# An Automatic System for Generating Artificial Fake Character Images

Yisheng Yue[1], Palaiahnakote Shivakumara[2], Yirui Wu[1,3], Liping Zhu[4], Tong Lu[1(✉)], and Umapada Pal[5]

[1] National Key Lab for Novel Software Technology, Nanjing University, Nanjing, China
abelyys@foxmail.com, wuyirui@hhu.edu.cn, lutong@nju.edu.cn
[2] Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia
shiva@um.edu.my
[3] College of Computer and Information, Hohai University, Nanjing, China
[4] School of Information Management, Nanjing University, Nanjing, China
chemzlp@163.com
[5] Computer Vision and Pattern Recognition Unit, Indian Statistical Institute, Kolkata, India
umapada@isical.ac.in

**Abstract.** Due to the introduction of deep learning for text detection and recognition in natural scenes, and the increase in detecting fake images in crime applications, automatically generating fake character images has now received greater attentions. This paper presents a new system named Fake Character GAN (FCGAN). It has the ability to generate fake and artificial scene characters that have similar shapes and colors with the existing ones. The proposed method first extracts shapes and colors of character images. Then, it constructs the FCGAN, which consists of a series of convolution, residual and transposed convolution blocks. The extracted features are then fed to the FCGAN to generate fake characters and verify the quality of the generated characters simultaneously. The proposed system chooses characters from the benchmark ICDAR 2015 dataset for training, and further validated by conducting text detection and recognition experiments on input and generated fake images to show its effectiveness.

**Keywords:** Fake characters · Generative adversarial network
Shape information · Character editing

## 1 Introduction

It is noted that day by day crimes are increasing in all the departments including tampering texts in natural scene images. In order to find solutions to this problem, researchers have started developing systems and methods for forgery detection [1] recently. The main issue these methods met is dataset creation, which requires a large number of real images for experimentation. In addition, since such real data are often


I. Kompatsiaris et al. (Eds.): MMM 2019, LNCS 11296, pp. 291–301, 2019.
https://doi.org/10.1007/978-3-030-05716-9_24

sensitive, forensic teams do not share the information for experimentation and developing methods. In the same way, we notice that deep learning based methods [2, 3] use synthetic character images and artificially created images for feature extraction, classification and recognition in the recent days. For creating such a huge dataset for training, it always requires a large amount of time to create and label the samples. Therefore, in order to cope with the above-mentioned challenges, there is an urgent need for developing a novel system which can generate any number of fake character images such that an accurate and robust method can be developed and validated without involving much manpower and cumbersome tasks. At the same time, deep learning based methods can use the same system for generating any number of samples that match with input images. This is the main advantage of the proposed system.

Examples of fake character image generation are shown in Fig. 1, where the left image refers to the original image, while the right gives the result of the proposed system by replacing the characters in the red rectangular with the fake characters in the blue rectangular in each same image. It can be seen that the generated characters appear the same or very similar as the characters in the input images. The interesting point here is that the proposed system is faster than we creating a fake character similar to real character by hand. To achieve the above target, we propose to explore Generative Adversarial Network (GAN) [4], which is designed with a generator and a discriminator to provide a simple but powerful way to estimate target distributions based on the input distribution. Researchers thus utilize GAN to generate image samples based on the inputs [5, 6]. In fact, GAN has different types of architectures due to different designs of loss functions, such as WGAN [7] and Least-squares GAN [8]. Here we train our GAN network using least-square loss and adopt the L1 reconstruction loss as the reconstruct loss. The proposed system feeds shapes and background color features to GAN to guarantee the quality of generated fake characters.

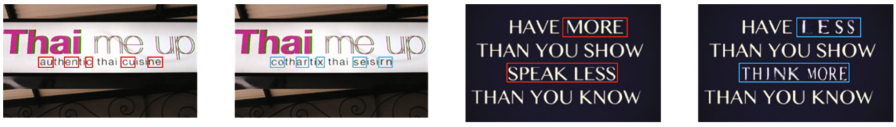


**Fig. 1.** The results of automatically generated faked images, where (a), (c) are original images and (b), (d) are generated fake images. Note that several characters in (b) and (d) have been changed using our proposed system by an automatic way (red: the original characters; blue: fake characters). (Color figure online)

The main contribution of the proposed system is proposing an end-to-end generation system, Fake Character GAN (FCGAN) that supports automatic editing of characters inside natural scenes. Unlike style transfer methods that only involve transforms at pixel level, the proposed system considers shape information and translates characters on a higher level. Additionally, feeding shapes and background color information to GAN and using GAN for fake character image generation is new in the field of text detection and recognition in natural scene.

