applied sciences

MDPI

Article

Seg2pix: Few Shot Training Line Art Colorization with Segmented Image Data

Chang Wook Seo and Yongduek Seo *

Department of Art and technology, Sogang University, Seoul 04107, Korea; mightylg9094@gmail.com
* Correspondence: yndk@sogang.ac.kr

Abstract: There are various challenging issues in automating line art colorization. In this paper, we propose a GAN approach incorporating semantic segmentation image data. Our GAN-based method, named Seg2pix, can automatically generate high quality colorized images, aiming at computerizing one of the most tedious and repetitive jobs performed by coloring workers in the webtoon industry. The network structure of Seg2pix is mostly a modification of the architecture of Pix2pix, which is a convolution-based generative adversarial network for image-to-image translation. Through this method, we can generate high quality colorized images of a particular character with only a few training data. Seg2pix is designed to reproduce a segmented image, which becomes the suggestion data for line art colorization. The segmented image is automatically generated through a generative network with a line art image and a segmentation ground truth. In the next step, this generative network creates a colorized image from the line art and segmented image, which is generated from the former step of the generative network. To summarize, only one line art image is required for testing the generative model, and an original colorized image and segmented image are additionally required as the ground truth for training the model. These generations of the segmented image and colorized image proceed by an end-to-end method sharing the same loss functions. By using this method, we produce better qualitative results for automatic colorization of a particular character's line art. This improvement can also be measured by quantitative results with Learned Perceptual Image Patch Similarity (LPIPS) comparison. We believe this may help artists exercise their creative expertise mainly in the area where computerization is not yet capable.

Keywords: colorization; generative adversarial network; line art

Citation: Seo, C.W.; Seo, Y. Seg2pix: Few Shot Training Line Art Colorization with Segmented Image Data. *Appl. Sci.* **2021**, *11*, 1464. https://doi.org/10.3390/app11041464

Academic Editor: Yongduek Seo
Received: 30 November 2020
Accepted: 1 February 2021
Published: 5 February 2021

1. Introduction

Line art colorization is an expensive and labor-intensive process especially in the animation and comics industry due to the repetitive tasks of the job. In the webtoon (web-publishing comics) industry, such an iterative work of colorizing many line art drawings makes it prone to coloring mistakes, as well as exhausting much cost and time. To mitigate this problem, automating the procedure of coloring should be considered to reduce the amount of actual coloring labor. Recently, many challenges and improvements for automating line art colorization have appeared, and most of the studies have been based on Generative Adversarial Networks (GANs) [1]. The GAN in the past only generated random images based on the training dataset, which cannot be processed based on the user's desired sketch input. However, Pix2pix [2], a modified method of GANs, which can generate images based on an input vector, has motivated research into automatic line art colorization. For example, the methods of [3–5] give user-guided suggestions about the desired data for colors and locations.

The two methods of [6,7] provided the desired style for colorization using the data of reference images. Even though these methods generate a fully colorized image for the input line art image, their quality and details are not sufficient for application in the webtoon industry. For example, as illustrated in Figure 1, a currently available technique,

here Style2paints [7], produces unmatching color tones and blurring in the generated images compared to the original designs. Of course, we could have obtained a better colorization, but hyper-parameter tuning and training of the model were barely possible in a reasonable testing time.

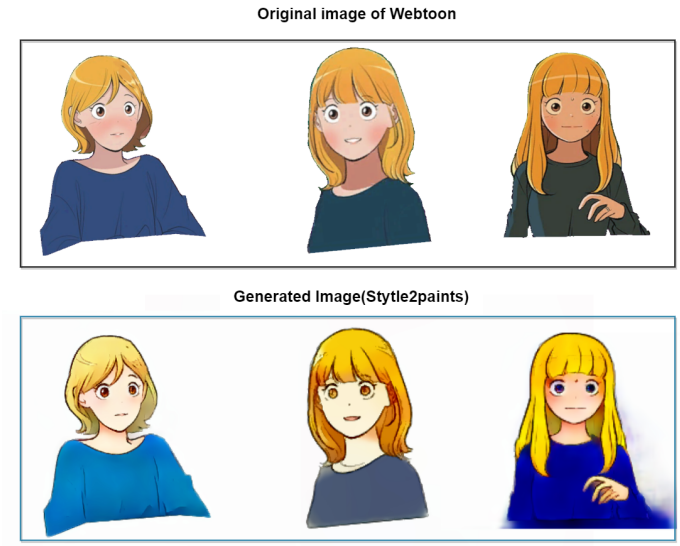

Figure 1. Original images used in a webtoon and a set of generated images by Style2paints, a line art colorization network [7].

Practically, it is a minimum requirement that the generated figures have almost an exact color match to their original character designs, if they are to be used in commercial illustration. This is not limited to the webtoon industry. Figure 2 shows examples of coloring mistakes in commercial animation; notice that the dress color changes instantly in the animation, which should not have occurred. Our goal in this paper is to remove such colorization mistakes in the mass production of webtoons or animations and to reduce the amount of time spent on the whole process of line art coloring. To accomplish this goal, we had to find a method to colorize a particular character with high-quality color matches, rather than colorizing random characters with hardly matching colors.

Figure 2. The same character in a sequence is mis-colored from bluish clothes to yellow clothes. This kind of mistake can be easily found in TV series animations due to the lack of time and labor [8].

Observing the colorization results from previous methods, we see mainly two features to be upgraded for the colorization used in the webtoon production pipeline. First of all, the coloring should be confined accurately by the boundaries given from the line art input. This will allow the artists to seamlessly do the job of adding highlights and details to the output when the output of automatic colorization needs a few more details. Second, the generative colorization model should be able to learn the particular character's pattern from only a small amount of input data. The number of original images created by the

chief artist is not many, usually about 10 or less than that. Furthermore, using only a few data can make it profitable for making multiple colorization models for each character.

To achieve our goal, we propose a new GAN-based neural network model, Seg2pix The model generates a segmented image to be used in generating a high-quality colorized image. The segmented image provides data that clearly distinguish the parts of the character according to borderlines of the line art. As shown in Figure 3 the segmented image shows an example of the first generated result of our Seg2pix model. The parts of the character are represented by labels, but illustrated using pseudo-color: the hair segment is in green, and the two eye segments are labeled in red. For Seg2pix to learn this functionality, we prepare segmentation-labeled image data for each of the original character creations to be used for training the targets. The labeling was done through simple colorization of the originals. This job is easy for anyone, even those who are not trained in illustration.

