DEHAZING OF MULTISPECTRAL REMOTE SENSING IMAGES USING DEEP LEARNING ALGORITHMS

This project report is submitted in partial fulfilment of the requirement

for the award of the degree

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

DORAGACHARLA LIZY

Reg.no: 208297601013

Under the esteemed guidance of

Dr. V. Persis

Associate Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ADIKAVI NANNAYA UNIVERSITY

RAJAMAHENDRAVARAM

2020-2024

ADIKAVI NANNAYA UNIVERSITY, RAJAMAHENDRAVARAM

UNIVERSITY COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that this project report entitled, "**DEHAZING OF MULTISPECTRAL REMOTE SENSING IMAGES USING DEEP LEARNING ALGORITHMS**" is a bonafIde work of **DORAGACHARLA LIZY**, Registration No: **208297601013** submitted in a partial fulfilment of the requirements for the award of Degree of BTech (CSE) during the period 2020-2024. This work carried out by her under my supervision and guidance and submitted to Department of Computer Science and Engineering, Adikavi Nannaya University.

INTERNAL GUIDE **HEAD OF THE DEPARTMENT**

EXTERNAL EXAMINER

DECLARATION

I **DORAGACHARLA LIZY,** reg.no: **208297601013**, hereby declare that the project report entitled **DEHAZING OF MULTISPECTRAL REMOTE SENSING IMAGES USING DEEP LEARNING ALGORITHMS** done by me under the guidance of **Dr. V. Persis, Associate Professor, Adikavi Nannaya University**, is submitted for the partial fulfilment of requirement of the award of the degree, Bachelor of Technology in Computer Science and Engineering in the academic period 2020-20024.

Signature of the student

Doragacharla Lizy

ACKNOWLEDGEMENT

I feel fortunate to pursue my **Bachelor of Technologies** degree in the **Department of Computer Science and Engineering**, **University College of Engineering, Adikavi Nannaya University**. It provided all the facilities in the area of Computer Science and Engineering. It's a genuine pleasure to express my deep sense of thanks and gratitude to my mentor and project guide of **Dr. V. Persis, Associate Professor, Department of Computer Science and Engineering, Adikavi Nannaya University** for her excellent guidance right from the selection of the summer internship and her valuable suggestions throughout the summer internship work. Her constant and timely counsel has caused me to succeed in completing this project in college.
I profusely thank **Dr. P. Venkateswara Rao**, Principal, University College of Engineering for all the encouragement and support.
I also thank **Dr. B. Kezia Rani**, Head of the Department, of Computer Science and Engineering for the guidance.
I also thank **Dr. D. Latha**, Project Coordinator, Department, Computer Science and Engineering for the guidance.
A great deal of thanks goes to Review Committee Members and the entire Faculty of the Department of Computer Science and Engineering, University College of Engineering, Adikavi Nannaya University, Rajamahendravaram for their excellent supervision and valuable suggestions.

Doragacharla Lizy
208297601013

ABSTRACT:

In the present scenario, the usage of satellite images for the data interpretation has increased drastically. Deep learning techniques are being widely used for the interpretation of data from the satellite images. At present satellite images are being affected by presence of the haze which causes difficulty in interpretation. Haze is caused due to presence of fine dust, smoke, or light vapors causing lack of transparency of the air. This creates a problem as the image regions affected by haze suffer lack of contrast resulting in difficulty of interpretation. In the mainstream, this issue is solved by applying atmospheric correction which is a tedious process and requires knowledge of several geo-physical quantities in order to give accurate results. The Objective is to enhance and implement image processing algorithms which can effectively restore pixels affected by presence of haze in order to improve image interpretability. The proposed work is divided into two phases. In the first phase we remove the noise from the image. In the second phase we dehaze the image using convolutional neural networks with residual structure. The proposed work is compared with Dark Object Subtraction method.

INDEX

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

1.1 INTRODUCTION:

In today's digital era, image processing has become a crucial aspect of various applications, including remote sensing, surveillance, and medical imaging. One of the significant challenges in image processing is the presence of haze, which reduces image clarity and visibility, thus hindering accurate interpretation and analysis. The removal of haze from images, known as dehazing, plays a vital role in enhancing image quality and enabling better decision-making in various domains.

The project focuses on the development of a deep learning-based approach for dehazing multispectral remote sensing images. Multispectral remote sensing involves capturing images across different wavelengths of light, providing valuable information for applications such as environmental monitoring, agriculture, and urban planning. However, these images are often affected by atmospheric haze, limiting their utility and interpretability.

To address this challenge, the project leverages Convolutional Neural Networks (CNNs) with a residual architecture for dehazing multispectral remote sensing images. CNNs have demonstrated remarkable capabilities in learning complex patterns and features from images, making them suitable for various image processing tasks. The residual architecture enhances the learning process by mitigating the vanishing gradient problem, enabling more effective training and better performance.

The proposed approach involves several key steps, including data preprocessing, model development, training, and evaluation. The project utilizes a dataset consisting of hazy and clear multispectral remote sensing images. The hazy images are pre-processed to remove noise and enhance contrast, preparing them for input to the CNN model. The CNN model is designed with a residual architecture to effectively learn and remove haze-induced distortions from the input images. during the training phase, the CNN model is trained on a subset of the dataset using an optimization algorithm to minimize the loss function. The model is evaluated using a separate subset of the dataset to assess its performance in dehazing images. Performance metrics such as Mean Squared Error (MSE) are used to quantify the quality of dehazed images compared to ground truth clear images. The project aims to provide a robust and efficient solution for dehazing multispectral remote sensing images, thereby enabling better analysis and interpretation of environmental data. The developed model has the potential to enhance various applications, including weather forecasting, land cover mapping, and disaster management. By improving image clarity and visibility, the project contributes to advancements in remote sensing technology and facilitates informed decision-making in diverse domains.

CNN

CNNs are neural networks designed for image recognition and analysis. They employ convolutional filters to extract features, followed by pooling and activation layers to condense the image representation. Training involves adjusting weights through backpropagation to minimize prediction errors. CNNs have transformed computer vision, finding applications in object and facial recognition, medical image analysis, and autonomous driving. Their success has spurred the development of other deep learning architectures for diverse tasks.

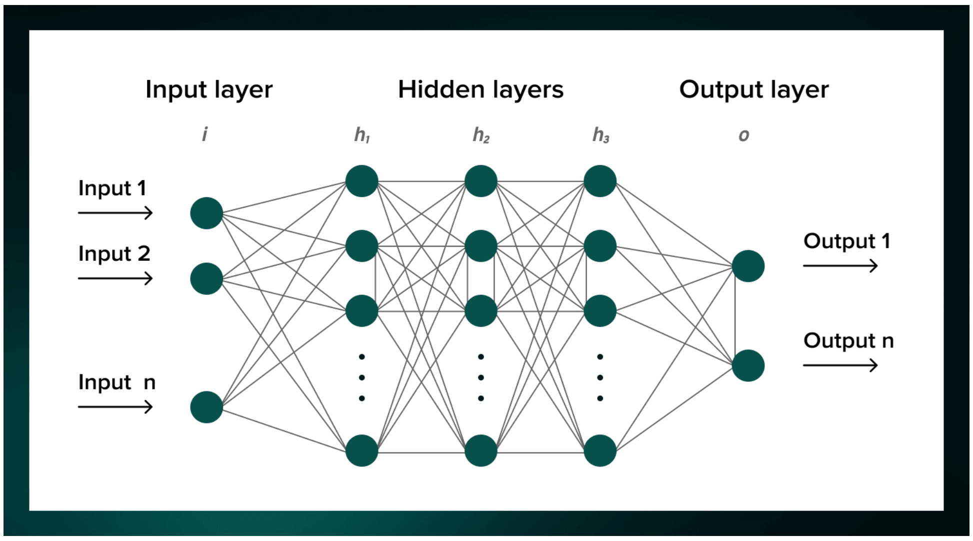


Fig 1.1.1: CNN architecture

ResNet

ResNet is a deep neural network architecture that uses shortcut connections to overcome the problem of vanishing gradients in very deep networks. It introduced the concept of residual

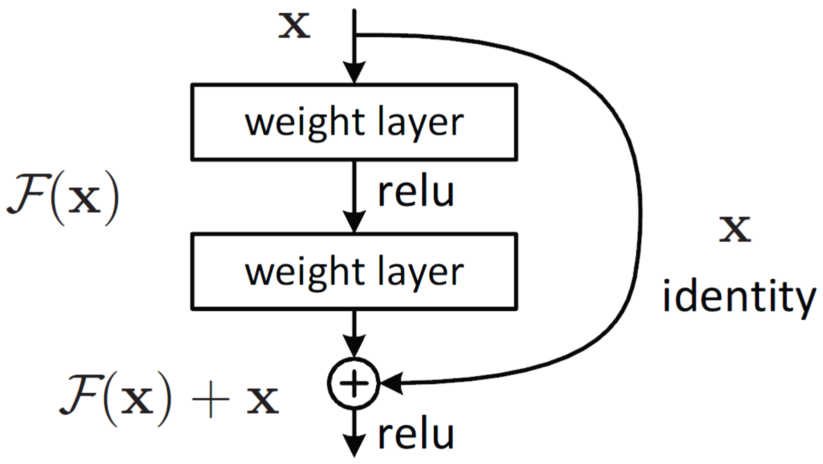

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig 1.1.2: ResNet Architecture

al learning, in which shortcut connections are added to skip some layers in the network and allow the gradient to flow more easily. ResNet achieved state-of-the-art results in many image classification tasks, including the ImageNet challenge.

TensorFlow

An open-source library called TensorFlow is used for numerical computing and for large-scale machine learning. It integrates several deep learning and machine learning models and algorithms (also known as neural networks) and aids in their applications.

VHSR Images

Very High-Resolution Satellite (VHRS) images refer to images with higher resolution, which means that pixel sizes are smaller and provide more details in the images. The resolution varies from 30cm to 5m per pixel. The clarity and sharpness of such images will be best suited for various research and application purposes.

1.2 EXISTING SYSTEMS:

1.2.1 Title: Haze Detection and Removal in Remotely Sensed Multispectral Imagery

Journal Details: IEEE Transactions on Geoscience and Remote Sensing, 52, 2014

Dataset: Landsat 8 OLI and WorldView-2

Description:

1. Haze decreases the quality of optical data and the precision of data interpretation. The tough and significant aspect of optical multispectral data correction is haze detection and removal.
2. An empirical and automatic method for detecting and removing inhomogeneous haze from medium and high-resolution satellite optical multispectral pictures is presented in the paper.
3. In order to compute a haze thickness map and remove haze from both calibrated and uncalibrated satellite multispectral data, the dark-object subtraction approach is further refined.
4. The approach has limits in rare settings with a uniform and strongly reflective landcover. The spectral consistency after haze reduction is evaluated using hazy multispectral data (Landsat 8 OLI and WorldView-2) and contrasted with haze-free reference data.

