Using Pre-trained Generators for Image Colorization

Yuepeng (Calvin) Wang
yuepengw@mit.edu

Abstract

We study the problem of image colorization: generating colored images from gray-scale inputs. Much of past research involves training convolutional encoder-decoder architecture or Generative Adversarial Networks (GAN) from scratch for colorization. However, recent research on general-purpose GAN has brought pre-trained generators that can produce realistic images with rich color schemes. This paper proposes an encoder-decoder architecture that first learns an embedding for any grayscale input image, then uses a pre-trained generator to transform the embedding to the colored image. Only the encoder is trained in our architecture to learn the embedding and fully utilize the power of the generator. We conducted extensive experiments on a subset of ImageNet. Our results show that even though training processes suggest loss convergence, test set results are worse than baseline methods. Numerical results and qualitative analysis on generated images show the difficulty of finding an accurate correspondence between grayscale images and the low-dimensional latent space for generators for ideal colorization results. We also show that adding a shrinkage factor in the architecture can lead to embeddings closer to what the generator expects and results in more concrete shapes and objects in the colored images.

1. Introduction

In recent years, multiple fields in computer vision involving tasks previously deemed computationally or technically infeasible are witnessing breakthroughs made possible by the advent of deep learning and, specifically, neural networks powered by convolutional layers. One of these tasks, image colorization, involves colorizing a gray-scale image with little or no prior information. This seems to be an ill-posed problem since, theoretically, there is an infinite number of ways to color an image. However, colors are often associated with certain groups of objects in the human world. A colorization of an image that respects these associations and the underlying semantic structure of an image will lead to a result more pleasing to the human eye and more logically sound than any random coloring.

Modern convolution-based neural networks made it possible to take an input image and extract its semantic structure for a wide variety of tasks, including colorization. These architectures use convolutional layers to map (encode) images to lower-dimensional embeddings and in turn use them for color channel prediction and/or image generation, often using de-convolutional layers (decoder). This encoder-decoder architecture has been widely and successfully used in this domain.

Meanwhile, Generative Adversarial Networks (GANs) have been proven to be very successful at generating synthetic images. It uses a generator-discriminator architecture trained on large real-world datasets. The trained generators from these models can take lower-dimensional embeddings as input and generate realistic, colored images.

The following question can, therefore, be naturally proposed and investigated: instead of training, from scratch, large systems specialized for colorization, can we find an ideal embedding for any grayscale image, and utilize the power of pre-trained generators to find a colorization given the embedding? We propose our own encoder-decoder architecture to study the question; we present the methodology in detail in Section 3.

2. Related Work

As mentioned before, many architectures and techniques have been developed for image colorization. [1] contains a systematic survey of image colorization research. It includes plain convolutional architectures, user-guided models and text-based learning models (caption or prompt is given to aid colorization). This paper aims at exploring architectures that do not require prior information and user guidance. [14] uses convolutional layers to map the lightness channel of the image to an embedding, and uses it to predict the probability distribution of the color channels, framing it as a multi-class classification problem with custom loss function. [2] uses an encoder-decoder architecture along with another pre-trained Inception-ResNet-v2 network to further extract features to enhance the encoding. [7] uses a more complicated network with multiple paths composed of convolutional layers to learn features at multiple scales and de-convolutional layers to generate colored pix-

els.

There also has been research on using the generator-discriminator architecture first proposed by [5] to colorize images. Examples include [11], [12], and [13]. They all involved training GAN from scratch for colorization, possibly transferring the trained model/generators to other classification and image generation tasks.

The main contribution of this paper is exploring the method of colorization that utilizes the power of pretrained generators in GANs for colorization: such pretrained GANs on a variety of datasets are abundant ([3], [9], [10], [8]) and deliver good performance. The only paper focusing on a similar task is [6]; however this paper considers starting directly from latent embeddings and optimize on them to arrive at generated pictures. We include the entire encoder-decoder pipeline in our approach (described in detail in the following section). To sum up, our approach has not been explored in depth in past literature.

