2020 25th International Conference on Pattern Recognition (ICPR)
Milan, Italy, Jan 10-15, 2021

Stylized-Colorization for Line Arts

Tzu-Ting Fang
National Tsing Hua University
Hsinchu, Taiwan
s106062544@m106.nthu.edu.tw

Duc Minh Vo
SOKENDAI
Tokyo, Japan
vmduc@nii.ac.jp

Akihiro Sugimoto
National Institute of Informatics
Tokyo, Japan
sugimoto@nii.ac.jp

Shang-Hong Lai
National Tsing Hua University
Hsinchu, Taiwan
lai@cs.nthu.edu.tw

***Abstract*—We address a novel problem of stylized-colorization which colorizes a given line art using a given coloring style in text. This problem can be stated as multi-domain image translation and is more challenging than the current colorization problem because it requires not only capturing the illustration distribution but also satisfying the required coloring styles specific to anime such as lightness, shading, or saturation. We propose a GAN-based end-to-end model for stylized-colorization where the model has one generator and two discriminators. Our generator is based on the U-Net architecture and receives a pair of a line art and a coloring style in text as its input to produce a stylized-colorization image of the line art. Two discriminators, on the other hand, share weights at early layers to judge the stylized-colorization image in two different aspects: one for color and one for style. One generator and two discriminators are jointly trained in an adversarial and end-to-end manner. Extensive experiments demonstrate the effectiveness of our proposed model.**

I. Introduction

Anime character illustration consists of three main steps: (i) drawing a sketch of a character, (ii) drawing a line art of the character, i.e., outline of the character in black and white, from the sketch, and (iii) colorizing the line art. Inspired by the coloring practice table available for the online artist community, the colorizing step is more than just filling with the color. Using different ways to colorize, one line art creates (colorized) illustrations in different styles, which we call "coloring styles".

Five typical coloring styles are shown in Fig. 1. If we use the "watercolor" style for example, we see that the saturation in colorized illustration is lower and the neighboring colors are more blended together, giving different impression to the same character, compared with the case of using the "anime" style. Accordingly, coloring styles takes an important role in anime character illustration. Different coloring styles are popularly used even for the same character depending on situations and scenarios.

We aim at colorizing line arts in coloring styles given in text. This task is challenging because it has two aspects: colorization and style transfer. The outline of the character in a line art is sparse unlike images, and, therefore, existing colorization methods developed for gray-scale images [1], [2], [3] or for comic images [4], [5] do not work so well. Moreover, colorization taking into account properties specific to anime is required. The difference from style transfer is that the structure of the outline of a character should be preserved without distortion. Methods for style transfer [6], [7], [8], [9] usually bring distortion of object contours to reflect painting styles. Moreover, we specify colorizing styles by texts unlike style transfer where styles are given as images.

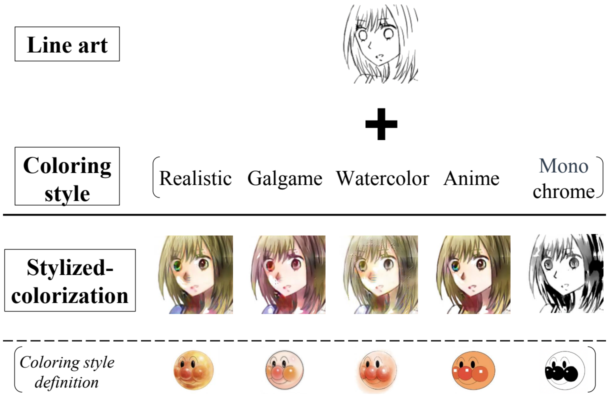

Fig. 1: Example of stylized-colorization.

We propose a GAN-based end-to-end model for colorizing line arts in coloring styles. Our model consists of one generator and two discriminators. Our generator is designed using the U-Net architecture and receives a pair of a line art and a coloring style in text as its input to produce a stylized-colorization image of the line art. Two discriminators are designed to share weights at early layers to judge the stylized-colorization image in two different aspects: one for color and one for style. One generator and two discriminators are jointly trained in an adversarial and end-to-end manner. Extensive experiments show that our proposed method outperforms the state-of-the-art image translation methods and style transfer methods.

II. Related work

Image-to-image translation: Pix2pix [10] is one of the most famous GAN-based image-to-image models. It learns the mapping from the input domain to the output domain through paired data, and can handle many image translating problems such as converting labeled segmentation images to scenes or transferring edges to realistic photos. Paired training data, however, are not easily available in many situations. CycleGAN [11] was then proposed for unpaired image translation. StarGAN [12] and AsymmetricGAN [13] were later proposed for multi-domain image translations using only a single model. Such models are flexible to translate an input image to any desired target domain.

Automatic/Semi-automatic colorization: Methods relying on GANs for automatic colorization have shown remarkable

978-1-7281-8808-9/20/$31.00 ©2020 IEEE

results. They include gray-scale photo colorization [1], [2], [3], colorization of sketch or line art to realistic image [14], [15], and anime illustration [5], [4], [16], [17].

The colors of resulting images may be uncontrollable if we do not give any color guidance to an input gray-scale image, sketch or line art. Guidance-based methods were thus proposed that allow users to input the color guidance optionally as they like. A simple and intuitive way is to put scribble color strokes onto an input image [1], [14], [18]. A reference image is also used as a palette to colorize [4], [5], [16]. Semantic methods [3], [19], [20] were also proposed to control colors through texts.