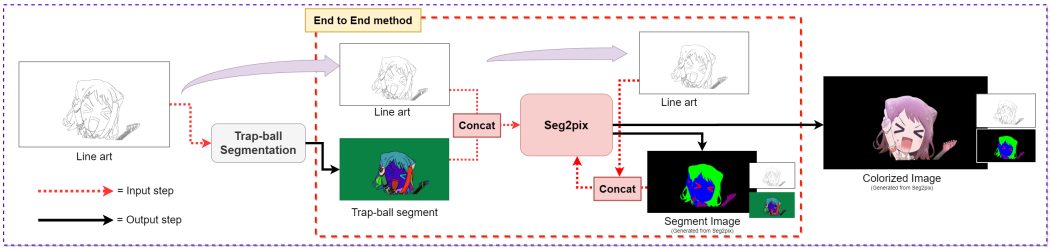

Figure 3. Seg2pixgenerates the output by two concatenated image data. The first step generates the "segmented image" by the line art and trap ball segment input; the second step generates the "colorized image" by the line art and segmented image. This generation works by an end-to-end method updating the same loss function data (sample image from ©Bushiroad Co., Ltd.) [9].

Then, Seg2pix receives two kinds of ground truth images for automatic colorization training: one is the input line art drawing, and the other is the trap ball segmented image.

First, the model generates fully segmented image data by concatenating the line art input and trap ball segmented image. Second, Seg2pix generates the final colorized image by concatenating the line art input and fully segmented image, which was achieved by the first generation.

Our two-step GAN model is found to be effective at minimizing the blurring near the complex borderlines of the character drawing in the line art image and avoiding the mis-coloring in the labeled parts such as blobs along the borderlines of the skin and hair, which will be later shown with the results of the experiments. We overcame the problem of having a small amount of data through a data augmentation specialized to our environment. In the data preparation for neural network learning, all the images underwent rotation and translation transformations. To produce as many line art drawings as possible, we applied basic edge filters like Sobel [10] and Canny [11], performed by morphological operations and the open-source neural network model called SketchKeras [12].

2. Related Works

2.1. GAN for Colorization

The GAN approach offers very efficient performance in image generation tasks compared to supervised methods of image generation. Early supervised methods for coloring [13,14] propagate stroke colors with low-level similarity metrics. The GAN research related to line art colorization such as [3,6] utilized user-guided color stroke maps. The colorization methods of [7] adopt StyleGAN [15] and extend the refinement stages to improve the quality of the output. While previous methods used color suggestion map data, Tag2pix [16] takes tag data written as text such as "blue shirt" or "blonde hair" for coloring line art drawings. Colorization methods like [17] propose two-stage methods with pixel parsing to overcome the lack of pixel information problem for line art drawings, which is a very similar approach to our Seg2pix. However, we observe that these methods scarcely obtain the same color matches as the original image, while we want to achieve

standardized patterns of the original image's exact color match. Since the last methods aim their colorization at random line art characters with undetermined quality, we tried to focus only on a particular line art character for the industrial application of webtoon series and animation.

2.2. Pix2pix

Pix2pix [2] is a convolution-based generative adversarial network for image-to-image translation. Through this method's architecture, we designed the neural network model for high-quality colorization. Unlike other previous GANs, Pix2pix generates a designated style of image data related to the input image vector. This means that the network model can generate a colorized character image with the same pose as the input sketch image, while previous GANs could only generate a colorized random character image with no input vector.

2.3. Sketch Parsing

To generate an accurate segmented image for our Seg2pix, we searched for methods of sketch parsing. Since sketch parsing analysis is one of the important topics in the multimedia community, several studies have contributed semantic parsing of freehand sketches [18,19]. Various methods of sketch parsing have been reported such as those focusing on sketch images [20–22] and 3D retrieval [23–25]. In this paper, we perform sketch parsing using a combination of algorithms: trap ball segmentation and Seg2pix network. With the brief mention of neural network semantic segmentation methods, including FCN [26], intensive study has been done in the computer vision society and consistently shown improved results [27–29]. Furthermore, with particular objectives such as semantic lesion segmentation for medical use, References [30,31] showed improved methods with highly GAN-based knowledge adaptation. However, we adopt trap ball segmentation, which is based on a supervised contour detecting line filler algorithm and the Seg2pix network, which we also use for colorization.

3. Webtoon Dataset for Seg2pix

Since our main goal is to develop an automatic colorization method for the webtoon industry, we made a dataset from two different webtoon series: "Yumi's cell" [32] and "Fantasy sister" [33] shown on Naver Webtoon. We chose four different characters from the webtoons to test the proper performance of Seg2pix.

Capturing the image from various episodes of the comics, we chose the target character for each webtoon title. The captured images were cropped to the character's shape with a whitened background. As shown in Figure 4, the cropped images were labeled as mask images into four classes: hair, eyes, skin, and clothes. Character A from "Yumi's cell" has been published over three years, and there have been changes in the character's hairstyle, as shown in Figure 4. Nonetheless, our method is found to admit a few minor changes to the character shapes such as hairstyle and emotions. Test data were obtained from different episodes against the training data, which were for the same character.

Line art extraction: To make the line art from a colored image for supervised learning, we used two methods to vary the style of line art. First, we used the Canny edge detector [11], which is the most traditional and well-known method to extract edge pixels, and also, we thickened the lines and edges from the Canny edge detector. As the main sketch data, we used the SketchKeras [12] network, which is specializes in line art extraction in the style of hand drawing.

Data augmentation: Assuming the actual use of our method in comics and animation works, we used only 10 images for one character's colorization for training. Then, the 10 image dataset was multiplied to 120 images by variation of the line art style and rotating random features (the rotating parameter was between −30 to +30 degrees). Therefore, we prepared 120 images for each character to train the model. The number of character datasets was four, as shown in Figure 4.

