Image Colorization through CNNs using Local and Global Image Features || Team iGAN

Bhumi Dinesh Bhanushali
bbhanush@andrew.cmu.edu
Kathan Nilesh Mehta
kathannm@andrew.cmu.edu
Avinash Hemaeshwara Raju
ahemaesh@andrew.cmu.edu
Atulya Ravishankar
aravisha@andrew.cmu.edu

November 16, 2019

Abstract

We propose a CNN-based deep model that uses a combination of low-level local image features and high-level semantic features to colorize grayscale images. Our approach is based on the model developed by Baldassaree et al. from KTH Royal Institute of Technology in 2017. The low-level features are computed using an encoder (CNN) while the high-level features are computed using a pretrained Inception-v3 network. These two sets of features are combined and the output is generated using a decoder (another CNN). We will assess the performance of our system through user surveys and plan to demonstrate our results on samples from the ImageNet or COCO dataset.

1 Problem Statement

Being able to take a black-and-white image and convert it to a realistic full-color image is a task that has a wide variety of applications. For instance, historians can use such technology to modernize old images and security systems can use this technique to convert grayscale feeds from CCTV cameras into color feeds for easier monitoring. The information contained in single-channel images (e.g. grayscale) is relatively limited. Thus, by being able to convert a grayscale image to a color image, we can enrich the image by adding more information in the form of additional channels. Not only will this improve the viewing experience for humans, but it should also enable automated systems to extra more insights from the image (e.g. richer image features, better semantic information about

the scene etc.). In fact, the performance of current deep learning models like ResNet and VGG is better on color images compared to grayscale images [1].

This project builds on a model proposed in 2017 by researchers at KTH Royal Institute of Technology [1]. In their approach, they combine both low-level local features and high-level global image features to improve colorization performance. They use an encoder for low-level feature extraction, Inception-ResNet-v2 for global feature extraction and a decoder to combine both of these to produce the output. The goal of this project is to make improvements on that baseline model and demonstrate our results. Some ideas we intend to explore in the next few weeks are refining the loss function (since MSE loss is not ideal for this particular application) and modifying the encoder and Inception-ResNet-v2 fusion process. We are using Pytorch to implement our pipeline from the ground up and we will train our model using either the ImageNet or COCO dataset, subject to availability (discussed later).

2 Literature Review

A lot of interesting work[3,5,6,8] has been done to solve the problem of colorization of grayscale images and it is still a very open-ended problem being pursued quite actively today as well. After going through multiple papers and blogs covering this topic, we decided to implement and modify the technique proposed by Baldassarre et al. [1]. The authors use a deep Convolutional Neural Network to process the luminance component of an image. In order to obtain high-level features, they have used the Inception-Resnet-v2 pre-trained model and an encoder to extract mid-level features.These two differently obtained features are combined in a fusion layer and then up-sampled and decoded using another Convolutional Neural network.

Another technique suggested by Mingming He et al. [2] uses exemplar based colorization network. This method expects the user to provide reference image which is semantically related to the target image. The authors have used Deep Image Analogy for matching the two semantically-related images. They have further used the VGG-19 model to train on the luminance channel of an image. The loss criteria used in this research is noteworthy. The authors have calculated two losses - Chrominance loss and Perceptual loss and tried to minimize both the losses separately. But, this method assumes that we have a semantically related image for every image we train, but the loss function is definitely something we would like to try on our model instead of the naive MSE loss.

We also found a couple of blog posts to be very useful towards improving our understanding of the architecture and the general methodology. The blog by Emil Wallner on FloydHub [4] is a pretty comprehensive overview of the different methods and architectures that can be applied along with the advantages and drawbacks of each. The author suggests three architectures which incrementally build in complexity, along with sample results, statistics and drawbacks of each. The final of the three architectures mentions the use of a fusion layer to merge the feature embeddings to the encoder output, which is quite similar to what we

