applied sciences

MDPI

Article

Progressive Full Data Convolutional Neural Networks for Line Extraction from Anime-Style Illustrations

Yuchen Xin, Hon-Cheng Wong *, Sio-Long Lo and Junliang Li

Faculty of Information Technology, Macau University of Science and Technology, Macao, China; dangoneko@hotmail.com (Y.X.); sllo@must.edu.mo (S.-L.L.); lijunliang_sherlock@hotmail.com (J.L.)

* Correspondence: hcwong@ieee.org

Received: 15 November 2019; Accepted: 17 December 2019; Published: 19 December 2019

Abstract: Anime-style comics are popular world-wide and an important industry in Asia. However, the output quantity and quality control of art workers have become the biggest obstacle to industrialization, and it is time consuming to produce new manga without the help of an intelligence assisted tool. As deep learning techniques have achieved great successes in different areas, it is worth exploring them to develop algorithms and systems for computational manga. Extracting line drawings from finished illustrations is one of the main tasks in drawing a manuscript and also a crucial task in the common painting process. However, traditional filters such as Sobel, Laplace, and Canny cannot output good results and require manual adjustments of the parameters. In order to address these problems, in this paper, we propose progressive full data convolutional neural networks for extracting lines from anime-style illustrations. Experimental results show that our progressive full data convolutional neural networks not only can learn as much as processing methods for the detailed regions, but also can accomplish the target extraction work with only a small training dataset.

Keywords: manga computation; convolutional neural networks; fine art; deep learning; line extraction

1. Introduction

The comic industry has been developing for many years and has become an important new industry. Comics are more and more popular all over the world; however, their productivity is still quite low. Comic artists are looking for solutions to save drawing time and improve the quality of comic artworks. In the general situation, they need to employ people to complete the time consuming work like re-drawing, sticking screen-tones, or coloring to produce comic artworks manually. They are trying to use computer technologies to improve the efficiency of creating comic works. However, there are still many problems. For example, in the animation production process, the quality of painting declines from the original image to the animated one because the painting style is hard to unify. Some painting details often cannot be improved due to the cost of time. In the industry, they often choose to ignore some defects such as the collapse of the character drawing quality from a distant view. In order to pursue a consistent visual effect, painters are often required to use a uniform drawing style. However, most of the painters cannot follow the style of the original paintings very well, so the correction to make the paintings that follow the stye of the original paintings becomes very laborious.

In recent years, deep learning has made important breakthroughs in many fields such as image recognition, speech recognition, and artistic creation, especially the outstanding performance of convolutional neural networks (CNNs). In order to solve some of the above mentioned problems in comics, we utilize CNNs to learn and create comic illustrations to reduce the time and costs in the production process. Although the outputs of comic illustrations are very complicated, we can decompose, learn, and create illustrations through CNNs to complete an illustration drawing system. The most important issue in the early days was how to prepare training data. At present, there are no ready-made databases that can be used. It will be inefficient if we rely on manual preparation of a database. In industrial production, there are already some rough data that can be used, but they are not enough to enable AI to complete high quality painting learning. For example, deciding whether some pixels belonging to the nearest character still relies on training with additional accurate data that are artificially provided.

For extracting lines, there are several traditional edge detectors such as Sobel, Laplace, and Canny [1,2]. There are also algorithms [3–5] based on texture information for line extraction. Some algorithms [6–8] were also designed for extracting comic screen patterns. However, these algorithms are mostly based on local gradient changes or only worked for certain patterns, so they are not good to extract lines with high accuracy from the human perspective. Usually, painters still need to adjust parameters manually to get better results after using these algorithms, and such adjustments cannot be done in a fully automatic or semi-automatic way.

In this paper, we propose a novel neural network structure to deal with line extraction from anime-style illustrations. The main contributions of this paper are listed as follows:

1. We implement our progressive full data convolutional neural networks (PFD-CNNs), which can obtain better accuracy in the line extraction problem.
2. Our approach has potential applications for non-photorealistic rendering.

Experimental results show the advantages of progressive full data convolutional neural networks on line extraction (see Figure 1), and these advantages have been recognized by some art workers through a preliminary user study. With our approach, we can also convert real-world images to line drafts or provide teaching materials for painting in the form of lines.

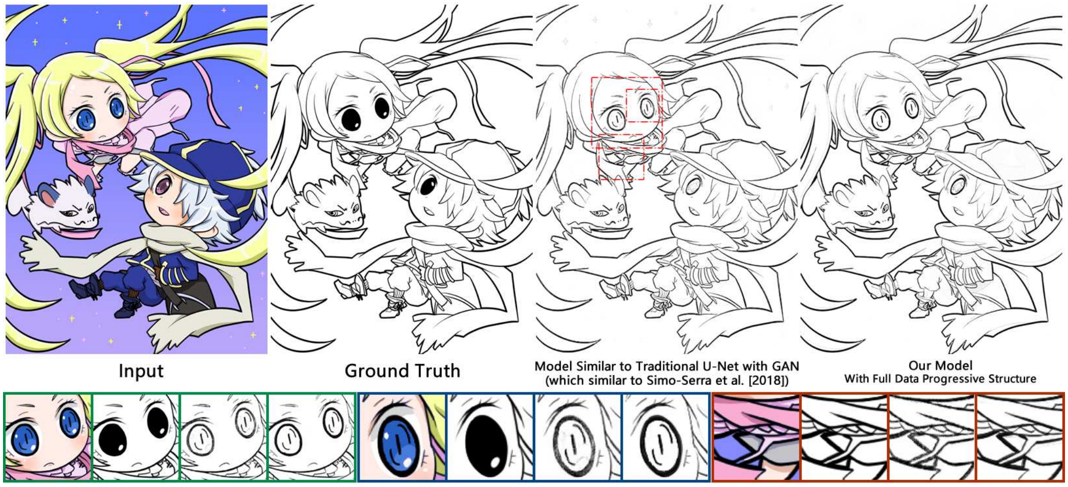

Input
Ground Truth
Model Similar to Traditional U-Net with GAN
(which similar to Simo-Serra et al. [2018])
Our Model
With Full Data Progressive Structure

Figure 1. Comparisons of input, ground truth, Simo-Serra et al.'s [9] model's output, and our model's output (artwork ids 12994997 and 14843300 from pixiv.net; the ground truth was drawn by Shinya Hragi and colored by Hamaguti Sakuya).

2. Related Work

Line extraction is one of the important topics in computational manga. There has been a certain number of papers published in recent years. Mao et al. [10] proposed an approach to extract lines based on regions for cartoons. Li et al. [11] removed the halftone screen and extracted manga structural lines from black-white manga with CNNs. Their network structural features were similar to that of U-Net [12], and they emphasized the establishment of a targeted training database, which would include targeted patterns. Liu et al. [13] developed a clear cartoon sketch simplification algorithm for sketch simplification with closure aware features.