Style transfer: Style transfer is to render a given image content in a given style. Gatys et al. [6] proposed a convolutional neural network to learn the representations of content and style from images. As follow-up work of [6], several improved methods for style transfer have been proposed [21], [22], [23].

Although visual effect and efficiency have been significantly improved, problems still remain open such as distorted content of stylized images, inconsistent spatial distribution, or insufficient stylization.

Different from the aforementioned work, our task requires solving colorization and style transfer simultaneously while retaining the outline structure of an input character and considering anime specific colorization properties such as lighting or shading of colors. Our proposed model deals with such task.

III. Proposed method

A. Network design

The stylized-colorization problem involves both colorization and style transfer aspects. We may approach this problem with a two-step model in which the image colorization step is followed by the style transfer step. However, this straightforward approach is not good because such a model tends to have a large amount of parameters to learn and thus training the model becomes difficult due to a limited number of available training data. We thus use an end-to-end GAN-based network having a less number of parameters to approach the problem.

The generator adopts the U-Net architecture as in image translation [10]. Since our task has two aspects, we prepare two discriminators for each: one for judging whether a generated image is real or fake in color, and one for judging it in style. In order to reduce the number of parameters in the discriminators, we enforce to share weights at the early layers of the two discriminators. The whole pipeline of our proposed model is illustrated in Fig. 2.

B. Network architecture

The input of our network is a pair of a line art x with the size of $256 \times 256 \times 1$ and a coloring style s in text.

1) Generator: We condition a pair of x and s on generator G in order to generate stylized-colorization image $G(x, s)$ with the size of $256 \times 256 \times 3$. For G, we employ the U-Net architecture [24] as in [10] with our modification so that G receives a pair of x (optionally together with the color guide) and s as its input. Namely, x is fed into the network from the encoder part while s is injected into the network from the decoder part (Fig. 2). The skip-connections between the encoder and the decoder are used to retain structure information of x.

Our generator firstly encodes line art x through the encoder to obtain the line art features with the size of $512 \times 8 \times 8$. At the same time, the model embeds coloring style s to produce a one-hot style vector with the initial size of $n \times 1$ and then replicates to the size of $n \times 8 \times 8$ where n is the number of coloring styles ($n = 5$ in our case). The line art features, the one-hot style vector, and a noise vector z with the size of $64 \times 8 \times 8$ are concatenated and fed into the decoder to produce the generated image $G(x, s)$.

2) Discriminators: We employ two discriminators D_{c} and D_{s} to judge $G(x, s)$ from two aspects: (i) color discriminator D_{c} evaluates whether the color of $G(x, s)$ is real or fake, and (ii) style discriminator D_{s} distinguishes whether the style of $G(x, s)$ is real or fake.

Color discriminator D_{c} is identical to PatchGAN [10]. This is because PatchGAN [10] has the ability of evaluating clearness and sharpness of color in details. D_{c} receives either a ground-truth color image or a generated image as its input, outputting real or fake.

Style discriminator D_{s} is designed following conditional GAN [25] where we condition coloring style s. D_{s} receives either a ground-truth stylized image or a generated image as its input, outputting whether the input agrees with the conditioned coloring style or not.

To stabilize training and to avoid collapsing our model, we replace the default normalization layers in both D_{c} and D_{s} by spectral normalization layers [26]. We also adopt shared-weights at the first three layers between the two discriminators to reduce the size of our model.

C. Loss function

Let y_{c} be a ground-truth color image, y_{s} be a ground-truth stylized image corresponding to a coloring style s. We jointly train our network in an adversarial and end-to-end manner by minimizing the weighted sum of three losses: adversarial loss, per-pixel loss, and style feature loss.

Adversarial loss: The cross-entropy used in [27] may cause a problem in evaluating the similarity of distributions between ground-truth and generated data since it does not take into account the distance between a data sample and decision boundary. We instead employ the hinge loss in our adversarial learning. This is because the hinge loss helps the adversarial learning to be strong and stable [26], [28].

Let $D(\cdot)$ denote the discriminator (D_{c} or D_{s}). At each iteration in training $D(\cdot)$, we randomly select a set $(y_{\text{c}}, y_{\text{s}}, x, s)$ from the training dataset, and feed it to $D(\cdot)$. The loss functions for $D(\cdot)$ are defined as follows:

$$\begin{aligned}\mathcal{L}_{D_{\text{c}}} &= \mathbb{E}_{y_{\text{c}} \sim p_{\text{data}}}\big[\max(0, 1 - D_{\text{c}}(y_{\text{c}}))\big] \\ &\quad + \mathbb{E}_{(x,s) \sim p_{\text{data}}}\big[\max(0, 1 + D_{\text{c}}(G(x, s))\big],\end{aligned} \tag{1}$$

$$\begin{aligned}\mathcal{L}_{D_{\text{s}}} &= \mathbb{E}_{(y_{\text{s}},s) \sim p_{\text{data}}}\big[\max(0, 1 - D_{\text{s}}(y_{\text{s}}, s))\big] \\ &\quad + \mathbb{E}_{(x,s) \sim p_{\text{data}}}\big[\max(0, 1 + D_{\text{s}}(G(x, s), s)\big],\end{aligned} \tag{2}$$

