A Project Stage-1 Report on

LOW LIGHT IMAGE ENHANCEMENT

Submitted to

Jawaharlal Nehru Technological University, Hyderabad

in partial fulfillment of requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

P.Dharan Tej(19BD1A057F)

S.Sudheendra (19BD1A057N)

D.Rohit(19BD1A056D)

C.Ajay(19BD1A0569)

Under the guidance of

Dr.Y.Alekya Rani
Assistant Professor
Department of CSE

Department of Computer Science and Engineering

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

Approved by AICTE, Affiliated to JNTUH

3-5-1206, Narayanaguda, Hyderabad – 500029

2022-2023

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

CERTIFICATE

This is to certify that the project entitled **Low Light Image Enhancement** being submitted by

P.Dharan Tej	**(19BD1A057F)**
S.Sudheendra	**(19BD1A057N)**
D.Rohit	**(19BD1A056D)**
C.Ajay	**(19BD1A0569)**

In partial fulfilment for the award of **Bachelor of Technology** in **Computer Science and Engineering** affiliated to the **Jawaharlal Nehru Technological University, Hyderabad** during the year 2022-23**.**

Internal Guide	**Head of the Department**
(Dr. Y. Alekya Rani)	**(Dr. S. Padmaja)**

Submitted for Viva Voce Examination held on _____

External Examiner

Vision of KMIT

Producing quality graduates trained in the latest technologies and related tools and striving to make India a world leader in software and hardware products and services. To achieve academic excellence by imparting in depth knowledge to the students, facilitating research activities and catering to the fast growing and ever- changing industrial demands and societal needs.

Mission of KMIT

- To provide a learning environment that inculcates problem solving skills, professional, ethical responsibilities, lifelong learning through multi modal platforms and prepare students to become successful professionals.
- To establish industry institute Interaction to make students ready for the industry.
- To provide exposure to students on latest hardware and software tools.
- To promote research based projects/act
- ivities in the emerging areas of technology convergence.
- To encourage and enable students to not merely seek jobs from the industry but also to create new enterprises.
- To induce a spirit of nationalism which will enable the student to develop, understand lndia's challenges and to encourage them to develop effective solutions.
- To support the faculty to accelerate their learning curve to deliver excellent service to students.

Vision & Mission of CSE

Vision of the CSE

To be among the region's premier teaching and research Computer Science and Engineering departments producing globally competent and socially responsible graduates in the most conducive academic environment.

Mission of the CSE

- To provide faculty with state of the art facilities for continuous professional development and research, both in foundational aspects and of relevance to emerging computing trends.
- To impart skills that transform students to develop technical solutions for societal needs and inculcate entrepreneurial talents.
- To inculcate an ability in students to pursue the advancement of knowledge in various specializations of Computer Science and Engineering and make them industry-ready.
- To engage in collaborative research with academia and industry and generate adequate resources for research activities for seamless transfer of knowledge resulting in sponsored projects and consultancy.
- To cultivate responsibility through sharing of know
- ledge and innovative computing solutions that benefit the society-at-large.
- To collaborate with academia, industry and community to set high standards in academic excellence and in fulfilling societal responsibilities.

PROGRAM OUTCOMES (POs)

1. Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis: Identify formulate, review research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences

3. Design/development of solutions: Design solutions for complex engineering problem and design system component or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern tool usage: Create select, and, apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

6. The engineer and society: Apply reasoning informed by the contextual knowledge to societal, health, safety. legal und cultural issues and the consequent responsibilities relevant to professional engineering practice.

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams and in multidisciplinary settings.

10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO1: An ability to analyze the common business functions to design and develop appropriate Computer Science solutions for social upliftment's.

PSO2: Shall have expertise on the evolving technologies like Mobile Apps, CRM, ERP, Big Data, etc.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Graduates will have successful careers in computer related engineering fields or will be able to successfully pursue advanced higher education degrees.

PEO2: Graduates will try and provide solutions to challenging problems in their profession by applying computer engineering principles.

PEO3: Graduates will engage in life-long learning and professional development by rapidly adapting changing work environment.

PEO4: Graduates will communicate effectively, work collaboratively and exhibit high levels of professionalism and ethical responsibility.

PROJECT OUTCOMES

P1: To provide a friendly environment to the user

P2: Accurate results of statistics are provided

P3: Clearly mention the percentage change in values

P4: Generate report for further reference

LOW - 1
MEDIUM - 2
HIGH - 3

PROJECT OUTCOMES MAPPING PROGRAM OUTCOMES

PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
P1	2	3			3	3			2			2
P2	2	2		1	2	3						2
P3					2			1		2		1
P4	2			1	2	1		3		1		1

PROJECT OUTCOMES MAPPING PROGRAM SPECIFIC OUTCOMES

PSO	PSO1	PSO2
P1	1	3
P2		3
P3	2	3
P4	2	3

PROJECT OUTCOMES MAPPING PROGRAM EDUCATIONAL OBJECTIVES

PEO	PEO1	PEO2	PEO3	PEO4
P1	3			2
P2	3	3		
P3	1	3	3	2
P4	2	3		2

DECLARATION

We hereby declare that the project report entitled "**Low Light Image Enhancement"** is done in the partial fulfillment for the award of the Degree in Bachelor of Technology in Computer Science and Engineering affiliated to Jawaharlal Nehru Technological University, Hyderabad. This project has not been submitted anywhere else.

P.DHARAN TEJ (19BD1A057F)

S.SUDHEENRA (19BD1A057N)

D.ROHIT (19BD1A056D)

C.AJAY(19BD1A0569)

ACKNOWLEDGEMENT

We take this opportunity to thank all the people who have rendered their full support to our project work.

We render our thanks to **Dr. Maheshwar Dutta**, B.E., M Tech., Ph.D., Principal who encouraged us to do this project

We are grateful to **Mr. Neil Gogte**, Director for facilitating all the amenities required for carrying out this project.

We express our sincere gratitude to **Mr. S. Nitin**, Director and **Dr. D. Jaya Prakash**, Dean Academics for providing an excellent environment in the college.

We are also thankful to **Dr. S. Padmaja**, Head of the Department for providing us with both time and amenities to make this project a success within the given schedule.

We are also thankful to our guide **Dr.Y.Alekya Rani**, for her valuable guidance and encouragement given to us throughout the project work.

We would like to thank the entire CSE Department faculty, who helped us directly and indirectly in the completion of the project. We sincerely thank our friends and family for their constant motivation during the project work.

CONTENT

ABSTRACT

Low-light image enhancement is generally regarded as a challenging task in image processing due to low photon count and low SNR. This can often when images are captured at night time or images taken in low-light conditions. As various factors pertaining to the images such as contrast, sharpness and colour co-ordination should be handled simultaneously and effectively to obtain an image of clear quality, the task of image enhancement becomes difficult task for humans to accomplish. Although a variety of existing techniques such as denoising, deblurring, and other enhancement techniques have been proposed, their efficiency is limited in extreme conditions, such as video-rate imaging at night. One such technique which has proved efficient in solving this problem is using a pipeline neural network and a convolutional neural network. The CNN algorithm uses a greater number of hidden layers in identifying features of the image in the pixel level, and the process of repairing in the pixel level will increase the intensity of the image and thus increases the clarity of the image. The proposed model using CNN increases the intensity accuracy percentage by 10–15%. The results demonstrate that the CNN based method outperforms other contrast enhancement methods. This documentation highlights the performance of the model, the challenges and opportunities for future work.

LIST OF FIGURES

CHAPTER -1

1. INTRODUCTION

1.1 Purpose of the Project

Image enhancement is one of the modern topics that are discussed now a days and Deep Learning play a major part in it. We see various types of smart phones with their night mode in camera which help us to take beautiful images at night or low light situation. The Camera is not able to take the perfect image at low light because of the noise created at the camera sensors. During night mode the noise is cancelled out by the camera to get the perfect image.Good quality images and videos are important for many tasks. However, not all images are in good qualities because they are captured in various light conditions. When an image is capture in a poor light environment the pixel values are in a low-value range, which will cause image quality to reduce apparently. Since the whole image appears very dark, it's hard to identify objects or textures clearly. So, it is very important to improve the quality of low-light images. Low light image enhancement is needed in many computer vision tasks for object detection and scene understanding. Sometimes there is a condition when image captured through satellite or in low light always suffer from very low contrast and brightness which increases the difficulty of subsequent high level task in great extent. Low light image enhancement using convolutional neural network system takes dark images as input and produces bright images as an output without disturbing the content of the image. So understanding the scene captured through image becomes easier task.

1.2 Problem with existing system

In daily life, people receive information from images, music, videos, etc., and the human brain is capable of effectively processing such visual information. In the modern age of smart phones in which social media is so popular, many people have become interested in capturing and sharing photos. The photos captured on professional or mobile phone cameras are impacted by various weather conditions, which influence the image quality. Thus, the important contents of an image are not always clearly visible. The conditions that most often lead to the degradation of such image quality include bad weather, low illumination, and moving objects, among many others. The influence of such conditions on the image quality can make it difficult for the human eye to clearly identify the contents of the image. Images with clear visibility tend to depict more details, and the useful contents of the image can be more easily identified.

