IEOR 242 Final Project Report
Image Colorization

Louis Tilloy, Georgios Prelorentzos, Chao Zhang, Parth Sanghvi, Emma Leiner

05/17/2019

Contents

1 Introduction

Image colorization is the process that, given digital images that are grey-scale, a colored version will be produced automatically by machine learning algorithms. The algorithm would guess the most likely color choice pixel-wise based on certain characteristics of the original image. This project is aiming to build a tool which predicts the most probable colorized version of an image given its gray-scale pixel intensity, as Figure 1 illustrates. We take full advantages of the strengths of convolutional neural network and the nature of CIELAB color-space to build our model, and our conceptual approach and machine learning model construction was greatly inspired by the work of Richard Zhange et al., 2016 [2], and many ideas were learnt from their work.

1.1 Project Motivation

Consider pictures of significant historical events; family photos generations back; or records from large national archives, many of which were taken or stored in black-and-white hard-copies, those valuable records are extremely susceptible to abrasion and oxidation. With current efforts on digitization of old archives, we want to take a step further, with the help of machine learning techniques, to build a automated colorization tool which can add a extra layer of truthfulness and richness to each story behind those precious moments.

Figure 1: Example of historical application of image colorization.

2 Data Collection and Processing

To build the convolutional neural network, a very large number of colored images were needed for the algorithm to be properly trained. At the same time, the variety in terms of image content is crucial to our project's success since the model will try to guess the most probable color outcome based on how objects are usually presented in images. Given the currently available online database, ImageNet is the most suitable option for this project due to its size and quality. At the same time CIFAR10, a smaller image database, was also used for local trials during early stages.

2.1 Database Description

ImageNet [1], an research project built to provide an easily accessible online image database for researchers, contains more than 14 millions images of varies sizes in RGB format and covers more than 20,000 different categories. The database is organized by the WordNet hierarchy [1], hence each category is built around a particular concept from WordNet, rather than a specific word, since there may exist many words that describe the same concept. According to the official document of ImageNet, all images went through quality-control process and were annotated manually by humans. Besides ImageNet, we used CIFAR10 during local model construction and trials. Collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, this database has 60,000 colored images in 10 different classes, all of which stored in RGB color-space with consistent size 32x32. A much lower size, 163MB for the entire database, meant it was ideal to work with locally for early trials and experiments.

2.2 Data Processing

Most of the processing efforts only apply to ImageNet since CIFAR10 already come with consistent size. while the file quality and quantity from ImageNet provided us with a solid database for training, it also posed few challenges in terms of data preparation.

2.2.1 Resizing

Firstly, since files under ImageNet come with different sizes and aspect ratios, the first step of processing was to resize them to the same size that could be used as model inputs. The converted images had size 256x256, the size was chosen to be consistent with [2] to have a reference for comparison.

2.2.2 Color-space Conversion

Secondly, the default RGB color-space of ImageNet is not useful for the purpose of this project, since 3 different color channels are at present, prediction will be much more trickier to implement. Hence a color-space conversion was needed.
CIELAB, also known as simply lab, is better suited for our model. In the lab color coding scheme, L corresponds to the luminescence of a image, i.e. the gray-scale intensity, which we use as feature; *(a, b)* correspond to the two color channels: where a encodes a value from red to green, and b encodes a value between blue and yellow. While the range for ab channel differs from on specific implementation, they often run in the range of -100 to $+100$ or -128 to $+127$, as Figure 2 illustrates below:

On a more direct presentation, Figure 3 below presents a decomposition of image as to how the lab color-space store image information:

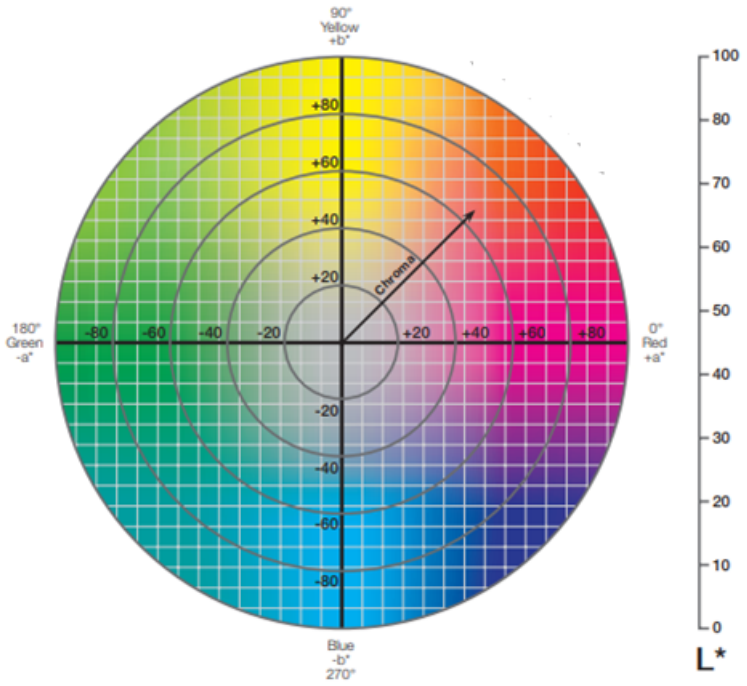

Figure 2: Representation of Lab color space. The x axis represents the a channel. The y axis represents the b axis.

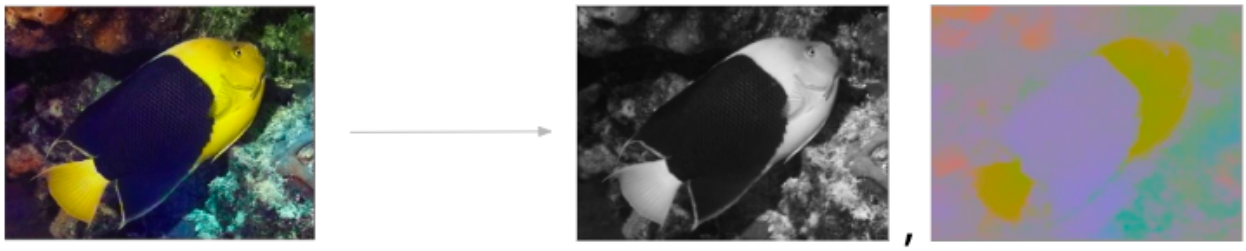

Figure 3: *On the left*: all three channels L, a, b are combined into a colored image. *On the right*: Decomposition of L channel (left) and *(a, b)* channels (right).

3 Methodology

3.1 Model Training Process

To build and train the model, due to the computing power required to process the amount of images, local solutions are infeasible. Hence the only option is to do so remotely through Amazon Web Services (AWS). We construct the model in the cloud using Keras Sequential API and TensorFlow 1.3 as the main tools, under AWS Accelerated Computing p3 instance equipped with a single NVIDIA V100 Tensor Core GPU. We also conducted some experiments with the latest TensorFlow 2.0 API as well.
All processing steps mentioned in previous section were carried out in the cloud after data directly loaded from ImageNet to AWS instance. A batch of 40 images were transferred and fed into the model, for more details on the model and reproducing our current results, please refer to appendix the accompanied code.

3.2 Convolutional Neural Network Architecture

The convolutional neural network structure illustrated in Figure 4 below is greatly inspired by *Colorful Image Colorization* [2] (Richard Zhange et al., 2016). Each layer is one convolution followed by a *relu* activation, with batch normalization inserted every 2 or 3 layers. Finally, there are no *Max Pooling* layers which is motivated by the article. From a high level what we want is to retrieve the information of inputs, hence irreversible information aggregation would be undesirable.
As to how we define the actual problem, We tried to implement the model as both regression and classification. Details of different methods are listed below. During earlier stage of model construction, We also tried 2 other architectures for experimentation based on the methods mentioned below, yet much more smaller and simpler compare to the full model.

