Unsupervised Diverse Colorization via Generative Adversarial Networks

Yun Cao(✉), Zhiming Zhou, Weinan Zhang, and Yong Yu

Shanghai Jiao Tong University, Shanghai, China
{yuncao,heyohai,wnzhang,yyu}@apex.sjtu.edu.cn

Abstract. Colorization of grayscale images is a hot topic in computer vision. Previous research mainly focuses on producing a color image to recover the original one in a supervised learning fashion. However, since many colors share the same gray value, an input grayscale image could be diversely colorized while maintaining its reality. In this paper, we design a novel solution for unsupervised diverse colorization. Specifically, we leverage conditional generative adversarial networks to model the distribution of real-world item colors, in which we develop a fully convolutional generator with multi-layer noise to enhance diversity, with multi-layer condition concatenation to maintain reality, and with stride 1 to keep spatial information. With such a novel network architecture, the model yields highly competitive performance on the open LSUN bedroom dataset. The Turing test on 80 humans further indicates our generated color schemes are highly convincible.

1 Introduction

Image colorization assigns a color to each pixel of a target grayscale image. Early colorization methods [15,21] require users to provide considerable scribbles on the grayscale image, which is apparently time-consuming and requires expertise. Later research provides more automatic colorization methods. Those colorization algorithms differ in the ways of how they model the correspondence between grayscale and color.

Given an input grayscale image, non-parametric methods first define one or more color reference images (provided by a human or retrieved automatically) to be used as source data. Then, following the Image Analogies framework [10], the color is transferred onto the input image from analogous regions of the reference image(s) [4,9,17,24]. Parametric methods, on the other hand, learn prediction functions from large datasets of color images in the training stage, posing the colorization problem as either regression in the continuous color space [3,6,26] or classification of quantized color values [2], which is a supervised learning fashion.

Whichever seeking the reference images or learning a color prediction model, all above methods share a common goal, i.e. to provide a color image closer to the original one. But as we know, many colors share the same gray value. Purely from a grayscale image, one cannot tell what color of clothes a girl is wearing or what color a bedroom wall is. Those methods all produce a deterministic

M. Ceci et al. (Eds.): ECML PKDD 2017, Part I, LNAI 10534, pp. 151–166, 2017.
https://doi.org/10.1007/978-3-319-71249-9_10

Fig. 1. Left: the original grayscale image. Middle: image colorized by non-adversarial CNNs. Right: image colorized by human on Reddit. (Figure from [14])

mapping function. Thus when an item could have diverse colors, their models tend to provide a weighted average brownish color as pointed out in [14] (See Fig. 1 as an example).

In this paper, to avoid this sepia-toned colorization, we use conditional generative adversarial networks (GANs) [8] to generate diverse colorizations for a single grayscale image while maintaining their reality. GAN is originally proposed to generate vivid images from some random noise. It is composed of two adversarial parts: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability of whether an image is real or generated by G. The generator part tries to map an input noise to a data distribution closer to the ground truth data distribution, while the discriminator part tries to distinguish the generated "fake" data, which comes to an adversarial situation. By careful designation of both generative and discriminative parts, the generator will eventually produce results, forming a distribution very close to the ground truth distribution, and by controlling the input noise we can get various results of good reality. Thus conditional GAN is a much more suitable framework to handle diverse colorization than other CNNs. Meanwhile, as the discriminator only needs the signal of whether a training instance is real or generated, which is directly provided without any human annotation during the training phase, the task is in an unsupervised learning fashion.

On the aspect of model designation, unlike many other conditional GANs [12] using convolution layers as the encoder and deconvolution layers as the decoder, we build a fully convolutional generator and each convolutional layer is splinted by a concatenate layer to continuously render the conditional grayscale information. Additionally, to maintain the spatial information, we set all convolution stride to 1 to avoid downsizing data. We also concatenate noise channels to the first half convolutional layers of the generator to attain more diversity in the color image generation process. As the generator G would capture the color distribution, we can alter the colorization result by changing the input noise. Thus we no longer need to train an additional independent model for each color scheme like [3].

As our goal alters from producing the original colors to producing realistic diverse colors, we conduct questionnaire surveys as a Turing test instead of

calculating the root mean squared error (RMSE) comparing the original image to measure our colorization result. The feedback from 80 subjects indicates that our model successfully produces high-reality color images, yielding more than 62.6% positive feedback while the rate of ground truth images is 70.0%. Furthermore, we perform a significance t-test to compare the percentages of human judges as real color images for each test instance (i.e. a real or generated color image). The resulting p-value is $0.1359 > 0.05$, which indicates that there is no significant difference between our generated color images and the real ones. We share the repeatable experiment code for further research[1].