Enhancement techniques are often used to make the hidden content of images visible, and they aim to facilitate the usability of valuable information for human and computers alike.

Thus, image enhancement is one of the fundamental research topics in image processing. To remove darkness and extract meaningful contents are very important tasks in applications such as medical imaging, object tracking, face detection, facial attractiveness, and object detection. Therefore, such enhancement techniques play an important role in many fields.

1.3 Proposed System

In the proposed system backpropagation algorithm is used which consists of three hidden layers namely 1) Gamma correction, 2) Different convolutional layers, 3) Color Restoration. Following is the brief description of proposed system and algorithm. The Backpropagation algorithm looks for the minimum value of the error function in weight space using a technique called the delta rule or gradient descent. The weights that minimize the error function is then considered to be a solution to the learning problem. While constructing a Neural Network, at the initial stage, we initialize weights with some random values or any variable for that fact. It's not necessary that whatever values of the weights we have selected need to be correct or it fits the model best. We have selected some weight values in the beginning, but our model output is somewhat different than the actual output i.e. the error value is huge. To reduce the errors, we need to explain the model to change the weights, such that error becomes minimum. One way to train our model is using a technique called as Backpropagation. Gamma correction is a method which is also known as the Power Law Transform. First, our image pixel intensities must be scaled in the range [0, 255] to [0, 1.0]. From there, we obtain our output is now a gamma corrected image by applying the following equation: O = I ^ (1 / G) Where I is our input image and G is our gamma value. The output image O is then scaled back to the range [0, 255]. Gamma values < 1 will shift the image towards the darker end of the spectrum while gamma values > 1 will make the image appear lighter. A gamma value of G=1 will have no changes on the input image.

The proposed model is based on multi-scale Retinex. This network designed to find the residual image by finding the mean square error between low/normal-light images using CNN. The final enhanced image can be compute by adding the residual image, which obtained from the convolution neural network, to the original input low-light image. Fig (1) illustrates the

proposed model residual structure. This model consisted from eleven layers connected to each other, As much as the network deeper means that proper perceiving of edges and texture of the input image. Consequently, results of good output image, every layer of them perform its own function in enhancement process. These layers will describe in the next sections. Network architecture can be considered as the main phase of designing the proposed model.

It is the first step of proposed model architecture, to extract the features from input images. This carried out by using convolutional layer with filter kernel size of [3x3],and filters number are 200 filters , which resulting of 200 various features maps for a single input image.

By using convolutional layer with kernel size of [3x3] and the numbers of filters are 128, to map the low-light image to normal-light image.

Mapping between high dimensional and high dimensional vector this carried out by convolution layer with filter kernel size of [3x3], filters number are 32 filters. After that, convolutional layer followed by Relu.

Then to increase PSNR value, batch normalization layer used. High dimensional vector from the previous layer reconstructed into three dimensional vectors (channels), this carried out by convolution layer kernel size of [3x3], and filters number are 3 filters. After that, convolutional layer is followed by Relu.

1.4 Scope of the project

Generally, the image enhancement technique improves the visual quality of the low-quality low-light images. It supports further high-level computer vision tasks to extract valuable information from the captured images. The low-light image enhancement technique unveils the poorly exposed regions of the image. Many kinds of research have been proposed to improve the visibility of images regardless of the lighting conditions.

Based on the retinex theory, several models were proposed for low-light image enhancement. Diverse methods were proposed to separate the illumination component from the given image. SSR, MSR, and MSR with color restoration techniques have been popularly employed as image enhancement techniques. Guo calculated the illumination of all the pixels by choosing the greatest value among RGB channels. The work non-uniformly enhanced the illumination instead of eliminating the color shift posed by light sources. Though these methods successfully extract the illumination invariant features in the estimated reflectance, the over-highlighted

edges and color inconsistency often make the enhanced images visually unnatural and significantly degrades the viewing experience. To overcome these issues, methods have been proposed to apply the dehazing algorithm. Dong proposed a dehazing based image enhancement algorithm. The method shows the illumination element of the image the inverted low-light version of the image. The enhanced image is acquired by inverting the unrealistic estimated image once again. The researcher proposed a fusion-based method where an illumination estimation algorithm to decompose the image into illumination and reflectance components. Then, contrast-enhanced and illumination-improved versions are derived from the illumination component by utilizing adaptive histogram equalization and sigmoid function. Although the fusion-based method provided promising results, it struggles to produce textual details across multiple scales.

The main focus of the histogram equalization algorithms is enhancing image contrast and balancing the histogram of the whole image as much as possible. It is frequently used as a prerequisite for object recognition and detection. However, the details hidden in the dark region are not enhanced properly as this method considers each image pixel individually, disregarding their neighborhood. This makes the image inconsistent with the real scene. Celik tried to overcome these issues by proposing variational methods that use various regularization terms on the histogram. The authors enhanced the contrast using the variational methods that mapped the elements between the histogram of the actual and the smoothened image. However, several dehazing methods can provide only reasonable results. Deep learning based Unsupervised GAN proposed by Yifan trains the model without image pairs captured under low-light and normal light. The authors used the information obtained from the input instead of using the ground truth image. Ning proposed a component GAN model to recover normal images from poor light images. The proposed network is composed of two components namely decomposition part and enhancement part. The decomposition part divides the low light images into illumination and reflectance component. The enhancement part generates good quality images.

Several techniques have been put forward for low-light image enhancement, but it still suffers from the over or under-enhanced images due to poor image visibility. Recently studies have been proposed to preserve the illumination structure by minimizing the color effects. Still, these methods suffer from loss of textural details resulting in smoothed out surfaces of objects. In contrast, the proposed model has a clear physical explanation preserving textural details of the

objects in the image. Technical aspects of the LIMET model will be discussed below. The below table shows the advantages and disadvantages of low-light image enhancement techniques.

References	Method	Dataset	Advantages	Disadvantages
Petro et al. (2014)	Multiscale retinex	Dataset collected from multiple sources	Images are enhanced under varied illumination conditions. Color restoration restores wrong colors	The method introduces artifacts and noises in the image resulting in a blurry image
Xiaojie Guo et al. (2016)	LIME: Low light image enhancement *via* illumination map estimation	High-dynamic range dataset	The algorithm is computationally inexpensive	The bright regions are over-enhanced and lose contrast
Li and Wu (2017)	Learning-based restoration of backlit images	Data obtained from Li's database	Backlit regions are identified and enhance the degraded image	Processing takes a long time, and the enhanced image quality depends highly on the accuracy of segmentation
McCann and Vonikakis (2018)	Estimating the retinal contrast from the image	High-dynamic range dataset	It illuminates both low-light and brighter areas of the image	Underexposed regions are slightly enhanced

