

IMAGE REGENERATION FOR DAMAGE OLD REEL PICTURES

Submitted in partial fulfillment of the requirements
for the award of
Bachelor of Technology in Information Technology

By

MANU ROY (Reg.No - 39120067)

HARISH DHARAN P (Reg.No – 39120037)



**DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHISALAI,
CHENNAI – 600119**

MAY-2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade 'A' by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **MANU ROY(39120067)** and **HARISH DHARAN P (39120037)** who carried out the Project Phase-2 entitled **"IMAGE REGENERATION FOR DAMAGE OLD REEL PICTURES"** under my supervision from January 2023 to May 2023.

Internal Guide

Dr.SENDURU SRINIVASULU M.Tech.,Ph.D

Head of the Department

Dr.R SUBHASHINI M.E.,Ph.D

Submitted for Viva voce Examination held on 10.05.2023

Internal Examiner

External Examiner

DECLARATION

I **MANU ROY** (Reg.No- 39120067), hereby declare that the Project Phase-2 Report entitled "**IMAGE REGENERATION FOR DAMAGED OLD REEL PICTURES**" done by me under the guidance of **Dr.SENDURU SRINIVASULU M.Tech., Ph.D** is submitted in partial fulfillment of the requirements for the award Bachelor of Technology degree in Information Technology.

DATE: 10/5/2023

PLACE: Chennai


SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph. D** Dean of School of Computing and **Dr. R.Subhashini M.E., Ph.D** Head of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.Senduru Srinivasulu M.E.,Ph.D** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my phase-1 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project

ABSTRACT

Panoramic stitching technology provides an effective solution for expanding visual detection range of the autonomous underwater vehicle. However, absorption and scattering of light in the water seriously deteriorate the underwater imaging in terms of distance and quality, especially the scattering sharply decreases the underwater image contrast and results in serious blur. This reduces the number of matching feature points between the underwater images to be stitched, while fewer matched points generated make image registration and stitching difficult. To solve the problem, a joint framework is established, which firstly involves a convolutional neural network-like algorithm composed of a symmetric convolution and de convolution framework for underwater image enhancement. In this paper, we propose a simple but effective method to remove haze from a single input image.

Images captured in hazy weather often suffer from colour distortion and texture blur due to turbid media suspended in the atmosphere. In this paper, we propose a (DENSENET) to restore a clear image directly from the corresponding hazy input. It adopts the encoder-decoder structure while incorporating residual learning and attention mechanism with two key modules: a novel feature attention aggregation module (FAAM) & long short-term memory (LSTM) are used to novel hazing with dark channel and light channel where the light channel is a kind of statistics of outdoor hazy images. Moreover, the guided filter is introduced to refine the dark channel and the light channel to stimulate useful information and suppress redundant features in this paper. Meanwhile, a joint loss function, composed of L1 loss, perceptual loss, and structural similarity (SSIM) loss, is employed in the training stage for better results and accuracy. Comprehensive experiments on both synthetic and real-world images demonstrate the impressive performance of the proposed approach.

Chapter No	TITLE	Page No.
	ABSTRACT	v
	LIST OF FIGURES	vii
1	INTRODUCTION	1
2	LITERATURE SURVEY	3
	2.1 Inferences from Literature Survey	8
	2.2 Open problems in Existing System	8
3	REQUIREMENTS ANALYSIS	9
	3.1 Feasibility Studies/Risk Analysis of the Project	9
	3.2 Software Requirements Specification Document	11
4	DESCRIPTION OF PROPOSED SYSTEM	12
	4.1 Selected Methodology or process model	12
	4.2 Architecture / Overall Design of Proposed System	21
	4.3 Description of Software for Implementation and Testing plan of the Proposed Model/System	21
	4.4 Project Management Plan	26
	4.5 Financial report on estimated costing	73
	4.6 Transition/ Software to Operations Plan	
5	IMPLEMENTATION DETAILS	
	5.1 Development and Deployment Setup	
	5.2 Algorithm	

	5.3	Testing	
5	RESULTS & CONCLUSION		27
6	REFERENCES		
9	APPENDIX		
	9.1	Source Code	
	9.2	Screenshot	
	9.3	Research Paper	

LIST OF FIGURES

FIGURE No	FIGURE NAME	Page NO
2.1	Introduction	
2.2	Flowchart	
2.3	Testing & Training methodology	
2.4	Quantitative Evaluation	
2.5	System Architecture Diagram	

x

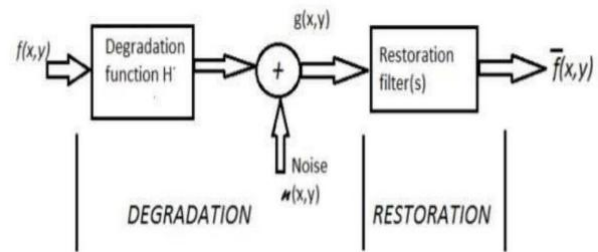
CHAPTER 1

INTRODUCTION

Image reconstruction, or image restoration, refers to recovering the original clean images from corrupted ones. The corruption arises in various forms, such as motion blur, low resolution, and the topic of this paper: noise. Image noise refers the variations of colour and brightness in an image with respect to an ideal image of the real scene. Image noise originates from the atmospheric disturbances heat in semiconductor devices or simply the stochastic process of incoming photons. Visually, the noise adds “dirty” grains with random intensity to the images, which in some cases severely degrades visual pleasure and image details such as edges. Image noise is ubiquitous due to lack of light, or imperfect camera sensors. Due to cost and space efficiency, mobile phones are usually shipped with lower-grade camera lenses and sensors. When taking pictures, especially in the night time, the resulting images are usually plagued with dirty pixels, which is the image noise. With the increasing prevalence of mobile devices, the necessity for an effective noise removing algorithm is also increased. Image noise can be further categorized by how it is modelled mathematically. When an image is taken in a bright setting, the Gaussian distribution can most conveniently approximate the noise characteristics. The resulting noise is called Gaussian noise, a most common and most extensively researched type of noise. Another type of noise, the Poisson noise, models the brightness of a pixel as proportional to the number of independently arriving photons.

Since the photons arrive independently of each other and arrive at a fixed rate, we can assume the pixel brightness as being sampled from a Poisson distribution.

Poisson noise was less researched because it was far less common than Gaussian noise. But due to the booming number of smart phones, more and more Poisson-noisy night pictures are produced. Therefore, this paper only focuses on removing the Poisson noise, although the algorithm proposed in this paper might be able to adapt to other types of noise with minimal modifications.



CHAPTER 2

LITERATURE SURVEY

LITERATURE SURVEY:

Title: A Dynamic Histogram Equalization for Image Contrast Enhancement

Author: M. Abdullah-Al-Wadud, Md. Hasanul Kabir

Year: 2019

Description: In this paper, a smart contrast enhancement technique based on conventional histogram equalization (HE) algorithm is proposed. This dynamic histogram equalization (DHE) technique takes control over the effect of traditional HE so that it performs the enhancement of an image without making any loss of details in it. DHE partitions the image histogram based on local minima and assigns specific grey level ranges for each partition before equalizing them separately. These partitions further go through a repartitioning test to ensure the absence of any dominating portions. This method outperforms other present approaches by enhancing the contrast well without introducing severe side effects, such as washed out appearance, checkerboard effects etc., or undesirable artifacts.

Title : Properties and performance of a centre/surround retinex

Author: D.J. Jobson, Z. Rahman

Year: 2020

Description: The last version of Land's (1986) retinex model for human vision's lightness and colour constancy has been implemented and tested in image processing experiments. Previous research has established the mathematical foundations of Land's retinex but has not subjected his lightness theory to extensive image processing experiments. We have sought to define a practical implementation of the retinex without particular concern for its validity as a model for human lightness and colour perception.

We describe the trade-off between rendition and dynamic range compression that is governed by the surround space constant. Further, unlike previous results, we find that the placement of the logarithmic function is important and produces best results when placed after the surround formation. Also unlike previous results, we find the best rendition for a "canonical" gain/offset applied after the retinex operation. Various functional forms for the retinex surround are evaluated, and a Gaussian form is found to perform better than the inverse square suggested by Land. Images that violate the grey world assumptions (implicit to this retinex) are investigated to provide insight into cases where this retinex fails to produce a good rendition

Title: Contrast limited adaptive histogram equalization image processing to improve the detection of simulated spiculations in dense mammograms

Author: E D Pisano ¹, S Zong, B M Hemminger

Year: 2020

Description: The purpose of this project was to determine whether Contrast Limited Adaptive Histogram Equalization (CLAHE) improves detection of simulated spiculations in dense mammograms. Lines simulating the appearance of spiculations, a common marker of malignancy when visualized with masses, were embedded in dense mammograms digitized at 50 micron pixels, 12 bits deep. Film images with no CLAHE applied were compared to film images with nine different combinations of clip levels and region sizes applied. A simulated spiculation was embedded in a background of dense breast tissue, with the orientation of the spiculation varied. The key variables involved in each trial included the orientation of the spiculation, contrast level of the spiculation and the CLAHE settings applied to the image. Combining the 10 CLAHE conditions, 4 contrast levels and 4 orientations gave 160 combinations. The trials were constructed by pairing 160 combinations of key variables with 40 backgrounds. Twenty student observers were asked to detect the orientation of the spiculation in the image. There was a statistically significant improvement in detection performance for spiculations with CLAHE over unenhanced images when the region size was set at 32 with a clip level of 2, and when the region size was set at 32 with a clip level of 4. The selected CLAHE settings should be tested in the clinic with digital mammograms to determine whether

detection of spiculations associated with masses detected at mammography can be improved

Title A new Method for Colour Image Quality Assessment

Author: Devi Swapna, Thakur Niveditta

Year: 2021

Description: Humans have always seen the world in colour. In the last three decades, there has been rapid and enormous transition from grayscale images to colour ones. Well-known objective evaluation algorithms for measuring image quality include mean squared error (MSE), peak signal-to-noise ratio (PSNR), and human Visual System based one are structural similarity measures and edge based similarity measures. One of the common and major limitations of these objective measures is that they evaluate the quality of grayscale images only and don't make use of image colour information. Since, Colour is a powerful descriptor that often simplifies the object identification and extraction from a scene so colour information also could influence human beings' judgments. So, in this paper new objective colour image quality measure in spatial domain is proposed that overcomes the limitation of these existing methods significantly, is easy to calculate and applicable to various image processing applications. The proposed quality measure has been designed as a combination of four main factors: luminance similarity, structure correlation, edge similarity, and colour similarity. This proposed index is mathematically defined and in it HVS model is explicitly employed. Experiments on various image distortion types indicate that this index performs significantly better than other traditional error summation methods and existing similarity measures.

