Into the Colorful World of Webtoons: Through the Lens of Neural Networks

Ceyda Cinarel
Department of Computer Science and Engineering
Seoul National University
Seoul 151-742, Republic of Korea
ceyda@bi.snu.ac.kr

Byoung-Tak Zhang
Department of Computer Science and Engineering,
Brain Science Program
Seoul National University &
Surromind Robotics
Seoul 151-742, Republic of Korea
btzhang@bi.snu.ac.kr

***Abstract*—The task of colorizing black and white images has previously been explored for natural images. In this paper we look at the task of colorization on a different domain: webtoons. To our knowledge this type of dataset hasn't been used before. Webtoons are usually produced in color thus they make a good dataset for analyzing different colorization models. Comics like webtoons also present some additional challenges over natural images, such as occlusion by speech bubbles and text. First we look at some of the previously introduced models' performance on this task and suggest modifications to address their problems. We propose a new model composed of two networks; one network generates sparse color information and a second network uses this generated color information as input to apply color to the whole image. These two networks are trained end-to-end. Our proposed model solves some of the problems observed with other architectures, resulting in better colorizations.**

***Keywords*—colorization, webtoons, convolutional neural networks**

I. Introduction

Webtoons are an online form of comics that have gained significant popularity with the spread of mobile devices. Usually released on a weekly schedule and produced in full color they can be the work of a single artist or the work of multiple artists working together; a writer, an illustrator, a colorist... They are tailored for smart phones thus stories are told in a vertical layout. A quick scroll can give us a sense of that webtoon's individual visual style. Apart from the drawing style we can easily see the color pallet being used distinctly reflecting the identity of that webtoon.

In image processing, colorization is the task of generating colored images from grayscale inputs. From a computational point of view it can be formulated as mapping one dimension onto a three dimensional space.

Purpose of our research is twofold. Training a model capable of generating good possible colorizations while capturing the necessary features for colorization, without the need for post-processing steps. As well as getting better insight into neural network models used for colorization by evaluating them on a different domain.

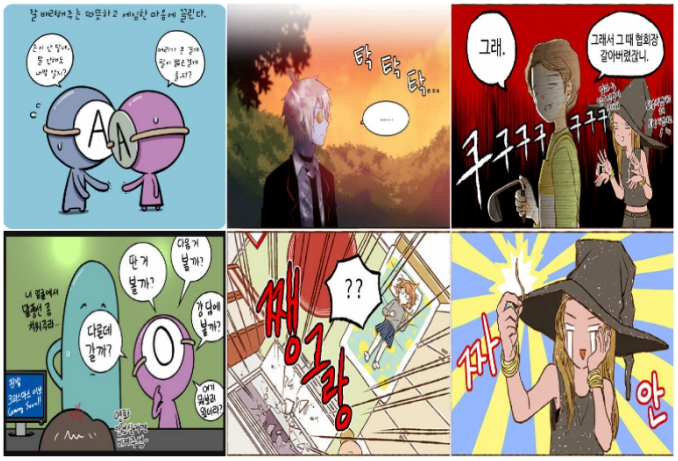


Fig. 1. Challenges of webtoons dataset; high multimodality, occlusion, texture...

A. Potential Uses

The task of colorization has so far been only researched on natural images. By looking at this task through the lens of a different domain such as comics we can gain insight into the inner workings of these models. Colorization was proven useful for downstream tasks and as a form of unsupervised pre-training [1], [2]. For example webtoons themselves can have (intentionally) uncolored panels, as in Figure 2, and colorizing these as a preprocessing step can help on tasks such as segmentation. We hypothesize that a network trained on webtoons data can learn features useful for other coloring tasks such as coloring sketches implicitly,i.e without needing a reference image.

Colorization is also explored as an avenue for image compression [3]. Although we focus on generating possible colorizations rather than being perfectly faithful to ground truth, we think our work is an important first step in this direction. Since webtoons are produced in color this makes them a good testbed dataset for these purposes.