## 2 Related Work

As one of the most significant improvements on the research of deep generative models, GAN [4] has drawn a quantity of attentions from both deep learning and pattern recognition communities. It has been widely used in image generation [9], image editing [10], and representation learning [11]. The key idea of GAN stems from the two-player game designed by GAN, i.e., a generator and a discriminator that provide a powerful way to estimate target distributions and generate novel image samples. With this power for distribution modeling, GAN is suitable for unsupervised tasks. By combining traditional content loss and adversarial loss, super-resolution generative adversarial networks [12] achieve state-of-the-art performance for image super-resolution. In addition, for unsupervised learning tasks, GANs also show impressive potential for semi-supervised learning. For example, Salimans et al. [13] propose a GAN-based framework, where the discriminator not only outputs the probability to define whether an input image is extracted from real data, but also computes the probability of belonging to each class.

It is noted from Conotter et al. [14] and Abramova [15] that the methods used generated data to improve the performance of forgery detection. Inspired by this idea, relevant to the proposed system, we propose the GAN to perform image-to-image translation. For example, Isola et al. [16] use a conditional generative adversarial network to learn the mapping from input to output images. Reed et al. [17] propose a model to synthesize images given text descriptions based on conditional GANs. More recently, CoGAN [18] uses a weight-sharing strategy to learn a common representation across domains. Furthermore, Liu et al. [19] extend this framework with the combination of variational autoencoders and generative adversarial networks. These successful applications of GAN motivate us to develop a new fake character generation system based on GAN.

## 3 Proposed FCGAN

In this paper, we propose an end-to-end system FCGAN to automatically generate fake scene characters. We show the network architecture of the proposed system as in Fig. 2. The input of the generator consists of three kinds of images, which represent input characters, shapes of input characters, and styles of target characters, respectively. The proposed system can generate different types of characters with different shape information.

### 3.1 Network of FCGAN

The proposed FCGAN consists of a generator network and a discriminator network, as shown in Fig. 2. The generator uses the shape of an input character to make the generated character reality and use the style of a target character to show what kind of character is to be generated. Unlike common discriminator of GAN, which only need to judge whether the input is true or false, the discriminator of FCGAN has to discriminate whether the generated character is similar with the target real character. So we put two images into the discriminator together, one is the generated fake character while the other one is the real target character image, regarded as fake data or two same real

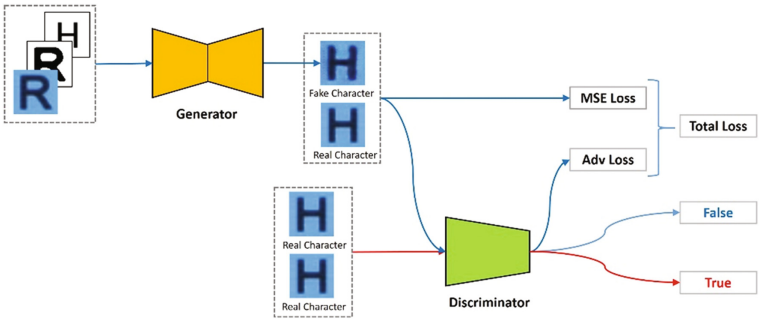


**Fig. 2.** Overview of the proposed FCGAN. The generator use an input character, the shape of the input character and the style of a target character to generate a forgery character image. The generator uses MSE loss to make sure the generated character reality. The discriminator uses two same real characters as real data and a generated fake character with a real character as fake data during training so that it can tell whether the generated character is like the target real character.

target character images regarded as real data. We also use the MSE loss to make sure that each generated character is similar to the real character.

The proposed generator network is composed of a series of convolutions, ResNet blocks [20] and transposed convolutions. The details of construction are shown in Table 1. The activation functions of the convolution and transposed convolution blocks are defined as a leaky-ReLu function, while the activation functions of the ResNet blocks refer to ReLu function. Specifically, we firstly concatenate the input character (3 channels), the shape image of the input character (1 channel), and the style image of the target character (1 channel) together, which are further involved into the Generator network. Then, we utilize a series of convolutions to produce 8x8x512 latent features, which describe the inherent information of the input data. After convolution operations, we encode the computation of the first layer with ResNet blocks. After encoding with 8 layers of ResNet blocks, we decode the latent feature with transposed convolution operations. Finally, we get a $64 \times 64 \times 3$ image, which is the result of the generated fake image.

