

Coloring Grayscale Images

Arihant Parsoya (15D260012), Sudip Kumar (150040080), Ayush Pandey (150040087)

Abstract. In this project we have implemented a neural network to color images from grayscale to RGB space. Our model uses Convolution Neural Network (CNNs) and which is combined which uses high level feature extraction from already trained Inception-ResNet-v2.

1. Introduction

Image colorization is the process of coloring images from grayscale to RGB color space. This has applications such as coloring old grayscale photographs and movies. Coloring grayscale image is primarily an information retrieval problem. When an RGB image is converted into grayscale, it intrinsically loses information about the images. Our primary task is to predict the lost information based on the already available grayscale information.

We are following the paper Deep Koalarization: Image Colorization using CNNs and Inception-Resnet-v2 by Federico and Diego [1]. Their model has shown good results in variety of images types. For the sake of this project we have restricted the scope of the project for limited set of images. Particularly, we focused on portrait type images due to limited computation availability and time constraints.

We have defined our problem statement in section 2 detailing the basics of image colour space which we have used in our model. We have explained our CNN model framework in section 3. Section 4 shows the details of the experiments we did on the data and the model.

2. Problem Statement

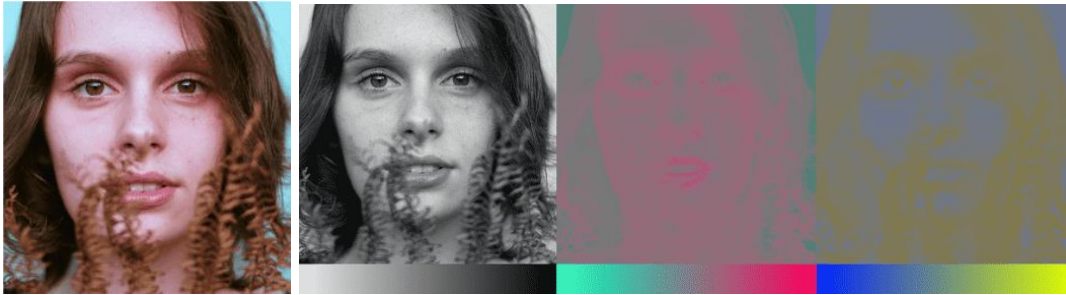
Given an image in grayscale color space, our task is to produce an image in RGB space as shown in figure 1.



Fig 1. Image on the left is an image in grayscale color space. The image on the right is of an image in RGB space. (Source: [Fstoppers](#))

2.1 Lab colour space

An image in RGB space can be represented as an image in **Lab** space. **L** stands for lightness/greyscale, **a** and **b** stands for the spectrums green-red and blue-yellow.



*Fig 2. An image in Lab space. Leftmost image shows an image in RGB color space. The remaining images are in **Lab** space (left to right).*

There is information loss when a coloured image is converted into grayscale space. Hence our aim is to predict the lost information using existing data. We already have the image information in greyscale space. We need to predict the **a** and **b** color spaces in order to produce the RGB image.

3. Model

Our model is loosely inspired by the fact that there is information available in the gradient of grayscale colours. Training a new NN to colour image from scratch is not favourable because the NN does not learn to do feature detection in the image before coloring the objects present in the image. The model tries to detect the features in the image based on given information such as local variation of the colour intensities of grayscale.

3.1 Inception-ResNet-v2

We have used pretrained Inception-ResNet-v2 do high level feature detection in the image. Inception-ResNet-v2 has already been trained on millions of images and is used for feature detection in our model directly using the fusion layer which is explained in next section.

3.2 Framework

We have divided our model into three parts: encoder, decoder and fusion modules.

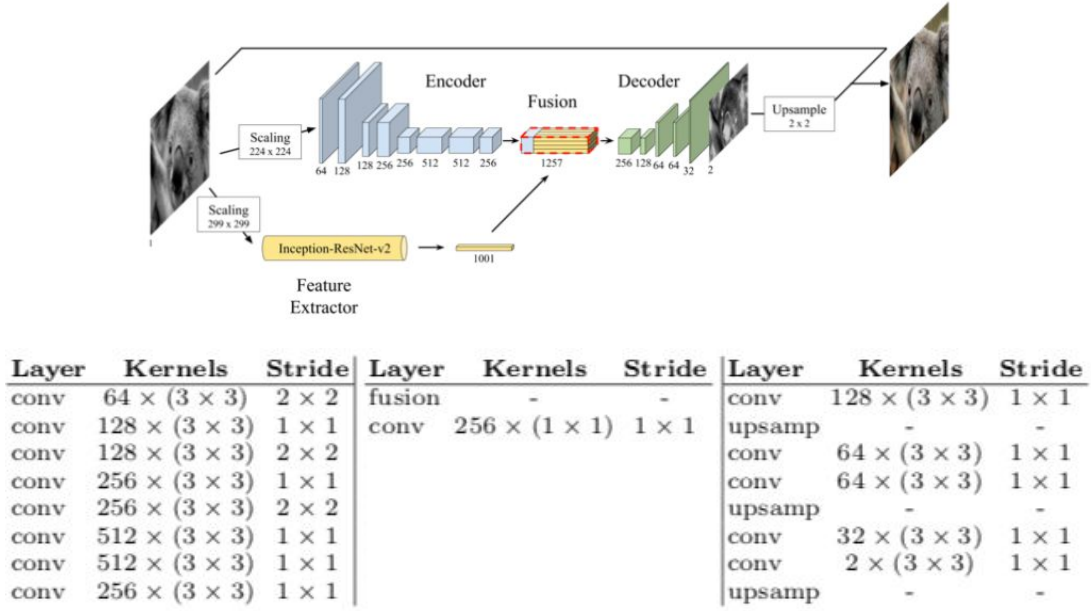


Fig 3. Figure showing NN architecture. (Source: Federico and Diego [1])