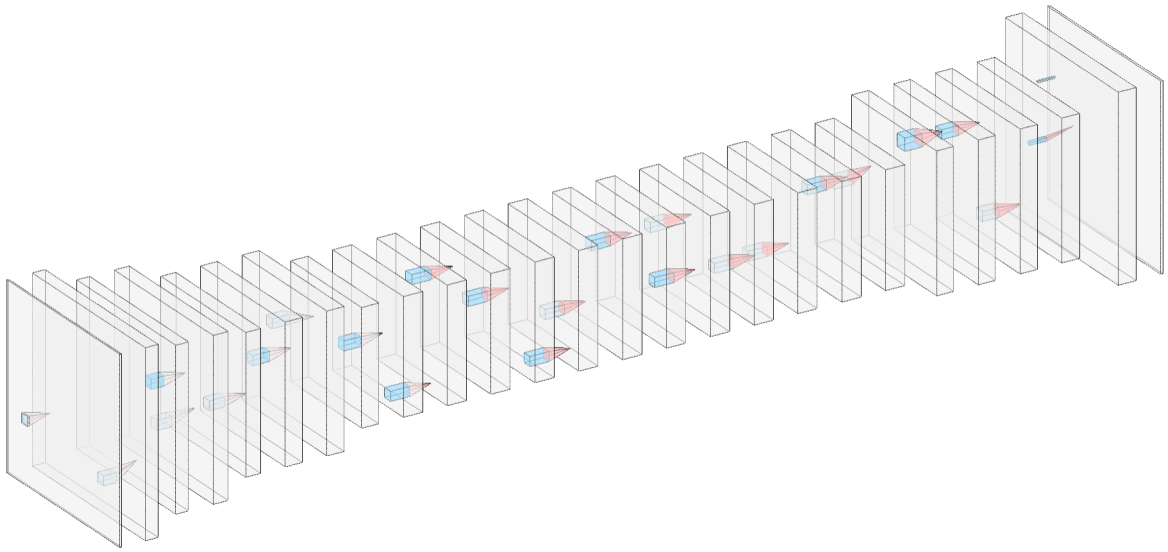

Figure 4: Convolutional Neural Network Architecture for Classification

3.3 Regression Approach

Under regression setting, we apply a regression to both channels a and b. This model uses a MSE loss function, which does not take into account for multiple possible modes and thus favors "safe" predictions. Indeed if 2 colors are plausible, the MSE loss will favor an average of these 2 colors, leading to unsaturated tones, close to sepia.

The loss to minimize is the following:

$$\mathbf{L} = \sum_{i=1}^{256}\sum_{j=1}^{256} \frac{(a_{ij} - \hat{a}_{ij})^2 + (b_{ij} - \hat{b}_{ij})^2}{2 \times 256 \times 256} \tag{1}$$

3.4 Classification Approach

To address the problem of unsaturated color predictions, we discretize the color space and mapping it to integers, then try to solve the colorization problem as a classification problem. In this case, we use a cross-entropy loss function and predict a probability distribution over the possible colors.Channels a and b are discretized separately, so probability distributions are computed separately for colors of channel a and channel b. This

approach helps the model being more "bold" in its predictions and render more colorful images since it does not suffer from the mode ignorance problem the regression model suffered from.

3.4.1 Classification With 1D Discretization

The loss to minimize is the following:

$$\mathbf{L} = \sum_{i=1}^{256} \sum_{j=1}^{256} \sum_{k=1}^{20} \frac{a_{ijk} \log(\hat{a}_{ijk}) + b_{ijk} \log(\hat{b}_{ijk})}{2 \times 256 \times 256 \times 20} \tag{2}$$

where a_{ijk} is the probability of the pixel in position (i, j) to have an a value that belongs to the bin number k. And similarly for b_{ijk}.

3.4.2 Classification With 2D Discretization

Our last approach was to model this problem as a classification problem by discretizing the *(a, b)* space as pairs of both channels. This approach was the one used in the research paper we based our model on, which seemed to be the most promising approach for us. As recommended in the paper, we used a total of 313 classes, each corresponding to a pair of values *(a, b)*. This approach allows both channels to be related and predict colors that are more relevant.

The loss to minimize is the following:

$$\mathbf{L} = \sum_{i=1}^{256} \sum_{j=1}^{256} \sum_{k=1}^{316} \frac{p_{ijk} \log(\hat{p}_{ijk})}{256 \times 256 \times 316} \tag{3}$$

where p_{ijk} is the probability of the pixel in position (i, j) to have (a, b) values that belong to the bin number k.

4 Results and Discussion

Here we present the current results from the three models described in the previous section. A unique part of this project is that, since the goal is to produce the most probable and believable colorization, common model evaluation metrics would not be suitable. An idea is to ask human as judges to see if one is able to tell which one is 'faked' by the model. However, at this stage, as the reader will discover in this section later, the current results we have are far from ideal for a proper evaluation process. Therefore as of now we will continue concentrating on perfecting the model, rather than looking for evaluation metrics.
Below, Figure 5 below captures the latest attempt of model training with respect to the number of epochs. The loss is noticeably high and doesn't decrease much even with more than 900 epochs:

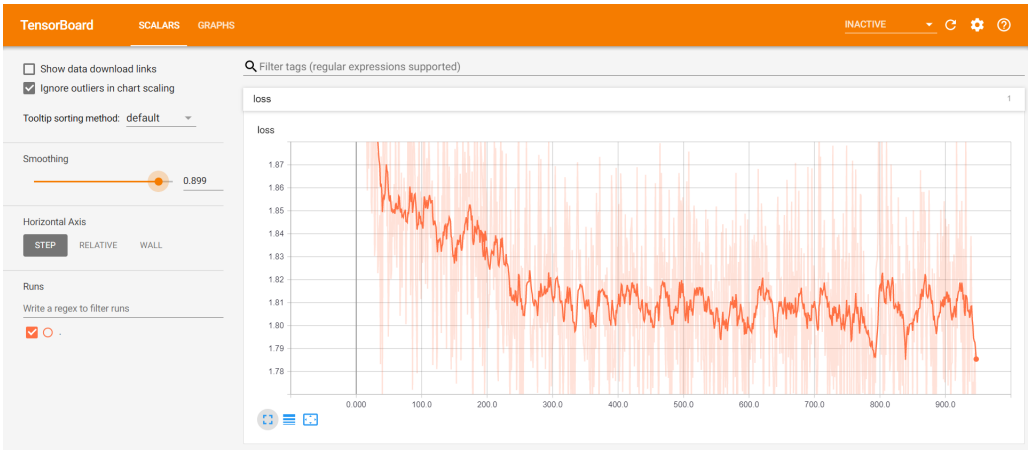

Figure 5: Training Loss w.r.t numbers of Epoch

The images illustrated below are our current results on CIFAR10. We did not perform train/test split for this batch of results since the model is still noticeably far from ready, hence obtaining out of sample results would not provide any useful information regarding model performance.
In general, results below show that even though the model could recognize the object/background in a image, and trying to assign a corresponding color, somehow it failed to be more specific pixel-wise hence it just assigned a single color to large amount of pixels without successfully distinguish different objects.

4.1 Classification Results

4.1.1 With 2D bins

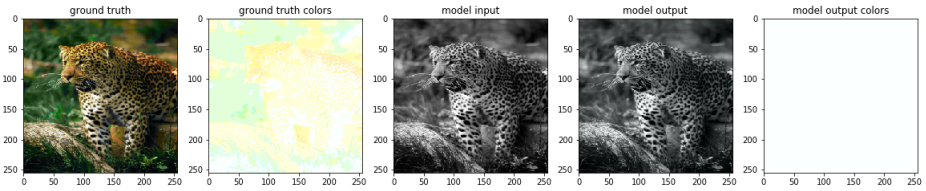

Figure 6: Classification 2D

The preliminary results of the classification model with 2D bins are displayed above. We notice that the model output is not in fact a colored image but just a similar rendition to the gray scale image itself. It is worthy to note that the output image is not exactly the same as the gray scale image that was used as input to the model, a very subtle color was assigned but to the entire image.

4.1.2 With 1D bins

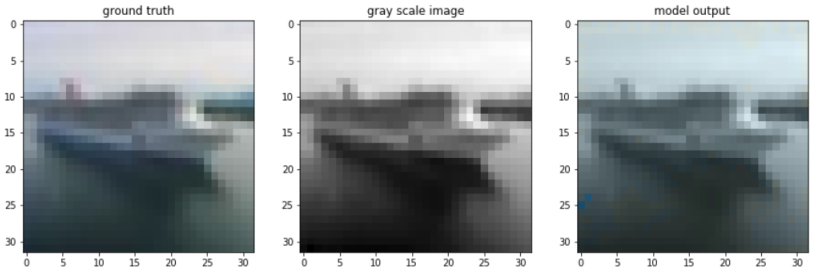

Figure 7: Classification 1D

The above result of classification using 1D bins shows that the model could successfully recognize water and tried to assign a corresponding color. Yet, same as the 2D bins model, the issue is that the entire output was rendered in the same color. Which means that, again, the model failed to distinguish different objects and their boundaries.