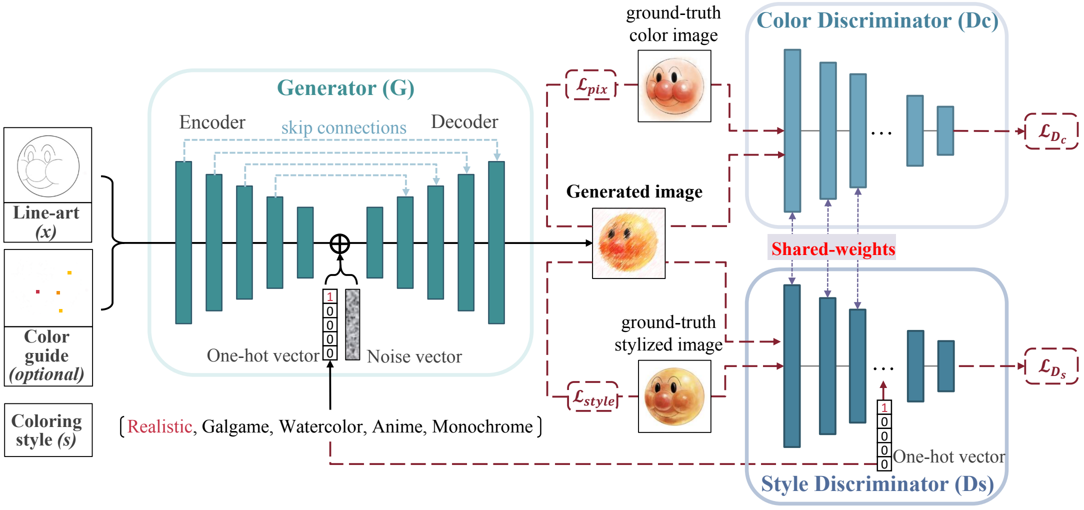

Fig. 2: Overview of our proposed model.

where p_{data} denotes all the training data and $\mathbb{E}_{(\cdot)\sim p_{\text{data}}}$ is the expectation over p_{data}.

Since our network is trained in the adversarial way, we minimize the loss function for generator G:

$$\begin{aligned}\mathcal{L}_G = &-\mathbb{E}_{(x,s)\sim p_{\text{data}}}\left[D_{\text{c}}(G(x,s)\right] \\ &-\mathbb{E}_{(x,s)\sim p_{\text{data}}}\left[D_{\text{s}}(G(x,s),s))\right].\end{aligned} \tag{3}$$

Per-pixel loss: $\mathcal{L}_{\text{pix}} = \mathbb{E}_{(y_{\text{c}},x,s)\sim p_{\text{data}}} \left\| y_{\text{c}} - G(x,s) \right\|_1$ keeps the structure of input line art x and is helpful in learning the color of y_{c}.

Style feature loss: We introduce the style feature loss to learn styles of images. We enforce our style feature loss to have the capability of evaluating how much the outline structure of the character in an input line art is undistorted. We also take into account lightness, shading, and saturation of colors, all of which are specifically important for anime specific colorization. Our style feature loss is thus able to learn what a coloring style is.

We first train style classifier C by employing the center loss [29]. The style classifier C learns center $\boldsymbol{c}_s$ of style features of images with coloring style s through penalizing the distances between the style features and their corresponding center.

We follow the ResNet-18 architecture [30] to design C. For each input image, its style feature $\boldsymbol{f}$ is obtained from the last fully connected layer in C. Like [29], we train C by minimizing:

$$\mathcal{L}_{\text{cls}} = -\log \frac{\exp\left(\boldsymbol{W}_s^\top \boldsymbol{f}_s + \boldsymbol{b}_s\right)}{\sum_{i=1}^{n} \exp\left(\boldsymbol{W}_i^\top \boldsymbol{f}_s + \boldsymbol{b}_i\right)} + \frac{\lambda_{\text{c}}}{2} \left\| \boldsymbol{f}_s - \boldsymbol{c}_s \right\|_2^2, \tag{4}$$

where $\boldsymbol{W}_i$ denotes the i-th column of the weight $\boldsymbol{W}$ in the last fully connected layer; i indicates the coloring style. $\boldsymbol{b}_i$ is the bias term for i. $\boldsymbol{f}_s$ and $\boldsymbol{c}_s$ denote the style feature and the center corresponding to coloring style s, respectively. n is the number of coloring styles ($n = 5$ in our case) and the hyper-parameter λ_{c} is used for balancing the two terms.

TABLE I: The numbers of pairs of line arts and their corresponding colored images for training, and the numbers of line arts for testing collected from each dataset.

Dataset name	Training data			Testing data		
	Line arts	Colored images	**Pairs**	Line arts	Colored images	**Pairs**
Style dataset	1077	5069	5069	85	425	425
Paired dataset	2111	2111	2111	0	0	0
Danbooru2018	0	0	0	498	10426	0
Total	3188	7180	7180	583	10851	425

In each iteration, $\boldsymbol{c}_s$ is updated by averaging all the style features of its corresponding coloring style s over a training mini-batch. After the training, $\boldsymbol{c}_s$ obtained from the trained C becomes the feature representation for coloring style s.

We then employ the pre-train classifier C to define our style feature loss:

$$\mathcal{L}_{\text{style}} = \mathbb{E}_{(x,s)\sim p_{\text{data}}} \left\| \boldsymbol{c}_s - C(G(x,s)) \right\|_2^2, \tag{5}$$

where $C(G(x,s))$ is the style feature of generated image $G(x,s)$ extracted using C.

In summary, our (total) loss function is defined by

$$\mathcal{L} = \mathcal{L}_G + \lambda_{\text{pix}}\mathcal{L}_{\text{pix}} + \lambda_{\text{style}}\mathcal{L}_{\text{style}}, \tag{6}$$

where λ_{pix} and λ_{style} are the hyper-parameters.

