Deep Learning Image Colorization

Daniel Dong
University of Michigan
dandong@umich.edu

Oliver Wang
University of Michigan
oliveraw@umich.edu

Abstract

Given a grayscale image as input, our paper attempts to recreate the results seen by Zhang et al. for hallucinating a plausible colorization. To accomplish this we implemented both encoder/decoder and GAN architectures and trained our models with L2 and multinomial and cross-entropy loss. We evaluate our models using PSNR and MSSIM metrics between original and generated colorizations.

1. Introduction

Color has always been an essential feature of human vision and contributes much to our perceptual and aesthetic understanding of the world. Though the first cameras were invented in 1816, it wasn't until 1932 that the first color cameras were invented, leaving over 90 years of photos in the black-and-white era.

Traditionally image colorization has been done by hand, with painstaking effort to colorize black and white photos and videos. However, in the modern-day, this task can be effectively handled by deep-learning computer vision models which accelerate and automate this task. These models take in a black-and-white image and output a plausible colorization for the photo. This process is fascinating as colorization can be done using different methods like regression and classification, and has the potential to augment lost historical moments with the addition of color.

In relation to other work, colorization models must learn the features of photos including shading, texture, and segmentation. The parameters obtained from training colorization models can then be transferred as starter parameters to other projects and thus provide a solid foundation for pretraining in other computer vision tasks.

In our paper, we attempt to re-implement a colorization architecture as described by Zhang et al. [5] while documenting our process, results, and conclusions for future researchers.

2. Related Work

Initial attempts at colorization included an approach called Scribble-based colorization introduced by Levin et al. [4] This method involved the user providing colorful scribbles on a grayscale target image and the color information was then propagated onto the rest of the image using least-squares optimization. Then Gupta et al. [2] introduced an approach that involved feature mapping at superpixel resolution. Colorization was achieved by mapping from one image to another with similar features. Early machine-learning colorizers used regression to learn, and Cheng et al. [1] introduced using L2 regression loss to train a deep-learning colorizer network. They were able to colorize photos, but their images appeared desaturated due to the model predicting the safest colors.

Then Zhang et al. [5] came along introducing colorization as a classification problem. They developed a solution to the desaturated colors of previous regression models by treating each pixel as a classification problem by placing them into quantized color bins in the AB color space (CIELAB). They also introduced a rebalancing term in the cross entropy loss which promoted vibrant and rare colors to be predicted. Larsson et al. [3] introduced a similar concept training their model to predict per-pixel color histograms. The intermediate output could be used to automatically generate a color image and it outperformed existing colorization methods at the time.

Zhu et al. [6] took the concepts of GANs (Generative Adversarial Networks) and applied them to the colorization task. GANs are a machine learning model where two neural networks (a generator and a discriminator) compete with one another in their predictions. The generator creates a fake image that tries to fool the discriminator who picks between a real and a fake image. Zhu et al. used a CycleGAN mapping inputs to outputs to inputs repetitively to create a cyclic structure.

3. Methods

3.1. CIELAB Color Space

Related work in colorization has typically used the CIELAB format for images, in which RGB is converted into L (lightness) and A/B (red-green/yellow-blue) channels. The reason for this conversion is because 2-dimensional prediction problems have more constrained outputs and thus tend to show improved results over tri-channel prediction. Our models take the L channel of an image as input and attempt to recreate the A/B channels.

Furthermore, we followed the procedure of separating the A/B color space into discrete bins as described by Zhang et al. [5] In doing so, we can transform colorization into a multinomial classification task as opposed to predicting A/B values on a continuous spectrum. CIELAB A and B values each span the interval $[-110, 110]$. We split each spectrum into 22 bins of size 10 and lay them onto the A/B plane, leaving us with $22 \times 22 = 484$ bins. However, only 313 of these color bins are visible to the human eye—these visible bins are referred to in the literature as ”in-gamut”.

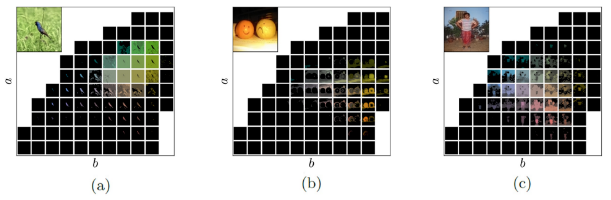

Figure 1. Examples visualizing the 313 A/B bins as well as 3 sample predictions from the model.

