AUTO COLORIZATION OF BLACK AND WHITE IMAGES USING AUTO ENCODERS TECHNIQUE

A Project report submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

B.Chandra Sekara Raju (320126512L04) R.Rakesh (319126512048)

P.Padma Ragam (319126512044) M.Manohar Reddy (320126512L05)

Under the guidance of

Mr.V.VIJAY KUMAR RAJU

(Assistant Professor)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES
(UGC AUTONOMOUS)
(*Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAAC*)
Sangivalasa, bheemili mandal, visakhapatnam dist.(A.P)
2022-2023

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES (UGC AUTONOMOUS)

(Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAAC)

Sangivalasa, Bheemili mandal, Visakhapatnam dist.(A.P)

PRAGNANAM BRAHMA

ANITS

CERTIFICATE

This is to certify that the project report entitled **"AUTO COLORIZATION OF BLACK AND WHITE IMAGES USING AUTO ENCODERS TECHNIQUE"** submitted by **B.Chandra Sekara Raju (320126512L04), R.Rakesh (319126512048), P.Padma Ragam (319126512044), M.Manohar Reddy (320126512L05)** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Electronics & Communication Engineering** of Anil Neerukonda Institute of technology and sciences, Visakhapatnam is a record of bonafide work carried out under my guidance and supervision.

Project Guide	**Head of the Department**
[illegible]	[illegible]
Mr.V.Vijay Kumar Raju	**Dr. B.Jagadeesh**
Assistant Professor	**Professor&HOD**
Department of E.C.E	Department of E.C.E
ANITS	ANITS

Assistant Professor
Department of E.C.E.
Anil Neerukonda
Institute of Technology & Sciences
Sangivalasa, Visakhapatnam-531 162

Head of the Department
Department of E C E
Anil Neerukonda Institute of Technology & Sciences
Sangivalasa - 531 162

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our project guide **Mr.V.Vijay Kumar Raju** (Assistant Professor), Department of Electronics and Communication Engineering, ANITS, for his/her guidance with unsurpassed knowledge and immense encouragement. We are grateful to **Dr.B.Jagadeesh**, Professor and Head of the Department, Electronics and Communication Engineering, for providing us with the required facilities for the completion of the project work.

We are very much thankful to the **Principal and Management, ANITS, Sangivalasa,** for their encouragement and cooperation to carry out this work.

We express our thanks to all **teaching faculty** of Department of ECE, whose suggestions during reviews helped us in accomplishment of our project. We would like to thank **all non-teaching staff** of the Department of ECE, ANITS for providing great assistance in accomplishment of our project.

We would like to thank our parents, friends, and classmates for their encouragement throughout our project period. At last but not the least, we thank everyone for supporting us directly or indirectly in completing this project successfully.

PROJECT STUDENTS

B.CHANDRA SEKARA RAJU(320126512L04),
R.RAKESH(319126512048),
P.PADMA RAGAM(319126512044),
M.MANOHAR REDDY(320126512L05).

ABSTRACT

Color information is the strong descriptor of an image and such information is brightness known as luminance and color known as chrominance. Colourization of images is done manually for a long time. In order to increase the speed and accuracy we use a technique called auto-encoding and auto-decoding. By using this method we perform down sampling of an image for processing and up-sampling of an image for reconstruction of target image. The electronic microscope produces a black & white images but some of the microscopes are producing color images based on their magnified level, so that to overcome this drawback we introduce a neural network to produce the color images and which is not depending on any magnified level. There are many types of colourization techniques there like RGB colorization, Pro-Photo, Rec709, YUV color spaces etc. In this project we implemented deep-learning and convolution neural networking techniques with keras sequential algorithm.

It includes pre-processing, feature extraction, classification and model generation. The Lab color space we used for obtaining better results and we employed Lab color space and auto-encoder architecture in the final model. This project is also useful to provide the color to the images coming from satellites. Image colourization is applicable in many areas such as colourization of old black and white photos, old movies and scientific images.

Keywords—Colorization, Convolution neural network, Gray scale, Lab Colour space, Chrominance;

CONTENTS

	LIST OF FIGURES	V
	LIST OF TABLES	VI
CHAPTER 1	INTRODUCTION	
	1.0 Introduction	2
	1.1 Project Objective	4
	1.2 Project Outline	5
CHAPTER 2	Images and color significance	
	2.1 Types Of Images	8
	2.1.1 Gray Scale Image	8
	2.1.2 Color Image	8
	2.1.3 Monochrome Image	9
	2.2 Applications	10
	2.2.1 Astronomy	10
	2.2.2 Archeology	11
	2.2.3 Electron Microscopy	12
	2.3 Colour Space	14
	2.3.1 Introduction	14
	2.3.2 RGB Colour Space	14
	2.3.3 LAB Colour Space	15
	2.3.4 Advantages	17
	2.3.5 Disadvantages	17
	2.3.6 Applications	18
CHAPTER 3	Methodology	
	3.1 Introduction	20
	3.2 Implementation steps	20
	3.2.1 Image Pre-Processing	20
	3.2.2 Gray Scale Image	21
	3.2.3 Model Training	21
	3.2.4 Evaluation	23
	3.3 Model Architecture	23
	3.3.1 Introduction	23
	3.3.2 Applications	25

CHAPTER 4	Model Training			
	4.1	Keras and Tensorflow		28
		4.1.1	Introduction	28
		4.1.2	Advantages	29
		4.1.3	Disadvantages	31
		4.1.4	Applications	33
	4.2	CNN Overview & Applications		33
		4.2.1	Introduction	33
		4.2.2	Architecture	34
		4.2.3	Advantages	36
		4.2.4	Disadvantages	37
		4.2.5	Applications	38
	4.3	Python		38
		4.3.1	Introduction	38
		4.3.2	Advantages	40
		4.3.3	Disadvantages	40
		4.3.4	Tools	41
CHAPTER 5	Results			44
	Conclusion			56
	Future work			57
	References			58

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
FIG 2.1	Gray Scale Image	8
FIG 2.2	Colour Image	9
FIG 2.3	Monochrome Image	10
FIG 2.4	Astronomy	11
FIG 2.5	Archeology	12
FIG 2.6	Electron Microscopy	13
FIG 2.7	RGB Colour Space	15
FIG 2.8	LAB Colour Space	16
FIG 3.1	CNN Architecture	23
FIG 4.1	Feedforward Neural Network With 2 Hidden Layers	35
FIG 4.2	Example of CNN Architecture	35

LIST OF TABLES

TABLE NO	TITLE	PAGE NO
TABLE 4.1	Keras VS Tensorflow	29
TABLE 4.2	Deep learning VS neural Networks	36
TABLE 4.3	Python VS Matlab	39

CHAPTER 1

INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1 Introduction:-

The topic of coloring black and white images has gained a lot of interest in image processing research, as it has a significant impact on image analysis and information retrieval. Color images contain more information than grayscale images, making them more useful in extracting information. Coloring historical images, improving surveillance footage, and enhancing microscopic images are some common applications of image colorization.

Colorizing a grayscale image is a complex task that requires prior knowledge about the image content and manual adjustments to achieve a high-quality result. There are multiple ways to assign colors to the pixels in an image, making it challenging to find a distinctive solution.

Autoencoders, a class of convolutional neural networks, are used to automate the image colorization process and generate results that look natural to the human eye. Proper selection of color spaces can improve the training and performance of the neural network. The CIE Lab* color space is employed for color conversion, where the model predicts chroma values given the Luminance value. InceptionResNetV2, a powerful network with high accuracy, is used as a global feature extractor to better understand the semantic content of the image during the colorization process. The Mean Squared Error (MSE) is used as the loss objective function to update the model parameters. This research proposes a deep neural network that can colorize cultural, heritage, and historical images of ancient Nepal in a few seconds without user intervention

Colorization of black and white images is the process of adding color to a grayscale or monochromatic image. There are several ways to colorize black and white images, including manual colorization using photo editing software, and automatic colorization using machine learning algorithms.

Manual colorization involves adding color to each pixel of the image manually using photo editing software. This method requires a lot of time and effort, but it can produce very accurate and detailed results.

Automatic colorization, on the other hand, uses machine learning algorithms to analyze and understand the content of the image, and then add color to the grayscale image automatically. This method is faster and less labor-intensive than manual colorization, but the results may not always be as accurate or detailed.

There are several machine learning algorithms used for automatic colorization, including deep neural networks. These algorithms learn from large datasets of colored images and grayscale images, and then use this knowledge to add color to new grayscale images.

Overall, colorization of black and white images can be a fun and creative way to bring old photographs and historical images to life Colorization is the process of adding colors to black and white or grayscale images. One approach to colorization is to use the LAB color space, which separates the luminance (brightness) component (L) from the chromaticity (color) components (a and b). In LAB space, the L component is the grayscale image, while the a and b components represent the color information.

Here are the steps for image colorization with LAB color spaces:

1.Convert RGB to LAB

2.Separate the L, a, and b components.

3.Apply colorization to the a and b components. This can be done using various techniques, such as nearest-neighbor interpolation, color transfer, or machine learning methods like convolutional neural networks (CNNs).

4.Combine the L component with the colorized a and b components to obtain the final colorized image.

5.Convert the colorized image back to RGB color space.

Note that colorization is a challenging task since it requires inferring the missing color information from the grayscale input. As a result, the colorization results may not always be accurate, and human intervention may be required to refine the colorization output.

There are several methods of image colorization, ranging from manual techniques to automated algorithms. Here are a few common methods:

Manual Colorization: This involves manually coloring the image using image editing software such as Photoshop or GIMP. This technique requires artistic skills and is time-consuming, but it provides a high degree of control over the final result.

Color Transfer: This technique involves transferring the color palette from a source image to a target image using statistical analysis. This method works best when the source and target images have similar content and lighting conditions.

Deep Learning-Based Methods: These methods use convolutional neural networks (CNNs) to learn the colorization process. These models are trained on a large dataset of grayscale and corresponding color images, and then use this knowledge to colorize new grayscale images.

Hybrid Methods: These methods combine manual and automated techniques to achieve the desired result. For example, an artist may start with a manual colorization and then use a deep learning model to refine the result.

Interactive Methods: These methods allow users to interactively colorize an image by providing hints or strokes on the grayscale image, which are then used by the algorithm to infer the color. This method is particularly useful when the user has some knowledge of the color distribution in the image.

Image colorization is used for various purposes, such as:

Restoration of Old Photographs: Image colorization can be used to restore old black and white photographs by adding color to them. This helps to preserve the historical significance of the image and make it more visually appealing.

Visual Effects: Image colorization is also used in the film and entertainment industry to add color to black and white films or to enhance the color of digital images.

Medical Imaging: Image colorization can be used in medical imaging to highlight specific structures or areas of interest in medical images. For example, colorization can be used to differentiate between different types of tissue in MRI or CT scans.

Augmented Reality: Image colorization can be used in augmented reality applications to add color to real-world objects or images captured by the camera of a mobile device.

Educational Purposes: Image colorization can be used for educational purposes, such as colorizing diagrams, charts, and maps, to make them more visually appealing and easier to understand.

Overall, image colorization can be a useful tool for enhancing the visual appeal of images, preserving historical photographs, and providing insights in various fields

1.2 Project Objective:-

Bringing color to electron microscope images and gray scale images for getting the appropriate color images is a tricky problem. It could possibly be said that color doesn't exist at that scale, because the things imaged by an electron microscope are smaller than the wavelength of visible light. But that hasn't stopped scientists from trying, or at least developing techniques to approximate it.

The latest, described in an article of cell by scientists from the University of California, San Diego, attaches artificial color to biological structures, which could help us better understand the structures and functions within cells. They're the first to use this method on organic material, matching up to three colors and making, in one example, a Golgi region appear green and a plasma membrane red.

The normal microscopic devices are producing the grayscale images as output. This gray scale image will contain only 2 levels of color one is black and another one is white .The range of these levels from 0-255. In digital images, grayscale means that the value of each pixel represents only the intensity information of the light. Such images typically display only the darkest black to the brightest white. In other words, the image contains only black, white, and gray colors, in which gray has multiple levels.

Nowadays some of the electronic microscopes are used to produces the colour images as output with some magnifile level that is 250 magnifiles levels only.so, to move forward the magnifiles level the getting colour will be changed so, to decrease this disadvantage or issue convolutional auto encoders are used to produces the colour information which will gives the more information compared to the previous identified problems.

1.3 Project Outline:-

To produce the colour images from gray scale images we used the following approach:-

In digital photography computer-generated imagery and colorimetry a grayscale image is one in which value of each pixel is a single sample representing only an amount of light; that is, it carries only intensity information. Grayscale images, a kind of black-and-white or gray monochrome, are composed exclusively of shades of gray. The contrast ranges from black at the weakest intensity to white at the strongest.

