towards Data Science. 2017. Review: SENet (Squeeze-and-Excitation Network) Winner of ILSVRC 2017 Image Classification. [Online]. [Accessed 30 April 2024]. Available from: https://towardsdatascience.com/review-senet-squeeze-and-excitation-network-winner-of-ilsvrc-2017-image-classification-a887b98b2883.

[55] Kirouane, A. Review: Squeeze-and-Excitation Networks (SENet). [Online]. [Accessed 30 April 2024]. Available from: https://www.linkedin.com/pulse/squeeze-and-excitation-networks-senet-ayoub-kirouane.

[56] Towards Data Science. 2017. Activation Functions in Neural Networks. [Online]. [Accessed 30 April 2024]. Available from: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6.

[57] Haouassi, S., and Wu, D. 2020. Dehazing Based on (CMTnet) Cascaded Multi-scale Convolutional Neural Networks and Efficient Light Estimation Algorithm. Applied Sciences. 10(3), 1190. https://doi.org/10.3390/app10031190.

[58] Ren, W., Liu, S., Zhang, H., Pan, J., Cao, X., and Yang, M.H. 2016. Single Image Dehazing via Multi-scale Convolutional Neural Networks. In: Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016.

[59] Li, C., Guo, J., Porikli, F., Fu, H., and Pang, Y. 2018. A Cascaded Convolutional Neural Network for Single Image Dehazing. IEEE Access. 6, pp. 24877–24887.

[60] Hinton, G.E., and Salakhutdinov, R.R. 2006. Reducing the Dimensionality of Data with Neural Networks. Science. 313, pp. 504–507.

[61] Nair, V., and Hinton, G.E. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010, pp. 807–814.

[62] Mehralian, M., and Karasfi, B. 2018. RDCGAN: Unsupervised Representation Learning With Regularized Deep Convolutional Generative Adversarial Networks. In: Proceedings of the 9th Conference on Artificial Intelligence and Robotics and 2nd Asia-Pacific International Symposium, Kish Island, Iran, 2018, pp. 31-38. doi: 10.1109/AIAR.2018.8769811.

[63] Bhola, A., Sharma, T., and Verma, N.K. 2021. DCNet: Dark Channel Network for single-image dehazing.

[64] Oppermann, A. 2022. Activation Functions in Deep Learning: Sigmoid, Tanh, ReLU. [Online]. [Accessed 30 April 2024]. Available from: https://artemoppermann.com/activation-functions-in-deep-learning-sigmoid-tanh-relu/.

[65] Zhixuhao. 2019. Implementation of deep learning framework -- Unet, using Keras. [Online]. [Accessed 30 April 2024]. Available from: https://github.com/zhixuhao/unet/tree/master.

[66] Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. [eprint] arXiv:1409.1556. Available from: https://arxiv.org/abs/1409.1556.

[67] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. 2015. Imagenet large scale visual recognition challenge. IJCV. 115(3), pp. 211–252.

[68] Cheng, D., and Lam, E.Y. Transfer Learning U-Net Deep Learning for Lung Ultrasound Segmentation. Department of Bioengineering, Imperial College London and Department of Electrical and Electronic Engineering, The University of Hong Kong.

[68] John, D., and Zhang, C. 2022. An attention-based U-Net for detecting deforestation within satellite sensor imagery. Journal of Applied Geomatics. [Online]. [Accessed 30 April 2024]. Available from: https://doi.org/10.1016/j.jag.2022.102685.

[69] PyTorch. 2023. MSE Loss. [Online]. [Accessed 14 April 2024]. Available from: https://pytorch.org/docs/stable/generated/torch.nn.MSELoss.html.

[70] Kato, S., and Hotta, K. 2021. MSE Loss with Outlying Label for Imbalanced Classification. CoRR. vol. abs/2107.02393. [Online]. [Accessed 14 April 2024]. Available from: https://arxiv.org/abs/2107.02393.

[71] Nguyen, T.-H., Nguyen, T.-N. and Ngo, B.-V. 2022. A VGG-19 Model with Transfer Learning and Image Segmentation for Classification of Tomato Leaf Disease. AgriEngineering. 4(4), pp. 871-887. [Online]. [Accessed 03 May 2024]. Available at: https://doi.org/10.3390/agriengineering4040056.

[72] Khattar, A. and Quadri, S.M.K. 2022. Generalization of convolutional network to domain adaptation network for classification of disaster images on Twitter. Multimedia Tools and Applications. 81(4). doi: 10.1007/s11042-022-12869-1.

[73] Dubey, S.R., Singh, S.K. and Chaudhuri, B.B. 2022. Activation functions in deep learning: A comprehensive survey and benchmark. Neurocomputing. 503, pp. 92-108. doi: 10.1016/j.neucom.2022.06.111. [Online]. [Accessed 03 May 2024]. Available at: https://www.sciencedirect.com/science/article/pii/S0925231222008426 [Accessed date].

[74] Chakraborty, K. and Hassanien, A.A. 2019. Sentiment Analysis on a Set of Movie Reviews Using Deep Learning Techniques. In Social Network Analytics.

[75] Godin, F., Degrave, J., Dambre, J. and De Neve, W. 2018. Dual Rectified Linear Units (DReLUs): A replacement for tanh activation functions in Quasi-Recurrent Neural Networks. Pattern Recognition Letters. 116, pp. 8-14. [Online]. [Accessed 03 May 2024]. Available at: https://doi.org/10.1016/j.patrec.2018.09.006.

[76] Zhang, S., Tao, Z., & Lin, S. 2024. WaveletFormerNet: A Transformer-based Wavelet Network for Real-world Non-homogeneous and Dense Fog Removal. Submitted on 9 Jan 2024.

[77] Yang, H.-H., Yang, C.-H. H., & Tsai, Y.-C. J. 2020. Y-Net: Multi-Scale Feature Aggregation Network With Wavelet Structure Similarity Loss Function For Single Image Dehazing. In ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, pp. 2628-2632. doi: 10.1109/ICASSP40776.2020.9053920.

[78] Miao, Y., Zhao, X., & Kan, J. 2022. An end-to-end single image dehazing network based on U-net. Signal Image and Video Processing. 16(3). doi: 10.1007/s11760-021-02129-4.

[79] Singh, S. 2022. [Comment on the article "Image Segmentation in Self-Driving Cars"]. [Online]. [Accessed 04 May 2024]. Retrieved from https://www.labellerr.com/blog/image-segmentation-in-self-driving-cars/.

[80] Juneja, A., Kumar, V., & Singla, S.K. 2023. Aethra-net: Single image and video dehazing using autoencoder. Journal of Visual Communication and Image Representation. 94, 103855. [Online]. [Accessed 03 May 2024]. Available at: https://doi.org/10.1016/j.jvcir.2023.103855.

[81] Xu, J., Hu, X., Zhu, L., Dou, Q., Dai, J., Qiao, Y., & Heng, P. 2023. Video Dehazing via a Multi-Range Temporal Alignment Network with Physical Prior. arXiv preprint arXiv:2303.xxxxx.

[82] Sutskever, I., Hinton, G., Krizhevsky, A., and Salakhutdinov, R.R. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting.

[83] Liu, X., Ma, Y., Shi, Z., Chen, J. 2019. Griddehazenet: Attention-based multi-scale network for image dehazing. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 7314–7323.

[84] Chen, Y., Wen, Z., Wang, C., Gong, L., & Yi, Z. 2024. PriorNet: A Novel Lightweight Network with Multidimensional Interactive Attention for Efficient Image Dehazing. arXiv preprint arXiv:2404.xxxxx.

[85] Chen, Z., He, Z., and Lu, Z.-M. 2024. DEA-net: Single image dehazing based on detail-enhanced convolution and content-guided attention. IEEE Transactions on Image Processing.

[85] Song, Y., He, Z., Qian, H., & Du, X. 2022. Vision Transformers for Single Image Dehazing. arXiv preprint arXiv:2204.xxxxx.

[86] Zhu, M., Wan, S., Jin, P., & Zhang, P. 2022. DFFNet: Dynamic Feature Fusion Network for Weakly Supervised Object Detection in Remote Sensing Images. In 2022 IEEE International Conference on Big Data (Big Data), Osaka, Japan, pp. 1409-1414. doi: 10.1109/BigData55660.2022.10020414.

[87] IBM. 2022. Computer Vision. [Online]. [Accessed 03 May 2024]. Available at: https://www.ibm.com/topics/computer-vision.

[88] Shao, Y., Li, L., Ren, W., Gao, C., & Sang, N. 2020. Domain Adaptation for Image Dehazing. National Key Laboratory of Science and Technology on Multispectral Information Processing, School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan, China, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China.

