Automated Image Colorization Using Deep Learning

Hanzhao Lin
Stanford University
hanzhao@stanford.edu

Rafael Ferreira
Stanford University
rcf2132@stanford.edu

Abstract

Image colorization task is to recover a plausible color version of the image from its grayscale version. In this work, we explore machine learning techniques that colorize grayscale images in an automated way, and further apply convolutional neural network with pre-trained EfficientNet [1] to produce realistic colorization. The model is trained on Places dataset [2] and evaluated in both subjective and quantitative ways. To explore its capability of generalization, we also collect real-world black-and-white photos and observe model performance on them.

1 Introduction

Before humans had devices to capture colored media, the only available option was to record grayscale images and videos. With the advance of technology, nowadays software has been created to help people recover such legacy grayscale media into original atmosphere. But automatically colorizing an image is still challenging due to its uncertainty. It's nearly impossible to perfectly colorize artificial items without prior knowledge, like T-shirts, signs and paintings. Besides, cameras may capture the same view with different tones, color saturation and lighting condition. Thus, we are not trying to recover the original colorization, but targeting to provide a *plausible* colorization for the given image.

In this work, we review a few approaches taken from literature to understand how related research has evolved. Then, we explain the datasets we used, and describe the models we developed as to understand their performance. Our final approach[1], based on convolutional neural network with pre-trained EfficientNet, takes luminance channel of the image as input and predicts the corresponding color channels as output, as shown in Figure 1. The model performance is measured in subjective and quantitative ways. Finally, we summarize our findings and propose potential improvements.

2 Related Work

Early work on automated colorization relied on simple approaches. For example, Gonzalez and Woods [3], Welsh et al. [4], Reinhard et al. [5] and Hertzmann et al. [6], were based on lookup tables or reference image to perform the grayscale-to-color mapping and color transfer. These techniques were far from being automated since considerable human interaction was needed. Newer techniques, e.g., color propagation in Levin et al. [7] and Yatziv and Sapiro [8], alleviated how much interaction was needed from users. These techniques produced good results, but still relied on user interaction.

Fully-automatic approaches using modern machine learning techniques were presented by Cheng et al. [9], Zhang et al. [10], Larsson et al. [11], which utilized Convolutional Neural Network (ConvNet) for automatic digital image colorization. These deep neural networks require long-time training for many parameters, as well as massive image datasets. To reduce the training cost, transferred learning approach was practiced in Dahl [12]. More recent work in Ci et al. [13], Nazeri and Ng [14] and Kumar et al. [15], involve Generative Adversarial Networks [16] or Transformers [17]. These

[1] Code available on GitHub: https://github.com/hanzhao/image-colorization

CS229: Machine Learning, Spring 2021, Stanford University.

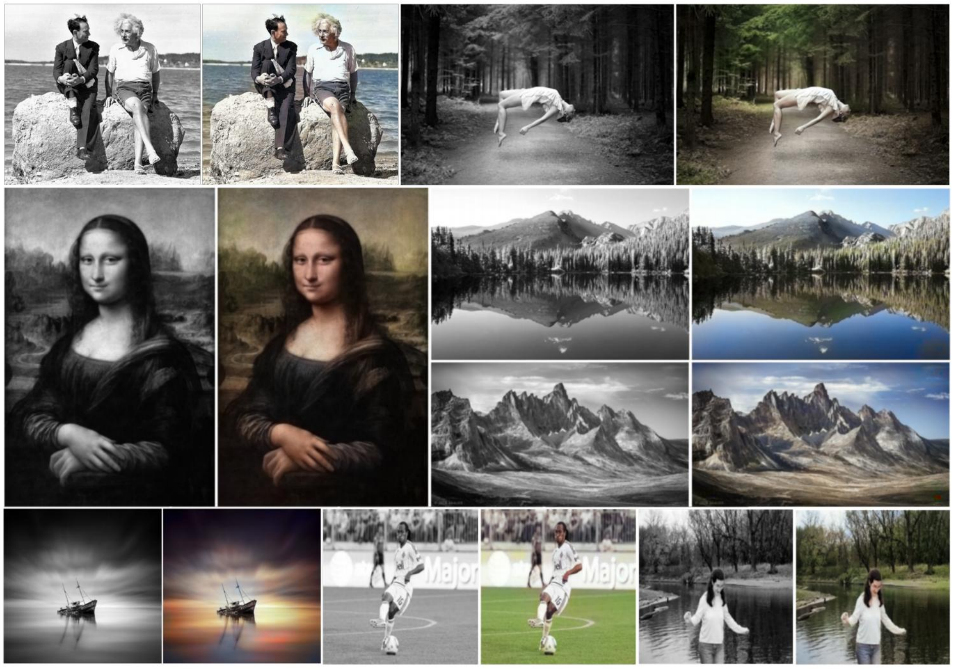


Figure 1: Selected examples of input grayscale photos and output from our model.

approaches have demonstrated better colorization quality than previous ConvNet models, though authors suggested the necessity of finding better metrics to measure colorization performance.