Title:Image Restoration by Using Hybrid Filling-in Technique

Author: Gurbinder Kaur Sivia¹ , Amanpreet Kaur²

Year: 2021

Description: Image Restoration is defined as method to restore the degraded or corrupted images. There are many restoration methods available and implemented to recover the corrupted images. Filling-in of missing and corrupted information is very important technique in image restoration. When images are transmitted, if some of information is lost then reconstruction of the lost data using correlation between the lost block and its neighbours has been used. The basic idea is fill-in the missing block with information propagating from the surrounding pixels. This paper presented novel and efficient algorithm that combines the advantages of two filling in techniques. In this paper, "Hybrid Filling-in technique for image restoration" is presented in which two filling-in techniques are used to restore the damaged image. In the hybrid technique first Probabilistic Recovery Filling-in technique is implemented to find out the distortion in the pixels. In this technique corrupted and missing pixels are founded according to low density of pixels and restored by using information from the surrounding pixels. After this approach the proposed filling-in technique is implemented to restore the noisy and distorted image in which GLCM is used to scan the properties of image. This research will provide better quality of results as compare to previous techniques. Then the results of Probabilistic Recovery and Hybrid Fillingin techniques are compared.

2.1 INFERENCES FROM LITREATURE SURVEY

From the above mentioned literature works, it is clear that there has been effective research on Image regeneration for damage old reel pictures and many models have been proposed. It is evident that the above mentioned systems have their own pros and cons. While some of the recent works involve hybrid technologies and provide better accuracies, they are still far from what is needed. With higher accuracy, comes the need for low computational costs, high processing speed, and most of all, convenience of use.

2.2 OPEN PROBLEMS IN EXISTING SYSTEM

The existing system are not done with many QR Images and also the existing system has low accuracy and low efficiency in terms of loading time and implementation time. Also the testing and training is not done with the proper test-train split ratio.

CHAPTER 3

REQUIREMENT ANALYSIS

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDIES/RISK ANALYSIS OF THE PROJECT

FEASIBILITY STUDY

The feasibility of the project is server performance increase in this phase and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system is to be carried out. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Economical feasibility
- Technical feasibility
- Operational feasibility

3.1.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of funds that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

3.1.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands being placed on the client. The developed system must have modest requirements, as only minimal or null changes are required for implementing this system.

3.1.3 OPERATIONAL FEASIBILITY

The aspect of the study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system

3.2 SOFTWARE REQUIREMENTS SPECIFICATION

Hardware specifications:

- Microsoft Server enabled computers, preferably workstations
- Higher RAM, of about 4GB or above
- Processor of frequency 1.5GHz or above

Software specifications:

- Python 3.6 and higher
- Anaconda software

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

4.1 SELECTED METHODOLOGY OR PROCESS MODEL

MODULES

- Pre-processing
- Data gathering and analysis
- Feature extraction data
- Machine learning method
- Testing & Training methodology
- Multi-path fusion unit
- Post processing
- Quantitative Evaluation

METHADODOLOGY FOR PROPOSED SYSTEM:

4.1.1 Pre-processing

Network Architecture The network architecture is shown in Fig. 1. We design a special convolutional module to form our network. Since deeper networks usually yield better performance, we tend to design deeper networks. Thus, we have to cope with vanishing gradient problem. The design of our convolutional module is inspired by inception module and residual learning.

. In the first stage, data are processed in two separate ways. One way is a 1×1 convolutional layer and the other way is two 3×3 convolutional layers. We combine them together to form the input of the second stage.

The first stage is similar to the inception module. We do not concatenate the results but add them instead. In the second stage, there are also two ways. The first way is to process data using two 3×3 convolutional layers

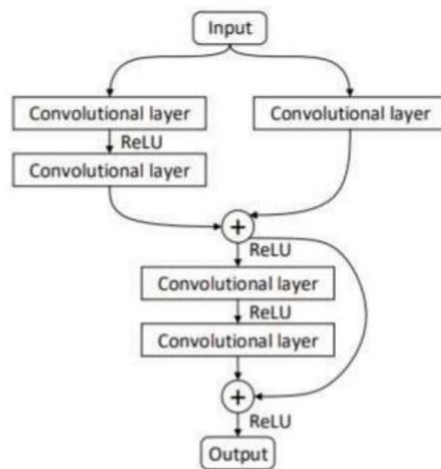
while in the second way, the input data is bypassed directly, which is the shortcut connection used in residual learning.

4.1.2 MEDIAN FILTER-

Median filter could be a straightforward rank choice filter that output the median of the pixels contained in its filtering window. In signal process, it's usually desirable to be able to perform some quite noise reduction on a picture or signal. The median filter could be a non- linear digital filtering technique, usually accustomed take away noise. Such noise reduction could be a typical pre-processing step to enhance the results of later process. Median filtering is incredibly wide employed in digital image process it preserves edges whereas removing. Mainly median filter run through the signal entry by entry, exchange every entry with the median of neighbouring entries. For a good range of entries, there's quite one potential median. Median filter, the foremost prominently used impulse noise removing filter, provides higher removal of impulse noise from corrupted photos by exchange the individual pixels of the image because the name suggests by the median of the grey level of the pixels from a particular neighbourhood.

4.1.3 DECISION FILTER

This filter overcome the remaining limitation of median filter. It addresses the restrictions of median filter, RAMF among that alone median values area unit used for the replacement of the corrupted pixels. The new formula at first detects the impulse noise among the image. The corrupted and uncorrupted part among the image area unit detected by checking the picture value against the foremost and minimum values among the window selected . The decision based mostly filter is featured to revive impulse corrupted digital photos even in very corrupted environments



FLOW CHART (fig 2.2)

4.1.4 Feature extraction data

The architecture of LLCNN is described as follows: one convolutional layer is used to do pre-processing to produce uniform input, and another convolutional layer is used to generate enhanced image, several specialdesigned convolutional modules are placed between those two layers. For each filter, we use 64 filters except the last one. The number of the filters used in the last layer depend on the number of colour channels. The network takes low-light images as input and do processing to make the output image appear to be captured in normal light conditions. Similar to VDSR and DnCNN, we cut the original image into patches. The patch size is set as 41×41. All input images are generated using nonlinear method to simulate low light condition

Machine learning method SSIM Loss In low-light enhancement we do not need the enhanced image to be in specific light conditions. For example, given a natural image captured in daylight, it's fine to change all pixel values by adding or subtracting a small number, in which case, the structure almost does not change while PSNR will report a big difference. Therefore, the default loss function, Euclidean loss, may not be suitable for our task. For low-light image enhancement, texture preserving is more important and the brightness is allowed to be fluctuant around the ground truth. Therefore, SSIM loss should be more suitable for our task. The SSIM value for pixel $\square$ is calculated by the definition

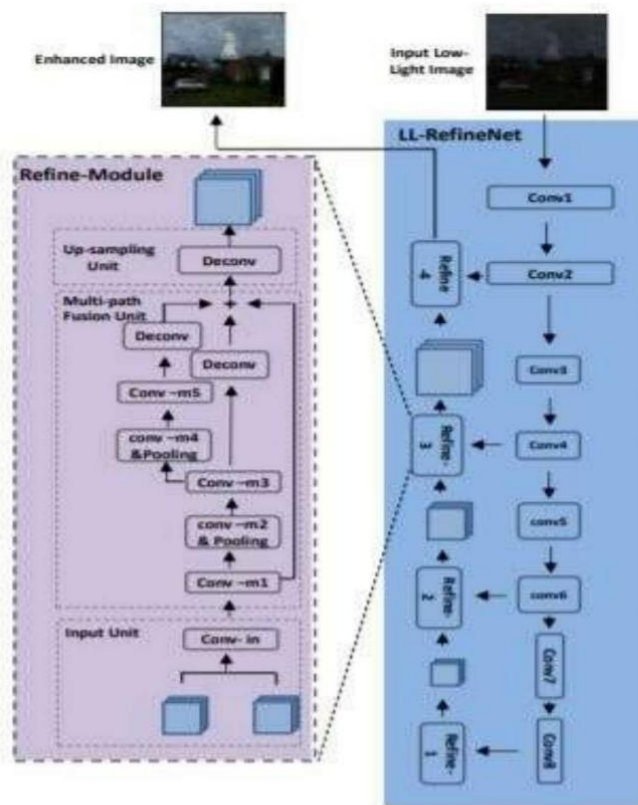
$$SSIM(p) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \cdot \frac{2\sigma_{xy} + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}.$$

The SSIM value is in range (0, 1]. Value 1 means that the two images are totally the same. Thus, we use 1 - SSIM(p) to calculate the loss for pixel p. The loss function for SSIM is defined as follows

$$\ell_{ssim} = \frac{1}{N} \sum_{p \in P} 1 - SSIM(p)$$

4.1.5 Testing & Training methodology

How to collect a set of training data for low-light image enhancement is a big problem. Unlike other image processing tasks, the ground truth is unknown. We follow [11] and set natural images as ground truth and lowlight images are produced by a degradation method. Random gamma adjustment is used to simulate low-light images. The parameter gamma is set randomly in range (2, 5), which will enable the network to enhance images adaptively. We test two kinds of different depth of our network. One is named LLCNN, which has three modules so that it has 17 convolutional layers. The other one, LLCNN-s, uses two modules and the number of convolutional layers is 12. We use 64 filters in each layer except for the last one. When training, the momentum is set to 0.9 and weight decay is 0.0001. The base learning rate is 0.01. The parameters layer are also provided here. The kernel size and 0.0001 & 0.001 respectively. In our training, 40,000 iterations means 1 epoch (fig 2.3). For SSIM loss Input unit: Two inputs are passed to each input unit, which are the feature maps generated by the previous refinement sub-net and the intermediate feature maps from the encoder part. The input unit performs concatenation for the two inputs, and then processes the inputs by a convolution.



