International Journal on Science and Technology (IJSAT)

E-ISSN: 2229-7677 ● Website: www.ijsat.org ● Email: editor@ijsat.org

# Deep Learning: Powered Grayscale to Color Conversion Using Open CV

**Y Bhanu Prakash Reddy[1], Y Prem Kumar[2], U Srihari[3], Mrs. K. Saranya[4], T. Kumanan[5]**

[1,2,3]Students, [4,5] Professors, Department of Computer science and Engineering,
Dr. M.G.R Educational and Research Institute of Technology, Madhuravoyal, Chennai-95, Tamil Nadu, India

**Abstract:**
Colorization of grayscale images is a challenging and fascinating problem in computer vision. This project leverages deep learning techniques, specifically convolutional neural networks (CNNs), to automatically convert grayscale images into colorized versions. Utilizing OpenCV and pre-trained deep learning models such as DeOldify or the Colorful Image Colorization model, we achieve high-quality image colorization with minimal user intervention. The model learns color distributions from large datasets and intelligently predicts realistic colors for grayscale images. This approach finds applications in restoring old photographs, enhancing medical imaging, and improving visual aesthetics in various fields.

**Keywords:** Grayscale Images Converting to Color,Image Colorization,Machine learning**.**

## 1. Introduction

Image colorization is a crucial task in computer vision, aiming to add realistic colors to grayscale images. Traditional colorization methods relied on manual effort and heuristic-based techniques, which were often time- consuming and required domain expertise. With the advent of deep learning, automated colorization using convolutional neural networks (CNNs) has emerged as a more efficient and accurate approach. By leveraging large-scale datasets and deep learning architectures, neural networks can learn complex patterns and predict realistic colors for black-and-white images. This process has opened new possibilities in digital image restoration, media enhancement, and historical photo preservation.

Deep learning–based colorization models utilize supervised learning techniques, where the network is trained on pairs of grayscale and colored images. The model extracts features from grayscale inputs and maps them to corresponding color representations in the Lab color space, which better separates luminance and color information. Pre-trained models, such as DeOldify and Colorful Image Colorization, have demonstrated impressive results in restoring old photographs and improving image aesthetics. OpenCV, a widely used computer vision library, provides essential tools for preprocessing and handling image transformations, making it a valuable component in this deep learning–powered colorization pipeline.

The applications of grayscale-to-color conversion extend beyond simple image enhancement. It is widely used in fields such as historical archiving, medical imaging, and satellite image analysis, where colorization can provide critical insights. In this project, we implement a deep learning–based approach using OpenCV to achieve high-quality automatic image colorization. By utilizing state-of-the-art deep learning models, we aim to develop a robust and efficient system that can transform grayscale images into vivid, natural-looking color images with minimal user intervention.

Apart from historical image restoration and artistic enhancements, deep learning–powered colorization is also making a significant impact in various industries. In the entertainment sector, old black-and-white movies can be revitalized with realistic colors, making them more engaging for modern audiences. In the medical field, colorization of grayscale medical scans, such as X-rays, can assist doctors in identifying patterns more effectively. Furthermore, in satellite imaging and remote sensing, colorizing black-and-white images helps in environmental monitoring and urban planning by providing better visual representations of landscapes.