Advantages:

1. This algorithm removes spatially varying haze and helps in interpretation of data.
2. An evaluation of the dehazing method using a clear and a hazy scene from the same area demonstrates that the dehazing results are spectrally consistent.
3. The dehazed data can further be used for atmospheric/ topographic correction

Disadvantages:

1. For scenes with absolutely flat and highly reflective surfaces (desert and snow), it could be impossible to locate dark objects and to estimate a haze thickness map (HTM).

1.2.2 Title: Memory-Oriented Unpaired Learning for Single Remote Sensing Image Dehazing

Journal Details: IEEE Geoscience and Remote Sensing Letters, 22, 2022.

Dataset: Landsat 8

Description:

1. Due to the irregular and nonuniform distribution of haze, remote sensing image dehazing (RSID) is a very difficult challenge.
2. The current RSID algorithms use deep learning to achieve good performance, however reliance on paired synthetic data limits their generalizability in different haze distributions.
3. In this letter, they introduce a memory-oriented generative adversarial network (MO-GAN), which aims to learn unpaired towards a single RSID while attempting to capture the required fuzzy aspects.
4. An innovative multistage attentive-recurrent memory module is created to direct an autoencoder neural network, which can record the diverse appearances of haze distribution at different stages, in order to better extract the haze-relevant features.

Advantages:

1. It proposed a novel MO-GAN for RSID, which makes full use of the memory network to record haze distribution patterns without paired labels.
2. In addition, a region-aware based discriminator is introduced to effectively regularize the output.

Disadvantages:

1. It is difficult to understand the process and execute the process.
2. It may give inaccurate results in some cases where are images are highly affected by other parameters.

1.2.3 Title: Multi-Input Attention Network for Dehazing of Remote Sensing Images

Journal Details: Applied Sciences, 12,2022.

Dataset: Sentinel - II

Description:

1. Multi input CNN based on an encoder-decoder structure to effectively restore remote sensing hazy images. This network can learn mapping between hazy and corresponding haze free images.
2. The dataset that is used to train this model was Sentinel-II with a total of 9 bands.
3. For feature extraction, double convolution layer n/w group is used for initial extraction for input images. Down sampling and feature extraction are implemented through double convolution network.
4. Decoder up samples high level features from the encoder and restore image to dehazed image. The training data consists 80% of dataset and 20% is used as testing dataset.
5. A total of 200epochs are performed and this method outperforms DCP, Dehazenet, and ADO-net. A maximum of 94.4% accuracy is achieved from band 8 and a min of 88.7% is achieved by band 7

Advantages:

1. The attention mechanism helps to suppress noise caused by haze, resulting in a more robust dehazing performance.
2. The multi-input attention network can be trained on one dataset and applied to other datasets, allowing for transferability across different remote sensing applications.

Disadvantages:

1. The dehazing performance of the method can be affected by the quality of the input image, and it may not work well for extremely low-quality images.
2. The method may not generalize well to unseen data or new environments, requiring retraining or fine-tuning.

1.2.4 Title: Smoke Net: Satellite Smoke Scene Detection Using Convolutional Neural Network with Spatial and Channel-Wise Attention

Journal Details: Remote Sensing, 11, 2019

Dataset: MODIS

Description:

1. Smoke Net is a technique for detecting smoke scenes in satellite imagery using convolutional neural networks (CNNs) with spatial and channel-wise attention mechanisms.
2. This approach can be used to monitor wildfire activity, pollution, and other smoke-related events from satellite data.
3. The first step includes the preprocessing of satellite image and the processed image is fed into CNN to learn features for smoke scenes. The second step is in add'n it uses spatial attention mechanism that focus on most imp regions of image.
4. The third step is it uses channel wise attention mech to further enhance CNN ability to detect smoke. The training is done by selecting 64% of images from 6225 satellite images and reported an accuracy of 92.5%.

Advantages:

1. High Accuracy: Smoke Net can achieve high accuracy in detecting smoke scenes in satellite imagery.
2. Real-Time Processing: The technique is capable of processing satellite imagery in real-time.

Disadvantages:

1. Limited Training Data: The technique requires a significant amount of high-quality training data to effectively train the neural network, which may be difficult to obtain in certain smoke-related events.

1.2.5 Title: Multi-Scale Residual Convolutional Neural Network for Haze Removal of Remote Sensing Images

Journal Details: Remote Sensing, 10, 2018

Dataset: Landsat8(5 bands)

Description:

1. MRCNN utilizes 3D convolutional kernels to extract spatial–spectral correlation information and abstract features from surrounding neighborhoods for haze transmission estimation.
2. Residual learning is utilized to avoid the loss of weak information while deepening the network.
3. The technique uses a multi-scale residual CNN architecture that incorporates both local and global features to effectively remove the haze.
4. The MRCNN utilizes 3D convolution kernels to extract spatial-spectral correlation information and extract features from surrounding neighbourhoods. The dataset that is used to train the model is Landsat8 considering 5 bands.
5. The overall architecture mainly contains four sequential modules: (1) spectral–spatial feature extraction (2) multi-scale context aggregation (3) residual learning and (4) fully connected layers.

Advantages:

1. Improve generalization ability and prevent overfitting
2. This method can converge faster and can achieve a higher prediction accuracy compared with DehazeNet and VGGNet

Disadvantages:

1. The complexity of the network can make it difficult to interpret how the network is making its decisions.
2. The method requires large amounts of training data to learn the complex relationships between the local and global features and the haze.

1.2.6 Title: Remote Sensing Image Dehazing Based on an Attention Convolutional Neural Network

Journal Details: IEEE Access, 10, 2022

Dataset: ORL and Caltech Face

Description:

1. Existing remote sensing dehazing methods based on simplified atmospheric degradation models are not suitable for the removal of heterogeneous haze that exist in remote sensing images.
2. Convolutional neural network based on attention mechanism, in which the residual block structure combines both channel and spatial attention mechanisms, and establishes a synthetic high-resolution haze image dataset for full training.
3. This method proposes a solution that an input image is given to the CNN model and process that image with multiple convolution layers by extracting the features from the hazy image and then the obtained output image is combined with attention mechanism to produce final dehaze image.
4. Overall, this technique is an effective way to improve the quality of remote sensing images captured in hazy or foggy environments.

Advantages:

1. The technique is capable of significantly improving the visibility of hazy images, resulting in clearer and more useful remote sensing data.
2. The trained network can be applied to different types of hazy images without the need for retraining.

Disadvantages:

1. The technique requires a large number of computational resources, including high-performance GPUs, making it difficult to apply in resource-constrained environments.
2. The technique requires a significant amount of high-quality training data to effectively train the neural network, which may be difficult to obtain in certain remote sensing applications.

1.2.7 Title: Remote Sensing Image Dehazing

Journal Details: Sensors, 21, 2021

Dataset: ORL and Caltech Face

Description:

1. The idea of categorizing the algorithm into three groups—Image Enhancement, Physical Dehazing, and Data Driven Dehazing—is put forth in this study.
2. Physical dehazing techniques like Dark channel prior and haze line prior may result in darker Remote Sensing Outputs, whereas image enhancement techniques like Histogram equalization and Retinex alg have high contrast and lost some features.
3. Data driven dehazing algorithms like Multi-scale Convolution Neural Network (MSCNN) are able to produce high quality haze free scenes.

Advantages:

1. Feasible and Effective method for haze removal of urban RS images and has a good application and promotion value
2. Having a low complexity to ensure real-time performance

Disadvantages:

1. The image dehazed by ASM (Atmospheric Scattering Model) will have a dim effect since ASM fails to consider the light trapping phenomenon related to the texture density and scene depth.
2. Data-driven dehazing is similar to a “black box”, which lacks interpretability and is specifically theoretical despite its effectiveness.

1.2.8 Title: A New Haze Removal Algorithm for Single Urban Remote Sensing Image

Journal Details: IEEE Access, 8, 2020

Dataset: ORL and Caltech Face

Description:

1. Monitoring and managing the urban environment and its resources effectively requires the use of remote sensing imaging detection technology.
2. Haze has a significant negative impact on the quality of optical remote sensing picture acquisition, leading to blurred remote sensing images, the loss of detail information, and colour distortion.
3. In this research, a new urban remote sensing haze removal (URSHR) algorithm is given. It integrates the picture phase consistency feature, multi-scale Retinax theory, and histogram characteristic to lessen the impact of haze.
4. Using multi-scale Retinex theory and histogram characteristics, the URSHR approach first removes image haze. Next, the detail information of the image is boosted using phase consistency features, and lastly, they are combined with the multi-scale wavelet transform.

Advantages:

1. Experimental results fully prove that the URSHR algorithm proposed in this paper can get better detail information and high contrast, and has a strong ability to remove haze.
2. This algorithm can be used for low resolution as well as high resolution satellite images.

Disadvantages:

1. Various number of parameters should be taken into consideration for dehazing.
2. There is a slight chance of loss of information through this method.