After that we perform rescaling and normalization operations on the gray scale image .The rescaleing operation means each input gray scale image having its own size so that to produce our target output we are performs the rescaling operation and normalization is the process of dividing image with target size of 255 this is used to maintain each pixel value of an image in the specified range this range will provides the enhancement.

We later perform the RGB to LAB conversion of image and we represent each channel with their pixel values that pixel values are having some weight and that weights are provided as input to the convolutional auto encoders.

Auto encoders are neural networks where the architecture itself is designed such that the target output is the input. Down sampling takes place at the input image and upsampling takes place at the reconstructed image and after that we perform image pre and post processing.for image preprocessing we have to move the image from one layer to another that means output of one layer is fed to input to the other layer this can be achieved by using keras algorithm with importing the sequential libraries.initially a series of CNN and down saplings that are used to reduces the dimensional representation of image data and then use the CNN enough samples to regenerate the colour images after getting the colour image we perform evaluation operation i.e, compare and observing the accuracy and mean error at the output.

For getting colour information of the gray scale images the render factor will perform the major role in this process.The default value of 35 has been carefully chosen and should work ok for most scenarios. This determines the resolution at which the color portion of the image is rendered. Lower resolution will render faster, and colors also tend to look more vibrant. Older and lower quality images in particular will generally benefit by lowering the render factor. Higher render factors are often better for higher quality images, but the colors may get slightly washed out.

CHAPTER 2

IMAGES AND COLOR SIGNIFICANCE

CHAPTER 2

IMAGES AND COLOR SIGNIFICANCE

2.1 TYPES OF IMAGES:-

2.1.1 Gray Scale Images:-

A grayscale image is an image in which each pixel is represented by a single value that represents the brightness or intensity of that pixel. The term "grayscale" refers to the fact that the image contains shades of gray, as opposed to color. In a grayscale image, the value of each pixel typically ranges from 0 (black) to 255 (white), with intermediate values representing shades of gray. Grayscale images are often used in applications where color is not necessary or would be distracting, such as medical imaging, scientific visualization, and some forms of digital art. They are also useful in situations where color information is not available, such as when working with older or low-resolution images.

To convert a color image to grayscale, one common method is to take the average of the red, green, and blue values for each pixel. This results in a single value that represents the overall brightness of the pixel. Other methods for grayscale conversion exist as well, such as using the luminance values of the colors.

Fig 2.1: Gray Scale Image

2.1.2 Colour Images:-

A color image is an image that contains information about the colors of the objects or scenes it represents. In a digital color image, each pixel is represented by a combination of red, green, and blue (RGB) values that determine the intensity of each color channel. Other color models, such as the cyan, magenta, yellow, and key (CMYK) model used in printing, may also be used for color images.

Color images are used in a wide range of applications, including photography, art, design, and scientific visualization. They provide a more realistic representation of the world than grayscale images, and can convey important information about the properties of objects and scenes, such as their hue, saturation, and brightness.

In addition to RGB and CMYK, there are many other color models used for color image representation, such as the hue, saturation, and value (HSV) model and the LAB color space. Each color model has its own advantages and disadvantages and is better suited to certain applications than others.

Color images can be captured using digital cameras or scanners, and can also be created using digital image editing software. They can be displayed on computer screens, printed on paper or other materials, or projected onto screens or walls.

Fig 2.2: Colour Image

2.1.3 Monochrome Image:-

A monochrome image is an image that consists of shades of a single color, typically black or white. Monochrome images are often used in applications where color is not necessary or would be distracting, such as in document scanning or black-and-white photography.

There are different types of monochrome images, including binary images, grayscale images, and sepia images.

Binary images are black-and-white images in which each pixel is either black or white, with no shades of gray in between. They are commonly used for image processing and analysis, such as in optical character recognition (OCR) and computer vision.

Grayscale images are images that contain shades of gray between black and white. Each pixel in a grayscale image is represented by a single value that represents the intensity or brightness of that pixel. Grayscale images are often used in scientific imaging, medical imaging, and digital art.

Sepia images are monochrome images that have a warm, brownish tone. They are often used to give photos a vintage or antique look.

Monochrome images can be created from color images through various methods such as desaturation or by applying a monochrome filter. They are also commonly used as an intermediate step in image processing and analysis, such as in edge detection or image segmentation.

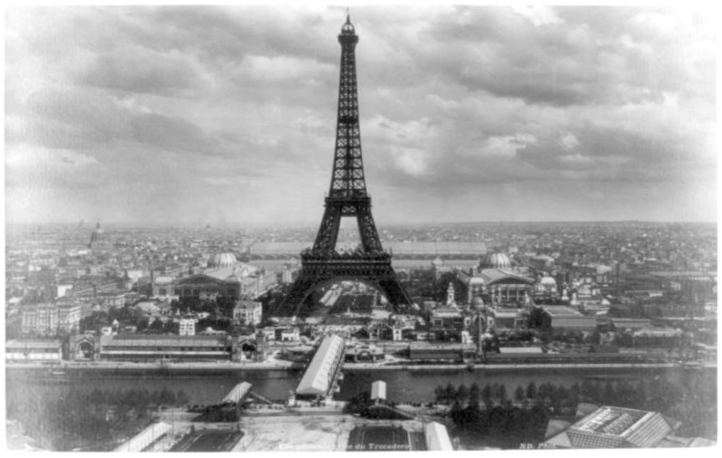


Fig 2.3:Monochrome Image

2.2 APPLICATIONS

2.2.1 Astronomy :-

Astronomy is the study of elysian objects, similar as stars, globes, worlds, and other marvels in the observable macrocosm. It's a natural wisdom that involves the observation and analysis of these objects, their stir, and their parcels, as well as the physical laws that govern their geste Astronomy has a long history, dating back to ancient societies similar as the Babylonians and the Greeks. moment, astronomers use a wide range of tools and ways to study the macrocosm, including telescopes, spacecraft, and computer simulations. Some of the major areas of study in astronomy include Astrophysics the study of the physical parcels and geste of elysian objects, similar as stars, worlds, and black holes. Planetary wisdom the study of globes and other objects in our solar system, including their conformation, composition, and elaboration.

Cosmology the study of the origin, elaboration, and structure of the macrocosm as a whole, including the Big Bang proposition and the hunt for dark matter and dark energy. Astronomy has numerous practical operations, including navigation, timekeeping, and satellite dispatches. It also has important counteraccusations for our understanding of the macrocosm and our place within it.

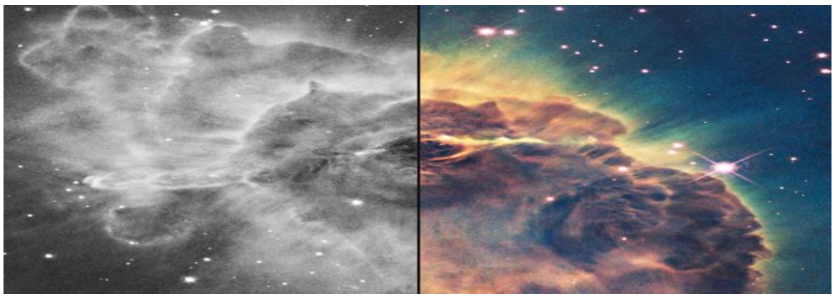


Fig 2.4: Astronomy

2.2.2 Archeology:-

Archaeology is the study of mortal history and prehistory through the excavation, analysis, and interpretation of material remains. It's a multidisciplinary field that combines rudiments of anthropology, history, art history, geology, and other lores to reconstruct and understand the history. Archaeologists study a wide range of vestiges and features, including armature, crockery, tools, bones, and other objects left before by once societies. They use a variety of ways to shovel and dissect these accoutrements, including remote seeing, surveying, and laboratory analysis. Some of the major areas of study in archaeology include neolithic archaeology the study of the societies and societies that was before written history, grounded on the material remains they left before. Classical archaeology the study of the societies and societies of ancient Greece and Rome, grounded on their material remains and erudite sources. literal archaeology the study of more recent societies and societies that left behind written records, similar as social agreements or artificial spots.

Archaeology has important counteraccusations for our understanding of mortal history and the development of mortal culture. It can help us to understand how different societies lived, worked, and interacted with each other, and can exfoliate light on the origins and elaboration of colorful mortal traditions and practices. Archaeological exploration can also give perceptivity into contemporary issues, similar as artistic heritage preservation and the impacts of mortal exertion on the terrain.

Fig 2.5: Archeology

2.2.3 Electron Microscopy:-

Electron microscopy is a technique that uses a beam of electrons to create high-resolution images of specimens at a very small scale, typically in the range of nanometers to micrometers. This makes it a valuable tool for a wide range of scientific fields, including biology, materials science, and nanotechnology.

There are several different types of electron microscopy techniques, including:

Scanning Electron Microscopy (SEM): In SEM, a focused beam of electrons is scanned across the surface of a sample, and the resulting backscattered or secondary electrons are detected to create a detailed image of the sample surface.

Transmission Electron Microscopy (TEM): In TEM, a focused beam of electrons is transmitted through a thin sample, and the resulting electrons that pass through the sample are detected to create an image of the internal structure of the sample.

Scanning Transmission Electron Microscopy (STEM): In STEM, a focused beam of electrons is transmitted through a thin sample, and the resulting electrons that pass through the sample are detected to create an image of the internal structure of the sample, similar to TEM. However, in STEM, the electron beam can also be scanned across the sample surface to create a more detailed image.

Electron microscopy has several advantages over other microscopy techniques, including its high resolution and ability to image samples at a very small scale. However, it also has some limitations, such as the requirement for vacuum conditions and the potential for sample damage due to the high-energy electron beam. Nonetheless, electron microscopy is an important tool for scientific research and has numerous applications in a variety of fields.

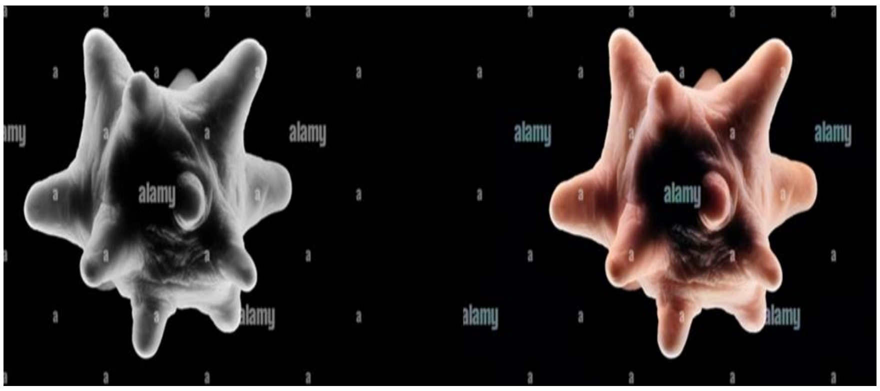


Fig 2.6: Electron Microscopy

2.3 COLOUR SPACE

2.3.1 Introduction:-

A color space, also known as a color model or color system, is a way of representing colors in a mathematical and numerical form. Color spaces are used in digital imaging, video, and other applications where color accuracy and consistency are important.

There are many different color spaces, each with its own advantages and disadvantages. Some of the most common color spaces include:

RGB: the most common color space used in digital imaging and display, which represents colors as a combination of red, green, and blue values.

CMYK: a color space used in printing, which represents colors as a combination of cyan, magenta, yellow, and black values.

HSB/HSL: color spaces that represent colors as combinations of hue, saturation, and brightness (HSB) or hue, saturation, and lightness (HSL) values.

LAB: a color space that separates color information into lightness (L), green-red (a), and blue-yellow (b) components, making it useful for color correction and image analysis.

Different color spaces have different gamuts, or ranges of colors that they can represent. Some color spaces, such as RGB, have a wider gamut than others, which can lead to more vibrant and accurate color representation. However, the choice of color space depends on the specific application and the requirements for color accuracy and consistency.

2.3.2 RGB Color Space:-

The RGB color space is a widely used color model that represents colors as a combination of red, green, and blue values. It is the primary color space used in digital imaging, including computer monitors, digital cameras, and other display technologies.

In the RGB color space, each color is represented as a combination of red, green, and blue values, typically ranging from 0 to 255. For example, pure red would be represented as (255,0,0), pure green as (0,255,0), and pure blue as (0,0,255). Other colors are represented as a combination of these values, such as yellow as (255,255,0) and purple as (128,0,128).

The RGB color space is an additive color model, meaning that the colors are created by adding different amounts of light together. When all three colors are added together at full intensity (255,255,255), it creates pure white, while adding no colors (0,0,0) creates pure black.

One of the advantages of the RGB color space is its wide gamut, or range of colors that it can represent. This makes it well-suited for digital imaging and display technologies, where accurate and vibrant color representation is important.

However, because the RGB color space is device-dependent, meaning that the color representation can vary between different devices and displays, it is important to use color management tools and techniques to ensure color accuracy and consistency.