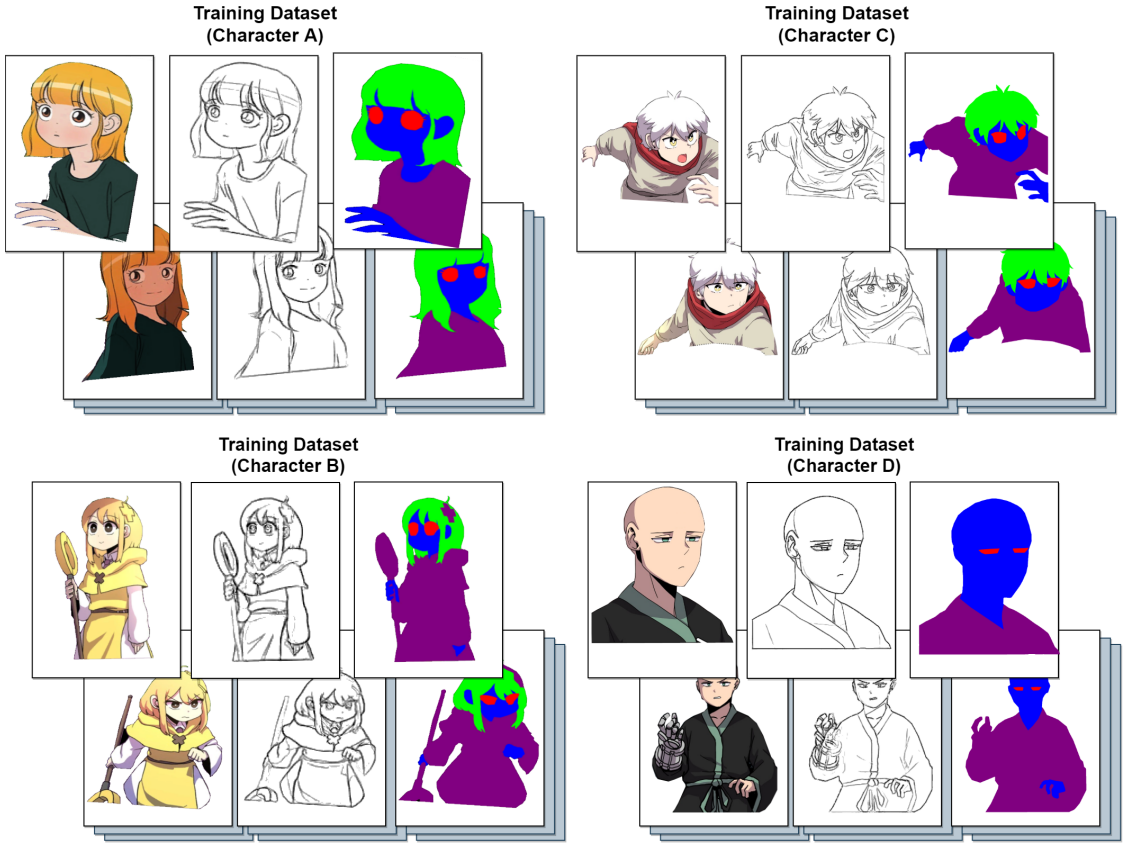

Figure 4. From left to right, (1) original image, (2) extracted line art be SketchKeras, (3) masked image for eyes, hair, skin, and clothes.

4. Workflow Overview and Details

As shown in Figure 5, training proceeded twice, generating the segmented image and the colorized image. The generated segmented image became the input to the Seg2pix network with line art, and then, it became the final colorized image.

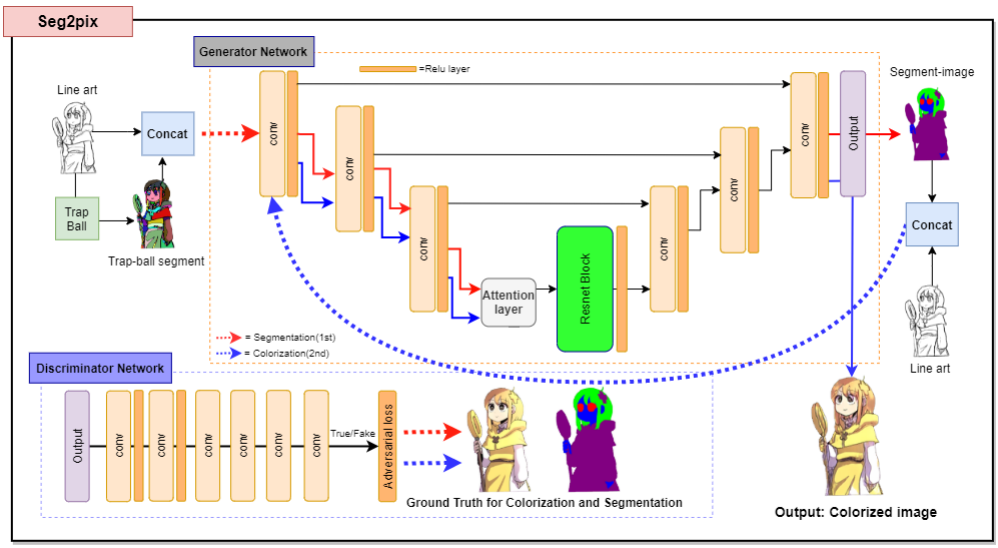

Figure 5. The architecture of the generator and discriminator network. Two types of ground truth are required: one is the hand-labeled segmented image, and the other is the colorized original image. Seg2pix generates both desired outputs by a two-step training using the same architecture of the network, and this training is accomplished by end-to-end methods.

4.1. Trap Ball Segmentation

To obtain high-quality segmented image data by Seg2pix, we adopted a trap ball segmentation [34] for data preprocessing. This is a traditional style image analysis method for segmenting a line art image based on contours by computing several small connected structures. Flood filling, which becomes the basic method of making a trap ball segmented image, is described in Algorithm 1. More detailed code with the full iteration and optimization can be seen in [35]. This pseudo-segmentation is introduced to help Seg2pix produce a better segmented mask.

Translating raw line art images to trap ball segmented data prevents Seg2pix from making mislabeled pixels on white or widely empty areas such as the forehead of a line art character image. Furthermore, this step allows the network to recognize detailed areas with better quality, such as the eyes and hair of the character, as shown in Figure 6.