2 Related Work

2.1 Diverse Colorization

The problem of colorization was proposed in the last century, but the research of diverse colorization was not paid much attention until this decade. In [3], they used additionally trained model to handle diverse colorization of a scene image particularly in day and dawn. [26] posed the colorization problem as a classification task and use class re-balancing at training time to increase the colorfulness of the result. And in the work of [5], a low dimensional embedding of color fields using a variational auto-encoder (VAE) is learned. They constructed loss terms for the VAE decoder that avoid blurry outputs and take into account the uneven distribution of pixel colors and finally developed a conditional model for the multi-modal distribution between gray-level image and the color field embeddings.

Compared with above work, our solution uses conditional generative adversarial networks to achieve unsupervised diverse colorization in a generic way with little domain knowledge of the images.

2.2 Conditional GAN

Generative adversarial networks (GANs) [8] have attained much attention in unsupervised learning research during the recent 3 years. Conditional GANs have been widely used in various computer vision scenarios. [22] used text to generate images by applying adversarial networks. [12] provided a general-purpose image-to-image translation model that handles tasks like label to scene, aerial to map, day to night, edges to photo and also grayscale to color.

Some of the above works may share a similar goal with us, but our conditional GAN structure differs a lot from previous work in several architectural choices mainly for the generator. Unlike other generators which employ an encoder-like front part consisting of multiple convolution layers and a decoder-like end part consisting of multiple deconvolution layers, our generator uses only convolution layers all over the architecture, and does not downsize data shape by applying convolution stride no more than 1 and no pooling operation. Additionally, we add

[1] Experiment code is available at https://github.com/ccyyatnet/COLORGAN.

multi-layer noise to generate more diverse colorization, while using multi-layer conditional information to keep the generated image highly realistic.

3 Methods

3.1 Problem Formulation

GANs are generative models that learn a mapping from random noise vector $\boldsymbol{z}$ to an output color image $\boldsymbol{x}$: $G : \boldsymbol{z} \rightarrow \boldsymbol{x}$. Compared with GANs, conditional GANs learn a mapping from observed grayscale image $\boldsymbol{y}$ and random noise vector $\boldsymbol{z}$, to $\boldsymbol{x}$: $G : \{\boldsymbol{y}, \boldsymbol{z}\} \rightarrow \boldsymbol{x}$. The generator G is trained to produce outputs that cannot be distinguished from "real" images by an adversarially trained discriminator D, which is trained with the aim of detecting the "fake" images produced by the generator. This training procedure is illustrated in Fig. 2.

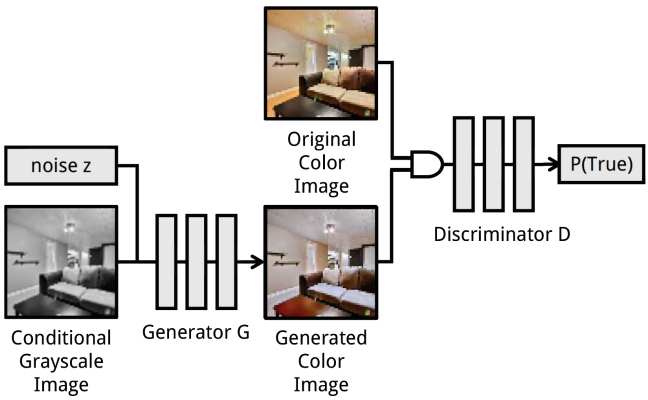

Fig. 2. The illustration of conditional GAN. Generator G is given conditional information (Grayscale image) together with noise $\boldsymbol{z}$, and produces generated color channels. Discriminator D is trained over the real color image and the generated color image by G. The goal of D is to distinguish real images from the fake ones. Both nets are trained adversarially. (Color figure online)

The objective of a GAN can be expressed as

$$\begin{aligned}\mathcal{L}_{\mathrm{GAN}}(G, D) =& \mathbb{E}_{\boldsymbol{x} \sim P_{\mathrm{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] \\ &+ \mathbb{E}_{\boldsymbol{z} \sim P_z(\boldsymbol{z})}[\log (1 - D(G(\boldsymbol{z})))],\end{aligned} \tag{1}$$

while the objective of a conditional GAN is

$$\begin{aligned}\mathcal{L}_{\mathrm{cGAN}}(G, D) =& \mathbb{E}_{\boldsymbol{x} \sim P_{\mathrm{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] \\ &+ \mathbb{E}_{\boldsymbol{y} \sim P_{\mathrm{gray}}(\boldsymbol{y}), \boldsymbol{z} \sim P_z(\boldsymbol{z})}[\log (1 - D(G(\boldsymbol{y}, \boldsymbol{z})))],\end{aligned} \tag{2}$$

where G tries to minimize this objective against an adversarial D that tries to maximize it, i.e.

$$G^* = \arg\min_G \max_D \mathcal{L}_{\text{cGAN}}(G, D). \tag{3}$$

Without $\boldsymbol{z}$, the generator could still learn a mapping from $\boldsymbol{y}$ to $\boldsymbol{x}$, but would produce deterministic outputs. That is why GAN is more suitable for diverse colorization tasks than other deterministic neural networks.

