Automatic Image Colorization with Convolutional Neural Networks and Generative Adversarial Networks

Changyuan Qiu* Hangrui Cao* Qihan Ren* Ruiyu Li* Yuqing Qiu*
University of Michigan
{peterqiu,hangrui,qihanren,ruiyuli,qyuqing}@umich.edu

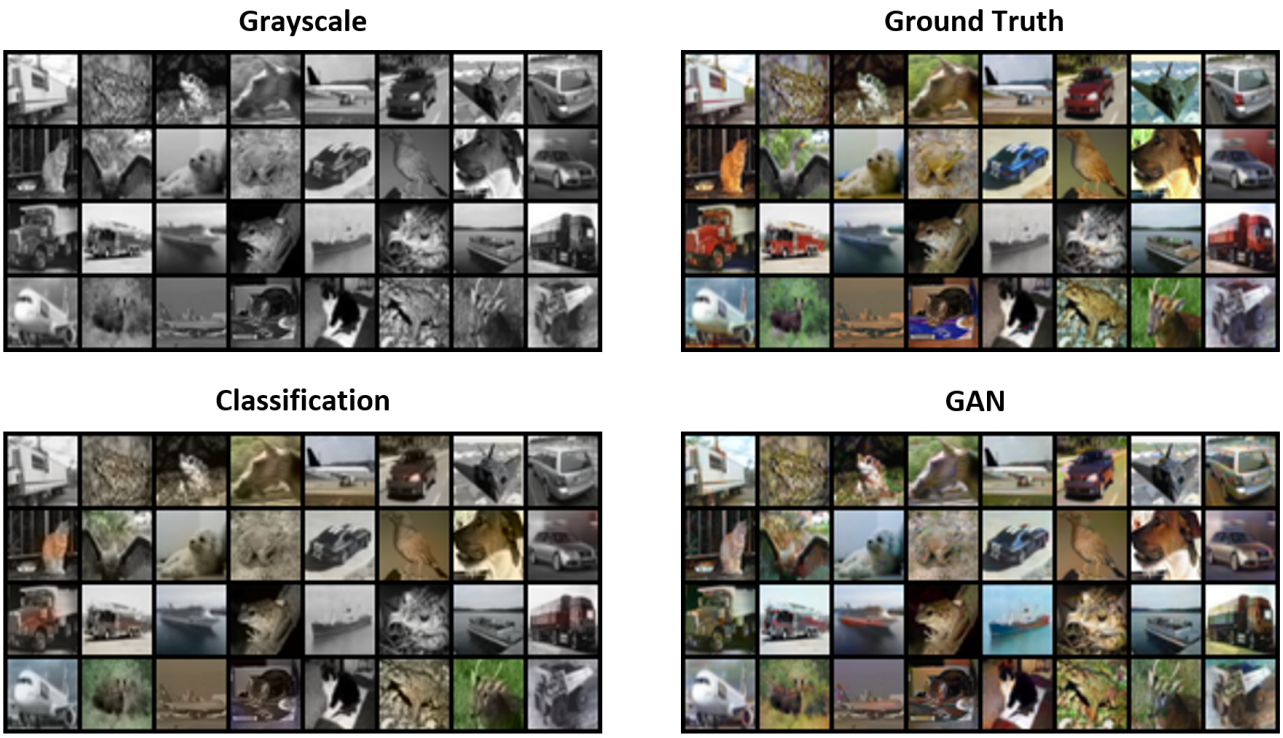

Figure 1: Colorization results on CIFAR10

1. Introduction

Image colorization, the task of adding colors to grayscale images, has been the focus of significant research efforts in computer vision in recent years for its various application areas such as color restoration and automatic animation colorization [15, 1]. The colorization problem is challenging as it is highly ill-posed with two out of three image dimensions lost, resulting in large degrees of freedom. However, semantics of the scene as well as the surface texture could provide important cues for colors: the sky is typically blue, the clouds are typically white and the grass is typically green, and there are huge amounts of training data available for learning such priors since any colored image could serve as a training data point [20].

Colorization is initially formulated as a regression task [5], which ignores the multi-modal nature of color prediction. In this project, we explore automatic image colorization via classification and adversarial learning. We will build our models on prior works, apply modifications for our specific scenario and make comparisons.