The discriminator network takes a $64 \times 64 \times 6$ vector as the input, which comprises a pair of images. The discriminator network considers a target real image and a generated fake image as the input. The construction details of the discriminator network are shown in Table 2. During testing, we evaluate the reality of the input image, i.e., the generated fake image.

### 3.2 End-to-End Joint Training

In this subsection, we present the process of end-to-end joint training. We utilize the three types of images for training, namely, the input character image, the shape image of the input character shape, and the style image of the target character. Note that the input character and the target character are chosen from the same class, which means that these two types of images have the same background and foreground color but different character types. Our task is thus to generate a target character, i.e., a fake character, using the input character and the style of the target character.

**Table 1.** Construction details of the generator network.

| | Type | Kernel | Stride | Padding | Out channel |
|---|---|---|---|---|---|
| Convolution blocks | Conv | 7 | 1 | 3 | 64 |
| | Conv | 3 | 2 | 1 | 128 |
| | Conv | 3 | 2 | 1 | 256 |
| | Conv | 3 | 2 | 1 | 512 |
| ResNet block x8 | Conv | 3 | 1 | 1 | 512 |
| | InstanceNorm | | | | |
| | Conv | 3 | 1 | 1 | 512 |
| | InstanceNorm | | | | |
| ConvTranspose blocks | ConvTrans | 3 | 2 | 1 | 256 |
| | ConvTrans | 3 | 2 | 1 | 128 |
| | ConvTrans | 3 | 2 | 1 | 64 |
| | ConvTrans | 1 | 1 | 0 | 3 |

**Table 2.** The Construction details of the discriminator network.

| | Type | Kernel | Stride | Padding | Out channel |
|---|---|---|---|---|---|
| Convolution blocks | Conv | 7 | 2 | 3 | 64 |
| | Conv | 3 | 2 | 1 | 128 |
| | Conv | 3 | 2 | 1 | 256 |
| | Conv | 3 | 2 | 1 | 512 |
| | Conv | 3 | 2 | 1 | 1024 |
| | Conv | 3 | 2 | 1 | 2048 |
| | Conv | 3 | 2 | 1 | 1 |

The proposed work explores the shape of the input character to improve the quality of the fake generated character. During the construction of generator, we adopt L1 loss to be the construction loss to improve our result, which is expressed as follows:

$$L_{construct} = E_{x\sim p_{source},y\sim p_{target},s,t}[||G(x,s,t)-y||] \tag{1}$$

where E represents the expectation value, function G(.) refers to the generator network, $s$ refers to the shape image of the input character, $t$ represents style image of target character, and $p$ denotes data distribution. Note that we adopt L1 loss rather than L2 loss since training with L1 loss can be optimized by a faster way and produce harper and cleaner images.

To evaluate the overall performance, we adopt the least-square loss as the loss function of the GAN, which is defined as:

$$L_{adv} = E_{y\sim p_{target}}\left[(D(y,y)-1)^2\right] + E_{x\sim p_{source},y\sim p_{target},s,t}\left[D(y,G(x,s,t))^2\right] \tag{2}$$

where D(.) is the discriminator network. Above all, the total loss of GAN can be computed as:

$$L_{total} = \arg min_G max_D L_{adv} + \lambda L_{construct} \tag{3}$$

To solve Eq. 3, we use Adam [21] as our optimization method.

## 4 Experimental Results

### 4.1 Implementation Details

For experimentation, we use character images from ICDAR 2015 [22] as our training data. We adopt the label of text box information to extract single characters from the dataset. As these extracted characters have different sizes, hence we first resize them to standard size of 64 × 64.

Next, we generate our training dataset for FCGAN. Note that input characters have the same background and font style with real characters. The system thus requires a pair of characters for training, i.e., an input character and a real character which have similar background and font style. To make characters as pairs, we first classify character images into different classes based on background and font styles. Some example images belonging to the same class are shown in Fig. 3. It is noted that we classify characters into classes manually in this step. After classifying, we use character pairs to train the GAN network.


**Fig. 3.** Some example images belonging to one class. Note that they have the same background and font style. When training the network, we randomly choose two images from one class of character images, which are regarded as the input character and the target character, respectively.