[89] Bian, Y., Zhang, E., Wang, J., Xie, R., & Jiang, S. 2022. Swin Transformer UNet for Very High Resolution Image Dehazing. School of Computer Science and Engineering, Faculty of Innovation Technology, Macau University of Science and Technology, Avenida Wai Long, Taipa, Macau SAR, China; School of Mechanical Engineering, Zhuhai College of Science and Technology, Jinwan, Zhuhai 519040, China. Received July 29, 2022; accepted September 22, 2022.

[90] Badarinarayanan, V., Kendall, A., & Cipolla, R. 2016. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. arXiv preprint arXiv:1511.00561v3.

[91] Long, J., Shelhamer, E., & Darrell, T. 2015. Fully Convolutional Networks for Semantic Segmentation. arXiv preprint arXiv:1411.4038v2.

[92] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. 2017. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. arXiv preprint arXiv:1606.00915v2.

[93] Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. 2017. Pyramid Scene Parsing Network. arXiv preprint arXiv:1612.01105v2.

[94] Bilen, H., & Vedaldi, A. 2016. Weakly supervised deep detection networks. In IEEE Conference on Computer Vision and Pattern Recognition.

Appendix A
Self-appraisal

A.1 Critical self-evaluation

The goal of the project was to develop a fast performing, pretrained CNN model for single image dehazing. Throughout the project, various methods were tested and the most effective one was selected. The dataset used was extensive and diverse, which is a notable improvement over many studies that rely on a single dataset for convenience. The system was evaluated continuously, and the final version was selected for its superior visual and technical performance, highlighting the extensive research and effort invested in this project. Although the results may not be comparable to professional standards, and there is room for improvement, the project's objectives were successfully met. Given my newness to the computer vision field, the learning curve was a challenge, and substantial time was devoted to understanding foundational concepts.

The decision to develop a dehazing algorithm was inspired by the discovery of the large and well structured Foggy Cityscapes dataset [32], based on the well known Cityscapes dataset [33]. Being completely new to computer vision and working with CNNs, it took significant time to determine a starting point. I decided on using the U-Net architecture in late February, which limited the time available for testing various methods and experimenting and developing custom loss or activation functions. However, the learning process was essential to understand and apply concepts from multiple other research papers.

Initially, I ambitiously planned to develop software that could create fog for images, dehaze them, and implement video dehazing. Each of these tasks could constitute an individual project. After reconsideration, I chose image dehazing as the primary task, as adding fog to images was relatively straightforward and would not have involved extensive research, whereas video dehazing would have required a solid understanding of image dehazing and more complex data, exceeding my available resources. Image dehazing struck a balance, offering substantial research opportunities, as evidenced by numerous doctoral studies. Fortunately, with guidance from my supervisor, I focused on image dehazing early in the project timeline.

A.2 Personal reflection and lessons learned

By the end of the project, the main objective of developing a single image dehazing software was successfully achieved. I am satisfied with the results and the progress I have made. Looking back, the amount of new information I have learned and how much I have evolved compared to eight months ago is significant, and I am very proud of myself. Since starting university, I have been interested in machine learning and AI, and I wanted my dissertation to explore something new and different that I hadn’t studied during my university years. Although it was a significant challenge and everything was full of new concepts, I progressed academically and enjoyed learning new things every day while working with state-of-the-art deep learning methods.

In my education I have never encountered the task of genuinely improving an algorithm's architecture and testing various components within its structure. In university courseworks, the focus is often on applying learned algorithms and sticking close to a given structure. However, this project allowed me to deconstruct and analyse an algorithm's architecture in depth. I have learned to appreciate the diverse and global contributions from researchers around the world, and I believe I have become proficient at quickly sifting

through papers to extract the information I need, proven by the large amount of references studied and used in this project.

Underestimating the importance of background research at the beginning of the project is a common mistake among students, including myself. Extensive background research, involving countless papers, YouTube videos, and Stack Overflow questions, has taught me a great deal and significantly aided this project. Without these resources, I would not have been able to advance my understanding of neural networks as far.

In conclusion, spending eight months on this project has demonstrated the importance of time management skills and maintaining a healthy lifestyle for a productive mindset. Even when there were weeks without noticeable improvements in my model, it was crucial to stay engaged, consider different perspectives and approaches, and sometimes even start anew. In the initial stages, I had to change approaches and architectures to find the best suited one, with a high chance of improvement and testing given the time and resources available.

For future work, I take the biggest lesson as maintaining a positive mindset and consistency. Even if something doesn't work initially, there will be a time when it does, and things will improve; this could take an hour, a day, a week, or even a month. Evolution is attainable no matter the circumstances.

A.3 Legal, social, ethical and professional issues

A.3.1 Legal issues

This project was developed through extensive research and reference to multiple academic papers, all of which have been cited according to the Harvard-Leeds style. Some papers required special access, which was personally requested, with the intent of academic research. Sources such as GitHub repositories have also been acknowledged for parts of the code used or for the methods that inspired this work. It is crucial to mention the two datasets employed referenced multiple times, Foggy Cityscapes and RESIDE, which contain both clear and hazy images. Without these datasets, the project would not have been possible. The organisation and variety of these datasets, as well as the specific subsets used, are detailed within the project documentation.

There were no papers with access restrictions used in this project; hence, no legal obligations have been violated as all sources were either open access or requested for academic and research purposes. Furthermore, the open-source library Keras, which is released under the permissive Apache License 2.0, was utilised. All libraries used are accessible to anyone and intended for benevolent purposes.

A.3.2 Social issues

This project aims to enhance image detection in autonomous vehicles and surveillance cameras, and is designed to be beneficial rather than harmful to society. However, given the prevalent concerns about artificial intelligence and robotics, this project could be seen as contributing to the advancement of AI vehicles. For this reason, considerable emphasis is placed on rigorous testing and complete transparency of results in Chapters 2 and 3. Should this software be implemented in real-world scenarios, it would require even more thorough testing on actual vehicles under real-life conditions. Chapter 1 discusses the hazards of driving in foggy weather and highlights how software like this is crucial for object segmentation and safety in such environments.

A.3.3 Ethical issues

Regarding ethical concerns, no images of people's faces are included in the training or testing datasets from the Foggy Cityscapes. Any potentially identifiable individuals have their faces blurred in the provided datasets.

Additionally, in terms of privacy, this software processes hazy images to produce dehazed versions without learning or storing any personal information, thereby preserving privacy principles.

A.3.4 Professional issues

There were no third-party collaborations in this project, so the main professional issues involved following academic practices and maintaining research integrity through the respectful use of resources and thorough referencing of sources. Professional standards were upheld in writing the report and in the code present in the Git repository. All code is properly commented and explained for easy accessibility and understanding, and a readme file is attached to it.

Appendix B
External Materials

B.1 Dataset

The datasets used for this project are Foggy Citiscapes[32], Cityscapes[33] and Reside[34], all open sourced and available for academic use.

B.2 Code

Personal GitHub repository: https://github.com/uol-feps-soc-comp3931-2324-classroom/final-year-project-sc21samg

B.3 Tools

Microsoft Word was used for writing this report, while Google Colab facilitated the creation and customisation of tables, as well as the automation of diagrams. The representation of data improvements within the report was effectively managed using these tools. Additionally, the bulk of the code execution, along with the training of models and storage of large datasets, was handled in Google Colab, supported by Google Drive. This integration of tools ensured efficient handling of extensive computational tasks and data management requirements.

