LETTER

# Robust CAPTCHA Image Generation Enhanced with Adversarial Example Methods*

**Hyun KWON**[†,††], ***Student Member***, **Hyunsoo YOON**[†], ***and*** **Ki-Woong PARK**[†††a)], ***Nonmembers***

**SUMMARY** Malicious attackers on the Internet use automated attack programs to disrupt the use of services via mass spamming, unnecessary bulletin boarding, and account creation. Completely automated public turing test to tell computers and humans apart (CAPTCHA) is used as a security solution to prevent such automated attacks. CAPTCHA is a system that determines whether the user is a machine or a person by providing distorted letters, voices, and images that only humans can understand. However, new attack techniques such as optical character recognition (OCR) and deep neural networks (DNN) have been used to bypass CAPTCHA. In this paper, we propose a method to generate CAPTCHA images by using the fast-gradient sign method (FGSM), iterative FGSM (I-FGSM), and the DeepFool method. We used the CAPTCHA image provided by python as the dataset and Tensorflow as the machine learning library. The experimental results show that the CAPTCHA image generated via FGSM, I-FGSM, and DeepFool methods exhibits a 0% recognition rate with $\epsilon = 0.15$ for FGSM, a 0% recognition rate with $\alpha = 0.1$ with 50 iterations for I-FGSM, and a 45% recognition rate with 150 iterations for the DeepFool method.

***key words:*** *CAPTCHA, adversarial example, machine learning, deep neural network*

## 1. Introduction

Internet services can be disrupted by using attack programs such as mass spam mail, posting on an Internet bulletin board, and increasing random subscribers on an Internet site. One solution to prevent such automated attacks is the completely automated public turing test to tell computers and humans apart (CAPTCHA) system [1]. CAPTCHA is a security service that identifies whether a service user is a machine or a person through the use of distorted letters, pictures, and voices that are difficult to be understood by an automated machine program, but can be recognized by humans. It provides a service user with letters, voices, and images that only humans can understand. If it receives a true value, it recognizes the user as a person and provides the service. CAPTCHA plays an important role as a security solution because if the malicious automatic attack program is run without any control, it results in malicious attack damage and significant consumption of network traffic.

The types of CAPTCHA include text-based CAPTCHA, speech-based CAPTCHA, and image-based CAPTCHA. Initially, text-based CAPTCHAs were developed. This was followed by the introduction of various types of CAPTCHAs such as audio, video, and images. However, the newer types of CAPTCHAs have their own disadvantages. For example, in the case of voice CAPTCHAs, a user requires a separate device such as a speaker, and in a noisy environments, the probability of recognising the person is low. In the case of an image, information can be conveyed more intuitively as compared to that in the case of letters; however, there are limitations in the generation of such images. Due to these limitations, text-based CAPTCHA is still widely used [2].

Existing methods of optical character recognition (OCR) [3] and deep neural networks (DNN) [4] are capable of bypassing CAPTCHA. However, the DNN method has a vulnerability in the adversarial example. In the method proposed by Szegedy et al. [5], the adversarial example is a method of adding very little noise to the original data such that the CAPTCHA can be properly recognized by humans, but can be misrecognized by the DNN. Generating a CAPTCHA image by using this adversarial example technique can result in machine misrecognition while maintaining human recognition rates. In this paper, we propose a robust method for CAPTCHA image generation, using the adversarial example method. This method adds a bit of noise to the original CAPTCHA image, creating a CAPTCHA that keeps the human perception rate intact and the machine unreadable. To demonstrate the performance of the proposed method, we analyzed the recognition performance of a conventional CAPTCHA image and that of a CAPTCHA image incorporating adversarial example techniques such as the fast-gradient sign method (FGSM) [6], iterative FGSM (I-FGSM) [7], and the DeepFool [8] method. In addition, we analyzed the performance of the proposed method by using the CAPTCHA image dataset provided by Python.

The remainder of this paper is organized as follows; the methodology is introduced in Sect. 2. The experiment and results are described in Sect. 3, and the proposed method is discussed in Sect. 4. Finally, the conclusions of this study are summarised in Sect. 5.

Manuscript received October 29, 2019.
Manuscript publicized January 15, 2020.

[†]The authors are with School of Computing, Korea Advanced Institute of Science and Technology, Korea.
[††]The author is also with Department of Electrical Engineering, Korea Military Academy, Korea.
[†††]The author is with Department of Computer and Information Security, Sejong University, Korea.

*This work was supported by the National Research Foundation of Korea (NRF) (NRF-2017R1C1B2003957) and the Institute for Information and Communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2018-0-00420).

a) E-mail: woongbak@sejong.ac.kr

DOI: 10.1587/transinf.2019EDL8194

## 2. Methodology

In this study, the adversarial example method applied to the CAPTCHA is generated through the feedback on the target model for the generated adversarial CAPTCHA, as shown in Fig. 1. There are three methods applied in this study: FGSM, I-FGSM, and DeepFool. The FGSM can be used to obtain $x^*$ by using $L_\infty$:

$$x^* = x + \epsilon \cdot \mathrm{sign}(\nabla loss_{F,t}(x)), \tag{1}$$

where $F$ is an object function, and $t$ is a target class. In the first step of the FGSM, the gradient descent is changed from the original $x$, based on the $\epsilon$ value, and $x^*$ is obtained through optimization. This is a simple method that demonstrates good performance.