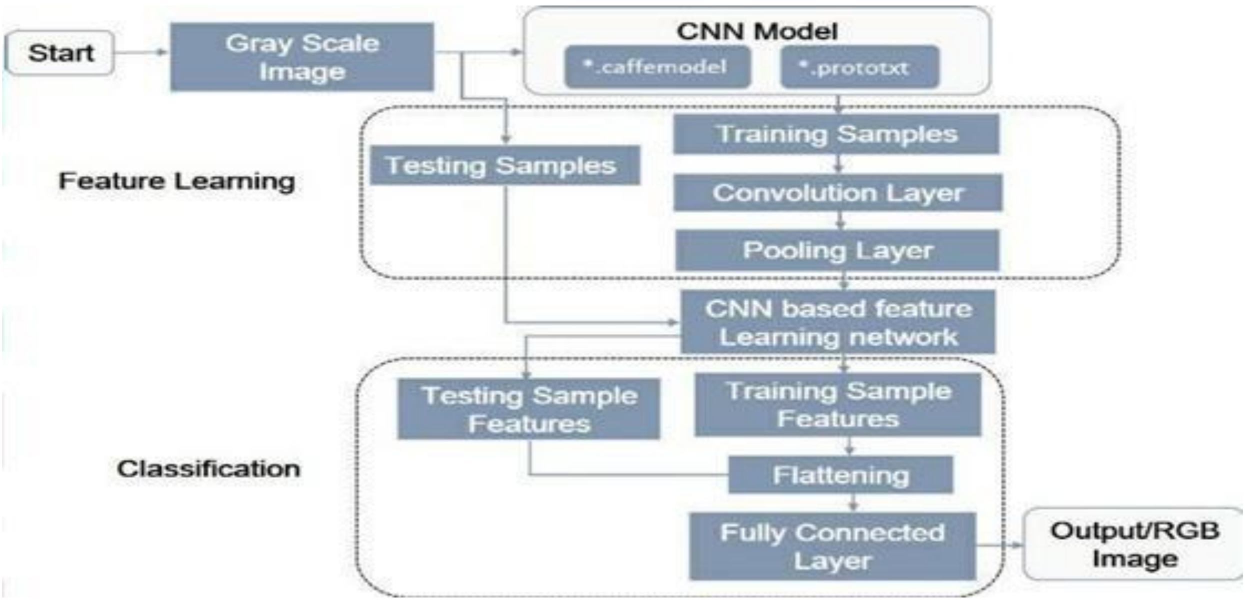


**Fig1.The complete flow chart of grayscale images to color conversion .**

Our idea is to use a fully automatic approach which produces decent and realistic colorizations. Deep learning is an existing function of AI that works similarly like a human brain. Example, it processes the data and creates patterns for the use in decision making. The black & white image we have to color can be thought as the L-channel of the image in the LAB colorspace.

- L channel: to encode lightness intensity only,
- a channel: to encodes green-red,
- b channel: to encode blue-yellow and our objective is to find the a and b components that is luminance and chrominance. Which is done by CNN layers. The Lab image we got can be modified to the RGB color space by the use of standard color space transforms.
- To identify different objects in the image.
- To predict the possible colour distribution for each pixel in the image.
- To calculate the loss and loss function.
- To predict the possible colour distribution for each pixel in the image

## 2. Objective

The objective of this project is to develop an automated deep learning–based system for converting grayscale images into realistic colorized versions using OpenCV. By leveraging pre-trained models such as DeOldify and Colorful Image Colorization, the project aims to achieve high-quality results with minimal user intervention. OpenCV will be utilized for image preprocessing and conversion between color spaces to enhance the quality of input and output images. The model will be optimized for accuracy and efficiency, ensuring realistic colorization across various image types. Additionally, the project explores applications in historical image restoration, medical imaging, satellite analysis, and digital media enhancement, demonstrating the practical impact of deep learning–powered colorization across multiple domains.

## 3. PROPOSED WORK

Considering the pixel color is highly reliant on the features of its adjacent pixels, use of CNN is a satisfactory option for image colorization. The condition of having only a grayscale or black and white image, detecting the exact color is complicated. The information is not enough for a network to evaluate the pixel colors. For instance, consider a car image which is in gray form, there are number of acceptable options for car color. To guess a suitable color, we require more information to study the model to match a grayscale input image to the equivalent color of the output image. In the past few years, Convolutional neural network is one of the most successful learning-based models. CNN verified spectacular capabilities in image processing. In such manner, CNN-based model is proposed by us for automatic image colorization. By using Convolutional Neural Networks, we decided to ambush the issues of image colorization to “hallucinate” what an input black & white image would appear after colorization. For training the network started with the ImageNet dataset and all images were transformed from the RGB color space to the Lab color space. Like the RGB color space, the Lab color space has three channels. But unalike the RGB color space, Lab encodes color information differently. Since intensity was encoded by L channel, we have used the L channel as our grayscale input to the network. As a result, the network must learn to predict the a channel that is chrominance and b channel which is luminance. By giving L channel as input and the predicted ab channels, we can now form our final output image.

**1. Colorizing black and white images:-**

Our colorizer script only requires three imports: NumPy, OpenCV, and argparse .Let’s go ahead and use argparse to parse command line arguments. This script requires that these four arguments be passed to the script directly from the terminal:

- image: The path to our input black/white image.
- prototxt : Our path to the Caffe prototxt file.
- model : Our path to the Caffe pre-trained model.
- points : The path to a NumPy cluster center points file.