3.2 Architecture and Implementation Details

The high-level structure of our conditional GAN is consistent with traditional ones [5,12], while the detailed architecture of our generator G differs a lot.

Convolution or deconvolution. Convolution and deconvolution layers are two basic components of image generators. Convolution layers are mainly used to exact conditional features. And additionally, many researches [5,12,26] use superposition of multiple convolution layers with stride more than 1 to downsize the data shape, which works as a data encoder. Deconvolution layers are then used to upsize the data shape as a decoder of the data representation [5,12,18]. While many other researches share this encoder-decoder structure, we choose to use only convolution layers in our generator G. Firstly, convolution layers are well capable of feature extraction and transmission. Meanwhile, all the convolution stride is set to 1 to prevent data shape from downsizing, thus the important spatial information can be kept along the data flow till the final generation layer. Some other researches [12,23] also takes this spatial information into consideration. They add skip connections between each layer i and layer $n - i$ to form a "U-Net" structure, where n is the total number of layers. Each skip connection simply concatenates all channels at layer i with those at layer $n - i$. Whether adding skip connections or not, the encoder-decoder structure more tends to extract global features and generate images by this overall information which is more suitable for global shape transformation tasks. But in image colorization, we need a very detailed spatial local guidance to make sure item boundaries will be accurately separated by different color parts in generated channels. Let alone our modification is more straightforward and easy to implement. See the structural difference between "U-Net" and our convolution model in Fig. 3.

YUV or RGB. A color image can be represented in different forms. The most common representation is RGB form which splits a color pixel into red, green, blue three channels. Most computer vision tasks use RGB representation [6,12, 19] due to its generality. Other kinds of representations are also included like YUV (or $YCrCb$) [3] and Lab [2,16,26]. In colorization tasks, we have grayscale image as conditional information, thus it is straightforward to use YUV or Lab representation because the Y and L channel or so called Luminance channel represents exactly the grayscale information. So while using YUV representation, we can just predict 2 channels and then concatenate with the grayscale channel to give a full color image. Additionally, if you use RGB as image representation,

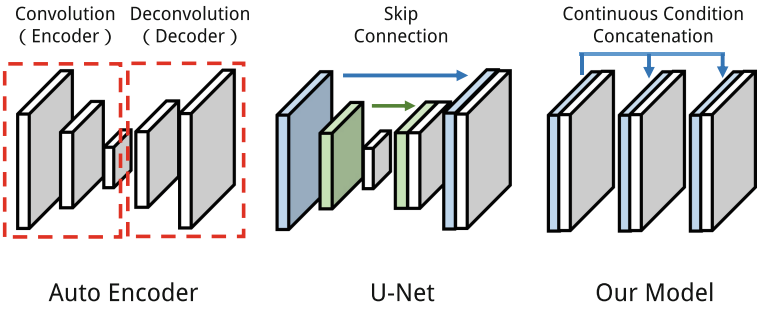

Fig. 3. Structure comparison of auto encoder, U-Net and our generator. Left: auto encoder with convolutional encoder part and deconvolutional decoder part. Middle: U-Net structure [12,23] with skip connections between layer i and $n-i$. Right: our fully convolutional generator with continuous condition concatenation

all result channels are predicted, thus to keep the grayscale of generated color image consistent with the original grayscale image, we need to add an additional $L1$ loss as a controller to make sure $Gray(G(\boldsymbol{y}, \boldsymbol{z})) \simeq \boldsymbol{y}$:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{\boldsymbol{y} \sim P_{\text{gray}}(\boldsymbol{y}), \boldsymbol{z} \sim P_z(\boldsymbol{z})}[\|\boldsymbol{y} - Gray(G(\boldsymbol{y}, \boldsymbol{z}))\|], \tag{4}$$

where for any color image $\boldsymbol{x} = (\boldsymbol{r}, \boldsymbol{g}, \boldsymbol{b})$, the corresponding grayscale image (or the Luminance channel Y) can be calculated by the well-known psychological formulation:

$$Gray(\boldsymbol{x}) = 0.299\boldsymbol{r} + 0.587\boldsymbol{g} + 0.114\boldsymbol{b}. \tag{5}$$

Note that Eq. (4) can still maintain good colorization diversity, because this $L1$ loss term only lays a constraint on one dimension out of three channels. Then the objective function will be modified to:

$$G^* = \arg\min_G \max_D \mathcal{L}_{\text{cGAN}}(G, D) + \lambda \mathcal{L}_{L1}(G). \tag{6}$$

Since there is no equality constraint between the recovered grayscale $Gray(G(\boldsymbol{y}, \boldsymbol{z}))$ and the original one $\boldsymbol{y}$, the $\mathcal{L}_{L1}(G)$ factor will normally be non-zero, which makes the training unstable due to this additional trade-off. The results will be shown in Sect. 4.2 with experimental comparison on both RGB and YUV representations.

