Colorization Black and White Images with GAN

Zhiyuan Zhu
12421105
Department of Computer Science
12421105@zju.edu.cn

Abstract

The colorization of black and white images is an interesting and meaningful problem, but also difficult due to the uncertainty of the object color (for example, a car might have many diffenrent plausible colors). Various methods have been proposed to tackle this problem, particularly using the data-driven or the user-guided approach. In this article, I try to build a coloarization alogorithm using the generative adversial network (GAN) and the ImageNet dataset. In addition to training both the generator and descriminator from scratch, I also implement a version in which the generator is pretrained for some epochs. The pretrained version achieves more stable color generation. The generated color images are judged by naturalness, which is evaluated by human users. Code availabel at https://github.com/DarthMurse/Colorization.

1. Introduction

Black and white image colorization is interesting and meaningful problem in computer vision. Although nowadays colored images are easily aquirable using digital cameras, many important historical photos are shot in black and white. Effective colorization algorithm can help reconstruct the colored scene of these historical events and provide more information about it. Meanwhile, the colorization of black and white images many inspire artist with more color patterns of an given image.

Despite its simple objective, colorization of black and white images is not an easy task due to its inversible nature. The transformation from the colored image to the black and white image is inversible, meaning that one black and white image may corresponds to many colored images. Although there are some priors of the color distribution hidden in the black and white images, for example, a continous region with similar grayscale should also have similar color and the sky should be blue, it is very difficult to recover the original color of all the objects in a black and white image. For example, the original color of the car in the black and white image can be red, green or blue, which is hard to determine only using the black and white image.

To address this problem, the user-guided colorization method is proposed, in which the user offers some priors about the color of the image, like scribble some color patched at different parts of the black and white image. Among the different methods using the user-guided approach, the work by Levin et al. [4] is the most famous. It simply uses the prior that pixels with similar intensities should also have similar colors, and achieves very accurate results with just a few scribbles on the black and white image.

However, the user-guided approach still needs the user to provide the color information of different regions with similar intensities in the black and white image. If the black and white image has many seperated color patches, it would be laborious for the user to assgin them with different colors respectively. Luckily, this problem can be addressed with the data-driven approach. With the rise of large scale parallel training and large image dataset such as ImageNet [1], using deep convolutional network to generate colorized image from black and white images has become a new trend. In the data-driven approach, the deep neural network is trained on a millions of grayscale and colored image pairs to achieve natural image colorization without using human priors. This makes the colorization easier for the user to use.

In this article, I adopt the data-driven approach to solve the colorization problem. Spefically, I trained a generative adversial network (GAN) with a generator and a descriminator on the ImageNet dataset. The generator generates the colored images from the grayscale ones, while trying to fool the descriminator to believe the generated image is the original image. And the descriminator is trained to recognize the fake colored

images from the true ones. In addition to this naive GAN baseline, I modify the training process to include a pretraining stage for the generator, and increase the stability of the generation process.

2. Related Work

There have been many works in the field of using data-driven approach to solve the colorization problem, particularly using deep neural networks to generate coloful images. Without the help of human priors about the color, it is crucial for the network to extract these priors from the itensity information of the black and white image, for example, the sky is blue and the cloud is white. Iizuka et al. [2] proposed a method that trains one part of the network to do image classification, and use the representation of this subnetwork to extract the global features of the grayscale image, which is fused with the lower features generated by another subnetwork to obtain the final representation of the black and white image, than a decoder is applied to this final representation to minimize the L2 error between the generated image and the groudtruth image. Using the global features, the authors greatly enhance the naturalness of the generated color images.

Although treating the colorization problem as a regression task is the most straightforward way, without further priors it often leads to a "sepia" effect [5]. This is because most images are low-contrast in color distributions, minimizing the L2 error between generated image and the low-contrast image will induce the network to also generate low-contrast sepia images. In order to solve this problem, Zhang et al. [6] quantized the color space and turned the problem from a regression task to a classification class, and each pixel becomes a probabilistic distribution of colors. With some efforts to treat the uneven distribution of colors, they achieved vivid colorization of black and white images.

In this work however, we use the generative adversial network (GAN), as proposed in the work by Isola et al. [3], to treat the colorization problem. Using the descriminator and the generator, GAN can generate colored images from grayscale that have color distribution similar to real colored images. In the next section, I will explain more thoroughly about this approach.

3. Method

In this paper, I use the GAN to generate colored images in the LAB format from grayscale image. The reason to choose LAB as the color format and not the RGB format is that in the LAB setting, we can simply use the grayscale input as the L channel of the output image, and only need to generate the A and B channel with GAN, while in the RGB setting we need to generate all the 3 channels R, G and B, which is more difficult.

