Automatic Image Colorization using Generative Adversarial Networks

Team: Yet Another Layer [YAL]
CSci 5561 - Computer Vision

Cameron Fabbri, Md Jahidul Islam

Abstract

Given the massive amounts of free color images available and the successes of convolutional neural networks (CNNs), image colorization has recently become a very active area. Generative models have also gotten much attention due to recent advances in Generative Adversarial Networks (GANs), which have shown to be very powerful generative models. Given a grayscale image as input, we use an adversarial approach towards automatically generating a plausible color output. Despite their success, GANs are notoriously difficult to train. For this reason, we compare the original GAN formation with three recent improvements on adversarial methods applied towards the task of image colorization: Least Squares GANs, Energy-Based GANs, and Wasserstein GANs.

1 Introduction

Image colorization [1, 2, 3] refers to colorizing a given grayscale image so that it appears real. A large amount of photographs, videos and movies, mainly antique, lack color; image colorization can provide a modern and vivid view to these images. In addition, surveillance cameras often capture (or store) gray-scale images for convenience. Several underwater inspection and surveillance applications [4, 5] often have to deal with colorless images due to lack of visible light in deep-water. Robust and efficient image colorization techniques can be used in these applications with substantial benefits.

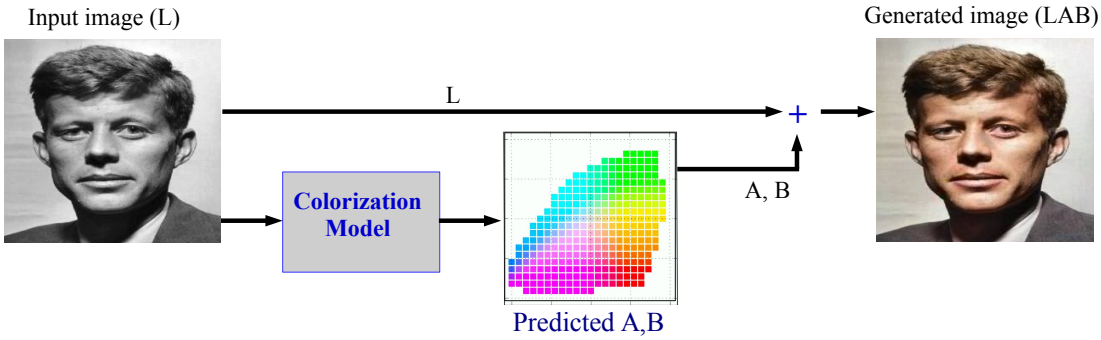


Figure 1: Basic image colorization procedure is shown. LAB color-space is generally used for convenience (*i.e.*, one less unknown dimension); given the lightness channel L, task for the colorization model is to predict A and B channels so that the colorized image appears natural.

Colorizing a grayscale image (*i.e.*, only intensity values are known) is a difficult and ill-posed problem. There have been many approaches to this problem over the last few decades [1, 2, 3, 6, 7, 8]. Before the advent of deep learning [9], researchers have tried many classical techniques [6, 7, 8, 10, 11] to capture relationships between color components (RGB or LAB) and image level features. Due to multi-modality and ill-posed nature of the problem, optimization based techniques [10, 6] and probabilistic models [11] were the only ones that achieved decent colorization performance. However, overall performance of these techniques in general were still poor due to the high non-linearity and abstract nature of color-feature relationship.

Recently, deep-learning based image colorization techniques [1, 2, 12, 13], trained over millions of images, have shown significantly better performance over the earlier classical methods. For instance, the current state-of-the-art, Colorful Image Colorization [1], can fool a human observer 32% of the time in a *colorization Turing-test* scenario. Despite their success, there are still many improvements to be made, as failure cases are still observed by their model. In theory, improperly colorized images should not appear natural, which is why we propose an adversarial approach towards this problem.

2 Background and Related Work

As mentioned previously, image colorization is an ill-posed problem due to multi-modality and ambiguity. While some natural objects commonly hold the same color (e.g. grass is *usually* green), many are left up for interpretation. For example, given a gray-scale image of someone wearing a dark colored shirt, there is no way of figuring out the true color. Instead, the objective is to come up with a colorization that appears real, *i.e.*, natural.