3.2. Original Model

We decided to follow an encoder/decoder style architecture in which resolution of the input image is decreased through strided convolution and subsequently increased to generate the final colorization. The goal is to learn a meaningful color represenation as the dimensionality of the image is reduced, as well as learning how to translate this representation into a higher resolution through transposed convolution. Our model is based heavily on that shown by Zhang et al. [5], with modifications to stay within our GPU usage limits. Their model is as follows:

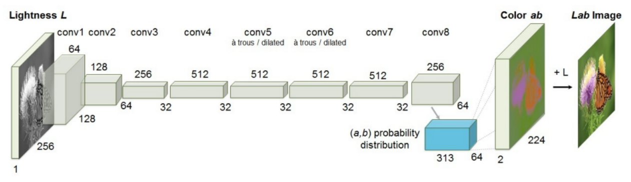

Figure 2. Original model shown in Colorful Image Colorization

Their described model takes in the L channel of an image as input and attempts to hallucinate the A/B color channels as output.

3.3. Baseline Regression

Our baseline model uses regression to predict color targets. The model is trained using L2 (Mean Squared Error) loss, which is calculated by taking the per-pixel difference, then squaring and summing the result over all pixels. We define the ground truth label $Y \in \mathbb{R}^{H \times W \times 2}$ and the prediction $\hat{Y} \in \mathbb{R}^{H \times W \times 2}$. The L2 loss is as follows:

$$L_2(Y, \hat{Y}) = \frac{1}{2} \sum_{h,w} ||Y_{h,w} - \hat{Y}_{h,w}||_2^2$$

The model shown in Figure 2 used 8 convolutional layers, each consisting of a 2-strided convolution with additional 1 (or more) 1-strided convolutions, and a ReLU following each. We have modified the architecture by removing many of the secondary convolutions in each layer in order to stay within GPU usage limits.

Our encoder consists of 3 strided convolutions which downsample the image, then 2 more convolutional layers which maintain the same resolution. Features are learned through these convolutional blocks, and are then upsampled through 2 strided deconvolutions and an upsampling to achieve the final output. The innermost layers of the model operate at a lower resolution (12x12 vs. 32x32) compared to the original Zhang model.

We decided to introduce more convolutional layers at the end of the network because our initial output tended to show colors in square patches. We believed this was an artifact of upsampling and that increasing the resolution through convolutions would create a smoother colorization. We have changed the x4 upsampling into an additional deconvolution with stride 2 and upsample x2 to achieve the original size.

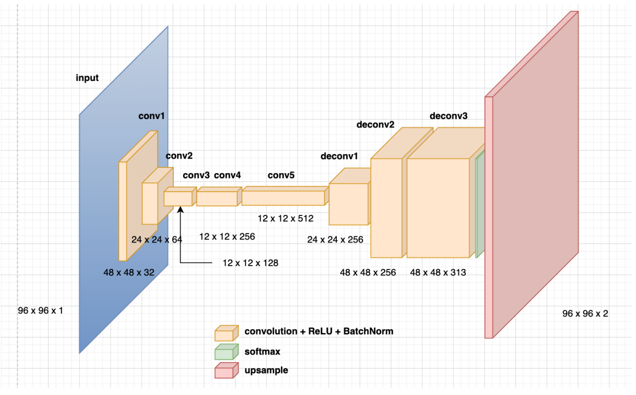

Figure 3. A smaller version of the original model: the A/B channels are outputted at the final layer and trained using L2 loss, with an additional layer of larger resolution convolutions before the final output.

3.4. Classification

Our second model treats the problem as multinomial classification, where for a given input $X \in [0,1]^{H\times W\times 1}$, a mapping is learned to $\hat{Z} \in [0,1]^{H\times W\times Q}$, where $\hat{Z}$ is the predicted probability distribution over the 313 possible color bins. The input X refers to the L channel of an image, while the output $\hat{Z}$ gives the probabilities that a given pixel corresponds to a specific color bin. The value $\hat{Z}$ is compared against the ground truth Z, which is a "soft encoding" of the one hot vector corresponding to the correct bin $\in Q = 313$.

This soft encoding is suggested as an alternative to one-hot encoding. For each A/B pixel in the input image, the 5 color bins with the closest Euclidean distance are selected and weighted according to their Gaussian distance with $\sigma = 5$. These 5 nearest neighbor probabilities are then considered the ground truth labels, Z, for multinomial classification.