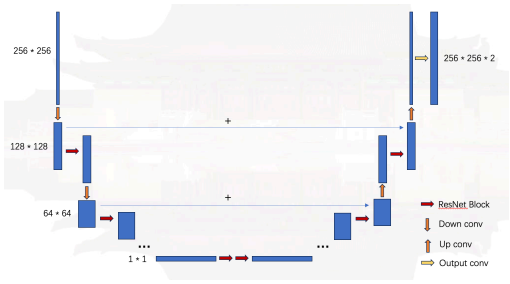

Figure 1. The residual net structure of the generator.

The GAN consists of two subnetworks, namely the generator and the descrimnator. The generator takes a grayscale image as input and outputs an image of the same size, but with two channels representing channel A and B. The output is concatenated with the grayscale input as the L channel to form a colored image. The descriminator is trained to distinguish the fake colored image made by the generator and the real colored image which the generator is trained on. Through the competition between these two networks, the generator learns to generate colored images that resemble the real ones. In the following section, I will explain the details of the networks.

3.1. Generator

The generator uses a Unet structure with residual connection to generate the a, b channels from the L channel of the input grayscale image, as shown in Figure 1. Each convolution block in the network consists of a 2D convolution, a 2D batch normalization and a nonlinear activation, which is majorly chosen to be ReLU and LeakyReLU. The down conv operation uses a convolution block with kernel size (4, 4), padding 1, and stride 2 in order to reduce the size of the image by a factor of 2. The up conv operation uses the same hyperparameter to enlarge the image by a factor of 2. Between the down conv and up conv operations are a Resnet block, which consists of two convolution block with kernel size (3, 3), padding 1, and stride 1, together with a residual connection. The Resnet blocks in the middle don't affect the size and channel number of the image, and are added to increase the ability of the network. Though the process of up sampling (with up conv), the down conv output at the former part of the network is added to the output of the up conv through a residual connection. At last, the output goes through another convolution block to increase the channel number from 1 to 2.

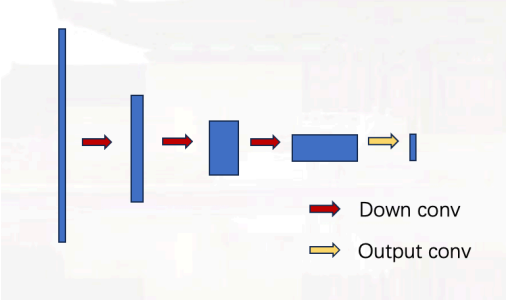

Figure 2. The residual net structure of the descriminator.

3.2. Descriminator

Instead of using a descriminator with a single label output, I use a patched descriminator, as shown in Figure 2, to generate a patched label for different regions of the image. The reason to apply a patched descriminator is to ensure that the colorization can be more accurate in details, since the colorization is a delicate process and each unnatural detail can spoil the whole generated image. The structure of the network is relatively simple, with three intermediate convolution blocks that shrink the size of the image and enlarge the channel of the image by a factor of 2. Then the output convolution block will reduce the channel number to 1 and give the final prediction of whether the corresponding patch of the image has fake or real colors.

3.3. Training Objective

The training of GAN is a competition between the descriminator network, namely D, and the generator network, namely G. The descriminator tries to distinguish the fake colored image generated by the generator from the grouhd truth colored image, while the generator tries to produce colored images that are similar to the ground truth images and fool the descriminator to believe that the generated images are real. We denote the grayscale image converted from ground truth colored image as x, the a,b channels of the ground truth colored image as y. Then the training objective for the descriminator is to minimize:

$$Loss_D(x,y) = -\mathbb{E}_{x,y} \log D(x,y) - \mathbb{E}_{x,y}(1 - \log D(x, G(x))) \tag{1}$$

For the generator, its training objective is minimize the following function:

$$Loss_G(x,y) = -\mathbb{E}_{x,y} \log D(x, G(x)) + \lambda ||G(x) - y||_1 \tag{2}$$

The L1 error is added to ensure that the generator don't generate that differs too much from the ground truth image and helps produce reasonble result at the beginning of the training process. Without this term, it is difficult to train the descriminator and generator from scratch because both the two networks are weak at the beginning and the generator can't learn to generate plausible images solely from the weak descriminator.

4. Experiment

4.1. Dataset

Other than using the Places365 [7] dataset as in Iizuka et al. [2], I use the ImageNet [1] dataset. This dataset contains 1000 diffenrent categories of objects and scenes, which is far more than the 365 places in the Places365 datast. Meanwhile, ImageNet also contains many sophisticated images other than places, and thus contains more diversified training examples. I believe the model can benefit from training with more complex data so that it can colorize complex grayscale images more naturally.