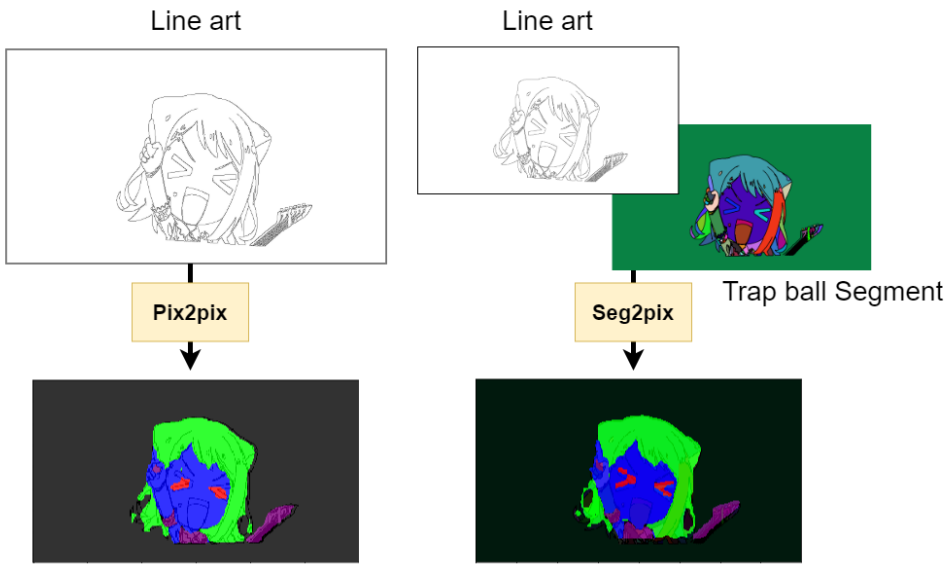

Figure 6. Image-to-image translation GAN output from only line art input (via pix2pix) and two inputs with the trap ball segmented image (via Seg2pix) showing different results in the detailed area, such as eyes and hair.

For generating the segmented image that is used for colorizing the line art, we trained the Seg2pix model with the hand-labeled segmented image as the ground truth. The input data for this training session were the concatenated data of the line art and trap ball segmented image, and the training pipeline is shown in Figure 3.

4.2. Segmentation

The network to make the segmented image was the same as the colorizing architecture, and it was trained using the hand-labeled segmented image as the ground truth, as shown inn Figure 4. The input image for this network was the concatenated image data of the line art image and the trap ball segmented image. While training, image transformations such as random rotation and random resizing were applied as the data augmentation procedure.

Algorithm 1: Flood filling algorithm for trap ball segmentation.

Result: Flood fill the single area of the input image

image = input sketch image (w*h*c) with the binarization preprocess (0,1)

ball = ellipse-shaped matrix by setting the size of the radius (=5 as an example)

pass1 = matrix size of the input **image** shape with a value of 255

inv = bitwise calculation from the input **image**(reversing the value of 0 and 1)

mask1 = copying from **inv** with 1 pixel of the border

pass1 = flood filling (find the empty pixel index and fill the color pixel value by comparing **mask1** and **pass1** by iterating the kernel size of **ball**. This ends when the iteration index finds the value of 0 from the matrix)

Return pass1 (this becomes the data of a single segmented area)

4.3. Colorize

After the segmented image has been achieved by the previous Section 4.1, Segmentation, we use it as the concatenating data with the line art image for the next training step. As shown in Figure 5, the Seg2pix network has an encoder-decoder network [36]. The rectangles in the figure imply convolution blocks and active layers that transform concatenated input data to feature maps. The convolution parameter grows by 64 per layer, and this encoder step progresses until the third layer of the encoder. Then, the features are progressively scaled up in dimension, and U-Net [29] follows the sub-networks to concatenate to the decoder part of the network. Here, the skip-connection is necessary because transporting information across the network to the generator to circumvent the bottleneck solves the problems of the encoder-decoder network's downsampling of the layers.

For the discriminator network, we used the PatchGAN style based on the Markovian discriminator [37]. This discriminator is structured by the convolution layer, batch normalization, and leaky ReLU, similar to the encoder-decoder network. While past discriminators for GAN tried to classify fake or real by recognizing all the image data, the Markovian discriminator recognizes the particular size of the patch data pixels and the correlation factors of the neighboring patch. This change decreases the number of parameters for the discriminator network, which results in faster computing times. Furthermore, the Markovian discriminator retains the lightweight complexity of its network.

4.4. Attention Layer

Unlike Pix2pix [2], we implemented the attention layer before the ResNet block of the generator. The design of the attention layer was based on the attention module of SAGAN [38], as shown in Figures 5 and 7. The attention layer of the generative model allows attention-driven, long-range dependency modeling for image generation tasks. The input image feature changes to a 1×1 convolution and produces three attention matrices query, key and value. The query and key value produce the attention map by the dot product of the two matrices. This attention map becomes the self-attention feature map by the dot product with the value matrix. As shown in Figure 8, using this attention feature map for the network produces modified details from all small feature locations. Due to the network's complexity problem, the attention layer is implemented only in the generator network of Seg2pix.

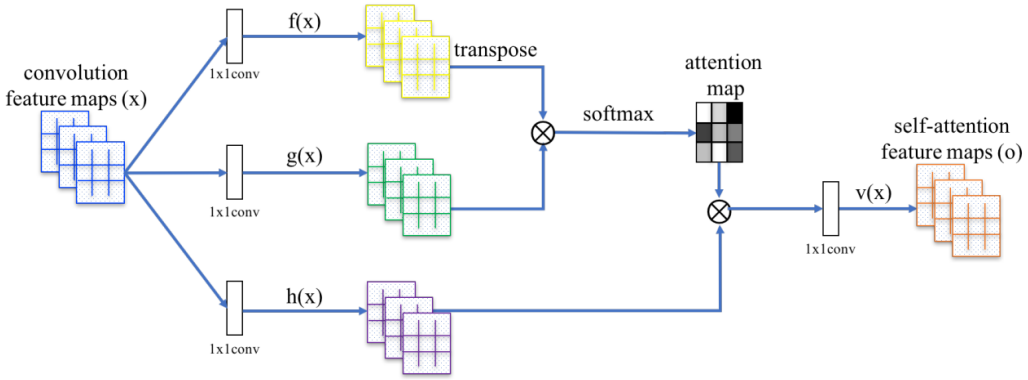

Figure 7. The block diagram of the self-attention layer [38].