The model is trained with multinomial cross entropy loss L_{cl} defined as:

$$L_{cl}(Z,\hat{Z}) = -\sum_{h,w} v(Z_{h,w}) \sum_{q} Z_{h,w,q} log(\hat{Z}_{h,w,q})$$

The rebalancing term, $v(Z_{h,w})$, is introduced to increase the frequency of rare colors.

$$v(Z_{h,w}) = w_q*\text{, where } q* = \arg\max_q Z_{h,w,q}$$

and

$$w \propto ((1-\lambda)\tilde{P} + \frac{\lambda}{Q})^{-1}, \mathbb{E}[w] = \sum_q \tilde{P}_q w_q = 1$$

The smoothed empirical distribution of color bin rarity $\tilde{P} \in \Delta^Q$ is obtained by estimating the empirical probability of colors in the quantized A/B space $P \in \Delta^Q$ from the full ImageNet training set and smoothing the distribution with a Gaussian kernel. This distribution is then mixed with a uniform distribution with weight $\lambda \in [0,1]$, then the reciprocal is taken so the weighting factor is 1 on expectation. Zhang et al. [5] found that values of $\lambda = \frac{1}{2}$ and $\sigma = 5$ worked well as a weighting for rare colors. In our implementation, we attempted to use their pre-generated weights to rebalance our ground truth labels, but ran into issues with the efficiency of the ground truth generation (see below).

We initially hit a roadblock when trying to re-implement the cross entropy loss on a 2 channel output $\in \mathbb{R}^{H\times W\times 2}$. The one-hot encoded cross entropy we were familiar with would have a prediction $\hat{Z} \in \mathbb{R}^{H\times W\times Q}$, with one-hot ground truth labels $Z \in [0,313)^{H\times W}$. For this reason, we chose to convert the 2 channel model output into a 313 channel output, predicting probabilities for each pixel being in each of the 313 bins.

In addition, we had difficulties efficiently generating ground truths with the 5 nearest neighbor bins to the original pixel color. Based on the Pytorch implementation of cross entropy loss, the ground truth could be formatted as either $[0,313)^{H\times W}$ with a one-hot bin label per-pixel, or as $\mathbb{R}^{H\times W\times Q}$, with Q probabilities per-pixel. We attempted to generate a mapping from the original A/B pixel space $\mathbb{R}^{H\times W\times 2}$ to the nearest neighbor space $\mathbb{R}^{H\times W\times 5}$ and then to the ground truth $\mathbb{R}^{H\times W\times 313}$, where we would then apply the color rebalancing function $v(Z_{h,w})$. However, we were not able to do this efficiently enough to effectively train our model. Our final model was trained using only the one-hot encoded cross entropy loss, without the class rebalancing included.

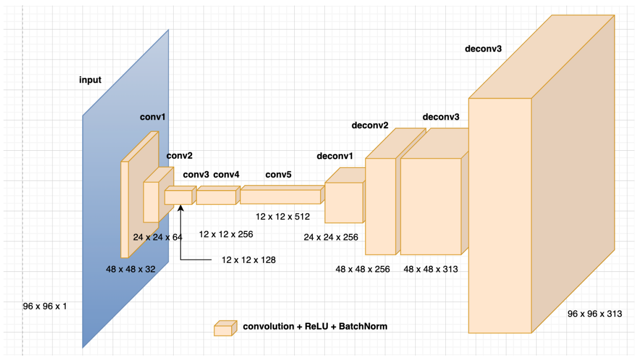

Figure 4. Our modified model for bin prediction: 313 channels are outputted at the final layer and trained using cross-entropy loss.

4. Experiments

4.1. Data

We trained our model on a subset of the STL-10 dataset. The full set contains 500 training and 800 test images per class, with 100,000 unlabeled images for unsupervised learning. We used a 10,000 image subset of the unlabeled images for our model, with 8,000 images for training and 2,000 for testing.

4.2. Models

We evaluated our results on both our L2 regression model and our cross entropy classification model. The architectures and details of both are further described in section 3. We used the Adam optimizer during training, with hyperparameters as described below.

4.3. Hyperparameters

We used the same hyperparameters for training across all two models.

Hyperparameters	
Batch Size	64
Epochs	40
Adam - lr	0.001
Adam - β_1	0.9
Adam - β_2	0.999
Adam - Weight Decay	0

Table 1. Hyperparameters used.

