Colorizing Paintings using Deep Convolutional GANs

Wes Peisch, Luca Pistor, and Samarpreet Singh Pandher
Stanford University

March 2, 2021

1 Introduction

In recent years there has been increasing interest in the area of using neural networks to colorize grayscale images. These attempts have been motivated by the many and varied applications of image colorization, ranging from historical image restoration to speeding up the workflow for animation studios. We attempt to train a series of neural network architectures to color sketches and paintings, in an attempt to find the best architecture and set of hyperparameters for our data. We chose sketches and paintings as a focus area because existing models focus on digital photographs, which capture objects objectively, without the detail and texture of a painting. By focusing on hand-created works, we can create a model that incorporates the nuances of artwork into its recoloration process.

2 Dataset

Our dataset is a collection of works from 30 artists who were alive in the 19th and 20th centuries. The dataset comes from Kaggle, titled "Best Artworks of All Time." The artists range widely with respect to style, so we removed hyper-realistic artists like Caravaggio as well as highly modern artists like Warhol. The paintings that remain are quasi-realistic but maintain a highly painterly style, often with visible brushstrokes. This was important to us because those small details are what differentiate our project from existing photo coloration projects–we wanted our algorithm to learn the coloration details of brushwork, not just those of photographs.

We experimented with three different data augmentation methods. We initially tried the Java environment Processing, but soon learned that on proving ineffective for images of nonuniform size, we tried other methods. Our second data augmentation strategy used Keras functions, and we were able to achieve robust augmentation through image rescaling, mirrorings, and rotation with edge reconciliation. Ultimately we decided to limit our data augmentation in this initial analysis in order to set a realistic level of baseline performance for the model architecture proposed by Zhang et al. (2016), which we hope to improve on by studying recent developments in the field.

We used inbuilt functions provided by PyTorch to perform our image preprocessing and data augmentation. For our preliminary investigation, having thousands of images, we decided to perform only a small amount of preprocessing and data augmentation, so that we could allow our model architecture to display a good baseline performance. Our preprocessing step began by rescaling images to 256x256 with bicubic interpolation, and then performing some data augmentation by flipping half of the test set images across their horizontal axis. Finally, we create a greyscale version of each image to complete our feature-label pairs. After preprocessing, our dataset consisted of 8500 pairs of images, which we partitioned into training and test sets with an 80:20 split. That gave us 7800

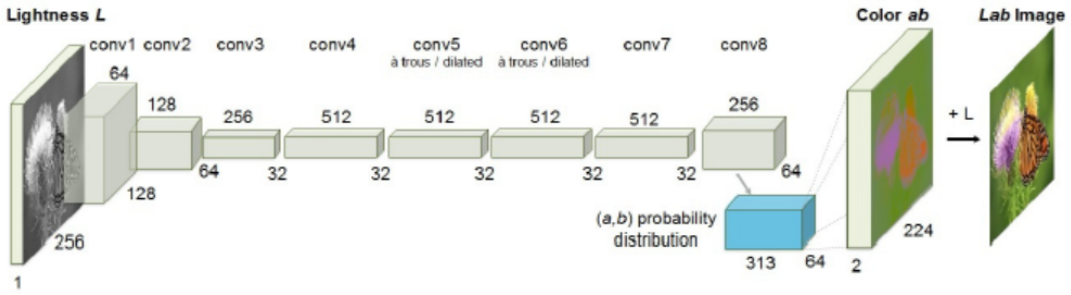
Figure 1: Our Data Preprocessing Pipeline

images for our training set, and 1700 for our test set. We ran our model for 40 epochs of 425 iterations each, and have collected a large amount of data on the performance of the Zhang et al. (2016) model architecture. Now that we have determined the baseline performance of our implementation of the model architecture from Zhang et al., we will also employ a development/validation set to evaluate different architectures and hyperparameter values as we move past our preliminary investigation.

3 Approach

In establishing a baseline, we build on the work of Zhang et al. (2016), as implemented by Shariatnia (2015). Rather than viewing colorization as a regression task, they break from the existing literature by modeling the problem as a question of multinomial classification in which individual pixels are assigned to color bins of a certain lightness value, which are proportionally sized in order to to give greater representation to rarer colors. Zhang et al. (2016) propose a neural network architecture composed of multiple blocks of convolution and ReLU layers, interspersed with BatchNorm layers to stabilize the network and prevent internal covariate shift. Given the Lightness channel L* in the CIELAB color space of the image as input, the model attempts to predict the a* and b* channels.