4.2 Regression Results

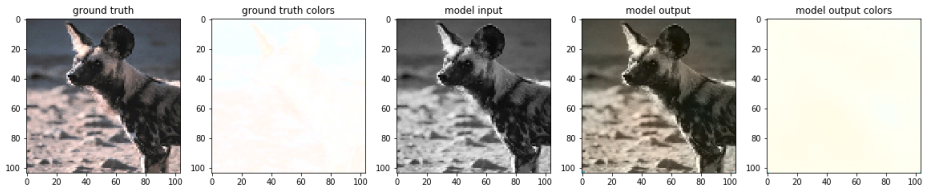

Figure 8: Regression Results

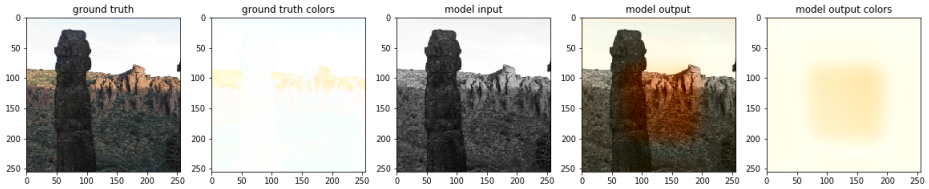

Figure 9: Regression Results

The images displayed above represent the output of the regression model along with the ground truth as detected by the model as well as the final output. The processing images allow us to understand the instinctive behavior of the model as opposed to just looking at the output. We notice that again the model failed to distinguish different objects and their boundaries, hence only output few different colors.

4.3 Discussion and Potential Project Impact

Due to the time consuming nature of training process, each trial requires long time to complete and therefore, we were unable to perform many different modifications of the model to catch possible errors or improve current results. One idea that we have for next step is to isolate only one category from ImageNet, possibly human portrays, and train the model on only this category with a large sample.

The potential benefits of this approach are two folds: first, training on just one category would save us lots of time therefore the cycles of trials and modifications could be much more frequent compare to the current training process; second, human portrays are relatively simple in terms of image structure yet complex enough with many different possible colors for eyes, skins, or hairs. Hence acts as a decent test for model performance.

We believe with more time and a more perfected model, this project could serve as a prototype in helping large national archives, museums or educational institutions during their digitization process; or as a archetype for applications facing normal customers, helping them preserving more vivid memories of loved ones and family members. The eventual goal for us is to construct an easily implemented tool to properly colorize black-and-white images, through which not necessarily helping business gain insights on the market, but to bring important historical events back to life, and smiles on people's faces.

5 Appendix

Layer (type)	Output Shape	Param #
conv2d_21 (Conv2D)	multiple	80
conv2d_22 (Conv2D)	multiple	584
batch_normalization_v2_7 (Ba	multiple	32
conv2d_23 (Conv2D)	multiple	1168
conv2d_24 (Conv2D)	multiple	2320
batch_normalization_v2_8 (Ba	multiple	64
conv2d_25 (Conv2D)	multiple	4640
conv2d_26 (Conv2D)	multiple	9248
conv2d_27 (Conv2D)	multiple	9248
batch_normalization_v2_9 (Ba	multiple	128
conv2d_28 (Conv2D)	multiple	18496
conv2d_29 (Conv2D)	multiple	36928
conv2d_30 (Conv2D)	multiple	36928
batch_normalization_v2_10 (B	multiple	256
conv2d_31 (Conv2D)	multiple	36928
conv2d_32 (Conv2D)	multiple	36928
conv2d_33 (Conv2D)	multiple	36928
batch_normalization_v2_11 (B	multiple	256
conv2d_34 (Conv2D)	multiple	36928
conv2d_35 (Conv2D)	multiple	36928
conv2d_36 (Conv2D)	multiple	36928
batch_normalization_v2_12 (B	multiple	256
conv2d_37 (Conv2D)	multiple	36928
conv2d_38 (Conv2D)	multiple	36928
conv2d_39 (Conv2D)	multiple	36928
batch_normalization_v2_13 (B	multiple	256
conv2d_transpose_3 (Conv2DTr	multiple	32800
conv2d_transpose_4 (Conv2DTr	multiple	9248
conv2d_transpose_5 (Conv2DTr	multiple	9248
conv2d_40 (Conv2D)	multiple	33
conv2d_41 (Conv2D)	multiple	33

Figure 10: Convolutional Neural Network Layers from Keras

References

[1] J. Deng et al. “ImageNet: A Large-Scale Hierarchical Image Database”. In: *CVPR09*. 2009.

[2] Richard Zhang, Phillip Isola, and Alexei A. Efros. “Colorful Image Colorization”. In: *CoRR* abs/1603.08511 (2016). arXiv: 1603.08511. URL: http://arxiv.org/abs/1603.08511.