Fig 1. Advantages and disadvantages of low light enhancement techniques

1.5 Architecture diagram

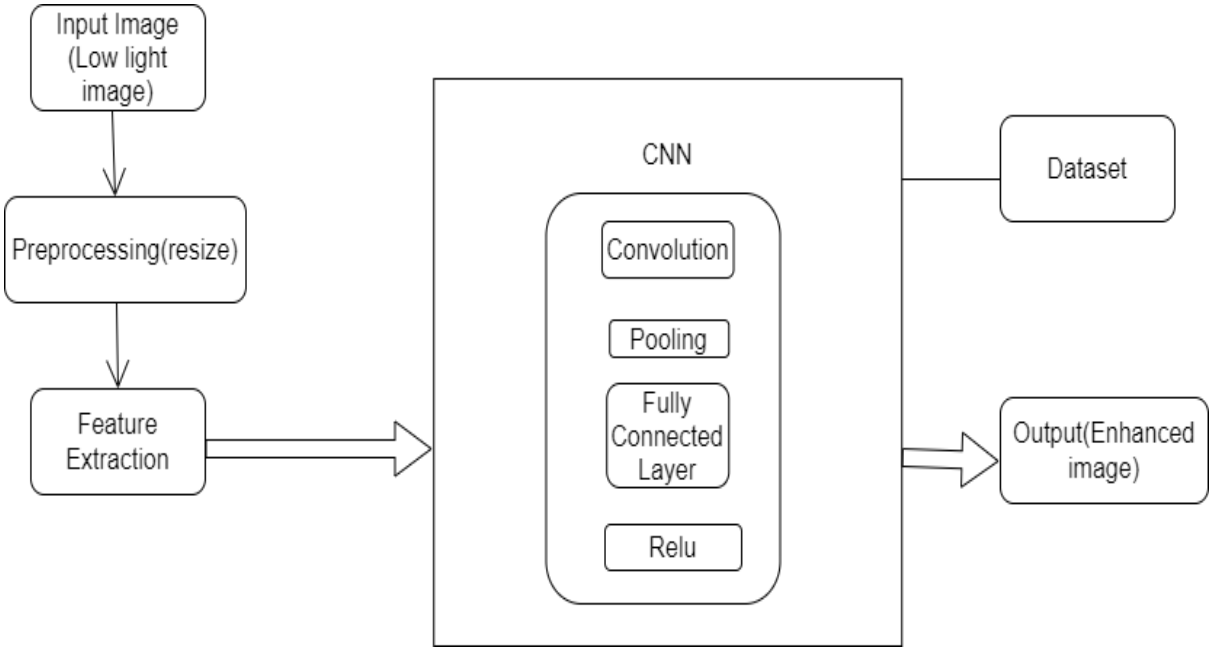

Fig 2. Architecture of CNN

Fig 3. Architecture of Lighten-Net model

1.6 Image Enhancement

1.6.1 Retinex Theory

The Retinex theory motivated by Land is based on the physical imaging model, in which an image I(x,y) is regarded as the product I(x,y) = R(x, y) · L(x,y) where R(x, y) is the reflectance and L(x, y) is the illumination at each pixel (x,y). Here, the nature of L(x, y) is determined by the illumination source, whereas R(x,y) is determined by the characteristics of the imaged objects. Therefore, the illumination normalization can be achieved by estimating the illumination L and then dividing the image I by it. However, it is impossible to estimate L from I, unless something else is known about either L or R. Hence, various assumptions and simplifications about L, or R, or both are proposed to solve this problem. A common assumption is that edges in the scene are edges in the reflectance, while illumination spatially changes slowly in the scene. Thus, in most Retinex methods, the reflectance R is estimated as the ratio of the image I and its smooth version which serves as the estimate of the illumination L, and many smoothing filters to estimate the illumination have been proposed. Retinx theory mainly compensate for the impact of images affected by illumination. Based on Retinx image formation model: S(x,y) = R(x, y) · L(x,y) (1). An image is pixel-by-pixel product of the ambient illumination and the scene reflectance. As the ambient illumination is independent of object itself, only the scene reflectance reflects the inhesion characteristic of object itself. Illumination is a kind of low-frequency image information which is slow changing, and reflectance contains the most high-frequency detailed image information. The Retinx theory

deal with the problem of separating the two quantities: first estimating the illumination and then obtaining the reflectance by division. From the mathematical point of viewbased on logarithmic domain, complex multiplication can be converted to a simple addition operation. So the first step taken by most Retinx Algorithms is the conversion of the given image into Logarithmic domain. As shown in formula 2: log S = logR + log L (2). Therefore, as shown in formula 3, the logarithm of the reflectance can be obtained by the logarithm of the image subtract the logarithm of the illumination. logR = log S - log L (3) Then the reflectance can be obtained by taken its index form, as shown in formula 4. The reflectance is inherent properties of object itself. R = exp(log S - log L) (4) As the illumination compared with the reflectance is low frequency component, so the Retinx Algorithm uses the low-pass filter to estimate the illumination .However, as Gaussian filter used in the filtering process will inevitably lose some high-frequency components, image will lose some of the details and edges, resulting in image distortion.

1.6.2 Image Enhancement Techniques

Image enhancement is the process of improving the quality and information content of raw data prior to processing. For picture improvement, a variety of software is employed, such as filters, image editors, and other tools for changing various aspects of a full image or parts of an image. A few basic sorts of image enhancement technologies merely alter an image's contrast or brightness or edit its grayscale or red-green-blue colour patterns. Basic filters can also be used to convert a colour image to black and white or sepia tone, as well as to add visual effects. Image enhancement software that is more advanced can apply adjustments to areas of an image. Professional packages, such as those offered by Adobe, enable designers to perform more specialized or professional image enhancements, or to pursue results for graphic design projects in which the original image is stylized or otherwise enriched. Other complex resources for restoring or clarifying images that may be in poor condition due to sub-optimal image capture conditions, ageing, or other causes are included in more advanced types of image enhancement tools, such as Wiener filters for actual de-blurring of images and other complex resources for restoring or clarifying images that may be in poor condition due to sub-optimal image capture conditions, ageing, or other causes.

1.6.3 Image Enhancement and Convolutional Neural Network

CNNs are used for image categorization and recognition because of their high accuracy. The

Convolutional Neural Network is a hierarchical model that builds a network in the shape of a funnel, after that produces a fully connected layer. Which is of all interconnected neurons which are connected to one another, and the output is obtained. CNN outperform traditional neural networks using visual, speech, or audio signal inputs. There are three different sorts of layers:

- Convolutional layer
- Pooling layer
- Fully connected (FC) layer.

The convolutional layer is the initial layer of a convolutional network. After convolutional layers, more convolutional layers or pooling layers can be added, but the fully connected layer is the last layer. With each layer, the CNN grows more sophisticated, identifying bigger portions of the image. The first layers focus on the most fundamental aspects, such as colours and borders. As the visual data passes through the CNN layers, it starts to recognize larger parts or features of the item, eventually identifying the target object.

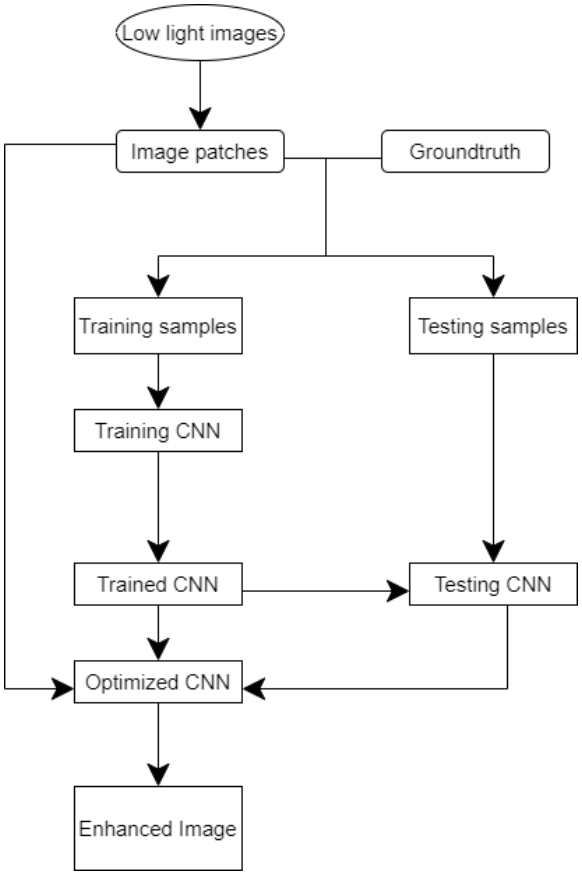

Fig 4. Flow chart of CNN

1.6.4 Image denoising

In image processing, denoising takes a major part. Vision is obstructed by Noise and diminishes its quality. As a result, denoising is the initial stage in image enhancement. The BM3D algorithm was utilized before neural networks. A Convolutional-Deconvolutional model with 30 layers was presented as part of a deep CNN architecture. Symmetric Skip Connections (SSC) were also added in the architecture between the Conv-DeConv layers. The findings of this method were superior to those of the BM3D algorithm. Direct Symmetric Connection (DSC) was the concept presented including 5 Conv and DeConv layer and 10 Convolutional Neural Network. Each layer of Conv is linked to its equivalent layer of Deconv in DSC, resulting in four direct connections. The pixelwise Mean Squared Error, which is the most generally used loss minimizer, is used to denoise the image.

1.6.4.1 CNN based Image Denoising

In general, the solving methods of the objective function is build upon the image degradation process and the image priors, and it can be divided into two main categories: model-based optimization methods and convolutional neural network (CNN)-based methods. The variational denoising methods discussed above belong to model-based optimization schemes, which find optimal solutions to reconstruct the denoised image. However, such methods usually involve time-consuming iterative inference. On the contrary, the CNN-based denoising methods attempt to learn a mapping function by optimizing a loss function on a training set that contains degraded-clean image pairs.Recently, CNN-based methods have been developed rapidly and have performed well in many low-level computer vision tasks. The use of a CNN for image denoising can be tracked back to, where a five-layer network was developed. In recent years, many CNN-based denoising methods have been proposed. Compared to that of, the performance of these methods has been greatly improved. Furthermore, CNN-based denoising methods can be divided into two categories: multi-layer perception (MLP) models and deep learning methods.

1) MLP models

MLP-based image denoising models include auto-encoders proposed by Vincent and Xi. Chen proposed a feed-forward deep network called the trainable non-linear reaction diffusion (TNRD) model, which achieved a better denoising effect. This category of methods has several

advantages. First, these methods work efficiently owing to fewer ratiocination steps. Moreover, because optimization algorithms have the ability to derive the discriminative architecture, these methods have better interpretability. Nevertheless, interpretability can increase the cost of performance; for example, the MAP model restricts the learned priors and inference procedure

2) **Deep learning-based denoising methods**

The state-of-the-art deep learning denoising methods are typically based on CNNs. The general model for deep learning-based denoising methods is formulated as

minΘloss(x^,x),s.t.x^=F(y,σ;Θ)minΘloss(x^,x),s.t.x^=F(y,σ;Θ)

where $F(\cdot)$ denotes a CNN with parameter set Θ, and *loss*(·) denotes the loss function. *loss*(·) is used to estimate the proximity between the denoised image x^x^ and the ground-truth *x*. Owing to their outstanding denoising ability, considerable attention has been focused on deep learning-based denoising methods.