3. Methodology

For a training dataset of colored images, we can obtain, for each sample, a pair of grayscale image x_g and its colored version x_c. We wish to find an encoder $f(\cdot)$ and a pre-trained generator $g(\cdot)$ so that we can solve, given a loss function $\mathcal{L}$:

$$\min_{f,g} \mathcal{L}(g(f(x_g)), x_c) \tag{1}$$

The architecture used in this paper is shown in Figure 1. It consists of an encoder path and a classifier path. The encoder path takes x_g (in RGB format but all 3 channels are repeated since it is grayscale) as input and feeds it into an encoder (a pre-trained ResNet50 is used as a starting point). The output, a 2048-dimensional vector, is fed into 2 repetitions of the combination of layers: (Fully-Connected Layer, 1-dimensional BatchNorm). The first FC layer has 512 units and the second has 128. The two repetitions are connected by the ReLU activation. This encoding path maps the input grayscale image to a 128-dimensional embedding. Before being fed into the generator, this embedding vector is multiplied by a shrinkage factor. This is inspired by the image-generation implementation in [3] and aims to bring the embedding closer to the actual distribution of latent vectors the generator expects. We experiment with this hyperparameter in the following sections.

The classifier path takes the same x_g and uses a pre-trained EfficientNet network to predict the class label for the image (the classes are the standard ImageNet 1000 classes). The result is a 1000-dimensional one-hot encoded prediction.

The embedding and prediction are finally both fed into the pre-trained generator to produce the colored image. The loss function used is the pixel-wise Mean Squared Error (MSE) loss on the RGB channels (channels of all data are normalized to the [0,1] range).

The system is trained end-to-end; however, both the classifier and the generator are frozen. We only train (fine-tune) the encoder to find the ideal mapping. This preserves the learned information in the generator and speeds up the training process.

Method	LS	GT	Shrinkage
Variant 1	No	No	No
Variant 2	Yes	No	No
Variant 3	No	Yes	No
Variant 4	Yes	Yes	No
Variant 5	No	Yes	0.5
Variant 6	No	Yes	0.1

Table 1. Summary of all experiments. LS means the alternative learning rate schedule is adopted. GT means Ground Truth Class Label of images are used. Shrinkage means what multiplicative shrinkage factor (if any) is applied.

4. Experiments

4.1. Setup

The dataset is the ImageNet dataset [4]: it is chosen due to its diverse composition and color scheme. Due to computational constraint, we choose a subset of ImageNet consisting of 5000 images, obtaining the grayscale-colored pairs from these images. All input images are resized to (128, 128). The images are randomly split into 60% training, 20% validation and 20% testing.

The pre-trained generator in these experiments are from the BigGAN paper [3]; the generator was trained on the entire ImageNet, and it takes in a 128-dimensional vector as well as the class label (hence the classifier in the architecture above is necessary) and returns a (128,128) colored image.

In all experiments the Adam optimizer with 0.001 learning rate is used to 50 epochs; in some cases we adopt the alternative learning rate schedule where learning rate is reduced to 0.0001 after 25 epochs. We also perform experiments where we bypass the classifier and feeds the ground truth class label of each image to the generator, in order to control for mistakes made by the classifier on the overall colorization process. We also experiment with the shrinkage factor. All variants of the experiments are presented in Table 1. We compare all results to the baseline which is the model in Zhang, et al. [14]

4.2. Results and Analysis

Here we present the results. Figure 2 shows the loss on the training set and the validation set for all experiment vari-

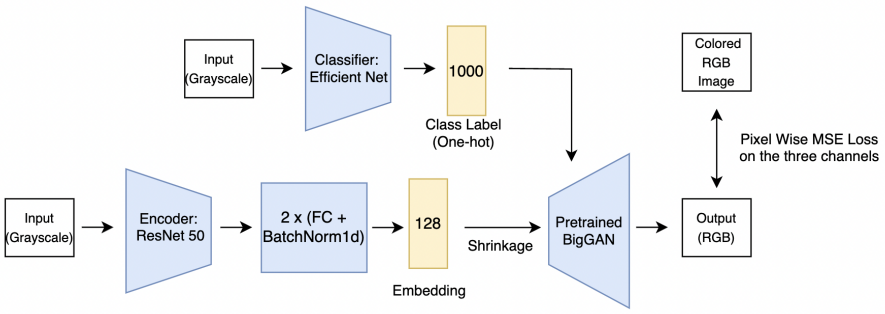

Figure 1. Model Architecture. The bottom part is the encoder path that maps grayscale input to 128-dimensional embedding; the upper part outputs class predictions (1000 dimensional one-hot vector) given the same grayscale input. Both results are fed into the pretrained BigGAN generator to produce colored image.