Appendix C
Additional Content - Large Images

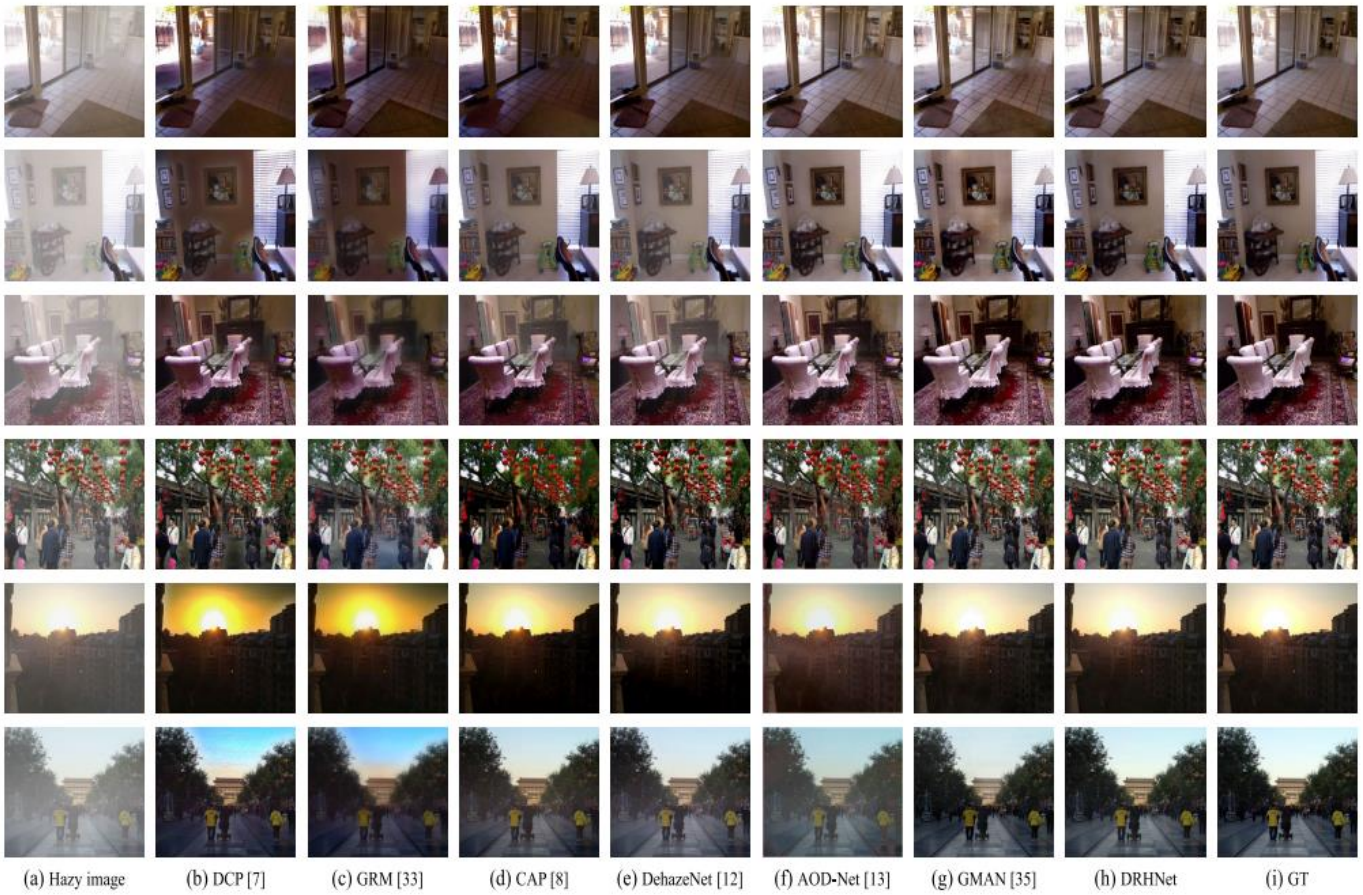


Figure C.0 Dehazed images using different dehazing algorithms. The references in the titles are not relevant here, Table 0.1 has the correct referencing.[51]