3 Datasets

To the best of our knowledge, there is no dataset dedicated for image colorization research. But there are many large-scale image datasets, like ImageNet [18], Food-101 [19] and Places [2], which have proven useful on image classification tasks. Since most images in these datasets are fully colored, it's relatively straightforward that we could use these datasets in a self-supervised manner. Concretely, the original images are transformed into grayscale images as the input of models, and we expect the model to output colored images that are as close as possible to the ground truth.

We choose the small version of Places dataset for our research. This dataset contains about 2 million images of 256×256 resolution comprising about 400 scene categories. For comparison, Food-101 contains 101,000 food photos, which is not diverse enough to train models working for generic real-world photos. While ImageNet, with 14 million images, is too challenging under the constraint of our resources. Thus, Places dataset makes a good balance in terms of the image diversity and size.

In terms of data preparation, we follow the suggested practice of Places dataset, which separates the entire dataset into a training set of 1,803,460 images, a validation set of 36,500 images and a test set of 328,500 images. To ensure easy integration with our models, all images are resized into 224×224 with exactly the same batch size. To achieve better generalization, two argumentation approaches were applied on the training set, including horizontal flipping and random rotation.

To manufacture grayscale images from datasets as shown in Figure 2, we transform the original images from RGB color space to CIE Lab color space and take only the `L*` channel of it, which represents the human-perceived lightness value. `L*` values are normalized into floating values ranging from 0 to 1 in our implementation. To summarize, the input grayscale image will be represented as $224 \times 224 \times 1$ floating number matrix whose all values are within `[0, 1]` after the preprocessing.

For modeling the output of model, we explored both YUV and CIE Lab color spaces. Both color spaces have one luminance channel and two color channels. Also, they both proved effective for this task by previous efforts. Since CIE Lab color space was designed for perceptual linearity, we took it as our main approach. The color channels of CIE Lab color space, `a*` and `b*`, are normalized into

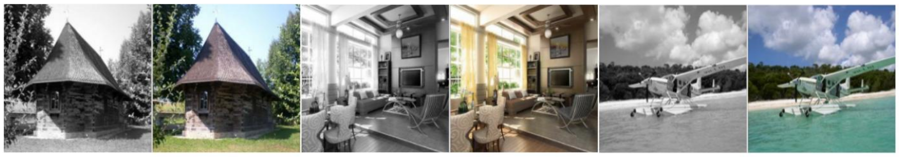


Figure 2: Example images after preprocessing in Places dataset.

`[-1, 1]`. We also implemented YUV color space to replicate the model proposed in Dahl [12]. The color channels in YUV color space, `U` and `V`, are normalized into `[-0.5, 0.5]`.

Generally, no matter which color space is being used, models will predict 2 missing color channels for each pixel, thus the output image is represented as $224 \times 224 \times 2$ floating number matrix. To display the model output, the input luminance channel and two output color channels need to be unnormalized, combined and transformed back into RGB color space.

4 Methods

Based on our review of literature and our constraint of computing resources, we prioritize transfer learning approaches as our main focus. We choose the model proposed in Dahl [12] as the starting point and iterate the approach based on our observation of its performance on the Places dataset. Finally, we propose improvements on objective functions and on the network architecture.

4.1 Objective Function

The objective is to minimize the mean squared error (MSE) for 2 output color channels, `a*` and `b*` in CIE Lab color space, between ground truth and prediction. In this way, models are expected to find the minimal perceptual color shifting. For a single training datapoint, the loss can be represented as

$$\mathcal{L}(Y^{(i)}, \hat{Y}^{(i)}) = \frac{1}{2hw} \sum_{h,w} ||Y^{(i)}_{h,w} - \hat{Y}^{(i)}_{h,w}||_2^2 \tag{1}$$