Multi-layer noise. The authors in [12] mentioned noise ignorance while training the generator. To handle this problem, they provide noise only in the form of dropout, applied on several layers of the generator at both training and test time. We also noticed this problem. Traditional GANs and conditional GANs receive noise information at the very start layer, during the continuous data transformation through the network, the noise information is attenuated a lot.

To overcome this problem and make the colorization results more diversified, we concatenate the noise channel onto the first half of the generator layers (the first three layers in our case). We conduct experimental comparison on both one-layer noise and multi-layer noise representations, with results shown in Sect. 4.2.

Multi-layer conditional information. Other conditional GANs usually add conditional information only in the first layer, because the layer shape of previous generators changes along their convolution and deconvolution layers. But due to the consistent layer shape of our generator, we can apply concatenation of conditional grayscale information throughout the whole generator layers which can provide sustained conditional supervision. Though the "U-Net" skip structure of [12] can also help posterior layers receive conditional information, our model modification is still more straightforward and convenient.

Wasserstein GAN. The recent work of Wasserstein GAN [1] has acquired much attention. The authors used Wasserstein distance to help getting rid of problems in original GANs like mode collapse and gradient vanishing and provide a measurable loss to indicate the progress of GAN training. We also try implementing Wasserstein GAN modification into our model, but the results are no better than our model. We make comparison between the results of Wasserstein GAN and our GAN in Sect. 4.2.

The illustration of our model structure is shown in Fig. 4.

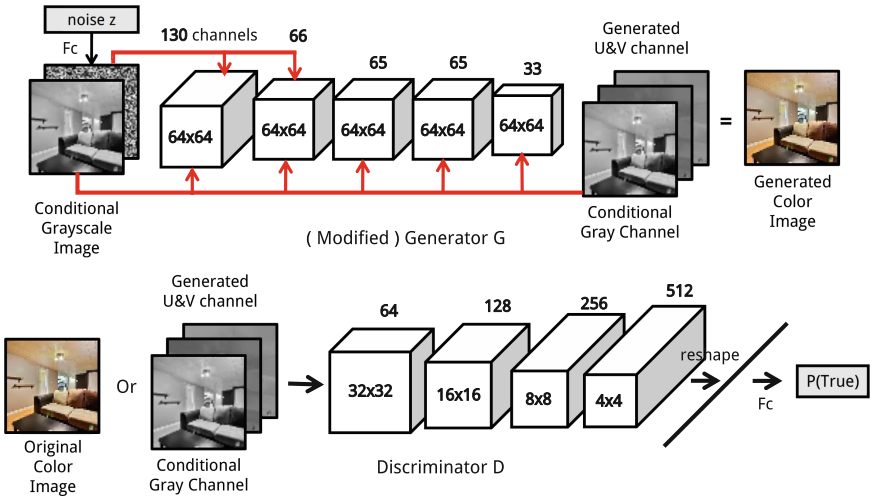

Fig. 4. Detailed structure of our conditional GAN. Top: generator G. Each cubic part represents a Convolution-BatchNorm-ReLU structure. Red connections represent our modifications of the traditional conditional GANs. Bottom: discriminator D (Color figure online)

Algorithm 1. Training phase of our conditional GANs, with the default parameters $k_D = 1, k_G = 1, m = 64, s_z = 100, s = 64$.

for the number of training iterations **do**
  for k_D steps **do**
    Generate minibatch of m randomly sampled noise $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ each of size $[s_z]$.
    Sample minibatch of m grayscale images $\{\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(m)}\}$ each of shape $[s, s, 1]$.
    Get the corresponding minibatch of m color images $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data distribution $p_{\text{data}}(\boldsymbol{x})$.
    Update the discriminator D by:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} [\log D(\boldsymbol{x}^{(i)}) + \log (1 - D(G(\boldsymbol{y}^{(i)}, \boldsymbol{z}^{(i)})))]$$

  end for
  for k_G steps **do**
    Generate minibatch of m randomly sampled noise $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$
    Sample minibatch of m grayscale images $\{\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(m)}\}$
    Update the generator by:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log(1 - D(G(\boldsymbol{y}^{(i)}, \boldsymbol{z}^{(i)})))$$

  end for
end for

Algorithm 2. Testing phase of our conditional GANs with the default parameters $m = 64, s_z = 100$.