1.3 PROPOSED SYSTEM:

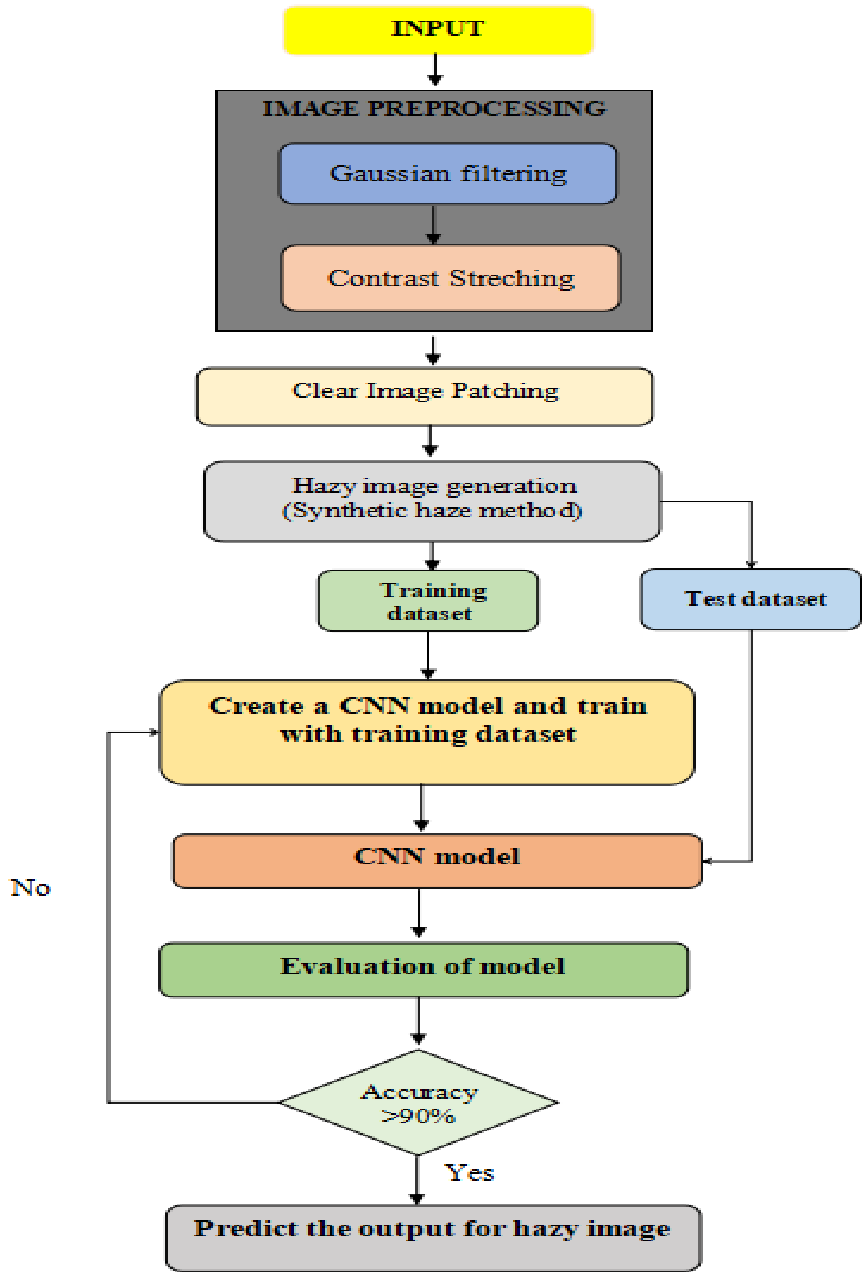


Fig 1.3.1: Flow chart of proposed model

Image processing

Gaussian Filtering: Gaussian filtering is a spatial filtering technique used to reduce noise and smooth images by applying a weighted average based on the Gaussian distribution across neighboring pixels. It is commonly employed in image processing to enhance clarity and suppress unwanted artifacts.

Contrast Stretching: Contrast stretching is an image enhancement technique that expands the range of pixel intensity values in an image, aiming to improve visibility and emphasize details by stretching the original intensity range to cover the full available scale. This method is often used to enhance the visual appearance of images with low contrast.

Clear image patching

Clear image patching involves extracting smaller, non-overlapping regions or patches from a high-resolution image to create a dataset with distinct, clear regions for analysis or training purposes in image processing and computer vision applications. This technique is often used to facilitate the processing of specific regions within an image.

Hazy image generation (synthetic haze method)

Synthetic haze generation involves artificially introducing atmospheric haze to clear images, simulating hazy conditions for the purpose of training or testing dehazing algorithms in computer vision applications. This method aids in the development and evaluation of algorithms for improving visibility in hazy conditions.

1.3.1 MODULES OF PROPOSED MODEL:

Module 1: Data Preprocessing

1. This module involves collecting a large dataset of hazy and its corresponding clear image.
2. Normalize the images using techniques such as z-score normalization or min-max scaling.
3. Use noise removal techniques such as median filtering or non-local means filtering to reduce or eliminate unwanted noise or artifacts from images.

Module 2: Training and Testing Module

1. Split the dataset into training and testing sets.
2. Train the model with the training dataset. Trained includes giving the hazy image as well as its corresponding clear image.
3. Test the model using the dataset and evaluate the performance of the model.
4. An optimization algorithm, such as stochastic gradient descent (SGD) or Adam is used to update the model's weights during training and minimize the loss function.

Module 3: Activation Functions Module

1. Activation functions, such as ReLU (Rectified Linear Unit), are applied to introduce non-linearity and enable the model to learn complex relationships between features.

Module 4: Output Module

1. The output layer generates the dehazed image, typically using an appropriate activation function, such as sigmoid or tanh, to ensure that the output pixel values are within the desired range.

CHAPTER-2

SYSTEM ANALYSIS

2.1 HARDWARE REQUIREMENTS:

1. **GPU (Graphics Processing Unit):** A specialized electronic circuit essential for accelerating graphics rendering, deep learning tasks, and parallel computations.
2. **Memory (RAM - Random Access Memory):** High-speed temporary storage vital for quick access to data and instructions, enhancing overall system performance.
3. **Storage:** Permanent data storage medium, available in various forms like HDDs, SSDs, and NVMe SSDs, influencing speed, capacity, and cost of computing systems.

2.2 SOFTWARE REQUIREMENTS:

Deep Learning Framework: Software libraries facilitating the implementation of deep neural networks for tasks like image recognition and natural language processing.

Python: A versatile and widely-used programming language known for its simplicity and readability, commonly used in deep learning projects.

CUDA Toolkit: A development platform enabling GPU-accelerated computing, essential for running deep learning algorithms efficiently.

CuDNN Library: NVIDIA's GPU-accelerated library providing highly optimized primitives for deep learning, enhancing performance of neural network computations.

Image Processing Libraries: Software tools offering functionalities for manipulating, analyzing, and enhancing digital images, often utilized in preprocessing tasks for deep learning models.

Google Collab Pro: A cloud-based platform providing access to GPU-accelerated computing resources, enabling collaborative and scalable deep learning experimentation and development.

Development Environment: The integrated set of tools and software configurations used by developers for writing, testing, and debugging code, crucial for efficient deep learning model development and deployment.

2.3 FUNCTIONAL REQUIREMENTS:

1. The primary function of the system is to effectively dehaze input images and enhance the image quality.

2. The dehazed images produced should closely resemble the ground truth clear images, with minimal distortions or colour shifts.

3. The system should be able to handle a wide range of hazy images with varying haze intensities and weather conditions

4. The system should be capable of processing images in real-time

5. The image dehazing system should have a user-friendly interface that allows users to input images, adjust parameters, and view the dehazed output.

2.4 NON-FUNCTIONAL REQUIREMENTS:

1.The image dehazing system should be computationally efficient.

2.The system should manage memory efficiently, avoiding excessive memory usage that could lead to memory overflow or performance degradation.

3. The system should be robust to handle different types and levels of haze.

4. The system should be easy to use, with a user-friendly interface.

5. The system should be reliable, able to consistently produce accurate and consistent results

CHAPTER-3

SYSTEM DESIGN

3.1 SYSTEM ARCHITECTURE:

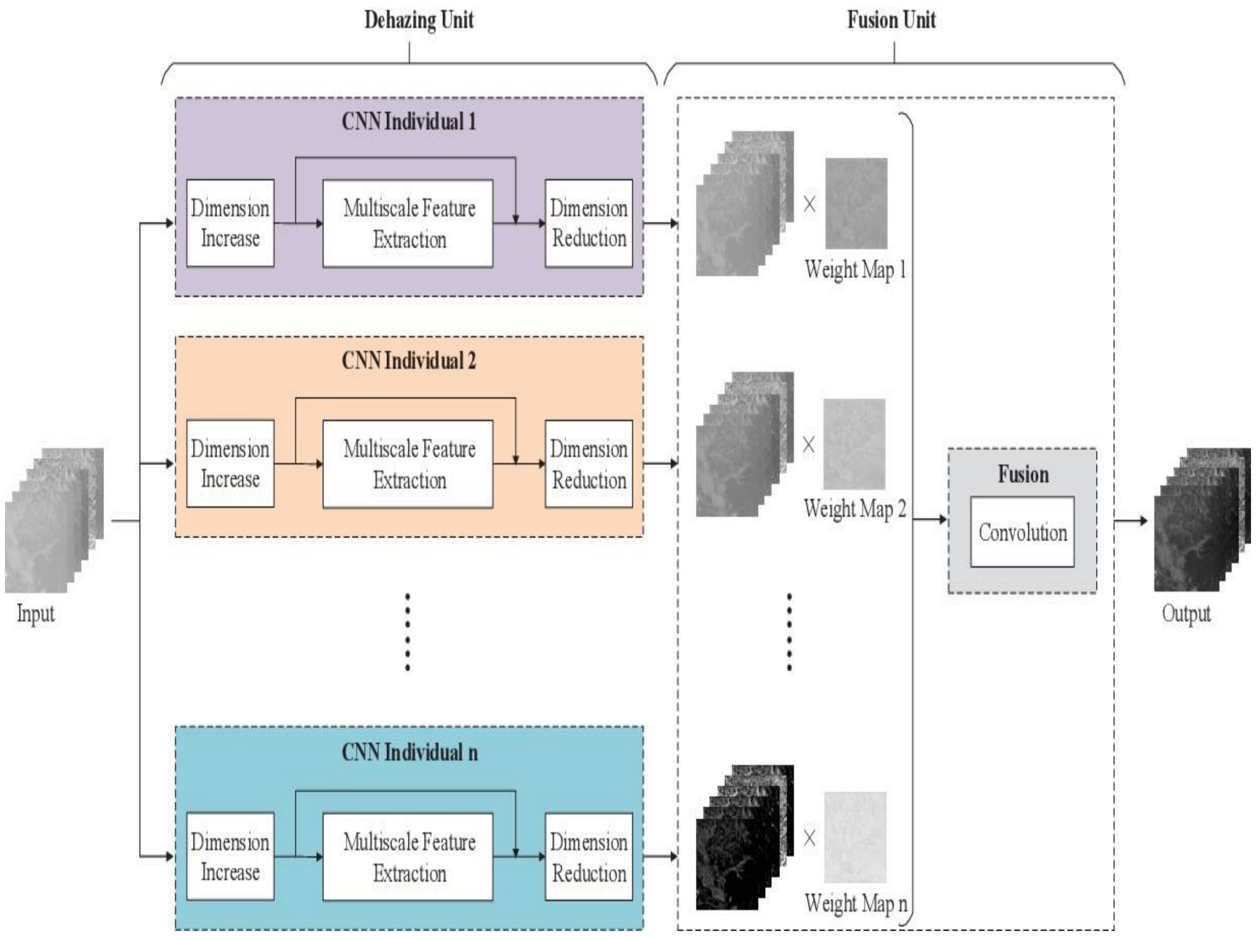


Fig. 3.1.1: Architecture of the designed dehazing network.