Simo-Serra et al.'s [14] learning to simplify method is a good example of applying CNNs to manga computation. They used fully convolutional networks (FCNs) [15] to complete the neural network training for simplifying drawing sketches. Their experimental results showed that CNNs are still superior to traditional algorithms, although they could not explain the parameters of CNNs well. They also used CNNs to colorize images automatically with simultaneous classification [16]. These works demonstrated that CNN based methods can help artists in their creation processes. Zhang et al. [17] proposed a deep learning approach for user guided image colorization. Han et al. [18] proposed a novel unsupervised learning method for pixelization.

Cao et al.'s works [19–21] and Pang et al. [22] work focused on manga layout problems. Their proposed layout algorithms can arrange cartoon picture sequences [20], extract robust panels [22], and make attention directing composition of manga elements [21]. In [19], even the static manga could be used to generate animation, and better experience could be built in storytelling. On the other hand, Liu et al. [23] provided a solution by text detection to extract text aware balloons from manga.

There have been several works on manga computation. Qu et al. [24] proposed a novel colorization technique that works effectively on coloring halftone area. Later, they proposed a better method [25] to generate halftone image from the source pictures. Ci et al. [26] proposed a user guided deep method for amine line art colorization. Hensman and Aizawa [27] developed a conditional generative adversarial network (cGAN)based manga colorization framework using only a single training image. Zhu et al. [28] proposed a toon tracking algorithm for animations. This method can be used for ensuring the color of tracking regions in the corresponding key frames. Liu et al. [29] provided a novel approach to create the depth of animation key frames to generate stereoscopic animations.

3. Progressive Full Data Convolutional Neural Networks

CNNs [30] have been widely used in computer vision; however, the direct usage of CNNs on graphics problems may not give the desired results. In addition, to our best knowledge, there is no approach proposed for extracting lines from anime-style illustrations. Therefore, we propose a progressive full data convolutional neural network structure, which is shown in Figure 2. In our network structure, the input layer is connected to the same resolution level layers, and the original input information is preserved during the whole convolution process, which the different resolution level layers also have a similar structure. Figure 3 shows how our network structure is used to extract lines from anime-style illustrations.

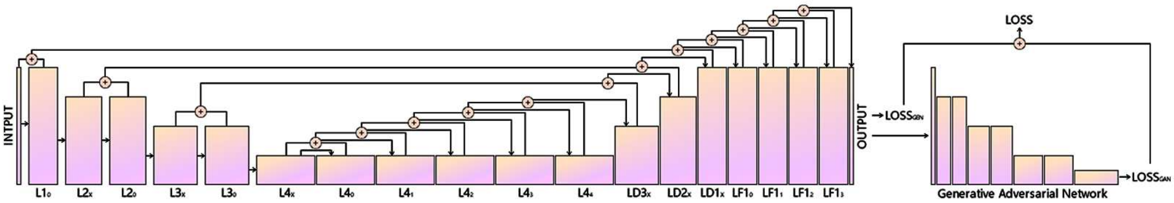


Figure 2. Our progressive full data convolutional neural network structure: different from traditional and state-of-the-art ones, our structure has more concatenated layers, which can provide more accurate pixel level results.

With the advantages of our network structure, we can reduce much of the preparation cost in training data. First of all, there is no requirement to recognize the content of the image; we only need to distinguish whether every pixel belongs to a line or not. The neural network does not need to classify the image content, so the training data can be easily prepared. Through our experimental tests, it was confirmed that our network structure could do pixel-wise image processing, and the line extraction ability was achieved with a small size of training data. Therefore, we did not need to highly increase the number of training samples. For most cases, we could apply general parameters to our neural network, which means that the process could be fully automated. Of course, it was also possible to adjust the parameters manually. The output quality was good enough, so we could work without post-processing such as vectorization to remove noise, which may in turn lead to the loss of image details, resulting in an unattractive picture.

Our approach focused on improving the neural networks for line extractions. The general convolution neural network extracts the features from the images in the convolution process, but also discards some of the detailed information, which is considered unnecessary. However, this information is exactly what we needed in this topic. In order to retain this information, a basic approach is not to use the down-sampling layer, that is not to reduce the layer resolution. However, this violates the basic ideas of the general convolutional neural network. We found that the convolution layer that was reduced and re-magnified back to the original resolution by convolutional layers could be merged with the original data as a new convolutional layer. In this way, we could guarantee that all the information of the input data was involved in the calculation of the convolutional neural network. Figure 2 shows all the data asymptotic structures.

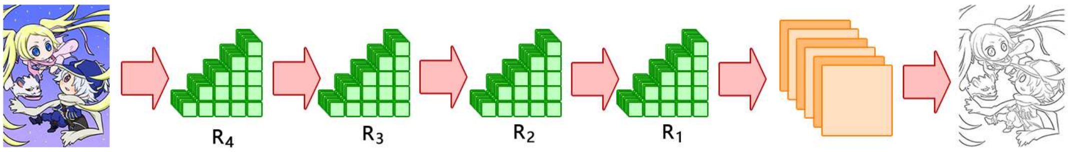


Figure 3. The conceptual view of our progressive full data neural network for extracting lines in anime-style illustrations.

The ability of convolutional neural networks has been demonstrated by many papers. Our network structure explored several popular network structures. The fully convolutional network (FCN) provides the idea of up-sampling convolution, making it possible for the convolutional neural network to deal with the pixel-wise problems. At the same time, U-Net uses the connection method to merge convolution layers with different resolutions and obtain more convolutional information. ResNet [31] provides ideas on how to connect more information effectively. Finally, the generative adversarial net (GAN) [32] was added to our network structure. In our network structure, GAN only used an unverified structure to assist the system. Therefore, it could not completely find the effect of GAN on the entire system and only ensured that the entire system could work well. If we did not add GAN, there may have been some noise. The GAN's main role here was to make the output style approach the style of the ground truth. The final output processed with GAN and without GAN results had its advantages and disadvantages, and one may choose whether to add it according to the actual requirements. Finally, we had our current network structure, a progressive and hierarchical U shaped neural network. Its global schematic diagram is shown Figure 2.

Our system's input was a color image with alpha channel $X \in \mathbb{R}^{H \times W \times 4}$, which presented the original anime-style illustration. The output of the system was $\hat{Y} \in \mathbb{R}^{H \times W \times 4}$, which was the line extraction result. Our CNN model F_{CNN} is specified in Figure 2, and parameterized by θ, trained to minimize the object function in Equation (1) with database D, which contained the source illustrations and the corresponding line extraction outputs. The loss function $\mathfrak{L}$ calculated in Equation (2) describes the accuracy of our model.