2. Related Works

Recently, deep learning techniques progressed notably for image colorization, and the fully-automatic colorization task (which does not take interactive input compared with the Scribble-based user-guided one) is commonly tackled

*equal contribution, ranked in alphabetic order

with 3 approaches: regression, classification and adversarial learning [1, 5, 20, 10, 15].

[5] takes the lead in investigating fully-automatic colorization with deep neural networks (DNNs) and formulates image colorization as a regression problem: the DNN takes the extracted feature descriptors at each pixel as input and outputs the continuous values of the colored channels at the corresponding pixel, with the loss function to be the Mean Square Error (MSE) between the predicted colored channel values and the ground truth. [20] first formulates the task as a classification problem by quantizing the ab channels of the CIE Lab color space into 313 discrete ab pairs and using the multinomial cross entropy loss to train the model. [10, 15] uses conditional Generative Adversarial Networks (cGANs) for automatic colorization of grayscale images.

3. Approach

We summarize the overall objective of colorization as follows: given a single lightness channel of an image $X \in \mathbb{R}^{H\times W\times 1}$, predict the two corresponding color channels $\hat{Y} = \mathcal{F}(X) \in \mathbb{R}^{H\times W\times 2}$, where $\mathcal{F}$ is the mapping function to be learned.

3.1. Classification-based Colorization

In this approach, we treat the colorization task as a classification problem instead of a regression problem due to the ambiguity of colorization. We employ an architecture similar to U-Net [18] without skip-connection and with dilated convolutions following [20, 4, 19]. Details of our architecture are shown in Figure 2 with two variants.

To formalize, the *ab* color space is quantized to bins of size 10, and yields a total of 313 possible *ab* pairs. Our network maps the input X to a probability distribution $\hat{Z} \in [0,1]^{H\times W\times Q}$, where $Q = 313$ is the number of quantized *ab* pairs. Our loss function is formulated as follows.

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

where $\hat{Z}$ is the predicted probability distribution and Z is the quantized ground truth, using the soft-encoding scheme in [20]. Therefore, the network is not strictly end-to-end learned. To obtain the final colorized image, we first map the predicted probability distribution $\hat{Z}$ to color channels $\hat{Y}$ by function $\hat{Y} = \mathcal{H}(\hat{Z})$, and then concatenate $\hat{Y}$ to the input lightness channel X. For function $\mathcal{H}$, we use the *annealed-mean* operation in [20] with temperature $T = 0.38$.

Unlike [20], we did not apply class rebalancing. The technique is originally used to correct the bias towards lower *ab* values, but we empirically observed that it disrupted the training process in our case.

3.2. GAN-based Colorization

Generative Adversarial Networks (GAN) [7] are composed of two competing neural network models. For this colorization problem setting, the generator takes grayscale images and generates colorized versions; the discriminator gets colored images either from the generator or the labels, concatenated with the grayscale images, and tries to identify which pair contains the real colored image [17]. Similar to the approach in [17], we utilize deep convolutional neural networks as generative models for our adversarial framework. Since for the colorization problem setting, the input is grayscale images instead of random noises, we employ a conditional GAN instead of the traditional one. The cost functions are formulated as follows.

$$\min_{\theta_G} J^{(G)}(\theta_D, \theta_G) = \min_{\theta_G} -\mathbb{E}_z\left[\log\left(D\left(G\left(\mathbf{0}_z|x\right)\right)\right)\right] + \lambda\left\|G\left(\mathbf{0}_z|x\right) - y\right\|_1$$

$$\max_{\theta_D} J^{(D)}(\theta_D, \theta_G) = \max_{\theta_D}\left(\mathbb{E}_y[\log(D(y|x))] + \mathbb{E}_z\,\log\left(1 - D\left(G\left(\mathbf{0}_z|x\right)|x\right)\right)\right])$$

where $G\left(\mathbf{0}_z|x\right)$ is the colorized image produced by the generator, with input as zero noise $\mathbf{0}_z$ with the grayscale image x as a prior and y is the ground truth.

We build up and train a Conditional Deep Convolutional Generative Adversarial Network (C-DCGAN) following [15]. We employ a modified U-Net [18] for our basic architecture. For the generator G, it is constructed as the modified U-Net model as shown in Figure 3. For the discriminator D, it only utilizes the contracting part (encoder) in the U-Net model, with the number of channels being doubled after each down-sampling. The output layer is a 4×4 convolutional layer with stride 1, which generates a 1 dimensional output. Finally, the sigmoid activation function is used to map the output to a probability of the input image being real.