B. Challenges

Compared to natural images colorization of webtoons can be more difficult due to occlusion by speech bubbles,

Fig. 2. Where generated colorizations are more colorful than the originals. This can be seen as success or failure depending on the intended application. We view this as success for its potential use.

text and texture effects, e.g. Figure 1. Moreover, comics also show higher multimodality where the possible set of colors for a given entity is potentially only limited by imagination. There can be purple apples in comics but not likely in natural images. There is also more variance in shapes.

There is currently no standard large dataset available for comics as in the case of natural images. For research we have to collect our own data which can be arduous and usually due to copyright sharing is not possible. The lack of pre-trained networks on this type of dataset which can provide important hints to the network such as semantic information also makes comics processing challenging. The work of [4] trained on illustrations for tag classification is the closest to comics domain.

II. Related Research

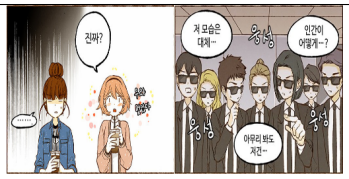
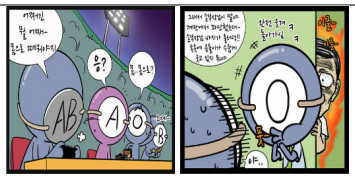
User input guided [5] or fully automatic [6], [7], [8], [1], [2], [9], [10], [11], [12] colorization systems based on neural networks making use of large datasets such as Imagenet [13] has been where the majority of colorization research focused on. To our knowledge there has been no previous works for colorizing comics such as webtoons. Although there are very recent works on colorizing sketches [14], [15], [16], [17], [18], [19], [20] our task is different in that we are colorizing grayscale images and not sketches.

Commonly when using natural images, pre-trained networks on tasks such as object classification and segmentation are leveraged. This type of auxiliary information is used in [2] when extracting features in form of hypercolumns, [10] uses features from the VGG[21] network, [6] uses additional classification loss,[7] also employ ResNet101 pre-trained on segmentation task. But for our task such pre-trained networks or labeled data is not present.

A key information from [7] is the observation that simply up-scaling the lower resolution version of the chrominance of an image can be sufficient for coloring, meaning only a little color information is needed. They train two networks individually; a coloring network that generates low resolution chrominance channels given grayscale image and a refinement network that combines this chrominance with the grayscale image. [16] also uses two networks trained individually for the task of colorizing sketches. Where a color generating network is trained by removing random patches of color. [8] introduces a model for the more abstract task of image-to-image translation. They use conditional generative adversarial networks(CGAN), conditioning on the input image.

Coloring can also be thought of as a style thus applying style transfer [22], [23] methods for this task also sounds viable. But there are a few things to be considered; style

TABLE I. Overview of the dataset.

Webtoon Name	Example Images	Data sizes
가우스전자 시즌3 by 곽백수		Train:1756 Val:98 Test:97
슈퍼 시크릿 by 이온		Train:5691 Val:316 Test:316
일상날개짓 by 나유진		Train:1056 Val:59 Test:58
크리퍼스쿨 by 밀치/얌치		Train:2468 Val:137 Test:137
혈액형에 관한 간단한 고찰 by 박동선		Train:2947 Val:164 Test:163

transfer methods such as [22] transfer the whole style, drawing and color. Whereas we would like to leave the drawing style untouched. [24], [25] transfers the drawing style and preserves the color which can be though of as the reverse direction of what we want to do. Color transfer methods [26] based on histogram matching or color mapping between two images exist for tasks like color correction. But all these methods rely on there being a reference image given explicitly. [27] does not require specific image pairs but sets of images, success of which depends on the similarity of the two images.

Colorization of sketches is similar to the task of colorizing grayscale images, and in a sense more difficult because the grayscale information is not present. That is why most systems rely on coloring based on a reference image [17], [18], [19]. But when the reference image differs in composition from the target image to be colorized, the colorization may fail altogether (grayish outputs). [19] modifies the main architecture introduced in [6] by adding the histogram of a reference image in the fuse layer, followed by post-processing steps. [18] uses the network of [8] on a single reference image, followed by post-processing using segmentation to deal with the extra colorfulness of results generated by conditional GANs [28]. This extra colorfulness is also something we observe in our experiments as a result of adversarial loss.