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{X,Y \sim D} \left[\mathfrak{L}(F_{CNN}(X,\theta), Y) \right] \tag{1}$$

Progressive full data convolutional neural network: The purpose of our network structure was to include all the detailed information in the convolution process. We merged all the important layers into each convolution process. For example, in the bottom of the gradual structure diagram, we connected $L4_X$, $L4_0$, $L4_1$, $L4_2$, $L4_3$ layers together and used them to generate $L4_4$ layer. Where $L4_0$ was generated by the convolution of $L4_X$, and $L4_1$ was generated by the merged layer of $L4_X$ and $L4_0$, they were proceeded step-by-step, as shown in the following equations.

$$L4_0 = F_{conv}(L4_X),$$

$$L4_1 = F_{conv}(F_{concat}(L4_X, L4_0)),$$
$$L4_2 = F_{conv}(F_{concat}(L4_X, L4_0, L4_1)),$$
$$L4_3 = F_{conv}(F_{concat}(L4_X, L4_0, L4_1, L4_2)),$$
$$L4_4 = F_{conv}(F_{concat}(L4_X, L4_0, L4_1, L4_2, L4_3)),$$

The reason why so many layers were set is that different layers could be extrapolated to the data of different natures. These data were becoming abstract layer-by-layer, and we chose the more meaningful first four or five layers to do the calculation. The structure of the last level is shown in Figure 3.

The key concepts in different stages of the network structure are explained here and depicted in Figures 3 and 4.

1. Full data: Our purpose was to keep all the data, so that all the previous convolutional outputs would participate in the outputs of the next convolutional layers, and the original input would also be added; in this case, the system's computational burden would increase, but it ensured that the following layers of the neural network would generate the result with the whole information in the original input.
2. Progressive full data: After ensuring that the source data could be obtained by each convolutional layer, as many as possible, we also considered the interim results produced as important information; from the visualization results of a neural network analysis [33], we could know that interim results represented different levels of feature data; therefore, in the same resolution, we used the layer concatenation function, as much as possible to make the convolution layer the same level that could refer to the results generated before.
3. Globally progressive full data: Finally, we needed to refer to and modify the U-Net network structure. We could not use layer concatenation to connect the output of different resolutions, thus all the concatenations must be completed at the same resolution, and finally, we had our neural network structure depicted in Figure 2.

According to the ideas provided in [33], the first five layers contained rich information such as color, edge, texture, and abstraction information. However, the information contained after the fifth layer was more abstract and of lower value. In our proposed globally progressive full data structure, we did not skip the links to certain convolutional layers because in our understanding, none of the first five layers of data could easily be skipped, even it were to be streamlined. Even the neural networks, which reduced the computational burden, should not omit the key convolutional layers.

Layer concatenation: We used the concatenate function to connect the different resolutions of convolutional layers. In this way, the low resolution and high resolution features were perceived together by neural network. We obtained the half-size layers with stride of two convolution layers and recovered the size in the corresponding de-convolution layers. Our network structure in Figure 2 shows how we concatenated these layers. The parameters of the layers are shown in Table 1.

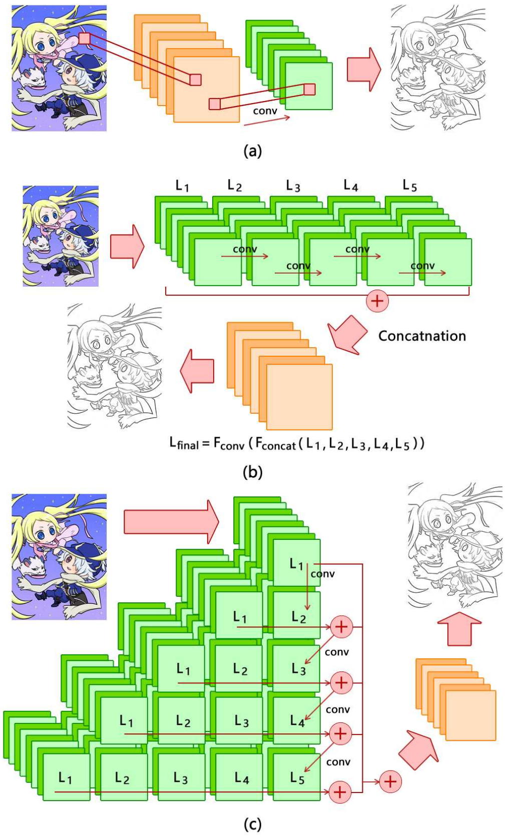


Figure 4. Different designs of neural network structures: (**a**) traditional neural network; (**b**) full data neural network; (**c**) progressive full data neural network.

Table 1. List of layer parameters.

Type	Kernel Size	Stride	Output Size
Input	-	-	$4 \times H \times W$
Convolution	3×3	1×1	$32 \times H \times W$
Convolution	5×5	2×2	$64 \times H/_2 \times W/_2$
Convolution	3×3	1×1	$64 \times H/_2 \times W/_2$
Convolution	5×5	2×2	$128 \times H/_4 \times W/_4$
Convolution	3×3	1×1	$128 \times H/_4 \times W/_4$
Convolution	5×5	2×2	$256 \times H/_8 \times W/_8$
Convolution	3×3	1×1	$256 \times H/_8 \times W/_8$
Convolution	3×3	1×1	$512 \times H/_8 \times W/_8$
Convolution	3×3	1×1	$768 \times H/_8 \times W/_8$
Convolution	3×3	1×1	$1024 \times H/_8 \times W/_8$
Convolution	3×3	1×1	$1280 \times H/_8 \times W/_8$
De-convolution	4×4	2×2	$128 \times H/_4 \times W/_4$
De-convolution	4×4	2×2	$64 \times H/_2 \times W/_2$
De-convolution	4×4	2×2	$32 \times H \times W$
Convolution	3×3	1×1	$32 \times H \times W$
Convolution	3×3	1×1	$32 \times H \times W$
Convolution	3×3	1×1	$32 \times H \times W$
Convolution	3×3	1×1	$32 \times H \times W$
Output	3×3	1×1	$4 \times H \times W$

4. The Line Extraction Pipeline

In order to make use of our proposed progressive full data convolutional neural networks, we developed a system for line extraction from anime-style illustrations. We had two concerns in our system design: a standardized dataset and user involvement. These concerns were considered in our implementation.