Figure C.3.0.a

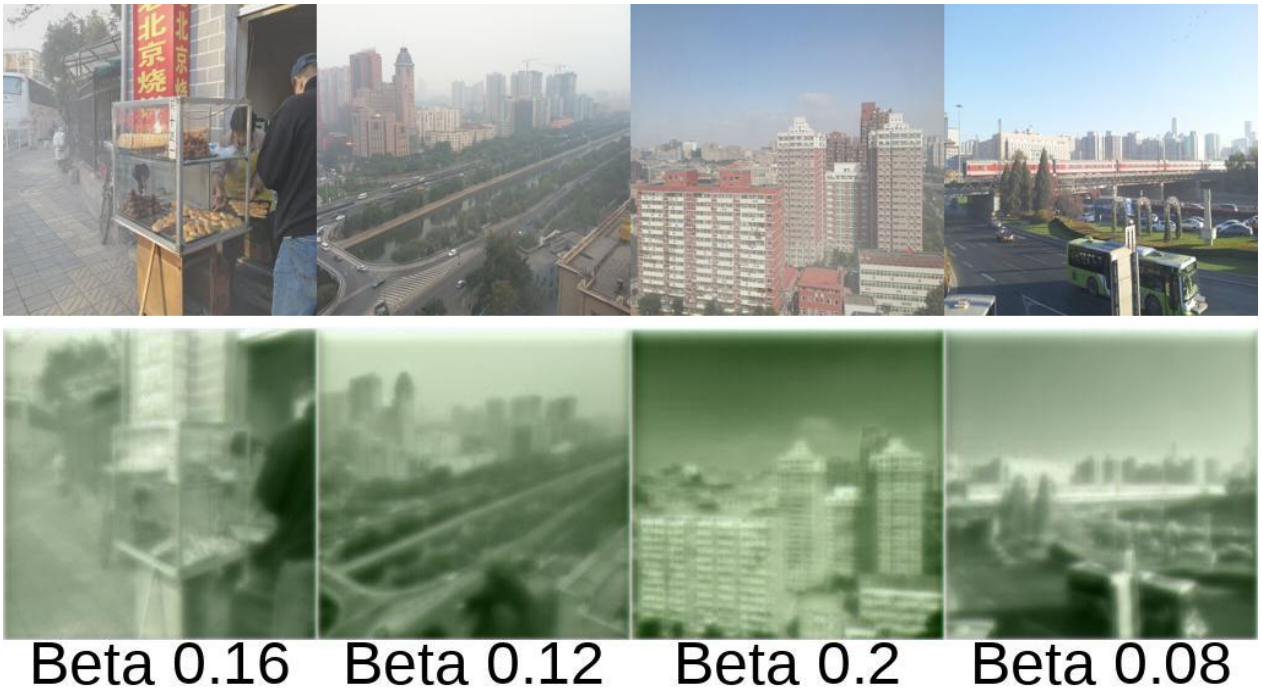


Figure C.3.0.b

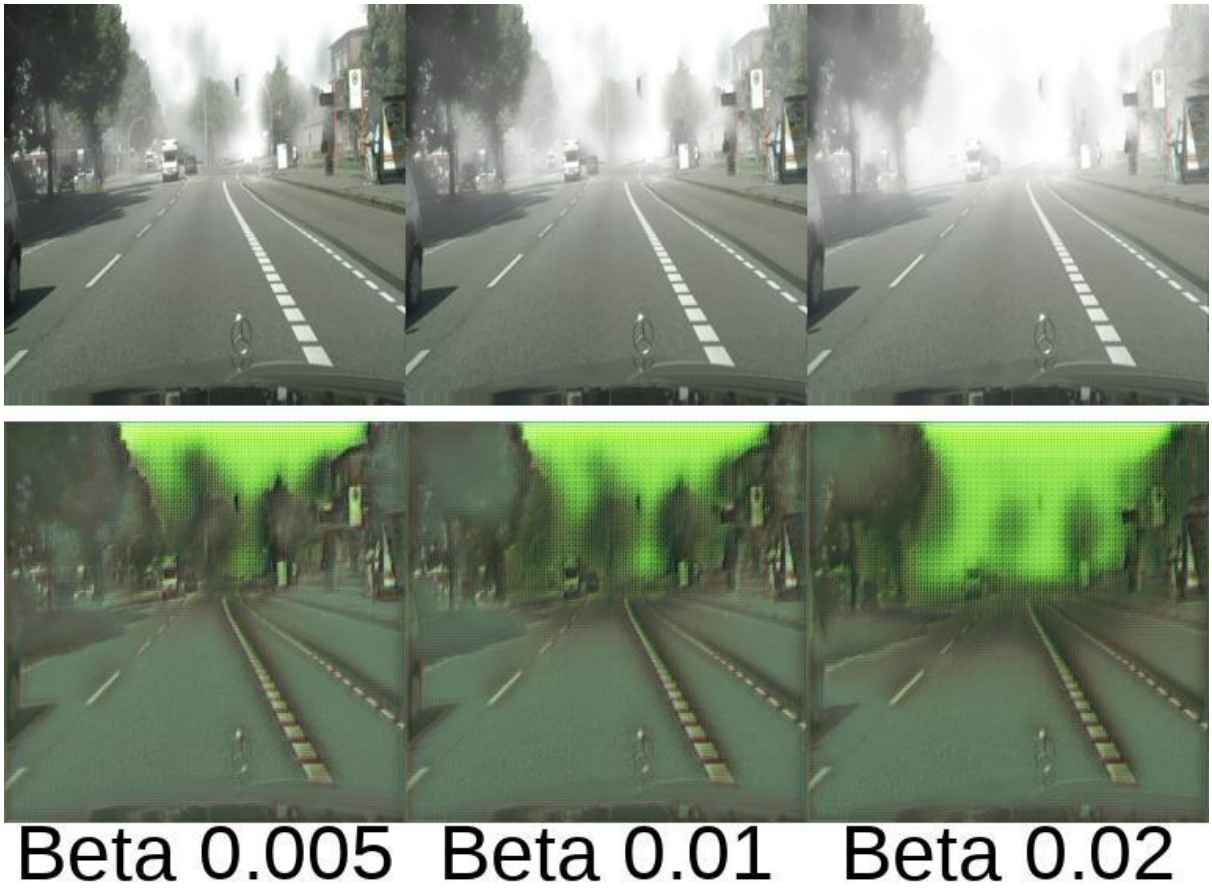


Figure C.3.1.a

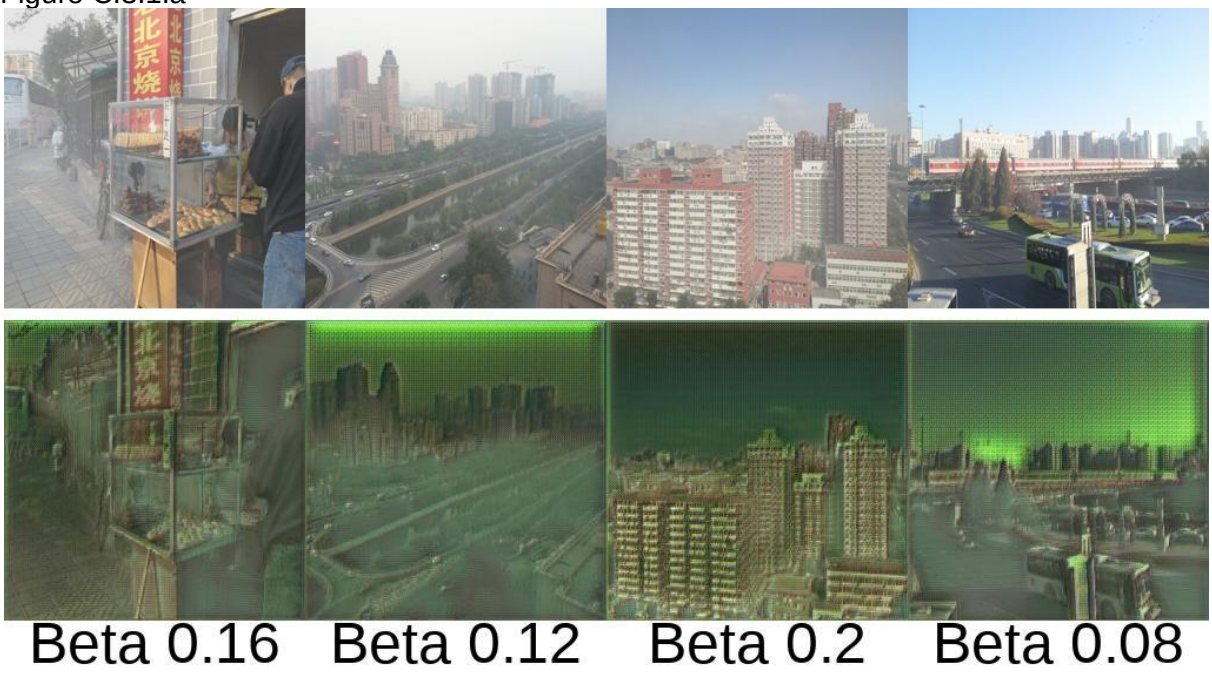


Figure C.3.1.b

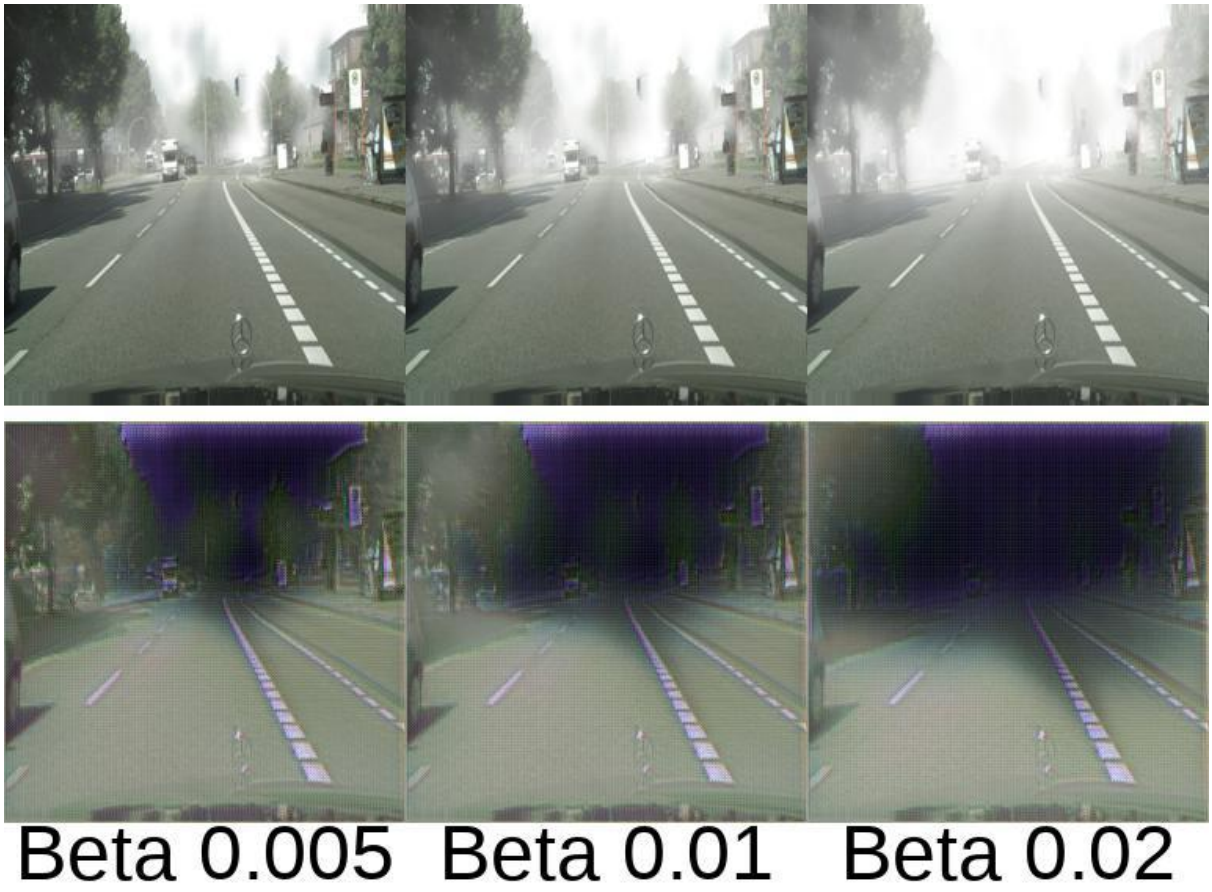