To better control the color space, we separate the brightness channel and color channels using the CIE Lab color space, where we only need to predict two color channels ab in the generator.

4. Experiment

Dataset We evaluate our models on the canonical dataset CIFAR-10 [13]. CIFAR-10 consists of 60000 images of resolution 32×32 uniformly partitioned in 10 classes, with each class having 6000 images. Specifically, for each class, 5000 images are randomly selected for training and the remaining 1000 images are left for testing.

Training We train both models using Adam [11], with learning rate of 1e-3 for the classfication-based model, and learning rate of 1e-4 for both generator and discriminator of the GAN-based model. The regularization term λ is set as 100 for the generator of the GAN-based model. Our classfication-based model is trained for 100 epochs, and takes 4.5 hours to train on one Tesla V100 hosted on Google Colab Pro; our GAN-based model is trained for 200 epochs, and takes 4 hours on one Nvidia GTX 2070 hosted on a remote server.

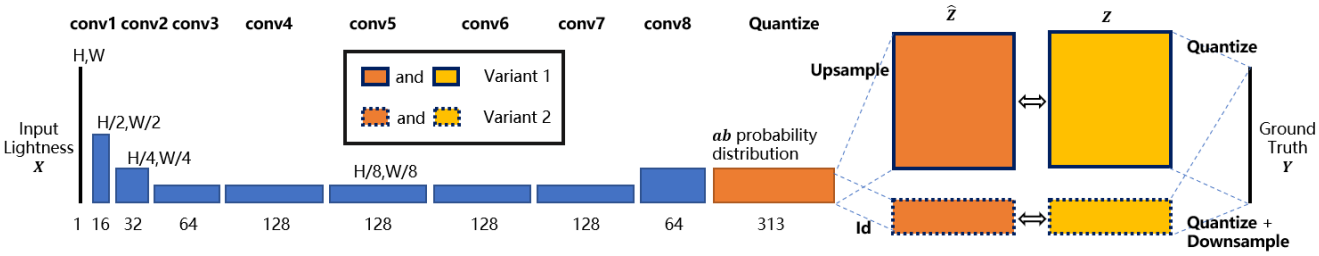

Figure 2: Network architecture for classification-based colorization. Each Conv block is composed of 2 or 3 repeated Conv and ReLU layers. All spatial downsamplings in the network are achieved by Conv layer with stride greater than 1. We have two variants for this architecture w.r.t the output layer. **Variant 1:** we upsample the ab probability distribution to the original size, and calculate loss w.r.t the quantized ground truth. The upsampling methods include bilinear interpolation and ConvTranspose. **Variant 2:** we keep the *ab* probability distribution unchanged (**Id** in the figure) , but downsample the quantized ground truth to make the size match.