Figure 2: Neural Network Architecture Zhang et al. (2016)



Zhang et al. note that previous regression-based colorization attempts tend to generate desaturated images, with the L_2 loss function being the likely reason for these overly conservative predictions. They instead use a multinomial cross-entropy loss function, partitioning the color space for a given

lightness value into the $Q = 313$ bins that act as the category labels, giving rise to the loss function:

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

where $v()$ is a weighting term that the authors introduce to rebalance label classes and emphasize rare colors. This is important because it helps to avoid the desaturation problem mentioned above, where a colorizer will desaturate an image because of the high relative frequency with which desaturated pixel values appear in the input layer. The 313-dimensional vector Z represents the label encoding for the bin that a given pixel has been assigned to. Zhang et al. note that this vector was created using a soft-label encoding, because this yielded better experimental results against the desaturation problem than a one-hot label encoding scheme. The vector of predicted color distributions, $\hat{Z}$, is mapped to the vector of predicted colors $\hat{Y}$ through the function $\hat{Y} = \mathcal{H}(\hat{Z})$, defined as follows:

$$\mathcal{H}(Z_{h,w}) = \mathbb{E}[f_T(Z_{h,w})], f_T(z) = \frac{exp(log(z)/T)}{\Sigma_q exp(log(z_q)/T)}$$

The constant $0 < T \leq 1$ is a hyperparameter that can be tuned to adjust the desaturation present in the prediction vector $\hat{Y}$. Choosing a large value for T may cause the predictions to be overly desaturated, while choosing a small value may cause the image colorings to look overly 'blotchy.' We intend to experiment with different hyperparameter values for T in order to discover a level that is appropriate for the distribution of our data.

In our results, we observed the presence of the same desaturation effect described in the paper by Zhang et al. (Appendix fig. 3). While the progression of our training reveals that the chosen loss function does indeed play a significant role in reducing the desaturation effect, there remains work to be done in further reducing this desaturation to suit the needs of our dataset.

4 Remaining Work

We want to expand our use of data augmentation to re-implement resizing and rotation so that we will be able to train on a much larger dataset. We also want to experiment with modifying the hyperparameters presented in the Zhang et al. model, to explore whether we are able to find values that better suit our dataset. Additionally, we want to experiment with other loss functions, like Frechet inception distance, which is commonly used for evaluating images generated by GANs.

Finally, we want to explore other model architectures that may perform well on image colorization tasks, such as the U-Net convolutional neural network architecture introduced by Ronneberger et al. (2015).

References

[1] Zhang, R., Isola, P., & Efros, A. A. *Colorful Image Colorization.* ECCV, 2016

[2] Zhang, R., Zhu, J-Y., Isola, P., Geng, X., Lin, A. S., Yu, T. & Efros, A. A. R*eal-Time User-Guided Image Colorization with Learned Deep Priors.* ACM Transactions on Graphics (TOG), 9.4, 2017, ACM

[3] Ronneberger, O., Fischer, P., Brox, T. (2015, October). *U-net: Convolutional networks for biomedical image segmentation.* In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Springer, Cham.

[4] Shariatnia, M. (2020, November). *Colorizing black white images with U-Net and conditional GAN — A Tutorial.* https://towardsdatascience.com/colorizing-black-white-images-with-u-net-and-conditional-gan-a-tutorial-81b2df111cd8

[5] F. Zhu, Z. Yan, J. Bu, and Y. Yu. Exemplar-based image and video stylization using fully convolutional semantic features. 26:3542–3555, 07 2017.

Appendix

Figure 3: At the end of the first epoch, our model was actually pretty advanced–some colors were accurately predicted. Saturation, however, was a major issue. (Top: input, Middle: predicted image, Bottom: ground truth)

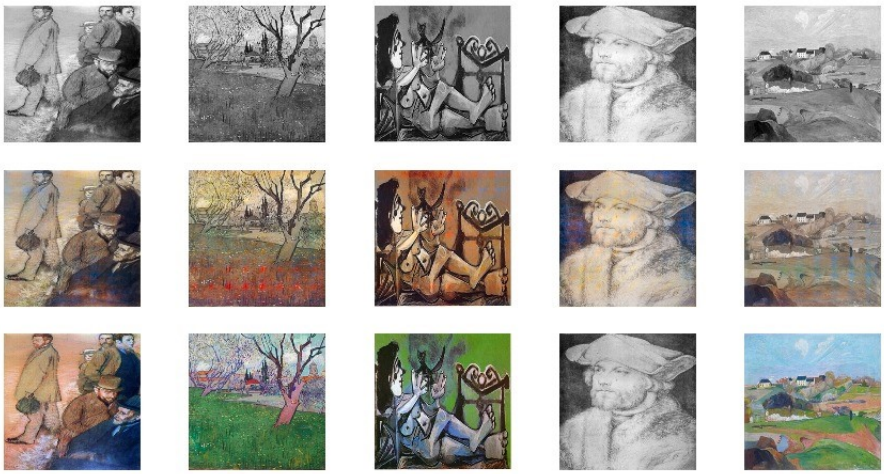


Figure 4: After 40 epochs, our model was able to figure out most colors. (Top: input, Middle: predicted image, Bottom: ground truth)

Figure 5: But even after 40 epochs, some images scored very poorly or had very weird results, as seen in the blue shade surrounding the eyes of the woman in the center right image. (Top: input, Middle: predicted image, Bottom: ground truth)