Fig. 3. Common types of fail cases of all models. 1st column; although models successfully color that character's hair in other panels (below) they fail on this instance due to the applied grainy texture. 2nd column; Occlusion results in murky coloring, although successfully colored when fully visible (below). 3rd column; Fails on multimodality, results in the most common colors being selected for coloring. 4th column;text and effects contains very similar light tones only. 5th column; complicated backgrounds

III. The Dataset

For the reasons we mentioned above we chose to work on webtoons. Unfortunately, there was no such dataset readily available at the time. Thus we chose to collect our own dataset by scraping Naver webtoons [1]

A. Preprocessing

As with any scraped raw data we needed to do some preprocessing, using the following steps:

Firstly, because the scraped images contained multiple panels per image we cropped them into panels using a method based on the common background color. Although this did not always produce exact results it was mostly accurate.
Further to get rid off any meaningless remnants made by the cropping process we removed images smaller or larger than 200 and 800 pixels in width or height. Because webtoons are usually intended for being read on a mobile device their panels' size more or less consistent.
Between each step we removed duplicate images since things like title and ending panels are repeated. And as a final step, naturally grayscale images were removed which got rid off things like panels with speech balloons only. We did no additional pruning of the data keeping noisy and challenging images.

B. Properties

An overview of the dataset can be seen in Table I. Since consecutive panels maybe very similar the train test data split was done based on different issues of the webtoon. Otherwise we may overestimate the success of a model since images with very similar composition can end up split between the train and test set.

We have picked these five webtoons as our dataset for highlighting some of the potential difficulties of the task we mentioned as seen in Figure 1. They contain; similar/same looking characters with different colors, fine color gradients in the light effects, panels with screen-tones only or with sparse color, complicated or simple solid backgrounds, colored speech balloons... By training using a large webtoon collection, compared to using a single webtoon, the network can generalize

[1] http://comic.naver.com/index.nhn

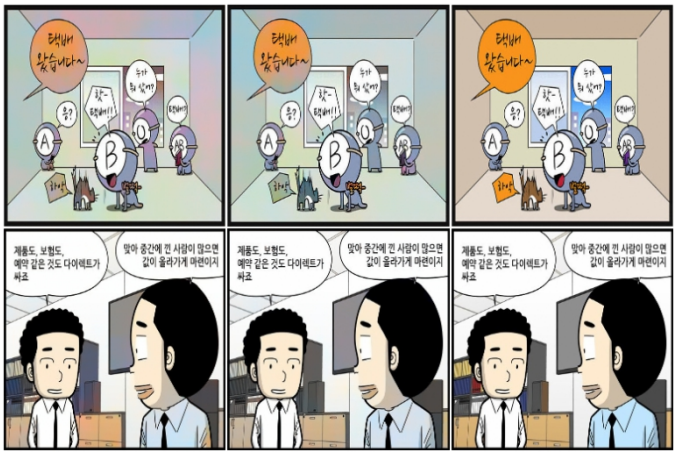


Fig. 4. From left to right: with adversarial loss, L2 loss only, ground truth. Top row: showing jumbled background colors. Bottom row: with adversarial loss in some cases it is possible to get less washed out results.

better enabling it to color the types of panels that might be uncharacteristic for a specific webtoon alone. We collected a larger dataset that consists of 25 different webtoons but decided to run our experiments first with this smaller dataset to investigate the characteristics of the possible fail cases made by various models in order to find the proper architecture for this task. Common failures of models include; missing small details in effects and accents, failing in presence of texture and occlusion, complicated backgrounds, color bleeding outside the lines. Some examples are shown in Figure 3.