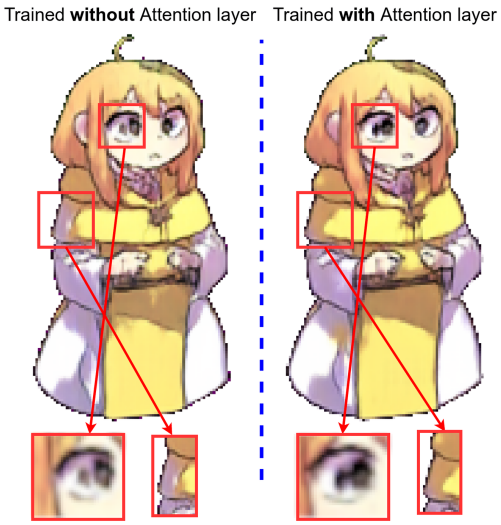

Figure 8. The image on left shows the generated image without the attention layer, and the right shows the generated image with the attention layer. The two images show different qualities of the color distribution on the small details.

4.5. Loss

The loss of Seg2pix is briefly given in Equation (3), which is the same as that of Pix2pix [2], mostly based on the shape of the conditional GAN. The loss of the conditional GAN is written as Equation (1):

$$\begin{aligned}\mathcal{L}_{cGAN}(G,D) = \mathbb{E}_{x,y}[logD(x,y)]+ \\ \mathbb{E}_{x,y}[logD(1-D(x,G(x,z))]\end{aligned} \tag{1}$$

To encourage less blurring, we use the L_1 distance:

$$\mathcal{L}_{L_1}(G) = \mathbb{E}_{x,y,z}[||y - G(x,z)||_1] \tag{2}$$

The final objective loss becomes:

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G,D) + \lambda\mathcal{L}_{L_1}(G) \tag{3}$$

x represents the input data, which are the concatenated data of the line art and trap ball segmentation data, and y represents the real colorized data of the ground truth. z is a random noise vector that prevents the generator from producing deterministic outputs. G and D define the generator and the discriminator

The generator tries to minimize this objective against an adversarial discriminator, which tries to maximize it. As a final objective, we sum with the L_1 distance loss to train the image's low-frequency content while the adversarial loss trains the high-frequency content. This makes the image sharper to avoid blurry features in the image.

5. Experiments

5.1. Experimental Settings (Parameters)

To implement this model, the Pytorch framework [39] was used. The training was performed with an NVIDIA GTX 2080ti GPU.

The networks were trained with the collected dataset. While training images with the dataset, we separated the training data by character as Datasets A, B, C, and D, so we could make the individual network for a particular character. We randomly flipped the images horizontally for training augmentation, as described on Section 3. Specific values of the parameter for model is shown on Table 1.

For optimization, the ADAM [40] solver was implemented using the minibatch function. The initial learning rate was 0.0002 for both the generator and the discriminator. The total epoch time was 200, and the learning rate decayed slightly after 100 times, as shown in Algorithm 2. Approximately 20 min were required for training each particular dataset model. The shortest time was 19 min 41 s and the longest time was 20 min 22 s with the same amount of image data.

Table 1. Values of the parameters.

Types of Parameter	Value
Learning rate discriminator	0.0002
Learning rate generator	0.0002
Learning rate policy	linear
Optimizer	Adam
Loaded image size	512 × 288
Epochs	200
Epochs' decay	100
Generator encoder layer conv	64-128-256-256 (attention)-512-512-512-512-512
Generator decoder layer conv	512-512-512-512-256-128-64
Discriminator layer conv	64-128-256-512-512-512

Algorithm 2: Optimizer learning rate algorithm.

Result: learning rate decays over 100 epoch counts with a total of 200 epochs
epoch count = 0

niter decay = 100 ← setting the epoch number when decaying starts

niter = 100 ← times of epochs for decaying duration

learning rate = 0.0002 ← initial learning rate

while *epoch count != 200* **do**

epoch count + = 1

learning rate linear = 1.0 − max(0, epoch count + 1 − niter)/float(niter decay + 1)

learning rate = learning rate * learning rate linear

end

5.2. Comparisons

Since the goal of our network is to colorize a particular character with few training data (10 images), there is no exact way to compare the result with previous colorization networks, which were trained with over 10,000 images with the desire to colorize random characters. Just for reference, we adopted Learned Perceptual Image Patch Similarity (LPIPS) [41] to make quantitative comparison results. LPIPS compares the generated image by the convolution feature extracting method rather than the color distribution matching method. This method can detect intra-class mode dropping and can measure diversity in the quality of the generated samples, which is a well-known proper comparison for GAN generated images. Even though the Frechet Inception Distance (FID) [42] is the best known method for generated image comparison, it requires at more than 5000 test images to have confident results, so it is not appropriate for our few shot line art image colorization experiment. In this experiment, we only cropped the image of the character except the background to make a fair comparison of the performance.

As given in Table 2, the comparison of the LPIPS score shows a huge gap between Seg2pix and the previous networks except for Pix2pix (trained with the same dataset as Seg2pix). Furthermore, we performed an ablation study by removing the attention layer and trap ball segment method from Seg2pix (the same two sketch images were used as the input while removing the trap ball method). Since our Seg2pix model was trained on a particular character to automate colorization, it obviously showed a good score compared to the previous colorization methods except Pix2pix. However, Seg2pix sill had a better score than Pix2pix. We trained the Pix2pix network with the same ground truth images used in our Seg2pix training, and the results are visualized in Figure 9. Compared with Pix2pix, we may also observe obvious differences in the color distribution and borderlines with less color bleeding in the qualitative comparison.

Table 2. Results of the Learned Perceptual Image Patch Similarity (LPIPS) in our experiments showing a better score compared to the other colorization models, even against Pix2pix, which was trained on the same dataset as Seg2pix.

Training Configuration	LPIPS (Average)
Style2paints	0.3372
PaintsChainer	0.1772
Tag2pix	0.0723
Seg2pix (without trap ball and attention)	0.0488
Seg2pix (without trap ball)	0.0478
pix2pix	0.0477
Seg2pix (without attention layer)	0.0411
Seg2pix (ours)	**0.0375**

5.3. Analysis of the Results

As shown in Figures 9–11, the raw generated image from the original image shows the correct color distribution and shading, as we expected to obtain with only a few training data. All these outputs were accomplished with one Seg2pix network. Generating segmented image data and colorized output was performed gradually by the two-step training.