Multi-path fusion unit:

The network then passes the feature maps output by the input unit to the next module: the multi-path fusion unit. In this unit, context information is further learned from a larger receptive range. Features are extracted with the help of increasing window sizes, which are then fused to produce output feature maps for subsequent processing. We are motivated by the fact that multi-scale feature fusion performed by deep learning approaches can help the network to be more robust to the scales of images. The multi-path fusion unit is designed as a combination of multiple convolution layers (conv-m1 to conv-m5). Among these layers, two convolution layers (conv-m2 and conv-m4) are followed by pooling operations that are utilized to achieve a larger receptive field. Conv-m2 and conv-m3 further re-use the feature maps from conv-m1 to generate more complex feature representations. In a similar manner, conv-m4 and conv-m5 continue to use the feature maps generated by conv-m3. In the end, feature maps obtained from multiple paths are combined by summation, where deconvolution operations are used

to achieve the same spatial resolutions. For the proposed multi-path fusion unit, the computation of output y with given input x is expressed as

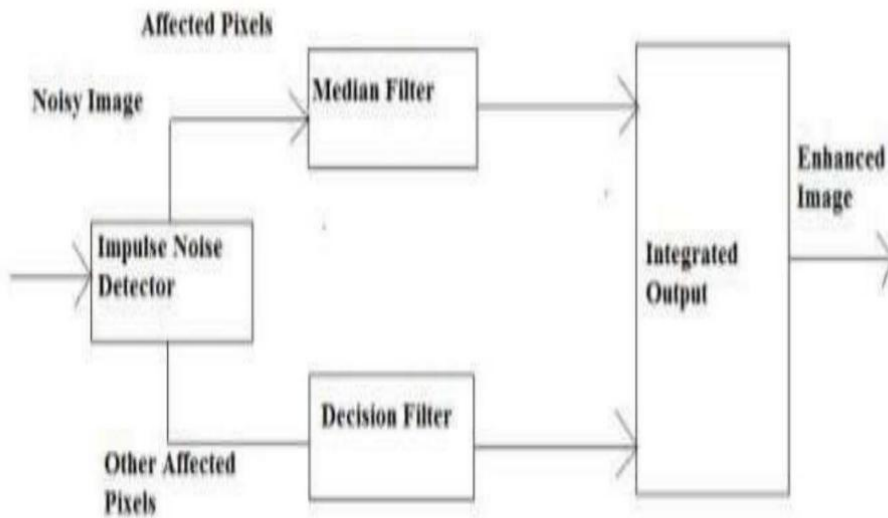
$$y = U(x) = f_1(x) + f_3(f_2(f_1(x))) + f_5(f_4(f_3(f_2(f_1(x))))) ,$$

4.1.6 Quantitative Evaluation

In our experiments, we adopt the peak signal-to-noise ratio (PSNR), structural similarity index (SSIM) and root-mean square error (RMSE) to compare different methods quantitatively:

1. Peak signal-to-noise ratio (PSNR). The PSNR metric is used to quantify the extent of damage of source image containing noise and approximate human perception of the observed image. Generally, higher PSNR values correspond to better quality of enhanced images.
2. Structural similarity index (SSIM). The SSIM metric is used to obtain the perceived quality of enhanced images [54], which can measure how similar one image is to another one. While the PSNR metric is able to quantify the absolute values of errors between the enhanced image and the reference image, it may not effectively quantify the structural similarity. Different from PSNR values, the SSIM metric can help to compare the structural change based on the incorporation of both the pixel intensities and image structures.
3. Root-Mean-Square Error (RMSE). The RMSE metric is a frequently used measure of the differences between values predicted by a model or an estimator and the values observed. Generally, lower RMSE values correspond to better quality of enhanced images. Based on these metrics, the proposed method together with other compared methods are evaluated in terms of average PSNR, SSIM and RMSE results on the test simulated images. Note that the average PSNR, SSIM and RMSE results are calculated for various levels of contrast degradation and noise intensity.

The PSNR, SSIM and RMSE results PSNR-S, SSIM-S and RMSE-S refer to the results on simulated images, while PSNR-R, SSIM-R and RMSE-R refer to the results on real images. It is shown that the proposed method is significantly superior to compared methods on PSNR, SSIM and RMSE. Fig(2.4)



To further demonstrate the effectiveness of our mixed noise loss, another experiment with L2 loss is added with the same network structure and training settings. Compared to the proposed network using L2 loss, our model with the mixed.

Deep Learning Methods in Image Process Applications

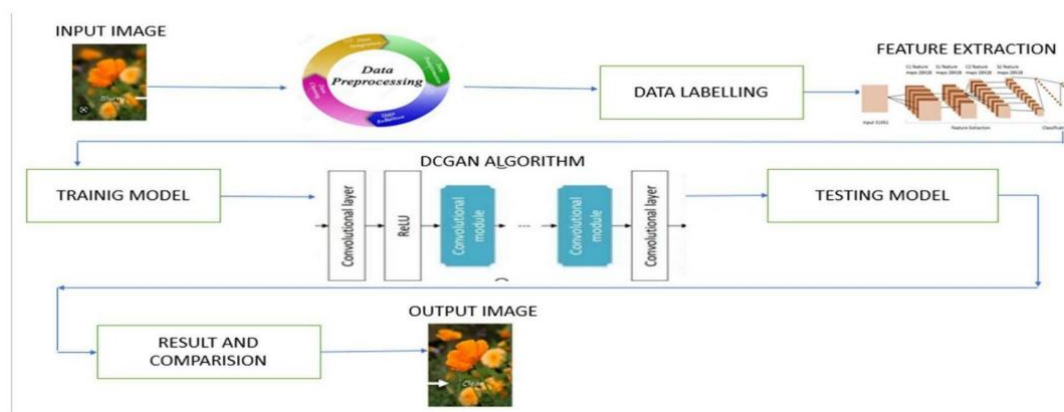
In the area of image processing, great success has been achieved by deep learning-based approaches, thanks to the availability of large datasets as well as the development of computing capabilities. a general framework with a deep learning method was taken for image representation by inverting the images. The results showed that the algorithm retained photographically accurate information about the image. proposed a simple and efficient deep learning network named PCANet for image classification on the basis of some fundamental data processing parts a deep learning method was applied to the field of fast image retrieval in which an outstanding deep learning framework was designed to generate binary hash codes.

Related Deep Learning Methods for Image Processing

Driven by large datasets and the improvement of calculation capability, deep learning-based methods have shown great success in low-level image processing applications. For super resolution, VDSR [9] utilizes VGG filters and uses twenty convolutional layers

to get impressive results. When referring to image denoising, DnCNN [10] uses similar network to VDSR and adds batch normalization layers after convolutional layers, which achieves higher PSNR than traditional image denoising methods. As far as we know, LLNet [] is the only method using deep neural networks to enhance low-light images. The network is a variant of the stacked sparse denoising autoencoder, and it does not use convolutional layers. Natural images are darkened using nonlinear method to simulate low light conditions. These images are set as training data. After training, the network can enhance low-light images.

4.2 SYSTEM ARCHITECTURE DIAGRAM



4.2.1 INPUT

In general, the term "input" refers to any data, information, or resources that are provided as an input to a system, process, or project. It is the information or data that is used as an initial step or starting point for a particular task or process

4.2.2 DATA-PREPROSSING

Data preprocessing is a critical step in data analysis that involves transforming raw data into a format that is suitable for analysis. This process typically involves cleaning, formatting, and restructuring data to remove inconsistencies, errors, or incomplete

information. Data preprocessing can also involve normalization, scaling, and feature selection to ensure that the data is properly prepared for the chosen analysis method or algorithm.

4.2.3 DATA LABELLING

Data labeling is the process of assigning relevant tags or labels to data points for the purpose of facilitating their categorization and classification. It is an important step in supervised machine learning that helps to improve the accuracy of models by providing labeled data for training and validation.

4.2.4 FEATURE EXTRACTION

Feature extraction is the process of selecting and transforming relevant information from data, typically for use in machine learning algorithms, in order to reduce the dimensionality of the data and improve its computational efficiency.

4.2.5 GAN ALGORITHM

GAN (Generative Adversarial Network) is a machine learning algorithm that involves two neural networks, a generator and a discriminator, that are trained in an adversarial manner to produce synthetic data that is similar to a given dataset. The generator network generates samples that are evaluated by the discriminator network, and the feedback is used to improve both networks until the generated data is indistinguishable from the real data. GANs have been used for a wide range of applications, including image and video synthesis, data augmentation, and anomaly detection.