Sample minibatch of m grayscale images $\{\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(m)}\}$
for round i in k_{test} rounds **do**
  Generate minibatch of m randomly sampled noise $\{\boldsymbol{z}^{(1,i)}, \ldots, \boldsymbol{z}^{(m,i)}\}$ each of size $[s_z]$.
  Generate color image using the trained model G:

$$\{\boldsymbol{x}^{(1,i)}, \ldots, \boldsymbol{x}^{(m,i)}\} \leftarrow G(\{\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(m)}\}, \{\boldsymbol{z}^{(1,i)}, \ldots, \boldsymbol{z}^{(m,i)}\})$$

end for
Rearrange generated results $\boldsymbol{x}$ into $\{\boldsymbol{x}^{(1,1)}, \ldots, \boldsymbol{x}^{(1,k_{\text{test}})}\}, \ldots, \{\boldsymbol{x}^{(m,1)}, \ldots, \boldsymbol{x}^{(m,k_{\text{test}})}\}$

3.3 Training and Testing Procedure

The training phase of our conditional GANs is presented in Algorithm 1. To assure the BatchNorm layers to work correctly, one cannot feed an image batch of the same images to test various noise responses. Thus we use multi-round testing with same batch and rearrange them to test different noise responses of each image, which is described in Algorithm 2.

4 Experiments

4.1 Dataset

There are various kinds of color image datasets, and we choose the open LSUN bedroom dataset[2] [25] to conduct our experiment. LSUN is a large color image dataset generated iteratively by human labeling with automatic deep neural classification. It contains around one million labeled images for each of 10 scene categories and 20 object categories. Among them we choose indoor scene bedroom because it has enough samples (more than 3 million) and unlike outdoor scenes in which trees are almost always green and sky is always blue, items in indoor scenes like bedroom have various colors, as shown in Fig. 5. This is exactly what we need to fully explore the capability of our conditional GAN model. In experiments, we use 503,900 bedroom images randomly picked from the LSUN dataset. We crop a maximum center square out of each image and reshape them into 64×64 pixel as preprocessing.

Fig. 5. Demonstration of LSUN dataset. Top: outdoor scene (church). Always blue sky and green trees. Bottom: indoor scene (bedroom). Various item colors. (Color figure online)

4.2 Comparison Experiments

YUV and RGB. The generated colorization results of a same grayscale image using YUV representation and RGB representation with additional $L1$ loss are shown in Fig. 6. Each group of images consists of 3×4 images generated from a same grayscale image by each model at a same epoch. Focus on the results in red boxes, we can see RGB representation suffers from structural miss due to the additional trade-off between $L1$ loss and the GAN loss. Take the enlarged image on the top right in Fig. 6 as an example, both the wall and the bed on the left are split by unnaturally white and orange colors, while the results of YUV setting acquire more smooth transitions. Moreover, the model using RGB representation

[2] LSUN dataset is available at http://lsun.cs.princeton.edu.

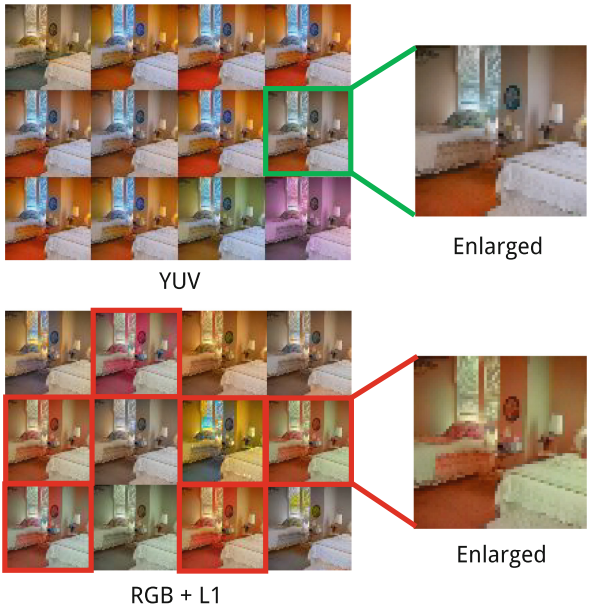

Fig. 6. Comparison of different color space representation. Top: training and testing with YUV representation. Bottom: training with RGB representation and $L1$ loss. Focus on the results in red boxes, RGB representation results lack of item continuity due to the additional trade-off between $L1$ loss and the GAN loss. (Color figure online)

shall predict 3 color channels while YUV representation only predicts 2 channels given the grayscale Y channel as fixed conditional information, which makes the YUV model training much more stable.

Single-layer and multi-layer noise. The generated colorization results of the same grayscale images using single-layer noise model and multi-layer noise model are shown in Fig. 7. Each group consists of 8×8 images generated from a grayscale image by each model at the same epoch. We can see from the results that multi-layer noise possesses our generator G of higher diversity as those results on the right in Fig. 7 are apparently more colorful.