Zhang introduced residual learning and batch standardization into image denoising for the first time; they also proposed feed-forward denoising CNNs (DnCNNs). The aim of the DnCNN model is to learn a function x^=F(y;Θσ)x^=F(y;Θσ) that maps between *y* and x^x^. The parameters Θ_σ are trained for noisy images under a fixed variance σ. There are two main characteristics of DnCNNs: the model applies a residual learning formulation to learn a mapping function, and it combines it with batch normalization to accelerate the training procedure while improving the denoising results. Specifically, it turns out that residual learning and batch normalization can benefit each other, and their integration is effective in speeding up the training and boosting denoising performance. Although a trained DnCNN can also handle compression and interpolation errors, the trained model under σ is not suitable for other noise variances.

When the noise level σ is unknown, the denoising method should enable the user to adaptively make a trade-off between noise suppression and texture protection. The fast and flexible denoising convolutional neural network (FFDNet) was introduced to satisfy these desirable characteristics. In particular, FFDNet can be modeled as x^=F(y,M;Θ)x^=F(y,M;Θ) (M denotes a noise level map), which is a main contribution. For FFDNet, M indicates an input while the parameter set Θ are fixed for noise level. Another major contribution is that FFDNet

acts on down-sampled sub-images, which speeds up the training and testing and also expands the receptive field. Thus, FFDNet is quite flexible to different noises.

Although this method is effective and has a short running time, the time complexity of the learning process is very high. The development of CNN-based denoising methods has enhanced the learning of high-level features by using a hierarchical network.

1.6.5 Different Convolutional Networks

In deep learning, a convolutional neural network (CNN or ConvNet) is a class of deep neural networks, that are typically used to recognize patterns present in images but they are also used for spatial data analysis, computer vision, natural language processing, signal processing, and various other purposes The architecture of a Convolutional Network resembles the connectivity pattern of neurons in the Human Brain and was inspired by the organization of the Visual Cortex. This specific type of Artificial Neural Network gets its name from one of the most important operations in the network: convolution.

What Is a Convolution?

Convolution is an orderly procedure where two sources of information are intertwined; it's an operation that changes a function into something else. Convolutions have been used for a long time typically in image processing to blur and sharpen images, but also to perform other operations. (e.g. enhance edges and emboss) CNNs enforce a local connectivity pattern between neurons of adjacent layers. CNNs make use of filters (also known as kernels), to detect what features, such as edges, are present throughout an image. There are four main operations in a CNN:

- Convolution
- Non Linearity (ReLU)
- Pooling and Sub Sampling
- Classification (Fully Connected Layer)

The first layer of a convolutional neural network is always a **Convolutional Layer.** Convolutional layers apply a convolution operation to the input, passing the result to the next layer. A convolution converts all the pixels in its receptive field into a single value. For example, if you would apply a convolution to an image, you will be decreasing the image

size as well as bringing all the information in the field together into a single pixel. The final output of the convolutional layer is a vector. Based on the type of problem we need to solve and on the kind of features we are looking to learn, we can use different kinds of convolutions.

1) The 2D Convolution Layer

The most common type of convolution that is used is the 2D convolution layer and is usually abbreviated as conv2D. A filter or a kernel in a conv2D layer "slides" over the 2D input data, performing an elementwise multiplication. As a result, it will be summing up the results into a single output pixel. The kernel will perform the same operation for every location it slides over, transforming a 2D matrix of features into a different 2D matrix of features.

2) The Dilated or Atrous Convolution

This operation expands window size without increasing the number of weights by inserting zero-values into convolution kernels. Dilated or Atrous Convolutions can be used in real time applications and in applications where the processing power is less as the RAM requirements are less intensive.

3) Separable Convolutions

There are two main types of separable convolutions: spatial separable convolutions, and depthwise separable convolutions. The spatial separable convolution deals primarily with the spatial dimensions of an image and kernel: the width and the height. Compared to spatial separable convolutions, depth wise separable convolutions work with kernels that cannot be "factored" into two smaller kernels. As a result, it is more frequently used.

4) Transposed Convolutions

These types of convolutions are also known as deconvolutions or fractionally strided convolutions. A transposed convolutional layer carries out a regular convolution but reverts its spatial transformation.

CHAPTER – 2

2. SOFTWARE REQUIREMENT SPECIFICATION

2.1 What is SRS?

Software Requirement Specification (SRS) is the starting point of the software developing activity. As system grew more complex it became evident that the goal of the entire system cannot be easily comprehended. Hence the need for the requirement phase arose. The software project is initiated by the client needs. The SRS is the means of translating the ideas of the minds of clients (the input) into a formal document (the output of the requirement phase.)
The SRS phase consists of two basic activities:

Problem/Requirement Analysis: The process is order and more nebulous of the two, deals with understand the problem, the goal and constraints.

Requirement Specification: Here, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools, and checking the specifications are addressed during this activity. The Requirement phase terminates with the production of the validate SRS document. Producing the SRS document is the basic goal of this phase.

2.2 Role of SRS

The purpose of the Software Requirement Specification is to reduce the communication gap between the clients and the developers. Software Requirement Specification is considered as the medium through which the client and user needs are accurately specified. It forms the basis of software development. A good SRS should satisfy all the parties involved in the system.

2.3 Requirements Specification Documentation

A Software Requirements Specification (SRS) is a document that describes the nature of a project, software or application. In simple words, SRS document is a manual of a project provided it is prepared before you kick-start a project/application. This document is also known by the names SRS report, software document. A software document is primarily prepared for a project, software or any kind of application. There are a set of guidelines to be followed while preparing the software requirement specification document. This includes the purpose, functional and non-functional requirements, software and hardware requirements of the

project. In addition to this, it also contains the information about environmental conditions required, safety and security requirements, software quality attributes of the project etc.

The purpose of SRS (Software Requirement Specification) document is to describe the external behaviour of the application developed or software. It defines the operations, performance and interfaces and quality assurance requirement of the application or software. The complete software requirements for the system are captured by the SRS.

This document introduces the requirement specification document for efficient image enhancement using Convolutional Neural Network. It also specifies the required software, functional or otherwise that have been incorporated.

In the process of development of a working model of image enhancement in low light which makes use of convolutional neural networks.

For the same, we incorporate, Python, CNN so that the model can cater both linear and non-linear associations among output and input data and other datasets are used in order to further facilitate the process of enhancing the quality of the image which is in low light conditions.

2.4 Functional Requirements

For documenting the functional requirements, the set of functionalities supported by the system are to be specified. A function can be specified by identifying the state at which data is to be input to the system, its input data domain, the output domain, and the type of processing to be carried on the input data to obtain the output data. Functional requirements define specific behavior or function of the application. Following are the functional requirements:

FR1) The dataset uploaded must remain intact and consist of clear specifications.

FR2) The image samples collected must be carefully stored for indefinite duration as its need` arise at any point.

FR3) The database is supposed to be efficient to handle large amounts of data and the required information.

FR4) The images obtained from camera should be less shaky.

FR5) The database is supposed to be efficient to handle large amounts of data and information.

FR6) The mapping done on the images should be stated clearly and must match with the dataset while comparison.

FR7) After mapping and giving an output of the enhanced images, the details of the image should match correctly with the information of the image that is provided.

2.5 Non-Functional Requirements

NFR1) The model should be capable of functioning with various types of datasets and images provided.

NFR2) Should be able to identify the pointers, special characteristics etc. from the images provided.

NFR3) The model should be able to perform efficiently even after new data is provided.

NFR4) The model must provide the results with very little delay time.

NFR5) The most important requirement is that the model must provide results with high accuracy irrespective of the data provided.

NFR6) The model must be developed in such a way that it has a capacity to interpret the minute details obtained from images or data correctly.

2.6 Software Requirements

2.6.1 Python 3.10.8 (latest)

The primary step for implementing this model is to extract the features from the image provided from the camera .

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured, object-oriented and functional programming.

Python can serve as a scripting language for web applications,e.g.,via mod_wsgi for the Apache webserver.

Web frameworks like Django, Pylons, Pyramid, TurboGears, web2py, Tornado, Flask, Bottle,

and Zope support developers in the design and maintenance of complex applications. Pyjs and IronPython can be used to develop the client-side of Ajax-based applications. SQL Alchemy can be used as a data mapper to a relational database. Twisted is a framework to program communications between computers, and is used (for example) by Dropbox.