IV. Experiments

A. Data setup

To our best knowledge, there is no well-collected open dataset for stylized-colorization. We thus collected our training data and testing data from three different datasets. We used 7180 pair data for training and 583 line arts for testing in total, see Table I. We note that we call a line art and its color image in a style as a *pair*. The details of these datasets are described below.

Style dataset. In order to practice coloring styles, the artists use the same line art template and colorize it in different styles. The typical styles are *realistic*, *galgame*, *watercolor*, *anime*,

TABLE II: Tags we used to collect line arts and illustrations from the Danbooru2018. We also show the total number of images we collected for each coloring style.

Coloring style	**Used tag**	**Collected number**
Line arts	line art, sketch	498
Realistic	realistic	2744
Watercolor	watercolor_(medium)	1811
Anime	anime_coloring	345
Monochrome	monochrome	5526
(Total)		(10426)

and *monochrome*. We found these kinds of practice tables from Pixiv [31] which is an online community aiming to provide a venue for artists to exhibit their illustrations, and collected 5154 pairs of line arts and their corresponding color images in styles. We used 5069 pairs (1077 line arts colorized in five styles (some line arts are colorized in less than five styles)) for training and 425 pairs (85 line arts colorized in five styles) for testing.

Paired dataset. Li [32] released the links between some line arts and their corresponding color images in Pixiv [31]. We thus crawled images using the links and collected 2111 pairs of line arts and their corresponding color images. We used them only for training. Note that we do not have any style information of the color images in this dataset.

Danbooru2018. Danbooru2018 [33] is a large-scale anime image dataset collected from a longstanding image database web-site, Danbooru [34]. Each image in this dataset is annotated with some detailed tags for the character, author or descriptions. Tags for descriptions include eye color, clothing, coloring medium and so on. We collected line arts and (color) illustrations using specific tags. The details are shown in Table II. The 498 collected line arts were used for testing, and the 10426 collected color images (together with 4 coloring styles) were used for evaluation (see Section IV-E).

B. Implementation and training details

We implemented the style classifier C used for the style feature loss and our proposed model in Pytorch. We firstly pre-trained the style classifier C using images from the style dataset. For optimization, we used the SGD optimizer with the batch size of 64 for 200 epochs and the learning rate of 0.01. We set $\lambda_c = 0.5$ in the classifier loss function (Eq. (4)).

Next, we jointly trained our proposed method in the end-to-end manner using images from both the style and paired datasets. Since there is no ground-truth stylized image from the paired dataset, we first randomly chose a coloring style as the target style. We then randomly selected a colored image in the target style from the style dataset as the ground-truth stylized image. To augment input data, we extracted line arts from ground-truth color images using the XDoG [35] filter. When training, our model randomly loaded a real line art or a line art extracted from the ground-truth color images.

Our model is optimized using the Adam optimizer [36] with the batch size of 8 for 200 epochs. The initial learning rate is 0.0002 and it decays linearly to zero after 100 epochs. Experimentally, we set the hyper-parameters $\lambda_{\text{pix}} = 50$, and $\lambda_{\text{style}} = 5$ in our loss function (Eq. (6)).

C. Compared methods

We compared our method with single/multi-domain image translation methods. They are pix2pix [10] (single-domain), StarGAN [12] (multi-domain), and AsymmetricGAN [13] (multi-domain). We also compared our method with style transfer methods. They are AAMS [23] and the method combining online colorization tool Petalica Paint [18] and AAMS [23] (called Petalica+AAMS). Since these compared methods do not address colorization in different coloring styles, we carefully retrained the official implementations (pix2pix[1], StarGAN[2], AsymmetricGAN[3], AAMS[4]) to adapt our problem setting. In particular, for pix2pix [10], we trained one model for each coloring style. For StarGAN [12] and AsymmetricGAN [13], we used the same setting as in our method to train one model for all coloring styles. For AAMS [23], we regarded line arts as content images and color images randomly selected from the style dataset as style images. For the Petalica+AAMS ([18]+[23]), we firstly colorized line arts using Petalica to obtain color images and then used those color images as content images. We randomly selected color images from the style dataset as style images.

D. Qualitative comparison

Fig. 3 shows the visual comparison of stylized-colorization results. From the results by the single/multi-domain image translation methods, we observe that StarGAN [12] and AsymmetricGAN [13] do not achieve pleasant colorization results. We also observe that the quality of the results by pix2pix [10] is unstable. Indeed, pix2pix [10] failed to fill the color well in some cases. From the results by AAMS [23], we observe that directly using a line art as a content image is not appropriate; the results are blurred. Petalica+AAMS solves this problem to some extent, however, some unnatural patterns newly appear. These observations indicate the difficulty of the stylized-colorization task and inability of existing methods for this task. In contrast, we observe that our method successfully colorized line arts in different styles.

E. Quantitative comparison

We adopted PSNR, SSIM [37], and FID [38] metrics for evaluation. PSNR and SSIM are two well-known image quality metrics. A higher PSNR (SSIM as well) value indicates that the image structure is preserved better. Since PSNR and SSIM evaluate only how well the image structure is preserved, we also used FID for evaluating the quality of generated images. FID measures the feature distribution distance between generated images and real samples through a pre-trained Inception-v3 model. Lower FID value indicates that the distribution of generated and real images is more similar.