The colorspace to represent the data in is also an important choice. Colorspaces such as YUV, HSV and Lab allow easily separating the chrominance of an image from its grayscale values. We chose to use the Lab colorspace because it was developed based on human vision and its channels are close to independent [26]. In order to keep the full range of colors we do not quantize (bin) the colorspace unlike [2], [1], [7].

IV. Experiments

We focus on the model of [8] using the encoder-decoder architecture of the generator for our networks. We use layers as described in [8] adjusting the first layer depending on the input. When using VGG we concatenate the features at the layers with the same dimensions. The U-net [29], [8], [17] structure that adds skip connections between encoder and decoder was very important without these skip connections models weren't able to generate good colorizations.

We explain the various architectures considered leading to the development of a model we will refer to as the color & apply network where we achieve the best results. In line with our purpose we evaluate models on whether the generated coloring is consistent with that webtoons style and free of artifacts we described. All the results in the figures are from the test dataset, with the exception of Figure 5 and 6. All models were were trained for their best results, decided on the validation set for fair comparison. Differences between images are best viewed on fullscreen.

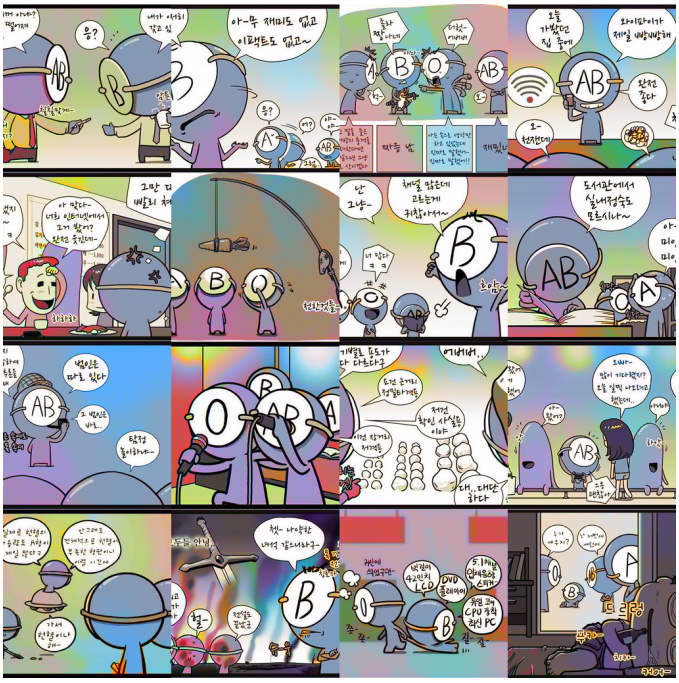


Fig. 5. Model of [9] trained on a single class. Even though the images were fully colored, there was no consistency and included many highly saturated areas even from results on the training dataset

A. The Loss function

We investigated the effects of several loss functions.

Adversarial loss - In [8] a PatchGAN discriminator is used and the loss function is defined as adversarial loss plus L1 loss. While many other works use L2 loss only; with [6], [1], [2] or without [10] some modifications. That is why we wanted to compare the effects of adversarial loss on colorization. Although using adversarial loss in addition to the L2 loss might be better for the more general task of image-to-image translation [8] we weren't able to observe significant benefits for the task of colorization. Adversarial loss did generate more colorful results as reported in the original paper [8]. Although this colourfulness may look more satisfying in some cases such as coloring sketches [16], [17] it results in inconsistent coloring. For example while coloring solid color backgrounds as seen in figure 4 top row. On the validation set there were no outstanding perceptual differences and numerically L2 losses throughout the training were also very similar. The average L2 loss on the test set was 1578.8 for the model using only L2 loss, whereas it was 1595.0 for the model with adversarial loss. In both cases, more so with adversarial loss, there was a type of color confusion/uncertainty happening around areas with speech bubbles.

Conditioning on the label - Since knowing which webtoon an image belongs to can reduce the set of possible colors we tried also conditioning on the label with the adversarial model. But this didn't improve and in some cases decreased the quality. We hypothesize that this is because the model was already able to infer the label without being conditioned on it, and this explicit conditioning decreased the model capacity by a small margin.