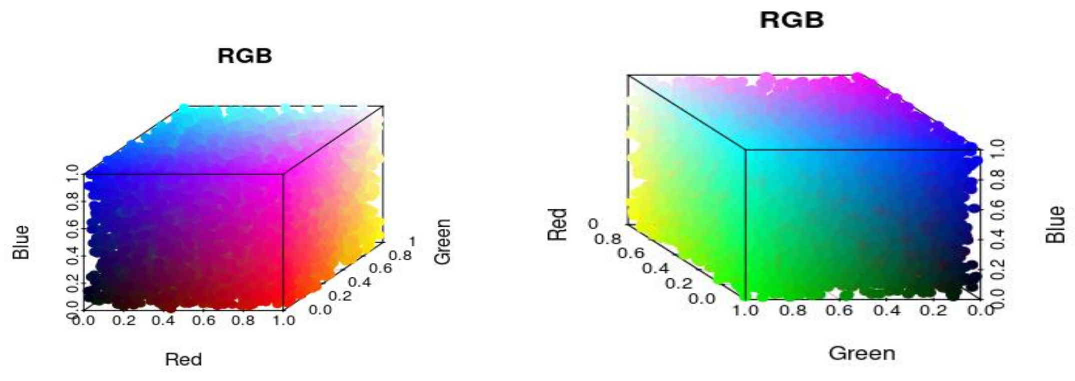


Fig 2.7: RGB Color Space

2.3.3 LAB Color Space:-

The LAB color space is a device-independent color model that separates color information into three components: lightness (L), green-red (a), and blue-yellow (b). This makes it well-suited for color correction, image analysis, and other applications where precise color information is important.

In the LAB color space, lightness (L) ranges from 0 (pure black) to 100 (pure white), while the green-red (a) and blue-yellow (b) axes can range from -128 to 127. Positive values along the a-axis represent green tones, while negative values represent red tones. Similarly, positive values along the b-axis represent yellow tones, while negative values represent blue tones.

One of the advantages of the LAB color space is that it has a wide gamut, or range of colors that it can represent, which makes it well-suited for color-critical applications such as printing, where accurate and consistent color reproduction is important.

Another advantage of the LAB color space is that it is device-independent, meaning that it is not tied to any specific device or display technology. This allows for consistent color representation across different devices and platforms.

However, because the LAB color space is not as intuitive as other color spaces such as RGB, it may require some training and experience to use effectively. Additionally, some image editing software may not support the LAB color space, which can limit its use in certain applications.

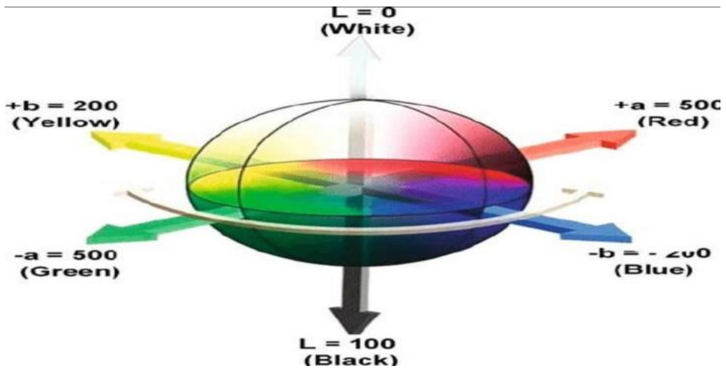


Fig (A)

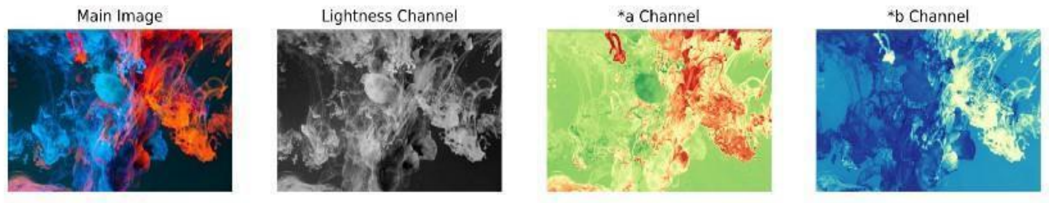


Fig (B)

Fig 2.8((a)(b)):LAB Color Space

2.3.4 Advantages:-

Color spaces have several advantages in various applications, including:

Accurate color representation: Color spaces allow for accurate representation of colors, which is important in many industries, such as printing, graphic design, and photography.

Wide gamut: Different color spaces have different gamuts, or ranges of colors that they can represent. Some color spaces, such as RGB, have a wider gamut than others, which can lead to more vibrant and accurate color representation.

Consistency: Color spaces ensure that colors are consistent across different devices and platforms, which is important in applications such as branding and marketing.

Color correction: Color spaces such as LAB are useful for color correction and image analysis, allowing for precise adjustments of color values.

Compatibility: Color spaces are widely used and supported by a range of software and hardware, making it easy to work with them in various applications.

Overall, color spaces provide a standardized way of representing colors that is accurate, consistent, and widely supported, making them an important tool in many industries and applications.

2.3.5 Disadvantages:-

While color spaces have many advantages, there are also some disadvantages to consider, such as:

Complex mathematics: Some color spaces, such as LAB, are based on complex mathematical formulas that can be difficult to understand and work with.

Device dependence: While some color spaces, such as LAB, are device-independent, others, such as RGB, are device-dependent, meaning that the color representation can vary between different devices and displays. This can lead to inconsistencies in color representation.

Limited gamut: Some color spaces have a limited gamut, or range of colors that they can represent, which can result in less accurate color representation.

Compatibility issues: Some software and hardware may not support certain color spaces, which can limit their use in certain applications.

Cost: Some color spaces, such as those used in professional printing, can be expensive to implement and maintain.

Overall, the advantages of color spaces often outweigh the disadvantages, but it is important to consider these factors when selecting a color space for a particular application.

2.3.6 Application:-

Color spaces have many practical applications in various industries and fields, including:

Graphic design: Color spaces such as RGB and CMYK are commonly used in graphic design to ensure accurate and consistent color representation in print and digital media.

Photography: Color spaces such as Adobe RGB and ProPhoto RGB are commonly used in photography to capture a wide range of colors and ensure accurate color representation in post-processing.

Printing: Color spaces such as CMYK and Pantone are used in printing to ensure accurate and consistent color representation in printed materials. **Video production**: Color spaces such as Rec. 709 and DCI-P3 are commonly used in video production to ensure accurate color representation in film and television. **Medical imaging**: Color spaces such as DICOM and CIELAB are used in medical imaging to ensure accurate and consistent color representation in medical images and scans.

Computer graphics: Color spaces such as sRGB and Adobe RGB are commonly used in computer graphics to ensure accurate and consistent color representation in digital images and animations.

Overall, color spaces play an important role in ensuring accurate and consistent color representation in a wide range of applications, from graphic design and photography to printing and medical imaging.

CHAPTER 3

METHODOLOGY

CHAPTER 3
METHODOLOGY

3.1 Introduction:-

Autoencoders are a type of neural network that can be used for unsupervised learning, which means they can find patterns in data without being explicitly told what to look for. The basic idea of an autoencoder is to take an input, encode it into a smaller representation, and then decode it back into the original input. The goal is to train the network to minimize the difference between the original input and the output produced by the decoder.

Autoencoders can be used for a variety of tasks, such as data compression, denoising, and anomaly detection. They have also been used in fields such as image and speech recognition, natural language processing, and recommender systems.

There are different types of autoencoders, such as the basic autoencoder, convolutional autoencoder, and variational autoencoder. Each type has its own unique characteristics and is used for different types of data.

Overall, autoencoders are a powerful tool for unsupervised learning and have a wide range of applications in various fields.

3.2 implementation steps:-

3.2.1 Image Pre-Processing:-

Image Resizing:-

In order to be processed by a neural network, images must be resized to a uniform size. The degree of resizing required depends on the fixed size used, with larger fixed sizes requiring less resizing. By minimizing the amount of shrinking needed, the features and patterns within the image are less distorted.

Normalization:-

Data normalization is an important step that ensures that each input parameter (pixel, in this case) has a similar data distribution. This makes convergence faster while training the network. Data normalization is done by subtracting the mean from each pixel and then dividing the result by the standard deviation. The distribution of such data would resemble a Gaussian curve centered at zero. For image inputs we need the pixel numbers to be positive, so we might choose to scale the normalized data in the range [0,1] or [0, 255].

3.2.2 Gray Scale Image:-

Autoencoders are neural networks that are trained to reconstruct their input data, usually by compressing it into a lower-dimensional representation and then decoding it back into the original input format.

Grayscale images can be obtained in autoencoders by treating the image as a matrix of pixel values and using a single channel to represent the grayscale intensity. This means that each pixel value is represented by a single number, typically between 0 and 255, with 0 being black and 255 being white.

During training, the autoencoder is fed with a set of grayscale images as input and the goal is to reconstruct these images with as little loss of information as possible. The encoder part of the autoencoder learns to encode the input image into a lower-dimensional representation, while the decoder part learns to decode this representation back into the original image format.

To obtain grayscale images, the input images are usually preprocessed to convert them from RGB format to grayscale format. This can be done by taking the average of the three RGB channels or by using a weighted average based on the luminosity of each color channel.

Once the autoencoder is trained, it can be used to generate new grayscale images by encoding a lower-dimensional representation of the image and then decoding it back into the grayscale format. These generated images may not be exact replicas of the original images, but they should be similar enough to be recognizable.

3.2.3 Modal Training:-

Upsampling:-

In auto encoders, upsampling is a process of increasing the spatial resolution of the input volume (i.e., the width and height dimensions), while reducing the depth dimension. It is the opposite of downsampling or pooling, which reduces the spatial resolution while increasing the depth dimension.

Upsampling is often used in CNNs for tasks such as image segmentation, where the output needs to be a pixel-wise classification of the input image. One common way to perform upsampling in a CNN is to use a transposed convolution (also known as a deconvolution or fractionally-strided convolution).

A transposed convolution applies a set of learnable filters to the input, but instead of performing a normal convolution operation, it performs an "inverse" operation, where the output pixels are spread out over a larger area of the input feature map. This effectively increases the spatial resolution of the output feature map.

Another common way to perform upsampling is to use nearest-neighbor interpolation or bilinear interpolation, which are simpler and faster than transposed convolutions, but may not be as effective in some cases.

Overall, upsampling is an important technique in CNNs for increasing the spatial resolution of feature maps, which can lead to better performance in tasks such as image segmentation and object detection.

Downsampling:-

In auto encoders, downsampling or pooling is a technique used to reduce the spatial resolution of the input feature maps while increasing the depth dimension. This is done to reduce the computational complexity of the network and extract more abstract features from the input.

There are several types of pooling operations that can be used in CNNs, including max pooling, average pooling, and stochastic pooling. Max pooling is the most common type of pooling and involves dividing the input feature map into non-overlapping rectangular regions and taking the maximum value within each region. Average pooling works similarly but takes the average value within each region. Stochastic pooling randomly selects the maximum value within each region based on a probability distribution.

Pooling operations can be applied at various stages of the network, typically after one or more convolutional layers. The pooling operation reduces the size of the feature maps, which reduces the computational cost and memory requirements of the subsequent layers. Additionally, pooling can help the network to be more robust to small translations of the input and can prevent overfitting.

However, too much pooling can cause the network to lose too much spatial information, leading to a loss in accuracy. Therefore, the choice of pooling operation and the amount of pooling should be carefully tuned based on the specific task and dataset.

3.2.4 Evaluation:-

we evaluate a trained model and for evaluation, we feed the image which was not shown to the trained image before.

3.3 MODEL ARCHITECTURE:-

1. ENCODER
2. DECODER
3. FUSION

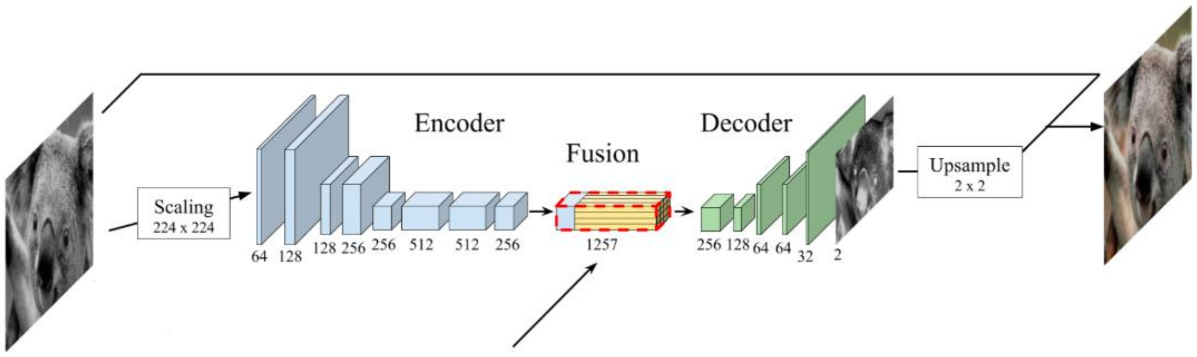


Fig 3.1: MODEL CNN ARCHITECTURE

3.3.1 Introduction:

In convolutional neural networks (CNNs), encoder, fusion, and decoder operations are often used in architectures that involve encoding the input data into a low-dimensional feature representation, fusing information across different modalities or spatial regions, and decoding the feature representation back into an output.

