
ASCII Art Synthesis with Convolutional Networks

Osamu Akiyama
Faculty of Medicine, Osaka University
oakiyama1986@gmail.com

1 Introduction

ASCII art is a type of graphic art that presents a picture with printable characters. It is commonly used for graphical presentations in text-based media. ASCII art can be categorized into two major styles: one is tone based, and the other is structure based. Tone-based ASCII art represents the intensity distribution of the original images by using the density of the characters. In contrast, structure-based ASCII art represents line structures of the original images using the directions of the lines in the characters. Thus, it is generally accepted that creating structure-based ASCII art is more demanding and difficult than creating tone-based ASCII art. There have been several attempts to automatically synthesize structure-based ASCII art [1–3], but there remains room for improvement in the quality of their products compared to ASCII art created manually by artists. To improve the quality of automatic synthesis, we develop a convolutional network (CNN) model [4] by training the network with ASCII art created manually by artists.

2 Materials and Methods

We collected ASCII art from the Japanese Bulletin Board Systems (BBS) “5channel” (formerly known as “2channel”) and Shitaraba BBS. The ASCII art consisted of a Japanese Shift-JIS character set, 12-pt MS PGothic (16 pixels in height) characters with 2-pixel blank spaces. We manually selected 500 structure-based ASCII art pieces with clear lines for training and validation (e.g. Fig. 1, left), each of which subtended 124-1056 pixels in width and 90-954 pixels in height and included 69-4309 characters. The dataset consisted of 899 character types. Usually, ASCII art is published without citing the original images. Thus, we estimated their originals by using a CNN that had been developed for the cleanup of rough sketches [5]. The network removed unnecessary lines and partially recovered boldness, continuity, and smoothness of lines that had been lost in the process of the ASCII art creation (Fig. 1, right).

We defined our task as selecting the true character for each local area in the images. We prepared a 10-layer deep CNN that converted a local square image (64×64 pixels) into a single character in the center (e.g. squares in Fig. 1). Because a single character subtended 16 pixels in width and 3-16 pixels in height, the input image (64×64) provided rich contexts surrounding the central region of interest. Of the 899 character types, we used 411 character types that appeared more than 10 times in the dataset for training. Overall, 90% of the dataset was used for training, and the remainder was used for validation. For augmentation, we performed vertical and horizontal shifts of the images in the range of 1 pixel and added Gaussian noise to the images. After training, the model was used to synthesize ASCII art from the original line images (e.g. Fig. 2, left) by sliding the 64×64 pixel window in a sequential manner (from left to right and then top to bottom). All codes and sample results are available at: <https://github.com/OsciiArt/DeepAA>

3 Results and Discussion

After training for 49 epochs with a batch size of 64, the network yielded a correct estimation rate of 89.0%. An example of ASCII art synthesis is shown in Fig. 2. Regarding the quality of the art,

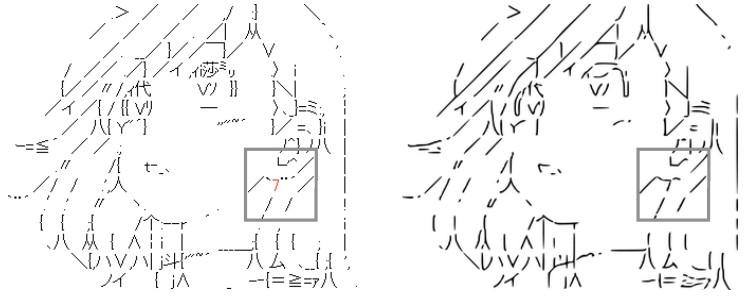


Figure 1: Datasets for training. Left: one example of a hand-made ASCII art piece. Right: an estimate of the original picture. Gray squares show 64×64 pixel size, and its true label is “7”.