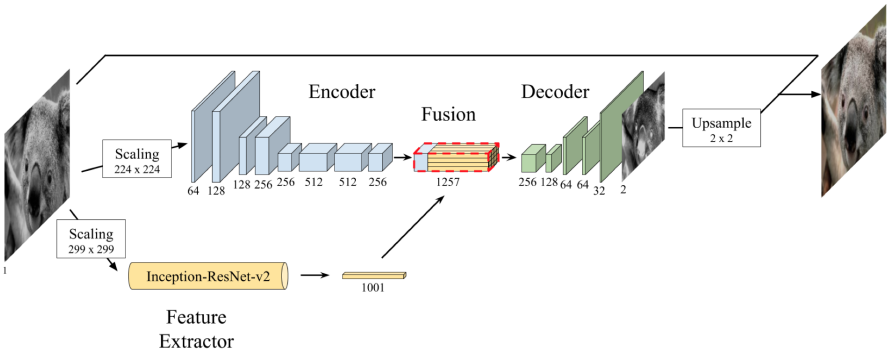

Figure 1: Baseline architecture.

do.

The Colorful Image Colorization by Richard Zhang et al.[9] treats the problem as that of optimizing a multinomial classification. This is another unique way of calculating the loss which can be potentially used. Along with that, the authors also come up with a 'Colorization Turing Test', wherein the true measure of a good model is the percentage of the output of the model which can be used to fool the human into believing that its the ground truth. There is a lot of work in this domain that uses classical Computer Vision techniques like Segmentation, key frame generation and color transfer illustrated by Vivek George Jacob and Sumana Gupta in [10].

3 Method

3.1 Baseline Model

We have chosen to adopt an approach strongly based on that proposed by Baldassarre et al. [1]. The input to the network is a single-channel image representing a grayscale input. In our case, this is the L-channel of an image that has been converted to the CIE Lab color space [7]. The pixel values of the input are normalized to the range [-1 1]. The output of the network is a 2-channel image that represents the predicted a- and b-channels of the colored image. The original L-channel is combined with the a- and b-channels output by the network to form the final color image.

At a high level, the network can be broken down into four distinct parts. Firstly, a scaled version of the grayscale input is passed through a pretrained model, which acts as a Feature Extractor. In the original paper, the model used was Inception-ResNet-v2, however in our baseline implementation we used Inception-v3. Simultaneously, a scaled version of the grayscale input is also passed through an Encoder, which is just a CNN. The first, third and fifth layers

of the Encoder downsample the image by a factor of two, which means the output of the encoder is a $\frac{H}{8} \times \frac{W}{8} \times 256$ tensor. The third component of the network is the Fusion layer, which simply concatenates the output of the Encoder with the output of the Feature Extractor (element-wise). This is then passed through a single Convolution layer that outputs 256 channels. Finally, this tensor is passed through a Decoder, which is also a CNN that upsamples and produces a 2-channel output that matches the size of the input. Each Convolution layer is followed by a ReLU layer.

The Feature Extractor finds high-level global features about the image, such as scene characteristics (indoor, outdoor, season etc.). This information is can be critical in determining appropriate colors for regions of the scene. For instance, there will be a higher likelihood of green in natural scenes during the summer as compared to winter, where brownish hues would be more prevalent. The Encoder's job is to find a low-level condensed feature representation of the input image. This conveys local and spatial information about the image that is helpful during the decoding. The Fusion layer exists solely to combine the high-level and low-level features computed by the Feature Extractor and Encoder. This ensures that the semantic information about the image is uniformly distributed across all spatial regions of the image. And finally, the Decoder is used to upsample (using basic nearest neighbor) and generate two channels that contain the color information for the image.

In the original paper, the loss function applied during training was Mean Square Error (MSE) Loss between the estimated values in the 2-channel output of the network and the ground truth values in the training image's a- and b-channels. The Adam Optimizer is used with an initial learning rate of 0.001.

3.2 Initial Results

The authors of the original paper provided code to run the model. However, their code was written using the Keras and TensorFlow libraries along with the ImageNet dataset. Due to our increased familiarity with Pytorch and some difficulty obtaining the ImageNet dataset, we decided to implement their pipeline from the ground up ourselves using Pytorch and the more easily available COCO dataset, which was quite time-consuming. However, we now do have the baseline implemented and working. Figure 2 shows the decay in loss as number of epochs increases for the train and validation data. Figure 3 shows three examples of an original color image (left) and the recolored image from our network (right). Our network was trained for 15 epochs on 70000 training images. We used Adam optimizer with a decaying learning rate starting at 0.001 and batch size 32. We used an Amazon AWS instance to run our code.

