Deep Learning – Case Study

Image Colorization using AutoEncoders and ResNet

Name: Het Suthar

Enrollment Number: 18012021083

Batch: DL2

1. Introduction

This case study is designed to convert black and white images to the colored images. It uses the AutoEncoders and ResNet model to colorize the images. The model built is able to convert the grayscale or Black & White images into the colorful RGB images. It is trained on the Paintings dataset and so it is able add colors to the painting.

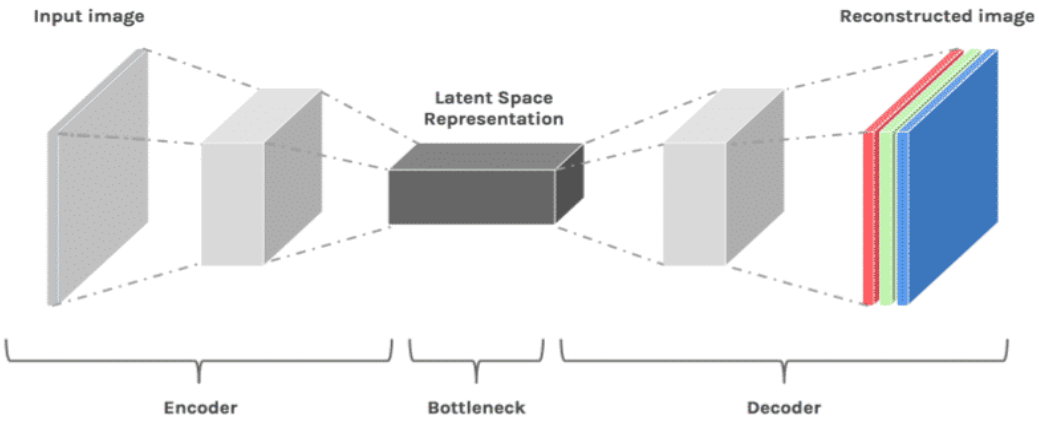
This case study can be used across various sectors where there is a need to have the colored images out of black and white images. Also, if this model is trained with more resources like GPU, RAM then it can be expanded to cover variety of image categories.

2. Tools and Technologies

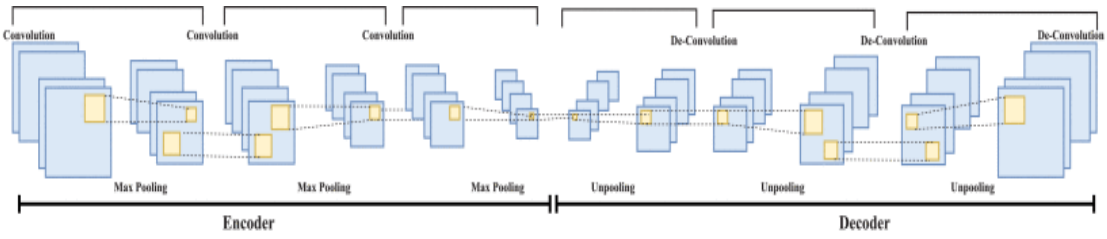
Tools and Libraries	Usage
Keras	This library is used for building the network architecture. It allows us to use several layers, callbacks, and InceptionResNetV2 model.
SkImage	Library provides various image transformation, colors, cropping related functions.
Tqdm	Provides a good progress bar that can be combined with a loop to get visualization of the current progress.
PIL	For loading the viewing the images.
Kaggle	Used as a platform to execute code as well as manage datasets and model weights

3. Model Explanation and Architecture

The model architecture is a combination of both AutoEncoders and ResNet classifier. The Encoder at its best is just shades everything in a brownish tone. Hence, to let the network an "idea" about what things to color, we add ResNet.



The InceptionResNetV2 model is loaded with pretrained weights available. These weights are taken from the model which was trained on nearly 1.2 million images.
AutoEncoder contains three main parts: 1. Encoder, Fusion, Decoder parts.



Encoder holds the Conv2D and MaxPooling2D Layers. Fusion holds the RepeatVector and concatenation with the previous output layers. Decoder holds the Conv2D and UpSampling2D layers.
We've used **"Art Images dataset"** which contains various categories of the image. For eg. Drawing/Painting/Sculptures/Engravings.

4. Working

The images we use daily contains three color channels namely Red, Green & Blue. These channels when combined together creates the colored image. The values of these

channels can vary from 0 to 255. While training the model, we use the colored images and extracted the grayscale image and we separate out the color channels.

Normally we have to work three channels of images, but we will convert the RGB channel to LAB parts. Where L is for Lightness, ‘a’ is for the color spectra ‘green-red’ and ‘b’ is for color spectra ‘blue-yellow’.

We are taking lightness into consideration because, 94% of the cells in our eyes determines the brightness. That leaves only 6% of our receptors to act as sensors for colors. The grayscale images are a lot sharper than the color layers. Hence, grayscale image is used to keep the origin sharpness into the final prediction.

We will use the grayscale image as an input and we’ll have to predict the color channels using the input image. In between we’ll use the convolutional filters which will let us achieve the final prediction.
The input images go across encoder, fusion and decoder parts of the network.

We’ve also used callbacks with model which monitors the loss and also adjusts the learning rate. These checkpoints save the best model.

5. Code

https://github.com/hetsuthar028/Image-Colorization-AutoEncoder-ResNet

** Code and Report is available in the GitHub repository.

6. Output