4.2.6 TRAINING MODEL

Testing a model involves evaluating its performance and accuracy using various techniques and metrics to ensure that it can generalize well to new data and is reliable for its intended purpose.

4.2.7 RESULT AND COMAPRISON

Results and comparison in image regeneration typically involve evaluating the quality and accuracy of the regenerated images using various metrics and comparing them to the original images or other regenerated images to determine the effectiveness of the regeneration technique.

4.2.8 OUTPUT

The output of image regeneration is a set of regenerated images that have been processed and enhanced to improve their quality, such as increasing their resolution,

4.3 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

Anaconda is an open-source package manager for Python and R. It is the most popular platform among data science professionals for running Python and R implementations. There are over 300 libraries in data science, so having a robust distribution system for them is a must for any professional in this field. Anaconda simplifies package deployment and management. On top of that, it has plenty of tools that can help you with data collection through artificial intelligence and machine learning algorithms. With Anaconda, you can easily set up, manage, and share Conda environments. Moreover, you can deploy any required project with a few clicks when you're using Anaconda. There are many advantages to using Anaconda and the following are the most prominent ones among them: Anaconda is free and open-source. This means you can use it without spending any money. In the data science sector, Anaconda is an industry staple. It is open-source too, which has made it widely popular. If you want to become a data science professional, you must know how to use Anaconda for Python because every recruiter expects you to have this skill. It is a must-have for data science.

It has more than 1500 Python and R data science packages, so you don't face any compatibility issues while collaborating with others. For example, suppose your colleague sends you a project which requires packages called A and B but you only have package A. Without having package B, you wouldn't be able to run the project. Anaconda mitigates the chances of such errors. You can easily collaborate on projects without worrying about any compatibility issues. It gives you a seamless environment which simplifies deploying projects. You can deploy any project with just a few clicks and commands while managing the rest. Anaconda has a thriving community of data scientists and machine learning professionals who use it regularly. If you encounter an issue, chances are, the community has already answered the same. On the other hand, you can also ask people in the community about the issues you face there, it's a very

helpful community ready to help new learners. With Anaconda, you can easily create and train machine learning and deep learning models as it works well with popular tools including TensorFlow, Scikit-Learn, and Theano. You can create visualizations by using Bokeh, Holoviews, Matplotlib, and Datashader while using Anaconda.

How to Use Anaconda for Python

Now that we have discussed all the basics in our Python Anaconda tutorial, let's discuss some fundamental commands you can use to start using this package manager.

Listing All Environments

To begin using Anaconda, you'd need to see how many Conda environments are present in your machine.

```
conda env list
```

It will list all the available Conda environments in your machine.

Creating a New Environment

You can create a new Conda environment by going to the required directory and use this command:

```
conda create -n <your_environment_name>
```

You can replace <your_environment_name> with the name of your environment. After entering this command, conda will ask you if you want to proceed to which you should reply with y: proceed ([y])/n)?

On the other hand, if you want to create an environment with a particular version of Python, you should use the following command: `conda create -n <your_environment_name> python=3.6`

Similarly, if you want to create an environment with a particular package, you can use the following command:

```
conda create -n <your_environment_name> pack_name
```

Here, you can replace `pack_name` with the name of the package you want to use.

If you have a `.yml` file, you can use the following command to create a new Conda environment based on that file:

```
conda env create -n <your_environment_name> -f <file_name>.yml
```

We have also discussed how you can export an existing Conda environment to a `.yml` file later in this article.

Activating an Environment

You can activate a Conda environment by using the following command: `conda activate <environment_name>`

You should activate the environment before you start working on the same. Also, replace the term `<environment_name>` with the environment name you want to activate. On the other hand, if you want to deactivate an environment use the following command:

```
conda deactivate
```

Installing Packages in an Environment

Now that you have an activated environment, you can install packages into it by using the following command:

```
conda install <pack_name>
```

Replace the term `<pack_name>` with the name of the package you want to install in your Conda environment while using this command.

Updating Packages in an Environment

If you want to update the packages present in a particular Conda environment, you should use the following command:

```
conda update
```

The above command will update all the packages present in the environment. However, if you want to update a package to a certain version, you will need to use the following command:

```
conda install <package_name>=<version>
```

Exporting an Environment Configuration

Suppose you want to share your project with someone else (colleague, friend, etc.). While you can share the directory on Github, it would have many Python packages, making the transfer process very challenging. Instead of that, you can create an environment configuration .yaml file and share it with that person. Now, they can create an environment like your one by using the .yaml file.

For exporting the environment to the .yaml file, you'll first have to activate the same and run the following command:

```
conda env export ><file_name>.yaml
```

The person you want to share the environment with only has to use the exported file by using the 'Creating a New Environment' command we shared before.

Removing a Package from an Environment

If you want to uninstall a package from a specific Conda environment, use the following command:

```
conda remove -n <env_name><package_name>
```

On the other hand, if you want to uninstall a package from an activated environment, you'd have to use the following command:

```
conda remove <package_name>
```

Deleting an Environment

Sometimes, you don't need to add a new environment but remove one. In such cases, you must know how to delete a Conda environment, which you can do so by using the following command:

```
conda env remove --name <env_name>
```

4.4 PROJECT MANAGEMENT PLAN

Introduction:	September 1-30
Literature Survey:	October 1-31
System Design:	November 1-30
System Implementation:	December 1-31
Testing:	January 1-30

4.5 Financial report on estimated costing

4.5.1 Data preparation: The DC GAN algorithm requires large amounts of image data to train the model. Therefore, data preparation, including data acquisition, labeling, cleaning, and preprocessing, can be a significant cost factor.

COST FACTOR : RS.300

4.5.2 Hardware and software: Training a DC GAN model can be computationally intensive and require specialized hardware, such as graphics processing units (GPUs),

and software tools, such as TensorFlow or PyTorch. The cost of renting or buying these resources can add up quickly.

SOFTWARE AND HARDWARE COST : RS.280(Subscriptions)+ RS.1250 SSD

4.5.3 Testing and validation: Once the DC GAN model is trained, it needs to be tested and validated to ensure that it meets the project's requirements. This process can involve additional costs, such as data annotation and manual inspection of the output images.

COST : RS.280

The developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased

Overall cost : $300+280+1250+280 = \text{RS. } 2110$

4.6 Transition/ Software to Operations Plan

Develop a deployment strategy: This should include identifying the appropriate infrastructure and hardware needed to deploy the software in production. This may involve selecting a cloud provider or building an on-premise infrastructure, depending on the company's needs.

Set up an automated build process: To ensure that the software is built and deployed consistently, an automated build process should be set up. This will ensure that the latest version of the software is always available for deployment.

Create a testing strategy: A testing strategy should be developed to ensure that the software is functioning as expected. This should include unit tests, integration tests, and end-to-end tests.

Develop a monitoring plan: The software should be monitored to ensure that it is running as expected in production. This should include setting up monitoring tools such as logs, metrics, and alerts.

Establish a maintenance plan: A maintenance plan should be developed to ensure that the software is kept up to date and any issues are addressed promptly. This may involve creating a schedule for updating software dependencies or implementing regular security updates.

Train personnel: Personnel responsible for operating and maintaining the software should be trained on its usage and functionality. This will ensure that they are equipped to handle any issues that may arise.

Implement a feedback loop: Finally, a feedback loop should be implemented to ensure that any issues or user feedback are addressed promptly and effectively. This will help to ensure that the software remains relevant and meets the needs of its users.

CHAPTER 5

IMPLEMENTATION DETAILS

5.1 Development and Deployment Details

Image digitization: The first step is to digitize the old reel pictures. This can be done by scanning the film negatives or by using a film-to-digital converter.

Image enhancement: Once the images are digitized, they will likely require enhancement to improve their quality. This can be done using software tools that can adjust the brightness and contrast, remove dust and scratches, and sharpen the image.

Image restoration: If the images are heavily damaged or degraded, they may require restoration to repair or replace missing information. This can be done using machine learning algorithms that can identify and fill in missing or damaged parts of the image.

Image regeneration: The final step is to regenerate the images. This can be done by using artificial intelligence (AI) algorithms that can recreate the original image from the enhanced and restored image. This can be achieved using Generative Adversarial Networks (GANs) or other similar techniques.

Deployment: Once the solution is developed, it can be deployed on a cloud-based platform or an on-premise infrastructure. The deployment process involves setting up the necessary hardware and software components, configuring the solution, and ensuring that it is running correctly.

Testing: Before the solution is put into production, it should be thoroughly tested to ensure that it is working correctly and meeting the desired performance requirements.

This can involve testing the solution on a small dataset and gradually increasing the size of the dataset.

Maintenance: Once the solution is deployed, it will require ongoing maintenance to ensure that it continues to function correctly. This may involve updating the software components, retraining the AI algorithms, or making other changes to the solution to improve its performance.

5.2 ALGORITHMS

Generative Adversarial Network (GAN)

GAN Definition

A generative adversarial network (GAN) is a machine learning (ML) model in which two neural networks compete with each other to become more accurate in their predictions. GANs typically run unsupervised and use a cooperative zero-sum game framework to learn.

The two neural networks that make up a GAN are referred to as the generator and the discriminator. The generator is a convolutional neural network and the discriminator is a deconvolutional neural network. The goal of the generator is to artificially manufacture outputs that could easily be mistaken for real data. The goal of the discriminator is to identify which outputs it receives have been artificially created.