Figure C.3.2.a

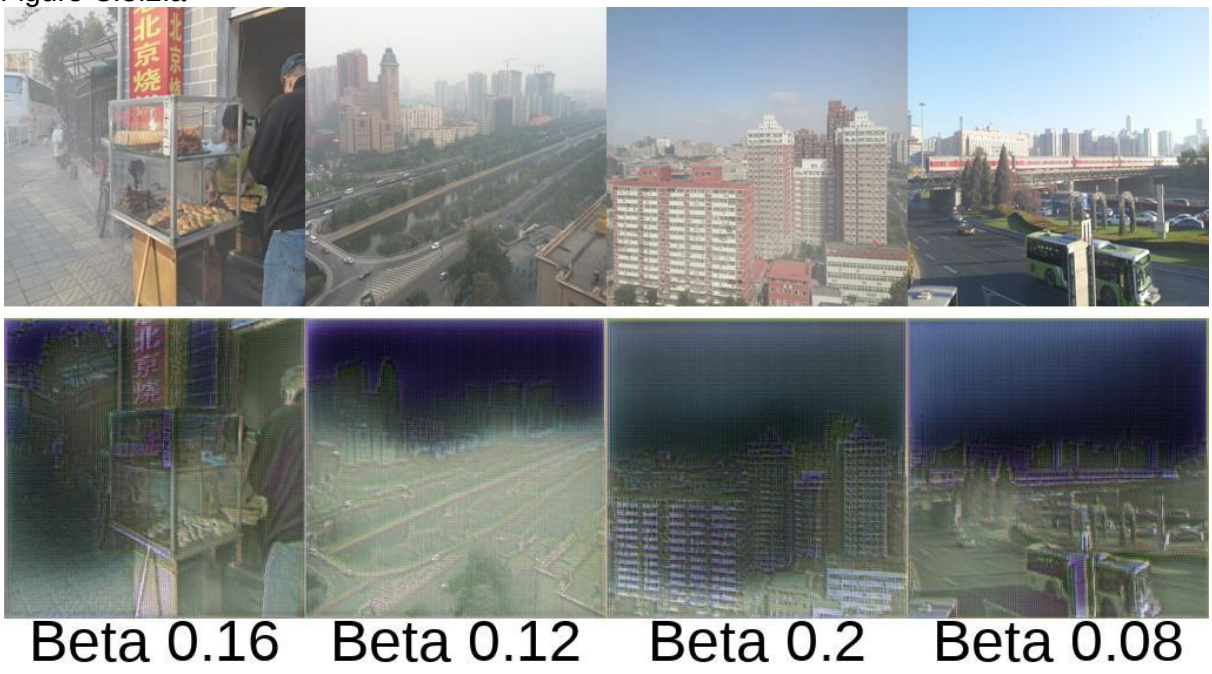


Figure C.3.2.b

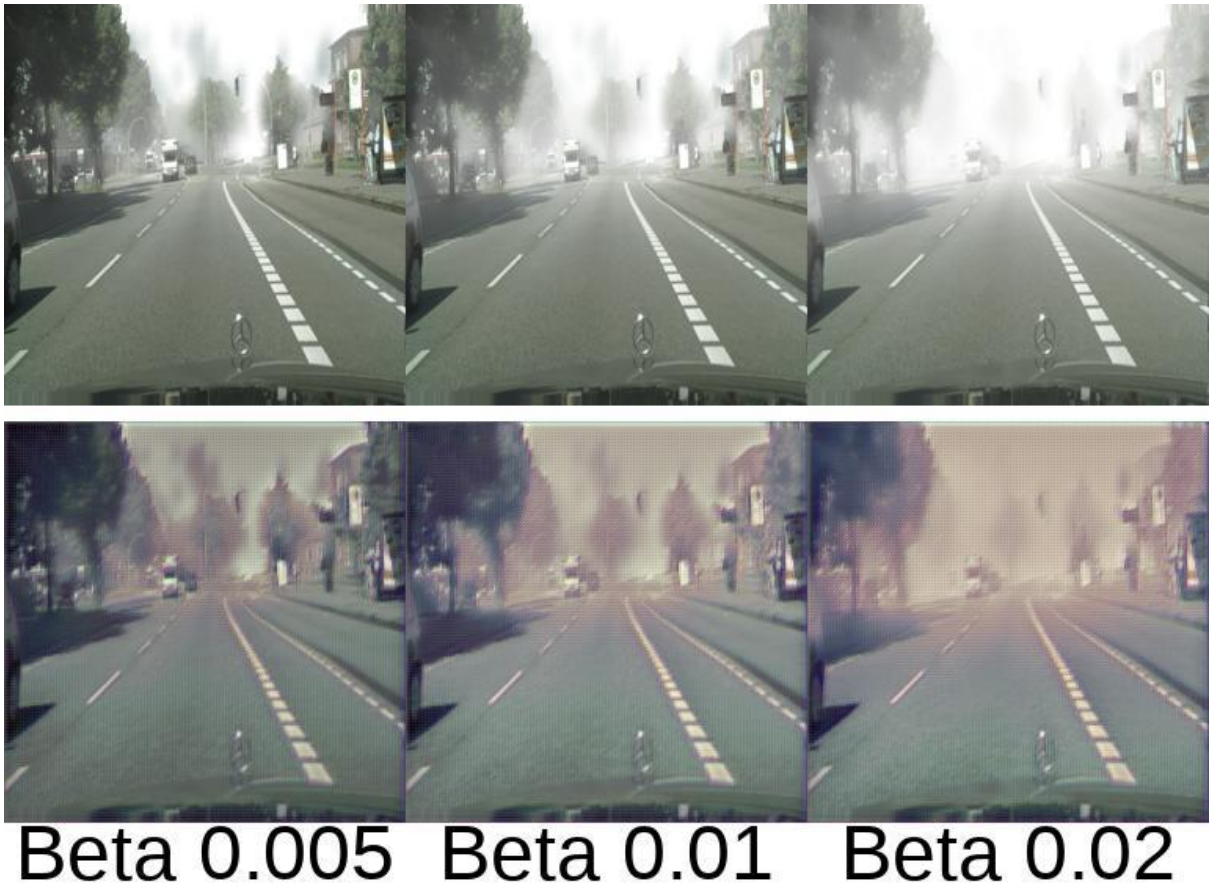


Figure C.3.3.a

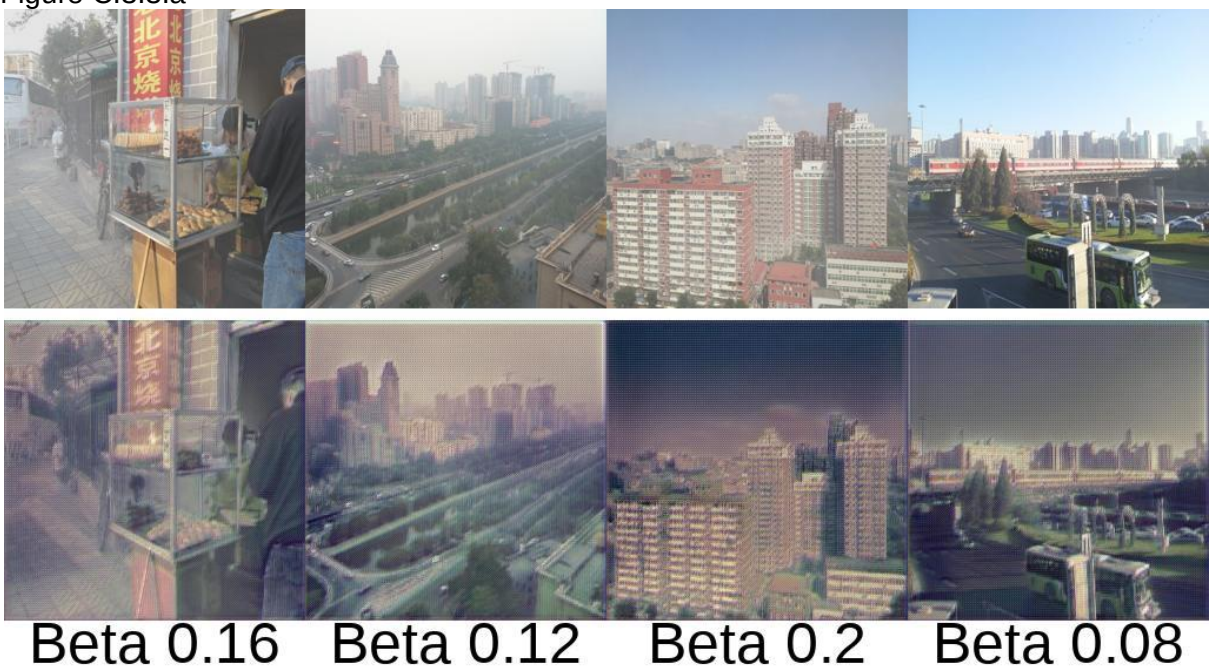


Figure C.3.3.b

Figure C.3.4.a

Figure C.3.4.b