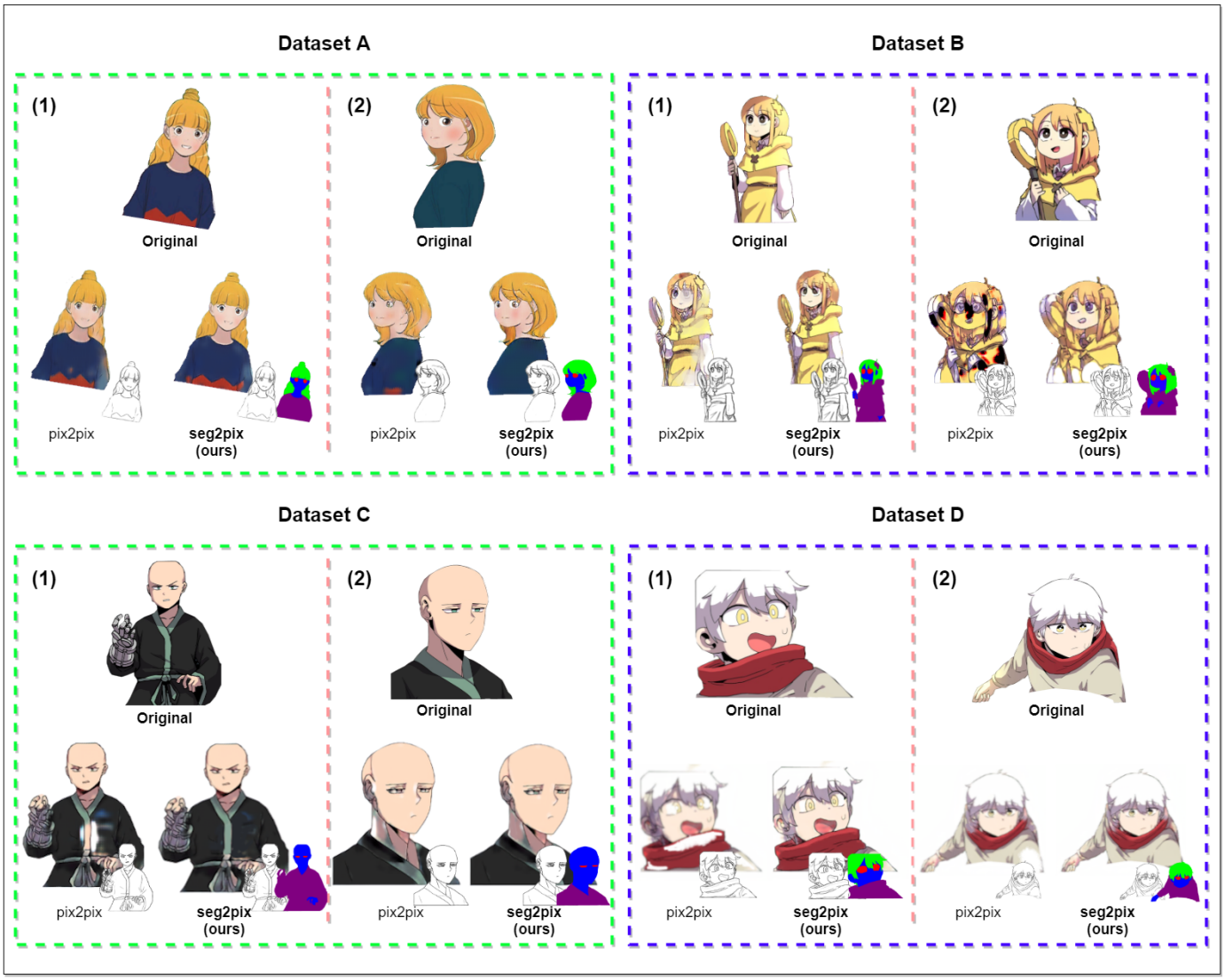

Figure 9. The Pix2pix generated images (1) from Dataset B and (2) from Dataset D show only slight differences in comparison to the Seg2pix output. However, the other Seg2pix generated images show very clear differences in the color bleeding and quality compared to the Pix2pix generated image trained on the same ground truth. Specifically, for the Pix2pix result (2) from Dataset B and (1) from Dataset C, these show a very low-quality color distribution between the skin and clothes.