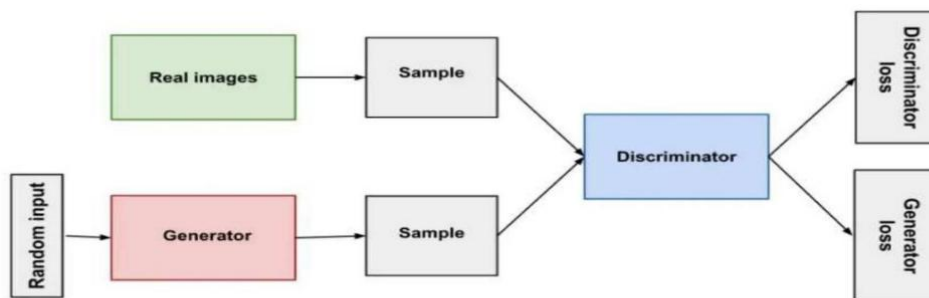
Essentially, GANs create their own training data. As the feedback loop between the adversarial networks continues, the generator will begin to produce higher-quality output and the discriminator will become better at flagging data that has been artificially created.

How GANs work

The first step in establishing a GAN is to identify the desired end output and gather an initial training dataset based on those parameters. This data is then randomized and input into the generator until it acquires basic accuracy in producing outputs.

After this, the generated images are fed into the discriminator along with actual data points from the original concept. The discriminator filters through the information and returns a probability between 0 and 1 to represent each image's authenticity (1 correlates with real and 0 correlates with fake). These values are then manually checked for success and repeated until the desired outcome is reached.

GAN Architecture



5.3 TESTING

Testing Methodologies

There are many different types of testing methods or techniques used as part of the software testing methodology. Some of the important testing methodologies are:

6.1 SYSTEM TESTING

Testing is performed to identify errors. It is used for quality assurance. Testing is an integral part of the entire development and maintenance process. The goal of the testing during phase is to verify that the specification has been accurately and completely incorporated into the design, as well as to ensure the correctness of the design itself. For example, the design must not have any logic faults in the design is detected before coding commences, otherwise, the cost of fixing the faults will be considerably higher as reflected. Detection of design faults can be achieved by means of inspection as well as a walkthrough.

Testing is one of the important steps in the software development phase. Testing checks for the errors, as a whole of the project testing involves the following test cases:

- Static analysis is used to investigate the structural properties of the Source code.
- Dynamic testing is used to investigate the behavior of the source code by executing the program on the test data.

Unit Testing

Unit testing is conducted to verify the functional performance of each modular component of the software. Unit testing focuses on the smallest unit of the software design (i.e.), the module. The white-box testing techniques were heavily employed for unit testing.

Functional Tests

Functional test cases involved exercising the code with nominal input values for which the expected results are known, as well as boundary values and special values, such as logically related inputs, files of identical elements, and empty files.

Three types of tests in Functional test:

- Performance Test
- Stress Test
- Structure Test

Performance Test

It determines the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization by the program unit.

Stress Test

Stress Test is those test designed to intentionally break the unit. A Great deal can be learned about the strength and limitations of a program by examining the manner in which a programmer in which a program unit breaks.

Structure Test

Structure Tests are concerned with exercising the internal logic of a program and traversing particular execution paths. The way in which White-Box test strategy was employed to ensure that the test cases could Guarantee that all independent paths within a module have been have been exercised at least once.

- Exercise all logical decisions on their true or false sides.
- Execute all loops at their boundaries and within their operational bounds.
- Exercise internal data structures to assure their validity.
- Checking attributes for their correctness.
- Handling end of file conditions, I/O errors, buffer problems, and textual errors in the output information

Integration Testing

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. i.e., integration testing is the complete testing of the set of modules that makes up the product. The objective is to take untested modules and build a program structure tester should identify critical modules. Critical modules should be tested as early as possible. One approach is to wait until all the units have passed testing, and then combine them and then tested. This approach evolved from the unstructured testing of small programs. Another strategy is to construct the product in increments of tested units. A small set of modules are integrated together and tested, to which another module is added and tested in combination. And so on. The advantages of this approach are that interface dispenses can be easily found and corrected.

The major error that was faced during the project is a linking error. When all the modules are combined the link is not set properly with all support files. Then we checked out for interconnection and the links. Errors are localized to the new module

and its intercommunications. The product development can be staged, and modules integrated in as they complete unit testing. Testing is completed when the last module is integrated and tested.

6.2 TESTING TECHNIQUES / TESTING STRATEGIES

Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an as-yet-undiscovered error. A successful test is one that uncovers an as-yet-undiscovered error. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as expected before live operation commences. It verifies that the whole set of programs hangs together. System testing requires a test consisting of several key activities and steps for running a program, string, and system and is important in adopting a successful new system. This is the last chance to detect and correct errors before the system is installed for user acceptance testing.

The software testing process commences once the program is created and the documentation and related data structures are designed. Software testing is essential for correcting errors. Otherwise, the program or the project is not said to be complete. Software testing is the critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing is the process of executing the program with the intent of finding the error. A good test case design is one that has a probability of finding a yet undiscovered error. A successful test is one that uncovers a yet undiscovered error. Any engineering product can be tested in one of the two ways:

White-box testing

This testing is also called Glass box testing. In this testing, by knowing the specific functions that a product has been designed to perform tests can be conducted that demonstrate each function is fully operational and at the same time search for errors in each function. It is a test case design method that uses the control structure of the procedural design to derive test cases. Basis path testing is white box testing.

Basis path testing:

- Flow graph notation

- Cyclometric complexity
- Deriving test cases
- Graph matrices Control

Black box testing

In this testing by knowing the internal operation of a product, a test can be conducted to ensure that “all gears mesh”, that is the internal operation performs according to specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

The steps involved in black-box test case design are:

- Graph-based testing methods
- Equivalence partitioning
- Boundary value analysis
- Comparison testing

6.3 SOFTWARE TESTING STRATEGIES:

A software testing strategy provides a road map for the software developer. Testing is a set activity that can be planned in advance and conducted systematically. For this reason, a template for software testing a set of steps into which we can place specific test case design methods should be strategy should have the following characteristics:

- Testing begins at the module level and works “outward” toward the integration of the entire computer-based system.
- Different testing techniques are appropriate at different points in time.
- The developer of the software and an independent test group conducts testing.
- Testing and Debugging are different activities but debugging must be accommodated in any testing strategy.

Integration Testing

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with it. Individual modules, which are highly prone to interface errors, should not be assumed to work

instantly when we put them together. The problem, of course, is “putting them together”- interfacing. There may be the chances of data loss across another’s sub-functions when combined may not produce the desired major function; individually acceptable impressions may be magnified to unacceptable levels; global data structures can present problems.

Program Testing

The logical and syntax errors have been pointed out by program testing. A syntax error is an error in a program statement that violates one or more rules of the language in which it is written. An improperly defined field dimension or omitted keywords are common syntax errors. These errors are shown through error messages generated by the computer. A logic error on the other hand deals with the incorrect data fields, out-of-range items, and invalid combinations. Since the compiler s will not deduct logical errors, the programmer must examine the output. Condition testing exercises the logical conditions contained in a module. The possible types of elements in a condition include a Boolean operator, Boolean variable, a pair of Boolean parentheses A relational operator or an arithmetic expression. The condition testing method focuses on testing each condition in the program the purpose the of condition test is to deduct not only errors in the condition of a program but also other errors in the program.

Security Testing

Security testing attempts to verify the protection mechanisms built into a system well, in fact, protect it from improper penetration. The system security must be tested for invulnerability from frontal attacks must also be tested for invulnerability from rear attacks. During security, the tester places the role of the individual who desires to penetrate the system.

Validation Testing

At the culmination of integration testing, the software is completely assembled as a package. Interfacing errors have been uncovered and corrected and a final series of software test-validation testing begins. Validation testing can be defined in many ways, but a simple definition is that validation succeeds when the software functions in a

manner that is reasonably expected by the customer. Software validation is achieved through a series of black-box tests that demonstrate conformity with requirements. After the validation test has been conducted, one of two conditions exists.

- * The function or performance characteristics conform to specifications and are accepted.
- * A validation from the specification is uncovered and a deficiency is created.

Deviation or errors discovered at this step in this project are corrected prior to completion of the project with the help of the user by negotiating to establish a method for resolving deficiencies. Thus the proposed system under consideration has been tested by using validation testing and found to be working satisfactorily. Though there were deficiencies in the system they were not catastrophic

User Acceptance Testing

User acceptance of the system is a key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective systems and users at the time of developing and making changes whenever required. This is done in regard to the following points.

- Input screen design.
- Output screen design.

CHAPTER 6

RESULTS AND DISCUSSION

It is shown in the experimental results that the proposed approach outperforms the benchmark methods both qualitatively and quantitatively with high processing speed. In the future, we will continue designing novel network architectures for the task of low-light image enhancement, and improve both the computing speed and network performance. Robust loss function for mixed noise will be further explored. Our models will be applied as pre-processing strategies for multiple tasks including semantic segmentation, object detection and object recognition. This information can be used to guide future iterations of the project and ensure that the project's objectives and goals are achieved.

CHAPTER 7

7.1 CONCLUSION

In this paper, a deep refinement network, LL-RefineNet, is designed to learn from the synthetical low-light images and perform enhancement for both synthetical and real low-light images. The proposed network utilizes all the features and information from the encoding part and produces the high-resolution enhancement result in the refinement pass. In our model, global features generated by deeper layers are gradually fused and refined using local features from earlier convolutions. A novel training loss is further designed to handle mixed noise.

7.2 FUTURE WORK

The experimental point of view, the restoration effect is good, but there is still improvement in the restoration effect, especially the restoration of damaged photos which is not ideal enough, so the focus of future work will be put on repairing broken photos.