4.4. Quantitative Evaluation

We evaluated our models based on 3 quantitative metrics. The first is MSSIM (Mean Structural Similarity Index Measure), which evaluates the similarity of two images based on key perceptual features including luminance, constrast, and structure. SSIM scores are in the range $[0, 1]$, where 1 is a perfect match and 0 indicates dissimilarity. The second is PSNR (Peak Signal-to-Noise Ratio), where a higher PSNR value indicates higher signal/noise quality between the modified and original image. The third is LPIPS (Learned Perceptual Image Patch Similarity), which runs the two images through a net (VGG in our experiments) and computes the similarities between the activations on two corresponding image patches. A lower LPIPS score indicates that the images are perceptually similar.

Evaluation Metrics			
Model	MSSIM	PSNR	LPIPS
L2	0.920	22.099	0.148
Cross Entropy	0.859	21.361	0.152
Control (Pure L channel)	0.935	21.797	0.146

Table 2. Our evaluation metrics for our two models.

Our L2 model surpassed the control in only the PSNR metric, while our cross entropy model performed worse on all three metrics. We believe that this may be a result of not training the models for long enough. As shown by the images in the qualitative evaluation section, the model does generate more colorful images in both cases. However, the segmentation on the image itself can sometimes be messy and may contribute to lower structural/perceptual similarity scores.

We believe these metrics may not have accurately evaluated our model as none of the three explicitly measure color differences. The black and white image is likely considered more structurally similar to the original image than our predictions, simply because of a lack of ”negatives” which detract from the similarity scores.

4.5. Qualitative Evaluation

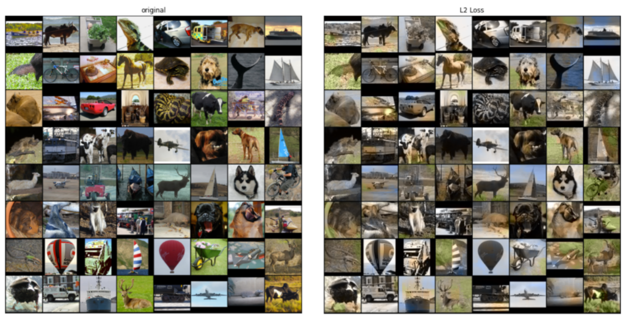

Figure 5. A comparison of the original images vs the images trained with our L2 regression model

As you can see, the colorization produced by our regression model tend towards green, blue, and brown colors, likely because the L2 model tends to predict the ”safest” result and there is a high distribution of these colors throughout our training sample.

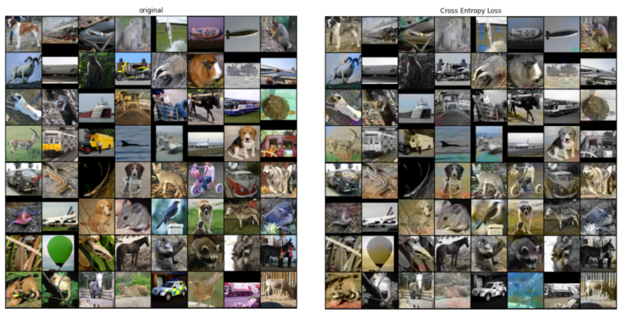

Figure 6. A comparison of the original images vs the images trained with our cross-entropy classifier model

The colorization produced by our cross entropy model has more vibrancy, with visible reds and turquoise colors. However, on close inspection, we saw that the model did not perform segmentation well and colors were frequently predicted outside object boundaries.

5. Conclusions

5.1. Results & Discussion

Reproducing the work of Zhang et al [5] wasn’t the easiest task. While Zhang’s model was available to replicate, we had to make many simplifications to the model due to GPU and time constraints. Furthermore, we found it difficult to reimplement the multinomial cross-entropy loss function as well as the binning methods as discovered in the paper. However, we believe the process of re-implementing the model and producing our own training loop helped give us a much better understanding of neural networks and allowed us to dig deeper into the content that we've learned. While our trained models still had certain limitations, we believe that with more computing resources and advice on implementation details, we could achieve results similar to those seen in other works.

References

[1] Zezhou Cheng, Qingxiong Yang, and Bin Sheng. Deep colorization. *CoRR*, abs/1605.00075, 2016.

[2] Raj Gupta, Alex Chia, Deepu Rajan, Ee Ng, and Zhiyong Huang. Image colorization using similar images. pages 369–378, 10 2012.

[3] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. *CoRR*, abs/1603.06668, 2016.

[4] Anat Levin, Dani Lischinski, and Yair Weiss. Colorization using optimization. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, page 689–694, New York, NY, USA, 2004. Association for Computing Machinery.

[5] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. *CoRR*, abs/1603.08511, 2016.

[6] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, abs/1703.10593, 2017.