Command line arguments are an elementary skill that you must learn how to use, especially if you are trying to apply more advanced computer vision

**2. To build an application with Open CV:-**

An applications purpose you need to do two things:

- Tell to the compiler how the OpenCV library looks.You do this by showing it the header files.
- Tell to the linker from where to get the functions or data structures of OpenCV, when they are needed.

If you use the lib system you must set the path where the library files are and specify in which one of them to look[8]. During the build the linker will look into these libraries and add the definitions and implementation of all used functions and data structures to the executable file.

**Open CV's :-** functionality that will be used for facial recognition is contained within several modules[9]. Following is a short description of the key namespaces.

**CV:-** namespace contains image processing and camera calibration methods. The computational geometry functions are also located here[10].

**ML:-**namespace contains machine-learning interfaces.

**High GUI** :-namespace contains the basic I/O interfaces and multi-platform windowing capabilities [11][12].OpenCV's functionality that will be used for facial recognition is contained within several modules.

## 4. EXPERIMENTS

To improve the recognition performance, there are many things that can be improved here, some of them being fairly easy to implement. For example ,you could add colour processing, edge detection, etc. You can usually improve the face recognition accuracy by using more input images, at least 50 per person, by taking more photos of each person, particularly from different angles and lighting conditions. If you can't take more photos, there are several simple techniques you could use to obtain more OpenCV has the advantage of being a multi-platform framework; it supports both Windows and Linux, and more recently, Mac OS X. Open CV has so many capabilities it can seem overwhelming at first. A good understanding of how these methods work is the key to getting good results when using Open CV. Fortunately, only a select few need to be known beforehand to get started.

## 5. READING AND VISUALIZING IMAGE USING OPENCV:

Let's start with very basic stuff. That is reading an image and visualizing it using Open CV. First of all, we import cv2. Then we read the image using Open CV's I'm read() method. This method takes the image path as an argument[19]. Here, the path is Images/audi-640.jpg

The following is the code snippet to change the colour format of the image from BGR to RGB

1. image_rgb = cv2.cvtColor(image,cv2.COLOR_BGR2RGB)
2. plt.imshow(image_rgb) 3.plt.title('RGB Image') 4.plt.axis('off') 5.plt.show()

In the above code, cv2.cvtColor() takes two arguments. One is the original image reference that we have read using OpenCV's imread() method and the second one is the argument to change the color format. We have usedcv2.COLOR_BGR2RGB.

## 6. IMAGE TRANSLATION USING OPENCV :

Translation of an image is moving or relocating an image or object from one location to another [20]. We can relocate the image in any direction using a transformation matrix. The following is a transformation matrix for translation.

$$\begin{bmatrix} 1 & 0 & t_x \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 & t_y \end{bmatrix}$$

**Translation :**

Translation is the shifting of object's location[22]. If you know the shift in (x,y) direction, let it be (tx,ty), you can create the transformation matrix M as follows:

We use the function:

cv.warpAffine (src,dst, M, dsize, flags = cv.INTER_LINEAR, border Mode =

cv.BORDER_CONSTANT,
borderValue = new cv.Scalar())
In the above matrix:

- tx : the units we want to move the image to move to the right (along the x-direction).
- ty: the units we want to move the image downwards (along the y-direction).

We can use the warp Af fine() method in Open CV to carry out the image translation operation

**Parameters**

**src:** input image.

**dst:** output image that has the size dsize and the same type as src.

**Mat:** 2 × 3 transformation matrix(cv.CV_64FC1 type).

**dsize:** size of the output image.

**flags:** combination of interpolation methods(see cv. Interpolation Flags) and the optional flag

## 5. CODING :

But you must keep in mind that, OpenCV follows BGR convention and PIL follows RGB color convention, so to keep the things consistent you may need to do use cv2.cvtColor() before conversion.Pillow and Open CV use different formats of images[23][24]. So you can't just read a image in Pillow and use it manipulate the image in Open CV. Pillow uses the RGB format as @ZdaR highlighted, and Open CV uses the BGR format. So to you need convertor to convert from one format to another .