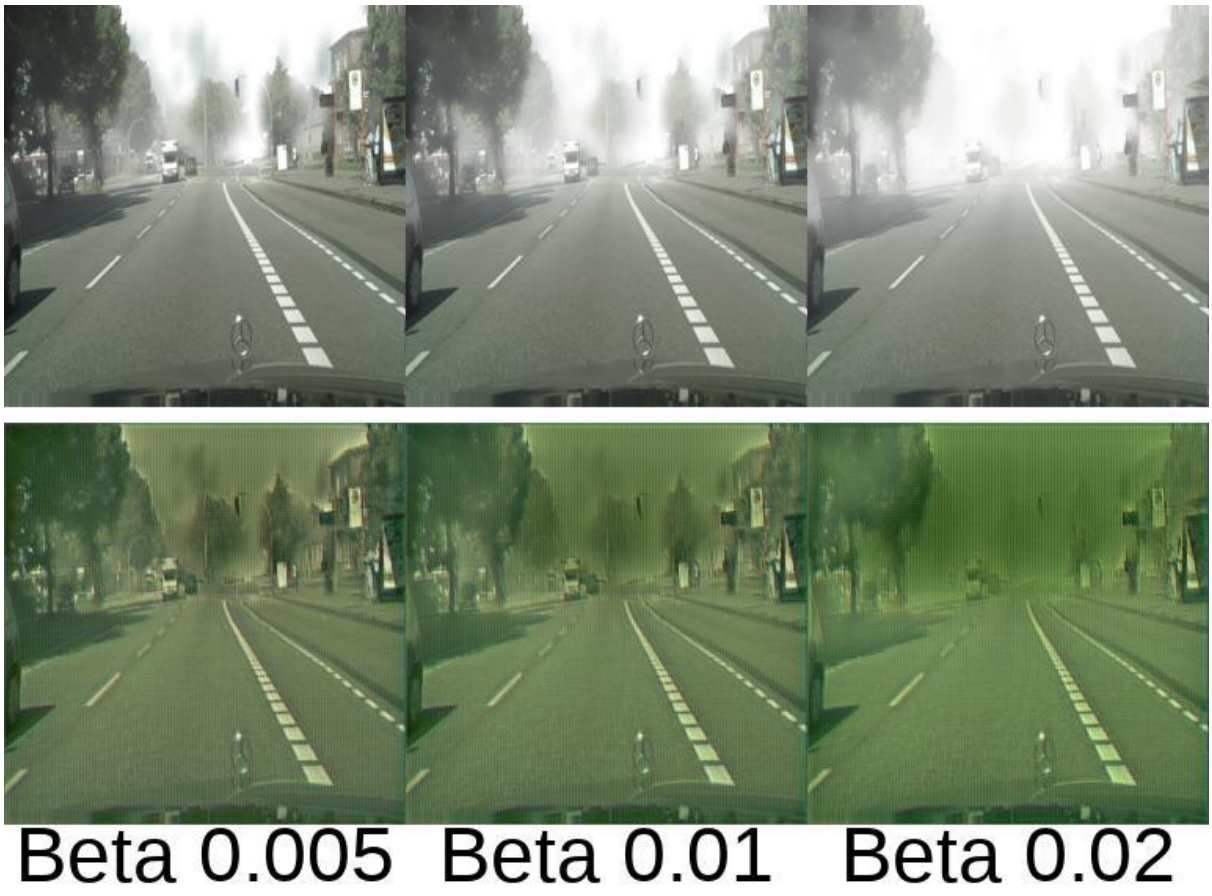


Figure C.3.5.a.

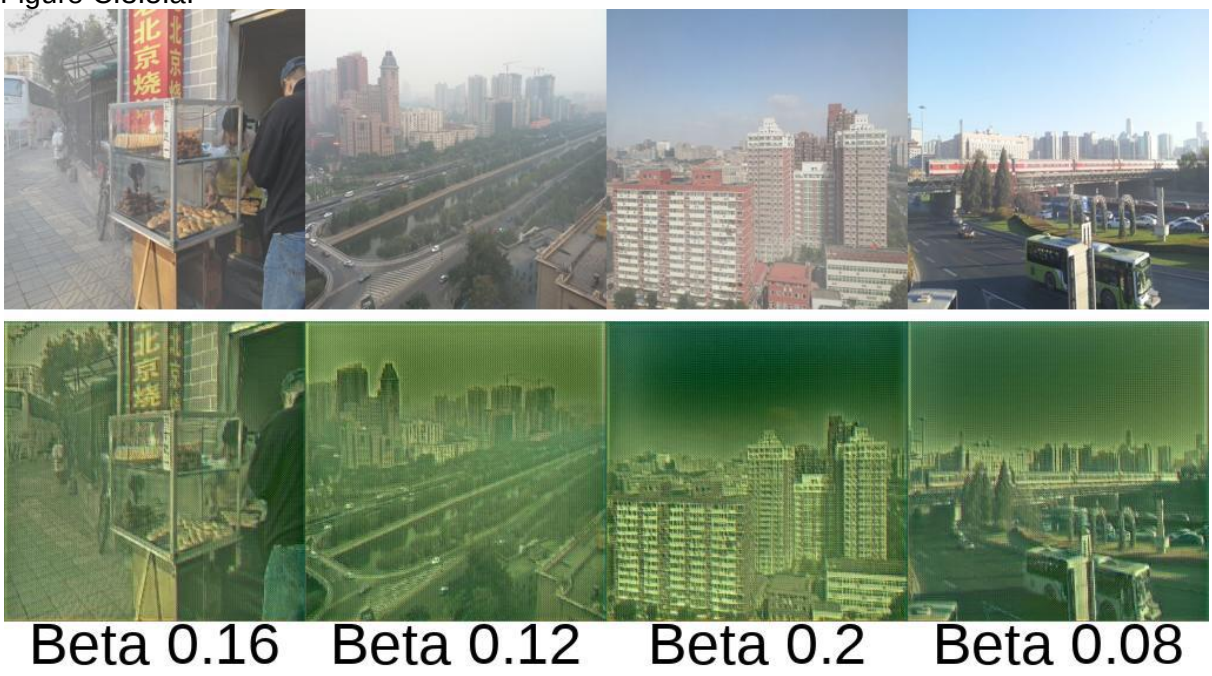


Figure C.3.5.b.

Figure C.3.6.1.a.

Figure C.3.6.1.b.

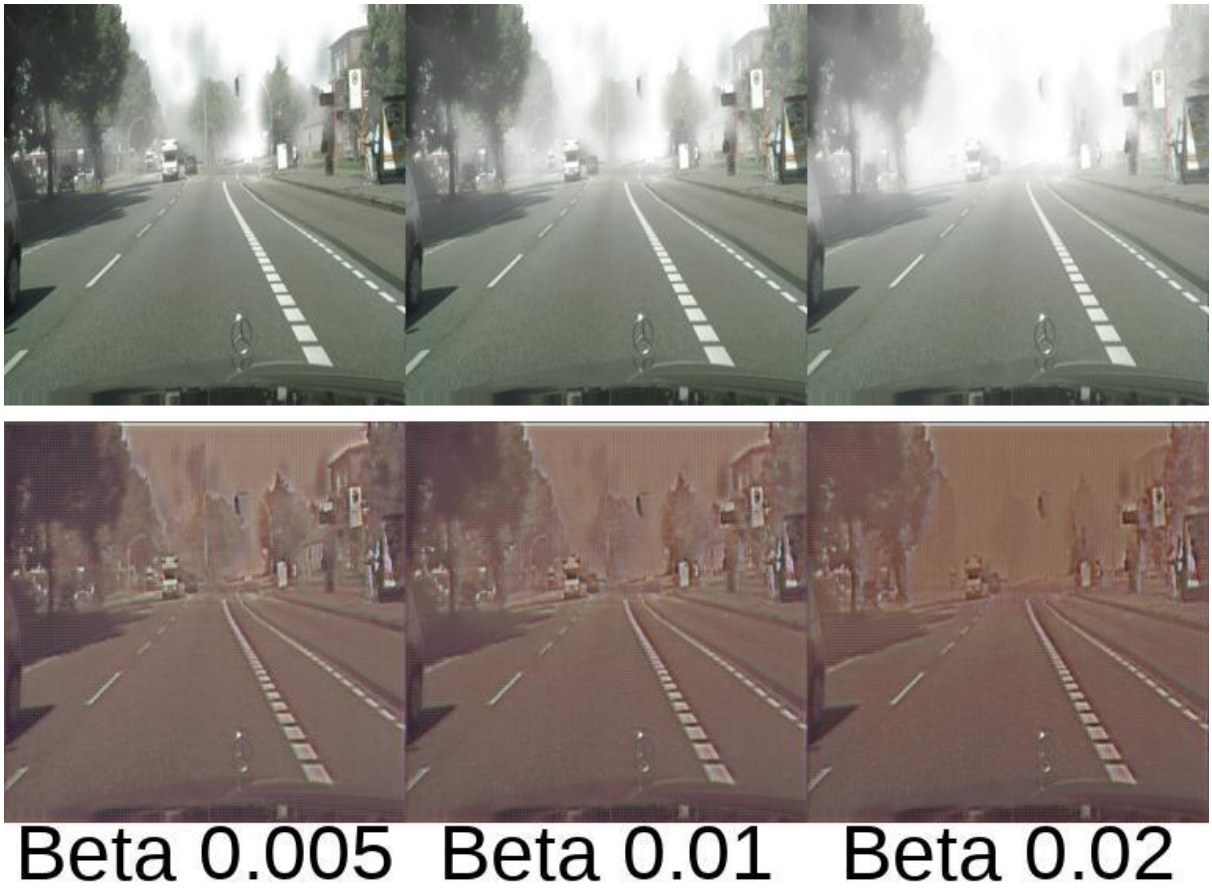


Figure C.3.6.2.a.