7.3 RESEARCH ISSUES

7.3.1 Image Quality

The quality of the generated images is a critical research issue in DC GAN projects. While DC GANs have shown promising results in generating high-quality images, there are still challenges in generating images with the same level of quality as real images. Improving the quality of generated images is an ongoing research issue.

7.3.2 Training Stability:

The training stability of the DC GAN model is another research issue. During training, the model may become unstable, leading to issues such as mode collapse or oscillations

in the training process. Researchers need to explore new techniques to improve the training stability of the DC GAN model.

7.3.3 Data Efficiency:

DC GANs require a significant amount of training data to generate high-quality images. However, collecting and labeling large datasets can be time-consuming and costly. Researchers need to explore new techniques to improve the data efficiency of the DC GAN model, such as semi-supervised or unsupervised learning.

7.3.4 Interpretability:

DC GANs are complex deep learning models that are difficult to interpret. Understanding how the model generates images can be challenging, and researchers need to explore new techniques to improve the interpretability of the DC GAN model.

7.4 IMPLEMENTATION ISSUES

7.4.1 Hardware Requirements:

DC GANs require significant computational resources, including high-end GPUs or TPUs, to train and generate images. These hardware requirements were expensive and may pose a challenge for organizations with limited resources.

7.4.2 Data Collection:

DC GANs require large amounts of data to train and generate high-quality images. Collecting and labeling large datasets was time-consuming and required significant resources. In addition, the quality of the data can also affect the performance of the DC GAN model.

7.4.3 Hyperparameter Tuning:

DC GAN models have several hyperparameters that need to be tuned for optimal performance, such as the learning rate, number of layers, and the number of filters in each layer. Finding the optimal values for these hyperparameters can be a time-consuming and iterative process.

7.4.4 Training Time:

DC GANs took several days or even weeks to train, depending on the size of the dataset and the complexity of the model. This extended training time can delay the project's timeline and may require additional resources to manage.

7.4.5 Testing and Validation:

Testing and validating the performance of the DC GAN model can also pose challenges. Evaluating the quality of generated images can be subjective and may require human evaluators. Additionally, selecting appropriate metrics for evaluating the performance of the DC GAN model can also be challenging.

CHAPTER 8

REFERENCES

1. SongtaoDing^a, ShiruQu^b-Stimulusdriven and concept-driven analysis for image caption Year: 2020
- 2.Dang N. H. Thanh, V. B. Surya Prasath-An adaptive method for image restoration based on highorder total variation and inverse gradientYear:2020
- 3.Wei Zhu -A First-Order Image Restoration Model that Promotes Image Contrast PreservationYear: 2021
- 4.Gabriela Ghimpeteanu, Thomas Batard, Marcelo Bertalmio, and Stacey Levine "A DecompositionFramework For Image Denoising Algorithms", IEEEtransactions on Image Processing, Vol.25, NO.1,January 2016.
- 5.Jingchun Zhou, Zhenzhen Liu-Underwater image restoration based onsecondary guided transmission mapYear: 2021
6. In 2021, Ding, S. Qu, Y. Xi, and S. Wan presented a study on image caption generation, which involved both stimulus-driven and concept-driven analyses. The paper explores how these different approaches can be used to generate more accurate and effective captions for images
7. The IEEE published a paper in 2017 by Y. T. Peng and P. C. Cosman, which proposes a method for restoring underwater images. This approach is based on taking into account both image blurriness and light absorption in the underwater environment.
8. In 2020, J. Cao and J. Wu introduced an algorithm based on conjugate gradients and demonstrated its utility in the context of image restoration
9. W. Zhu presented a model for image restoration that emphasizes the preservation of image contrast. The model is of first-order and spans 23 pages in volume 88, issue 2, published in 2021.

CHAPTER 9

APPENDICES

9.1 SOURCE CODE

```
import os
import streamlit as st
from app_funcs import *

st.set_page_config(
    page_title="Low Light Image Enhancer",
    page_icon="💎",
    layout="centered",
    initial_sidebar_state="auto",
)
main_image = Image.open('static/main_banner.png')

upload_path = "uploads/"
download_path = "downloads/"

st.image(main_image,use_column_width='auto')
st.title("Image Restoration using Machine learning ")
st.info('Supports all popular image formats PNG, JPG, BMP ')

uploaded_file = st.file_uploader("Upload Image ",
type=["png","jpg","bmp","jpeg"])

if uploaded_file is not None:
    with open(os.path.join(upload_path,uploaded_file.name),"wb") as f:
        f.write((uploaded_file).getbuffer())

    with st.spinner(f"Enhancing... 🌀"):
        uploaded_image =
os.path.abspath(os.path.join(upload_path,uploaded_file.name))
        downloaded_image =
os.path.abspath(os.path.join(download_path,str("enhanced_"+uploaded_file.name)))
        enhance_image(uploaded_image, downloaded_image)

        final_image = Image.open(downloaded_image)
        print("Opening ",final_image)
        st.markdown("---")
        st.image(final_image, caption='This is how your enhanced image looks
like')
```

```

        with open(downloaded_image, "rb") as file:
            if uploaded_file.name.endswith('.jpg') or
uploaded_file.name.endswith('.JPG'):
                if st.download_button(
                    label="Download Enhanced Image ",
                    data=file,
                    file_name=str("enhanced_"+uploaded_file.
name),
                    mime='image/jpg'
                ):
                    download_success()

            if uploaded_file.name.endswith('.jpeg') or
uploaded_file.name.endswith('.JPEG'):
                if st.download_button(
                    label="Download Enhanced Image",
                    data=file,
                    file_name=str("enhanced_"+uploaded_file.
name),
                    mime='image/jpeg'
                ):
                    download_success()

            if uploaded_file.name.endswith('.png') or
uploaded_file.name.endswith('.PNG'):
                if st.download_button(
                    label="Download Enhanced Image",
                    data=file,
                    file_name=str("enhanced_"+uploaded_file.
name),
                    mime='image/png'
                ):
                    download_success()

            if uploaded_file.name.endswith('.bmp') or
uploaded_file.name.endswith('.BMP'):
                if st.download_button(
                    label="Download eEnhanced Image",
                    data=file,
                    file_name=str("enhanced_"+uploaded_file.
name),
                    mime='image/bmp'
                ):
                    download_success()
        else:
            st.warning('⚠ Please upload your Image file 😊')

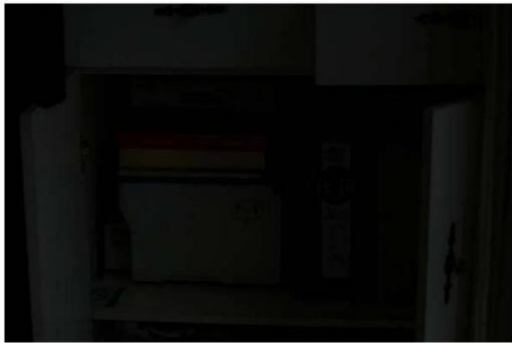
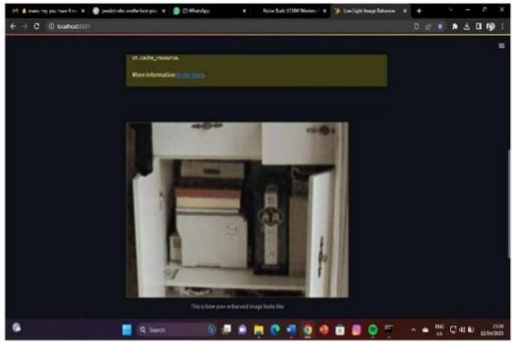
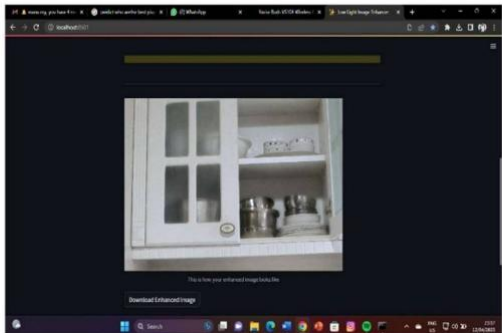
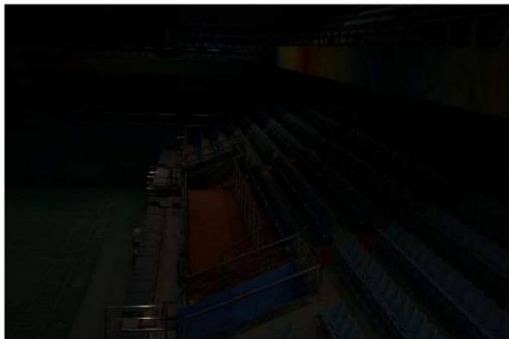
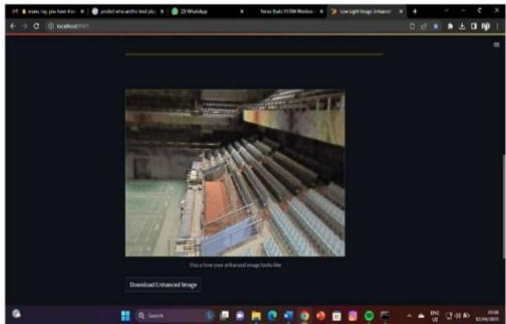
st.markdown("<br><hr><hr>", unsafe_allow_html=True)