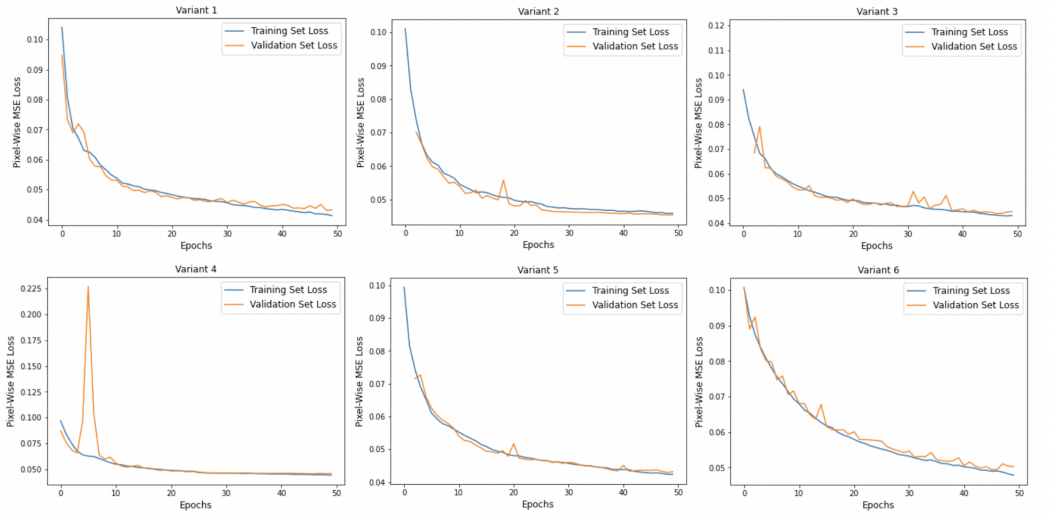

Figure 2. Training and Validation losses for all 6 variants. All are trained to 50 epochs (x-axis) and y-axis shows the MSE loss.

ants against the number of training epochs. We see that in all cases the losses show convergence towards the end of the 50 epochs, and the training and validation losses largely agree with each other, showing no signs of over-fitting. But none of these models is on par with the baseline, as is shown by Table 2, detailing the loss on the test set for all our models and the baseline. We see that the losses are 1 order of magnitude higher in our models, signaling the fact that our model has not been able to find an accurate embedding for the input grayscale images that results in a matching colored picture.

Method	Test Set Loss
Baseline	0.0091
Variant 1	0.0424
Variant 2	0.0447
Variant 3	0.0401
Variant 4	0.0452
Variant 5	0.0422
Variant 6	0.0489

Table 2. Experiment results on Test Set

This is further proven by looking at the resulting images and analyzing qualitatively. Figure 3 shows the results of all models on a random subset of images in the test set. As we can see, the baseline model performs a satisfactory colorization, but all the models using the BigGAN generator has failed to find the right embedding. The resulting images do not match up with the input in any way. However, we also see that the shrinkage has some impact in the images: the images show, in general, more concrete shapes and ob-

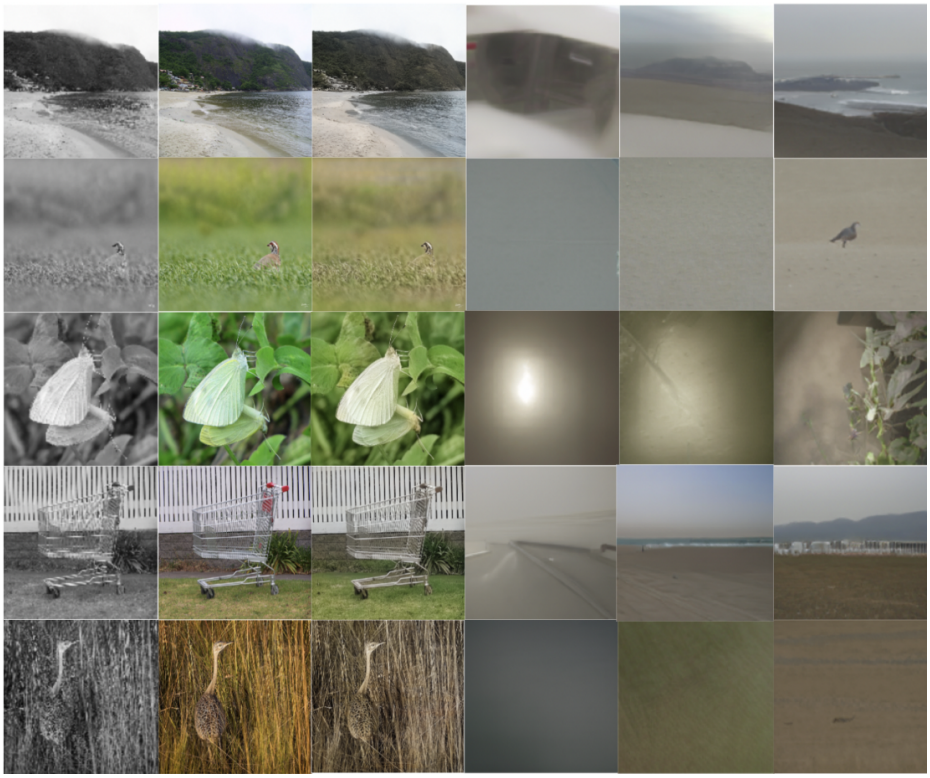

Figure 3. Results on a random subset of test set images. From left to right: Grayscale input, ground truth colored image, baseline (Zhang, et al. [14]), Variant 1, Variant 3, Variant 6