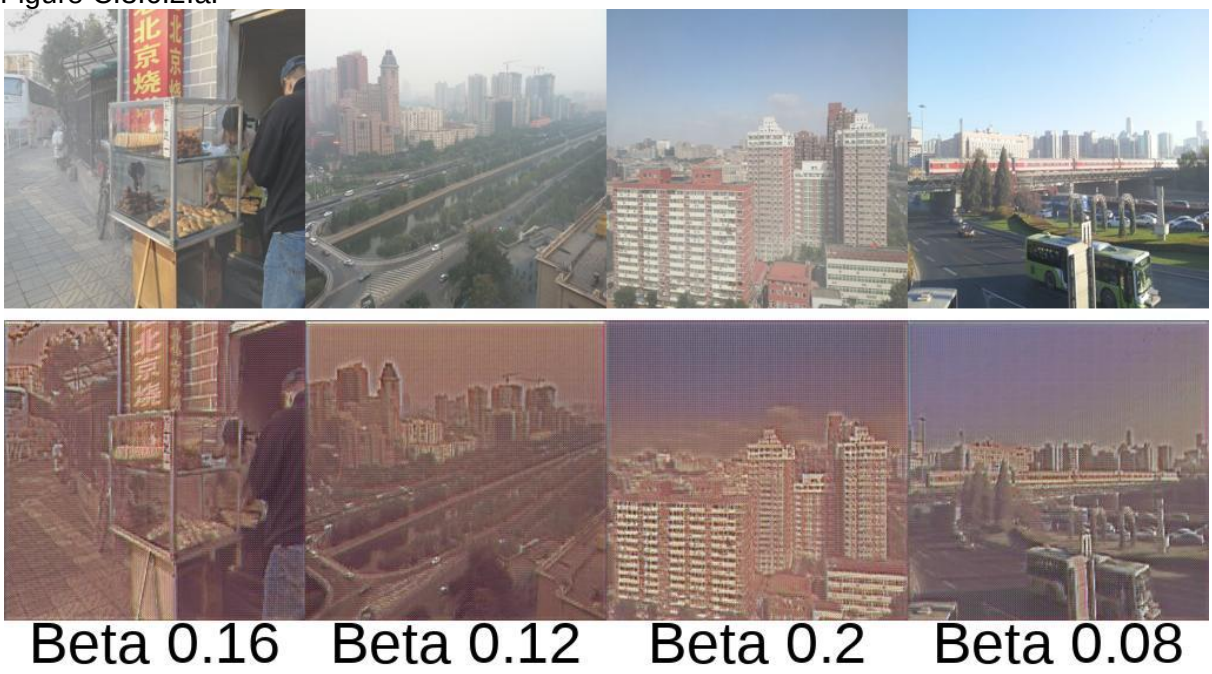


Figure C.3.6.2.b.

Figure C.3.6.3.a.

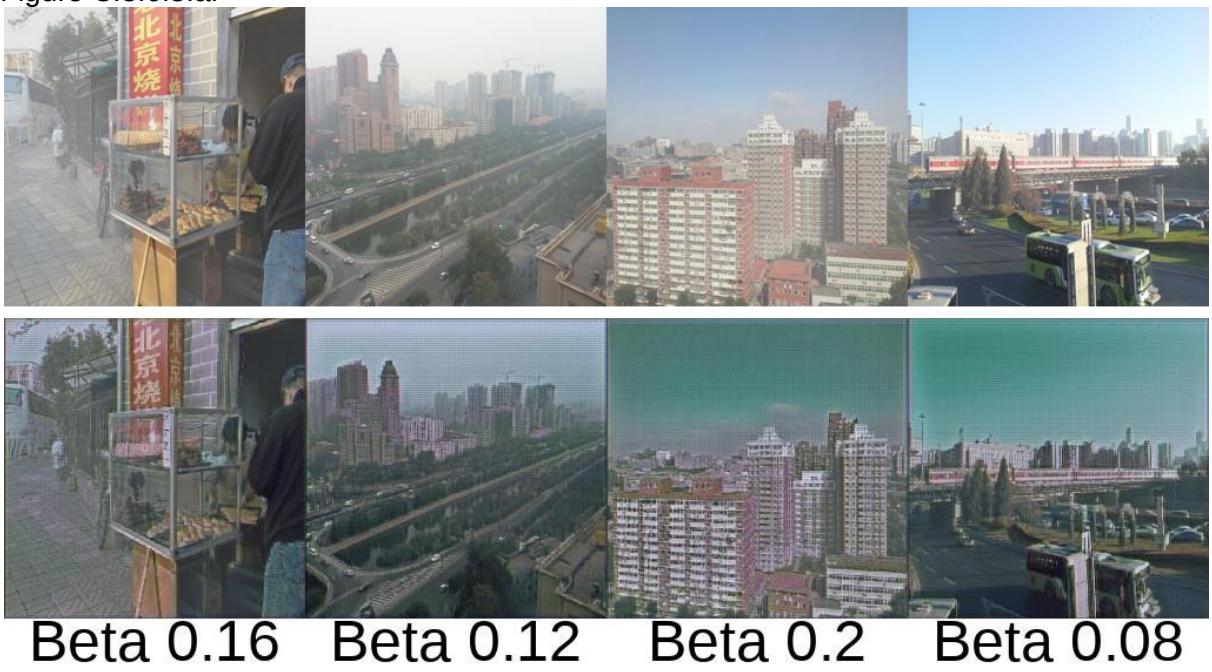


Figure C.3.6.3.b.

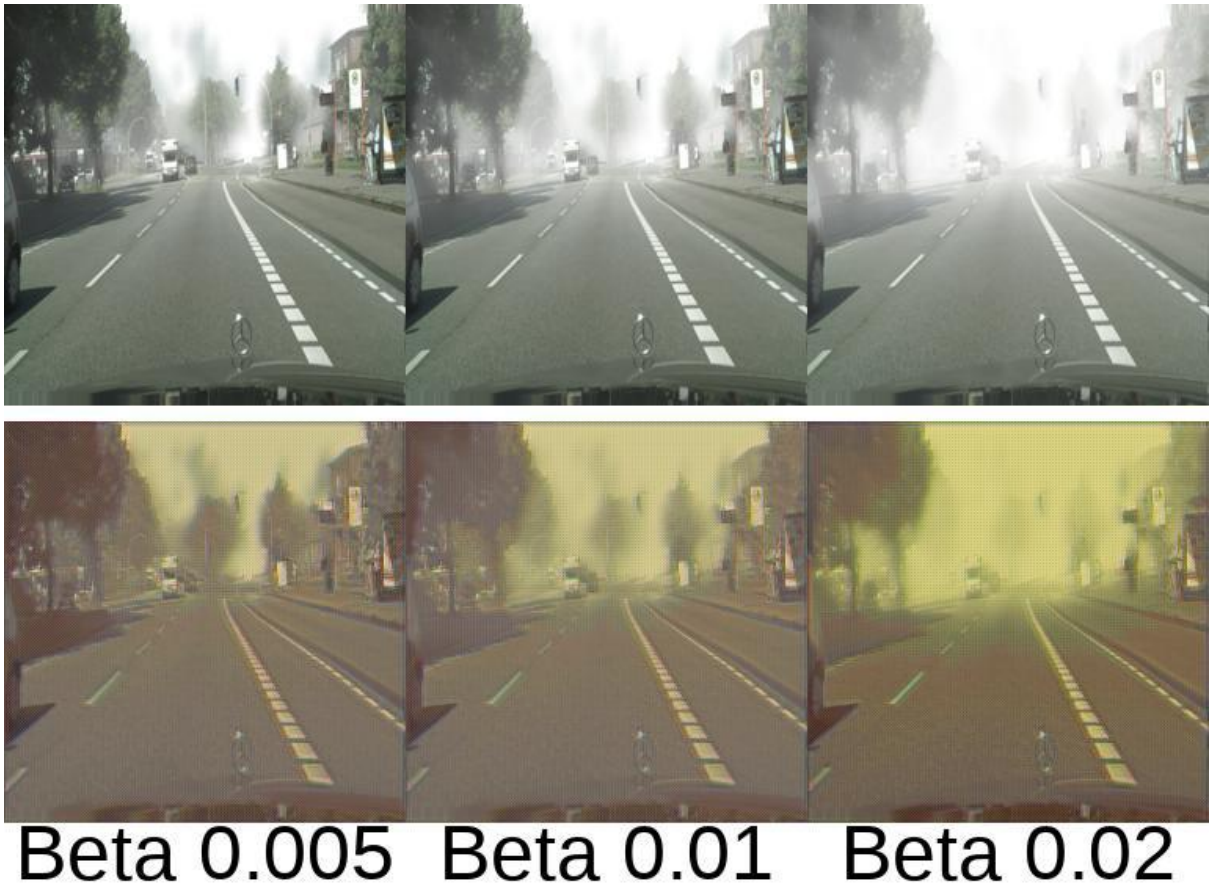


Figure C.3.7.a.