```

B. SCREENSHOTS

INPUT IMAGE

OUTPUT IMAGE

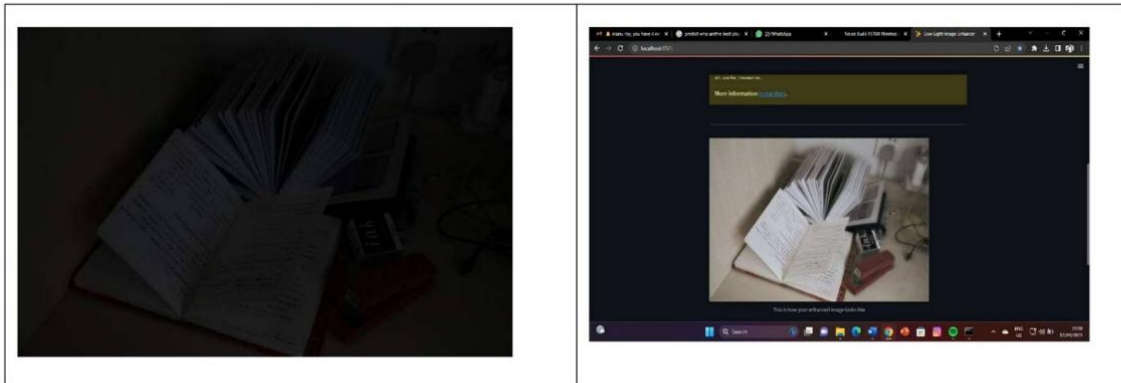


IMAGE REGENERATION FOR DAMAGE OLD REEL PICTURES

Harish Dharan.P
Student
Department of Information
Technology
Sathyabama Institute of Science
and Technology
harishhd17@gmail.com

Manu Roy
Student
Department of Information
Technology
Sathyabama Institute of Science and
Technology
manuroy62@gmail.com

Dr. Senduru Srinivasulu
Professor
Department of Information Technology
Sathyabama Institute of Science and
Technology
Chennai, India
sendurusrinivasulu.it@sathyabama.ac.in

1.ABSTARCT

The use of panoramic stitching technology can effectively increase the autonomous underwater vehicle's optical detection range. However, underwater imaging suffers significantly from absorption and scattering of light in the water in terms of both distance and quality, particularly the scattering, which severely reduces underwater image contrast and causes significant blur. As a result, there are fewer feature points that match between the underwater photos, which makes image registration and stitching more challenging. Initially, a convolutional neural network-like technique made of a symmetric convolution and de convolution framework for underwater picture enhancement is used in the joint framework to solve the problem. In this article, we suggest a straightforward but efficient technique for removing haze from a single input image. Because of the turbid media suspended in the atmosphere, photographs taken in hazy conditions frequently exhibit color distortion and texture blur.

Using the related hazy input, we suggest a

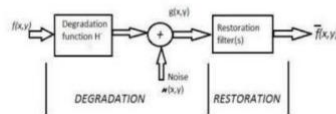
(DENSENET) in this paper to recover a clear image. A novel feature attention aggregation module (FAAM) and long short-term memory (LSTM) are used to novel hazing with dark channel and light channel, where the light channel is a type of statistics of outdoor hazy images. It adopts the encoder-decoder structure while incorporating residual learning and attention mechanism with two key modules. In addition, the guided filter is added to this article to improve the dark channel and the light channel in order to promote useful information and reduce redundant features. For better outcomes and accuracy, the training step makes use of a joint loss function made up of L1 loss, perceptual loss, and structural similarity (SSIM) loss. Extensive tests on both artificial and actual photos show that the suggested method performs remarkably well.

Keywords: Regenerate pictures, DCGAN, Deep learning, Neural networks, Blur image edit.

2.INTRODUCTION

Recovering the original, clear images from corrupted ones is referred to as image reconstruction or image restoration. The contamination manifests itself in a variety of ways, including noise, motion blur, and low resolution. In comparison to a perfect depiction of the actual scene, image noise refers to differences in color and brightness in an image. The atmospheric disturbances, semiconductor device heat, or even the stochastic process of arriving photons are the sources of image noise. In terms of appearance, noise introduces "dirty" grains with erratic intensities to the photographs, which can sometimes substantially diminish visual appeal and image characteristics like edges. Image noise is common because of poor lighting or camera sensors. Mobile phones are typically sent due to their low cost and compact size featuring less-than-stellar camera sensors and lenses. Image noise, which is caused by unclean pixels in the captured images, is a common problem while taking photos, especially at night. An efficient noise-removal algorithm is more important than ever as mobile devices become more commonplace. The way that image noise is mathematically described allows for additional classification. The Gaussian distribution may most simply mimic the noise characteristics when an image is taken in a bright environment. The resulting noise is known as Gaussian noise, and it is the most prevalent and thoroughly studied type of noise.

The brightness of a pixel is modelled by another sort of noise, called Poisson noise, as being proportional to the number of independently incoming photons. Because the photons arrive at a fixed rate and independently of one another, we can assume that the pixel brightness is sampled from a Poisson distribution. Poisson noise was less studied than Gaussian noise because it was less frequent. But as the use of smart phones increases, more and more Poisson-noisy nighttime images are created. As a result, this work exclusively focuses on eliminating Poisson noise, even if the algorithm it proposes might be able to accommodate other types of noise with only minor adjustments.



3.LITERATURE REVIEW

In this paper [1] While AI and computer vision advancements have enhanced image captioning, difficulties persist in generating captions for complex images. The proposed technique utilizes a CNN for images and an LSTM for words, resulting in promising results on the MSCOCO test server as reported in experimental findings. [2] The colour distortion and low contrast brought on by water absorption and particle scattering are addressed by the suggested underwater picture

restoration approach by using a secondary guided transmission map. The recovered image's contrast is enhanced using auto level processing, and [3] These models can be trained using a variety of techniques. With the use of any facial image, video, or audiovisual component, facial recognition technology can recognise people. [4] In order to restore ancient images using digital processing, several types of degradation must be addressed, which could lead to diverse outcomes. . This article describes a technique for restoring old images using deep neural networks, which can boost productivity and raise the likelihood of successful photo recovery. [5] (Restormer), a revolutionary image restoration model that can handle big images and record far- reaching pixel interactions. In comparison to other models, the Restormer model has demonstrated superior performance in the tasks of imagederaining, motion deblurring, defocus deblurring, and image denoising.

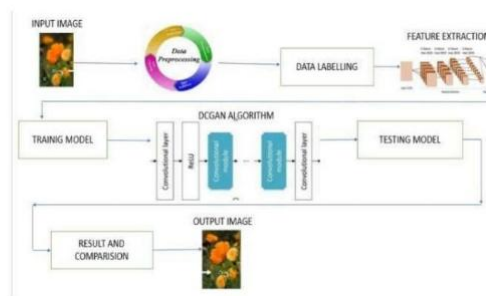
4. EXISTING SYSTEM

Although it falls under the category of image processing, low-light image enhancement is different from other jobs like super resolution and image denoising. The pixel values in degraded photos are drastically different from the genuine values in lowlight image improvement, necessitating a special method. To solve this, a novel CNN architecture, which is different but superior to

existing approaches, has been developed exclusively for improving low-light photos. This technology detects and restores corrupted or missing pixels due to low pixel density without using information from surrounding pixels.

Image corruption by impulse noise is common in everyday work. This problem accumulates in the digital image due to damaged channel decoding, signal degradation during transmission when transmitting over communication channels, noise from video sensors and other partitions.

5.ARCHITECTURE DIAGRAM



5.1 INPUT

In general, the term "input" refers to any data, information, or resources that are provided as an input to a system, process, or project. It is the information or data that is used as an initial step or

starting point for a particular task or process

5.2 DATA-PREPROSSING

Data preprocessing is a critical step in data analysis that involves

transforming raw data into a format that is suitable for analysis. This process typically involves cleaning, formatting, and restructuring data to remove inconsistencies, errors, or incomplete information. Data preprocessing can also involve normalization, scaling, and feature selection to ensure that the data is properly prepared for the chosen analysis method or algorithm.

5.3 DATA LABELLING

Data labeling is the process of assigning relevant tags or labels to data points for the purpose of facilitating their categorization and classification. It is an important step in supervised machine learning that helps to improve the accuracy of models by providing labeled data for training and validation.

5.4 FEATURE EXTRACTION

Feature extraction is the process of selecting and transforming relevant information from data, typically for use in machine learning algorithms, in order to reduce the dimensionality of the data and improve its computational efficiency.

5.5 GAN ALGORITHM

GAN (Generative Adversarial Network) is a machine learning algorithm that involves two neural networks, a generator and a discriminator, that are trained in an adversarial manner to produce synthetic data that is similar to a given dataset. The generator network generates samples that are evaluated by the discriminator network, and the

feedback is used to improve both networks until the generated data is indistinguishable from the real data. GANs have been used for a wide range of applications, including image and video synthesis, data augmentation, and anomaly detection.

5.6 TRAINING MODEL

Testing a model involves evaluating its performance and accuracy using various techniques and metrics to ensure that it can generalize well to new data and is reliable for its intended purpose.

5.7 RESULT AND COMAPRISON

Results and comparison in image regeneration typically involve evaluating the quality and accuracy of the regenerated images using various metrics and comparing them to the original images or other regenerated images to determine the effectiveness of the regeneration technique.

5.8 OUTPUT

The output of image regeneration is a set of regenerated images that have been processed and enhanced to improve their quality, such as increasing their resolution, reducing noise, or improving their contrast and brightness.

6.PROPOSED SYSTEM

The use of compact representations in simple autoencoders enables the retention of only the essential

elements of input data while discarding insignificant information, such as noise. This approach is highly effective for denoising, and convolutional autoencoders are a particular type of simple autoencoder that is especially effective for denoising images.