4.2. Details of Training

The training of the network is conducted on ImageNet. In training the shorter dimension of the image is resized to 300 and the longer dimension rescaled proportionally, then a random crop of size 256×256 and a random horizontal flip with 50% probability are applied to increase the input diversity. Before feeding into the network, the RGB image is converted to LAB format and normalized so that the value of each channel ranges from 0 to 1. For both the generator and the descriminator, I use an Adam optimizer with $\beta_1 = 0.5$, $\beta_2 = 0.999$ and learning rate 2e-4 with no weight decay. The GAN is trained on 6 A100-40G-PCIE GPUs with a batch size of 16 on each gpu and a total epoch of 100. For the pretraining of the generator, I use the same setting, except that the training epoch is reduced to 20.

In the end, I choose the model with the lowest L1 error to do the evaluation. Although L1 error may not be a good metric to assess the naturalness of the generated image, I find that model with higher L1 error tends to produce unnatural color patches or "purple margins" around some edges in the generated image, and the model with lower L1 error suffers less from this phenomenon.

4.3. Main Results

In this study I trained two GANs with different the strategies. In the baseline GAN, both the generator and the descriminator are trained from scratch, while in the other setting, the generator is pretrained for 20 epochs using the L1 error as the optimization objective,

Figure 3. The grayscale image.

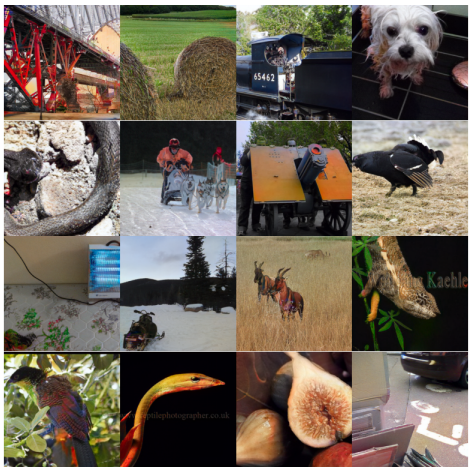

Figure 4. The colored image generated by the generator without pretraining.

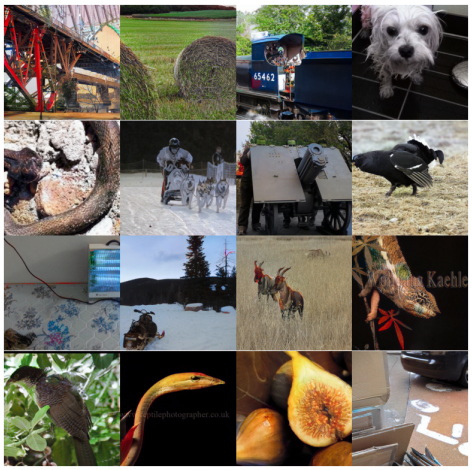

Figure 5. The colored image generated by the generator with pretrainning.

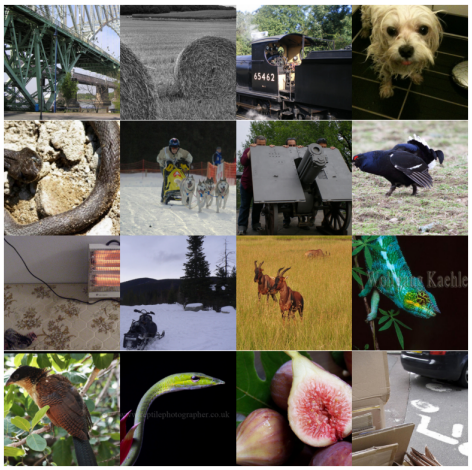

Figure 6. The ground truth colored image

then trained together with the descriminator as in the baseline setting. Figure 3 shows some examples of the input grayscale images. Figure 4 shows the colored image generated by the GAN trained from scratch, without pretrained generator. Figure 5 shows the colored image generated by the GAN with a generator trained for 20 epochs before training together with a descriminator. Figure 6 shows the ground truth colored images.

In general, both the GAN with pretrained generator and the GAN without pretrained generator can achieve reasonable colorization results. Although the generated color images might be different from the ground truth images in color, but they seems natural given only the grayscale image. Figure 8 show an example of this. There is another perspective to view this phenomenon. If we train a generator with only the su-

Figure 7. The colored image generated by the generator only with pretraining, without the help of the descriminator.