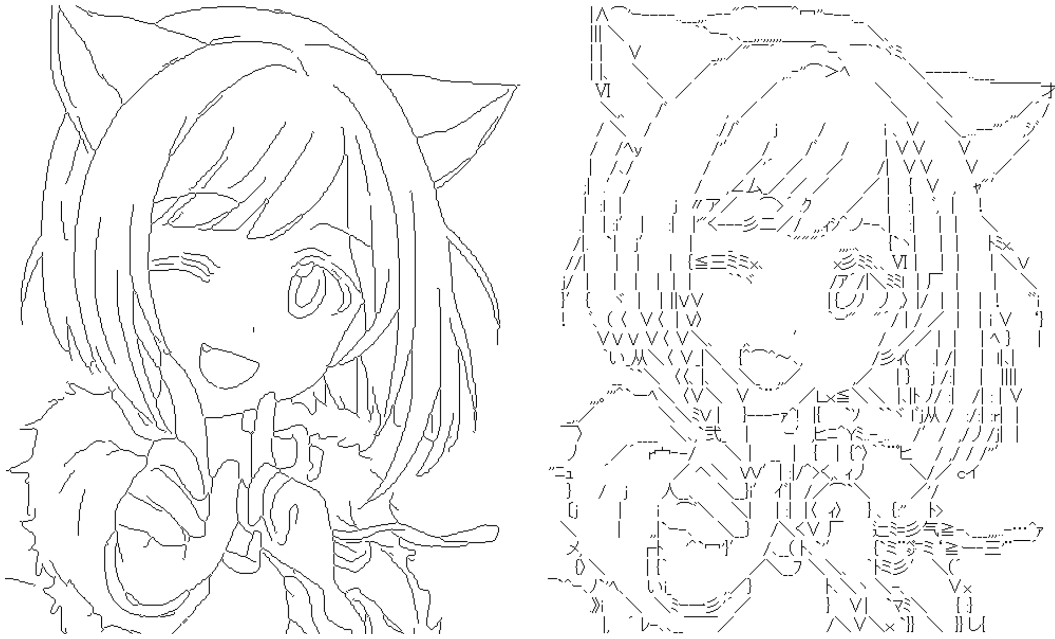


Figure 2: An example of input line images (left) and the synthesized ASCII art (right).

we feel that the synthesized ASCII art looks comparable to those made by artists (e.g. Fig. 1, left). We compared the performance of our model with the publicly available ASCII art synthesis tools, (` Π `) Auto [2] and AsciiArtConverter [3], and artist works using image similarity metrics, structural similarity (SSIM) [6] and Illustration2Vec [7]. Our method achieved the best scores in the I2V metric and the differences were statistically significant (two-sided Wilcoxon signed-rank test, $p < 0.05$; $n = 20$). Since I2V measures semantic distances between illustrations, we may suggest that our method adequately represents the semantic information of the original images. However, evaluation by similarity metrics is not ideal because ASCII artists often reform original images to improve the final impressions. Indeed, ASCII art created by artists is often less similar to the original images than that made by automatic algorithms [1]. Accordingly, we may have to ask human raters to evaluate the quality of the art in the future.

Despite this limitation, we would like to describe our contribution as threefold. First, this is the first approach for ASCII art synthesis that uses human-made ASCII art for training. Second, we proposed an efficient method for preparing an “original” image to be paired with each ASCII art. Finally, we were able to realize a CNN model suitable for ASCII art synthesis.

Acknowledgments

The author would like to thank Prof. Shigeru Kitazawa for his support and helpful comments. This work is supported by the Osaka University Medical Doctor Scientist Training Program.

References

- [1] X. Xu, L. Zhang, and T. Wong. Structure-based ASCII art. *ACM Trans. Graph.*, 29(4):52:1–52:9, 2010.
- [2] Kuronowish. (´ ɀ `) Auto. http://sky.geocities.jp/edit_hinanzyo/, 2004. Accessed: 2017-11-3.
- [3] UryuP. AsciiArtConverter. <https://onedrive.live.com/?authkey=%21APuDfORKnM1d3YE&id=2A550F2DD4D15CFC%21116&cid=2A550F2DD4D15CFC>, 2010. Accessed: 2017-11-3.
- [4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.
- [5] E. Simo-Serra, S. Iizuka, K. Sasaki, and H. Ishikawa. Learning to simplify: Fully convolutional networks for rough sketch cleanup. *ACM Trans. Graph.*, 35(4):121:1–121:11, 2016.
- [6] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612, 2004.
- [7] M. Saito and Y. Matsui. Illustration2Vec: A semantic vector representation of illustrations. In *SIGGRAPH Asia 2015 Technical Briefs*, 2015.