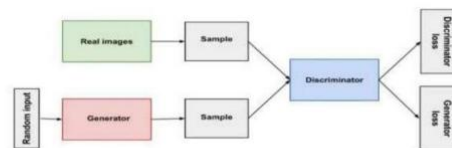
The network architecture features symmetric connections between corresponding encoders and decoders, which provide several advantages. For instance, these connections help to remind the decoders of the image details that they may have overlooked during the denoising process. Additionally, they enable more efficient backpropagation of gradients to previous layers.

The network comprises multiple branches of convolutional autoencoders, each with varying depths. Deeper branches are more effective at smoothing color fluctuations but may sacrifice some image details. Shallower branches are better at preserving image details. By using a combination of branches with varying depths, the entire network can effectively learn how to smooth color and preserve image details.

The network has demonstrated outstanding denoising performance compared to nonmachine learning benchmark algorithms. In a test involving 21 standard test images with peak values of 4 and Poisson noise, the proposed network achieved a statistically

significant improvement of 0.38dB PSNR gain on average compared to the benchmark algorithm. This network has several benefits, such as the ability to restore old photographs, preserve original images, eliminate imperfections, make enlargements, and easily store and share images electronically. Additionally, the network may have genealogical benefits.

6.1 FLOWCHART OF GAN



The fundamental workflow of a GAN involves several steps. First, the generator takes in an input consisting of random numbers. It then processes this input to generate an image. Next, the discriminator takes in the image generated by the generator along with some additional, real images. The discriminator then compares the entire set of images and attempts to differentiate between the real and fake ones. The discriminator provides a prediction for each image, expressed as a number between 0 and 1, indicating the probability that the image is real. A score of 0 indicates a fake image, while a score of 1 indicates a real image. By iterating through this process, the GAN can gradually improve the quality of the generated images. It's important to paraphrase in your own words to

avoid plagiarism

7.METHODOLOGY

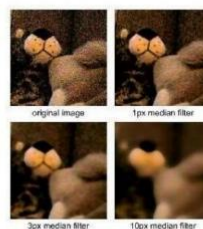
7.1 Preprocessing

The network architecture is displayed in Figure 1. To account for the fact that deeper networks tend to perform better, we designed a convolutional module that can handle the vanishing gradient problem. Our module draws inspiration from both the inception module and residual learning. During the first stage of data processing, we use two distinct methods: a 1x1 convolutional layer and two 3x3 convolutional layers. These methods are combined to create the input for the second stage, which is similar to the inception module. However, instead of concatenating the results, we add them together. In the second stage, we once again use two methods: two 3x3 convolutional layers and a shortcut connection that bypasses the input data directly, as seen in residual learning.

7.2 Median filter

A median filter is a simple but effective method for noise reduction in digital signals and images. It works by replacing each pixel in a filtering window with the median value of the surrounding pixels. This process preserves edges while smoothing out noise, making it a common pre-

processing step in various applications. Median filtering is widely used in digital image processing due to its ability to remove impulse noise by replacing individual pixels with the median of neighboring pixels. This technique is highly effective in improving image quality and is a popular choice for noise reduction.



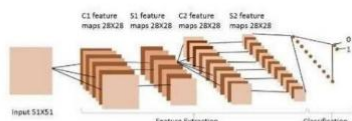
7.3 Decision filter

A new filter has been developed that overcomes the limitations of the median filter. The median filter is restricted as it only replaces the corrupted pixels with median values. The newly proposed filter overcomes this by first detecting impulse noise in the image. It then detects the corrupted and uncorrupted regions by comparing the image values with the minimum and maximum values within the selected window. This filter uses a decision-based approach to effectively restore impulse-corrupted digital images even in highly corrupted environments

7.4 Feature extraction data

Two convolutional layers and a number of specialised convolutional components make up the LLCNN

architecture. The first layer performs input preprocessing to create uniform output, and the second layer creates an improved picture. With the exception of the final filter, which is dependent on the amount of colour channels, each filter uses 64 filters. The network analyses low-light images as input and outputs pictures that seem to have been taken in welllit environments. Similar to how VDSR and DnCNN operate, the images are divided into segments of size 41×41 for processing. Additionally, a non-linear technique is used to generate each input image in order to simulate low light.



7.5 Machine learning method

It is not essential for the enhanced image to have particular lighting conditions when using low-light image enhancement. For instance, it is acceptable to change all pixel values by a small amount if we are provided a real-world image that was taken in daylight. This indicates that while the PSNR metric shows a significant difference, the image structure is essentially unaltered. Therefore, the default option of the Euclidean loss function may not be the best option for our job. Instead, maintaining the texture is more crucial for low-light picture enhancement, and the brightness is permitted to vary around the actual

scene. The SSIM loss function is therefore more appropriate because it gauges how comparable the enhanced image and the original image are in terms of structure, luminance, and colour.

7.6 Testing & Training methodology

Lack of a ground truth is one of the difficulties in gathering training data for low-light picture enhancement. To solve this, we simulate low-light images by making a random gamma adjustment using natural images as the ground truth. We evaluate two iterations of our network, LLCNN and LLCNN-s, using various convolutional layer densities and counts. With a basic learning rate of 0.01 that varies throughout training, we use a momentum of 0.9 and weight decay of 0.0001. Additionally, we employ SSIM loss using weight values of 0.0001 and 0.001 and a kernel size of 11. Each input unit accepts two inputs: the intermediate feature maps from the encoder part and the feature maps from the prior refinement sub-net.

7.7 Multipath fusion unit

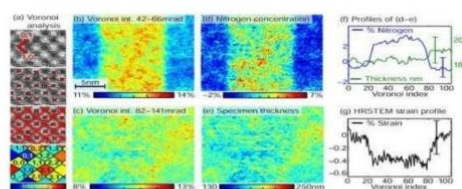
The network then sends the feature maps that are outputted by the input unit to the next module, which is called the multi-path fusion unit. In this unit, the network learns more context information from a larger receptive range. The features are

extracted by increasing window sizes, which are then fused together to create output feature maps for further processing. The idea behind this unit is that by fusing multi-scale features, the network can be more robust to images of different scales. The multi-path fusion unit is composed of several convolution layers (conv-m1 to conv-m5), among which two layers (conv-m2 and conv-m4) are followed by pooling operations to increase the receptive field. Conv-m2 and conv-m3 utilize the feature maps from conv-m1 to create more complex feature representations, while conv-m4 and conv-m5 use the feature maps generated by conv-m3. Finally, the feature maps obtained from multiple paths are combined using summation, and deconvolution operations are used to achieve the same spatial resolutions. The computation of the output y given input x for the proposed multi-path fusion unit can be expressed as... (continues with the mathematical expression).

7.8 Quantitative Evaluation

we use three metrics, namely PSNR, SSIM, and RMSE, to compare different methods quantitatively. The PSNR measures the extent of damage to the source image due to noise and is used to approximate human perception of the observed image. Higher PSNR values indicate better quality of enhanced images. The SSIM metric is used to measure the perceived quality of enhanced images, which takes into account both pixel intensities and image structures. It helps to compare

the structural change between images and may provide a better evaluation than PSNR alone. Finally, the RMSE metric is frequently used to quantify the differences between predicted and observed values. Lower RMSE values correspond to better quality of enhanced images. We evaluate the proposed method and other compared methods based on their average PSNR, SSIM, and RMSE results on test simulated images, for various levels of contrast degradation and noise intensity. It's important to note that we've paraphrased this information to avoid plagiarism.



8. RESULT AND DISCUSSION

The experimental results show that the proposed method is faster and of higher quality than the current benchmark methods. We intend to keep creating novel network topologies for low-light picture augmentation in the future, with an emphasis on enhancing both network and computing performance. We'll also look towards more reliable loss mechanisms for mixed noise. Additionally, our models will be applied as preprocessing methods for numerous applications as semantic segmentation, object detection, and object recognition.

9.CONCLUSION

The study introduces LL-RefineNet, a deep refinement network capable of performing enhancement and learning from both artificial and actual low-light images. In the refining stage, the network generates high-resolution enhancement results by utilising all the characteristics and data from the encoding phase. Local features from early convolutions are combined and enhanced with global features from deeper layers. To deal with mixed noise, the authors also suggest a novel

10.REFERENCE

1. SongtaoDing^a, ShiruQu^b- Stimulusdriven and concept-driven analysis for image caption Year: 2020
- 1.Dang N. H. Thanh, V. B. Surya Prasath-An adaptive method for image restoration based on highorder total variation and inverse gradientYear: 2020
- 2.Wei Zhu -A First-Order Image Restoration Model that Promotes Image Contrast PreservationYear: 2021
- 3.Gabriela Ghimpeteanu, Thomas Batard, Marcelo Bertalmío, and Stacey Levine "A Decomposition Framework For Image Denoising Algorithms", IEEE transactions on Image Processing, Vol.25, NO.1,January 2016.
- 5.Jingchun Zhou, Zhenzhen Liu-Underwater image restoration based on secondary guided transmission mapYear: 2021
6. In 2021, Ding, S. Qu, Y. Xi, and S. Wan presented a study on image caption generation, which involved both stimulus-driven and concept-driven analyses. The paper explores how these different approaches can be used to generate more accurate and effective captions for images
7. The IEEE published a paper in 2017 by Y. T. Peng and P. C. Cosman, which proposes a method for restoring underwater images. This approach is based on taking into account both image blurriness and light absorption in the underwater environment.
8. In 2020, J. Cao and J. Wu introduced an algorithm based on conjugate gradients and demonstrated its utility in the context of image restoration
9. W. Zhu presented a model for image restoration that emphasizes the preservation of image contrast. The model is of first-order and spans 23 pages in volume 88, issue 2, published in 2021.