User-based approaches [10, 8, 14, 15] were popular for being fast and relatively accurate as a user can provide a good prior for the inherent color distribution. However, these methods are not applicable for large scale automatic colorization, which led researchers to adopt optimization and probabilistic approaches [6, 3, 11]. These approaches model a likelihood based color approximation for each pixel given the neighborhood information. Few methods introduce additional step for spatial coherency through image based segmentation as well. However, overall colorization performance of these approaches are not very appealing [16] for general usage in a large scale. This is because the prior distribution of color-space is domain-dependant; for instance, face images, underwater images, outdoor and satellite images, all have different color distributions. Besides, it is difficult to capture the highly non-linear and abstract color-feature relationships without large-scale training.

Recently, deep-learning based approaches [1, 2, 12, 13] have produced significantly better colorization performance as they can extract highly non-linear spatial relationships if trained over large datasets. The convolutional layers learn appropriate filters to produce good feature-space representations from raw images. These feature extraction and filtering is performed over multiple layers to capture complex spatial relationships within the image-space, which is useful for image-to-image translation tasks. There have also been adversarial approaches, as seen in [**?**]. We adopt the method of using an adversary in order to find natural looking colorizations, and compare several recent approaches towards training adversarial networks. Furthermore, we compare these results with more traditional loss functions by using the architecture from [1].

3 Generative Approaches

We first experimented with two classical methods: *colorization using optimization* [10], *colorization via multi-modal predictions* [6]. The former is not an automatic approach (user provides color distribution prior), whereas, the latter uses a set of reference images to formulate color distribution. We focus on deep learning approachs, highlighting the Colorful Image Colorization [1] architecture as our base generative architecture.

3.1 Adopted Model: Colorful Image Colorization

Colorful image colorization [1] is considered a major breakthrough on this problem, and has set a new benchmark for performance. The problem is posed as a multi-modal classification problem. The objective function is carefully designed to map the image-to-image translation problem to a classification problem. First, it takes advantage of the fact that the A, B color components of LAB colorspace for natural images are concentrated in a small region, which are discretized into finite number of bins (Q). Given the lightness channel (L_p) of a pixel p, its A, B pair corresponds to a particular bin (out of 313 bins in total), which is mapped to a 1-hot vector (Z_p). Consequently, task of the classification model, is to predict which bin each pixel corresponds to. That is, the output is a 313-mode probability distribution ($\hat{Z}_p$) for each pixel p. The objective function is modeled as a cross entropy loss between Z and $\hat{Z}$, expressed as follows:

$$L_{col}(Z, \hat{Z}) = -\sum_{p} Z_p \sum_{q \in Q} Z_p[q] log(\hat{Z}_p[q])$$

This cross-entropy loss is further augmented with class rebalancing, to encourage rare colors. The detailed model specification, as shown in Fig. **??**, is an 8-layer CNN architecture where each conv layer refers to a block of 2 or 3 repeated conv and ReLU layers [17], followed by a BatchNorm layer [18]. The network has no pool layers; all changes in resolution are achieved through spatial down-sampling or up-sampling between conv blocks. It learns the highly non-linear and abstract color-feature relationships within the image-space and predicts plausible colors for each pixel based on that. The original paper discusses how the colorization is learnt at each layer in addition to handling multi-modality and ambiguity.

3.2 Implementation Details and Model Changes

While working on designing a generator for our GAN-based model, we investigated this model with different objective functions (L_1 loss, L_2 loss, least-squared loss, etc.) instead of their cross-entropy based loss function. This is due to the fact that in a GAN-based model, *discriminator* expects an image from the *generator*, and tries to discriminate it as real or fake in order to force the generator to get better. Therefore, rather than adopting their classification model directly, we implemented their architecture using objective functions based on L_1, L_2,