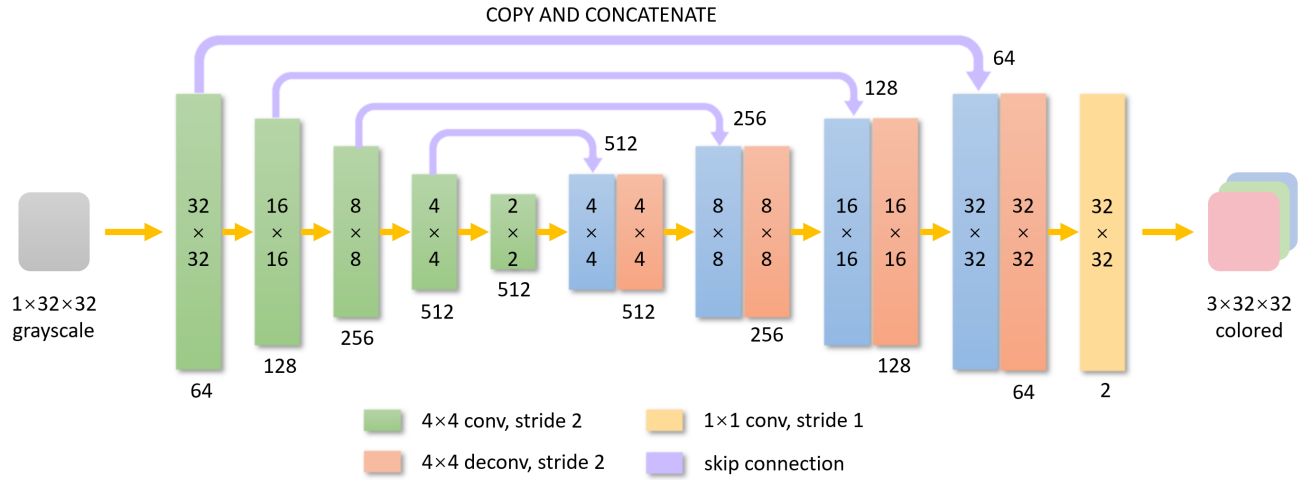

Figure 3: Network architecture for generator (U-Net) of GAN. The symmetric architecture consists of left contracting path and right expansive path. For the green contracting path (encoder), each block is a 4×4 convolutional layer with stride 2 for down-sampling, followed by batch normalization and Leaky-ReLU activation function with slope 0.2. For the right expansive path (decoder), each block consists of a 4×4 transposed convolutional layer with stride 2 for up-sampling and concatenation with the mirroring layer from the contracting path. Then, the concatenated block goes through a 3×3 convolution layer with stride 1 for halving channels, followed by batch normalization and ReLU activation function. The output layer is a 1×1 convolution layer followed by a Tanh activation function.

4.1. Evaluation Metrics

Pixel-wise Accuracy To measure the difference between the learned images from our models and the true images, we first measure the mean absolute error between pixels. For one test image, the accuracy is measured by the ratio of the number of pixels whose errors are smaller than the error threshold ϵ. Formally, denote $pred(i, j)$ as one image pixel in image generated by the model and $real(i, j)$ as the original image pixel values. Both pixel values are normalized to $[0, 1]$. If the absolute difference is smaller than ϵ, namely

$$|pred(i, j) - real(i, j)| < \epsilon,$$

then $pred(i, j)$ will be classified as true in terms of color information compared with the pixel in original ones. As each pixel has R, G, B values, we mark one pixel as true when all of its three channel values satisfy the above equation.

PSNR and SSIM Besides, we use peak-signal-to-noise ratio (PSNR) and structrural similarity index (SSIM index) to measure image qualities [9]. PSNR is defined in log form of the mean sqaure error and we calculate PSNR for each generated images(after transformation from Lab to rgb). SSIM is a method to measure the similarity between images. We calculate average PSNR and SSIM values on the test dataset and the results are shown below.

4.2. CIFAR-10 Results

We carried out experiments on the CIFAR-10 dataset. Both the classification method and the GAN method are able to automatically colorize grayscale images to an ac-

Model	Pixel-Acc $\epsilon = 2\%$	Pixel-Acc $\epsilon = 5\%$	PSNR (dB)	SSIM
Classification (Upsample - Bilinear Interpolation)	0.888%	5.272%	21.491	0.913
Classification (Upsample - Deconv)	0.919%	5.189%	21.220	0.908
Classification (Downsample)	0.923%	5.828%	21.848	0.913
GAN (ours)	33.255%	57.510%	24.608	0.910
GAN [15]	24.100%	65.500%	—	—

Table 1: Model Performance Comparison on CIFAR-10

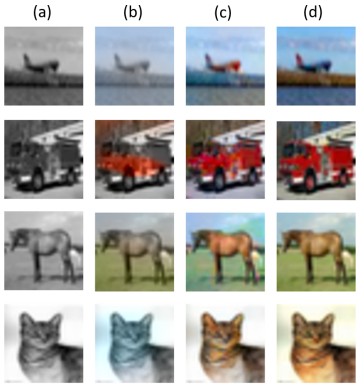

Figure 4: Comparison of Colarization Results on CIFAR-10. (a) Grayscale. (b) Classification. (c) GAN. (d) Ground Truth.

ceptable visual degree. Qualititative results are shown in Figure 1, 4. Quantatitive results are shown in Table 1. Compared with classficiation methods, images generated by GAN has much higher pixel-wise accuracy and higher PSNR(dB) values, indicating that in general the GAN method performs better than classification. The SSIM values observed by two methods are approximately the same. As we calculate the metrics based on the three channels (R, G, B), we compare the pixel-wise accuracy of three channels and find that for both two methods, the accuracy in R channel is lower than the other two channels, which implies our models are weak to generate colors in R channels.