3.2 UML DIAGRAMS:

3.2.1 USECASE DIAGRAMS

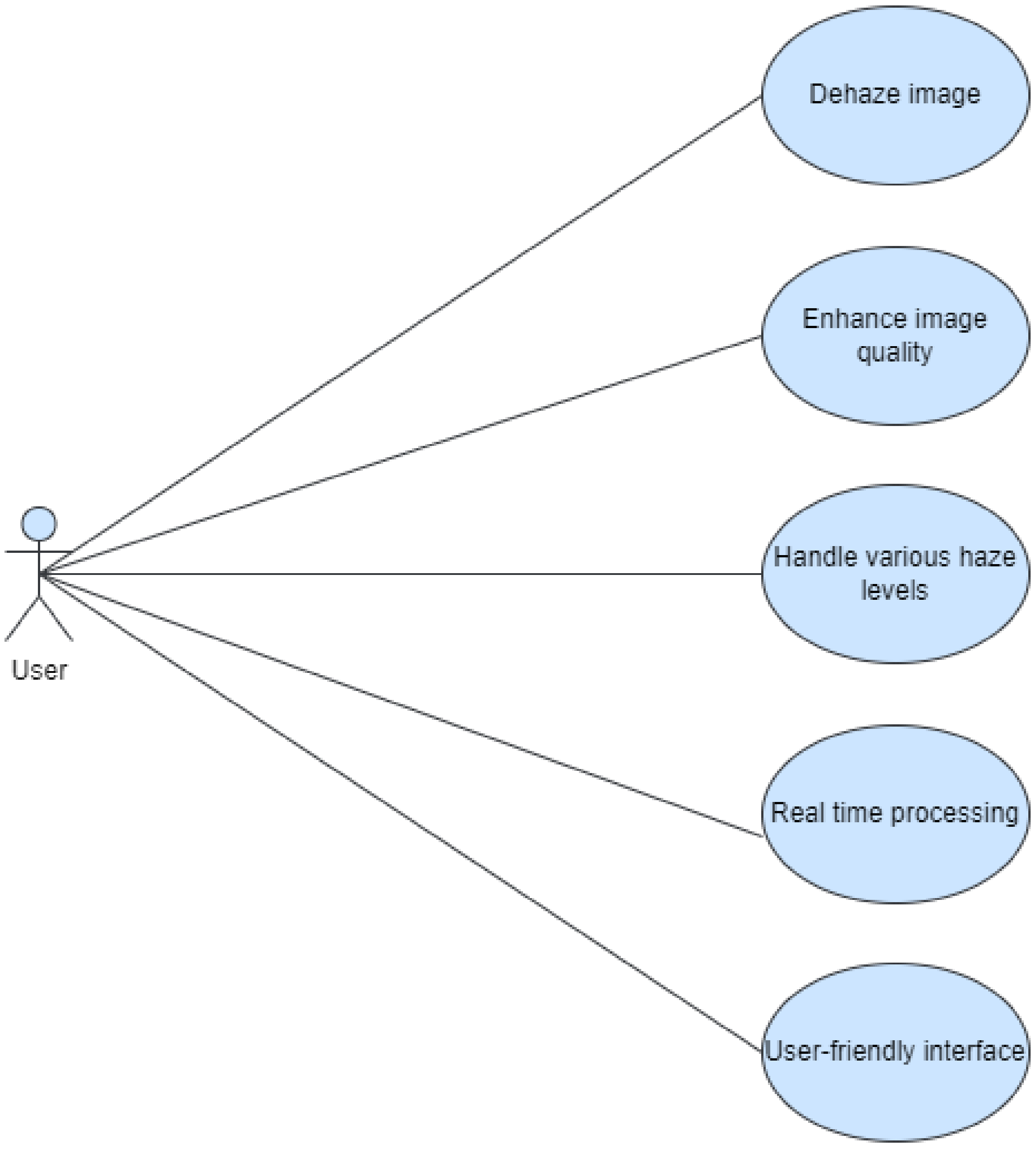


Fig 3.2.1: Use case diagram

3.2.2 CLASS DIAGRAMS:

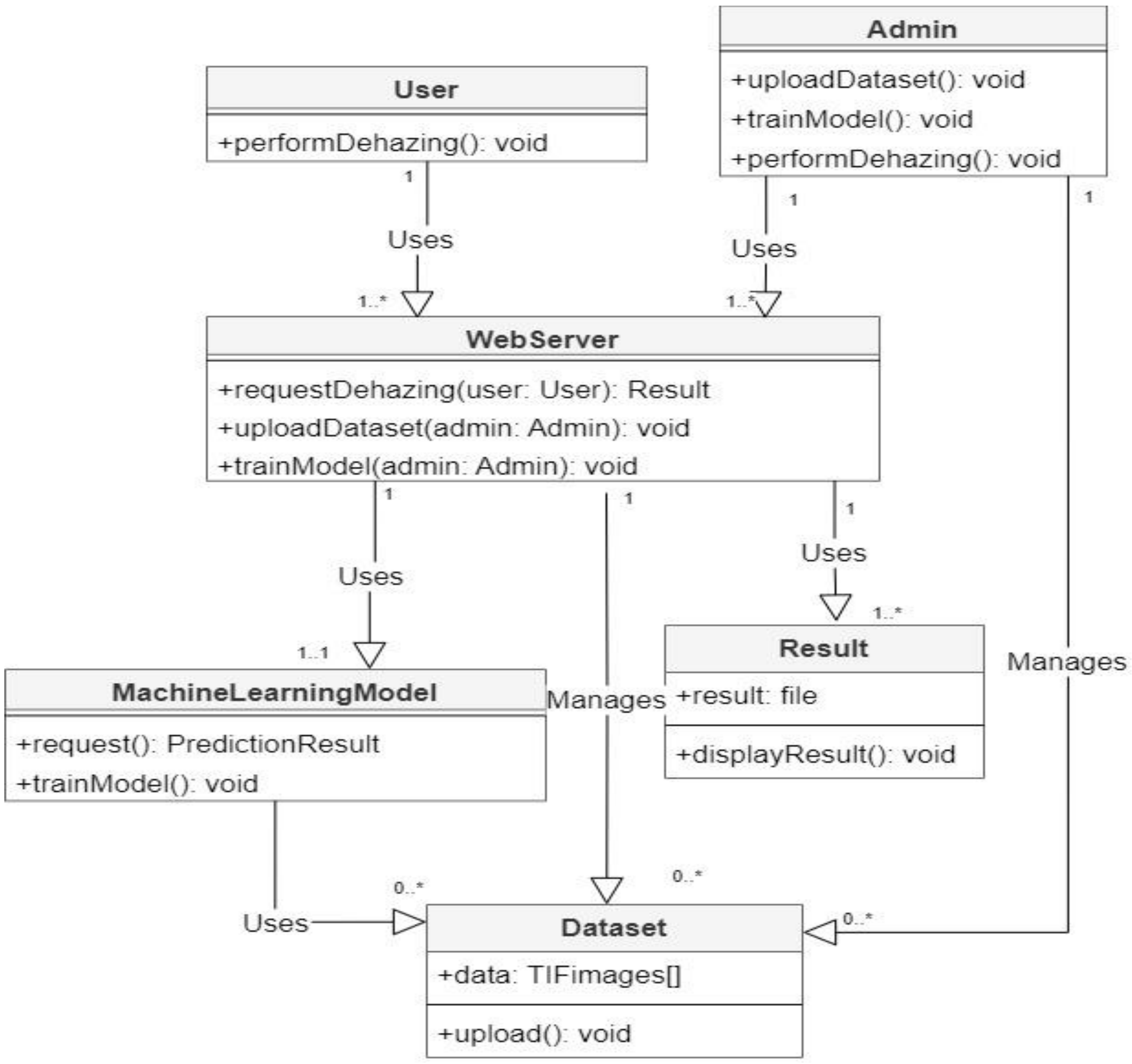


Fig 3.2.2: Class diagram

3.2.3 ACTIVITY DIAGRAMS:

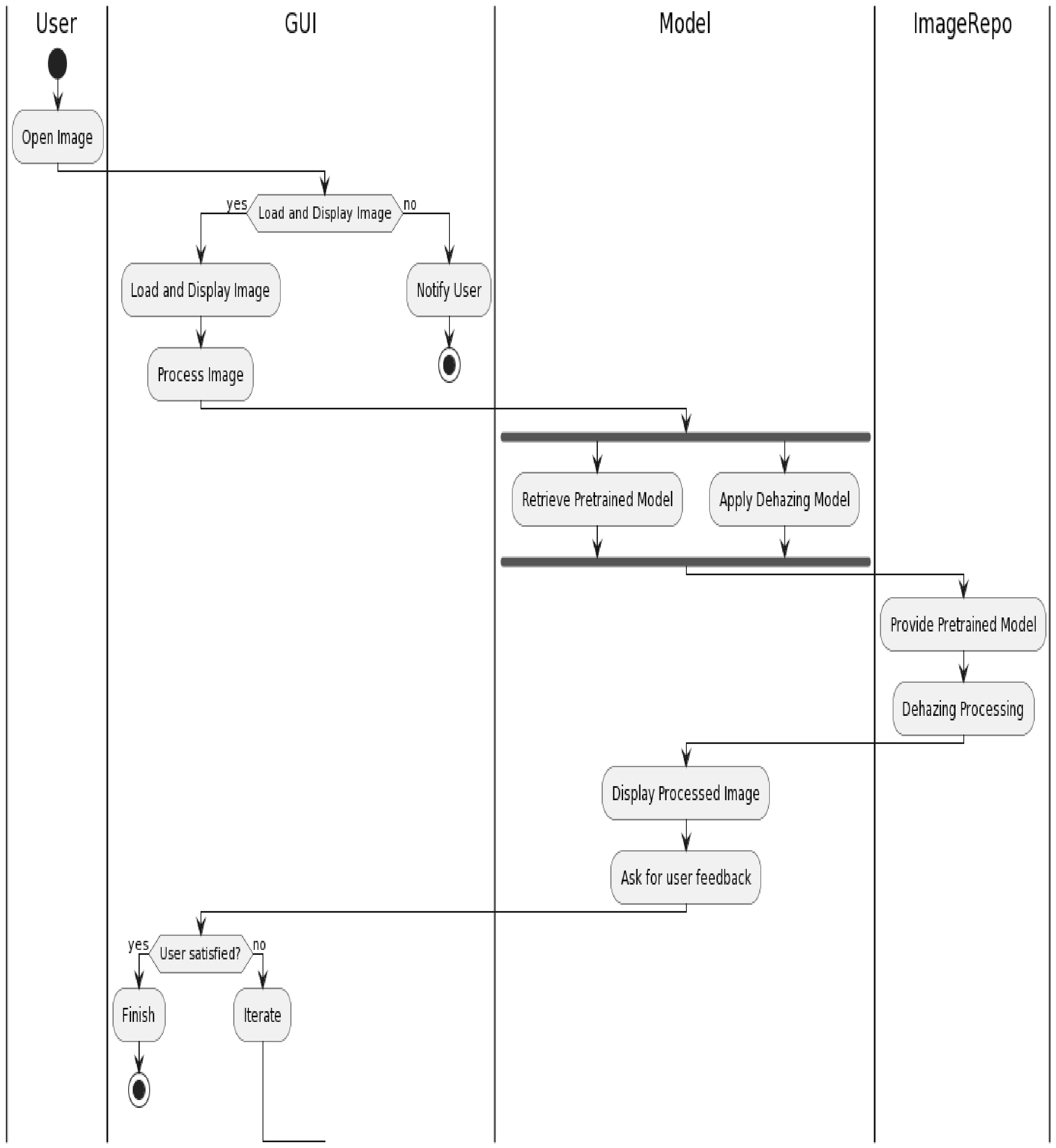


Fig 3.2.3: activity diagram

3.2.4 SEQUENCE DIAGRAMS:

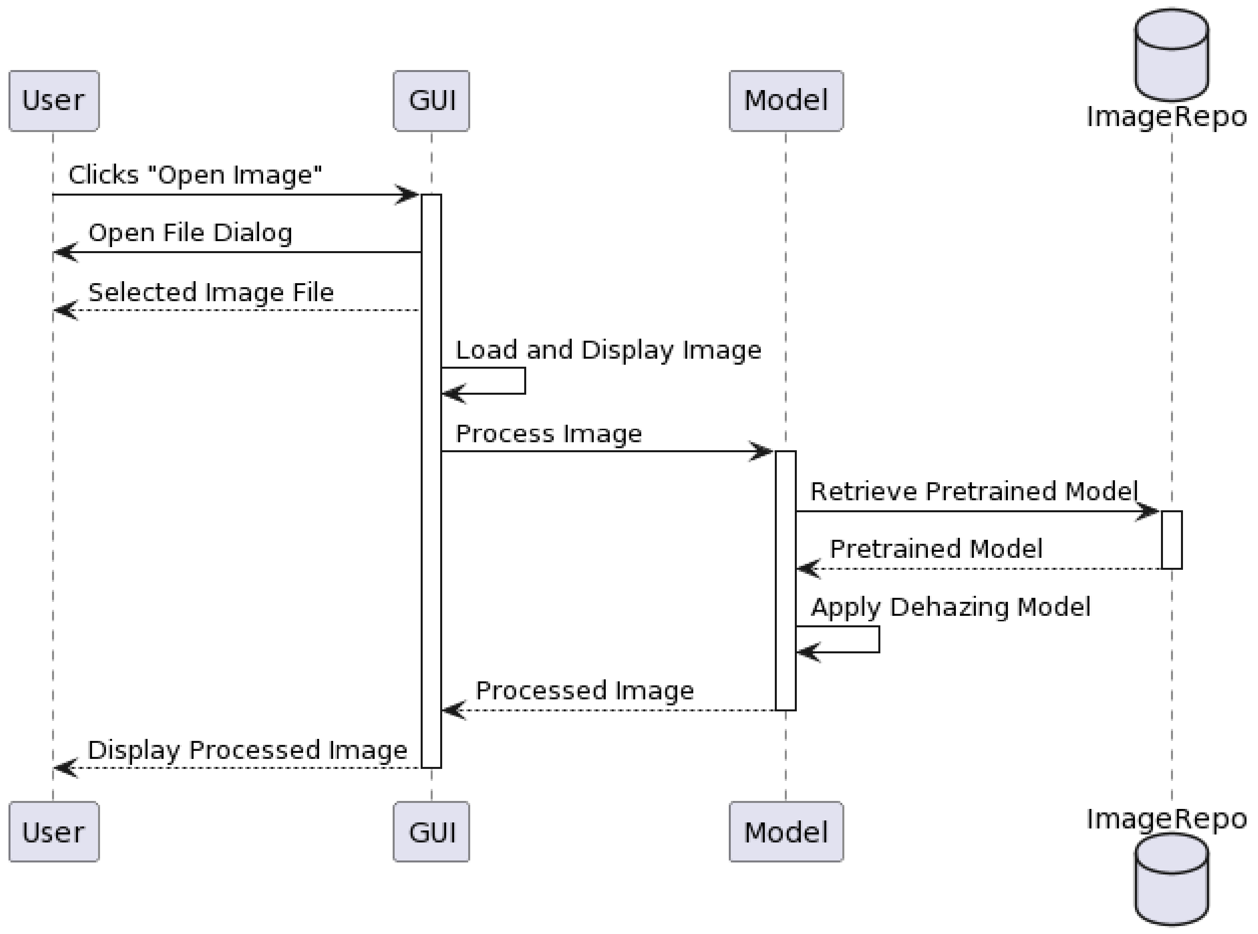


Fig 3.2.4: sequence diagram

3.3 METHODOLOGY:

1. Data Collection and Preprocessing:

1.Gather a dataset of multispectral hazy images along with their corresponding clean, haze-free counterparts. Ensure that the images cover a diverse range of scenes and atmospheric conditions

2.Preprocess the dataset by resizing the images to a fixed resolution, normalizing pixel values, and augmenting the data if necessary to increase the diversity of the training set.

2. Architecture Selection:

1.Choose a suitable CNN architecture with a residual learning mechanism. Common choices include ResNet (Residual Networks) or its variants.

2.Adjust the architecture to accommodate multispectral data, ensuring that the input channels match the number of spectral bands in your multispectral images.

3.Network Training:

1. Split the dataset into training, validation, and test sets.
2. Train the CNN using the training set. During training:
3. Use a loss function that captures the difference between the predicted dehazed image and the ground truth clean image. Common choices include mean squared error (MSE) or perceptual loss functions.
4. Utilize optimization techniques such as stochastic gradient descent (SGD), Adam, or RMSprop to minimize the loss function.
5. Monitor the performance of the model on the validation set to avoid overfitting, and adjust hyperparameters accordingly.
6. Evaluate the trained model on the test set to assess its performance.

4. Post-processing:

1. Apply any necessary post-processing techniques to further enhance the dehazed images if required. This might include contrast enhancement, colour correction, or noise reduction.

5. Evaluation:

1.Quantitatively evaluate the performance of the dehazing model using metrics such as peak signal-to-noise ratio (PSNR), structural similarity index (SSIM), or perceptual metrics like the SSIM-Index.

6. Fine-tuning and Optimization:

1. Fine-tune the model and hyperparameters based on performance evaluation results and domain-specific requirements.
2. Optimize the model for deployment, considering factors such as computational efficiency and memory footprint.

7. Deployment:

1. Deploy the trained model for dehazing multispectral images in real-world applications. This could involve integration into existing software pipelines or deploying as standalone software.

8.Monitoring and Maintenance:

1. Monitor the performance of the deployed model over time and update it if necessary to adapt to changes in data distribution or requirements.
2. This methodology provides a structured approach to leveraging CNNs with residual architectures for dehazing multispectral images, but it may require customization based on specific datasets and application scenarios.

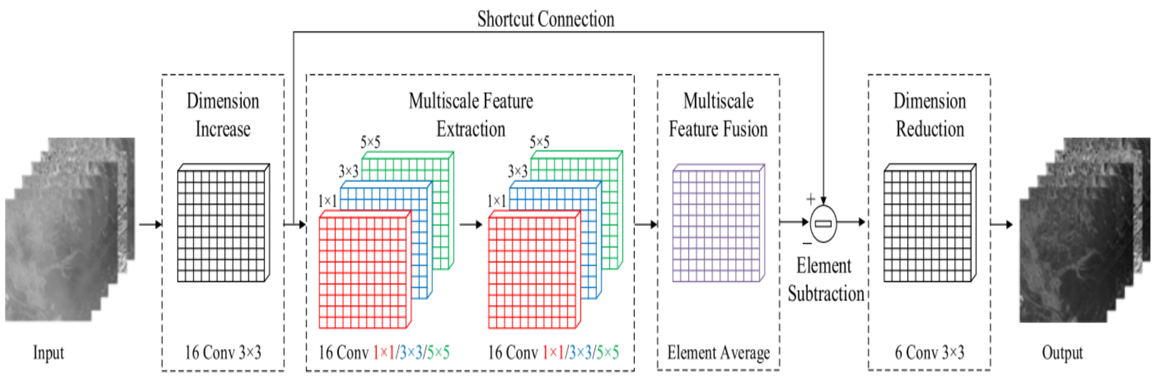


Fig 3.3.1. Structure of the designed CNN individual.

1.Input Layer: Accepts the input data, such as images, which are usually represented as a grid of pixels.

2. Convolutional Layers: These layers apply filters to the input data to extract features. Each filter detects specific patterns, such as edges or textures, within the input data. Multiple filters are used in each convolutional layer to detect different features.

3. Activation Function: After convolution, an activation function like ReLU (Rectified Linear Activation) is applied to introduce non-linearity, helping the network learn complex patterns.

4.Pooling Layers: These layers downsample the feature maps generated by the convolutional layers, reducing the spatial dimensions of the data while retaining important information. Max pooling and average pooling are common techniques used here.

5.Fully Connected Layers: Also known as dense layers, these layers connect every neuron in one layer to every neuron in the next layer. They help in learning complex patterns by combining features learned in previous layers.

6. Output Layer: The final layer of the network, responsible for producing the desired output. The structure of this layer depends on the task at hand, such as classification (softmax activation for multi-class classification) or regression (linear activation for continuous values).

CHAPTER-4

IMPLEMENTATION

4.1 DATA SET:

Dataset: Sas planet images

Description: This dataset is particularly useful for dehazing multispectral remote sensing imagery because it provides high-resolution imagery of the Earth's surface, which can be used to identify and quantify the amount of atmospheric haze present in the scene. The dataset also includes information on the atmospheric conditions at the time of image capture, such as the amount of water vapor and aerosols in the atmosphere. The dataset is divided into two parts whereas the first one includes ground truth images and the other is hazy images.