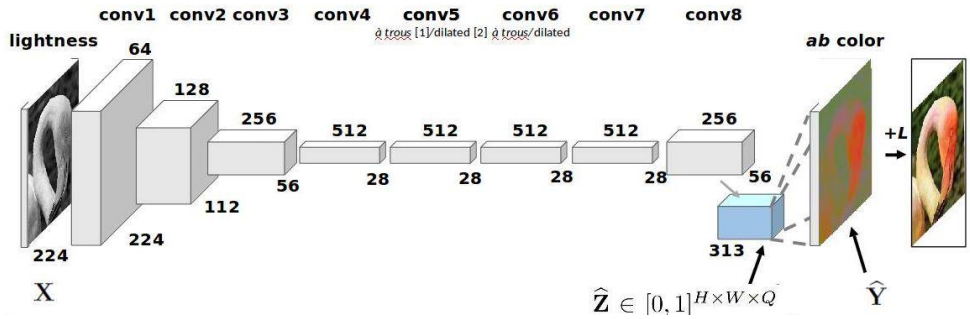


Figure 2: Architecture of colorful image colorization model [1] that is adopted in our evaluation

least-squared loss (so that it outputs an image, not classification probabilities). Additionally, it made our model end-to-end trainable, which can be easily incorporated in a GAN-based architecture.

Given an input image I, the network is fed with its L channel I_L; the output layer of the network is adjusted to predict $\hat{I}_{AB}$. We have found L_1 and L_2 loss functions perform quite well with this model. These loss functions between true I_{AB} and $\hat{I}_{AB}$ can be expressed as follows:

$$L_1(I_{AB}, \hat{I}_{AB}) = \lambda \sum_p |I_{AB}[p] - \hat{I}_{AB}[p]|$$

$$L_2(I_{AB}, \hat{I}_{AB}) = \lambda \sum_p (I_{AB}[p] - \hat{I}_{AB}[p])^2$$

where λ is a normalization constant.

3.3 Results

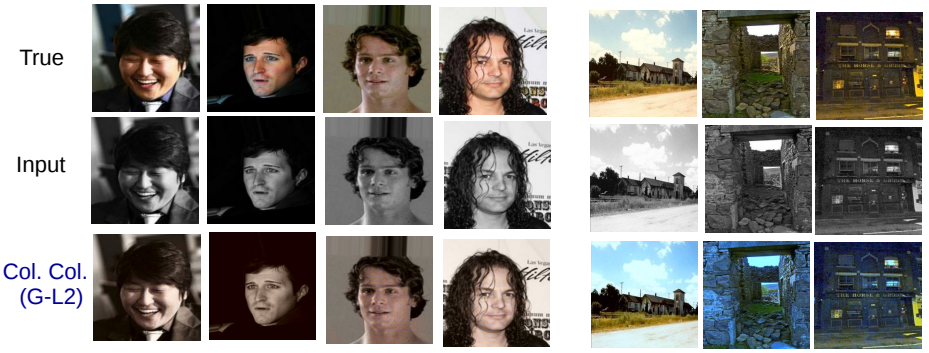
We trained our model on the CelebA dataset [19] as well as the Places2 [20] dataset. CelebA contains about 200,000 images of faces, for which we used 195,000 for training. TensorFlow [22] was used for implementation [1]. We found that this model performs better with L_1 loss function compared to L_2 loss. Results can be seen in Fig. 2 (a)-(b). While able to colorize CelebA reasonably well, the model had a tough time on the backgrounds, usually leaving them gray. The L_2 loss shows an averaging effect as expected, generating very washed out colors. Using this architecture as the generator for a GAN model shows to alleviate some of these issues, as washed out colors should be rejected by the discriminator. The next section discusses various GAN based models for image colorization.