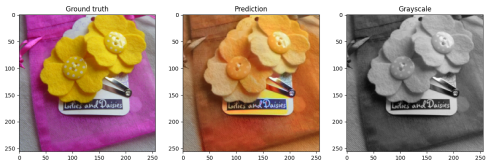

Figure 8. A case showing that a generated image that seems "natural" may not resemble the ground truth image.

pervision from the grayscale and colored image and L1 error as the optimization objective, we can only derive images that are low-contrast, without vibrant colors, and resemble very much to grayscale images, as shown in Figure 7. This also shows that L1 error is not a good metric to evaluate the result of the colorization algorithm.

Although the colored images generated by the two GANs trained with different settings are both natural, the generation of the GAN with a pretrained generator produces more stable result, as shown in Figure ??. Without the pretrained generator, the GAN sometimes generate images that has strange color patches, like the

Figure 9. A comparison between the two GANs trained with different settings. The first line shows the result generated by the GAN without the pretrained generator. The second line show the result generated by the GAN with the pretrained generator.

orange shade on the grean T-shirt in the first image, the purple patch on the instrument in the second image, and the purple paints in the third image. And with the GAN with a pretrained generator is less likely to produce these unnatural colors. This is probably because during the pretraining stage, the generator learns to generate plain, low-contrast images, so that it is more unlikely to produce an image with high-contrast strange color patches.

At last, to assess the "naturalness" of the generated images, I perform a user survey on 100 fake colored images generated by each GAN model from grayscale images in the validation set of ImageNet, which is unused in the training. The user is asked to judge if the colored image is natural or unnatural, in a sense that the image resembles a regular color photo, doesn't contain any strange color patches, lines or shades, and doesn't contain objects of unusual color, for example, a purple dog. Due to limited time and resources, I only asked my family members to evaluate the naturalness of the 100 images, which is 7 people in total. The final average percentage of naturalness for the GAN without the pretrained generator is 87.2%, while the score for the GAN with the pretrained generator is 91.3%, which somehow show that the GAN with pretrained generator is more powerful.

4.4. Failure Cases

As we can see, the GAN with the pretrained generator produces plausible colored images, but failure cases also exists, and can be categorized into two main classes which I called color randomness and wrong color semantics. Color randomness refers to the case when the main distribution of color of the image is correct,

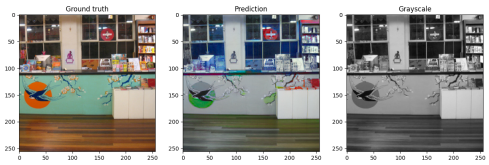

Figure 10. Purple color shades.

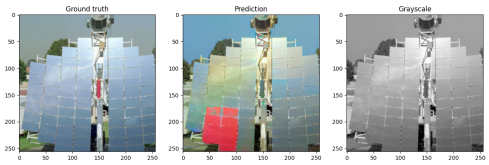

Figure 11. Wrong color semantics

but there are some unnatural shades of other colors in the image, as shown in Figure 10. And wrong color semantics refers to the case when several regions in the image should have a certain pattern of color, but the generated image does not. For example, the color of a national flag. Figure 11 shows an example of this failure. I suspect that the first failure case is related to the randomness of the generation process, which is further enhanced by the nature of GAN, because the descriminator introduces the extra randomness of the generated color distribution. To further prove this point, we can see that the generator trained solely with the L1 error doesn't contain such color shades at all. For the second failure case, I believe it is related to the limited capability of the generation network, since the understanding of image semantics requires a powerful network. The classfication of the ImageNet image is a difficult task for the generator network, let alone using the extracted information to generate reasonble colors for onjects with a complex semantics.

5. Conclusion

In this paper I proposed a method to colorize grayscale image using the GAN. Apart from directly training the GAN from scratch, I also train a GAN with the generator pretrained for 20 epochs. The result of the colorization shows that the GAN with the pretrained generator achieves more stable generation of colored images, achieving 91.1% percent of naturalness in an user survey conducted among my family members. After studying the generated colored images, I categorize the failure cases into two main classes, the color randomness and wrong color semantics. The most important insight I obtained from this project is that the success of image colorization doesn't solely depend on the L1 error between the generated image and the ground truth image, but more importantly on the higher semantic difference between these two images.

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In CVPR09, 2009. 1, 3

[2] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Let there be color! joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. ACM Trans. Graph., 35(4), July 2016. 2, 3

[3] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2018. 2

[4] Anat Levin, Dani Lischinski, and Yair Weiss. Colorization using optimization. ACM Trans. Graph., 23(3):689–694, Aug. 2004. 1

[5] Timothe´e Darcet Paul Jacob. mage colorisation using deep convolutional networks. 2

[6] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization, 2016. 2

[7] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017. 3