Classification loss - Additional to L2 loss we defined another type of loss which is calculated by running the generated image through a VGG16 classifier network. This VGG16 network, explained in more detail below, was trained to classify colored webtoon images. We would like to note this method is highly dependent on the quality of the classifier used. In our experiments we didn't see any benefits from adding this loss probably due to the classifier being able to classify the image easily, a classifier more sensitive to the color might be necessary for this approach to work which we leave for our future work.

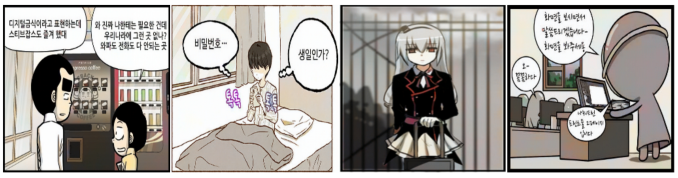


Fig. 6. Training with sparse color information led the network to focus on the details (smaller regions)

Fig. 7. On the left are images generated at the first iteration, on the right are images generated at the second iteration by using left image as input.

We also tried the models of [9] that use variational auto encoders. Results can be seen in Figure 5. Although this model was trained on a single class for a longer time it didn't reach the results we expected so we didn't followup on the variations of this architecture.

In order to alleviate the problems of jumbled colors and color bleeding seen in models so far, we tried two different approaches. Using an auxiliary pre-trained network and training two networks each focused on a different part of the task. We will talk about these methods in the following subsections.

B. Using Features from Pre-trained Networks

As we have mentioned in our review of the related research, pre-trained networks are often used for adding semantic information to a model. At first we looked at the work of [4] since classifying tags for illustrations is closer to our domain than natural images. In our preliminary work we used pre-trained weights of [4] with the model of [10] but didn't see significant gains compared to training from scratch.

We also trained ResNet-50 and VGG16 models on webtoon classification task. Two versions of each were trained; one on color images another on grayscale. There wasn't a significant difference in classification accuracy between these two versions. Looking at the activations the ResNet-50 [30] network didn't learn any features beyond the first few layers even on the larger dataset with 25 classes. This is probably due to the task being to simple for training such a deep network. ResNet-50 also achieved lower accuracy compared to VGG16 when the number of classes were 5.

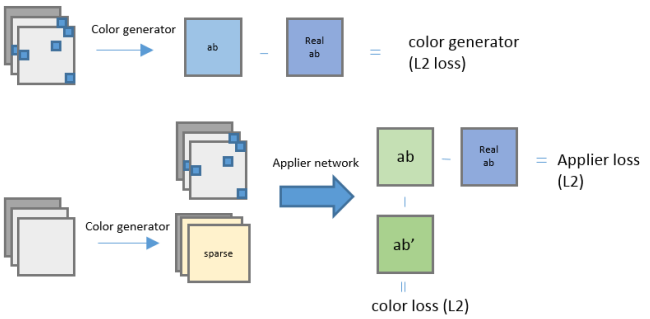


Fig. 8. The general model architecture. ab' is chroma generated by using the output of color generator where the color generator only gets the grayscale image as input.

Overall we found the features learned by VGG16 [21] network architecture better suited. We used the outputs of layers *conv_1_2*, *conv_2_2* and *conv_3_1* by concatenating them to the encoder at the suitable layers. Looking at the results on the test set there was some minor perceptual improvements on some cases. The average L2 loss was also slightly lower 1561.3.

There are variations as to how to train this auxiliary network best for it to select features useful for the colorization task. We chose to train on grayscale images as this would be the type of image available on inference time.