Standardized dataset: Illustrations are industrial or commercial assets, so they cannot be used freely. Under such circumstances, researchers need to create a dataset by themselves or collect data that are already available on the Internet. There is no doubt that search engines with powerful spiders or public photo galleries are the most powerful collecting channels, but the direct use of these data often fails to achieve good results because there is no uniform standard for training data; such as the background of line drawings, where white background and transparent background have two different standards. If both of these are reasonable outputs in the process of extracting line drawings, it is clearly not reasonable in the strict definition. In this paper, we adopted the standard that line drawings must have a transparent background. The pupils of the characters in the illustrations were sometimes completely blacked out, and we chose to ignore this kind of ground truth area. In order to obtain correct outputs, we still needed to clear the colored areas in the pupils while preserving the lines.

User involvement: In art creation, the user's involvement is very important. Usually, neural networks can only react to user-entered training data. Therefore, the neural network needs to learn the stages of the drawing process. First of all, we should do some preliminary painter user study and pay attention to the painter user feedback data to ensure that painter users' knowledge can guide the model training direction. The second is to ensure the tool is easy for use. It is better if users do not need to make any settings at first and can rely on a fully automatic method to generate preliminary results. If the users have more detailed requirements, then they may configure the parameters to obtain semi-automatic output results. The efficiency of training and production are also important. We tried to use simpler structures and smaller resolution training data, and users only needed to provide around 30 training source images to our system to complete the line drawing extraction model training. Finally, we tried to avoid extra pre-processing or post-processing of the data, such as vectorizing the results, which may remove the details together while de-noising.

In order to maintain the balance between training data size and neural network structure complexity, our model adopted a strategy of increasing neural network complexity in exchange for reducing training data size. Our system had three key points: the design of the neural network system, the preparation of training data, and the determination of both subjective and objective evaluation criteria. The pipeline of our system is depicted in Figure 5.

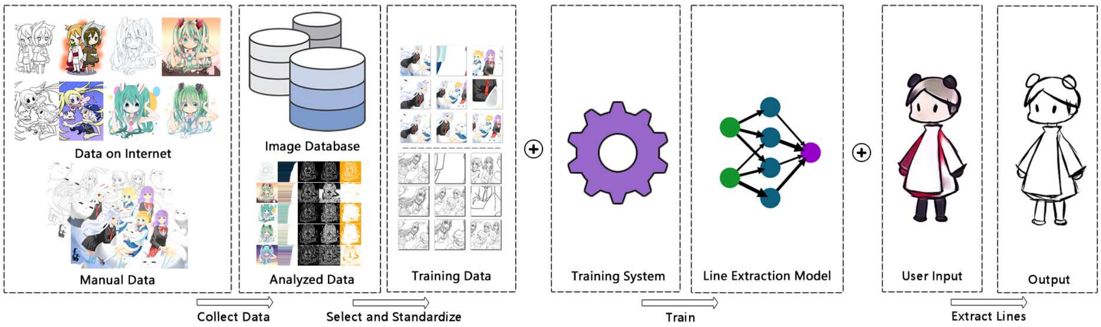


Figure 5. The pipeline of line extraction.

Here, we briefly explain the different blocks and factors of the pipeline as follows:

Data collection: The initial work of the pipeline was data collection. This part of the work is often the least creative, but it is also the most labor intensive. We could collect data through the Internet or manually create a corresponding database for management. Then, we needed to conduct data analysis and select the data that were suitable for training. Using raw data for training would cause noise problems and could not achieve high quality output. After the data analysis, we could standardize the selected data to reduce the output noise and use less training data. Then, we could ensure the uniqueness of the training direction.

The neural network of the training system adopted our proposed progressive full data convolutional neural network. According to different training datasets, we could complete the extraction of line drawings for different types of images. Here, we mainly chose anime-style illustration data for testing, because the patterns of anime-style illustrations are similar. General users could extract lines by using the line extracting model for getting outputs automatically. Usually, users did not need to train the model by themselves. If there were specific requirements, then users could re-select the training dataset and re-train the corresponding model. Users only needed to adjust the output of the existing model and input the adjusted result into the data collector to regenerate the training dataset. Then, an updated model could be generated by re-training.

The training data were collected from the Internet or created by ourselves. It would be better if we could directly obtain the corresponding line drawings from artists. Completed colorized illustrations were used as input data and the corresponding line drawing images as output data. The data were selected manually so the quantity may be not enough. By using noise generation, clipping, and scaling methods, we increased the amount of input images. The training input image size was 128 × 128 pixels.

Patterns' collection: Usually, the size of a source illustration is much larger than the training input image size. Thus, we needed to find the patterns of source illustrations instead of submitting full illustration images to the training system in order to save computational resources. We had 20 pairs of training input pattern images. The quantity of the patterns was still not enough for covering all drawing styles, but it was good enough to extract line drawings from some illustrations. We used pattern images to describe the data features in a simple way, and our convolutional neural network could extend these features. The sample images of the training data generated by using the following guidelines are shown in Figure 6.

1. The first consideration was whether the background was transparent or white. We recommend using a background transparent ground truth, because in some images, the white color may be the line color. Only the alpha value can directly provide a certain value of a transparent background.
2. The second consideration was the thickness of the line. In the current mainstream anime-style illustrations, slight changes in the thickness of the line are worth attention. In the line extraction topic, sometimes a line may be too thick and does not look like a line, but the training data included these situations; thus, the model could satisfy our needs.
3. The third consideration was the background color or the difference in the surrounding color patches. The flat coated color patches were the basic pattern, then the color gradient patches. The celluloid color is also common type. These color blocks were not lines. However, the narrow color blocks were easily detected as lines in the case of insufficient training. Thus, we needed to optimize the selected training data dynamically according to the training results.
4. Finally, we used some basic image transformation methods such as translation, rotation, scaling, etc., to generate more data. These data can also further optimize the training results.

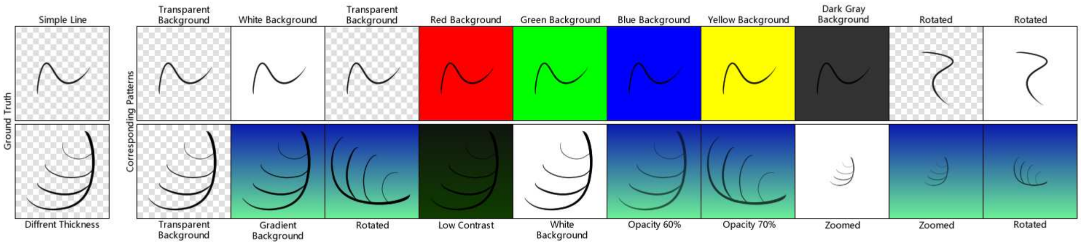


Figure 6. The main source patterns in our training data.