Encoder:The encoder operation in a CNN is used to extract features from the input data. This typically involves applying a series of convolutional and pooling layers to the input image or volume, resulting in a compressed feature representation that captures important visual or spatial patterns in the data.

Fusion:Fusion operations in CNNs are used to combine information from multiple sources or modalities. For example, in multi-modal image classification tasks, features from different image modalities (e.g., RGB and infrared) can be fused to improve performance. Similarly, in spatial fusion, features from different regions of an image can be combined to capture a more holistic view of the scene.

Decoder:The decoder operation in a CNN is used to convert the feature representation back into an output, such as a segmentation map or a class label. This typically involves applying a series of deconvolutional or upsampling layers to the compressed feature representation, followed by a final classification or regression layer to produce the output.

These operations can be combined in various ways to form different CNN architectures, such as U-Net, which uses an encoder-decoder architecture with skip connections to preserve spatial information, or DenseNet, which uses a dense fusion approach to encourage feature reuse across layers.

Number of layers:The number of layers in a Convolutional Neural Network (CNN) varies depending on the specific architecture and the task at hand. However, a typical CNN consists of multiple layers, such as convolutional layers, pooling layers, and fully connected layers.

A simple CNN architecture may consist of just a few convolutional and pooling layers followed by fully connected layers. In contrast, a more complex architecture such as ResNet can have hundreds of layers.

In general, deeper networks tend to have better performance for complex tasks such as image recognition, but they may also require more computational resources and longer training time.

3.3.2 Applications of autoencoders

Autoencoders are neural networks that can be used for a variety of applications in machine learning and artificial intelligence. Some of the common applications of autoencoders are:

Dimensionality Reduction: Autoencoders can be used to reduce the dimensionality of data by learning a lower-dimensional representation of it. This can be useful in applications where high-dimensional data needs to be processed more efficiently or where the data is noisy and needs to be simplified.

Image and Video Compression: Autoencoders can be used for image and video compression by encoding the image or video into a lower-dimensional representation and then decoding it back into the original format.

Anomaly Detection: Autoencoders can be used to detect anomalies in data by comparing the reconstruction error of the input data and the output data. If the error is high, it means that the input data is an anomaly.

Denoising: Autoencoders can be used to remove noise from data by learning a compressed representation of the data and then using it to reconstruct the data without the noise.

Generative Models: Autoencoders can be used as generative models to generate new data that is similar to the training data. This can be useful in applications such as image synthesis or text generation.

Recommendation Systems: Autoencoders can be used in recommendation systems by encoding user preferences and item features into a lower-dimensional representation and then using this representation to generate recommendations. Time Series Analysis: Autoencoders can be used to analyze time series data by learning a compressed representation of the data and then using it to forecast future values.

Missing Value Imputation:-

Denoising autoencoders can be used to impute the missing values in the dataset. The idea is to train an autoencoder network by randomly placing missing values in the input data and trying to reconstruct the original raw data by minimizing the reconstruction loss. Once the autoencoder weights are trained the records having missing values can be passed through the autoencoder network to reconstruct the input data, that too with imputed missing features.

Image Compression:-

Image compression is another application of an autoencoder network. The raw input image can be passed to the encoder network and obtained a compressed dimension of encoded data. The autoencoder network weights can be learned by reconstructing the image from the compressed encoding using a decoder network. Usually, autoencoders are not that good for data compression, rather basic compression algorithms work better.

CHAPTER 4

MODEL TRAINING

CHAPTER 4
MODEL TRAINING

4.1 KERAS And TensorFlow

4.1.1 Introduction:-

Keras is a Python-based open-source library for building neural networks that facilitates fast experimentation and smooth transition from research to production. Developed by François Chollet, a Google engineer, it has become popular due to its flexibility, user-friendly interface, and ease of use. With a high-level API, Keras allows users to construct complex models using only a few lines of code and offers a wide range of built-in layers, activations, and loss functions to create various neural network architectures. It can work with various deep learning frameworks such as TensorFlow, Theano, Microsoft Cognitive Toolkit, and PlaidML and supports both CPU and GPU acceleration, making it ideal for large-scale deep learning projects. Overall, Keras is a powerful and versatile tool that is widely utilized by data scientists, researchers, and developers worldwide for building and training deep learning models.

TensorFlow is an open-source software library that was created by the Google Brain team in 2015. It has since grown to become one of the most popular machine learning frameworks available. TensorFlow allows users to build and train machine learning models using various neural network architectures, including CNNs and RNNs. It offers built-in tools for data manipulation, visualization, and preprocessing to help users prepare their data for machine learning tasks. TensorFlow also supports distributed computing, enabling models to be trained across multiple machines and accelerators. The platform has an active community of users and developers who contribute to its library of tools and resources, including data-specific libraries and model interpretability and deployment tools. Overall, TensorFlow is a versatile and powerful framework suitable for various machine learning tasks, from basic classification and regression to more advanced natural language processing and computer vision applications.

Table 4.1: Keras VS Tensorflow

Keras	**TensorFlow**
Keras is a high-level API that is running on top of TensorFlow, CNTK, and Theano.	TensorFlow is a framework that offers both high and low-level APIs.
Keras is easy to use if you know the Python language.	You need to learn the syntax of using various TensorFlow functions.
Perfect for quick implementations.	Ideal for Deep learning research, and complex networks.
Uses another API debug tool such as TFDBG.	You can use Tensor board visualization tools for debugging.
It was started by François Chollet as a project and developed by a group of people.	It was developed by the Google Brain team.

4.1.2 Advantages:-

Keras:-

Keras is a popular high-level neural network API that is built on top of lower-level machine learning frameworks like TensorFlow and Theano. Some of the advantages of using Keras include:

Ease of Use: Keras provides a simple and intuitive interface for building neural networks. Its user-friendly design allows developers to easily define and train neural networks without having to worry about the underlying implementation details.

Portability: Keras is designed to be highly portable and can run on a variety of platforms and hardware configurations. This makes it easy to deploy models to different environments and to take advantage of specialized hardware like GPUs.

Flexibility: Keras is highly flexible and allows developers to build a wide variety of neural network architectures, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and deep belief networks (DBNs).

Modularity: Keras is designed to be modular, with a wide range of pre-built layers, loss functions, and optimizers that can be combined to create custom neural network architectures.

Large Community: Keras has a large and active community of developers and researchers who contribute to the framework and provide support to other users. This makes it easy to find resources and solutions to common problems.

Integration with TensorFlow: Keras is tightly integrated with TensorFlow, which allows developers to take advantage of TensorFlow's advanced features while still using Keras's user-friendly API.

Tensorflow:-

TensorFlow is a popular and powerful open-source library for building and deploying machine learning models. Some of the key advantages of using TensorFlow include:

Scalability: TensorFlow is designed to be highly scalable, making it well-suited for large-scale machine learning applications. It can efficiently train and run models on large datasets and distributed computing systems.

Flexibility: TensorFlow provides a flexible framework for building and training a wide range of machine learning models, including neural networks, decision trees, and support vector machines. This makes it a versatile tool for a wide range of applications.

High Performance: TensorFlow is optimized for speed and performance, making it capable of handling complex models and large datasets with ease. It can take advantage of specialized hardware like GPUs and TPUs to further improve performance.

Rich Ecosystem: TensorFlow has a large and active community of developers who contribute to the library and provide support to other users. This has led to the development of a rich ecosystem of tools, libraries, and resources that can be used to enhance TensorFlow and extend its functionality.

Easy to Use: TensorFlow provides a user-friendly interface that makes it easy to define, train, and deploy machine learning models. Its high-level APIs like Keras make it accessible to developers with varying levels of experience in machine learning.

Platform Agnostic: TensorFlow can run on a variety of platforms, including desktop computers, mobile devices, and cloud servers. This makes it easy to deploy models to different environments and to take advantage of specialized hardware when necessary.

Support for Multiple Languages: TensorFlow provides support for multiple programming languages, including Python, C++, and Java. This makes it accessible to a wide range of developers with different programming backgrounds.

4.1.3 Disadvantages:-

Keras:-

While Keras is a popular and highly versatile neural network API, it does have some limitations and disadvantages that developers should be aware of:

Limited Low-Level Control: While Keras provides a high-level interface that makes it easy to build neural networks quickly, it may not provide the level of control and customization needed for certain complex or specialized tasks. For example, low-level modifications to the optimizer may be difficult to implement in Keras.

Performance Limitations: Although Keras can be optimized for speed, it may not be as efficient as lower-level frameworks like TensorFlow or PyTorch. This can become a concern when dealing with very large datasets or complex models.

Limited Graph Customization: Keras has some limitations in terms of graph customization, which may be necessary for some specialized use cases. For example, it may be challenging to implement certain types of attention mechanisms or dynamic architectures in Keras.

Compatibility Issues: As Keras is built on top of lower-level frameworks like TensorFlow and Theano, there may be compatibility issues with different versions of these frameworks. This can lead to challenges in reproducing models across different environments.

Limited Support for Non-Neural Network Models: While Keras is primarily designed for building neural networks, it may not be the best choice for other types of machine learning models. For example, decision trees or random forests may be better implemented using other libraries or frameworks.

Tensorflow:-

While TensorFlow is a powerful and versatile machine learning library, it does have some limitations and disadvantages that developers should be aware of:

Steep Learning Curve: TensorFlow can have a steep learning curve for developers who are new to machine learning or unfamiliar with the library's API. It can take some time to become proficient in using the library effectively.

Complex Architecture: TensorFlow's architecture can be complex, with many different components and abstractions to learn. This can make it challenging to debug and optimize models, especially for developers who are new to the library.

Limited Support for Non-Neural Network Models: While TensorFlow is a powerful tool for building neural networks, it may not be the best choice for other types of machine learning models. For example, decision trees or random forests may be better implemented using other libraries or frameworks.

Compatibility Issues: Different versions of TensorFlow may not be compatible with each other, leading to challenges in reproducing models across different environments. Additionally, some third-party libraries may not be compatible with certain versions of TensorFlow.

Performance Limitations on Small Datasets: While TensorFlow is highly scalable and can handle large datasets with ease, it may not perform as well on smaller datasets. This can be a concern for developers who are working with limited data or resources.

Lack of Transparency: TensorFlow's complexity can make it challenging to understand what is happening inside a model. This lack of transparency can make it difficult to debug and optimize models, especially for developers who are new to the library.

4.1.4 Applications

Keras and TensorFlow are highly utilized in different fields for creating and implementing deep learning models. Some of the commonly used applications of Keras and TensorFlow are:

Image and object recognition: Keras and TensorFlow are widely used for tasks such as image classification, segmentation, and object detection, which involve recognizing objects in images, detecting faces, and tracking objects in videos.

Natural Language Processing (NLP): Keras and TensorFlow are also employed for constructing language models that can comprehend and generate human language. This encompasses tasks such as text classification, language translation, and sentiment analysis. Recommendation systems: Keras and TensorFlow can be leveraged to establish recommendation systems that suggest movies, products, or other items based on the users' preferences and behavior.

Speech recognition: Keras and TensorFlow are utilized in developing speech recognition models that can transcribe speech into text and vice versa.Autonomous vehicles: Keras and TensorFlow are applied in creating autonomous vehicles, including object recognition, localization, and designing optimal routes.

Medical diagnosis: Keras and TensorFlow can be utilized to develop models for identifying diseases and detecting irregularities in medical images.
In summary, Keras and TensorFlow are highly versatile and user-friendly, making them ideal for various industries' deep learning model creation and implementation.

4.2 CNN Overview & Applications

4.2.1 Introduction:-

Convolutional Neural Networks (CNNs) are a type of artificial neural network that are primarily used for analyzing visual imagery. They are designed to recognize patterns in images and classify them into different categories, such as identifying objects or detecting anomalies.

CNNs are made up of multiple layers that extract features from the input images. The first layer of a CNN is usually a convolutional layer that applies filters to the image to identify patterns and features. The output of the convolutional layer is then passed through a pooling layer, which reduces the dimensionality of the image and makes the subsequent layers faster to compute. The final layers of a CNN are typically fully connected layers that classify the features extracted from the image into different categories.