4.3. User Study

We carried out a fool study process by randomly picking 200 generated sets, where each set has three images, the ground truth and colorized images learned by classification and by GAN respectively. We asked 16 students who did not know the image labels in advance to identify which image is the ground truth in each set. The identified class ratios are listed in table 2. We also asked users to rate images ranging from 1 - 5, with higher score indicating better image reality and quality.

Identified class ratio	Ground Truth	Classification	GAN
identified / total	54.91%	4.80%	40.69 %
Avg Score	4.0	2.3	3.7

Table 2: User Study Table

The above results show that images generated by GAN and CNN can fool users to some extent.

5. Implementation

For classification-based approach, we adopt the architecture in [20] but reduce the number of channels by a factor of $\frac{1}{4}$ to align with the smaller image size (32×32) in CIFAR-10. In addition, we implement two variants of the architecture, with different downsampling or upsampling methods for the output layer (see Figure 2). We implement the training and evaluation code from scratch in PyTorch, and tune the hyperparameters by ourselves. We utilize the tensorboard function in PyTorch to display synchronous colorization results. The code snippets for color conversion and quantization of *ab* color space are borrowed from [21] and [3] respectively.

For GAN-based approach, we adopt the architecture of C-DCGAN and hyperparameters given in [15] but reduce the number of layers in the U-Net basic architecture as we are dealing with smaller images (see Figure 3). We implement the models of generator and discriminator in PyTorch from scratch and simplify the code to make it more straightforward. We utilize the tensorboard function in PyTorch to display synchronous colorization results. We refer to dataloader code, training code and color conversion and quantization code from *rgb* to *Lab* in [2] but we implement our own evaluation code, color conversion and quantization code from *Lab* to *rgb*.

6. Conclusion

In this project, we compare and evaluate the performance of convolutional neural networks and generative adversarial networks on automatic image colorization tasks. Both of them are able to automatically colorize grayscale images to an acceptable visual degree. Compared with the classification-based CNN method, C-DCGAN perform much better while is also more computationally expensive at the same time.

7. Future Work

We plan to experiment with images of higher resolutions from dataset like ImageNet [6] (224×224) or MS COCO [14] (640×480). Besides, modifying the backbone of the classifier (like change to ResNet [8]) can potentially improve the performance. We also plan to further explore other generative models like VAE [12] and VQ-VAE [16].

References

[1] Saeed Anwar, Muhammad Tahir, Chongyi Li, Ajmal Mian, Fahad Shahbaz Khan, and Abdul Wahab Muzaffar. Image colorization: A survey and dataset. *arXiv preprint arXiv:2008.10774*, 2020. 1, 2

[2] Harshit Bansal. Image colorization. https://github.com/harshitbansal05/Image-Colorization, 2018. 4

[3] BingWin789. colorization-traininglayers-tf. https://github.com/BingWin789/colorization-traininglayers-tf, 2020. 4

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, 2017. 2

[5] Zezhou Cheng, Qingxiong Yang, and Bin Sheng. Deep colorization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 415–423, 2015. 1, 2

[6] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 4

[7] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014. 2

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[9] A. Horé and D. Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th International Conference on Pattern Recognition*, pages 2366–2369, 2010. 3

[10] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 2

[11] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 2

[12] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 4

[13] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 2

[14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 4

[15] Kamyar Nazeri, Eric Ng, and Mehran Ebrahimi. Image colorization using generative adversarial networks. In *International conference on articulated motion and deformable objects*, pages 85–94. Springer, 2018. 1, 2, 4

[16] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *arXiv preprint arXiv:1711.00937*, 2017. 4

[17] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016. 2

[18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. 2

[19] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016. 2

[20] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 649–666, Cham, 2016. Springer International Publishing. 1, 2, 4

[21] Richard Zhang, Jun-Yan Zhu, Phillip Isola, Xinyang Geng, Angela S Lin, Tianhe Yu, and Alexei A Efros. Real-time user-guided image colorization with learned deep priors. *ACM Transactions on Graphics (TOG)*, 9(4), 2017. 4