Manual data: Pattern data alone were not enough, and we also needed some manual painting data. These data are usually available from Internet, but the strictness of the content selection determines the final training quality. Currently, there is no automatic screening tool with high quality and high efficiency. Therefore, the screening of data by professionals still has a certain significance. People have to learn some relevant artistic creation knowledge, then complete the data screening work. Simply using the batch download method to establish a training database is not suggested.

Reasonable color format: Because the line drawing extraction process is sensitive to color, we used the RGBAformat for input data. Although the YCbCr format shows the changes in brightness channel intuitively, sometimes it does not mean there is a line, but a border there. The alpha channel is important for further drawing of anime-style illustrations and can help the predictions better.

Small training dataset: The information obtained through web crawling is currently facing legal issues. Some of the neural network training results are based on the results of illegal data collection, and it is easy to retain the common features of the training dataset. Sometimes, a user may only want to learn a specific drawing style, but this cannot be trained by a huge dataset. In this issue, our model used only a small amount of raw data, and the problem of large scale illegal data collection was avoided, while users may just do training with a specific drawing style.

Line weight: The initial experimental results showed that if the bold and thin line samples were used for training at the same time, this would cause difficulty in convergence. The solution only considered the thin line situation, and we could add another network to process bold lines.

Training data weights: At the same time, we also needed to pay attention to the weight distribution of training samples. If the weights of important samples were too low, we would get errors or bad results even if the loss function converged, because potentially unimportant samples dominated the effect of the convergence direction. For example, if the pattern was all black, no line should be extracted. Such a sample was needed, but only a few could be added into the training database. Otherwise, any black area may be ignored, and the neural network chose not to extract any lines.

Content scaling loss: Experimental results showed that the feature data should be plentiful. For example, in a 128×128 pixel image, if only one line was drawn with a thickness of about one to three pixels and the length was only 80 pixels or less, a large amount of this sample data would cause insufficient feature extraction. Instead, this may lead to a training focused on removing the background color. In this situation, the network may output the lines that cannot be seen.

Color contrast: Experimental results showed that dark lines and light colored samples were high contrast samples, and they usually helped obtain better extraction results. Therefore, we needed to add samples with lower contrast and increase the weight of these samples; otherwise, it would be difficult to extract the lines we required in low contrast images. For example, lines and the color of black hair are easily mixed together. Similarly, there are cases where the dark areas are edged with light colored lines. In order to adapt to more situations, we needed to adjust the training data content and weights based on incorrect or unsatisfactory outputs, as many as possible.

Training direction: If there was an error in the training direction, it was better to select the backup file and continue the training from the good model because the correction of the training direction may take a long time or even could not be corrected.

Simplified network: For quick testing, we designed a simplified version of the progressive full data convolutional neural network by appropriately reducing the channel amount of convolutional layers. It could do a good job of rough line extraction, but there were many noises. For practical use, there is still no good way to simplify the standard model provided in this paper.

Batch normalization: The neural network structure proposed in this paper did not use batch normalization because the convergence rate slowed down after using batch normalization, and the results after the reference were not significantly improved.

Without pre-training: The model in this paper was based on new training without any pre-trained models. Even if we started training from scratch, we found that we could usually get preliminary results in a few hours. With the current computational power on a typical PC, it is usually possible to train a neural network model with the most appropriate loss and output effects in about three days. If the training data were changed and retrained, unexpected situations would often occur.

Batch size: The batch training size value was fixed at 16 in our test. Too small batch training size values, such as one, could lead to problems such that the loss of model could not converge because each training direction could not be fixed. Excessive values could lead to memory overflow because our model was complex, and the progressive whole-data convolution process was also performed at the level of the original image size, so the memory consumption was very large. To sum up, we obtained a value of 16 based on trial-and-error.

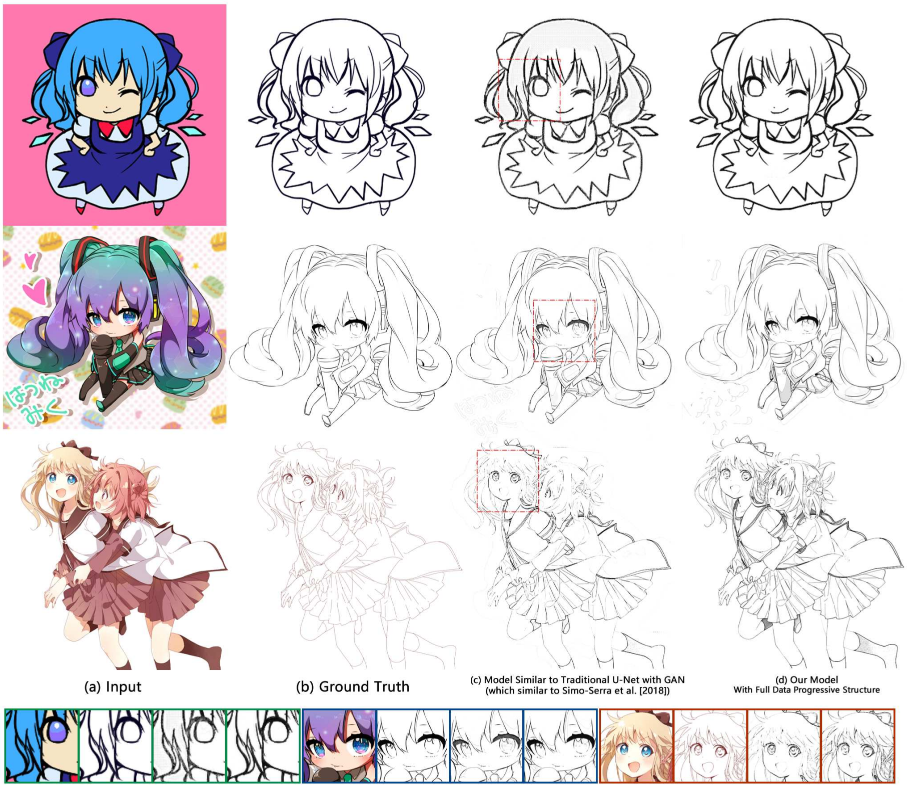
Loss function: We use the mean squared error (MSE) to calculate the loss of our model. The following equation expressed the loss function:

$$\mathfrak{L} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2}$$

5. Experimental Results

We implemented our network with Chainer, and we trained and tested our model on a machine with Intel E3 1230 v3 CPU and Nvidia GTX 1080 Ti GPU. There were finally around 10 pairs of source images and 10,000 pairs of input images generated from the source images by clipping and scaling operations. The training step took about 93 h for 1,000,000 iterations with a batch size of 16. The database size was 10,000; thus, there were about 1600 epochs.