One of the significant advantages of CNNs is their ability to learn hierarchical representations of images. This means that the network can identify simple features, such as edges and corners, and combine them to recognize more complex structures, such as faces or objects.

CNNs have numerous applications, including image classification, object detection, facial recognition, and medical diagnosis. They have proven to be highly effective in these tasks and have been instrumental in the development of many advanced technologies.

Overall, CNNs are a powerful tool for analyzing visual data and have opened up many possibilities for machine learning applications.

4.2.2 Architecture:-

The architecture of a Convolutional Neural Network (CNN) consists of multiple layers that work together to extract and classify features from input images. The typical layers in a CNN architecture are:

Convolutional Layer: The first layer of a CNN applies a set of learnable filters to the input image to identify patterns and features in the image. Each filter produces a feature map, which is a transformed version of the input image that highlights specific features.

ReLU Layer: The Rectified Linear Unit (ReLU) layer applies an element-wise activation function to the feature maps produced by the convolutional layer. The ReLU function sets negative values to zero, and leaves positive values unchanged, which introduces non-linearity into the network.

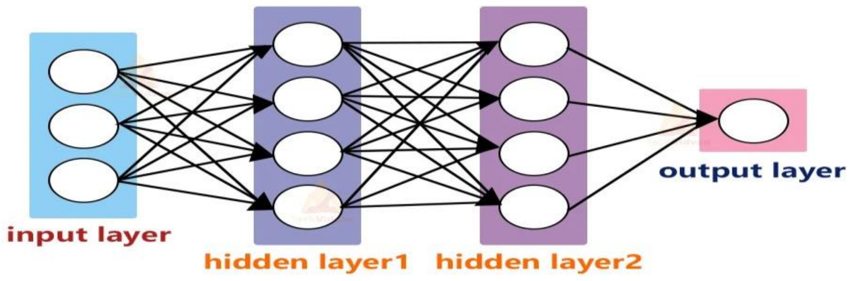


Fig 4.1: Feedforward Nerual Network With 2 Hidden Layers

Pooling Layer: The pooling layer reduces the dimensionality of the feature maps produced by the convolutional layer. The most common type of pooling is max pooling, which selects the maximum value from a set of values in each sub-region of the feature map.

Fully Connected Layer: The fully connected layer is a traditional neural network layer that takes the flattened output of the previous layers and performs classification or regression tasks. This layer uses the softmax activation function to predict the probability distribution of the output class.

Dropout Layer: The dropout layer is used to prevent overfitting in the network. It randomly drops out a fraction of the connections between neurons during training, forcing the network to learn more robust and generalized features.

The architecture of a CNN can vary depending on the specific task and dataset. However, most CNNs follow the general structure of these layers, with variations in the number of layers, filters, and parameters. By stacking multiple layers and combining them with non-linear activation functions, CNNs can learn hierarchical representations of features and classify images with high accuracy.

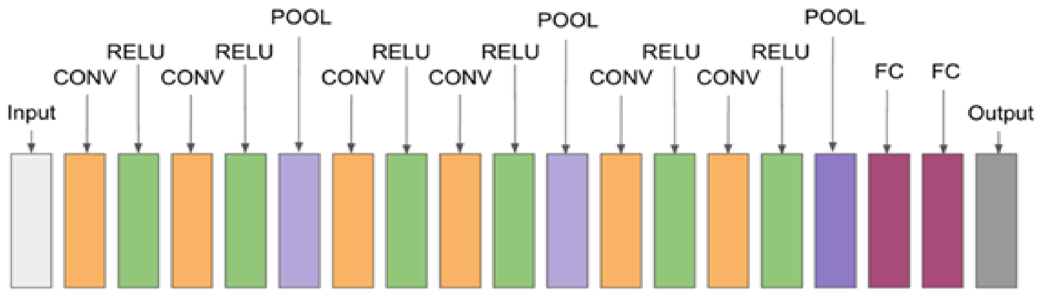


Fig 4.2: Example of a CNN Architecture

Table 4.2 : Deep Learning VS Neural Network

Deep Learning	Neural Network
Deep learning is an approach to AI and a technique that enables computer systems to improve with experience and data.	Neural Networks are computational models inspired by the idea of how the nervous system works.
It is a particular kind of machine learning method based on artificial neural networks.	It is a biologically inspired network of neurons configured to perform certain tasks.
Deep learning architecture is based on artificial neural networks.	Neural Network consists of three layers: an input layer, an output layer, and a hidden layer.
Deep learning architecture is one of the many applications of artificial neural networks.	Neural Networks make great tools for pattern recognition, clustering, prediction and analysis, control and optimization, machine translation, etc.

4.2.3 Advantages:-

Convolutional Neural Networks (CNNs) have several advantages over traditional neural networks when it comes to processing and classifying image data. Some of the key advantages of CNNs are:

Local connectivity: CNNs leverage the spatial relationships between pixels in an image by applying filters to local regions of the input image. This local connectivity allows the network to capture local patterns and features that are important for image classification.

Parameter sharing: In traditional neural networks, each weight in the network is learned independently for each neuron. In contrast, CNNs apply the same set of filters across the entire input image, which allows the network to learn and reuse patterns across different regions of the image. This reduces the number of parameters needed to train the network, making it more efficient.

Translation invariance: CNNs are able to recognize objects in images regardless of their position or orientation. This is because the convolution operation is translation invariant, meaning that it can detect the same pattern regardless of where it appears in the input image.

Hierarchical representations: CNNs use multiple layers of convolutional and pooling operations to learn hierarchical representations of features in an image. The lower layers capture simple features like edges and corners, while the higher layers capture more complex features like shapes and textures.

Robustness to noise: CNNs are able to filter out noise and other irrelevant features in an image, making them more robust to variations in lighting conditions, background clutter, and other types of image noise.

Overall, these advantages make CNNs highly effective for image classification tasks, and have led to their widespread adoption in computer vision applications.

4.2.4 Disadvantages:-

While Convolutional Neural Networks (CNNs) have several advantages, there are also some limitations and disadvantages that should be considered. Some of the key disadvantages of CNNs are:

High computational requirements: CNNs can be computationally expensive to train and require a large amount of processing power and memory. This can make it difficult to train large models on low-end hardware or in resource-constrained environments.

Large amounts of data required: CNNs require large amounts of labeled training data in order to learn effective representations of features in images. This can be a limiting factor in applications where labeled data is scarce or expensive to acquire.

Difficulty with fine-grained classification: While CNNs are effective at recognizing broad categories of objects, they can struggle with fine-grained classification tasks where subtle differences between similar objects need to be detected.

Limited interpretability: The hierarchical nature of CNNs can make it difficult to interpret the learned representations and understand how the network is making its predictions. This can be a limitation in applications where interpretability is important, such as in medical diagnosis or legal contexts.

Vulnerability to adversarial attacks: CNNs are susceptible to adversarial attacks, where small perturbations to an input image can cause the network to misclassify the image. This can be a concern in security-critical applications where the integrity of the model is important.

Overall, while CNNs have many advantages and are highly effective for many computer vision applications, they are not a one-size-fits-all solution and should be carefully evaluated and optimized for each specific use case.

4.2.5 Applications:-

Convolutional Neural Networks (CNNs) find applications in diverse fields. They are often used in image and object recognition tasks for identifying objects in images, detecting and tracking objects in videos, and recognizing faces. CNNs also find their usage in Natural Language Processing (NLP) applications like sentiment analysis, text classification, and language translation. Moreover, CNNs are crucial in developing autonomous vehicles for object detection, localization, and path planning. In medical fields, CNNs can diagnose diseases and anomalies in medical images such as X-rays, MRIs, and CT scans. Robotics and video and audio analysis applications also employ CNNs for object recognition, localization, manipulation, speech recognition, synthesis, action recognition, and video summarization. With the ability to learn complex representations of data, CNNs have wide-ranging applications, and their state-of-the-art performance is widely acclaimed in industry and research.

4.3 Python

4.3.1 Introduction:-

Python is a protean high- position scripting language that's extensively used for colorful tasks similar as textbook processing, system administration, and internet- related conditioning. It stands out among analogous languages due to its small core language that's easy to learn, yet it allows the addition of modules to perform a vast range of functions. Python is also an object- acquainted language and is available on multiple platforms, with a Python practitioner indeed written entirely in Java. Python was created in the early 1990s by Guido van Rossum, with the thing of designing a language that would be easy for newcomers to learn while being important enough for advanced druggies. Python's syntax is terse and clean, and its perpetration of object- acquainted programming is thorough, making it an excellent first programming language without immolating advanced capabilities.

Although python is named after snakes, it was inspired by Guido van Rossum's favorite television show," Monty Python's Flying Circus." Python attestation and online coffers frequently have a light and humorous touch. Python has some distinct features that may be strange to programmers of other languages, similar as statements not demanding a special character to end and the use of indentation to indicate the presence of circles rather of delimiters.

It's pivotal to follow some rules while composing programs in Python, similar as harmonious indentation within a given depth of a circle and statements that aren't notched must begin in the first column. numerous Python programmers prefer to use an editor that automatically provides harmonious indentation.

Table 4.3: Python VS Matlab

Python	Matlab
It is a general-purpose programming language used to develop fully-fledged applications or other software tools.	It is a commercial programming language interactive environment for numerical computing and programming.
It used O-based indexing meaning the arrays are indexed from 0.	It uses 1-based indexing meaning the arrays are indexed from 1.
It determines the scope of a block based on indentation.	It uses statements are closures.
No interactive UI development platform.	Interactive UI development platform.
More expressive and readable than Matlab scripts.	More comprehensive numerical functionality.
Graphics relies on external packages.	Graphical capabilities are more convenient.

4.3.2 Advantages:-

Python has several advantages that make it a preferred language for many developers. One of the main benefits is its ease of use and learnability. With a syntax similar to the English language, Python is easy to learn and adapt to, particularly for beginners. Additionally, Python requires fewer lines of code compared to languages such as Java and C, resulting in increased productivity and faster execution of tasks. Another advantage of Python is its flexibility, allowing users to develop new types of applications and try new things without restrictions. Python also boasts an extensive library, including almost every function needed to perform various tasks, thanks to its supportive community and corporate sponsorship. Speaking of the community, Python has a mature community that supports developers at all levels, providing guides, tutorials, and documentation to help them understand and use the language more efficiently. As a result, Python continues to experience rapid growth in popularity compared to other programming languages.

4.3.3 Disadvantages:-

Although Python is a popular and widely used programming language, there are some potential drawbacks to consider. For example, because Python is an interpreted language, it may be slower than compiled languages like C or Java, especially when working with large datasets or building high-performance applications. Additionally, Python's Global Interpreter Lock (GIL) can limit the ability to scale multithreaded applications across multiple CPU cores, which can make it more challenging to take full advantage of modern hardware. Python's automatic memory management can also lead to slower performance and less predictable memory usage compared to languages that allow for manual memory management. Furthermore, Python may not be the best choice for mobile app development since it doesn't provide native support for mobile platforms like iOS or Android. However, many of these limitations can be addressed with the use of external libraries, frameworks, and tools, and Python's ease of use and large developer community make it a popular choice for many projects.

4.3.4 Tools:-

Anaconda Navigator:-

Anaconda Navigator is a graphical user interface (GUI) that comes with the Anaconda distribution of Python programming language. It provides an easy and intuitive way to manage and launch various data science and machine learning tools, environments, and packages from a single platform. It allows users to create and manage virtual environments, install and update packages, access Jupyter Notebook, and launch other popular data science tools like Spyder, RStudio, and VS Code. Anaconda Navigator simplifies the process of setting up and managing a Python data science environment, making it a popular choice among data scientists, machine learning engineers, and researchers .

Spyder:-

Spyder is a free and open-source Integrated Development Environment (IDE) that is specially designed for scientific programming in Python. It offers a user-friendly environment for interactive development, debugging, and testing of Python code. Spyder focuses on data science, numerical analysis, and scientific computing. Its features include a code editor with syntax highlighting, code completion, and an integrated debugging tool. Spyder also comes with an interactive console, which allows users to test code snippets and view the output in real-time. In addition, Spyder has built-in support for popular data science libraries like NumPy, SciPy, and Pandas. Overall, Spyder is a powerful IDE that can greatly benefit scientific programmers working with Python.

Visual Studio Code:-

Visual Studio Code (VS Code) is a popular, free, and open-source code editor created by Microsoft that is widely used by developers in various programming languages, including Python. It comes with a modern user interface and numerous functionalities that make it a versatile editor. The editor has a vast collection of extensions that can be utilized to customize its capabilities. VS Code has numerous functionalities that are useful for Python development, such as syntax highlighting, debugging, code completion, and Git integration. VS Code also supports Python-specific features such as code refactoring, formatting, and linting with the installation of the Python extension. Overall, VS Code is a customizable and powerful code editor that is ideal for Python development.