Single-layer and multi-layer condition. The generated colorization results of the same grayscale images using single-layer conditional model and multi-layer conditional model are shown in Fig. 8. We show 2×5 images generated by single-layer condition setting and multi-layer condition setting at the same epoch. We can see from the results that the multi-layer condition model makes the generator more structural information and thus the results of multi-layer condition model are more stable while the single-layer conditional model suffers from colorization derivation like those images in red boxes in Fig. 8.

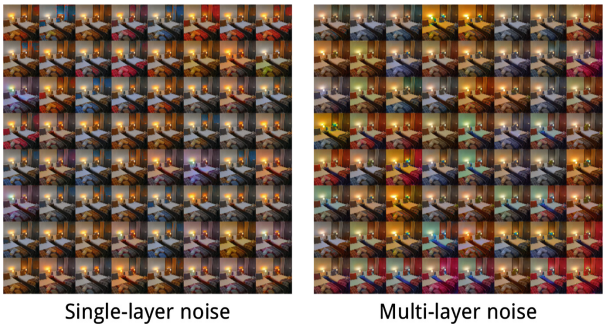

Fig. 7. Comparison of single-layer and multi-layer noise model results. Left: results of single-layer noise model. Right: results of multi-layer noise model. Apparently multi-layer noise possesses our generator G of higher diversity.

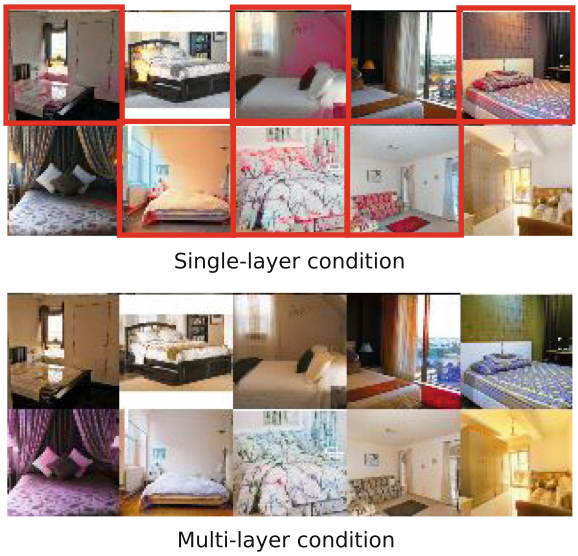

Fig. 8. Comparison of single-layer and multi-layer condition model result. Top: results of single-layer condition model, suffer from colorization derivation (red box). Bottom: results of multi-layer condition model, smooth transition. (Color figure online)

Wasserstein GAN. Three groups of colorization results of the same grayscale images using GAN and Wasserstein GAN are shown in Fig. 9. We can see from the result that Wasserstein GAN can produce comparable results as the first two column of Wasserstein GAN shows, but there are still failed results by Wasserstein GAN like the last column, even to note that the Wasserstein GAN results (40 epoch) come after training twice the number of epochs of the GAN results (20 epoch). This is mainly due to that our training LSUN bedroom dataset is quite large (503,900 image), the discriminator will not overfit easily, which pre-

vents the gradient vanishing problem. And also because of the large dataset, the discriminator needs quite a lot times of optimization to convergence, not to mention Wasserstein GAN shall not use momentum based optimization strategies like Adam due to the non-linear parameter value clipping, Wasserstein GAN needs much longer training to produce comparable results as our model. Since Wasserstain GAN only helps to improve the stability of GAN training at a price of much longer training time and we have achieved results of good reality through our GAN, we did not use Wassserstein GAN structure.

Fig. 9. Comparison between the results of GAN and Wasserstein GAN. Each line consists of the leftmost ground truth color image and three results by GAN, then three results by Wasserstein GAN. The rightmost images are failed results by Wasserstein GAN. (Color figure online)

More results and discussion of our final model will be shown in the next section.

5 Results and Evaluation

5.1 Colorization Results

A variety of image colorization results by our conditional GANs are provided in Fig. 10. Apparently, our fully convolutional (without stride) generator with multi-layer noise and multi-layer condition concatenation produces various kinds of colorization schemes while maintaining good reality. Almost all color parts are kept within correct components without deviation.

5.2 Evaluation via Human Study

Previous methods share a common goal to provide a color image closer to the original one. That is why many of their models [5,13,20] take image distance