[1] https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

[2] https://github.com/yunjey/stargan

[3] https://github.com/Ha0Tang/AsymmetricGAN

[4] https://github.com/JianqiangRen/AAMS

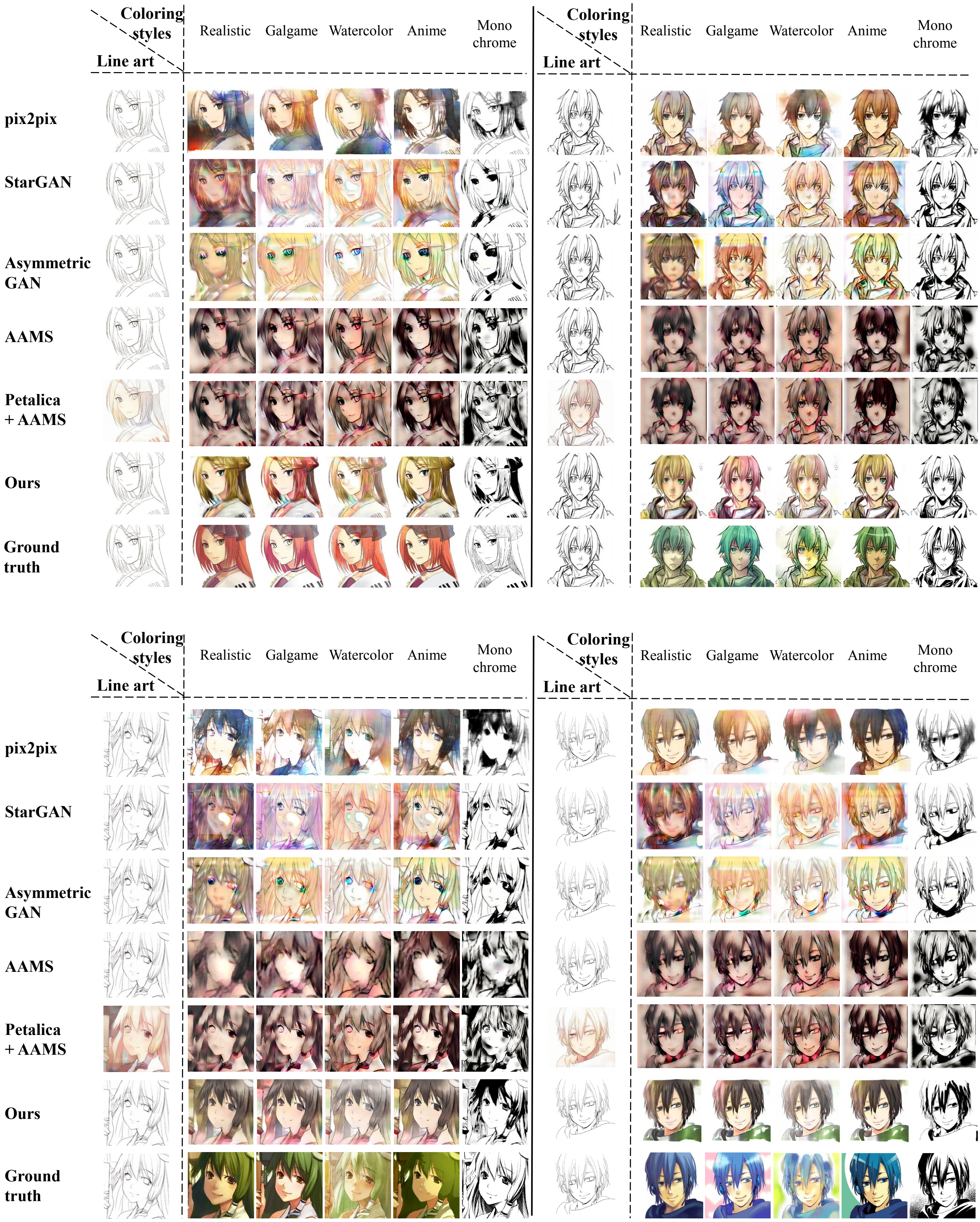

Fig. 3: Visual comparison against pix2pix [10], StarGAN [12], AsymmetricGAN [13], AAMS [23], and Petalica+AAMS. For each example, we show line arts and generated images in different coloring styles. Note that the line arts used in Petalica+AAMS are firstly colorized by Petalica Paint [18].

TABLE III: Comparison of the overall quality using $PSNR$ (larger is better), $SSIM$ (larger is better) and FID (smaller is better). R, G, W, A, and M stand for **R**ealistic, **G**algame, **W**atercolor, **A**nime, and **M**onochrome, respectively.

Metric	PSNR					SSIM					FID			
Coloring style	R	G	W	A	M	R	G	W	A	M	R	W	A	M
pix2pix [10]	8.378	11.028	11.488	10.535	7.357	0.389	0.516	0.501	0.506	0.407	146.62	123.91	129.03	93.87
StarGAN [12]	4.978	5.062	6.288	5.112	3.649	0.145	0.158	0.222	0.184	0.165	165.83	161.03	176.79	106.89
AsymmetricGAN [13]	5.180	5.095	6.047	4.927	3.486	0.165	0.151	0.226	0.201	0.157	173.20	163.75	197.24	114.30
AAMS [23]	8.995	9.379	10.867	8.873	6.866	0.363	0.436	0.414	0.362	0.297	167.27	151.42	187.63	189.27
Petalica+AAMS	9.266	9.504	10.764	9.454	6.688	0.377	0.459	0.442	0.412	0.302	151.86	126.66	160.61	141.69
Ours	**10.753**	**11.511**	**12.957**	**11.165**	**7.799**	**0.455**	**0.573**	**0.563**	**0.548**	**0.445**	**118.88**	**117.38**	**125.49**	**88.34**

Since PSNR and SSIM needs pair data, we used only the style dataset to compute PSNR and SSIM. We firstly generated 425 (85 line arts colorized in five styles) stylized-colorization images. Then, PSNR and SSIM were computed using the stylized-colorization images and ground-truth color images. FID, on the other hand, requires a massive amount of images to compute data distribution. We thus used the style and the Danbooru2018 datasets together to compute FID. More precisely, 2332 generated images (85 + 498 line arts colorized in four styles) and 10766 color images (340 + 10426) were used for computing FID. We remark that we do not compute FID for the "Galgame" style (see Section IV-A).