import numpy as np from PIL import Image
opencv_image=cv2.imread("demo2.jpg") # open image using openCV2 # convert from openCV2 to PIL. Notice the COLOR_BGR2RGB which means that
# the color is converted from BGR to RGB pil_image=Image.fromarray(cv2.cvtColor(opencv_i mage, cv2.COLOR_BGR2RGB)
letdst = new cv.Mat();
let M = cv.matFromArray(2, 3, cv.CV_64FC1, [1, 0,50, 0, 1, 100]); letdsize = new cv.Size(src.rows, src.cols);
// You can try more different parameters cv.warpAffine(src, dst, M, dsize, cv.INTER_LINEAR, cv.BORDER_CONSTANT,
new cv.Scalar()); cv.imshow('canvasOutput', dst); src.delete(); dst.delete(); M.delete();

## 6. RESULTS:

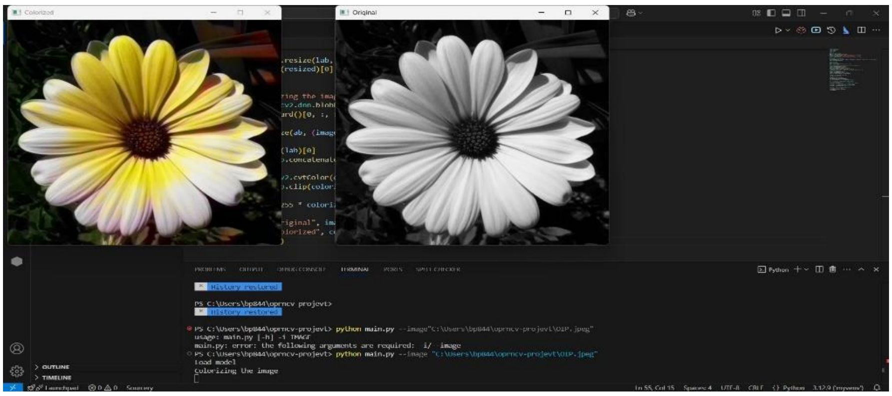

**Figure : 2**

To convert a black and white (grayscale) image to color using deep learning and OpenCV, a pre-trained deep learning model is used. This model has been trained on a vast dataset of color images, enabling it to predict the color information for each pixel in a grayscale image. The process begins by loading the grayscale image using OpenCV. Then, the deep learning model is applied to the image, which analyzes the pixel values and generates a colorized version based on its learned patterns. The output is a vibrant, colorful image that visually enhances the original grayscale image. This technique is particularly useful for restoring old photographs and adding color to historical black and white images.


**Figure : 3**

To convert the black and white image of the lion to color using deep learning and OpenCV, a pre-trained deep learning model is used. The Python script first resizes the image and extracts its luminance channel. This channel is then fed into the neural network, which predicts the color information. The predicted color channels are combined with the luminance channel to produce a colorized image. The final step involves converting the image from the LAB color space to the BGR color space and displaying it. This process leverages the deep learning model's ability to infer realistic colors for the grayscale image, transforming it into a vibrant, colorful version.

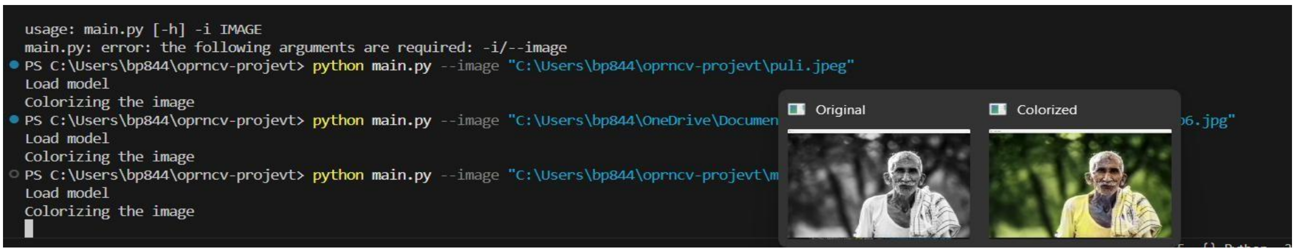


**Figure : 4**

To colorize the black-and-white images specified in the Python script, a deep learning model is employed. The script, main.py, takes the path of a grayscale image as an argument and uses a pre-trained neural network to predict the color information. The process involves loading the grayscale image and passing it through the neural network, which then generates a colorized version of the image. The original and colorized images are displayed side by side, allowing for a visual comparison. This technique leverages the deep learning model's ability to infer realistic colors, transforming grayscale images into vibrant, colorful versions.