After several experiments, we noticed that using this objective function led to a tendency of desaturated colors. Inspired by [10], we analyzed the training set and confirmed that desaturated colors appear much more frequently than vivid colors in the real world as Figure 3a, 3b. We apply a penalization to common colors to encourage output of more vivid colors. The cost function is then modified to be

$$\mathcal{L}(Y^{(i)}, \hat{Y}^{(i)}) = \frac{1}{2hw} \sum_{h,w} v(Y^{(i)}_{h,w}) ||Y^{(i)}_{h,w} - \hat{Y}^{(i)}_{h,w}||_2^2 \tag{2}$$

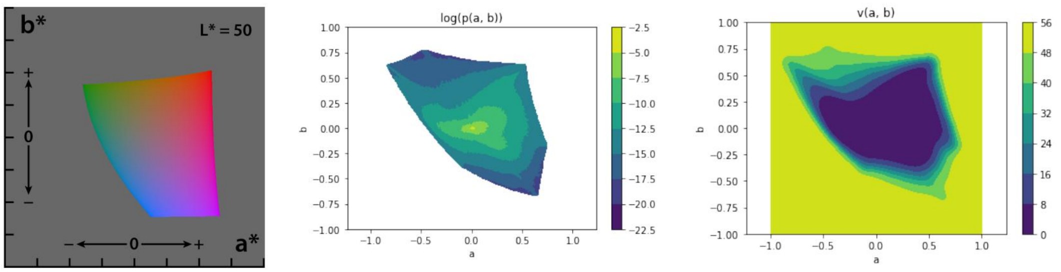
where v is the color rebalancing function, which can be considered a weight that relates to the rarity of color. Concretely, we represent the weight of a color as

$$v(a,b) \propto \frac{1}{(1-\lambda) \times \tilde{p}(a,b) + \lambda \div 4}, \quad s.t. \quad \mathbb{E}[v(a,b)] = \sum_{a,b} \tilde{p}(a,b) v(a,b) = 1 \tag{3}$$

Similar to Zhang et al. [10], to obtain an empirical probability $\tilde{p}$, we calculated the possibility of discretized `(a*, b*)` pairs in full training set and smooth the distribution with a Gaussian kernel. We also introduced a hyperparameter $\lambda \in [0, 1]$ to mix the probability with a uniform distribution. We normalize the weight table so the weighting factor is 1 on expectation, as shown in Figure 3c.

4.2 ConvNet with EfficientNet Feature Extractor

The base model has provided acceptable performance with limited computing resources and dataset, thanks to its transfer learning setup. To explore the limits of this approach, we make several enhancements accordingly. First, the original approach was to minimize squared error on YUV color space, and we replace it with CIE Lab color space for perceptive linearity, so that minimizing squared error is identical to minimizing the perceptive color shift. Second, the original approach utilized 4



(a) Colors in CIE Lab color space when $L* = 50$.

(b) Possibility of color appearance in training set.

(c) Adjusted weight of each color when $\lambda = 0.9$.

Figure 3: Exploration on color distribution and weight rebalancing.

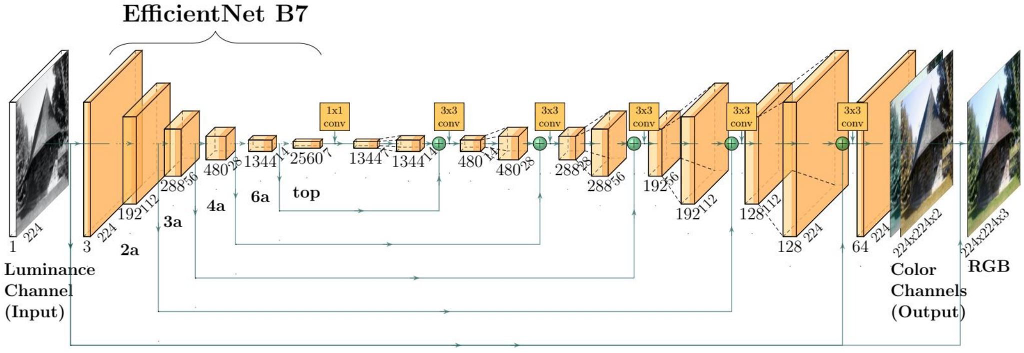


Figure 4: Model architecture and expected input/output.

intermediate layer outputs of pretrained VGG-16 model [20], and we replace it with 5 intermediate layer outputs of EfficientNet B7 to match the state of the art. Those layers have a similar shape with VGG-16 and are also suggested by original authors to be useful for transfer learning.

