Deep Colorization with CNNs

Rishabh Aryan Das(UFID:0536-9273) Amit Bhadra(UFID:6649-4087)

Abstract

In this project given a grayscale image as input, the project addresses the problem of visualizing a plausible color version of the image. The colorized image is generated by combining both global priors and local image features. Our Convolutional Neural Network contains a fusion layer that permits us to merge local information dependent on small image patches with global priors computed from the entire image. The entire network including the global and local priors along with the colorization model is trained in an end to end fashion.

1 Introduction

Colorizing an image is assigning a color to each pixel of a target grayscale image. Colorization methods can be roughly divided into two categories: scribbled-based colorization adn example-based colorization. The previous methods require much efforts from the user to provide considerable scribbles on the target grayscale images. It is thus a very strenuous procedure to colorize an image with fine-scale structures. This project explores a method where the color information is transferred from a reference image to a target grayscale image.

2 Dataset

2.1 Image Augmentation

The dataset is provided to us in the form of face images. The path to the dataset is given below:
$ cd ./colorNet/face_images/
which contains 750 images of faces of random people. We split the dataset into test_data and train_data in the ratio 1:9. Then we augmented the train_data by a factor of 10x along the first dimension of the batch tensor.
$ python ./colorNet/img_aug.py
This commands creates the ./aug_images/ directory which contains 6750 images generated by applying random transform in the train_data.

2.2 Image Splitting

We need to convert the augmented images from **RGB** colorspace to **LAB** colorspace. This is done by **TrainImageFolder()** function of the **utils.py** script which generates l and ab channels seperately.

3 Network Architechture

The model consists of four major components

- Low-Level Feature Network
- Mid-Level Feature Network
- Global Feature Network
- Colorization Regressor CNN

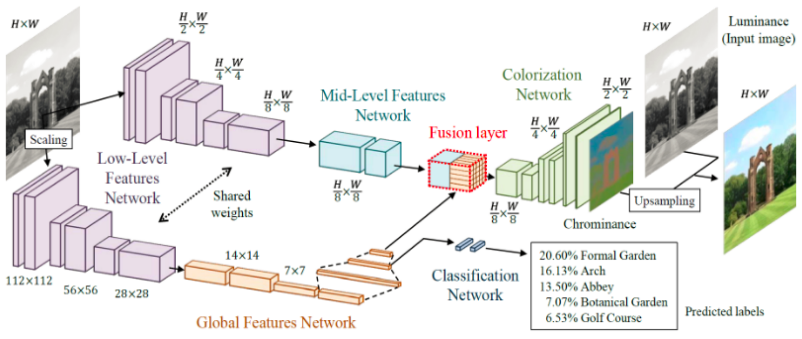

Figure 1: Colorization Architechture

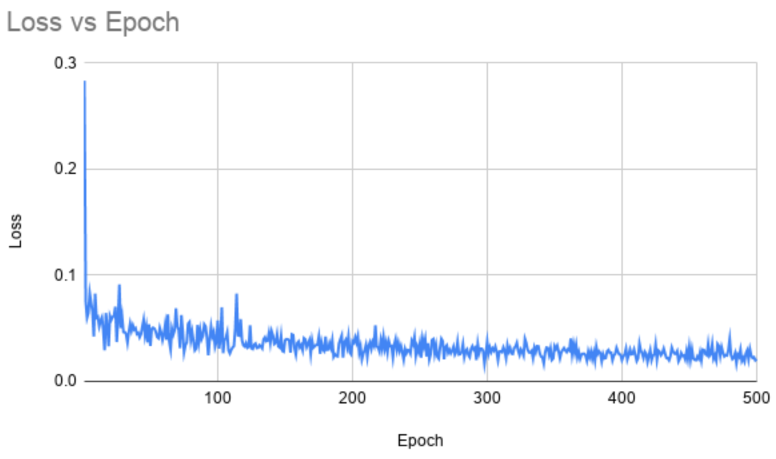

Figure 2: Plot of Loss Function over epoch

The componenets are tightly coupled and trained in an end to end fashion. The model generates the chrominance of the image which is fused with the luminance to produce the colored image. As a baseline, we also test our model without the global features, consisting of the low-level feature network, the mid-level feature network and the colorization regressor CNN with a 3x3 convolution kernel, all concatenated together. The fusion layer denotes this kernel.