4 Adversarial Approaches

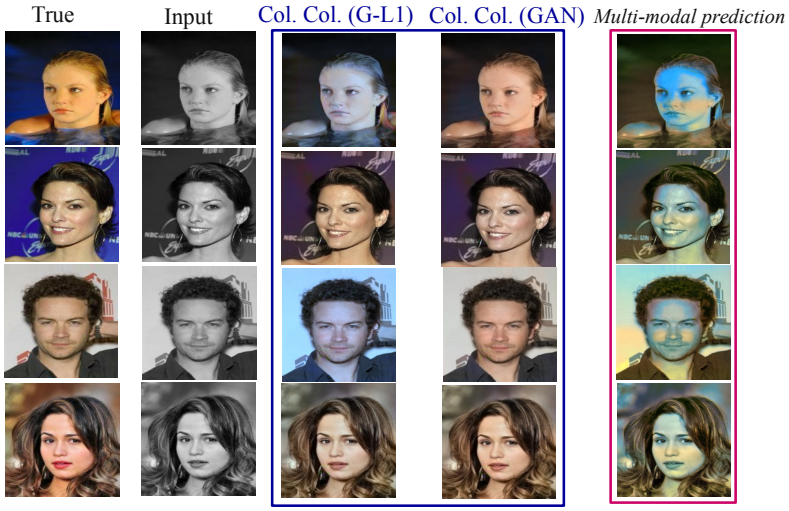
Generative Adversarial Networks (GANs) [**?**] are a recent class of generative models based on game theory in which a generator network is pitted against an adversary. The adversary, or discriminator, D, is a neural network trained to discriminate between real samples and samples generated by the generator. The generator, G, is trained to fool the adversary. Due to the difficulty in training them in practice, there have been several theoretical approaches towards stabalizing their training by providing better gradients for the generator in terms of changing the loss function for the discriminator. Regular GANs modeled the discriminator as a classifier with the sigmoid cross entropy loss function, which has been shown to suffer from the vanishing gradient problem. In addition to regular GANs, we experimented with three different variations: Least Squares GANs (LSGAN), Energy-Based GANs (EBGAN), and Wasserstein GAN (WGAN). There has been little work in using GANs for the task of image colorization. Most notable is the Pix2Pix model which uses the original GANs loss, but also show that often their generator fails to generate any color at all. We hypothesize that in comparison to the common task in which GANs are trained to generate an image from a noise prior, the discriminator for the task of colorization has a much more difficult job due to the visual similarity between images. For this reason, a much stronger discriminator is needed, but as shown in [**?**], with the orinigal GANs loss, as the discriminator gets better, the gradients passed to the generator, and therefore the generator itself, become worse. *This leaves us with both the need for a better discriminator, and proof that a better discriminator may in fact hurt performance.* For this reason we chose to explore other options for the discriminator loss. Our model is set up as a conditional GAN (cGAN), in which the generator is conditioned on the grayscale image. The rest of this section compares

[1] All code can be found at https://github.com/cameronfabbri/Colorizing-Images-Using-Adversarial-Networks

(a) Colorful Colorization [1] model with L1 loss



(b) Colorful Colorization [1] model with L2 loss



(c) Results in comparison with different models

Figure 3: Results for the colorful colorization model [1] with (a) L1 and (b) L2 loss is shown; First 4 columns show few examples from the test set of CelebA [19] dataset, while the rest (3) columns correspond to Places2 [20] dataset. Comparison for colorful colorization model used as a generator only (Col. Col. (G)), and as a generator in a GAN (Col. Col. (GAN)), is shown in (c); also, results obtained by colorization via multi-modal prediction [6] is provided in the last column to demonstrate the degree of improvements using colorful colorization model.

the different GAN formulations, and ends with our architecture designs. We combine the GAN objective with L_1 and L_2 in order to provide some sense of ground truth. These can be expressed as:

$$\mathcal{L}_{L_1}(G) = \mathbb{E}_{x,y\sim p_{data}(x,y)}[||y - G(x)||_1]$$
$$\mathcal{L}_{L_2}(G) = \mathbb{E}_{x,y\sim p_{data}(x,y)}[(y - G(x))^2]$$

where x is the grayscale image, y is the true corresponding ab color channels, and $G(x)$ is the generated ab color channels. These are able to capture low frequencies, but have been shown to lack in capturing high frequencies, resulting in blurry images in the case of autoencoders, and saturated colors in the case of colorization. These imperfections should in theory be rejected by the discriminator, which results in successfully capturing high

frequencies in images such as sharp edges and bright colors. Note that using these losses along with the GAN loss does not change the discriminator's objective, it is the generator that is required to fool the discriminator as well as be near the ground truth.