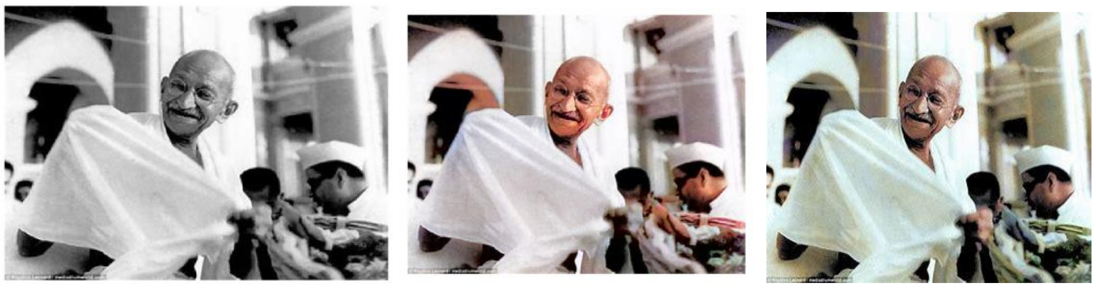
Google Colab:-

Google Colaboratory (Google Colab) is a cloud-based platform that provides users with a free Jupyter notebook environment. It allows users to write, run, and share Python code from their browser without requiring any setup or installation. Google Colab provides access to powerful computing resources, including CPUs, GPUs, and TPUs, allowing users to run large-scale machine learning and deep learning models. It also has built-in support for popular Python libraries such as TensorFlow, PyTorch, and OpenCV. Google Colab is an excellent tool for collaborative coding as users can share their notebooks with others and work on them simultaneously. Additionally, Google Colab integrates with Google Drive, allowing users to save and access their notebooks from anywhere. Overall, Google Colab is a convenient and powerful platform for Python development and data science.

CHAPTER 5

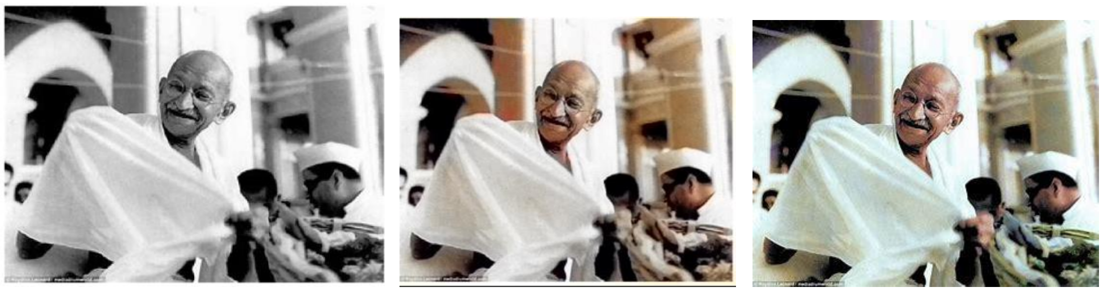
RESULTS

RESULTS



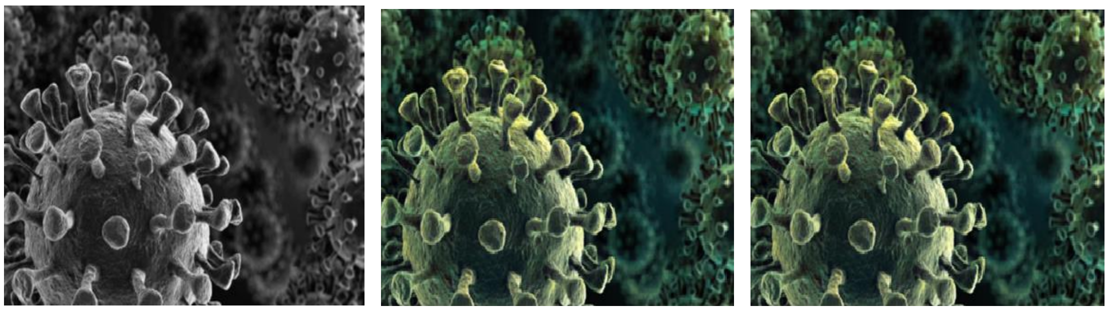
(A) (B) (C)

Fig5.1: (a)Grayscale image(input image) (b) Autoencoder output (c) Original image



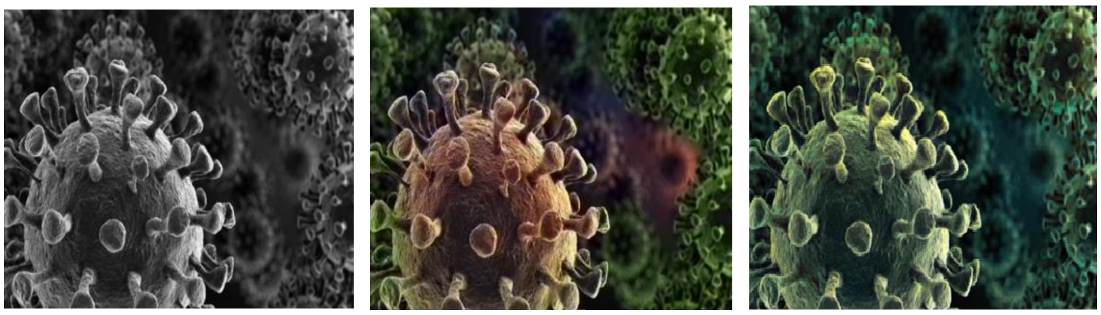
(A) (B) (C)

Fig5.2: (a)Grayscale image(input image) (b) Caffe model output (c) Original image



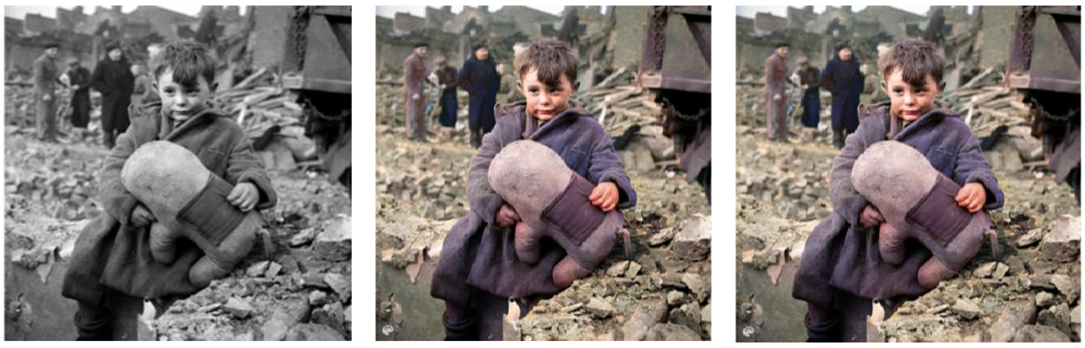
(A) (B) (C)

Fig5.3: (a)Grayscale image(input image) (b) Autoencoder output (c) Original image



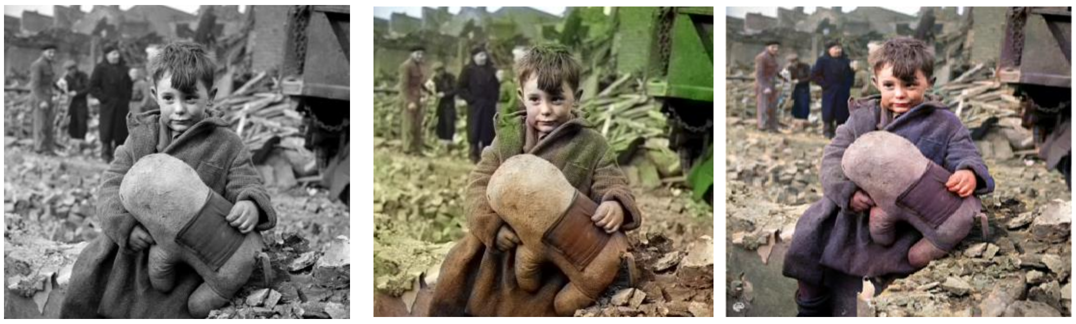
(A) (B) (C)

Fig5.4: (a)Grayscale image(input image) (b) Caffe model output (c) Original image



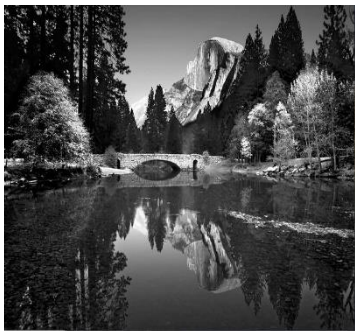
(A) (B) (C)

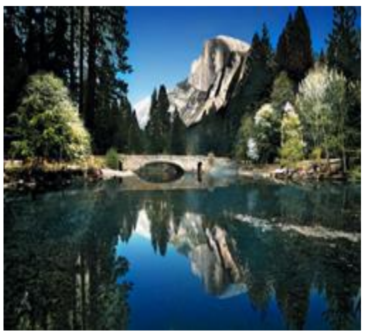
Fig5.5: (a)Grayscale image(input image) (b) Autoencoder output (c) Original image



(A) (B) (C)

Fig5.6: (a)Grayscale image(input image) (b) Caffe model output (c) Original image


(A)


(B)

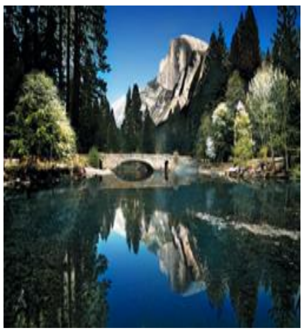
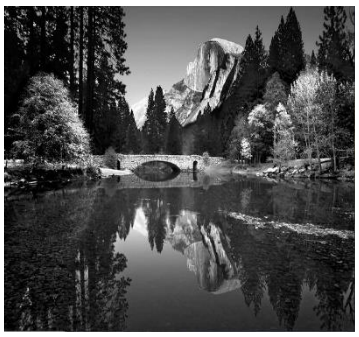
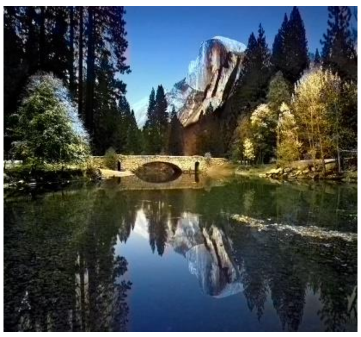

(C)

Fig5.7: (a)Grayscale image(input image) (b) Autoencoder output

(c) Original image


(A)


(B)

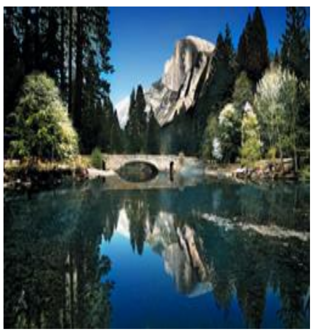

(C)

Fig5.8: (a)Grayscale image(input image) (b) Caffe model output

(c) Original image

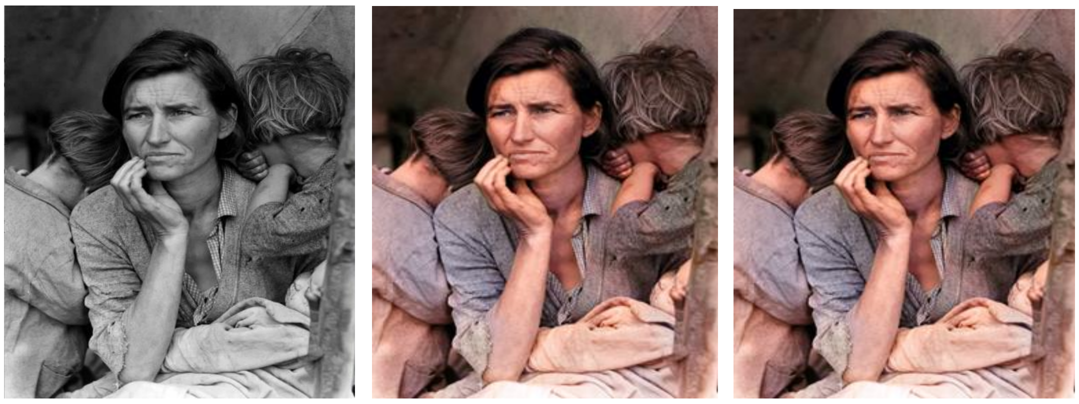


Fig5.9: (a)Grayscale image(input image) (b) Autoencoder output (c) Original image