Supplementary materials for: ASCII Art Synthesis with Convolutional Networks

Osamu Akiyama

Faculty of Medicine, Osaka University
oakiyama1986@gmail.com

1 Network Architecture and Training Details

The network architecture of our model is mainly inspired by the VGG network [1]. Since the input size is smaller ($64 \times 64 \times 1$, grayscale) than the original VGG network ($224 \times 224 \times 3$, RGB) and important features are localized at the center of the input image, the number of resolution-decreasing processes by max-pooling is reduced. The network is composed of seven convolutional layers, three max-pooling layers, two fully connected (FC) layers and an output layer. We use convolutional layers with 3×3 kernels followed by batch-normalization layers, and ReLU as the activation function. We use max-pooling layers with 2×2 pooling windows. FC layers have 4096 neurons followed by batch-normalization, ReLU and dropout with the corrupting probability of 0.5. The output layer has 411 neurons representing each character label, which are followed by final softmax activation. The summary of the whole network is C64-C64-P-C128-C128-P-C256-C256-C256-P-FC4096-FC4096-O411, where Ck denotes the convolutional layer with k filters, P denotes the max-pooling layers, FCk denotes the FC layers with k neurons, and Ok denotes the output layer with k neurons.

The model was implemented in Keras [2] with TensorFlow [3] backend. The networks learned to minimize the cross-entropy loss function over the training data. Overall, 90% of the dataset was used for training, and the remainder was used for validation. For the estimated images were greatly changed with respect to their native ASCII art images, we used not only the estimated images but also the native ASCII art images for preparing the input images to achieve efficient training. For validation, we used only the estimated images. The input images were normalized in the 0-1 range before use. For data augmentation, we performed vertical and horizontal shifts of the images in the range of 1 pixel and added Gaussian noise with a mean of 0 and standard deviation of 0.3 to the images. Weights of convolutional layers were initialized from a Gaussian distribution with a mean of 0 and a standard deviation of 0.05. Weights of FC layers were initialized by the Glorot uniform initializer [4]. As an optimizer, we used the stochastic gradient descent with a learning rate of 0.01, and reduced the learning rate by a factor of 10 when the validation loss stopped improving for 8 epochs. We trained our model on a NVIDIA Tesla P-100 GPU with a batch size of 64. We stopped training when the validation loss did not improve for 16 epochs. We saved the model at 49 epochs, which was evaluated as the best based on the validation data during the learning process.

2 Comparison to Other Methods

There are some studies regarding structure-based ASCII art synthesis. In contrast to creating tone-based ASCII art, creating structure-based ASCII art is a difficult task because it includes higher-order cognitive processes, such as detecting structure lines from images, selecting perceptually important structures, and reconstructing structures into a character-suitable form. Fundamentally, ASCII art is synthesized by evaluating the similarities between a local image and character images using some image similarity metrics and then selecting the most similar character. For better character matching, the metrics need to tolerate deformation of images. Therefore, metrics comparing images in a pixel-by-pixel manner, such as the mean-squared error, are not appropriate because they are sensitive to local deformation. Xu et al. (2010) [5] proposed the first metric designed for character matching,

which tolerates misalignment and local deformation. Miyake et al. [6] employed the histograms of oriented gradients (HOG), which is designed for image quality assessment and tolerates small local deformations. Xu et al. (2017) [7] used multi-orientation phase congruency obtained by an extended Gabor filter and achieved state-of-the-art quality.

Using a proportional font (in which each character has a different width), ASCII art obtains more representation ability by adjusting the location of characters. However, this makes ASCII art synthesis more complicated because of its tendency to get trapped in local minima. To avoid local minima, Xu et al. (2015) [8] employed a dynamic programming approach for character placement optimization.

There also exist some tools for structure-based ASCII art synthesis published in the internet. Ascii Art Editor (AAE) [9] generates ASCII art by evaluating image similarities in a pixel-by-pixel manner. (` Π `) Auto (AAAUTO) [10] evaluates similarities using the distance between black pixels, which gives it some tolerance of deformation. AsciiArtConverter (AAConverter) [11] is a tool that has a line-drawing extraction function and generates ASCII art by evaluating image similarities based on phase congruency, which achieves favorable qualities.

We compared the ASCII art generated by AAAuto, AAConverter, our method and artist work (Fig. 1). All of them were created in the same settings with 12-pt MS PGothic and 2-pixel blank spaces and in the same size with respect to each image. To generate line images from drawings or photographs, we extracted contours by hand from the images and then processed them with AAConverter. It extracts thin lines from images by the Laplacian filter and skeletonization. It also removes excessively short lines as noise and connects gaps between lines. Then, processed line images were used as inputs for each method. We can see that our method generated ASCII art that is more similar to the artist’s, though it is still poor in aesthetic aspects.