4.1 DCGANs

Because of the high dimensionality and spacial structure of images, we chose to use Deep Convolutional GANs (DCGANs), which take advantage of the recent successes of CNNs as part of their architecture, as opposed to the simple neural networks used in [**?**]. The objective function is given as:

$$\min_G \max_D \mathbb{E}_{x \sim p_{data(x)}}[logD(\boldsymbol{x})] + \mathbb{E}_{z \sim p_z(z)}[log(1 - D(G(z)))]$$

where p_{data} represents the true data, and p_z represents some noise prior, e.g. $z \sim \mathcal{N}(0, 1)$. Our conditional DCGAN can be expressed as:

$$\mathcal{L}_{DCGAN}(G, D) = \mathbb{E}_{x,y \sim p_{data}} logD(x, y) + \mathbb{E}_{x \sim p_{data}} log(1 - D(x, G(x)))$$

where x represents the grayscale image, and y represents the ab color values associated with x. Combining this with our L_1 and L_2 loss functions gives us our final objective:

$$G^* = \lambda_1 \mathcal{L}_{DCGAN} + \lambda_2 \mathcal{L}_{L_1} + \lambda_3 \mathcal{L}_{L_2}$$

where λ_1, λ_2, and λ_3 are hyperparameters. One should note that z is discarded in this formulation, resulting in a deterministic model. Because of the high volume of information given in the grayscale image in comparison to something like conditioning on a class label, we believe the model would simply learn to discard z as noise (because that's quite literally what it is). Instead, we provide noise in the form of droupout in the generator, but have not seen much difference in the output. We evaluated our method on the CelebA dataset, for which results can be seen in Figure 4.

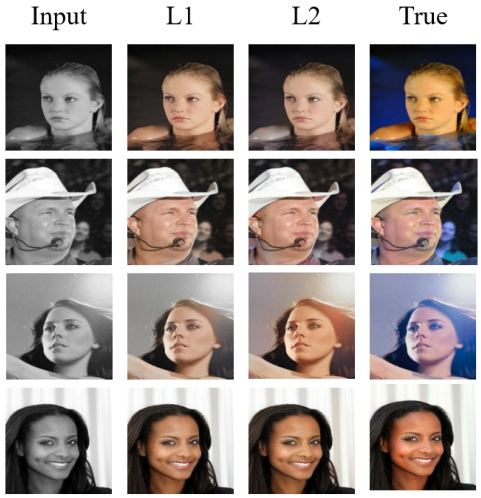


Figure 4: Left: Colorization results using the original GAN loss. Right: Colorization results using LSGANs

4.2 LSGANs

Least Squares GANs (LSGANs) provide a least squares loss for the discriminator in order to penalize samples that may have fooled the discriminator but do not lie close to the true data distribution. The objectives for conditional LSGANs are defined as:

$$\min_D V_{cLSGAN}(D) = \frac{1}{2}\mathbb{E}_{x,y \sim p_{data(x,y)}}[(D(x, y) - b)^2] + \frac{1}{2}\mathbb{E}_{x \sim p_{data}}[(D(G(x)) - a)^2]$$

$$\min_G V_{cLSGAN}(G) = \frac{1}{2}\mathbb{E}_{x \sim p_{data(x)}}[(D(G(x)) - c)^2]$$

where $a = 0$ to denote the fake data, $b = 1$ to denote the true data, and $c = 1$ in order to try and fool D. We combine the LSGAN objective with L_1 and L_2 to give our final objective:

$$G^* = \lambda_1 V_{cLSGAN} + \lambda_2 \mathcal{L}_{L_1} + \lambda_3 \mathcal{L}_{L_2}$$

where λ_1, λ_2, and λ_3 are hyperparameters. We evaluated LSGANs on the CelebA dataset, for which results can be seen in Figure 4.

4.3 EBGANs

Energy-Based GANs (EBGANs) model the discriminator as an energy function that attributes low energies to the regions near the data manifold and higher energies to other regions. Unlike the other GAN variations mentioned, the EBGANs discriminator is modeled as an autoencoder, which aims to provide a more diverse set of outputs as opposed to the binary logistic loss. This also has the unique property of being able to learn the data manifold on its own if given only true samples, whereas the discriminator modeled as a binary classifier would not. The objectives for conditional EBGANs are defined as:

$$\mathcal{L}_{cEBGAN} D(x, y) = D(x, y) + max(0, (m - D(G(z))))$$
$$\mathcal{L}_{cEBGAN} G(x) = D(G(x))$$

where m is some margin, x represents the grayscale image, y represents the corresponding ab color channels for x. We combine the EBGAN objective with L_1 and L_2 to give our final objective:

$$G^* = \lambda_1 \mathcal{L}_{cEBGAN} + \lambda_2 \mathcal{L}_{L_1} + \lambda_3 \mathcal{L}_{L_2}$$

where λ_1, λ_2, and λ_3 are hyperparameters. We evaluated EBGANs on the CelebA dataset. Results can be seen in Figure 5.