Libraries such as NumPy, SciPy, and Matplotlib allow the effective use of Python in scientific computing, with specialized libraries such as Biopython and Astropy providing domain-specific functionality. SageMath is a computer algebra system with a interface programmable in Python:its library covers many aspects of mathematics, including algebra, combinatorics, numerical mathematics, number theory. OpenCV has Python bindings with a rich set of features for computer vision and image processing. Python is commonly used in artificial intelligence projects and machine learning projects with the help of libraries like TensorFlow, Keras, Pytorch, and Scikit-learn. As a scripting language with a modular architecture, simple syntax, and rich text processing tools, Python is often used for natural language processing.

History

Guido van Rossum, often referred to as, Benevolent Dictator for Life (BDFL), is the principal author of the Python programming language. Guido van Rossum, has recently, however, stepped down as leader on July 12, 2018. Python itself is actually named after Monty Python's Flying Circus. The first version of the code published by Van Rossum was labelled version 0.9.0. Even in this early version of code there were classes with inheritance, exception handling, functions, and many other major datatypes. Python's userbase form comp.lang.python in 1994 was formed, leading to more growth and use of the language. As you can see, Python had a strong core foundation right out of the gate. It is no wonder the potential for this flexible language is being pushed everyday.

Features

Rather than building all of its functionality into its core, Python was designed to be highly extensible via modules. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach.

Python strives for a simpler, less-cluttered syntax and grammar while giving developers a

choice in their coding methodology. In contrast to Perl's "there is more than one way to dmotto, Python embraces a "there should be one—and preferably only one—obvious way to do it" philosophy. Alex Martelli, a Fellow at the Python Software Foundation and Python book author, wrote: "To describe something as 'clever' is not considered a compliment in the Python culture."

Python's developers strive to avoid premature optimization and reject patches to non-critical parts of the CPython reference implementation that would offer marginal increases in speed at the cost of clarity. When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C; or use PyPy, a just-in-time compiler. Cython is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter.

Python's developers aim for it to be fun to use. This is reflected in its name—a tribute to the British comedy group Monty Python—and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (a reference to a Monty Python sketch) instead of the standard foo and bar.

The programming language's name 'Python' came from the BBC Comedy series Monty Python's Flying Circus. Guido van Rossum thought he needed a name that was short, unique and slightly mysterious, And so, he decided to name the programming language 'Python'.

A common neologism in the Python community is pythonic, which has a wide range of meanings related to program style. "Pythonic" code may use Python idioms well, be natural or show fluency in the language, or conform with Python's minimalist philosophy and emphasis on readability. Code that is difficult to understand or reads like a rough transcription from another programming language is called unpythonic.

Python users and admirers, especially those considered knowledgeable or experienced, are often referred to as Pythonistas.

Development

Python's development is conducted largely through the Python Enhancement Proposal (PEP) process, the primary mechanism for proposing major new features, collecting community input on issues, and documenting Python design decisions. Python coding style is covered in PEP 8. Outstanding PEPs are reviewed and commented on by the Python community and the steering council.

Enhancement of the language corresponds with the development of the CPython reference implementation. The mailing list python-dev is the primary forum for the language's development. Specific issues are were originally discussed in the Roundup bug tracker hosted at by the foundation. In 2022, all issues and discussions were migrated to GitHub. Development originally took place on a self-hosted source-code repository running Mercurial, until Python moved to GitHub in January 2017.

CPython's public releases come in three types, distinguished by which part of the version number is incremented:

- Backward-incompatible versions, where code is expected to break and needs to be manually ported. The first part of the version number is incremented. These releases happen infrequently—version 3.0 was released 8 years after 2.0. According to Guido van Rossum, a version 4.0 is very unlikely to ever happen.
- Major or "feature" releases are largely compatible with the previous version but introduce new features. The second part of the version number is incremented. Starting with Python 3.9, these releases are expected to happen annually. Each major version is supported by bug fixes for several years after its release
- Bugfix releases which introduce no new features, occur about every 3 months and are made when a sufficient number of bugs have been fixed upstream since the last release. Security vulnerabilities are also patched in these releases. The third and final part of the version number is incremented

Many alpha, beta, and release-candidates are also released as previews and for testing before final releases. Although there is a rough schedule for each release, they are often delayed if the code is not ready. Python's development team monitors the state of the code by running the large unit test suite during development.

The major academic conference on Python is PyCon. There are also special Python mentoring programmes, such as Pyladies.

Python 3.10 deprecated (to be removed in Python 3.12; meaning Python extensions need to be modified by then), and added pattern matching to the language.

Since 2003, Python has consistently ranked in the top ten most popular programming languages in the TIOBE Programming Community Index where, as of October 2021, it is the most popular language (ahead of Java, and C). It was selected Programming Language of the Year (for "the highest rise in ratings in a year") in 2007, 2010, 2018, and 2020 (the only language to do so

four times).

An empirical study found that scripting languages, such as Python, are more productive than conventional languages, such as C and Java, for programming problems involving string manipulation and search in a dictionary, and determined that memory consumption was often “better than Java and not much worse than C or C++”. Large organization like Wikipedia, Google, CERN, NASA, Facebook, Amazon, Instagram, Spotify and some other small entities make use of Python. The social news networking site Reddit has been written mostly in Python.

2.6.2 Pandas 1.5.1 (latest)

Pandas provides rich set of functions to process various types of data. Further, working with Panda is fast, easy and more expressive than other tools. Pandas provides fast data processing as Numpy along with flexible data manipulation techniques as spreadsheets and relational databases. Lastly, pandas integrate well with matplotlib library, which makes it very handy tool for analysing the data.

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term “panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals. Its name is a play on the phrase "Python data analysis" itself. Wes McKinney started building what would become pandas at AQR Capital while he was a researcher there.

2.6.3 NumPy 1.23.4 (latest)

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, high-level mathematical functions to operate on arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors. NumPy is a NumFOCUS fiscally sponsored project. NumPy targets the CPython reference implementation of Python, which is a non- optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents due to the absence of compiler optimization. NumPy

addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays; using these requires rewriting some code, mostly inner loops, using NumPy.

Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available; SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that provides MATLAB-like plotting functionality. Internally, both MATLAB and NumPy rely on BLAS and LAPACK for efficient linear algebra computations.

Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array as universal data structure in OpenCV for images, extracted feature points, filter kernels and many more vastly simplifies the programming workflow and debugging.

2.6.4 Keras 2.10. 0 (latest)

Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library.

Up until version 2.3, Keras supported multiple backends, including TensorFlow, Microsoft Cognitive Toolkit, Theano and PlaidML. As of version 2.4, only TensorFlow is supported. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a Google engineer. Chollet is also the author of the Xception deep neural network model.

Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural

network code. The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel. In addition to standard neural networks, Keras has support for convolutional and recurrent neural networks. It supports other common utility layers like dropout, batch normalization, and pooling.

Keras allows users to productize deep models on smartphones (iOS and Android), on the web, or on the Java Virtual Machine. It also allows use of distributed training of deep-learning models on clusters of Graphics processing units (GPU) and tensor processing units (TPU).

2.6.5 TensorFlow

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML-powered applications.

TensorFlow was originally developed by researchers and engineers working on the Google Brain team within Google's Machine Intelligence Research organization to conduct machine learning and deep neural networks research. The system is general enough to be applicable in a wide variety of other domains, as well.

TensorFlow provides stable Python and C++ APIs, as well as non-guaranteed backward compatible API for other languages.

2.6.6 Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.

Matplotlib was originally written by John D. Hunter. Since then it has had an active development community and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012 and was further joined by Thomas Caswell Matplotlib is a NumFOCUS fiscally sponsored project.

Matplotlib 2.0.x supports Python versions 2.7 through 3.10. Python 3 support started with

Matplotlib 1.2. Matplotlib 1.4 is the last version to support Python 2.6. Matplotlib has pledged not to support Python 2 past 2020 by signing the Python 3 Statement.

2.6.7 OpenCV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. OpenCV was originally developed by Intel back in 1999 and is now maintained by Willow Garage and ItSeez. OpenCV (Open Source Computer Vision) is a library of programming functions for realtime computer vision, itt uses a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java (Android) interfaces and supports Windows, Linux, Android, iOS and Mac OS. It has more than 2500 optimized algorithms. Adopted all around the world, OpenCV has more than 9 million downloads growing by nearly 200K/month. Usage ranges from interactive art, to mines inspection, stitching maps on the web on through advanced robotics.

2.7 Hardware Requirements

OS: Windows 7 OS or above Recommended: Windows 11 (latest)

CPU: Intel core i5 or above, AMD Ryzen 5 or Above

Disk Storage: 8GB or above of free disk space

RAM: 4GB or above

GPU: Intel Integrated Graphics or higher

For Execution: Spyder using Anaconda Framework in Python or Google Collab

CHAPTER – 3

3. LITERATURE SURVEY

This research explores the methodologies that have been employed to help solve problems related to image enhancement. To obtain a clear image it's contrast, sharpness and color co-ordination should be handled simultaneously and effectively which can be a difficult task for humans when done manually. Typically, the images can be enhanced using the techniques such as denoising, deblurring, and other enhancement techniques their efficiency is limited in extreme conditions, such as video-rate imaging at night. Therefore, we proposed a low light image enhancement using convolutional neural network system takes dark images as input and produces bright images as an output without disturbing the content of the image.