Table III shows PSNR, SSIM, and FID values illustrating comparison of our method and the other methods. We observe that our method overall outperforms the other methods on all the metrics. We see that StarGAN [12] and AsymmetricGAN [13] are poor on PNSR and SSIM, meaning that they are not able to preserve the image structure well. Pix2pix [31], AAMS [23] and Petalica+AAMS, on the other hand, have similar PSNR (and SSIM) values with each other whereas pix2pix is better than AAMS and Petalica+AAMS on FID. This indicates that pix2pix has better ability of generating images having closer distribution to ground-truth color images than AAMS and Petalica+AAMS. Although the compared methods do not show the advantage on both preserving the image structure and better distribution, our method is not the case. Our method not only achieves better image structure but also learns closer illustration distributions than the others.

F. Ablation study

To evaluate the contribution of each individual loss term, we compared our complete model with models dropping each component (except for the per-pixel loss) in our loss function, and the model without using shared-weights between the two discriminators.

Some visual results are shown in Fig. 4. Obviously, we see that the model w/o shared-weights is not able to generate pleasant color images. This is because as discussed earlier, the size of the model is too large to train from a given limited number of training data. The model w/o $\mathcal{L}_{D_\text{c}}$ brings serious unnatural artifacts in some cases while the model w/o $\mathcal{L}_{D_\text{s}}$ fails to colorize line arts in the monochrome style. We see that the generated results by the model w/o $\mathcal{L}_\text{style}$ become similar with each other even in different coloring styles. These all observations indicate that all the terms of our loss function are necessary and that each term contributes to generate better images. We remark that we do not show the visual results by the model without per-pixel loss. This is because the per-pixel loss value dominates the full loss value and the model was not trained well when we drop the per-pixel loss.

We also evaluated these models on PSNR, SSIM, and FID metrics, see Table IV. Overall, our complete model achieves better scores on these metrics, in particular on SSIM. Although in some cases, the model w/o $\mathcal{L}_{D_\text{s}}$ or the model w/o shared-weights performs better than our complete model on three metrics, the performances of our complete model are comparable with those of the best method even in those cases. This indicates that our complete model has the ability to generate well-balanced results in structure and quality.

To allow users to control the color as they like, we implemented a semi-colorization method based on our model. Following [14], [18], we sampled a random number of color strokes from ground truth illustration at random locations as the color guide when training. Fig. 5 shows an example of our generated results with the color guide. We observe that our model successfully colorizes images according to the given color guide while the style difference is not significant due to the saturation of the color guide. Handling the color guide more flexibly is left for our future work.

G. User study

To further evaluate the effectiveness of our method, we conducted two user studies using 24 subjects. The first user study is for the evaluation with respect to the coloring style while the second one is with respect to the colorization. In these studies, we used 10 line arts to generate 300 ($= 10 \times 5 \times 6$) stylized-colorization images in five coloring styles by six methods including our method, and our five compared methods.

For the first user study, we created 50 sets, each of which consists of a line art, the definition of the five coloring styles, and randomly aligned six generated images from the line art in the same coloring style. The six generated images are obtained by our method and the other five methods. We then displayed for 5 seconds with 2 seconds interval break, each set one by one (50 sets in total) to a subject, and asked the subject to give an absolute rating (from 1 (insufficient) to 5 (excellent); higher is better) with respect to the coloring styles to each of six shown images.