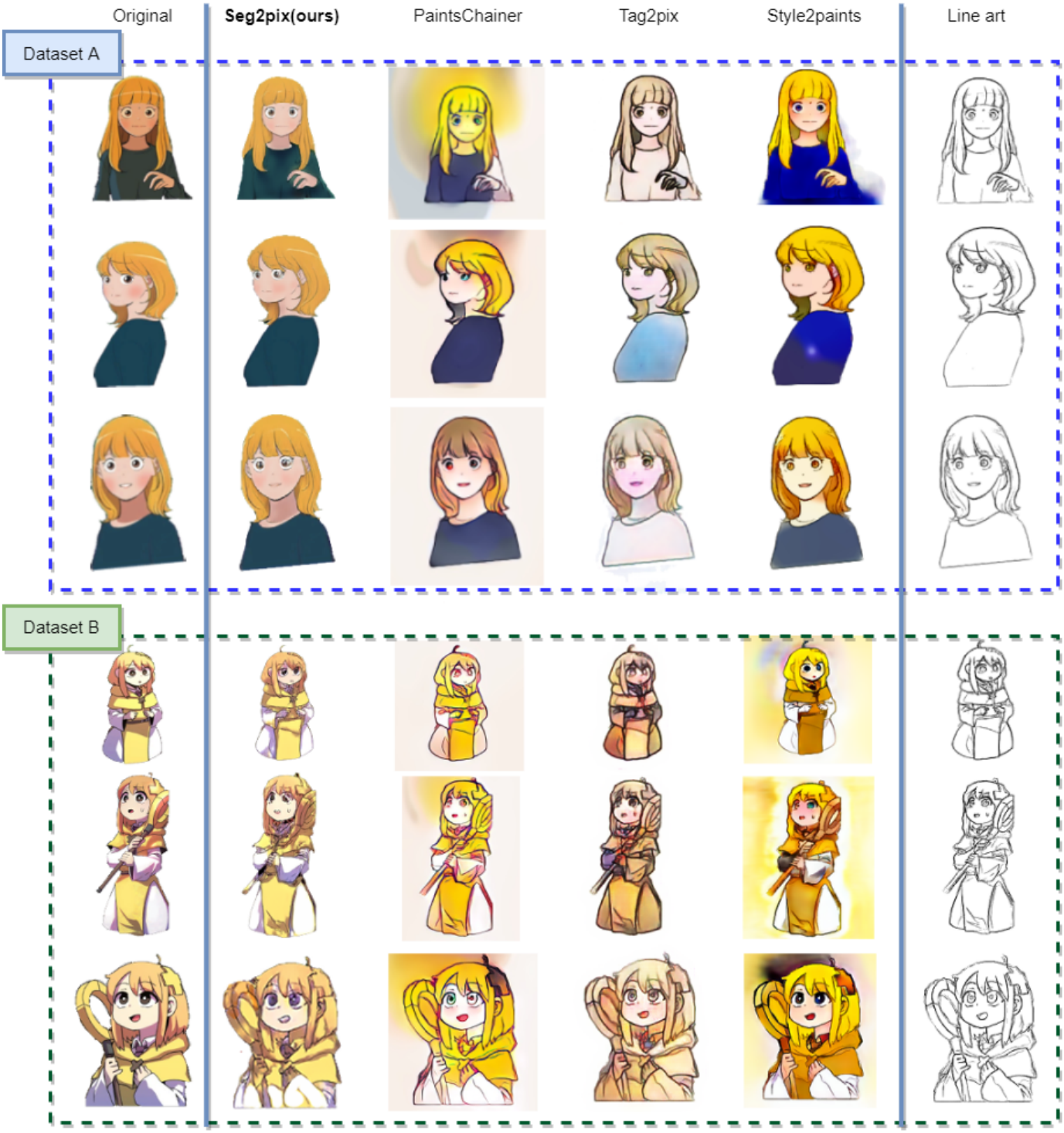

Figure 10. Comparison of the line art colorization results. The first column shows the original ground truth image. Columns 2–5 show the automatic colorization results; from left to right: Seg2pix, PaintsChainer (Canna) [6], Tag2pix [16], and Style2paints [7]. Rows 1–3 of the Seg2pix column are the results from the model trained on Dataset A (Yumi's cell series), and Rows 4–6 are the results from the model trained on Dataset B (Fantasy sister series).

Figure 11. Comparison of the line art colorization results. The first column shows the original ground truth image. Columns 2–5 show the automatic colorization results; from left to right: Seg2pix, PaintsChainer (Canna) [6], Tag2pix [16], and Style2paints [7]. Rows 1–3 of the Seg2pix column are the results from the model trained on Dataset C (Fantasy sister series), and Rows 4–6 are the results from the model trained on Dataset D (Fantasy sister series).

6. Conclusions

In this paper, we propose an automatic colorization network, which produces high-quality line art colorization with only a few training images. We applied data preprocessing steps before training to achieve high-quality segmentation images from the generator.

The generated segmented image becomes the suggestion map for auto-colorization and generating the step runs again for coloring, which is the step that is the most different from the Pix2pix method. We showed its performance through experiments with sets of character image data obtained from well-known webtoon series. The experimental results showed the potential of the method to reduce the amount of iterative work in practice and supporting artists to focus on the more creative aspects of the webtoon creation pipeline. As an idea for real use scenarios, we can colorize 100 sketch images of the same character by coloring and labeling only 10 or up to 20 sketch images with this Seg2pix method. Especially in the webtoon and animation industries, coloring over thousands of the same character's sketch images is a repetitive job, so the contribution of this method is very suitable in these industries. For future research, it will be required to implement a random character's line art with a single trained model to achieve application quality in the industry, while our method focuses only a particular character's line art. To accomplish this future objective, we need to collect more and various data of character line art images and hand-labeled segmented images by utilizing our Seg2pix network in industrial applications.

Author Contributions: Writing—original draft and data curation, S.C.W.; formal analysis, S.C.W.; writing—review and editing, S.C.W.; validation S.Y.; supervision S.Y. All authors have read and agreed to the published version of the manuscript

Funding: This work was funded by the Korean government (MSIT) (No. GK19P0300, Real-time 4D reconstruction of dynamic objects for ultra-realistic service).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Restrictions apply to the availability of these data. Data was obtained from Naver Webtoon services and are available with the permission of Naver webtoon company and original authors.

Acknowledgments: This work was supported by NAVER Webtoon corporation's copyright data and "The Cross-Ministry Giga KOREA Project" grant funded by the Korean government (MSIT) (No. GK19P0300, Real-time 4D reconstruction of dynamic objects for ultra-realistic service).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Dutchess County, NY, USA, 2014; Volume 27, pp. 2672–2680.
2. Isola, P.; Zhu, J.Y.; Zhou, T.; Efros, A.A. Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1125–1134.
3. Ci, Y.; Ma, X.; Wang, Z.; Li, H.; Luo, Z. User-guided deep anime line art colorization with conditional adversarial networks. In Proceedings of the 26th ACM International Conference on Multimedia, Seoul, Korea, 22–26 October 2018; pp. 1536–1544.
4. Hati, Y.; Jouet, G.; Rousseaux, F.; Duhart, C. PaintsTorch: A User-Guided Anime Line Art Colorization Tool with Double Generator Conditional Adversarial Network. In Proceedings of the European Conference on Visual Media Production, London, UK, 13–14 December 2019; pp. 1–10.
5. Furusawa, C.; Hiroshiba, K.; Ogaki, K.; Odagiri, Y. Comicolorization: Semi-automatic manga colorization. In Proceedings of the SIGGRAPH Asia 2017 Technical Briefs (SA '17), Association for Computing Machinery, New York, NY, USA, 27–30 November 2017; Article 12, pp. 1–4. [CrossRef]
6. Zhang, L.; Ji, Y.; Lin, X.; Liu, C. Style transfer for anime sketches with enhanced residual u-net and auxiliary classifier gan. In Proceedings of the 2017 4th IAPR Asian Conference on Pattern Recognition (ACPR), Nanjing, China, 26–29 November 2017; pp. 506–511. [CrossRef]
7. Zhang, L.; Li, C.; Wong, T.T.; Ji, Y.; Liu, C. Two-stage sketch colorization. *ACM Trans. Graph. (TOG)* **2018**, *37*, 1–14. [CrossRef]
8. Naoto Hosoda, Wasure Rareta Mirai Wo Motomete. 2014. Available online: http://ushinawareta-mirai.com/ (accessed on 22 April 2020).
9. BanG Dream! Girls Band Party! PICO 2018. ©Bushiroad Co., Ltd. Available online: https://www.youtube.com/c/BanGDreamGirlsBandParty/featured (accessed on 13 March 2020).