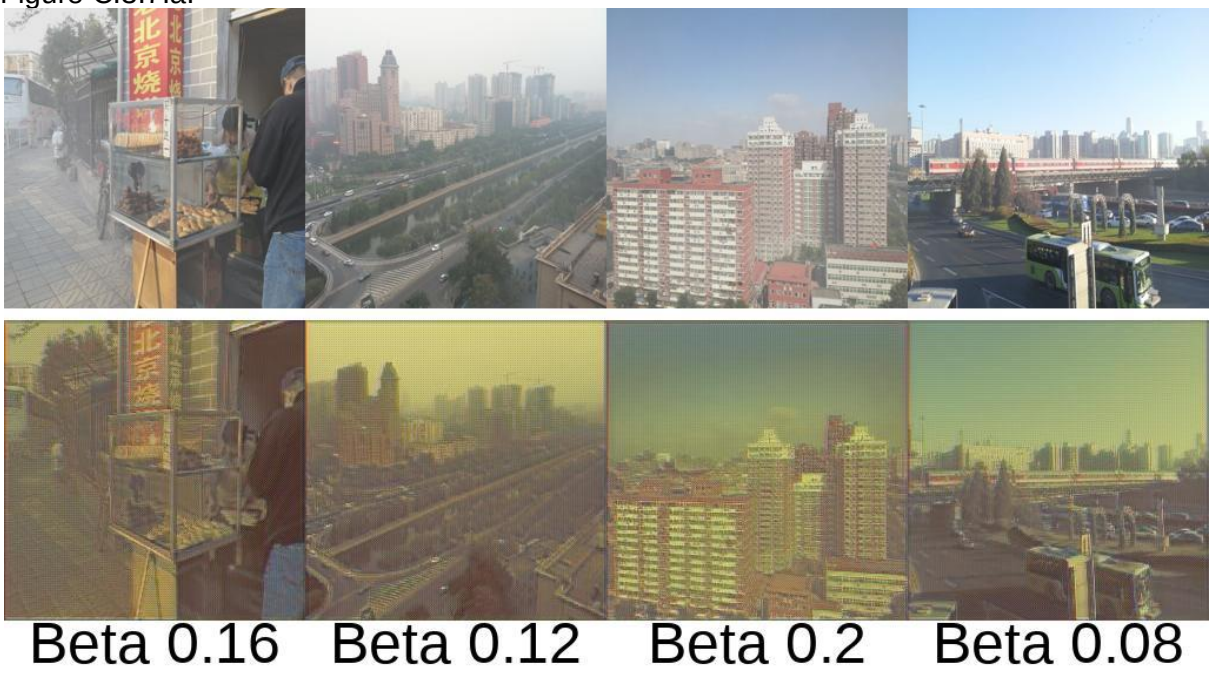


Figure C.3.7.b.

Figure C.3.8.1.a.

Figure C.3.8.1.b.

Figure C.3.8.2.a.

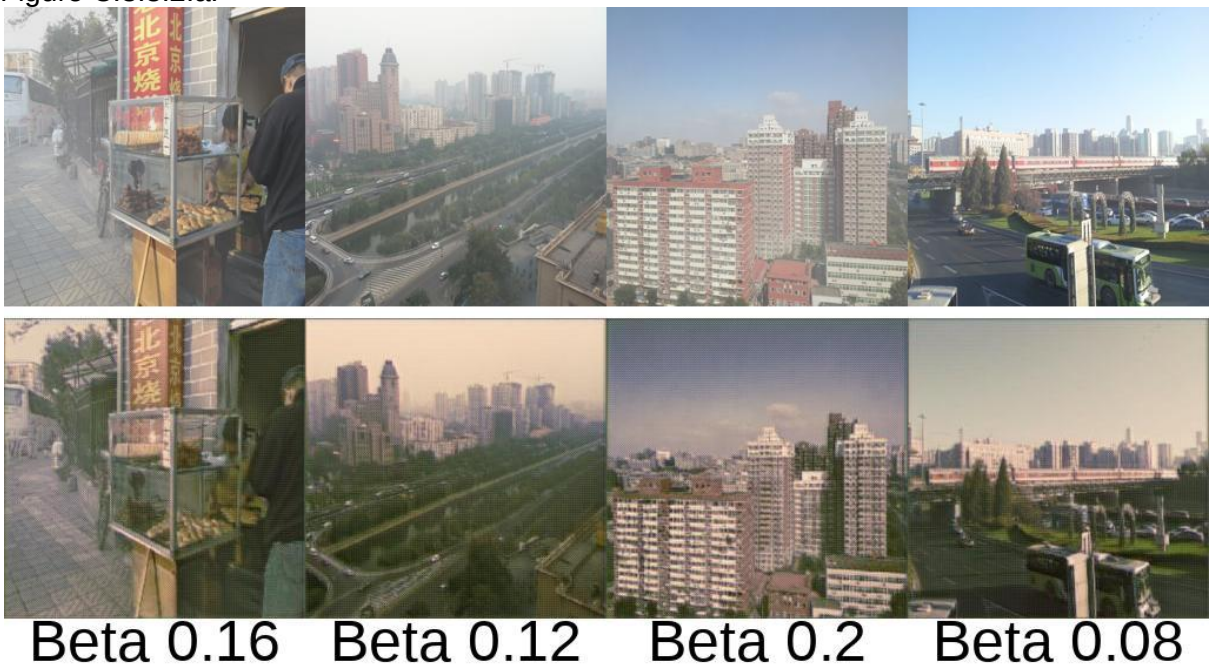


Figure C.3.8.2.b.

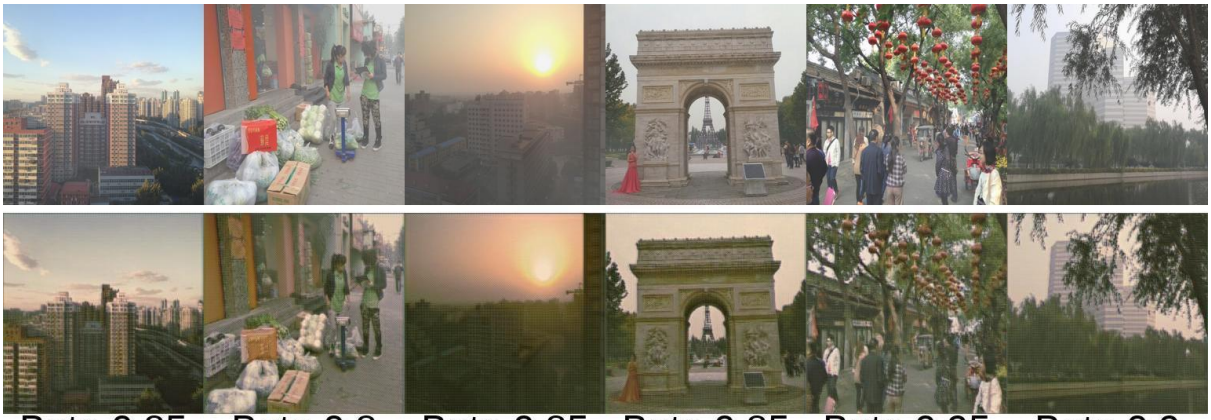


Table Image Performance Metrics

Image Name	PSNR	SSIM
0006_0.85_0.08.jpg	13.87	**0.5909**
0009_0.8_0.16.jpg	13.79	0.4818
0023_0.85_0.12.jpg	**11.75**	0.564
0109_0.85_0.08.jpg	**15.33**	0.5533
0297_0.95_0.08.jpg	12.47	**0.3766**
0299_0.9_0.08.jpg	13.31	0.498

Legend:

Highest Value

Lowest Value

Figure C.4.