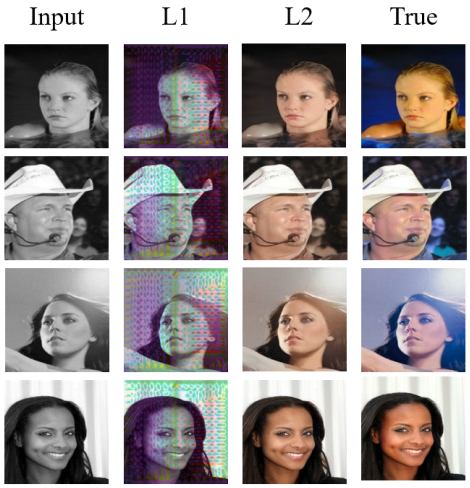


Figure 5: Left: Colorization results using EBGANs. Right: Colorization results using WGAN.

4.4 WGAN

The Wasserstein GAN (WGAN) approximates the Earth Mover (EM) distance given a set of K-Lipschitz functions f. In order to have the parameters w lie in a compact space and ensure f is K-Lipschitz, the weights of the network are clamped to some range. Because the EM distance is continuous and differentiable, the discriminator (critic) should be trained to optimality, which offers better gradients to the generator. This offers stable training at the expense of slow training, because the critic must be updated multiple times for one update of the generator. Our conditional WGAN objective function is:

$$\mathcal{L}_{cWGAN} = \max_{w \in W} \mathbb{E}_{x,y \sim \mathbb{P}_{data}}[f_w(x, y)] - \mathbb{E}_{x \sim p_{data}}[f_w(G(x))]$$

where x represents the grayscale image and y represents the corresponding ab color channels for x. We combine the WGAN objective with L_1 and L_2 to give our final objective:

$$G^* = \lambda_1 \mathcal{L}_{cWGAN} + \lambda_2 \mathcal{L}_{L_1} + \lambda_3 \mathcal{L}_{L_2}$$

where λ_1, λ_2, and λ_3 are hyperparameters. We evaluated WGAN on the CelebA dataset. Some results can be seen in Figure X.

5 Architecture and Training Details

We adopted the architecture seen in the Pix2Pix paper [**?**] for our generator in each GAN variation, which is set up as an encoder-decoder. The discriminator set up as a PatchGAN, which instead of classifying an image as real or fake, classifies a $N \times N$ patch in the image. This allows for a texture style loss given by the discriminator. All GAN variations except EBGANs use the PatchGAN discriminator model, where the discriminator in EBGANs is kept as an autoencoder as described in [**?**].

Early layers in CNNs are able to learn low level features such as edges and corners, which are used in subsequent layers to learn higher level features. As a result, these low level features are somewhat lost as you progress through the network, which for problems such as classification is okay, given the output of the network is ultimately a label. For tasks such as image-to-image translation, where the network's output is visually very similar to the input, these low level features are important. In the case of colorization, the output color image has very similar edges and corners as the input grayscale image. It is for this reason that we choose to use skip connections in their network, which concatenates the activations from layer i to layer $n - i$, where n is the total number of layers. Hyperparameter details are shown in Table 1. All models were trained on the CelebA dataset with batch size 4.

GAN Model	λ_1	λ_2	λ_3	**Learning Rate**	**Epochs**
DCGAN	100.0	0.0	1.0	1e-4	8
DCGAN	0.0	1.0	1.0	2e-5	8
LSGAN	100.0	0.0	1.0	1e-4	4
LSGAN	0.0	1.0	1.0	1e-4	6
EBGAN	100.0	0.0	1.0	1e-3	1
EBGAN	0.0	1.0	1.0	1e-3	1
WGAN	100.0	0.0	1.0	5e-5	2
WGAN	0.0	1.0	1.0	5e-5	3

It is worth noting that although WGAN and EBGANs did not train for nearly as long as DCGAN and LSGAN (due to time constraints), they still provided very competive performance. At only one or two epochs, they show better performance than DCGAN did at four epochs, showing great promise for future work. Although training data is essentially free due to the fact that any color image can be converted to black and white, the resulting image is not a "true" black and white photo. To compare our results, we also tested on true black and white images of John F. Kennedy and Muhammad Ali, seen in Figure 10. Note these images not only weren't trained on, but are completely outside of our dataset.