C. Coloring Within the Lines

There were many instances where the network would generate the correct color for a small area of the object but fail to apply it to the whole. Also another common type of situation was where the correct hue would be generated but with lower saturation than expected.
To address these problems we thought of using an iterative generation process where at generation time the output of the network could be fed back to itself as the input, iterating as many times as needed. To make this network learn how to generate colors, at training time the network would be given very sparse color information. This can be though of as applying dropout to the inputs but only on the chroma channels. One interesting result of training the network on sparse color information was that it led the network to focus on the less commonly occurring colors in the smaller details, seen in Figure 6. We also tried drop out on the channel level, omitting one channel at a time, which learned the more general colors.

But this single network approach resulted in very saturated results because the output at first iteration would contain more color information than the network is trained on resulting on the following iterations being more and more saturated, as can be seen in Figure 7. Also the errors made at the first iteration would get propagated. So instead of tasking one network with both generation and application of color we decided to train two networks. Where one network generates sparse colors and the other network uses this as input to color the whole picture. We will refer to them as the coloring network and the applier network as it learns how to apply the given colors to the image. An overview of the model architecture can be seen

Fig. 9. Left images L2 loss only, right images from the color & apply network. Our colorize & apply model is better at generating more consistent colors for continuous regions with solid color and reduces color bleeding

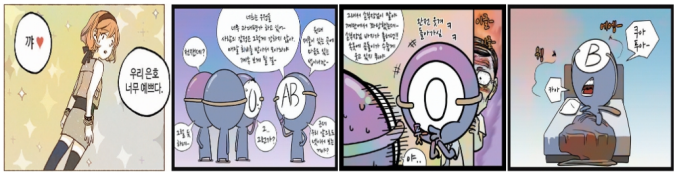


Fig. 10. Typical mistakes of the color & apply network; segments(1st and 2nd pictures) or boundary like areas(4th and rightmost picture) are colored differently. In the 3rd picture the green segment is colored wrong due to segment shape similarity

in Figure 8 This can be though of as a person being given a choice of colors for coloring within the lines. In theory the first network can be replaced with user input when user interaction is required, although we haven't tried this.

We define *image'* to be the image where dropout is applied to the ab channels with a keep probability of 0.01. And *generated'* to be the image chroma generated by that network from the input *image'*. The loss for the applier network is the L2 loss between *generated'* chroma and the ground truth image chroma. The color generator is trained with color generator loss plus color loss.

This method is somewhat similar to that of [16], [8]. Differently from these models our colorization network generates sparse color information rather than a low resolution one. Also differently, we train our two networks end-to-end. [8] uses a pre-trained ResNet-101[30] on segmentation, we do not employ but believe that our network can also benefit from such segmentation information. The method of removing random patches for training used in [16] was only able to produce sepia tones when we tried to replicate its results using its open sourced code.

This model was especially effective on reducing the multiple color effect observed on backgrounds. Comparison between this model and model using L2 loss only can be seen in Figure 9. The amount of dropout applied while training the applier network effects the color information learned by coloring network indirectly through color loss.

V. Discussions and Future Work

In this paper we introduced the task of colorizing webtoons, establishing webtoons as a worthwhile and challenging dataset. We covered the existing models and brought to attention the potential of using a dataset of a different domain. Through this approach, during our research we were able to see the artifacts created by different models and gained a deeper understanding of adversarial networks. We believe it is important to also focus on the examples the models fail on and shared some instances that fell short of our expectations.

For our future work we will be focusing on improving the color & apply network while using a larger dataset. Training using dropout also for the lightness channel would be a simple extension of this model, potentially allowing the use of pre-trained features for colorizing sketches. Which is something we will be doing in our future work.

Appendix

Implementation details

All image preprocessing was done with ImageMagick®. In all the experiments we use images resized to 256x256. Conversions between different color spaces were done using OpenCV library. We suggest to be careful of mixing different libraries for this purpose as numerical ranges used to represent the images may not be consistent between libraries. Also the precision loss due to conversions between integer and float is another point to be careful of. All experiments were implemented with Tensorflow and run on Nvidia Titan X GPU.