10. Kanopoulos, N.; Vasanthavada, N.; Baker, R.L. Design of an image edge detection filter using the Sobel operator. *IEEE J. Solid State Circuits* **1988**, *23*, 358–367. [CrossRef]
11. Canny, J. A Computational Approach to Edge Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1986**, *PAMI-8*, 679–698. [CrossRef]
12. lllyasviel, SketchKeras. Available online: https://github.com/lllyasviel/sketchKeras (accessed on 22 February 2020).
13. Huang, Y.C.; Tung, Y.S.; Chen, J.C.; Wang, S.W.; Wu, J.L. An adaptive edge detection based colorization algorithm and its applications. In Proceedings of the 13th Annual ACM International Conference on Multimedia, Singapore, 2–10 November 2005; pp. 351–354.
14. Levin, A.; Lischinski, D.; Weiss, Y. Colorization using optimization. In Proceedings of the ACM SIGGRAPH 2004 Papers (SIGGRAPH '04), Association for Computing Machinery, New York, NY, USA, 8–12 August 2004; pp. 689–694. [CrossRef]
15. Karras, T.; Laine, S.; Aila, T. A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 4401–4410.
16. Kim, H.; Jhoo, H.Y.; Park, E.; Yoo, S. Tag2pix: Line art colorization using text tag with secat and changing loss. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019; pp. 9056–9065.
17. Ren, H.; Li, J.; Gao, N. Two-stage sketch colorization with color parsing. *IEEE Access* **2019**, *8*, 44599–44610. [CrossRef]
18. Zheng, Y.; Yao, H.; Sun, X. Deep semantic parsing of freehand sketches with homogeneous transformation, soft-weighted loss, and staged learning. *IEEE Trans. Multimed.* **2020**. [CrossRef]
19. Sarvadevabhatla, R.K.; Dwivedi, I.; Biswas, A.; Manocha, S. SketchParse: Towards rich descriptions for poorly drawn sketches using multi-task hierarchical deep networks. In Proceedings of the 25th ACM International Conference on Multimedia, Mountain View, CA, USA, 23–27 October 2017; pp. 10–18.
20. Wang, S.; Zhang, J.; Han, T.X.; Miao, Z. Sketch-based image retrieval through hypothesis-driven object boundary selection with HLR descriptor. *IEEE Trans. Multimed.* **2015**, *17*, 1045–1057. [CrossRef]
21. Tolias, G.; Chum, O. Asymmetric feature maps with application to sketch based retrieval. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2377–2385.
22. Zhang, Y.; Qian, X.; Tan, X.; Han, J.; Tang, Y. Sketch-based image retrieval by salient contour reinforcement. *IEEE Trans. Multimed.* **2016**, *18*, 1604–1615. [CrossRef]
23. Xu, K.; Chen, K.; Fu, H.; Sun, W.L.; Hu, S.M. Sketch2Scene: Sketch-based co-retrieval and co-placement of 3D models. *ACM Trans. Graph. (TOG)* **2013**, *32*, 1–15. [CrossRef]
24. Shao, T.; Xu, W.; Yin, K.; Wang, J.; Zhou, K.; Guo, B. Discriminative sketch-based 3d model retrieval via robust shape matching. In *Computer Graphics Forum*; Blackwell Publishing Ltd.: Oxford, UK, 2011; Volume 30, pp. 2011–2020.
25. Wang, F.; Kang, L.; Li, Y. Sketch-based 3d shape retrieval using convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1875–1883.
26. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
27. Chen, L.C.; Papandreou, G.; Schroff, F.; Adam, H. Rethinking Atrous Convolution for Semantic Image Segmentation. *arXiv* **2017**, arXiv:1706.05587.
28. Chen, L.C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 834–848. [CrossRef] [PubMed]
29. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015; pp. 234–241.
30. Dong, J.; Cong, Y.; Sun, G.; Hou, D. Semantic-transferable weakly-supervised endoscopic lesions segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019; pp. 10712–10721.
31. Dong, J.; Cong, Y.; Sun, G.; Zhong, B.; Xu, X. What can be transferred: Unsupervised domain adaptation for endoscopic lesions segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 4023–4032.
32. Yumi's Cell 2015. Yumi's Cell Webtoon Series from Naver Webtoon. Available online: https://comic.naver.com/webtoon/list.nhn?titleId=651673 (accessed on 2 March 2020).
33. Fantasy Sister 2019. Fantasy Sister Webtoon Series from Naver Webtoon. Available online: https://comic.naver.com/webtoon/list.nhn?titleId=730425 (accessed on 2 March 2020).
34. Zhang, S.H.; Chen, T.; Zhang, Y.F.; Hu, S.M.; Martin, R.R. Vectorizing cartoon animations. *IEEE Trans. Vis. Comput. Graph.* **2009**, *15*, 618–629. [CrossRef] [PubMed]
35. Hebesu,Trap-Ball Segmentaion Linefille. Available online: https://github.com/hepesu/LineFiller (accessed on 15 March 2020).
36. Hinton, G.E.; Salakhutdinov, R.R. Reducing the dimensionality of data with neural networks. *Science* **2006**, *313*, 504–507. [CrossRef] [PubMed]
37. Li, C.; Wand, M. Precomputed real-time texture synthesis with Markovian generative adversarial networks. In Proceedings of the European Conference on Computer Vision, Munich, German, 8–16 October 2016; pp. 702–716.
38. Zhang, H.; Goodfellow, I.; Metaxas, D.; Odena, A. Self-attention generative adversarial networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 May 2019; pp. 7354-7363.

39. Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic differentiation in pytorch. In Proceedings of the NIPS 2017 Workshop Autodiff Submission, Long Beach, CA, USA, 4–9 December 2017.
40. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv*, **2014**, arXiv:1412.6980.
41. Zhang, R.; Isola, P.; Efros, A.A.; Shechtman, E.; Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 586–595.
42. Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *arXiv* **2017**, arXiv:1706.08500.