Figure 6: Top: Results using various GAN models. Bottom: Results using the [**?**] model.

6 Conclusion and Future Work

We investigated automatic image colorization using Generative Adversarial Networks (GANs). We explored few classic algorithms, current state-of-the-art based on deep-learning, and a number of other models based on GANs. We also presented a comparative study of performance of these algorithms which are trained over CelebA dataset. Our GAN-based models provide very good colorization performance and run at real-time (30fps on cpu, 60fps on gpu). In our future work, we want to train our models on larger datasets such as ImageNet. Additionally, we look forward to design a good objective function to boost our models in terms of training time and colorization performance.

References

[1] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European Conference on Computer Vision*, pages 649–666. Springer, 2016.

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

[3] Aurelie Bugeau, Vinh-Thong Ta, and Nicolas Papadakis. Variational exemplar-based image colorization. *IEEE Transactions on Image Processing*, 23(1):298–307, 2014.

[4] Huimin Lu, Yujie Li, and Seiichi Serikawa. Underwater image enhancement using guided trigonometric bilateral filter and fast automatic color correction. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 3412–3416. IEEE, 2013.

[5] Luz A Torres-Méndez and Gregory Dudek. Color correction of underwater images for aquatic robot inspection. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 60–73. Springer, 2005.

[6] Guillaume Charpiat, Matthias Hofmann, and Bernhard Schölkopf. Automatic image colorization via multimodal predictions. *Computer Vision–ECCV 2008*, pages 126–139, 2008.

[7] Qing Luan, Fang Wen, Daniel Cohen-Or, Lin Liang, Ying-Qing Xu, and Heung-Yeung Shum. Natural image colorization. In *Proceedings of the 18th Eurographics conference on Rendering Techniques*, pages 309–320. Eurographics Association, 2007.

[8] Vadim Konushin and Vladimir Vezhnevets. Interactive image colorization and recoloring based on coupled map lattices. In *Graphicon2006 conference proceedings, Novosibirsk Akademgorodok, Russia*, pages 231–234, 2006.

[9] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[10] Anat Levin, Dani Lischinski, and Yair Weiss. Colorization using optimization. In *ACM Transactions on Graphics (ToG)*, volume 23, pages 689–694. ACM, 2004.

[11] Przemyslaw Lagodzinski and Bogdan Smolka. Digital image colorization based on probabilistic distance transformation. In *ELMAR, 2008. 50th International Symposium*, volume 2, pages 495–498. IEEE, 2008.

[12] Domonkos VARGA and Tamás Szirányi. Fully automatic image colorization based on convolutional neural network. *4th Winter School of PhD Students in Informatics and Mathematics*, page 36, 2016.

[13] Jie Li, Katherine A Skinner, Ryan M Eustice, and Matthew Johnson-Roberson. Watergan: Unsupervised generative network to enable real-time color correction of monocular underwater images. *arXiv preprint arXiv:1702.07392*, 2017.

[14] Erik Reinhard, Michael Adhikhmin, Bruce Gooch, and Peter Shirley. Color transfer between images. *IEEE Computer graphics and applications*, 21(5):34–41, 2001.

[15] Michael J Vrhel and HJ Trussell. Color correction using principal components. *Color Research & Application*, 17(5):328–338, 1992.

[16] Aditya Deshpande, Jason Rock, and David Forsyth. Learning large-scale automatic image colorization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 567–575, 2015.

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*, 2016.

[18] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[19] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[20] Celeba dataset. `http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html`. Accessed: 04-20-2017.

[21] Places2 dataset. `http://places2.csail.mit.edu/download.html`. Accessed: 04-20-2017.

[22] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[23] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[24] Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. In *NIPS 2016 Workshop on Adversarial Training. In review for ICLR*, volume 2016, 2017.

[25] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016.