After pairing, we extract shape information from the image of the input character and the style of the target character. Note that we use the style image to define what type of a character to be generated, and use the shape to ensure the quality of generation. Specifically, we use Otsu method with an adaptive threshold to extract the shape information of the input character. We also augment the data by exchanging the RGB channel of one image. It is noted that all the character images are defined as 64 × 64. In total, we choose 52 styles of images as our target characters, which are used to show the detailed character types to be generated. The training data are shown in Fig. 4. After dataset processing, we totally get 12435 characters and 52 styles of target characters for training.

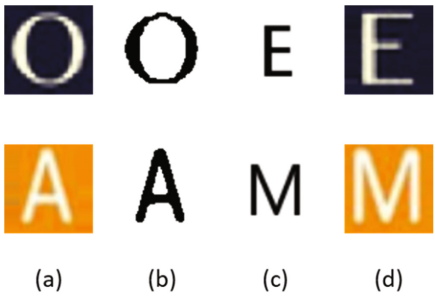


**Fig. 4.** The training data. From left to right: (a) are input character images, (b) are the shape of input characters, (c) are the style of target characters and (d) are the target character image.

During training, we randomly pairs of images, i.e., an input character and a real character belonging to the same class. The learning rate of the generator and the discriminator network are 0.0001 and 0.00001, respectively. We train the network with 32 batch size.

### 4.2 Performance Analysis

To test the quality of fake character images by the proposed system, we estimate the standard quality metrics, namely, SSIM and PSNR. SSIM is defined as:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{4}$$

where $\mu$ refers to the average of data, $\sigma$ means the variance of data, $\sigma_{xy}$ means the covariance of $x$ and $y$, and $c_1 = (k_1 L)^2, c_2 = (k_2 L)^2$, where L is the dynamic range of pixel-values, $k_1 = 0.01$ and $k_2 = 0.03$ by default.

PSNR is defined by

$$PSNR = 10 \cdot log_{10}\left(\frac{MAX_I^2}{MSE}\right) \tag{5}$$

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2 \tag{6}$$

where $MAX_I^2$ is the maximum possible pixel value of the image.

For estimating the quality measures, fake generated and target character images are considered. It is noted that high PSNR and SSIM values indicate a fake character image has better quality.

Qualitative results of the proposed work at image level can be seen in Fig. 5, where the fake generated characters are inserted back to the same image. At the same time, individual fake characters for real characters can be seen in Fig. 6. It is observed from Figs. 5 and 6 that the proposed system generates fake characters well. The PSNR and SSIM are reported in Table 3. This shows that we can rely the proposed system for generating fake character images.


**Fig. 5.** The generated results, where the upper refers to the real source images, while the bottom represents images containing generated forgery characters. Note that we have changed several characters in these images.

To validate the effectiveness of the proposed system, we conduct experiments for text detection and recognition, respectively. For the text detection experiment, we use two state of the art text detection methods, namely, CTPN [23] and EAST [24] to detect the characters in real scene images and the images with generated characters. Note that generated fake images contain both fake generated characters and original ones. The results are reported in Table 4, where it can be seen that the measures of real images and the images containing fake character images score almost the same. We also expand the training dataset with the generated fake images to test the text detection result. The results can be seen in Table 5, which shows the detection result is improved after add images which contain generated characters. This means that most of the generated fake characters are treated as real ones.

We also conduct recognition experiments for real and generated fake characters to test whether the generated fake characters preserve the actual shapes or not. For this, we use CRNN algorithm [25], which explores deep learning for text recognition. The results of the recognition methods are reported in Table 6, where one can see that the recognition rate of the fake character is almost the same as the recognition rate of real characters.

**Fig. 6.** The examples of generated characters and its real target characters, where the upper shows real target character images, while the bottom gives generated forgery character images.

**Table 3.** Measurement results.

| | SSIM | PSNR |
|---|---|---|
| The proposed | 0.6719 | 17.457 |

**Table 4.** Comparison on performance of text detection.

| Method | Real scene images | | | Generated fake images | | |
|---|---|---|---|---|---|---|
| | Precision | Recall | F-Measure | Precision | Recall | F-Measure |
| CTPN [23] | 0.66 | 0.535 | 0.59 | 0.65 | 0.519 | 0.577 |
| EAST [24] | 0.298 | 0.464 | 0.363 | 0.28 | 0.445 | 0.344 |

**Table 5.** Comparison on adding generated fake images as training data.

| Method | 900 real scene images | | | 900 real scene images + 100 generated fake images | | |
|---|---|---|---|---|---|---|
| | Precision | Recall | F-Measure | Precision | Recall | F-Measure |
| EAST [24] | 0.315 | 0.481 | 0.381 | 0.336 | 0.51 | 0.405 |

**Table 6.** Comparison on performance of text recognition.

| Category | CRNN |
|---|---|
| Real characters | 0.523 |
| Generated fake characters | 0.541 |

## 5 Conclusion

In this paper, we propose an automatic system namely FCGAN to generate artificial fake characters. We use ICDAR 2015 dataset for experimentation. The experimental results on text detection and recognition shows that the proposed system preserve the quality of the images as real images, which is very useful and effective.