The reason the recolored images have different aspect ratios than the originals is because we have currently forced the input to be a square image. Despite only training for 15 epochs, it's clear that the network is learning high-level semantics such as the sky being blue, snow being white, skin being brownish etc. From these three sample outputs, it is also evident that the network is able to operate across different domains (outdoors, indoors etc.). If we trained for more epochs,

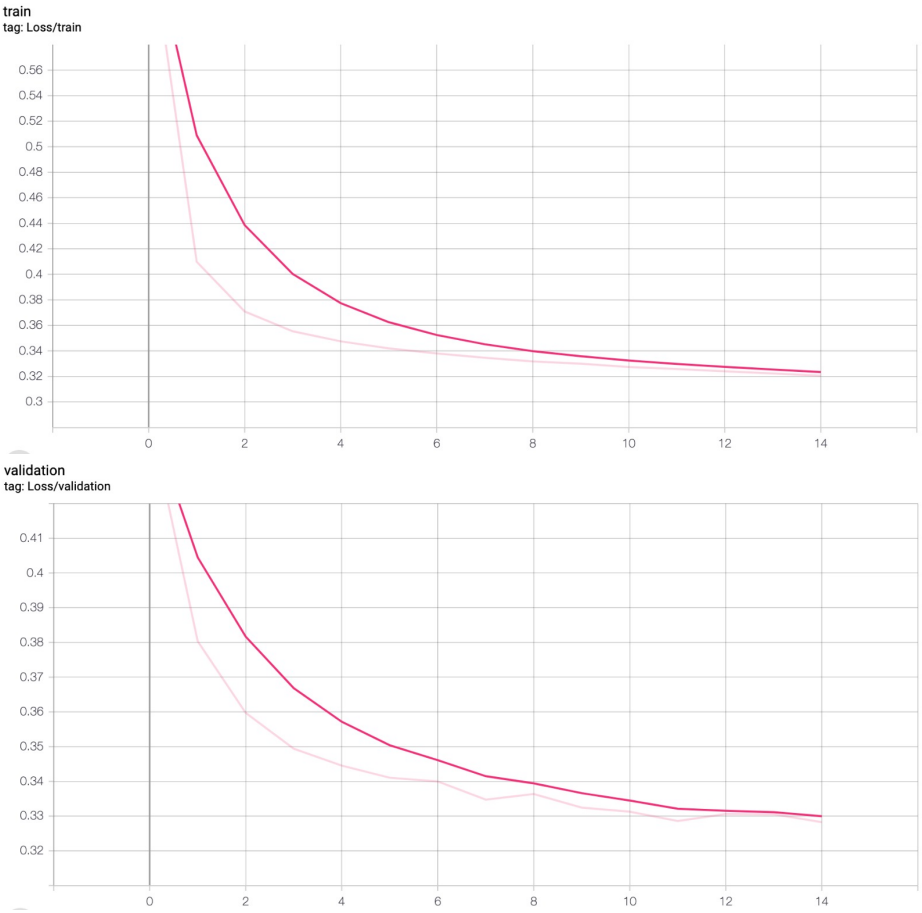

Figure 2: Train Loss (top) and Validation loss (bottom) vs Number of Epochs

we are confident the results would improve further. For instance, the images appear to be relatively desaturated. However overall, the decaying loss clearly indicates that the network is learning the color properties as it is being trained.

3.3 Next steps

Now that we have our baseline model implemented and producing reasonable results, we will attempt to improve it by making some of our own modifications. While we are still in the process of determining what exactly these will be, we have identified broad areas we plan to target. Firstly, we believe that MSE Loss is not well-suited for image colorization. This is because two images can have very different colors (e.g. red car vs. blue car) and hence a high MSE Loss, but both can be equally valid colorizations of a grayscale image. Therefore,