1) Low-light image enhancement using CNN and bright channel prior - 2017 IEEE International Conference on Image Processing (ICIP)

Authors : Li Tao; Chuang Zhu; Jiawen Song; Tao Lu; Huizhu Jia; Xiaodong Xie

The researchers formulated a low-light model based on the atmosphere scattering model, in which the light conditions at night are taken into consideration. They proposed a simple but effective image prior, bright channel prior, to estimate parameters in our low-light model. Besides, a CNN based model is designed to denoise low-light images to improve image quality before the model-based enhancement. Inspired by super-resolution convolutional networks and atmosphere scattering model, the researchers proposed a joint approach to enhance low-light images. The first stage is using an image in-image-out network to remove image noise. In the second stage, they applied the low-light model on the noise-free images.

2) LightenNet: A Convolutional Neural Network for weakly illuminated image enhancement

Authors: Chongyi Li, Jichang Guoa, Fatih Porikli, Yanwei Pang

The researchers proposed a trainable CNN for weak illumination image enhancement. Different from several existing CNN-based image enhancement methods which directly estimate the enhanced or restored image, the LightenNet learns to predict the mapping relations between weakly illuminated image and the corresponding illumination map. Thus, LightenNet is easy to be trained. It just takes one hour to optimize our LightenNet. Based on Retinex model, the researchers proposed a new method for weakly illuminated image synthesis, which can be used as a guide for subsequent network training and full-reference image quality assessment.

Compared to existing weak illumination image enhancement methods, the proposed CNN-based method achieves the state-of-the-art performance on both synthetic and real weakly illuminated images. The proposed LightenNet is inspired by the success of deep learning in low-level vision tasks. The purpose of LightenNet is to learn a mapping, which takes a weakly illuminated image as input and outputs its illumination map that is subsequently used to obtain the enhanced image based on Retinex model. The architecture of LightenNet is shown in Fig. 1. In the LightenNet architecture, input is the weakly illuminated image and the output is the corresponding illumination map. Similar with Dong et al. [3], LightenNet contains four convolutional layers with specific tasks. Observing the feature maps, different convolutional layers have different effects on final illumination map. For example, the first two layers focus on the high light regions and the third layer focuses on low light regions while the last layer is to reconstruct the illumination map.

3) LLCNN: A Convolutional Neural Network for Low-light Image Enhancement - 2017 IEEE Visual Communications and Image Processing (VCIP)

Authors: Li Tao, Chuang Zhu, Guoqing Xiang, Yuan Li, Huizhu Jia, Xiaodong Xie

The researchers proposed a network which is referred to as Low-light Convolutional Neural Network (LLCNN). It learns to filter low-light images with different kernels and then combine multiscale feature maps together to generate enhanced images, which seem to be captured under normal light conditions, and reserve original features and textures. Besides, SSIM loss is integrated into the LLCNN to reconstruct more accurate image textures. The researchers then compared the results against other methods and the results demonstrated that our method achieves the best performance among common low-light image enhancement methods.

4) A Convolutional Neural Network Based Method for Low-illumination Image `Enhancement

Authors : Huang Huang, Haijun Tao, Haifeng Wang

The researchers proposed a new model called as Unet. Unet is a special U-shaped CNN model proposed for biomedical image segmentation. It consists of a contracting path and a symmetric expanding path. Unet produces low scales images by pooling and reconstructs the images scales by deconvolution. At the same time of shrinking and enlarging image scale, convolutions of model are used for feature extraction and image processing. The Unet processing is similar

to the well-known image pyramid method. And in the field of digital image processing, a variety of pyramid methods are used for image data compression, image enhancement and so on. Thus, Unet is more suitable than other classical CNN model for image processing tasks in deep learning. In this paper, our model is based on Unet.

The researchers reduced the depths of Unet and have some convolution layers embedded in each layer of the skip connection inside the U-shaped structure. This has a transitional effect on the shallow and deep features of each skip connection while enhancing the performance of the network. Also, two additional convolution layers with kernels of size 1×1 are added as deep-supervised introduction. There are some benefits to this behavior. At first, it prevents the gradient vanishing during the training. Second, it makes the model more flexible..To sum up, the proposed model is a flexible model which can adjust the structure according to the training performance. At the same time, it still has the character of encoding and decoding of Unet, so the speed of processing is also very fast.

CHAPTER – 4

4. SYSTEM DESIGN

4.1 Introduction to UML

The Unified Modelling Language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic, semantic and pragmatic rules. The creation of UML was originally motivated by the desire to standardize the disparate notational systems and approaches to software design. It was developed at Rational Software in 1994–1995, with further development led by them through 1996.

In 1997, UML was adopted as a standard by the Object Management Group (OMG), and has been managed by this organization ever since. In 2005, UML was also published by the International Organization for Standardization (ISO) as an approved ISO standard. Since then, the standard has been periodically revised to cover the latest revision of UML. In software engineering, most practitioners do not use UML, but instead produce informal hand drawn diagrams; these diagrams, however, often include elements from UML.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagrams, which is as follows:

1. User Model View

This view represents the system from the users' perspective. The analysis representation describes a usage scenario from the end-users' perspective.

2. Structural Model View

In this model, the data and functionality are arrived from inside the system. This model view models the static structures.

3.Behavioural Model View

It represents the dynamic of behavioural as parts of the system, depicting he interactions of collection between various structural elements described in the user model and structural model view.

4. Implementation Model View

In this view, the structural and behavioural as parts of the system are represented as they are to be built.

5.Environmental Model View

In this view, the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

4.2 UML Diagrams

4.2.1 USE CASE DIAGRAM

To model a system, the most important aspect is to capture the dynamic behaviour. To clarify a bit in details, dynamic behaviour means the behaviour of the system when it is running/operating. So only static behaviour is not sufficient to model a system rather dynamic behaviour is more important than static behaviour. In UML there are five diagrams available to model dynamic nature and use case diagram is one of them. Now as we have to discuss that the use case diagram is dynamic in nature there should be some internal or external factors for making the interaction. These internal and external agents are known as actors. So, use case diagrams are consisting of actors, use cases and their relationships. Use cases are a technique for capturing, modelling and specifying the requirements of a system. A use case corresponds to a set of behaviours that the system may perform in interaction with its actors, and which produces an observable result that contribute to its goals. Actors represent the role that human users or other systems have in the interaction. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system. So, to model the entire system numbers of use case diagrams are used. Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. So, when a system is analysed to gather its functionalities use cases are prepared and actors are identified. In brief, the purposes of use case diagrams can be as follows:

a. Used to gather requirements of a system.

b. Used to get an outside view of a system.

c. Identify external and internal factors influencing the system.

d. Show the interacting among the requirements are actors.