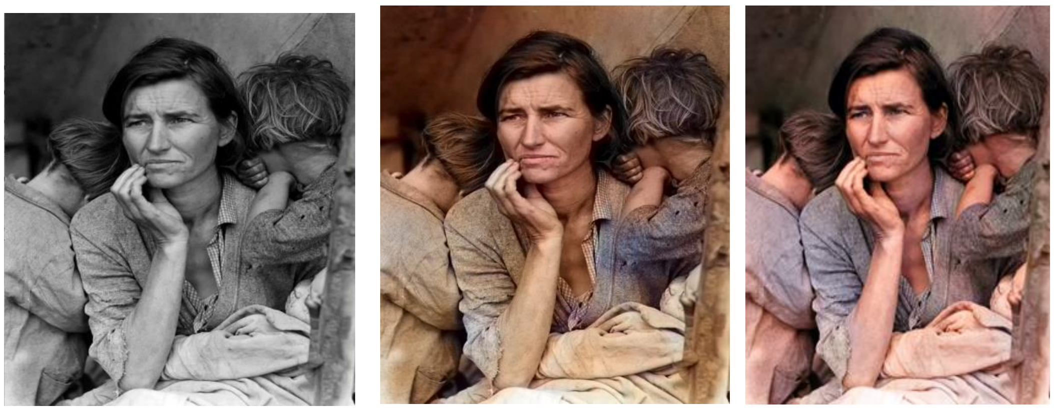
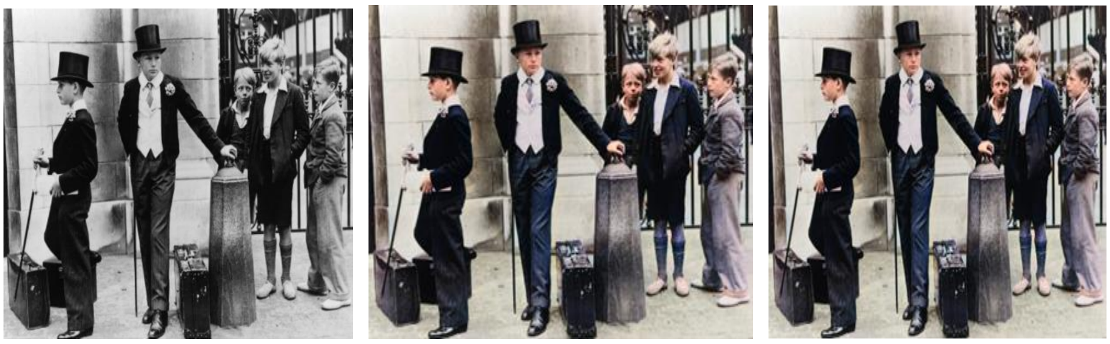
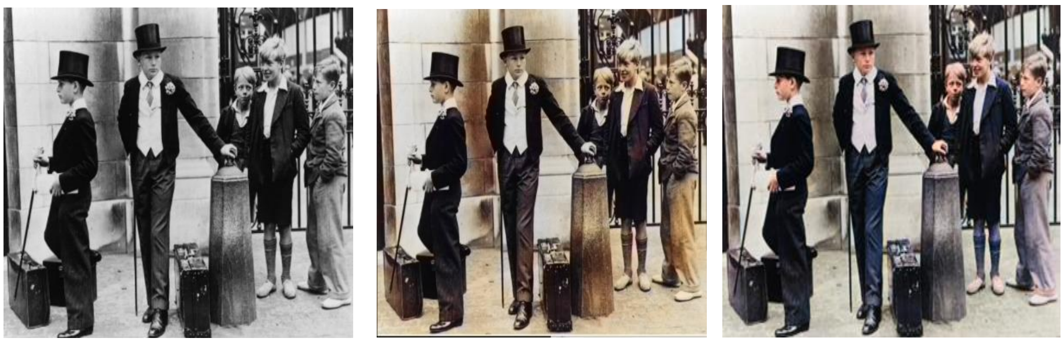


Fig5.10: (a)Grayscale image(input image) (b) Caffe model output (c) Original image



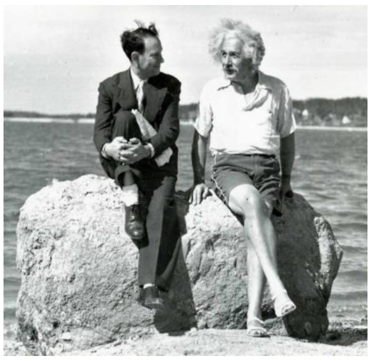
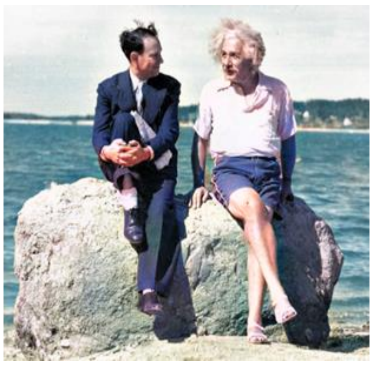
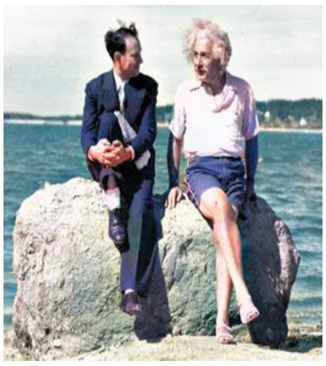
(A) (B) (C)

Fig5.11: (a)Grayscale image(input image) (b) Autoencoder output (c) Original image



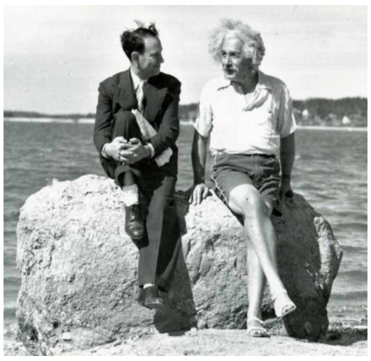
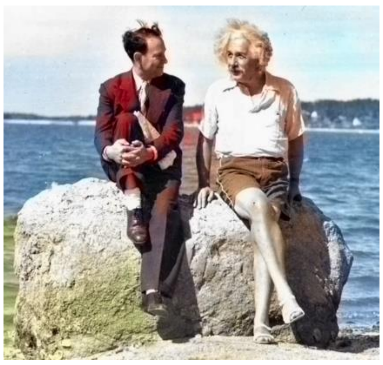
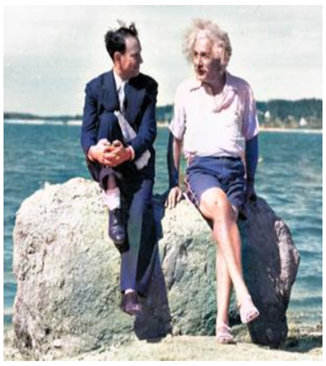
(A) (B) (C)

Fig5.12: (a)Grayscale image(input image) (b) Caffe model output (c) Original image

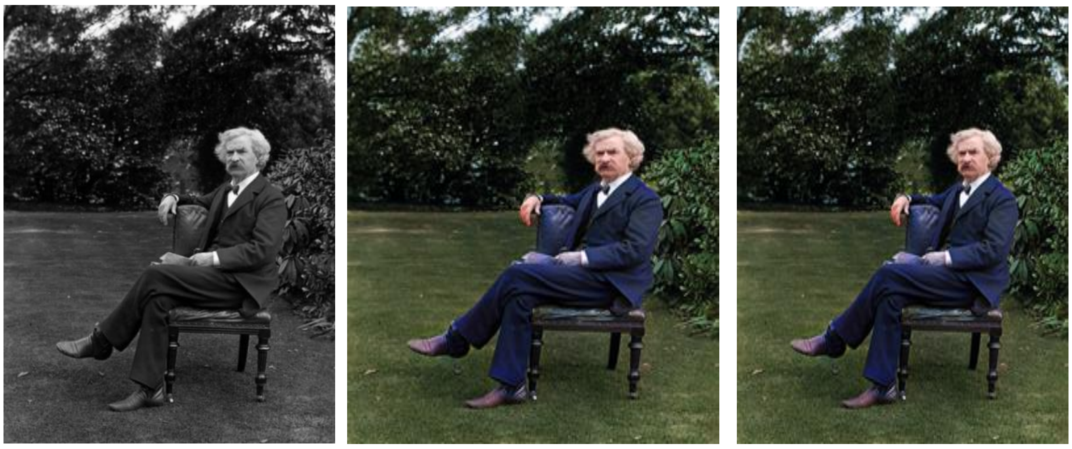
(A) (B) (C)

Fig5.13: (a)Grayscale image(input image) (b) Autoencoder output (c) Original image

(A) (B) (C)

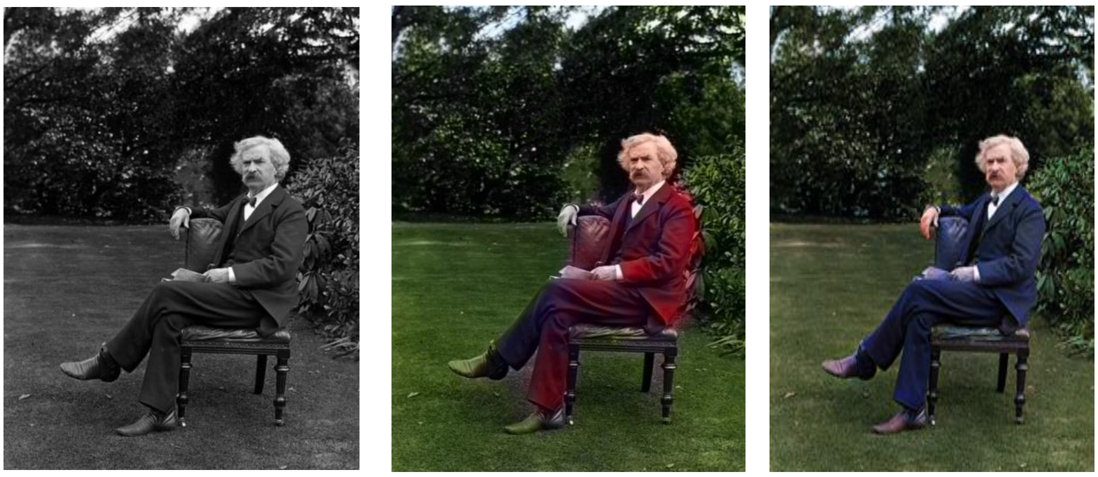
Fig5.14: (a)Grayscale image(input image) (b) Caffe model output (c) Original image



(A) (B) (C)

Fig5.15: (a)Grayscale image(input image) (b) Autoencoder output

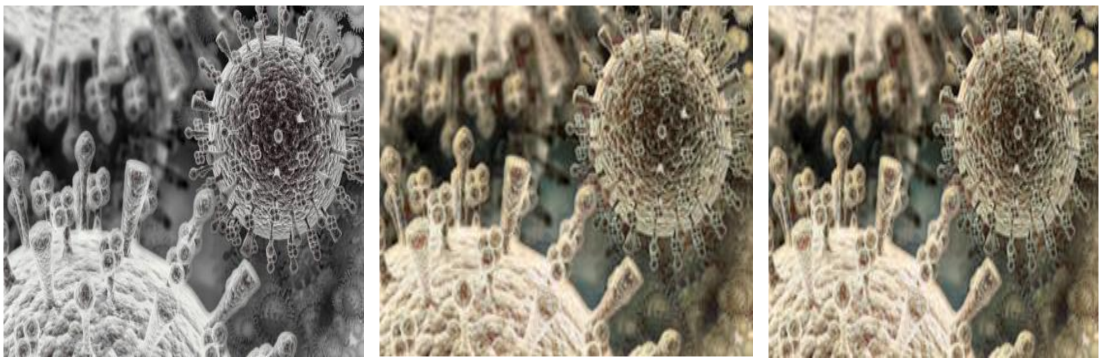
(c) Original image



(A) (B) (C)

Fig5.16: (a)Grayscale image(input image) (b) Caffe model output

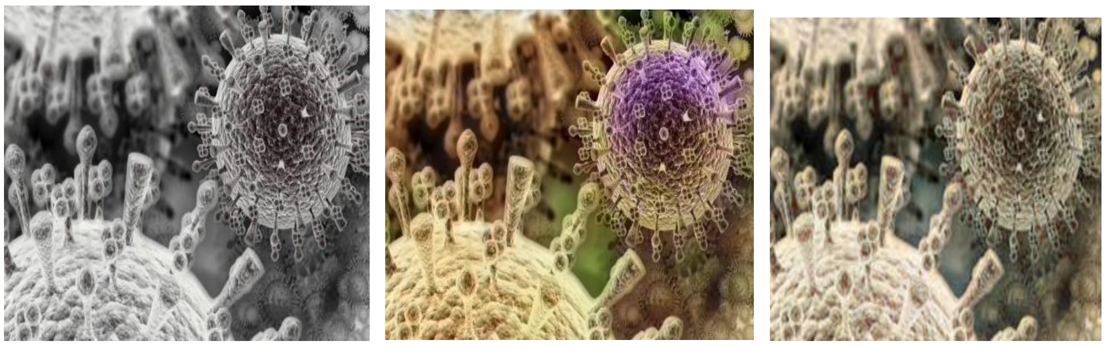
(c) Original image



(A) (B) (C)