The results in Figure 7 show the advantages of our model. The traditional model similar to U-Net with GAN was confused about how to handle the lines, so the lines were not as clear as ours. The image in Column (c) in the second line shows that the background removal was not completed, so it had more noise in the background than the image in Column (d), which was generated by our model.


(a) Input

(b) Ground Truth

(c) Model Similar to Traditional U-Net with GAN (which similar to Simo-Serra et al. [2018])

(d) Our Model With Full Data Progressive Structure

Figure 7. Comparisons of experimental results: Images of Column (**a**) are input and images of Column (**b**) are ground truth; Images of Column (**c**) and images of Column (**d**) were generated by different models; both of these models' loss values were converged for rough comparison, and the same parameters and training database were used. We can see clearly that our model obtained better results than those of the models based on the traditional U-Net structure.

From the results in Figure 8, we can see that the outputs in the control group had more faults than the outputs generated by our model. Our lines were clear.er Our model and the models we compared had the same training database, but had different understanding of the lines. The results showed that our model correctly removed the dark area and reserved the lines around it. The model recognized by itself that we only needed the lines, but the model without a progressive structure may regard the black as some kind of very bold lines.

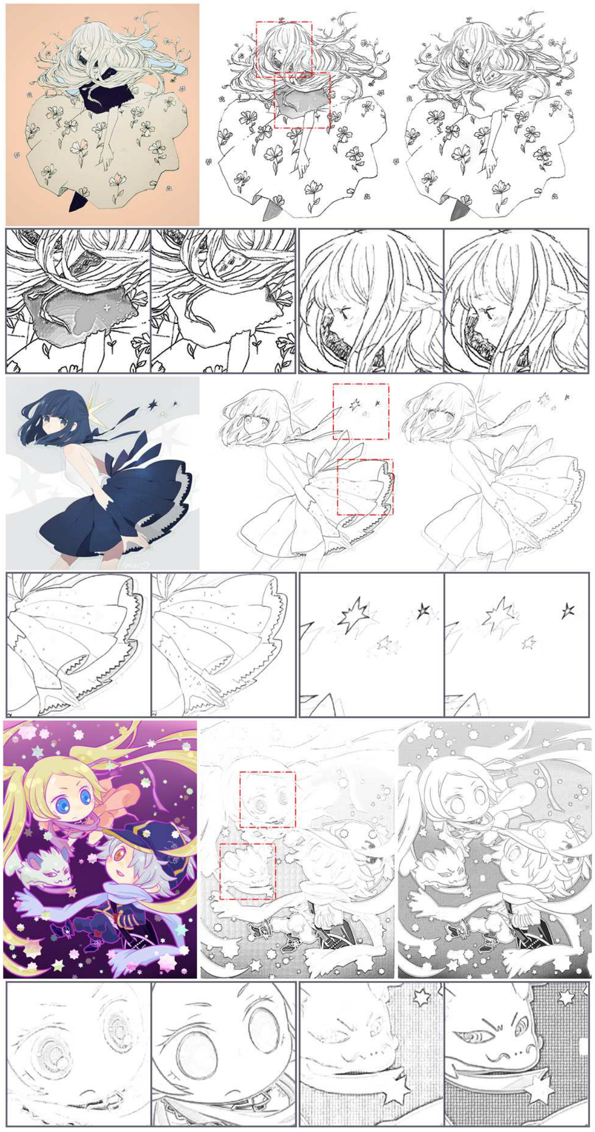


Figure 8. Experimental results: the original images are shown in the first column, the images generated by a model similar to U-Net with GAN (Simo-Serra et al. [9]) are shown in the second column, and the images generated by our model are shown in the third column. Details are highlighted and compared in the two small images below the results, and the right one of the two was generated by our model.

There are more results in Figure 9. These outputs showed that most of the time, our model worked well. The traditional models also reached the goal of line extraction, but the results were not good enough. There were mainly three points by which our model worked better than the traditional U-Net models:

1. Clearer lines.
2. Prediction based on pixel level and global level views.
3. Better background removal.

Based on the feedback from users, we obtained the following evaluation. Our system had a greater precision than those of other recent systems. The basic reason was that the system that integrated with our model retained the details of the data as much as possible, while other systems preferred fuzzy processing. The fully automatic mode was more convenient for users who did not understand how a neural network works. The cumbersome training data preparation and the working principle of the neural network are difficult for general users to understand, so it is not yet possible to achieve their needs through simple steps. For example, professionals have great interest in whether the system can be used to train high precision line corrections.