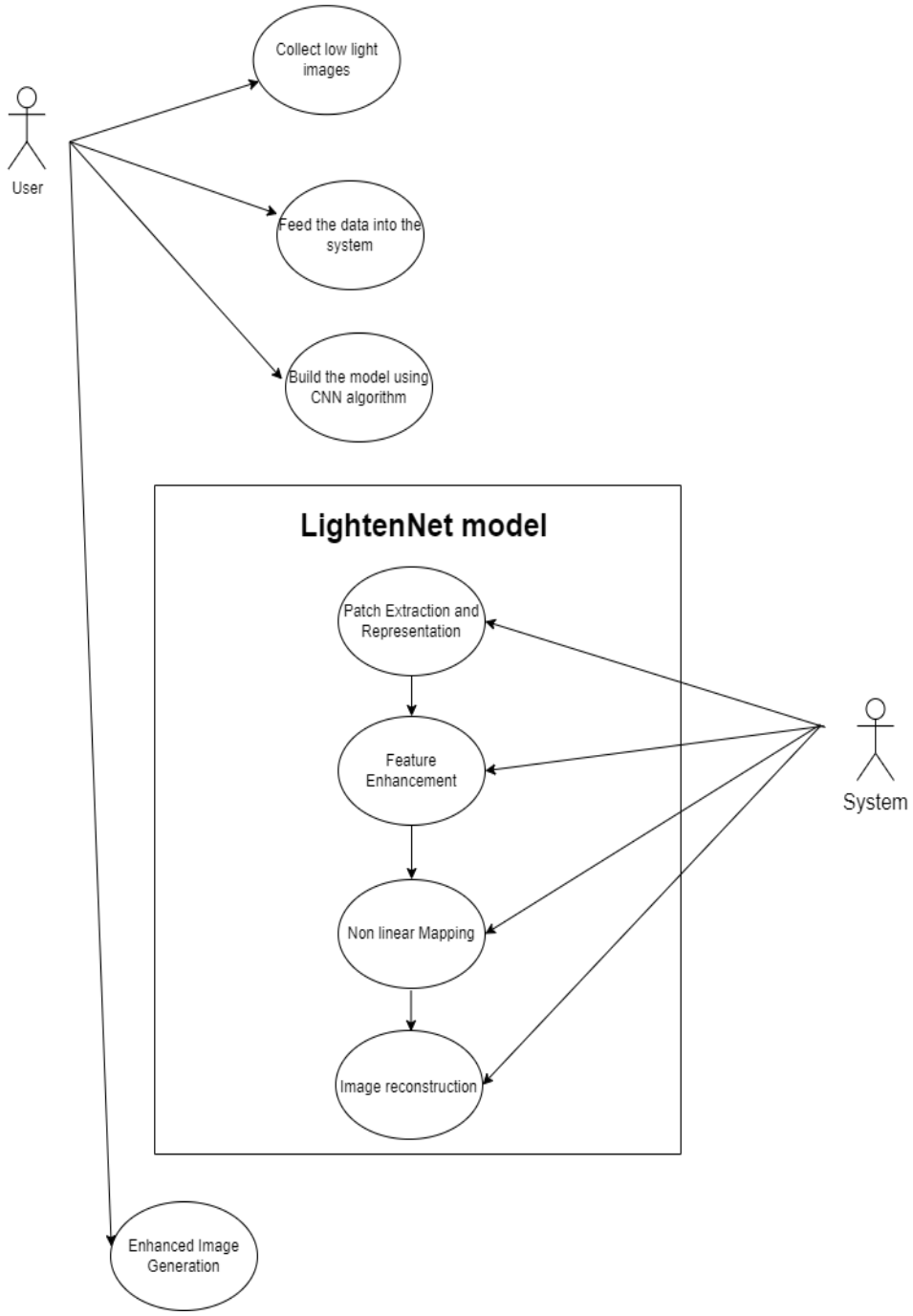

Fig 5. Use case diagram

4.2.2 SEQUENCE DIAGRAM

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time. They're also called event diagrams. For a particular scenario of a use case, the diagrams show the events that external actors generate, their order, and possible inter-system events. All systems are treated as a black box; the diagram places emphasis on events that cross the system boundary from actors to systems. A system sequence diagram should be done for the main success scenario of the use case, and frequent or complex alternative scenarios. A sequence diagram is a good way to visualize and validate various runtime scenarios. These can help to predict how a system will behave and to discover responsibilities a class may need to model, and in what order.

If the lifeline is that of an object, it demonstrates a role. Leaving the instance name blank can represent anonymous and unnamed instances.

Messages, written with horizontal arrows with the message name written above them, display interaction. The solid arrow heads represent synchronous calls, open arrow heads represent asynchronous messages, and dashed lines represent reply messages. If a caller sends a synchronous message, it must wait until the message is done, such as invoking a subroutine. If a caller sends an asynchronous message, it can continue processing and doesn't have to wait for a response. Asynchronous calls are present in multithreaded applications, event-driven applications and in message-oriented middleware. Activation boxes, or method-call boxes, are opaque rectangles drawn on top of lifelines to represent that processes are being performed in response to the message.

Objects calling methods on themselves use messages and add new activation boxes on top of any others to indicate a further level of processing. If an object is destroyed (removed from memory), an X is drawn on bottom of the lifeline, and the dashed line ceases to be drawn below it. It should be the result of a message, either from the object itself, or another.

A message sent from outside the diagram can be represented by a message originating from a

- Return values (if any) associated with previous messages
- Indication of any loops or iteration area

The aim of a sequence diagram is to define event sequences, which would have a desired outcome. The focus is more on the order in which messages occur than on the message per se. However, the majority of sequence diagrams will communicate what messages are sent and the order in which they tend to occur.

Basic Sequence Diagram Notations

Class Roles or Participants

Class roles describe the way an object will behave in context. Use the UML object symbol to illustrate class roles, but don't list object attributes.

Activation or Execution Occurrence

Activation boxes represent the time an object needs to complete a task. When an object is busy executing a process or waiting for a reply message, use a thin grey rectangle placed vertically on its lifeline.

Messages

Messages are arrows that represent communication between objects. Use half-arrowed lines to represent asynchronous messages. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks.

Lifelines

Lifelines are vertical dashed lines that indicate the object's presence over time.

Destroying Objects

Objects can be terminated early using an arrow labelled "<< destroy >>" that points to an X. This object is removed from memory. When that object's lifeline ends, you can place an X at the end of its lifeline to denote a destruction occurrence.

Loops

A repetition or loop within a sequence diagram is depicted as a rectangle. Place the condition for exiting the loop at the bottom left corner in square brackets [].

Guards

When modelling object interactions, there will be times when a condition must be met for a message to be sent to an object. Guards are conditions that need to be used throughout UML diagrams to control flow.

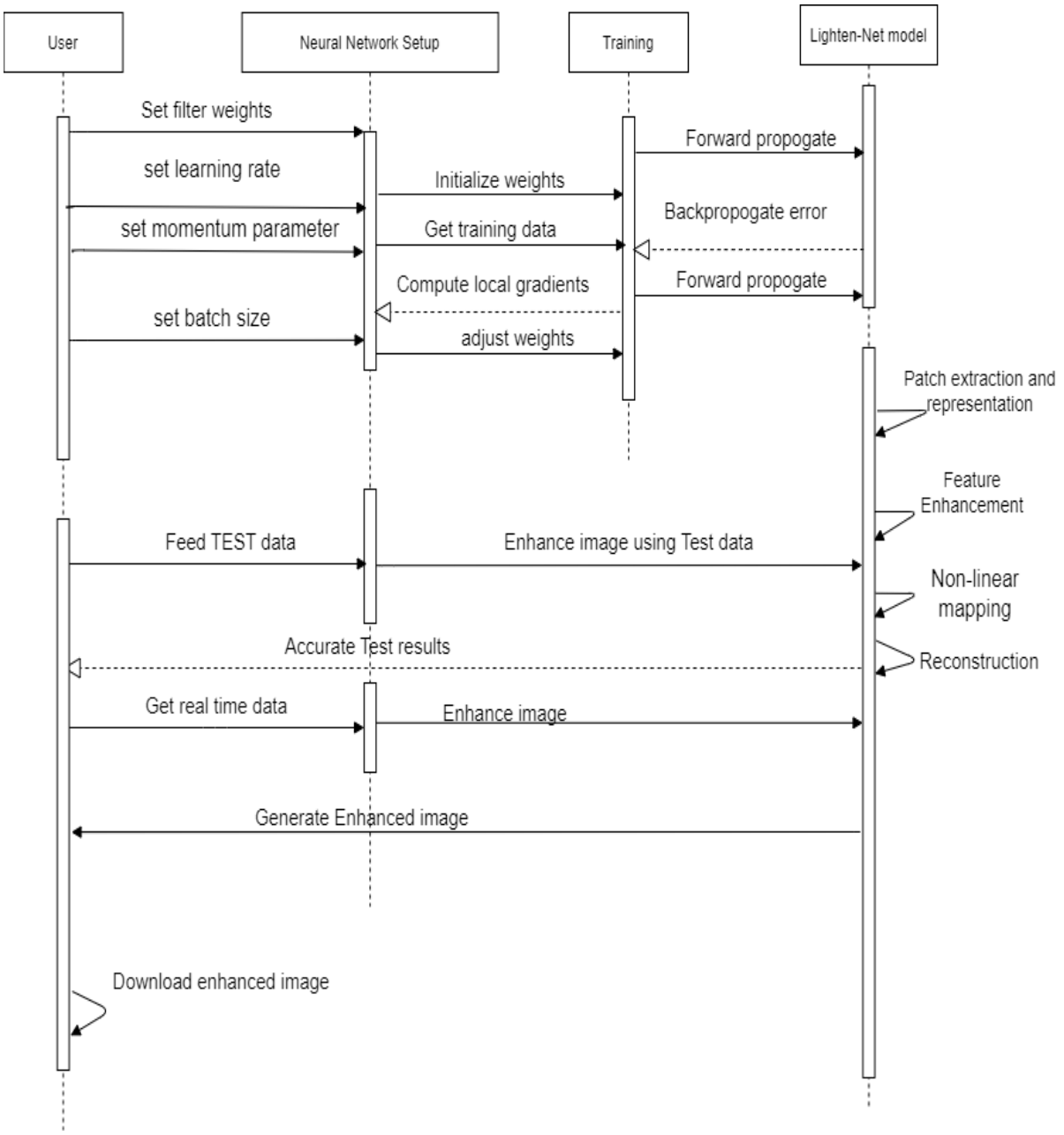

Fig 6. Sequence Diagram

4.2.3 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. Class

diagrams are the main building blocks of every object-oriented method. The class diagram can be used to show the classes, relationships, interface, association, and collaboration.

UML is standardized in class diagrams. Since classes are the building block of an application that is based on OOPs, so as the class diagram has appropriate structure to represent the classes, inheritance, relationships, and everything that OOPs have in its context. It describes various kinds of objects and the static relationship in between them. The main purpose to use class diagrams are:

1.This is the only UML which can appropriately depict various aspects of OOPs concept.