**Acknowledgment.** This work was supported by the Natural Science Foundation of China under Grant 61672273, Grant 61832008 and Grant 61702160, the Science Foundation for Distinguished Young Scholars of Jiangsu under Grant BK20160021, Scientific Foundation of State Grid Corporation of China (Research on Ice-wind Disaster Feature Recognition and Prediction by Few-shot Machine Learning in Transmission Lines), National Key R&D Program of China under Grant 2018YFC0407901, the Fundamental Research Funds for the Central Universities under Grant 2016B14114, the Science Foundation of JiangSu under Grant BK20170892, and the open Project of the National Key Lab for Novel Software Technology in NJU under Grant K-FKT2017B05.

## References

1. Farid, H.: Image forgery detection. IEEE Signal Process. Mag. **26**(2), 16–25 (2009)
2. Jaderberg, M., et al.: Synthetic data and artificial neural networks for natural scene text recognition (2014). arXiv preprint: arXiv:1406.2227
3. Zheng, Z., Zheng, L., Yang, Y.: Unlabeled samples generated by GAN improve the person re-identification baseline in vitro (2017). arXiv preprint: arXiv:1701.07717
4. Goodfellow, I., et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems (2014)
5. Zhu, J.-Y., et al.: Unpaired image-to-image translation using cycle-consistent adversarial networks (2017). arXiv preprint: arXiv:1703.10593
6. Iizuka, S., Simo-Serra, E., Ishikawa, H.: Globally and locally consistent image completion. ACM Trans. Graph. (TOG) **36**(4), 107 (2017)
7. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein GAN (2017). arXiv preprint: arXiv:1701.07875
8. Mao, X., et al.: Least squares generative adversarial networks. In: 2017 IEEE International Conference on Computer Vision (ICCV). IEEE (2017)
9. Denton, E.L., Chintala, S., Fergus, R.: Deep generative image models using a Laplacian pyramid of adversarial networks. In: Advances in Neural Information Processing Systems (2015)
10. Zhu, J.-Y., Krähenbühl, P., Shechtman, E., Efros, A.A.: Generative visual manipulation on the natural image manifold. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016, Part V. LNCS, vol. 9909, pp. 597–613. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46454-1_36
11. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks (2015). arXiv preprint: arXiv:1511.06434
12. Ledig, C., et al.: Photo-realistic single image super-resolution using a generative adversarial network (2016). arXiv preprint: arXiv:1609.04802
13. Salimans, T., et al.: Improved techniques for training GANs. In: Advances in Neural Information Processing Systems (2016)
14. Conotter, V., Boato, G., Farid, H.: Detecting photo manipulation on signs and billboards. In: 2010 17th IEEE International Conference on Image Processing (ICIP). IEEE (2010)
15. Abramova, S.: Detecting copy-move forgeries in scanned text documents. Electron. Imaging **2016**(8), 1–9 (2016)
16. Isola, P., et al.: Image-to-image translation with conditional adversarial networks (2016). arXiv preprint: arXiv:1611.07004
17. Reed, S., et al.: Generative adversarial text to image synthesis (2016). arXiv preprint: arXiv:1605.05396

18. Liu, M.-Y., Tuzel, O.: Coupled generative adversarial networks. In: Advances in Neural Information Processing Systems (2016)
19. Liu, M.-Y., Breuel, T., Kautz, J.: Unsupervised Image-to-Image Translation Networks (2017). arXiv preprint: arXiv:1703.00848
20. He, K., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
21. Kingma, D., Ba, J.: Adam: A method for stochastic optimization (2014). arXiv preprint: arXiv:1412.6980
22. Karatzas, D., et al.: ICDAR 2015 competition on robust reading. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR). IEEE (2015)
23. Tian, Z., Huang, W., He, T., He, P., Qiao, Y.: Detecting text in natural image with connectionist text proposal network. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 56–72. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_4
24. Zhou, X., et al.: EAST: An Efficient and Accurate Scene Text Detector (2017). arXiv preprint: arXiv:1704.03155
25. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**(11), 2298–2304 (2017)