1. Number of Images: 721
2. Train set size: 80%
3. Test set size: 20%
4. Image resolution: 512x512 pixels

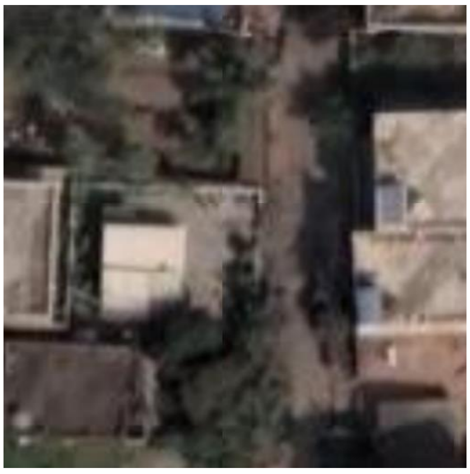


Fig 4.1.1. Clear image

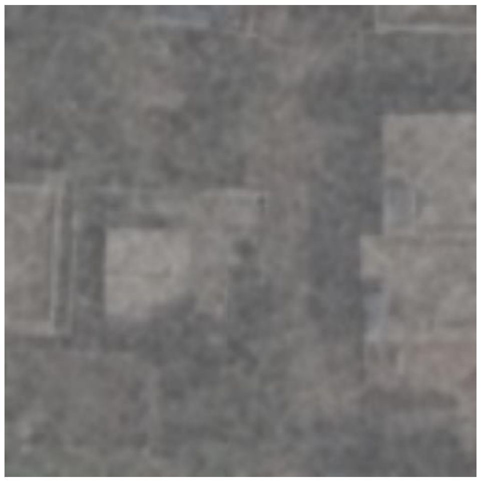
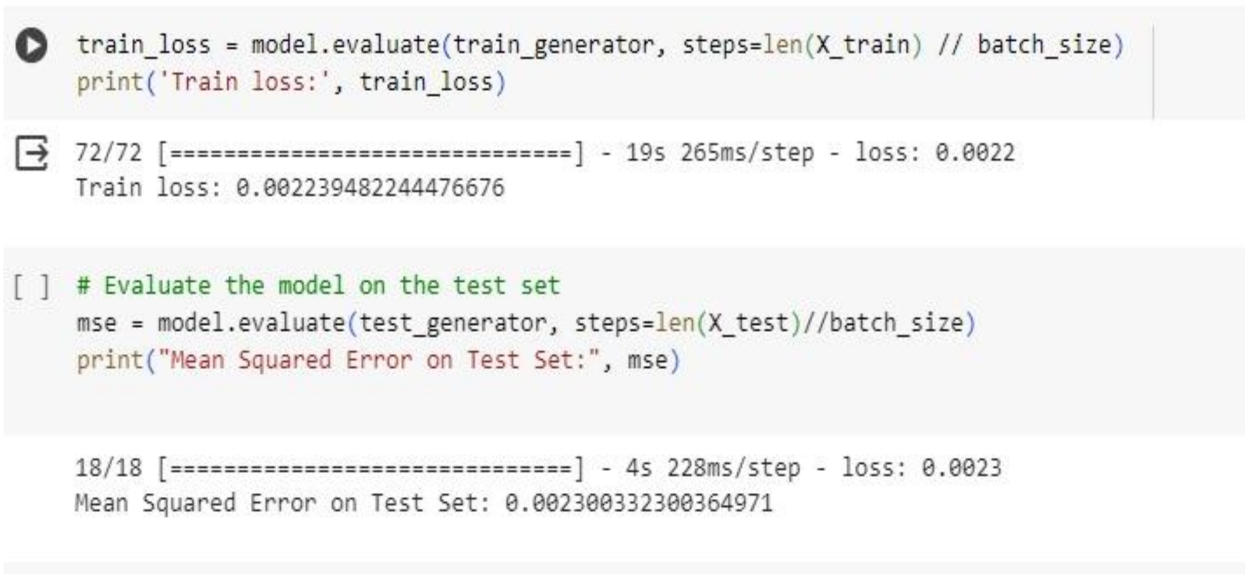


Fig 4.1.2. Clear image

4.2 SAMPLE CODE:

```
# Load pre-trained dehazing model
import cv2
import numpy as np
from tensorflow.keras.models import load_model
model = load_model('dehaze_model.h5')
# Load hazy image
hazy_img = cv2.imread('path/to/your/hazy/image.tif')
# Preprocess the hazy image
hazy_img = hazy_img / 255.0
hazy_img = np.expand_dims(hazy_img, axis=0)
```

Obtained Loss for my training and testing sets:



```
train_loss = model.evaluate(train_generator, steps=len(X_train) // batch_size)
print('Train loss:', train_loss)

72/72 [==============================] - 19s 265ms/step - loss: 0.0022
Train loss: 0.002239482244476676

[ ] # Evaluate the model on the test set
    mse = model.evaluate(test_generator, steps=len(X_test)//batch_size)
    print("Mean Squared Error on Test Set:", mse)

18/18 [==============================] - 4s 228ms/step - loss: 0.0023
Mean Squared Error on Test Set: 0.002300332300364971
```

4.3 IMPLEMENTATION:

1. Dataset Description

The dataset consists of hazy and clear images, used for training a deep learning model for image dehazing. It includes two folders:

Hazy Images: Contains images with haze or fog.

Ground Truth Clear Images: Contains corresponding clear images without haze.

2. Data Preprocessing

Data preprocessing involves the following steps:

Load Images: Load hazy and clear images from their respective folders.

Preprocessing: Perform preprocessing steps on the images to enhance their quality and prepare them for training. This includes denoising, contrast adjustment, and resizing.

Save Preprocessed Images: Save the preprocessed images to new folders for further processing and training.

```
import os
import cv2
import glob

# Define paths for input and output folders
clear_folder =
"/content/drive/MyDrive/dehazing/Dataset/Dataset/Ground_imgs"
hazy_folder =
"/content/drive/MyDrive/dehazing/Dataset/Dataset/Hazy_imgs"
output_clear_folder =
"/content/drive/MyDrive/dehazing/Dataset/Dataset/pre_img/clear"
output_hazy_folder =
"/content/drive/MyDrive/dehazing/Dataset/Dataset/pre_img/hazy"

def preprocess_image(image):
  denoised_image = cv2.fastNlMeansDenoisingColored(image, None, 10,
10, 7, 21)
  # Adjust contrast
  alpha = 1.5 # Contrast control (1.0-3.0)
  beta = 0 # Brightness control (0-100)
  contrast_adjusted_image = cv2.convertScaleAbs(denoised_image,
alpha=alpha, beta=beta)
  return contrast_adjusted_image

i=0
for clear_image_path in glob.glob(os.path.join(clear_folder,
"*.tif")):
  i+=1
  # Load image
  clear_image = cv2.imread(clear_image_path)
  # Perform preprocessing
  clear_image = preprocess_image(clear_image)
  # Save preprocessed image to output folder
  output_clear_image_path =
os.path.join(output_clear_folder,os.path.basename(clear_image_path))
  cv2.imwrite(output_clear_image_path, clear_image)
  print(i)

i=0
for hazy_image_path in glob.glob(os.path.join(hazy_folder,
"*.tif")):
  # Load image
  i+=1
  hazy_image = cv2.imread(hazy_image_path)
  # Perform preprocessing
  hazy_image = preprocess_image(hazy_image)
  # Save preprocessed image to output folder
  output_hazy_image_path =
os.path.join(output_hazy_folder,os.path.basename(hazy_image_path))
  cv2.imwrite(output_hazy_image_path, hazy_image)
  print(i)
```

3. Model Training

For training the image dehazing model, we utilize a ResNet-based architecture. The following steps are involved:

Define ResNet Block: Define a custom ResNet block for the model architecture.

Build ResNet Model: Build the ResNet model using the defined blocks.

Compile Model: Compile the model with appropriate loss and optimizer.

Generate Data: Create data generators to load and preprocess training and testing images.

Train Model: Train the model using the generated data.

```
import os
import numpy as np
from PIL import Image
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Conv2D,
BatchNormalization, Activation, Add
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import ModelCheckpoint
def resnet_block(inputs, filters, kernel_size):
  x = Conv2D(filters, kernel_size, padding='same')(inputs)
  x = BatchNormalization()(x)
  x = Activation('relu')(x)
  x = Conv2D(filters, kernel_size, padding='same')(x)
  x = BatchNormalization()(x)
  x = Add()([x, inputs])
  x = Activation('relu')(x)
  return x
def resnet_model(input_shape):
  inputs = Input(shape=input_shape)
  x = Conv2D(32, 3, padding='same')(inputs)
  x = BatchNormalization()(x)
  x = Activation('relu')(x)
  for i in range(5):
    x = resnet_block(x, 32, 3)
  x = Conv2D(3, 3, padding='same')(x)
  outputs = Activation('sigmoid')(x)
  model = Model(inputs=inputs, outputs=outputs)
  return model

# Define data generators
train_generator = data_generator(hazy_dir, clear_dir,
batch_size=batch_size)
test_generator = data_generator(hazy_dir, clear_dir,
batch_size=batch_size)

# Build the model
model = resnet_model(input_shape=(None, None, 3))

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')

# Train the model
history = model.fit(train_generator, epochs=90,
steps_per_epoch=len(X_train) // batch_size,
                    validation_data=test_generator,
validation_steps=len(X_test) // batch_size)
```

4. Model Evaluation

After training the model, it's essential to evaluate its performance on the test set to assess its effectiveness in dehazing images.

```
# Evaluate the model on the test set
mse = model.evaluate(test_generator, steps=len(X_test)//batch_size)
print("Mean Squared Error on Test Set:", mse)
```