Acknowledgment

We would like to thank all the artist for their beautiful works. All the webtoons used belong to their respective original artists, published by Naver Corporation. This work was partly supported by the Ministry of Science and ICT (2015-0-00310-SW.StarLab) and Korea Evaluation Institute of Industrial Technology (10044009-HRI.MESSI, 10060086-RISF).

References

[1] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *European Conference on Computer Vision*. Springer, 2016, pp. 649–666.

[2] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *European Conference on Computer Vision*. Springer, 2016, pp. 577–593.

[3] M. H. Baig and L. Torresani, "Multiple hypothesis colorization and its application to image compression," *Computer Vision and Image Understanding*, 2017.

[4] M. Saito and Y. Matsui, "Illustration2vec: a semantic vector representation of illustrations," in *SIGGRAPH Asia 2015 Technical Briefs*. ACM, 2015, p. 5.

[5] 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," *arXiv preprint arXiv:1705.02999*, 2017.

[6] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Let there be color!: Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification."

[7] S. Guadarrama, R. Dahl, D. Bieber, M. Norouzi, J. Shlens, and K. Murphy, "Pixcolor: Pixel recursive colorization," *arXiv preprint arXiv:1705.07208*, 2017.

[8] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arXiv preprint arXiv:1611.07004*, 2016.

[9] Y. Cao, Z. Zhou, W. Zhang, and Y. Yu, "Unsupervised diverse colorization via generative adversarial networks," *arXiv preprint arXiv:1702.06674*, 2017.

[10] R. Dahl, "Automatic colorization," 2016.

[11] A. Deshpande, J. Lu, M.-C. Yeh, and D. Forsyth, "Learning diverse image colorization," *arXiv preprint arXiv:1612.01958*, 2016.

[12] Z. Cheng, Q. Yang, and B. Sheng, "Deep colorization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 415–423.

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 248–255.

[14] Y. Güçlütürk, U. Güçlü, R. van Lier, and M. A. van Gerven, "Convolutional sketch inversion," in *European Conference on Computer Vision*. Springer, 2016, pp. 810–824.

[15] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays, "Scribbler: Controlling deep image synthesis with sketch and color," *arXiv preprint arXiv:1612.00835*, 2016.

[16] K. Frans, "Outline colorization through tandem adversarial networks," *arXiv preprint arXiv:1704.08834*, 2017.

[17] L. Zhang, Y. Ji, and X. Lin, "Style transfer for anime sketches with enhanced residual u-net and auxiliary classifier gan," *arXiv preprint arXiv:1706.03319*, 2017.

[18] P. Hensman and K. Aizawa, "cgan-based manga colorization using a single training image," *arXiv preprint arXiv:1706.06918*, 2017.

[19] C. Furusawa, K. Hiroshiba, K. Ogaki, and Y. Odagiri, "Comicolorization: Semi-automatic manga colorization," *arXiv preprint arXiv:1706.06759*, 2017.

[20] Y. Liu, Z. Qin, Z. Luo, and H. Wang, "Auto-painter: Cartoon image generation from sketch by using conditional generative adversarial networks," *arXiv preprint arXiv:1705.01908*, 2017.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[22] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2414–2423.

[23] ——, "A neural algorithm of artistic style," *arXiv preprint arXiv:1508.06576*, 2015.

[24] L. A. Gatys, M. Bethge, A. Hertzmann, and E. Shechtman, "Preserving color in neural artistic style transfer," *arXiv preprint arXiv:1606.05897*, 2016.

[25] F. Luan, S. Paris, E. Shechtman, and K. Bala, "Deep photo style transfer," *arXiv preprint arXiv:1703.07511*, 2017.

[26] H. S. Faridul, T. Pouli, C. Chamaret, J. Stauder, E. Reinhard, D. Kuzovkin, and A. Trémeau, "Colour mapping: A review of recent methods, extensions and applications," in *Computer Graphics Forum*, vol. 35, no. 1. Wiley Online Library, 2016, pp. 59–88.

[27] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *arXiv preprint arXiv:1703.10593*, 2017.

[28] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[29] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.