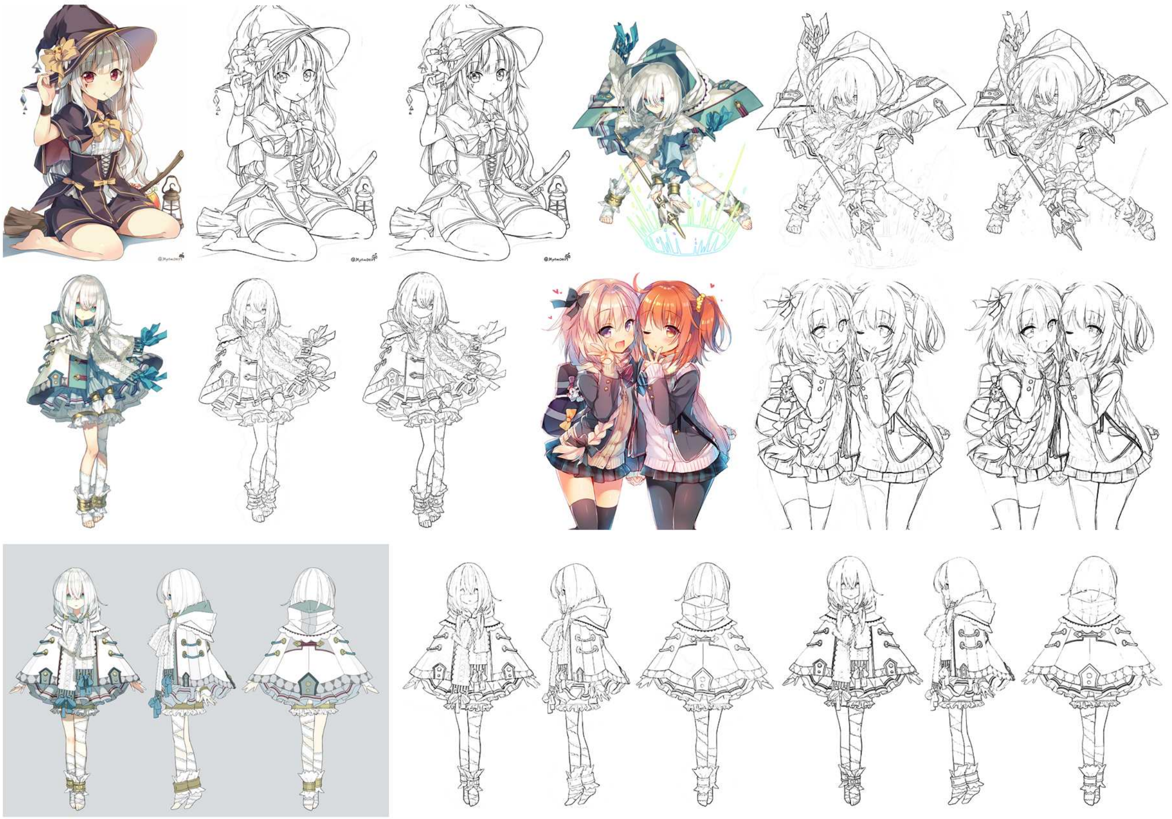


Figure 9. More experimental results: There are five groups of images. The first image of each group is the original image; the second one is generated by the model similar to U-Net with GAN (Simo-Serra et al. [9]); and the third one is generated by our model.

6. Application

Through the preliminary user study, we believe that the current system has two practical applications. One is to extract line drawings of the landscape according to the style of manga drawing, and the other is to extract the line drawings from the existing illustrations and provide them to the students who want to learn coloring.

Figure 10 shows the results of extracting line drawings from real landscape photos. The details of buildings and trees are completely preserved. It can directly replace traditional photo filters and even eliminate the requirement of manually setup parameters.

Because the comic line drawing style ignores textures and light effects on human and animal skin, the vision effect of painting did not look good. It is appropriate to learn the pencil sketch painting style for human or animals, but this is beyond the scope of this paper. In addition, a more important application is to use our system to provide analytical data for subsequent drawing simulations. We can also try to extract the basic color layer and add light, shadow, or texture blending effects to the basic color layer.

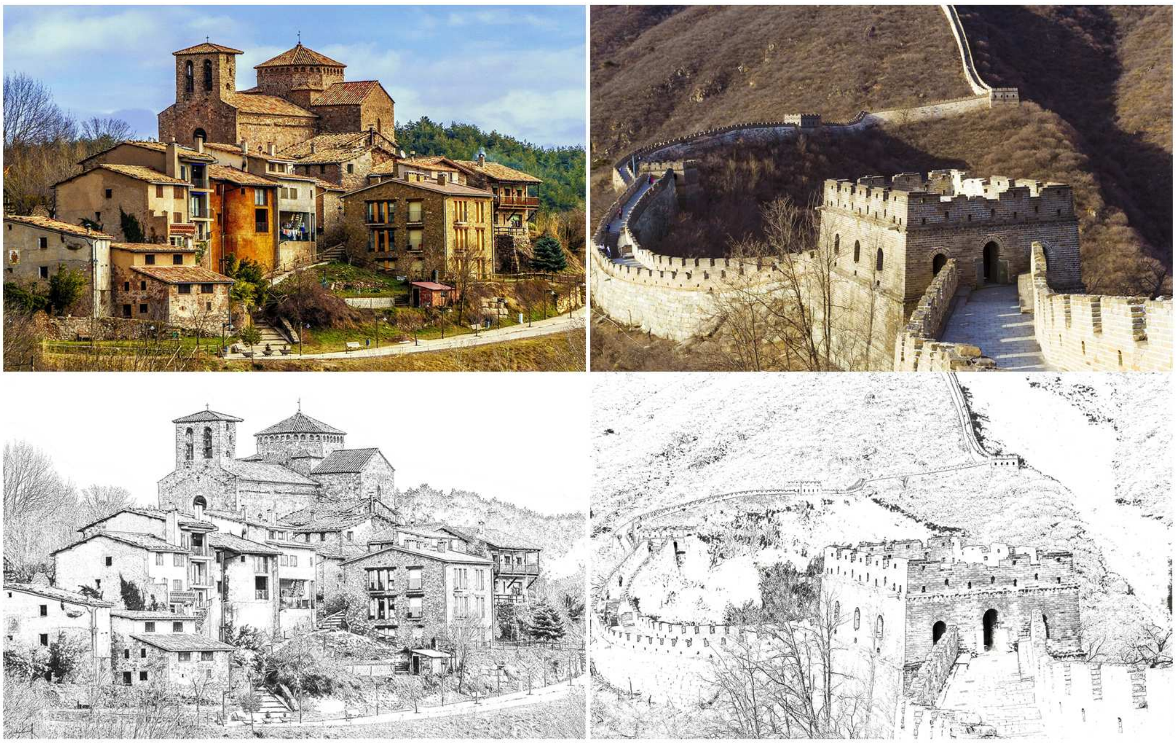


Figure 10. Application: extracting lines from realistic photos.

7. Conclusions and Future Work

In this paper, we proposed progressive full data convolutional neural networks specific for extracting lines from anime-style illustrations. In addition, we also implemented a system that integrated our proposed networks. Experimental results showed that our model could more accurately extract lines from anime-style comic illustrations compared to those obtained with the latest neural network structures. Besides, our results were closer to the human perspective. We also showed the potential of applying our model for non-photorealistic rendering applications. We also suggested that data standardization should be considered for the reduction of training samples and the improvement of output accuracy.

Regarding the limitations of our model, our model wasted some computational memory space. In fact, in order to extract lines, it did not need RGBA channels, and we may only need the alpha or intensity channel. In addition, if the training data range could exceed the range of integers from 0–255, it could provide more training information. Of course, the accuracy may be increased if we could train the neural network with 256 × 256 resolution inputs or add $L5$ layers. However, based on our current equipment's performance, we already optimized the model for 128 × 128 resolution inputs and the deepest $L4$ layers.

Based on our experiments, we recognized that the network should be designed as a targeted network structure. If the complexity of a neural network structure is limited, a network needs to be divided into several neural networks to process the problems separately. At the same time, a high quality training database should be prepared. The network was not suitable for real-time applications because of the long processing time required. In the future, we expect to improve the networks for real-time applications, so artists can have semi-automatic interactions to get involved in the drawing simulation process.

Author Contributions: Y.X. contributed the idea and methodology and performed the experiments; Y.X., H.-C.W., S.-L.L., and J.L. wrote the original draft; H.-C.W. supervised the project and revised the paper. All authors read and agreed to the published version of the manuscript.

Funding: This work was supported by the Science and Technology Development Fund of Macao SAR(0027/2018/A1 and 0016/2019/A1) and the Faculty Research Grant of Macau University of Science and Technology.