2. Proper design and analysis of application can be faster and efficient.

3. It is base for deployment and component diagram.

4. Each class is represented by a rectangle having a subdivision of three compartment name, attributes and operation.

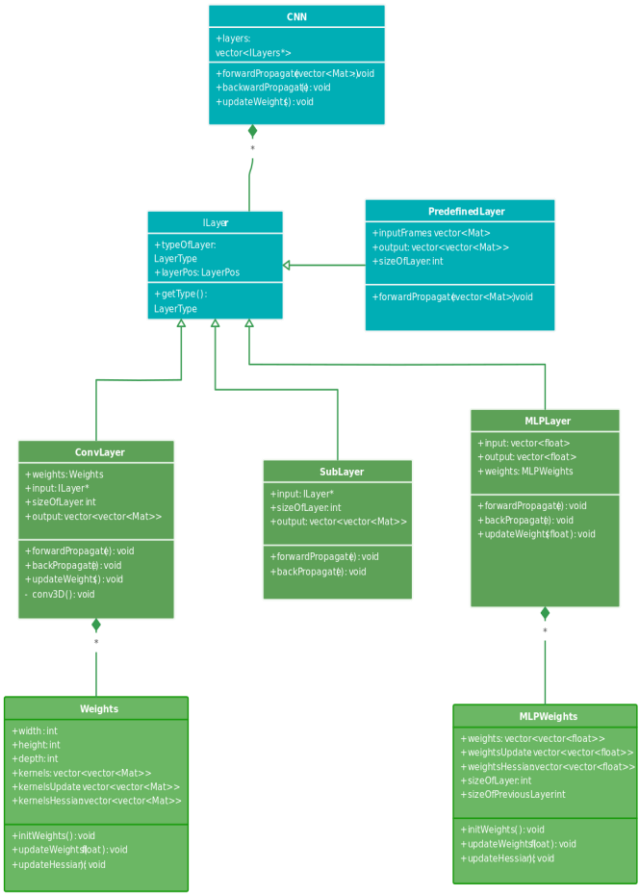

Fig 7. Class Diagram

4.2.4 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities. Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.

Activity diagrams are constructed from a limited number of shapes, connected with arrows. The most important shape types:

• ellipses represent actions;

• diamonds represent decisions;

• bars represent the start (split) or end (join) of concurrent activities;

• a black circle represents the start (initial node) of the workflow;

• an encircled black circle represents the end (final node).

Arrows run from the start towards the end and represent the order in which activities happen.

Activity diagrams can be regarded as a form of a structured flowchart combined with a traditional data flow diagram. Typical flowchart techniques lack constructs for expressing concurrency. However, the join and split symbols in activity diagrams only resolve this for simple cases; the meaning of the model is not clear when they are arbitrarily combined with decisions or loops.

While in UML 1.x, activity diagrams were a specialized form of state diagrams, in UML 2.x, the activity diagrams were reformalized to be based on Petri net-like semantics, increasing the scope of situations that can be modeled using activity diagrams. These changes cause many UML 1.x activity diagrams to be interpreted differently in UML 2.x.

UML activity diagrams in version 2.x can be used in various domains, e.g., in design of embedded systems. It is possible to verify such a specification using model checking technique.

The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.

The purpose of an activity diagram can be described as −

- Draw the activity flow of a system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched and concurrent flow of the system.

To draw an activity diagram, one must understand and explore the entire system. All the elements and entities that are going to be used inside the diagram must be known by the user. The central concept which is nothing but an activity must be clear to the user. After analyzing all activities, these activities should be explored to find various constraints that are applied to activities. If there is such a constraint, then it should be noted before developing an activity diagram. All the activities, conditions, and associations must be known. Once all the necessary things are gathered, then an abstract or a prototype is generated, which is later converted into the actual diagram.

Following rules must be followed while developing an activity diagram,

1. All activities in the system should be named.

2. Activity names should be meaningful.

3. Constraints must be identified.

4. Activity associations must be known.

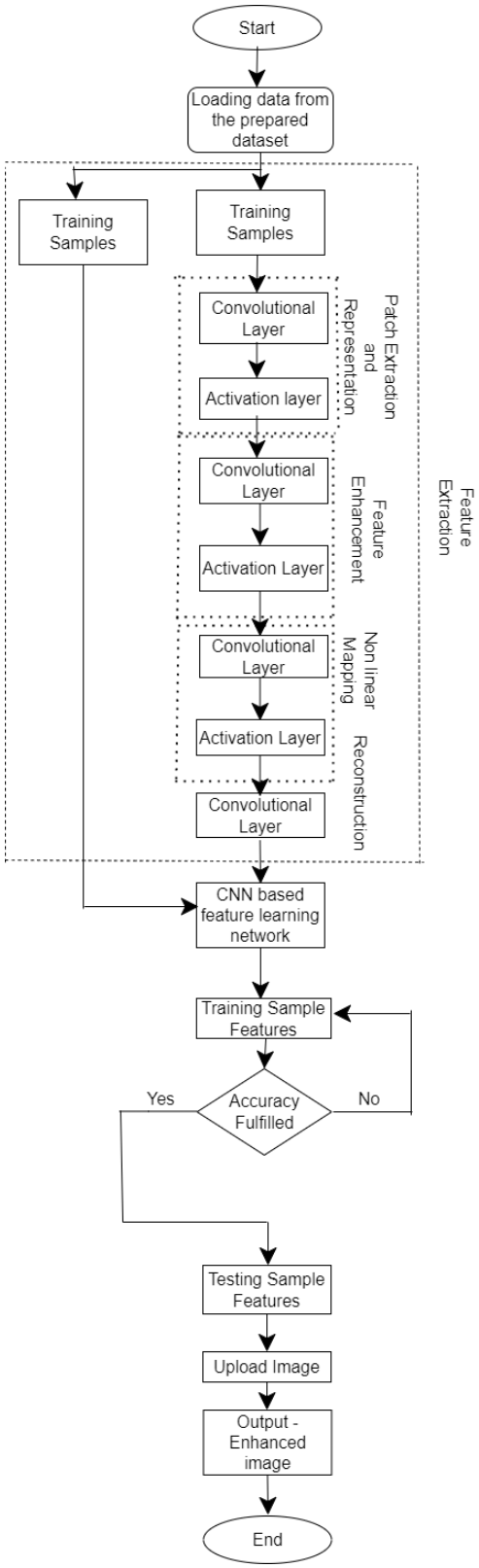

Fig 8. Activity Diagram

4.2.5 Deployment Diagram

A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them.

Deployment diagrams are typically used to visualize the physical hardware and software of a system. Using it you can understand how the system will be physically deployed on the hardware.

Deployment diagrams help model the hardware topology of a system compared to other UML diagram types which mostly outline the logical components of a system.

A node, represented as a cube, is a physical entity that executes one or more components, subsystems or executables. A node could be a hardware or software element.

Artifacts are concrete elements that are caused by a development process. Examples of artifacts are libraries, archives, configuration files, executable files etc.

This is represented by a solid line between two nodes. It shows the path of communication between nodes.

A device is a node that is used to represent a physical computational resource in a system. An example of a device is an application server.

Deployment specifications is a configuration file, such as a text file or an XML document. It describes how an artifact is deployed on a node.

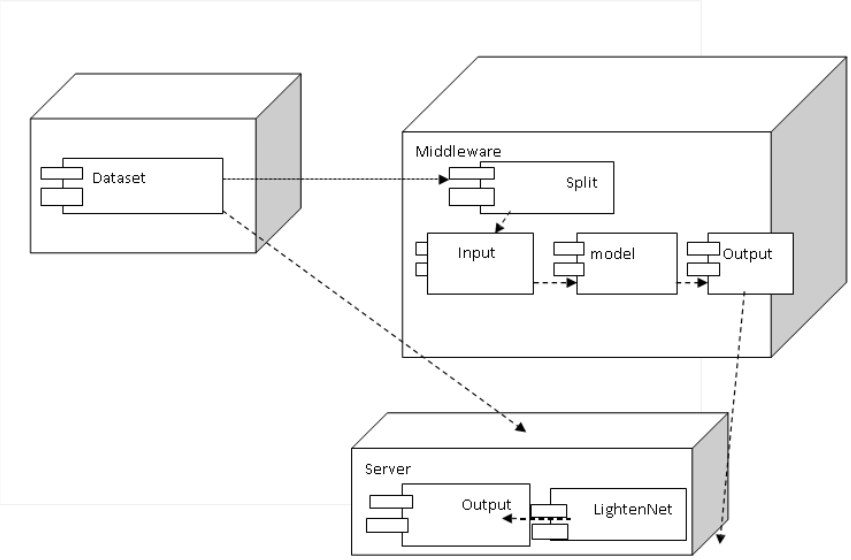

Fig 9. Deployment Diagram