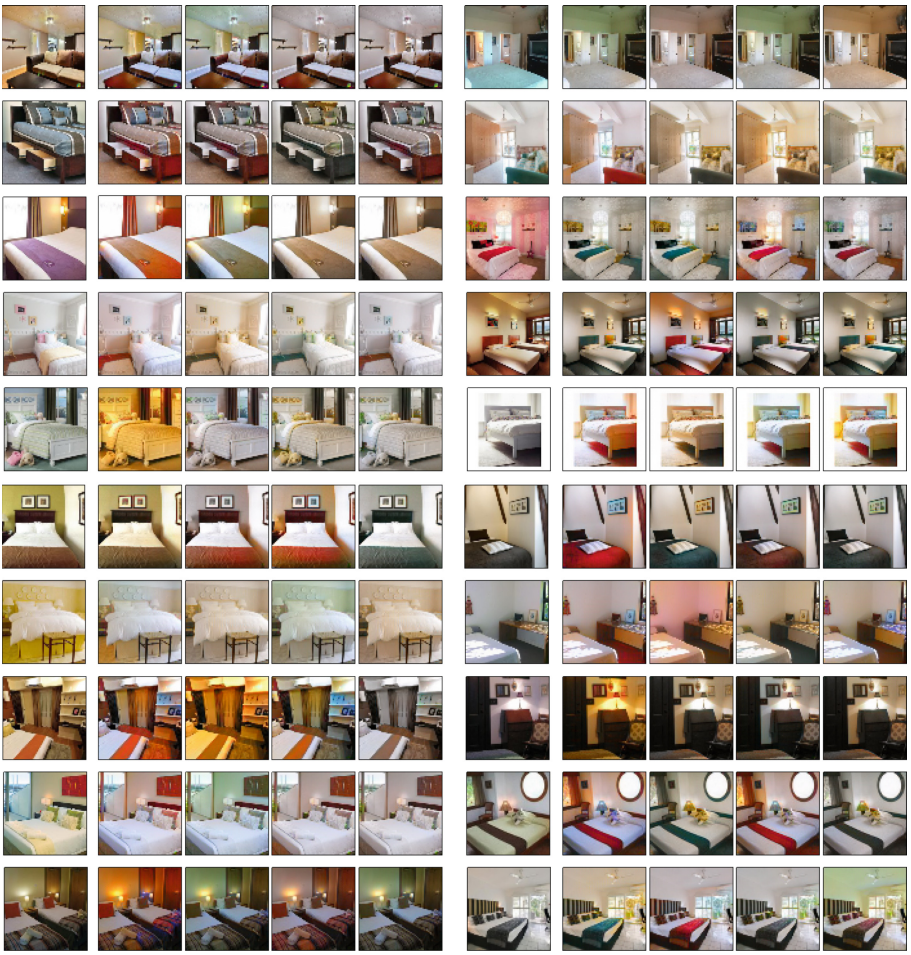

Fig. 10. Example results of our conditional GAN on LSUN bedroom data. 20 groups of images, each consists of the **leftmost** ground truth color image and 4 different colorizations generated by our conditional GANs given the grayscale version of the ground truth image. One can clearly see that our novel structure generator produces various colorization schemes while maintaining good reality. (Color figure online)

like RMSE (Root Mean Square Error) and PSNR (Peak Signal-to-Noise Ratio) as their measurements. And others [11,12] use additional classifiers to predict if colorized image can be detected or still correctly classified. But our goal is to generate diverse colorization schemes, so we cannot take those distance as our measurements as there exist reasonable colorizations that diverge a lot from the original color image. Note that some previous work on image colorization [3,7,18,19] does not provide quantified measurements.

Therefore, just like some previous researches [12,26], we provide questionnaire surveys as a Turing test to measure our colorization results. We ask each of the total 80 participants 20 questions. In each question, we display 5 color images, one of which is the ground truth image, the others are our generated colorizations of the grayscale image of the ground truth, and ask them if any one of them is of poor reality. We add ground truth image among them as a reference in case of participants do not think any of them is real. And we arrange all images randomly to avoid any position bias for participants. The feedback from 80 participants indicates more than 62.6% of our generated color images are convincible while the rate of ground truth images is 70.0%. Furthermore, we run significance t-test between the ground truth and the generated images on the percentages of humans rating as real image for each question. The p-value is $0.1359 > 0.05$, indicating our generated results have no significant difference with the ground truth images. Also we calculate the credibility rank within each group of the ground truth image and the four corresponding generated images. An image gets higher rank if higher percentage of participants mark it real. And the average credibility rank of the ground truth images is only 2.5 out of 5, which means at least $(2.5-1)/(5-1) = 37.5\%$ of our generated results are even more convincible than true images.

6 Conclusion

In this paper, we proposed a novel solution to automatically generate diverse colorization schemes for a grayscale image while maintaining their reality by exploiting conditional generative adversarial networks which not only solved the sepia-toned problem of other models but also enhanced the colorization diversity. We introduced a novel generator architecture which consists of fully convolutional non-stride structure with multi-layer noise to enhance diversity and multi-layer condition concatenation to maintain reality. With this structure, our model successfully generated diversified high-quality color images for each input grayscale image. We performed a questionnaire survey as a Turing test to evaluate our colorization result. The feedback from 80 participants indicates our generated colorization results are highly convincible.