As shown in Figure 4, the model utilizes 5 intermediate activation layers in EfficientNet B7. The output of 7×7 activation layer of EfficientNet is passed through a 1×1 Conv2D block to produce $7 \times 7 \times 1344$ output matrix. To fuse it with other activation layers, the output is later up-scaled and added up to the output of previous activation layer, followed by another Conv2D block to resize its dimension. This process is repeated 5 times so that the original 224×224 image is recovered. Each Conv2D block consists of a 3×3 Conv2D layer, a Batch Normalization layer and a ReLU activation layer. Finally, to output color channels in CIE Lab color space, the output block is defined as a 3×3 Conv2D layer with 2 output channels and a tanh activation layer mapping values to `[-1, 1]`.

5 Experiments

We implemented our approaches on Google Colab. As comparison, we also implemented a naive ConvNet model and a replication of Dahl [12] as baseline models. The outcome of these models is evaluated on a test set, based on both perceptive plausibility and MSE defined in Equation 1.

On choices of hyper-parameters, we use Adam optimizer [21] with learning rate as 0.001, β_1 as 0.9 and β_2 as 0.999 when training. To obtain the empirical probability $\tilde{p}$ and calculate weight for color rebalancing, we discretize `a*` and `b*` into 256×256 grids, apply 10×10 Gaussian kernel with $\sigma = 5$ to smooth distribution, and select $\lambda = 0.9$ to mix it with uniform distribution in Equation 3. The EfficientNet model parameters are obtained from pretrained weights on ImageNet dataset and frozen.

Output of our implemented models is selected and shown in Figure 5. Also, we manually gathered some black-white photos and observed the output as shown previously in Figure 1. Intuitively, all models work well on colorizing natural views like sky, river and grass, but fail to colorize artificial items like clothes and shelves. These unrecognized artificial items tend to be colored as dark yellow,

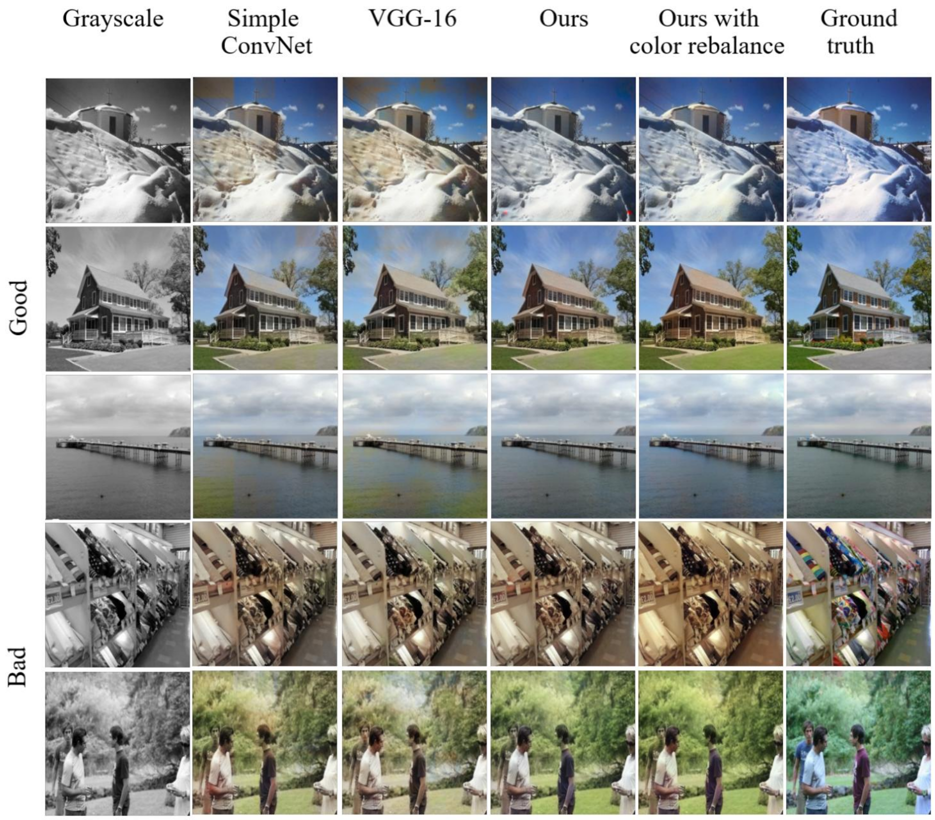


Figure 5: Output images of our implemented models.