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

References

1. Canny, J. A Computational Approach to Edge Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1986**, *PAMI-8*, 679–698. [CrossRef]
2. Kang, H.; Lee, S.; Chui, C.K. Coherent Line Drawing. In Proceedings of the 5th International Symposium on Non-photorealistic Animation and Rendering, NPAR '07, San Diego, CA, USA, 4–5 August 2007; ACM: New York, NY, USA, 2007; pp. 43–50.
3. Tomasi, C.; Manduchi, R. Bilateral Filtering for Gray and Color Images. In Proceedings of the Sixth International Conference on Computer Vision, ICCV '98, Bombay, India, 4–7 January 1998; IEEE Computer Society: Washington, DC, USA, 1998; p. 839.
4. Vese, L.A.; Osher, S.J. Modeling Textures with Total Variation Minimization and Oscillating Patterns in Image Processing. *J. Sci. Comput.* **2003**, *19*, 553–572. [CrossRef]
5. Xu, L.; Yan, Q.; Xia, Y.; Jia, J. Structure Extraction from Texture via Relative Total Variation. *ACM Trans. Graph.* **2012**, *31*, 139. [CrossRef]
6. Ito, K.; Matsui, Y.; Yamasaki, T.; Aizawa, K. Separation of Manga Line Drawings and Screentones. *Eurographics 2015 Short Paper*, 2015.
7. Kopf, J.; Lischinski, D. Digital Reconstruction of Halftoned Color Comics. *ACM Trans. Graph.* **2012**, *31*, 140. [CrossRef]
8. Yao, C.Y.; Hung, S.H.; Li, G.W.; Chen, I.Y.; Adhitya, R.; Lai, Y.C. Manga Vectorization and Manipulation with Procedural Simple Screentone. *IEEE Trans. Vis. Comput. Graph.* **2017**, *23*, 1070–1084. [CrossRef]
9. Simo-Serra, E.; Iizuka, S.; Ishikawa, H. Mastering Sketching: Adversarial Augmentation for Structured Prediction. *ACM Trans. Graph.* **2018**, *37*, 11. [CrossRef]
10. Mao, X.; Liu, X.; Wong, T.T.; Xu, X. Region-based structure line detection for cartoons. *Comput. Vis. Media* **2015**, *1*, 69–78. [CrossRef]
11. Li, C.; Liu, X.; Wong, T.T. Deep Extraction of Manga Structural Lines. *ACM Trans. Graph. (TOG)* **2017**, *36*, 117. [CrossRef]
12. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015*; Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 234–241.
13. Liu, X.; Wong, T.T.; Heng, P.A. Closure-aware Sketch Simplification. *ACM Trans. Graph.* **2015**, *34*, 168. [CrossRef]
14. Simo-Serra, E.; Iizuka, S.; Sasaki, K.; Ishikawa, H. Learning to Simplify: Fully Convolutional Networks for Rough Sketch Cleanup. *ACM Trans. Graph.* **2016**, *35*, 121. [CrossRef]
15. Shelhamer, E.; Long, J.; Darrell, T. Fully Convolutional Networks for Semantic Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 640–651. [CrossRef] [PubMed]
16. Iizuka, S.; Simo-Serra, E.; Ishikawa, H. Let There Be Color!: Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification. *ACM Trans. Graph.* **2016**, *35*, 110. [CrossRef]
17. Zhang, R.; Zhu, J.Y.; Isola, P.; Geng, X.; Lin, A.S.; Yu, T.; Efros, A.A. Real-time User-guided Image Colorization with Learned Deep Priors. *ACM Trans. Graph.* **2017**, *36*, 119. [CrossRef]
18. Han, C.; Wen, Q.; He, S.; Zhu, Q.; Tan, Y.; Han, G.; Wong, T.T. Deep Unsupervised Pixelization. *ACM Trans. Graph.* **2018**, *37*, 243. [CrossRef]
19. Cao, Y.; Pang, X.; Chan, A.B.; Lau, R.W.H. Dynamic Manga: Animating Still Manga via Camera Movement. *IEEE Trans. Multimed.* **2017**, *19*, 160–172. [CrossRef]
20. Cao, Y.; Chan, A.B.; Lau, R.W.H. Automatic Stylistic Manga Layout. *ACM Trans. Graph.* **2012**, *31*, 141. [CrossRef]

21. Cao, Y.; Lau, R.W.H.; Chan, A.B. Look over Here: Attention-directing Composition of Manga Elements. *ACM Trans. Graph.* **2014**, *33*, 94. [CrossRef]
22. Pang, X.; Cao, Y.; Lau, R.W.; Chan, A.B. A Robust Panel Extraction Method for Manga. In Proceedings of the 22Nd ACM International Conference on Multimedia, MM '14, Orlando, FL, USA, 3–7 November 2014; ACM: New York, NY, USA, 2014; pp. 1125–1128.
23. Liu, X.; Li, C.; Zhu, H.; Wong, T.T.; Xu, X. Text-aware balloon extraction from manga. *Vis. Comput.* **2016**, *32*, 501–511. [CrossRef]
24. Qu, Y.; Wong, T.T.; Heng, P.A. Manga Colorization. *ACM Trans. Graph.* **2006**, *25*, 1214–1220. [CrossRef]
25. Qu, Y.; Pang, W.M.; Wong, T.T.; Heng, P.A. Richness-preserving Manga Screening. *ACM Trans. Graph.* **2008**, *27*, 155:1–155:8. [CrossRef]
26. 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 2018 ACM Multimedia Conference, Seoul, Korea, 22–26 October 2018; pp. 1536–1544.
27. Hensman, P.; Aizawa, K. cGAN-based Manga Colorization Using a Single Training Image. *arXiv* **2017**, arXiv:1706.06918.
28. Zhu, H.; Liu, X.; Wong, T.T.; Heng, P.A. Globally Optimal Toon Tracking. *ACM Trans. Graph.* **2016**, *35*, 75. [CrossRef]
29. Liu, X.; Mao, X.; Yang, X.; Zhang, L.; Wong, T.T. Stereoscopizing Cel Animations. *ACM Trans. Graph.* **2013**, *32*, 223. [CrossRef]
30. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
31. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, 27–30 June 2016.
32. Goodfellow, I.; 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 27*; Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q., Eds.; Curran Associates, Inc.: Dutchess County, NY, USA, 2014; pp. 2672–2680.
33. Zeiler, M.D.; Fergus, R. Visualizing and Understanding Convolutional Networks. In Proceedings of the Computer Vision—ECCV 2014, Zurich, Switzerland, 6–12 September 2014; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 818–833.