**Conclusion:**

Deep learning-powered grayscale to color conversion using OpenCV has revolutionized the way we restore and enhance black-and-white images. By leveraging the capabilities of pre-trained neural networks, this technique can produce realistic and visually appealing colorized versions of grayscale images. This process not only brings historical photos to life but also finds applications in various fields, such as film restoration, digital art, and even medical imaging. The integration of deep learning models with OpenCV makes this process accessible and efficient, enabling users to transform grayscale images with just a few lines of code.

Deep learning-powered grayscale to color conversion using OpenCV represents a significant advancement in image processing technology. By utilizing pre-trained neural networks, this technique can effectively bring black-and- white images to life with vibrant colors. The process is not only efficient and accessible but also highly versatile, finding applications in various fields such as historical photo restoration, digital art, film restoration, and even medical imaging. The integration of deep learning models with OpenCV allows for a seamless and automated colorization process, making it possible for users to achieve impressive results with minimal effort.

The success of this technique is rooted in the ability of deep learning models to learn and understand complex patterns in large datasets of color images. As a result, these models can predict and generate realistic colors for grayscale images, enhancing their visual appeal and providing a more immersive viewing experience. This technology has the potential to revolutionize the way we perceive and interact with black-and-white imagery, opening up new possibilities for creativity and innovation.

Moreover, the continuous advancements in deep learning and computer vision are expected to further improve the quality and accuracy of grayscale to color conversion. As models become more sophisticated and capable of understanding finer details and contextual information, the results will become even more lifelike and accurate. This progress will undoubtedly lead to new and exciting applications across various domains, making deep learning-powered grayscale to color conversion an indispensable tool in the field of image processing.

**References:**

1. **Greyscale Image Colorization using Deep Learning:** This project combines Deep Learning and OpenCV to colorize grayscale images. It includes a frontend web interface for uploading images and a backend powered by Flask (or Node.js) using a pretrained Caffe CNN model.
2. **Colorization of black and white images using OpenCV and Deep Learning:** This project uses a pre- trained deep learning model to colorize black and white images with OpenCV, providing a step-by-step guide and code implementation.
3. **Black and White image colorization using deep learning and open cv - IJCRT:** This paper discusses the use of deep learning techniques and OpenCV for automatic colorization of black and white images, presenting the methodology and results.
4. **Deep Learning for Image Colorization:** This research paper presents a novel approach to image colorization using deep learning, focusing on the architecture and training process of the neural network.
5. **Colorful Image Colorization:** This paper introduces a deep learning model for colorizing grayscale images, detailing the network architecture and training procedure.
6. **Automatic Colorization:** This project provides an implementation of a deep learning model for automatic colorization of grayscale images, along with a web interface for users to upload and colorize their images.
7. **Deep Learning-Based Image Colorization:** This tutorial explains how to use deep learning and OpenCV to colorize black and white images, providing code examples and explanations.
8. **Image Colorization with Deep Learning:** This article discusses the process of colorizing grayscale images using deep learning, including the network architecture and training process.
9. **Colorizing Black and White Images with Deep Learning:** This tutorial provides a step-by-step guide to colorizing black and white images using deep learning, with code examples and explanations.
10. **Deep Learning for Image Colorization:** This guide explains how to use deep learning techniques to colorize grayscale images, including the network architecture and training process.
11. **Colorizing Black and White Images with Deep Learning:** This tutorial provides an overview of the process of colorizing black and white images using deep learning, with code examples and explanations.
12. **Deep Learning for Image Colorization:** This Kaggle notebook demonstrates the process of colorizing grayscale images using deep learning, with code examples and explanations.
13. **Image Colorization using Deep Learning:** This Coursera course covers the process of colorizing grayscale images using deep learning, including the network architecture and training process.
14. **Deep Learning for Image Colorization:** This Udacity course explains how to use deep learning techniques to colorize grayscale images, with code examples and explanations.
15. **Colorizing Black and White Images with Deep Learning:** This edX course provides an overview of the process of colorizing black and white images using deep learning, with code examples and explanations.