	ConvNet Baseline	VGG-16 Baseline	Ours	Ours w/ Color Rebal
MSE (CIE Lab)	0.0082	0.054	0.0075	0.5848

Figure 6: MSE in CIE Lab color space of all implemented models.

which is proven the most common color in the dataset as analyzed in Figure 3b. Due to the simplicity of the baseline models, there could be visible flaws in the output images, which has been fairly mitigated by a more sophisticated model architecture. Besides, color rebalancing is impressive in terms of boosting the color saturation, as shown in the second and third example.

By observing the quantitative difference, MSE in CIE Lab color space between output image and ground truth, we notice that our model based on EfficientNet doesn't perform a lot better than baseline models, though we have the impression of reduced flaws on output images. Even more, the model with color rebalancing produced more vivid images with much higher MSE than all other models. This suggests us the key metric we used has a disconnect between humans' preferences on this task.

6 Conclusion

With the experiments conducted, we have implemented a system that automatically colorizes images with intuitive improvements from a baseline approach. We also applied the work to some random black-and-white photos out of the original dataset, and observed some pleasant results.

But in terms of quantitative metric, we didn't make considerable progress and even regress. It reinforced us with the importance of correct metrics, as we notice that our key metric, pixel difference between prediction and ground truth, may not be the best fit for image colorization. Even if the model produces a result very close to the real image, it's not immediately true the result will be satisfying for humans. Some papers measure results with surveys conducted to test whether the system could trick the human, but that's not an applicable cost function for optimization. We believe a better metric is necessary to further improve the performance on this task, and this is probably a paper on its own.

7 Contributions

Hanzhao focused on setting up experiments of utilizing Places dataset, EfficientNet, CIELab color space and color rebalancing. Rafael focused on building up data integration with ImageNet, Food-101, setting up ConvNet and VGG-16 models.

References

[1] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.

[2] 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] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., USA, 2nd edition, 2001. ISBN 0201180758.

[4] Tomihisa Welsh, Michael Ashikhmin, and Klaus Mueller. Transferring color to greyscale images. *ACM Trans. Graph.*, 21:277–280, 07 2002. doi: 10.1145/566570.566576.

[5] E. Reinhard, M. Adhikhmin, B. Gooch, and P. Shirley. Color transfer between images. *IEEE Computer Graphics and Applications*, 21(5):34–41, 2001. doi: 10.1109/38.946629.

[6] Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, and David H. Salesin. Image analogies. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, page 327–340, New York, NY, USA, 2001. Association for Computing Machinery. ISBN 158113374X. doi: 10.1145/383259.383295. URL `https://doi.org/10.1145/383259.383295`.

[7] Anat Levin, Dani Lischinski, and Yair Weiss. Colorization using optimization. *ACM Trans. Graph.*, 23(3):689–694, August 2004. ISSN 0730-0301. doi: 10.1145/1015706.1015780. URL `https://doi.org/10.1145/1015706.1015780`.

[8] L. Yatziv and G. Sapiro. Fast image and video colorization using chrominance blending. *IEEE Transactions on Image Processing*, 15(5):1120–1129, 2006. doi: 10.1109/TIP.2005.864231.

[9] Zezhou Cheng, Qingxiong Yang, and Bin Sheng. Deep colorization. *CoRR*, abs/1605.00075, 2016. URL `http://arxiv.org/abs/1605.00075`.

[10] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. *CoRR*, abs/1603.08511, 2016. URL `http://arxiv.org/abs/1603.08511`.

[11] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. *CoRR*, abs/1603.06668, 2016. URL `http://arxiv.org/abs/1603.06668`.

[12] Ryan Dahl. Automatic colorization. `https://tinyclouds.org/colorize/`. Accessed: 2021-05-30.

[13] Yuanzheng Ci, Xinzhu Ma, Zhihui Wang, Haojie Li, and Zhongxuan Luo. User-guided deep anime line art colorization with conditional adversarial networks. *CoRR*, abs/1808.03240, 2018. URL `http://arxiv.org/abs/1808.03240`.

[14] Kamyar Nazeri and Eric Ng. Image colorization with generative adversarial networks. *CoRR*, abs/1803.05400, 2018. URL `http://arxiv.org/abs/1803.05400`.

[15] Manoj Kumar, Dirk Weissenborn, and Nal Kalchbrenner. Colorization transformer. *CoRR*, abs/2102.04432, 2021. URL `https://arxiv.org/abs/2102.04432`.

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL `https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf`.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL `http://arxiv.org/abs/1706.03762`.

[18] 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 conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[19] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

[20] Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv 1409.1556*, 09 2014.

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