3.1.1 Encoder

The encoder process uses $H \times W$ gray-scale images and outputs $H/8 \times W/8 \times 512$ feature representation. It uses 8 convolutional layers with 3×3 kernels.

3.1.2 Fusion

Fusion layer takes the input from the feature vector extracted using inception ResNet-v2, replicates it $HW/8^2$ times and attaches it to the feature volume obtained from the encoder along the depth axis. It obtains a single volume with the encoded image and feature vectors of shape $H/8 \times W/8 \times 1257$. By doing this we ensure that the semantic information conveyed by the feature vector is uniformly distributed over the whole spatial region of the image. Finally, we apply 256 conventional kernels of size 1×1 , ultimately generating a feature volume of dimension $H/8 \times W/8 \times 256$.

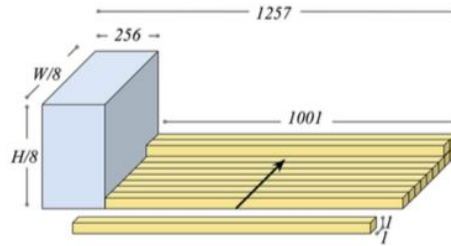


Fig 4. Fusion Layer (Source: Federico and Diego [1])

3.1.3 Decoder

The decoder takes this $H/8 \times W/8 \times 256$ volume and applies a series of convolutional and up-sampling layers in order to obtain a final layer with dimension $H \times W \times 2$.

3.2 Objective Function

We have used MSE loss function to train our model with Adam optimizer[1]. Our loss function is:

$$C(\mathbf{X}, \boldsymbol{\theta}) = \frac{1}{2HW} \sum_{k \in \{a,b\}} \sum_{i=1}^H \sum_{j=1}^W (X_{k_{i,j}} - \tilde{X}_{k_{i,j}})^2$$

Where $\boldsymbol{\theta}$ represents all model parameters, $X_{k_{i,j}}$ and $\tilde{X}_{k_{i,j}}$ denote the ij :th pixel value of the k :th component of the target and reconstructed image, respectively. H is the height of the image and W denotes the width of the image. We have used Adam operator with learning rate of 0.001. Total number of parameters in our model is approximately 6500.

4 Experiment Details

4.1 Dataset

We have collected the data by randomly sampling frames from downloaded youtube videos. We downloaded interview videos which allowed us to get data for different facial expressions and facial orientations. The complete dataset can be found here[3].

4.2 Experiments

We have restricted the scope of the project to colorize only human portrait images. We trained our model on a dataset containing 4000 images.




Batch	Steps	Epoch	Comments	
4000	10	1	No colourization of images as seen. The input images was more grayish	
100	10	1	Brownish/Reddish image output	
10	100	2	Brownish/Reddish image output	
100	10	3	No significant difference was measured	

Table 1. Outputs from different experiment settings

Table 1 details the experiments done on our model. We ran the training on around 100 training examples to test our program on CPU before training on the complete dataset. We used Adam optimizer [2] with learning rate of 0.001. We did not observe any overfitting in our results hence we did not use any regularization in our model.

4.3 What worked and what didn't work

- While running the NN, we did not observe any significant gain on increasing the epochs for when the epoch is below 10.
- When using diverse dataset, we observed that there were no visible changes in the images. As compared to using the dataset for a specific scenario such as a specific video.
- We tried to measure how many epoch we require to overfit a single grayscale image to get an estimate of the number of epoch we require to train the model. However, the loss was not reducing any further after 4 epochs.

5. Conclusion

In this project we have explored the process of grayscale image colorization by downsampling the image using CNN, performing feature detection on the image to detect the objects and finally upsampling the image while colouring it. We hope using larger training dataset will improve the performance of the model.

We planned to apply this model on other types of images such as sceneries and animals. We also planned to train the model using different training rates and regularizing the network but couldn't due to high computational power required to train this NN. The model can also be trained by using different loss function other than MSE as done by Richard and Phillip[4].

References

1. Deep Koalarization: Image Colorization using CNNs and Inception-Resnet-v2 by Federico and Diego (*link: <https://arxiv.org/abs/1712.03400>*)
2. Adam, A Method for Stochastic Optimization (*link: <https://arxiv.org/abs/1412.6980>*)
3. Link for dataset used: Floydhub: aDeep Koalarization: Image Colorization using CNNs and Inception-Resnet-v2 (*link: <https://arxiv.org/abs/1712.03400>*)
4. Colourful Image Colourization by Richard and Phillip (*link: <https://arxiv.org/abs/1603.08511>*)