We evaluated the similarity between the original images and the generated ASCII art images. We used structural similarity (SSIM) [12] and Illustration2Vec (I2V) [13] for image similarity metrics. I2V is a semantic vectorization of illustrations using features of a CNN trained with the task of tag estimation of illustrations. Using I2V semantic vectors, we calculated semantic distances between the original images and the ASCII art. We prepared 20 images for testing and evaluated the ASCII art generated by AAAuto, AAConverter, our methods and artists. We used the line images generated by the process described above as original images (e.g. Fig. 1 a). Average scores of each metric are shown in table 1. Our methods achieved the best score with the I2V metric and the differences were statistically significant (two-sided Wilcoxon signed-rank test, $p < 0.05$). So we may suggest that our method adequately represents the semantic information of the original images. This also indicates that these similarity metrics are insufficient for use as evaluations because the scores of the artists were also poor. Accordingly, a user study needs to be performed for a quantitative evaluation based on human perceptions such as similarity, recognition ability and aesthetics [8].

Table 1: Comparison of the existing methods by similarity metrics. The best measures in bold.

	AAAUTO	AAConverter	our method	Artist
SSIM	0.7319	0.6898	0.7168	0.5841
I2V	84.65	68.14	61.88	79.41

3 Limitations and Future Plan

As mentioned above, the main limitation of our study is the lack of quantitative evaluation. There were also a few other limitations: First, since our model is trained by data with a specific font type and size, it cannot change font type or size like other methods using similarity metrics. Second, since the placement of the characters is fixed in our task, our model may not have learned the ability to place characters in the best place. Since our model estimates characters without evaluating their similarity, it is difficult to optimize the character placement using similarity metrics. However, we can see that our model selects 2-byte space or 1-byte space as blank areas, according to images surrounding the target area, to adjust character placement. Third, our model didn’t learn enough to reform original images to improve the final impressions as artists do. Artists reform the structures of the original images using their knowledges and experiences. But, our estimated images used for training are still too similar to the native ASCII art images. To learn such a reformation ability, we may have to collect the real original images or generate more realistic estimated images.



Figure 1: Comparison with the existing ASCII art generation methods and the artist's creation. (a) Input (b) AAAuto (c) AAConverter (d) Our method (e) Artist

References

- [1] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICPR*, 2015.
- [2] F. Chollet et al. Keras. <https://github.com/fchollet/keras>, 2015.
- [3] M. Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems. <https://www.tensorflow.org/>, 2015.
- [4] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proc. AISTATS*, 2010.
- [5] X. Xu, L. Zhang, and T. Wong. Structure-based ASCII art. *ACM Trans. Graph.*, 29(4):52:1–52:9, 2010.
- [6] K. Miyake, H. Johan, and T. Nishita. An interactive system for structure-based ASCII art creation. In *Proc. NICOGRAPH Int.*, 2011.
- [7] X. Xu, L. Zhong, M. Xie, X. Liu, J. Qin, and T. Wong. ASCII art synthesis from natural photographs. *IEEE Trans. Vis. Comput. Graph.*, 23(8):1910–1923, 2017.
- [8] X. Xu, L. Zhong, M. Xie, J. Qin, Y. Chen, Q. Jin, T. Wong, and G. Han. Texture-aware ASCII art synthesis with proportional fonts. In *Proc. 5th Int. Symp. Non-Photorealistic Animation Rendering*, 2015.
- [9] geroimo. Ascii Art Editor. <http://www22.atpages.jp/monahokan/AAE/>, 2001. Accessed: 2017-11-3.
- [10] Kuronowish. (` ㏄ `) Auto. http://sky.geocities.jp/edit_hinanzyo/, 2004. Accessed: 2017-11-3.
- [11] UryuP. AsciiArtConverter. <https://onedrive.live.com/?authkey=%21APuDfORKnM1d3YE&id=2A550F2DD4D15CFC%21116&cid=2A550F2DD4D15CFC>, 2010. Accessed: 2017-11-3.
- [12] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612, 2004.
- [13] M. Saito and Y. Matsui. Illustration2Vec: A semantic vector representation of illustrations. In *SIGGRAPH Asia 2015 Technical Briefs*, 2015.