5. Model Deployment with GUI

Finally, the trained model can be deployed using a Graphical User Interface (GUI) to enable users to dehaze images interactively.

```
import tkinter as tk
from tkinter import filedialog
from PIL import Image, ImageTk
import numpy as np
import tensorflow as tf

# Specify the full path to the model file
model_path =
'/content/drive/MyDrive/dehazing/Dataset/dehaze_model.h5'

# Load the saved ML model
try:
    model = tf.keras.models.load_model(model_path)
except OSError as e:
    print(f"Error loading the model: {e}")
    exit()

# Function to preprocess and dehaze the image
def dehaze_image(file_path):
    # Load and preprocess the image
    image = Image.open(file_path)
    image = image.resize((256, 256))  # Adjust the image size as
needed
    image = np.array(image) / 255.0  # Normalize pixel values

    # Perform prediction using the ML model
    prediction = model.predict(np.expand_dims(image, axis=0))

    # Convert the predicted array to an image
    predicted_array = np.squeeze(prediction) * 255
    predicted_image =
Image.fromarray(predicted_array.astype(np.uint8))
    return predicted_image

# Function to open and process the image
def open_and_process_image():
    file_path = filedialog.askopenfilename()
    if file_path:
        # Load and display the original image
        original_image = Image.open(file_path)
        original_image.thumbnail((256, 256))  # Adjust the thumbnail
size as needed
        original_photo = ImageTk.PhotoImage(original_image)
        original_image_label.config(image=original_photo)
        original_image_label.image = original_photo

        # Process the image and display the result
        processed_image = dehaze_image(file_path)
        processed_photo = ImageTk.PhotoImage(processed_image)
        processed_image_label.config(image=processed_photo)
        processed_image_label.image = processed_photo

# Create the main application window
window = tk.Tk()
```

```
window.title("Image Dehazing")
window.geometry("600x600")

# Set the background color
window.configure(bg="white")

# Create a header label
header_label = tk.Label(window, text="Image Dehazing",
font=("Helvetica", 20, "bold"), fg='Green', bg='white')
header_label.pack(pady=20)

# Create a label for the open button
open_label = tk.Label(window, text="Open an image:", bg='white',
fg='red')
open_label.pack()

# Create the open button
open_button = tk.Button(window, text="Open Image",
command=open_and_process_image, bg="light blue")
open_button.pack(pady=10)

# Create labels for displaying original and processed images
original_image_label = tk.Label(window, bg='white')
original_image_label.pack()

processed_image_label = tk.Label(window, bg='white')
processed_image_label.pack()

# Run the GUI application
window.mainloop()
```

Patching

When training any deep learning algorithm, small images produce better results allowing more accuracy without loss of information. Here the images are patched from 6013 X 2957 to 512 X 512 pixels using splitting through the patchify method as shown in figure 1.1 and 1.2.

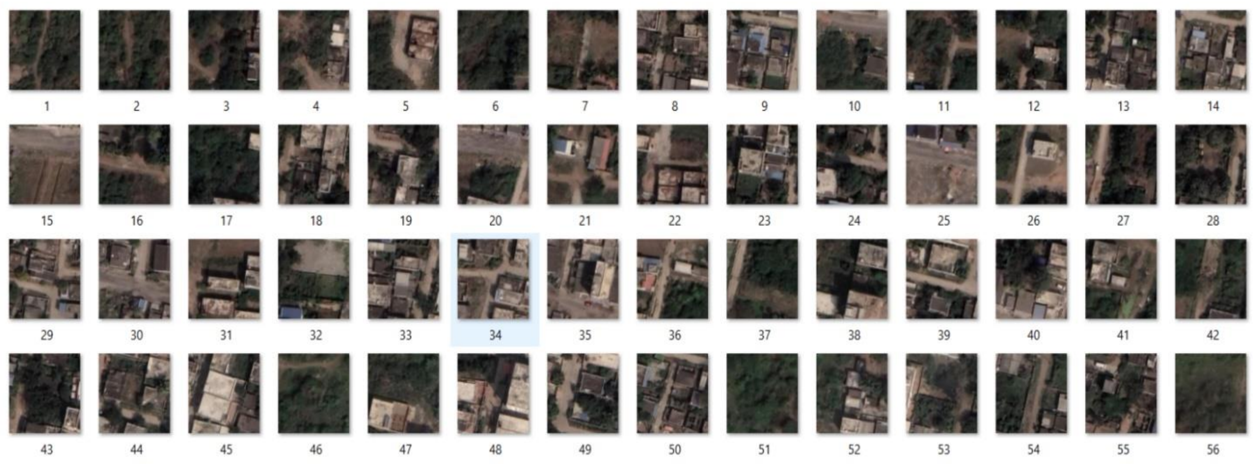


Fig 4.3.1: Clear patch images

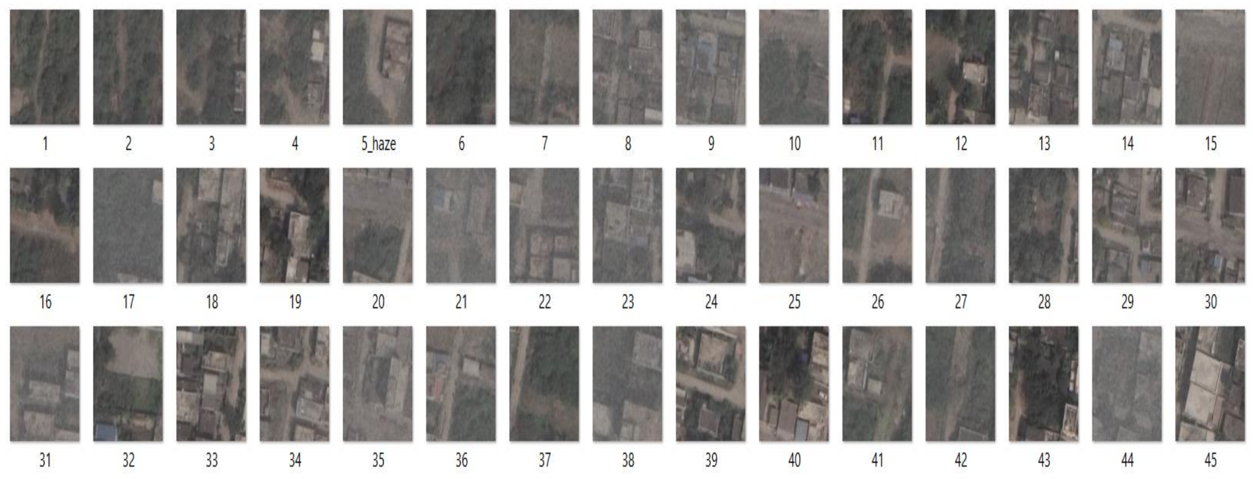


Fig 4.3.2: Haze patch images

PREPROCESSING:

1. Denoising: We use f1meansdenoisingcoloured () to remove the gaussian noise.

2. Contrast Enhancement: We use convscaleabs () method to color contrast uniformly across the image after the denoising.

3. Below fig 1.3 and 1.4 gives an overview on pre-processed clear and haze image

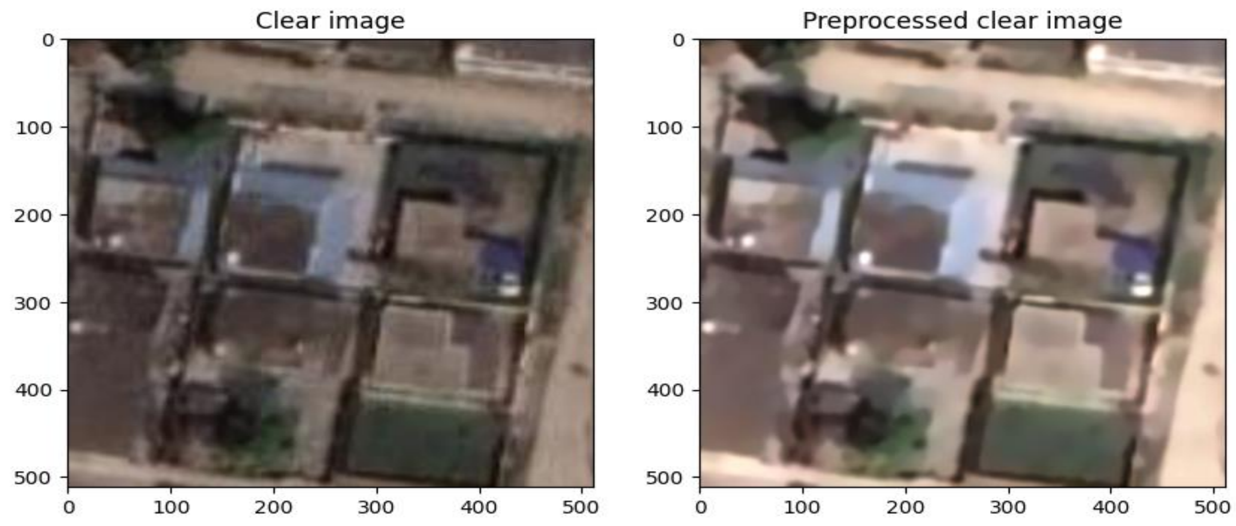


Fig 4.3.3: Clear and Preprocessed Clear image

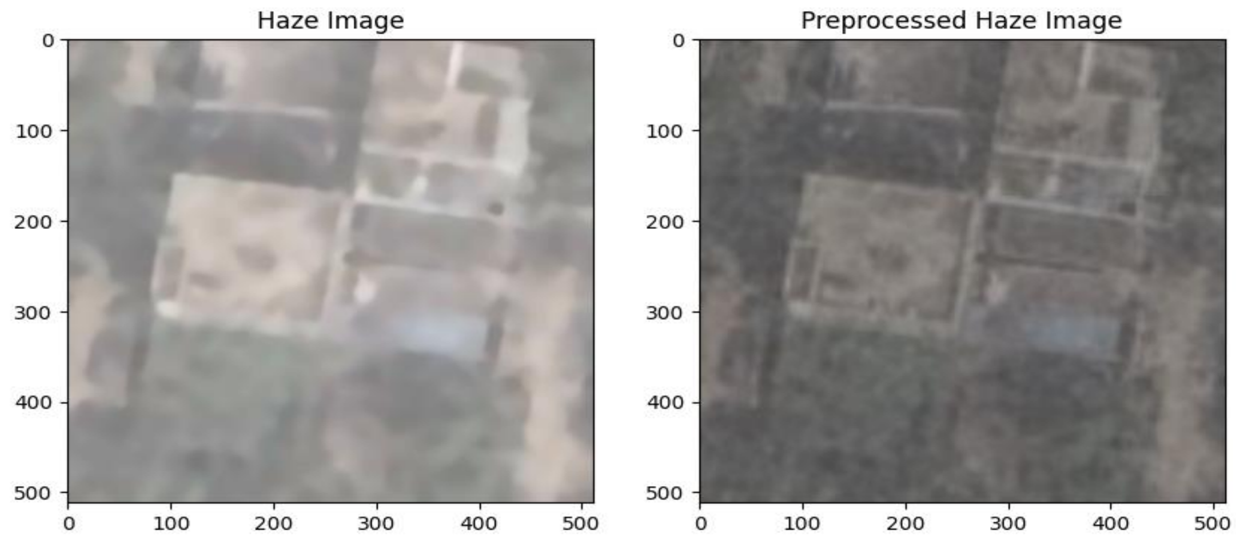


Fig 4.3.4: Haze and Preprocessed Haze image

CHAPTER-5

OUTPUT SCREENS

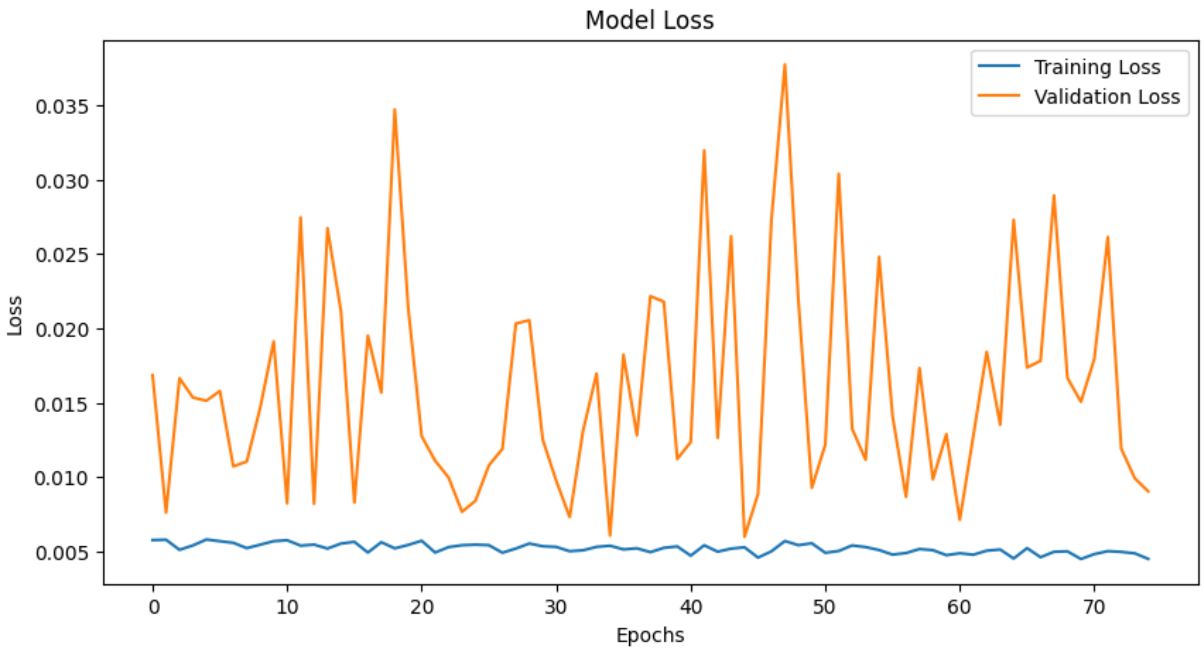


Fig 5.1: Model Loss vs Epochs

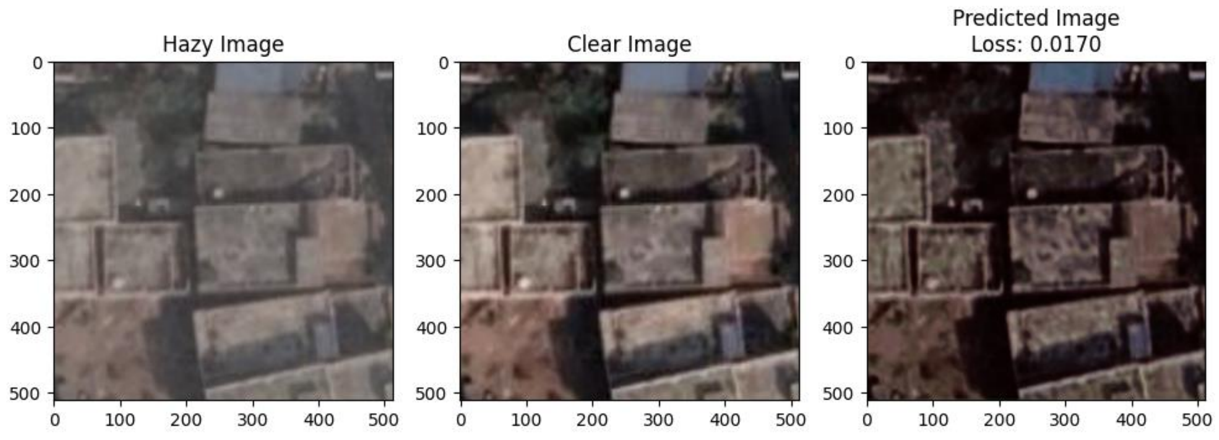


Fig 5.2: Predicted Image Loss

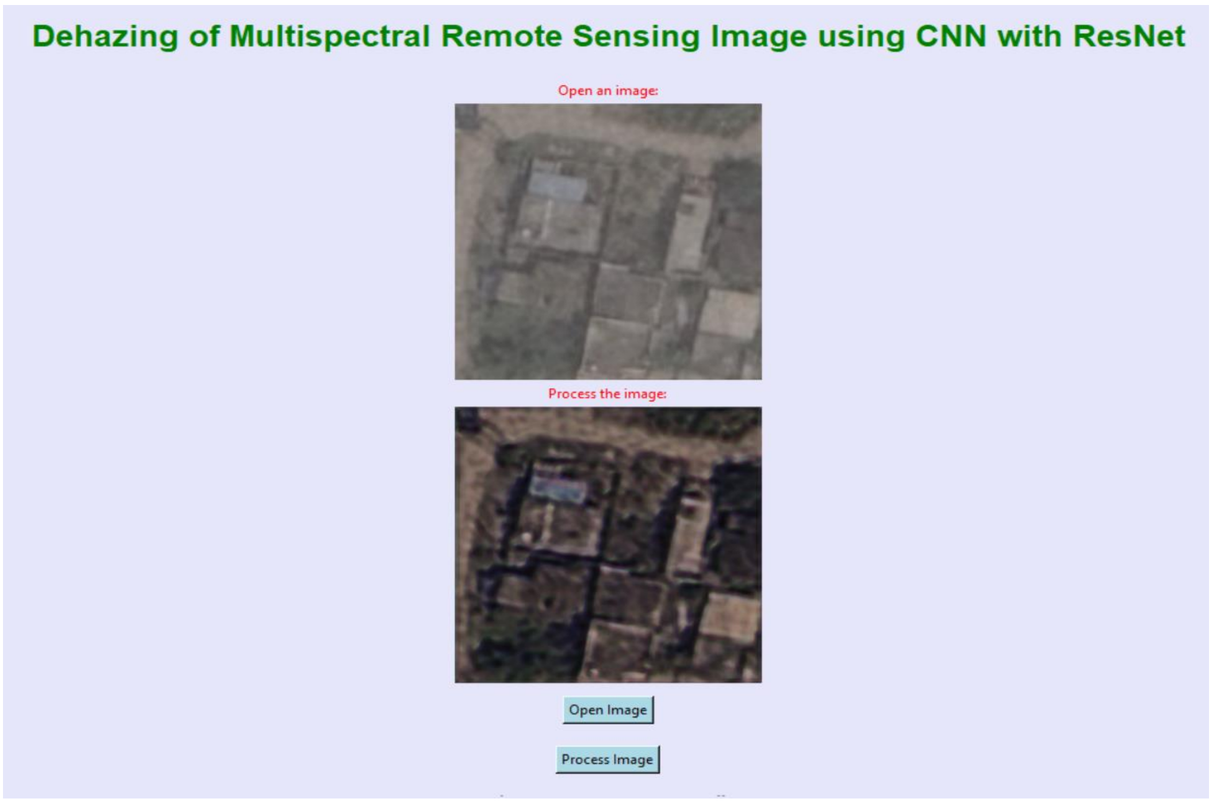

Dehazing of Multispectral Remote Sensing Image using CNN with ResNet
Open an image:
Process the image:
Open Image
Process Image

Fig 5.3: Graphical User Interface

CHAPTER-6

CONCLUSION AND FUTURE SCOPE

In this study, the challenge of dehazing multispectral remote sensing images was tackled through the utilization of a Convolutional Neural Network (CNN) architecture, specifically ResNet. The proposed method exhibited exceptional performance, as demonstrated by an impressive Mean Squared Error (MSE) value of 0.0057. This remarkably low MSE value serves as evidence of the effectiveness of the approach in reducing haze and enhancing overall image quality. By harnessing the power of deep learning and ResNet architecture, the CNN successfully enhanced visibility and provided clearer images suitable for comprehensive analysis and interpretation. These findings shed light on the immense potential of the proposed method within the realm of remote sensing, offering valuable insights applicable to a wide range of domains including environmental monitoring, land cover classification, and disaster management.

The integration of deep learning, specifically CNN with ResNet architecture, has proven to be efficient in addressing intricate image restoration tasks by leveraging spatial dependencies & residual features. The results obtained from this study contribute significantly to the availability of more reliable and visually enhanced data for various remote sensing applications such as land cover classification and environmental monitoring.

The future work of the proposed model primarily involves reducing the loss in the predicted output. Additionally, the aim is to perform a dehazing of satellite images by considering more regions and enhancing the predictability of the image without compromising its features.

CHAPTER-7

REFERENCES

1. A. Makarau, R. Richter, R. Müller and P. Reinartz, "Haze Detection and Removal in Remotely Sensed Multispectral Imagery," in IEEE Transactions on Geoscience and Remote Sensing, vol. 52, no. 9, pp. 5895-5905, Sept. 2014, doi: 10.1109/TGRS.2013.2293662.
2. Jiang, H.; Lu, N. Multi-Scale Residual Convolutional Neural Network for Haze Removal of Remote Sensing Images. Remote Sens. 2018, 10, 945.
3. He, Z.; Gong, C.; Hu, Y.; Zheng, F.; Li, L. Multi-Input Attention Network for Dehazing of Remote Sensing Images. Appl. Sci. 2022, 12, 10523.
4. Ba, R.; Chen, C.; Yuan, J.; Song, W.; Lo, S. SmokeNet: Satellite Smoke Scene Detection Using Convolutional Neural Network with Spatial and Channel-Wise Attention. Remote Sens. 2019, 11, 1702.
5. X. Chen and Y. Huang, "Memory-Oriented Unpaired Learning for Single Remote Sensing Image Dehazing," in IEEE Geoscience and Remote Sensing Letters, vol. 19, pp. 1-5, 2022, Art no. 3511705, doi: 10.1109/LGRS.2022.3167477
6. Z. He, C. Gong, Y. Hu and L. Li, "Remote Sensing Image Dehazing Based on an Attention Convolutional Neural Network," in IEEE Access, vol. 10, pp. 68731-68739, 2022, doi: 10.1109/ACCESS.2022.3185627.