For future work, as so far we have investigated methods to generate color images by conditional GAN given only corresponding grayscale images, which provides the model maximum freedom to generate all kinds of colors, we can also lay additional constraints on the generator to guide the colorization procedure. Those conditions include but not limited to (i) specified item color, such as blue bed and white wall etc.; and (ii) global color scheme, such as warm tone or cool tone etc. And note that given those constraints, generative adversarial networks shall still produce various vivid colorizations.

References

1. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein GAN. CoRR abs/1701.07875 (2017)
2. Charpiat, G., Hofmann, M., Schölkopf, B.: Automatic image colorization via multimodal predictions. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008. LNCS, vol. 5304, pp. 126–139. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88690-7_10
3. Cheng, Z., Yang, Q., Sheng, B.: Deep colorization. In: ICCV 2015, Santiago, Chile, 7–13 December 2015, pp. 415–423 (2015)
4. Chia, A.Y.S., Zhuo, S., Gupta, R.K., Tai, Y.W., Cho, S.Y., Tan, P., Lin, S.: Semantic colorization with internet images. ACM Trans. Graph. **30**(6), 156:1–156:8 (2011)
5. Deshpande, A., Lu, J., Yeh, M.C., Forsyth, D.A.: Learning diverse image colorization. CoRR abs/1612.01958 (2016)
6. Deshpande, A., Rock, J., Forsyth, D.A.: Learning large-scale automatic image colorization. In: ICCV 2015, Santiago, Chile, 7–13 December 2015, pp. 567–575 (2015)
7. Dong, H., Neekhara, P., Wu, C., Guo, Y.: Unsupervised image-to-image translation with generative adversarial networks. CoRR abs/1701.02676 (2017)
8. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A.C., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, 8–13 December 2014, Montreal, Quebec, Canada, pp. 2672–2680 (2014). http://papers.nips.cc/paper/5423-generative-adversarial-nets
9. Gupta, R.K., Chia, A.Y.S., Rajan, D., Ng, E.S., Huang, Z.: Image colorization using similar images. In: Proceedings of the 20th ACM Multimedia Conference, MM 2012, Nara, Japan, 29 October–02 November 2012, pp. 369–378 (2012)
10. Hertzmann, A., Jacobs, C.E., Oliver, N., Curless, B., Salesin, D.: Image analogies. In: SIGGRAPH 2001, Los Angeles, California, USA, 12–17 August 2001, pp. 327–340 (2001)
11. Iizuka, S., Simo-Serra, E., Ishikawa, H.: 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), 110:1–110:11 (2016)
12. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. CoRR abs/1611.07004 (2016)
13. Jung, M., Kang, M.: Variational image colorization models using higher-order mumford-shah regularizers. J. Sci. Comput. **68**(2), 864–888 (2016)
14. Koo, S.: Automatic colorization with deep convolutional generative adversarial networks (2016)
15. Levin, A., Lischinski, D., Weiss, Y.: Colorization using optimization. ACM Trans. Graph. **23**(3), 689–694 (2004)
16. Limmer, M., Lensch, H.P.A.: Infrared colorization using deep convolutional neural networks. In: ICMLA 2016, Anaheim, CA, USA, 18–20 December 2016, pp. 61–68 (2016)
17. Liu, X., Wan, L., Qu, Y., Wong, T.T., Lin, S., Leung, C.S., Heng, P.A.: Intrinsic colorization. ACM Trans. Graph. **27**(5), 152:1–152:9 (2008)
18. Nguyen, T.D., Mori, K., Thawonmas, R.: Image colorization using a deep convolutional neural network. CoRR abs/1604.07904 (2016)
19. Nguyen, V., Sintunata, V., Aoki, T.: Automatic image colorization based on feature lines. In: VISIGRAPP 2016, VISAPP, Rome, Italy, 27–29 February 2016, vol. 4, pp. 126–133 (2016)

20. Perarnau, G., van de Weijer, J., Raducanu, B., lvarez, J.M.A.: Invertible conditional gans for image editing. CoRR abs/1611.06355 (2016)
21. Qu, Y., Wong, T.T., Heng, P.A.: Manga colorization. ACM Trans. Graph. **25**(3), 1214–1220 (2006)
22. Reed, S.E., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H.: Generative adversarial text to image synthesis. In: ICML 2016, New York City, NY, USA, 19–24 June 2016, pp. 1060–1069 (2016)
23. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: Proceedings of the MICCAI 2015–18th International Conference Munich, Germany, 5–9 October 2015, Part III, pp. 234–241 (2015)
24. Welsh, T., Ashikhmin, M., Mueller, K.: Transferring color to greyscale images. In: SIGGRAPH 2002, San Antonio, Texas, USA, 23–26 July 2002, pp. 277–280 (2002)
25. Yu, F., Zhang, Y., Song, S., Seff, A., Xiao, J.: LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. CoRR abs/1506.03365 (2015)
26. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 649–666. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_40