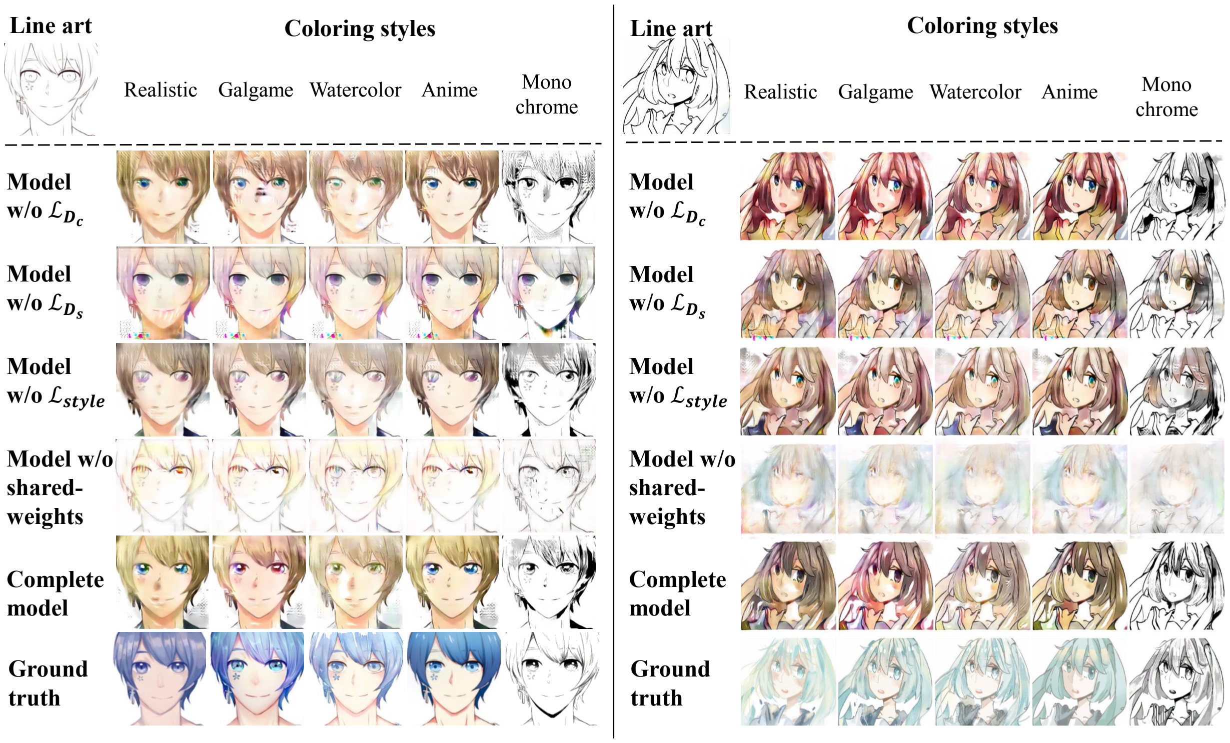

Fig. 4: Visual examples generated images by ablation models.

TABLE IV: Ablation study using $PSNR$ (larger is better), $SSIM$ (larger is better) and FID (smaller is better).

Metric	PSNR					SSIM					FID			
Coloring style	R	G	W	A	M	R	G	W	A	M	R	W	A	M
Model w/o $\mathcal{L}_{\mathrm{D_c}}$	10.646	11.553	12.548	11.020	7.647	0.428	0.549	0.522	0.507	0.423	121.68	117.95	126.95	90.52
Model w/o $\mathcal{L}_{\mathrm{D_s}}$	**10.895**	**11.971**	12.914	**11.371**	**8.465**	0.448	0.571	0.550	0.527	**0.448**	132.36	118.41	**123.18**	114.28
Model w/o $\mathcal{L}_{\mathrm{style}}$	10.504	11.549	12.750	10.982	7.500	0.434	0.561	0.544	0.522	0.435	129.74	121.06	128.76	90.58
Model w/o shared-weights	4.625	4.703	5.634	4.519	4.830	0.193	0.217	0.212	0.188	0.183	171.41	**114.09**	168.31	108.63
Our complete model	10.753	11.511	**12.957**	11.165	7.799	**0.455**	**0.573**	**0.563**	**0.548**	0.445	**118.88**	117.38	125.49	**88.34**

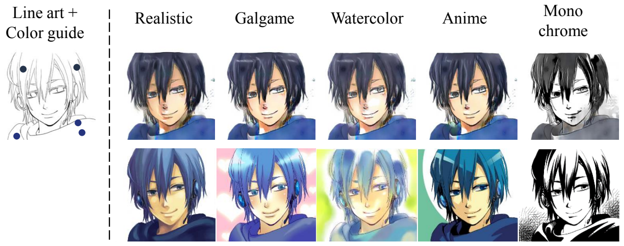

Fig. 5: Visual example generated images by model with color guide. The first row shows our results, the second row shows the ground-truth images.

We measured the Mean Opinion Score (MOS) to evaluate the overall sufficiency in coloring style. MOS is calculated by averaging the scores that each method received from the subjects. A higher MOS indicates better performance in coloring style. The MOS results are shown in Table V. Our method received the best scores among all the methods. We see that our method received significantly higher scores than the other methods in all the five styles, showing that our method performs better visual results on the coloring style characteristics than the other methods. We also see that the image translation methods (pix2pix, StarGAN, Asymmetric-GAN) outperform the style transfer methods (AAMS and Petalica+AAMS). This means that the existing style transfer methods are limited to handle transferring the anime specific colorization properties.

For the second user study, we employed Petalica Paint [18] (colorization method), pix2pix [10], and Petalica+AAMS for comparison. We created 50 sets, each of which consists of a line art, and randomly aligned four generated images from the line art in the same coloring style. We then displayed for 5 seconds with 2 seconds interval break, each set one by one (50 sets in total) to a subject, and asked the subject to vote for the best result in colorization among four shown images. We collected 1200 votes in total from the 24 subjects.

TABLE V: Mean opinion score (MOS) on sufficiency in coloring styles on line arts.

Method	Coloring style					Total
	R	G	C	A	M	
pix2pix [10]	3.27	3.31	3.98	3.65	3.11	3.46
StarGAN [12]	2.46	2.30	2.80	2.35	3.78	2.74
AsymmetricGAN [13]	2.25	2.33	2.73	2.35	3.16	2.56
AAMS [23]	1.53	1.55	1.76	1.56	1.59	1.60
Petalica+AAMS	2.01	1.65	1.95	2.03	2.24	1.98
Ours	**4.35**	**4.59**	**4.15**	**4.58**	**4.55**	**4.44**

TABLE VI: Votes for choosing best colorization on line arts.

Method	Votes (percentage(%))
pix2pix [10]	140 (11.67)
Petalica Paint [18]	255 (21.25)
Petalica+AAMS	20 (1.67)
Ours	**785 (65.41)**

We computed the percentage of the votes that each method received, showing in Table VI. We see that our method received over 65% supports and significantly outperforms the other methods. This means that our method is recognized as most excellent in colorization.

