Image Colorization

Zimri Leisher, Morgan Martin

I. Introduction

Color provides important context to the viewer of an image. Time of day, weather and lighting can all be significantly harder to identify when only looking at a black and white image. There are also qualitative things a colorization provides: immersion and emotion, both important for art and photography, can often be conveyed partially through color. Thus, it would be beneficial to be able to convert black and white images, produced perhaps by cameras before color picture technology was invented, to color images. We present a deep learning model using a convolutional neural network which solves the problem in a limited domain: colorizing pictures of different types of fruits.

II. Previous Solutions

Before convolutional neural networks were used, previous approaches often used manual human annotation. This was time-intensive and laborious. Therefore Zhang[1] came up with a different approach that used Convolutional Neural Networks to mimic the colorization of grayscale images. They did this by converting the ImageNet dataset from RGB colorspace into LAB Color Space. Specifically, by using the L channel (lightness intensity channel) to predict the A (green-red) and B channels (blue-yellow). Afterward, they combined the L channel with the predicted A and B channels. Lastly, they converted the LAB images back into RGB.

III. Dataset

Our dataset, the Fruits360 dataset[2], consists of 90,430 images. Out of a possible training set size of 67,692, we chose 15,377 images, and for the test set, 5,135 images out of 22,688. We included 3 different fruits (apple, cherry, pear) and our image size was 100 x 100 pixels. The reason we cut down on the overall dataset was to reduce the training load to ensure that we didn't overrun Google Colab usage limits.

IV. Proposed Method

From the dataset, we load the images and convert them with CV2 from the BGR colorspace to the LAB colorspace. Then, we normalize the image pixel values to -1..1, and split the images

[1] BEGUERADJ, Billal, et al. "Black and White Image Colorization with Opencv and Deep Learning." PyImageSearch, 17 Apr. 2021, https://www.pyimagesearch.com/2019/02/25/black-and-white-image-colorization-with-opencv-and-deep-learning/.
[2] https://www.kaggle.com/moltean/fruits

into just the L (lightness) component, which is the input to the model, and the AB (color) component, which is the output of the model.

We define a multi-layered convolutional neural network with an autoencoding structure, constricting the flow of data in the middle so that the first half encodes the input into high-level features, and the second half takes the high-level features and turns them into low-level output. We use Conv2D and Upsampling2D layers to accomplish this, with all neurons using the rectified linear unit activation function and the HE Normal kernel initializer. We compile the model, seeking to lower the mean squared error with the Adam optimizer with a carefully tuned learning rate.

Because of limits on our training time and processing power, we chose a relatively small network (17k parameters) in addition to the already mentioned smaller dataset. This produces a training time of around 10 minutes per epoch, and we ran the model for 1 epoch. Naturally, this is far too few epochs, but due to time constraints it was necessary.

The structure is as follows:

1. 5x5 convolution, 16 filters
2. 5x5 convolution, 16 filters
3. 2x2 max pooling
4. 5x5 convolution, 12 filters
5. 5x5 convolution, 9 filters, stride 2
6. 2x2 upsampling
7. 3x3 convolution, 32 filters
8. 2x2 upsampling
9. 3x3 convolution, 2 filters

V. Evaluation Method

Keras provides a quantitative evaluation for the model, which we use on the test dataset. However, this number has little meaning on its own, as it depends on the format of the data and without comparison can't really help us decide if our model is functional or not.

In order to remedy this problem, we gave 10 pairs of images, in grayscale and color, to 4 people and asked them to rank how good the colorization was from 0-10. 5 of the pairs of images were controls, and 5 of the pairs of images were generated by our model.

VI. Results

The quantitative evaluation function resulted in a loss of 0.0151 on the test dataset. On average, for the qualitative testing, the control images scored an 8.55 out of 10, and the generated images scored a 3.225 out of 10.

Clearly, the network is not performing at a level to where it would be hard to distinguish between reality and its results. However, as we have mentioned above, time constraints and processing power constraints have forced us to keep the model small, and so we did not expect to be very successful.

VII. Discussion

a. Future Ideas

If we had more time we would explore other ideas such as using image colorization on motion pictures and videos. This would be a cool thing to explore for families with home videos or even something that production companies could use to convert black and white movies into color. Additionally, if we had more time we would create more data and use more epochs

b. Applications

Currently, image colorization is broadly used to add style and uniqueness to photos by "filtering" them. However, in the context of our project we wanted to use image colorization to restore the color of images taken in black and white. This usually applies to old photos that were taken before the 20th century before color photography existed.