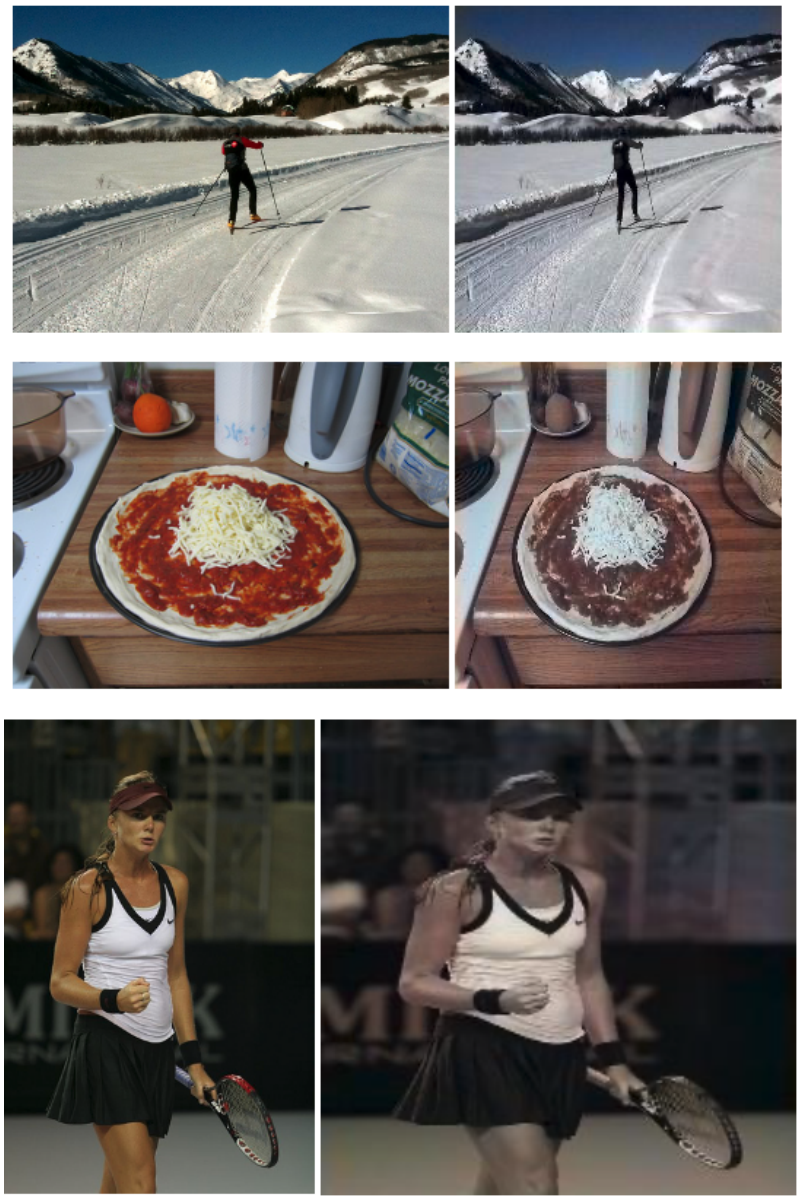

Figure 3: Initial results. Left: original colored image. Right: recolored image from our network.

Table 1: COCO Data Set

Name	Size
Train	70,000
Validation	5,000
Test	1,000

we will explore other loss functions that attempt to model this. Second, we plan to explore different feature extractors to determine if we can extract higher quality semantic information from the images. And thirdly, we will refine the architectures of the Encoder and Decoder to determine if there is room for improvement there (to address the desaturation issue, for example).

4 Data sets

The problem of Image Colorization has been studied in great detail previously and a lot of different data sets have been used depending on the implementations of the task. Also, because of the nature of the problem, it is possible to use pretty much any diverse data set which is readily available. This is because the training requirement of the network is just a grayscale image which can be obtained from any color image. Owing to this, we had a vast choice of data sets to choose from.

To begin with, we considered training our model on the ImageNet dataset as used by Federico Baldassarre et al. in [1]. But, due to availability issues we decided to look elsewhere at other viable options. Also, overhead of training the model on a data set as large as ImageNet had its limitations. We finally settled for the COCO(Common objects in context) data set[11] for a number of reasons. The data set being readily available and also being smaller in size as compared to ImageNet meant that it was feasible to use it. COCO is a large scale data set which is one of the primary data sets used in Image Segmentation as well as Recognition tasks. We train the model on 70,000 train color images. We convert each of these images to grayscale and compare the output color image of our model to the ground truth to calculate the loss. The data set consists of 91 common object categories, each of which have a considerable number of images. It has 40 scene categories which occur prominently alongside the the objects which adds to the diversity of the data set. Apart form the train images, we use a validation data set of 5,000 images and a test data set of 1,000 images as shown in Table 1.