4 Training the Network

Use the run_script.sh shell script to start training the model.

$ bash run_script.sh

The below parameters should be passed into the argument parser. They are displayed along with their default values below.

Hyperparamter	**Default Value**
Workers	10
Epoch	500
Batch Size	32
Learning Rate	0.1

$ python train.py ./aug_image/ -j 10 –epochs 100 -b 32

The script picks up the augmented images from the ./aug_image/ directory. The images being stored here should be stored as per their class directory. This will enable the classifier to learn images based on their classes. We have used Adam optimizer for gradient descent and mean squared error(MSE) as our loss function. The loss data per epoch is stored in the log file log.

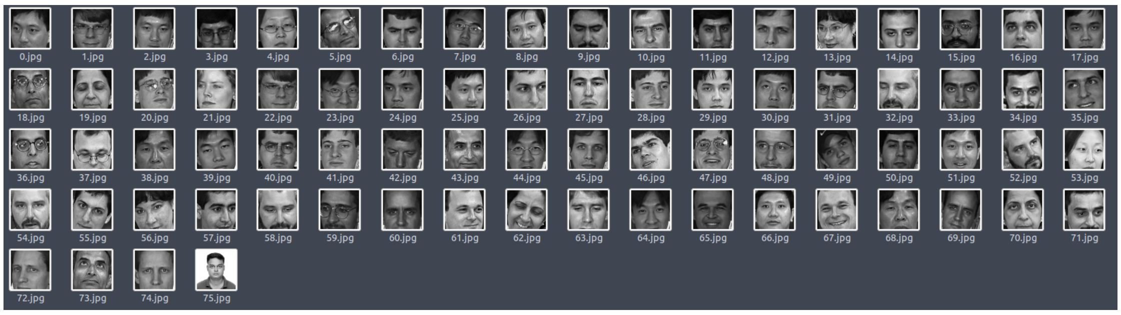

Figure 3: Labels

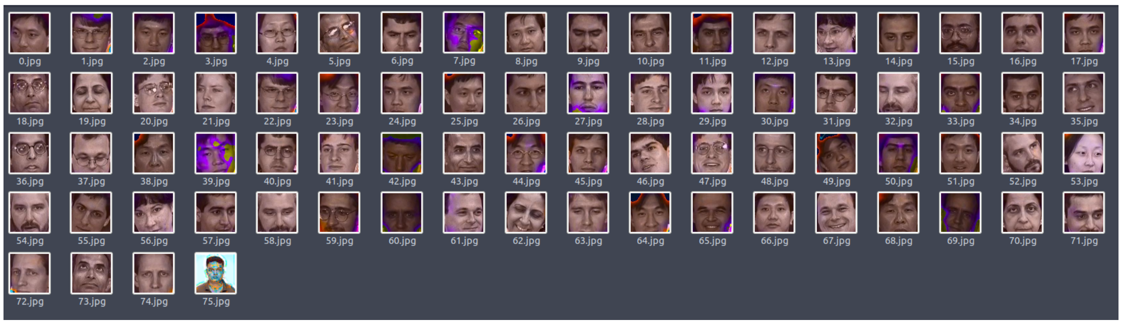

Figure 4: Predictions

5 Colorize Greyscale Images

The **./test.py** script generates the greyscale images from the **./test_data** directory. The greyscale images are stored in **./black_white** directory. These greyscale images are then fed as input to the pretrained CNN regressor which generates the corresponding ab channel predictions as per the pretrained weights. The predicted images are stored in the directory **./color/**.

6 Conclusion

The regression network outputs are somewhat reasonable. Green tones are restricted to areas of the image with foliage, and there seems to be a slight amount of color tinting in the sky. We, however, note that the images are severely desaturated and generally unattractive. These results are expected given their similarity to Dahl's sample outputs and our hypothesis.
In contrast, the classification network outputs are amazingly colorful yet realistic. Colors are lively, nicely saturated, and generally tightly restricted to the regions they correspond to. For example, for the top image, the system managed to infer the reflection of the sky in the water and colorized both with bright swaths of blue. The foliage on the mountain is colorized with deep tones of green. Overall, the output is highly aesthetically pleasing