These user studies enable us to conclude that our method outperforms the others in both stylization and colorization.

V. Conclusion

We addressed a novel problem of stylized-colorization for line arts, and presented a GAN-based end-to-end model having two discriminators for this problem. The stylized-colorization problem requires to handle colorization and style transfer tasks simultaneously. Our two discriminators work for judging in each task, stylized-colorization images produced by the generator. Our experiments demonstrated that our model successfully retains the outline structure of a character in an input line art with considering anime specific colorization properties, which is difficult to achieve with just combining existing colorization and style transfer methods.

Although we demonstrated stylized-colorization of face line arts in our experiments, there is no component in our model that is specific to the face. Our model is able to handle more complex line arts such as a complete character with background if training data are available.

References

[1] R. Zhang, J.-Y. Zhu, P. Isola, X. Geng, A. S. Lin, T. Yu, and A. A. Efros, "Real-time user-guided image colorization with learned deep priors," *ACM Transactions on Graphics*, 2017. 1, 2

[2] Y. Xiao, P. Zhou, and Y. Zheng, "Interactive deep colorization with simultaneous global and local inputs," in *ICASSP*, 2019. 1, 2

[3] V. Manjunatha, M. Iyyer, J. Boyd-Graber, and L. Davis, "Learning to color from language," in *NAACL*, 2018. 1, 2

[4] C. Furusawa, K. Hiroshiba, K. Ogaki, and Y. Odagiri, "Comicolorization: semi-automatic manga colorization," in *SIGGRAPH Asia 2017 Technical Briefs*, 2017. 1, 2

[5] P. Hensman and K. Aizawa, "cgan-based manga colorization using a single training image," in *ICDAR*, 2017. 1, 2

[6] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *CVPR*, 2016. 1, 2

[7] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *ECCV*. Springer, 2016. 1

[8] X. Wang, G. Oxholm, D. Zhang, and Y.-F. Wang, "Multimodal transfer: A hierarchical deep convolutional neural network for fast artistic style transfer," in *CVPR*, 2017. 1

[9] A. Sanakoyeu, D. Kotovenko, S. Lang, and B. Ommer, "A style-aware content loss for real-time hd style transfer," in *ECCV*, 2018. 1

[10] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *CVPR*, 2017. 1, 2, 4, 5, 6, 7, 8

[11] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *ICCV*, 2017. 1

[12] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *CVPR*, 2018. 1, 4, 5, 6, 8

[13] H. Tang, D. Xu, W. Wang, Y. Yan, and N. Sebe, "Dual generator generative adversarial networks for multi-domain image-to-image translation," in *ACCV*, 2018. 1, 4, 5, 6, 8

[14] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays, "Scribbler: Controlling deep image synthesis with sketch and color," in *CVPR*, 2017. 2, 6

[15] W. Chen and J. Hays, "Sketchygan: Towards diverse and realistic sketch to image synthesis," in *CVPR*, 2018. 2

[16] L. Zhang, Y. Ji, X. Lin, and C. Liu, "Style transfer for anime sketches with enhanced residual u-net and auxiliary classifier gan," in *ACPR*, 2017. 2

[17] Y. Ci, X. Ma, Z. Wang, H. Li, and Z. Luo, "User-guided deep anime line art colorization with conditional adversarial networks," in *ACMMM*, 2018. 2

[18] P. Networks, "Petalica paint," https://petalica-paint.pixiv.dev/, 2017. 2, 4, 5, 6, 7, 8

[19] C. Zou, H. Mo, R. Du, X. Wu, C. Gao, and H. Fu, "Lucss: Language-based user-customized colourization of scene sketches," *arXiv:1808.10544*, 2018. 2

[20] H. Kim, H. Y. Jhoo, E. Park, and S. Yoo, "Tag2pix: Line art colorization using text tag with secat and changing loss," in *ICCV*, 2019. 2

[21] F. Luan, S. Paris, E. Shechtman, and K. Bala, "Deep photo style transfer," in *CVPR*, 2017. 2

[22] R. Mechrez, E. Shechtman, and L. Zelnik-Manor, "Photorealistic style transfer with screened poisson equation," in *BMVC*, 2017. 2

[23] Y. Yao, J. Ren, X. Xie, W. Liu, Y.-J. Liu, and J. Wang, "Attention-aware multi-stroke style transfer," in *CVPR*, 2019. 2, 4, 5, 6, 8

[24] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015. 2

[25] M. Mirza and S. Osindero, "Conditional generative adversarial nets," in *NIPS*, 2014. 2

[26] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *ICLR*, 2018. 2

[27] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, 2014. 2

[28] J. H. Lim and J. C. Ye, "Geometric gan," *arXiv:1705.02894*, 2017. 2

[29] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *ECCV*, 2016. 3

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016. 3

[31] "Pixiv," https://www.pixiv.net/. 4, 6

[32] J. Li, "Pixiv dataset," https://github.com/jerryli27/pixiv_dataset, 2017. 4

[33] Anonymous, D. community, G. Branwen, and A. Gokaslan, "Danbooru2018: A large-scale crowdsourced and tagged anime illustration dataset," https://www.gwern.net/Danbooru2018, January 2019. 4

[34] "Danbooru," https://danbooru.donmai.us/. 4

[35] H. WinnemöLler, J. E. Kyprianidis, and S. C. Olsen, "Xdog: an extended difference-of-gaussians compendium including advanced image stylization," *Computers & Graphics*, 2012. 4

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015. 4

[37] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE TIP*, 2004. 4

[38] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *NIPS*, 2017. 4