Table 2: Timeline of Remaining Work

Period	Tasks
Now - 11/16	Finish implementing and training baseline model. Conduct user studies to determine performance.
11/17 - 11/23	Identify and implement proposed improvements on baseline model.
11/23 - 11/30	Choose 1-2 most promising improvement techniques and generate final results. Identify edge/failure cases and analyse what worked, didn't work and why. Conduct final use studies.
12/1 - End	Poster presentation and final report.

5 Evaluation

While we can use quantitative meansures such as MSE Loss to quantify the performance of our network, it is not ideal for this particular application. Because colorized image quality can be highly subjective, we believe the best way to assess the quality of our outputs is to survey humans. We will adopt a user study approach similar to that taken by the original paper [1], where we show humans a mixture of genuine color images and artificially recolored images and ask them to determine which ones are genuine and which ones are recolored. We will compute what percentage of humans were fooled by images recolored by our network and use that to compare our modified network to the baseline model. We will also record our average MSE Loss for both networks and compare them. While this loss may not be ideal for images with generic objects, it would probably still be reasonable for images of nature where colors are generally fixed (i.e. grass is green and the sky is blue).

6 Timeline

Table 2 outlines the timeline for the remainder of the semester.

7 Division of Work

Table 3 outlines the team member responsibilities moving forward.

Table 3: Team Member Responsibilities

Team Member	Responsibilities
Bhumi Bhanushali	Try different fusion techniques and determine if the results are better. Help with final report and poster presentation.
Kathan Mehta	Try different loss functions and determine if the results are better. Help with final report and poster presentation.
Avinash Raju	Modify the encoder-decoder architecture and determine if the results are better. Help with final report and poster presentation.
Atulya Ravishankar	Try different feature extractors and determine if the results are better. Help with final report and poster presentation.

References

[1] Federico Baldassarre, Diego Gonzalez Morin and Lucas Rodes-Guirao. *Deep Koalarization: Image Colorization using CNNs and Inception-Resnet-v2.* In: CVPR(2017)

[2] Mingming He, Dongdong Chen, Jing Liao, Pedro V. Sander and Lu Yuan. *Deep Exemplar-Based Colorization.* In: ACM Transactions on Graphics (TOG). Volume 37., ACM (2018)

[3] Welsh T., Ashikhmin M. and Mueller K. *Transferring color to greyscale images* In: ACM Transactions on Graphics (TOG). Volume 21., ACM (2002)

[4] FloydHub : `https://blog.floydhub.com/colorizing-b-w-photos-with-neural-networks/`

[5] Larsson G., Maire M. and Shakhnarovich G. *Learning Representations for Automatic Colorization.* In: Leibe B., Matas J., Sebe N., Welling M. (eds) Computer Vision – ECCV 2016. Lecture Notes in Computer Science, vol 9908. Springer, Cham

[6] Charpiat G., Hofmann M. and Scholkopf B. *Automatic image colorization via multimodal predictions.* In: Computer Vision-ECCV 2008 (2008)

[7] CIE LAB Color Space : `http://docs-hoffmann.de/cielab03022003.pdf`

[8] Iizuka, S., Simo-Serra, E. and Ishikawa H. *Let there be color!: joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification.* In: ACM Transactions on Graphics (TOG) 35(4) (2016)

[9] Richard Zhang, Phillip Isola, Alexei A. Efros. *Colorful Image Colorization.* In: Computer Vision-ECCV 2016 (2016)

[10] Vivek George Jacob and Sumana Gupta. *Colorizing of Grayscale Images and Videos using a Semi-Automatic Approach.* In: 16th IEEE International Conference on Image Processing (ICIP) (2009)

[11] Lin TY. et al. *Microsoft COCO: Common Objects in Context.* In: Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) Computer Vision – ECCV 2014. Lecture Notes in Computer Science, vol 8693. Springer, Cham