Fig5.17: (a)Grayscale image(input image) (b) Autoencoder output

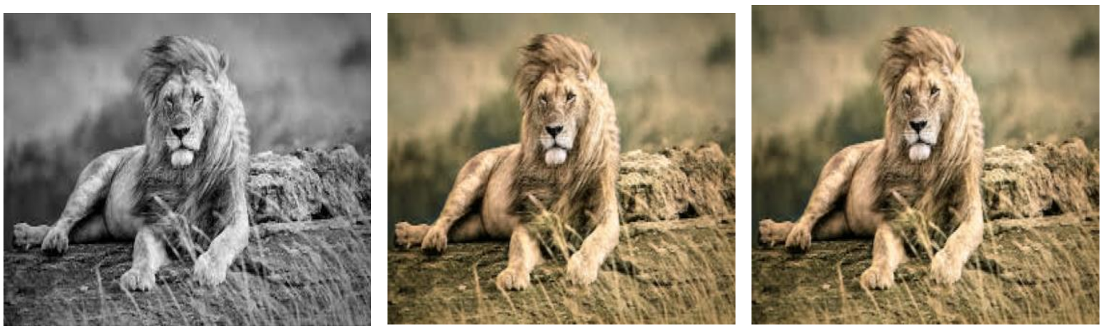
(c) Original image



(A) (B) (C)

Fig5.18: (a)Grayscale image(input image) (b) Caffe model output

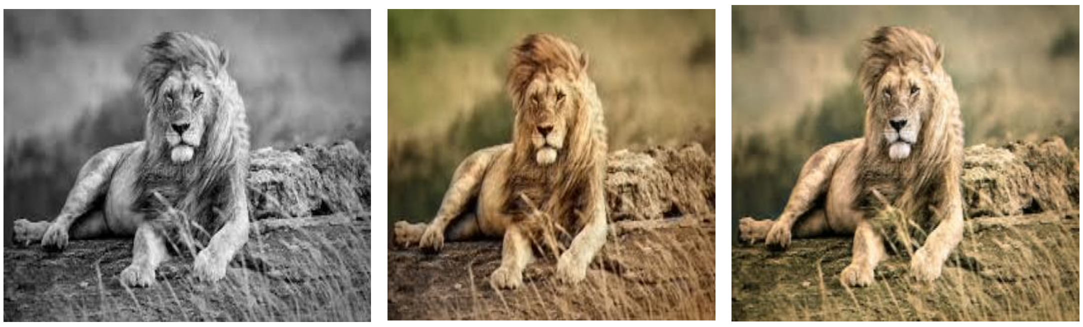
(c) Original image



(A) (B) (C)

Fig5.19: (a)Grayscale image(input image) (b) Autoencoder output

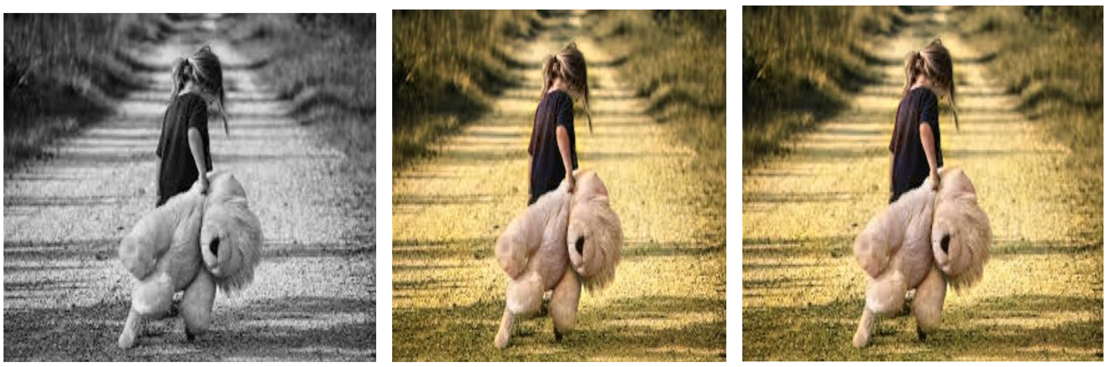
(c) Original image



(A) (B) (C)

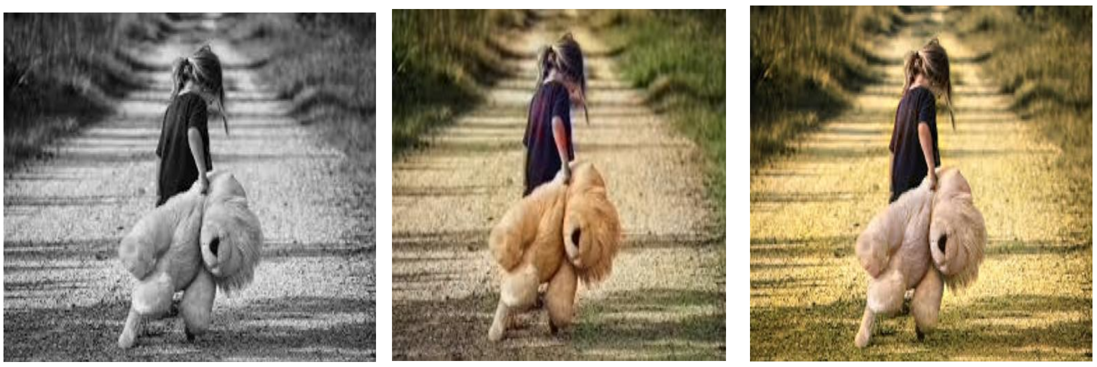
Fig5.20: (a)Grayscale image(input image) (b) Caffe model output

(c) Original image



(A) (B) (C)

Fig5.21: (a)Grayscale image(input image) (b) Autoencoder output (c) Original image



(A) (B) (C)

Fig5.20: (a)Grayscale image(input image) (b) Caffe model output (c) Original image

Result Analysis

Caffe Model:-

Caffe model output contains the rust type color information if any image with white background along with the lightness information it will generate the output non-rendering image with low visibility and low accuracy.

This Caffe model works on L-channel only; it will file to predict the color channels.

Autoencoder:-

autoencoders outputs contain the color information with proper visibility and are good at passing the information on virus images like

1.the virus is healthy or not 2. the virus color is changing to the respective health conditions or not etc...

We are training the model with some virus images with different colors of health environments and also we train the model with grayscale images and color images.during predicting the color information by passing the grayscale image as input to the model it will produce the color information that it predicts color channels like A-channel and B-channel with good accuracy and with low mean square error(low loss).

whenever a grayscale image is applied as input to our model it will able to produce the RGB image with a reasonable understanding of the spatial relationship of color compared to the Caffe framework model from the results we analyzed the autoencoder technique will be able to produce the more color information as output than Caffe framework.

CONCLUSION

The proposed work is implemented in python. we have a picture of old black and white images, microscopic images, and some ultrasound images we applied the convolutional learnable filters for the error occurrence from that we performed a loop of the same operation for getting high accuracy and quality of output colored images. the reestablished went under the decoder section. the CNN is trained with both color images and black & white images and classified image input for normal and abnormal conditions. we preprocess colored images to create grayscale images to use as the input for the model. our model is then trained with these grayscale images as input and the original colored images as output.so, if we feed the new unseen grayscale image to the model, it would be able to generate an RGB image with a reasonable understanding of the spacial relationship of color, etc. this work proves that image pre-processing like normalization and resizing of image and convolution of image segmentation, upsampling, downsampling and classification. the results show that mounting the color to each area of the picture with more accurate compared to the previously developed approaches.

FUTURE WORK

Earth’s atmosphere alters and blocks the light that comes from space. Hubble orbits above Earth’s atmosphere, which gives it a better view of the universe than telescopes have at ground level.

Hubble transmits about 140 gigabytes of science data every week back to Earth. That's equal to about 45 two-hour, HD-quality movies or about 30,000 mp3 songs. The digital signals are relayed to satellites, then to a ground station, then to NASA's Goddard Space Flight Center, and finally to the Space Telescope Science Institute. The STScI translates the data into images and information we can understand.

Hubble pictures start out as shades of black and white. The Space Telescope Science Institute adds colors to the pictures for different reasons. Sometimes colors are chosen to show how an object might look to the human eye. Other times colors are used to highlight an important detail. Or they can be used to show details that would otherwise be invisible to the human eye.

To get colorized image from hubble telescope it requires large electronic filter and it increses the cost so,our autoencoder technique will be used for getting color images of telescopes by changing A and B channels with some mathematical operations we hope that it will give color image as output with low cost.

REFERENCES

[1]. Leila kiani," Image Colorization Using a Deep Transfer Learning", Proceedings of the IEEE International Conference on Computer Vision, pp. 415-423, 2020.

[2].Swathy Titus, Jency Rena N.M “ Fast Colorization of Grayscale Images by Convolutional Neural Network” 2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)02 September 2019.

[3].Ashwin Nair, “grayscale image to colorized image using GAN” Proceedings of the IEEE International Conference on Computer Vision.09 September 2018.

[4]. N. Anagnostopoulos, C. Iakovidou, A. Amanatiadis, Y. Boutalis, and S. Chatzichristofis, "Transferflow learning method with VGG architecture", IEEE International Conference on Imaging Systems and Techniques, pp. 381-385, 2014.

[5].Zhang, Richard, et al. "Real-time user-guided image colorization with learned deep priors." arXiv preprint arXiv:1705.02999 (2017).

[6].Joshi, Madhab & Nkenyereye, Lewis & Joshi, Gyanendra Prasad & Islam, S. M. Riazul & Abdullah-Al-Wadud, Mohammad & Shrestha, Surendra. (2020). Auto-Colorization of Historical Images Using Deep Convolutional Neural Networks. Mathematics. 8. 2258. 10.3390/math8122258.

[7].Hao Wang,Xuedong Liu,”overview of image colorization and its applications” 2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC),12-14 march 2021.

[8].Ivana zeger, Sonja Grgic, Josip vukovic, gordan sisul, “grayscale image colorization methods: overview and evaluation”ieee access, 12 august 2021.

[9].K. Nassau, Colour, 2020, [online]

Available: https://www.britannica.com/science/color.

[10]. T. Welsh, M. Ashikhmin and K. Mueller, "Transferring color to greyscale images", *ACM Trans. Graph.*, vol. 21, no. 3, pp. 277-280, 2002.

[11]. A. Levin, D. Lischinski and Y. Weiss, "Colorization using optimization", *ACM Trans. Graph.*, vol. 23, no. 3, pp. 689-694, 2004.

[12].E.Trex, "How (and why) are black and white films colorized", 2011, [online] Available:https://www.mentalfloss.com/article/26956/how-and-why-are-blackand-white-films-colorized.

[13].I Žeger and S Grgić, "An Overview of Grayscale Image Colorization Methods[C]", *2020 International Symposium ELMAR*, pp. 109-112, 2020.

[14] Joshi, Madhab & Nkenyereye, Lewis & Joshi, Gyanendra Prasad & Islam, S. M. Riazul & Abdullah-Al-Wadud, Mohammad & Shrestha, Surendra. (2020).Auto-Colorization of Historical Images Using Deep Convolutional Neural Networks. Mathematics. 8. 2258. 10.3390/math8122258.

[15].p, Ajay kalyan and R, Puviarasi and Ramalaingam, Mritha, Image Colorization Using Convolutional Neural Networks (August 23, 2019). Proceedings of International Conference on Recent Trends in Computing, Communication & Networking Technologies (ICRTCCNT) 2019.

[16].Pandey A., Sahay R., Jayavarthini C. (2020). Automatic Image Colorization using Deep Learning. International Journal of Recent Technology and Engineering, Volume-8 Issue-6, 1592-1595.

[17].Kotala S., Tirumalasetti S., Nemitha V., Munigala S. (2019). Automatic Colorization of Black and White Images using Deep Learning. IJCSN Journal Volume 8, Issue 2, 125-131.

[18].Nguyen, Tung & Mori, Kazuki & Thawonmas, Ruck. (2016). Image Colorization Using a Deep Convolutional Neural Network.

[19].R. Dahl. Automatic colorization. http://tinyclouds.org/colorize/, 2016.

[20].Koo, S. (2016). Automatic Colorization with Deep Convolutional Generative Adversarial Networks.

[21].https://keras.io/api/models/model/

[22]Nazeri, K., Ng, E., & Ebrahimi, M. (2018). Image Colorization Using Generative Adversarial Networks. AMDO.

[23].Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scaleimage recognition. arXiv preprint arXiv:1409.1556 (2014).