jects that matched with the ones in the original image. Figure 4 furthers proves this point: it shows the distribution of the 128 elements in the encoding constructed by some of the variants, given one same input grayscale image. We compare it to a same-sized sample of truncated normal distribution (truncated at -2 and 2), a distribution that BigGAN generator expects based on the official implementation in the paper. We see that Variant 5 and 6 with the shrinkage factor shows a distribution with the range much closer to the desired one than Variant 1. This shows that the shrinkage factor can create embeddings that are closer to the latent embeddings expected by the pre-trained generator and thus generate more believable images.

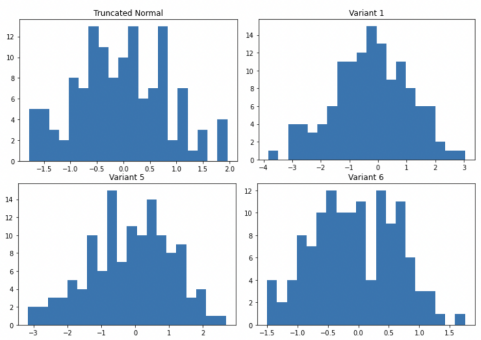

Figure 4. Histograms of 128 elements of embedding vectors produced by different variants of our trained models, given the same input grayscale, compared to a size-128 sample of Truncated Normal Distribution.

5. Conclusion and Future Work

In this paper we studied a specific method for image colorization: learning a lower-dimensional embedding for a given input grayscale image, and using that embedding with a pre-trained generator to generate colored image. Unfortunately, in the scope of this project and given the time and computational constraint, this method did not work well in the experiments. However, this was still a valid exploration into the method and showed the difficulty finding such embeddings. We also showed that having the shrinkage factor in the architecture and the training process makes the resulting images more believable due to presence of more concrete shapes matching the input images.

If more time and computational resources could be accessed, more epochs could be used to train the model on a much larger dataset (larger subset of ImageNet) in order to further evaluate its performance. Due to the difficulty of

this problem, one other potential direction is to follow [6] and construct the encoder network to output multiple embeddings instead of just one, in hope of alleviating the difficulty of accurately generating the input images.

References

[1] Saeed Anwar, Muhammad Tahir, Chongyi Li, Ajmal Mian, Fahad Shahbaz Khan, and Abdul Wahab Muzaffar. Image colorization: A survey and dataset, 2020. 1

[2] Federico Baldassarre, Diego González Morín, and Lucas Rodés-Guirao. Deep koalarization: Image colorization using cnns and inception-resnet-v2, 2017. 1

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 2

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society, 2009. 2

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial networks. *Commun. ACM*, 63(11):139–144, 2020. 2

[6] Jinjin Gu, Yujun Shen, and Bolei Zhou. Image processing using multi-code GAN prior. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 3009–3018. Computer Vision Foundation / IEEE, 2020. 2, 5

[7] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Let there be color! *ACM Transactions on Graphics*, 35(4):1–11, jul 2016. 1

[8] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *CoRR*, abs/2106.12423, 2021. 2

[9] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4396–4405, 2019. 2

[10] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. *CoRR*, abs/1912.04958, 2019. 2

[11] Kamyar Nazeri, Eric Ng, and Mehran Ebrahimi. Image colorization using generative adversarial networks. In Francisco José Perales and Josef Kittler, editors, *Articulated Motion and Deformable Objects*, pages 85–94, Cham, 2018. Springer International Publishing. 2

[12] Sandra Treneska, Eftim Zdravevski, Ivan Miguel Pires, Petre Lameski, and Sonja Gievska. GAN-based image colorization for self-supervised visual feature learning. *Sensors*, 22(4):1599, feb 2022. 2

[13] Patricia Vitoria, Lara Raad, and Coloma Ballester. Chromagan: Adversarial picture colorization with semantic class distribution, 2019. 2

[14] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. In *Computer Vision – ECCV 2016*, pages 649–666. Springer International Publishing, 2016. 1, 2, 4