I-FGSM is an extension of FGSM. In this method, instead of updating the amount of $\epsilon$ at every step, a smaller amount, $\alpha$, is changed and eventually clipped by $\epsilon$:

$$x_i^* = x_{i-1}^* - \mathrm{clip}_\epsilon(\alpha \cdot \mathrm{sign}(\nabla loss_{F,t}(x_{i-1}^*))). \tag{2}$$

I-FGSM creates an adversarial example during a given iteration on a target model. Compared to FGSM, it exhibits a higher attack prevention rate in terms of white box attacks. Therefore, I-FGSM exhibits better performance as compared to FGSM.

The DeepFool method creates an adversarial example that is similar to the original image. To create an adversarial example, this method focuses on $x^*$, using the linearization approximation method in a neural network. However, the DeepFool method is more complicated than the FGSM because the neural network requires many iterations and is not completely linear.

## 3. Experiments and Analysis

### 3.1 Dataset

The CAPTCHA data set used the CAPTCHA image provided by the python library [9]. It is composed of four numbers or English alphabets, and each letter consists of distortion, horizontal lines, and background dots. A total of 200,000 training datasets and 10,000 test datasets were used in this study.

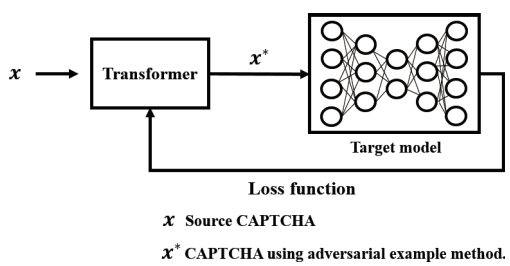



**Fig. 1** Overview of adversarial example method applied to CAPTCHA.

### 3.2 Target Model

The structure of the target model for the CAPTCHA consists of the convolutional neural network (CNN) structure [10]. Table 1 shows the structure of the target model, and Table 2 presents information about the training parameters of the model. The target model, which trained a total of 200,000 datasets, had an accuracy of 71.42% for the new 10,000 test datasets.

### 3.3 Generating CAPTCHA Image Using Adversarial Example Methods

Regarding the adversarial example generation applied to the CAPTCHA, we analyzed the recognition rate of the target model by changing the value of $\epsilon$ in the FGSM method. Additionally, we analyzed the recognition rate of the target model by changing the value of $\alpha$ and the iteration in I-FGSM. Furthermore, we analyzed the recognition rate of the target model by changing the iteration in Deepfool. To verify the performance of the adversarial CAPTCHA, we generated 1000 CAPTCHAs for each adversarial example method.

### 3.4 Experimental Results

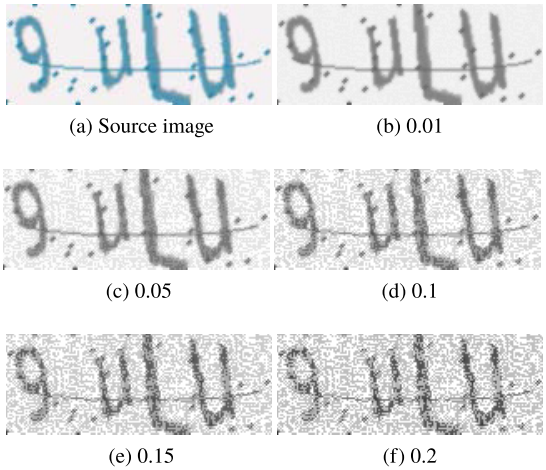
In this section, we present the experimental results and the analysis of the performance of the CAPTCHA image sample, based on the FGSM, I-FGSM, and DeepFool methods by evaluating the recognition rates of the target model. The recognition rate refers to the match rate between the CAPTCHA letters recognized by the target model and the actual letters. For example, if 60 of the 100 samples match exactly, then the match rate is 60%.

**Table 1** Target model architecture.

| Layer type | Shape |
|---|---|
| Convolution+ReLU | [2, 32, 3] |
| Max pooling | [2, 2] |
| Convolution+ReLU | [3, 64, 3] |
| Max pooling | [2, 2] |
| Convolution+ReLU | [64, 64, 3] |
| Max pooling | [2, 2] |
| Convolution+ReLU | [64, 128, 3] |
| Max pooling | [2, 2] |
| Convolution+ReLU | [128, 128, 3] |
| Max pooling | [2, 2] |
| Fully connected+ReLU | [640] |
| Softmax | [1024] |

**Table 2** Target model parameter.

| Parameter | Values |
|---|---|
| Learning rate | 0.001 |
| Momentum | 0.9 |
| Dropout | 0.5 |
| Batch size | 100 |
| Epochs | 50 |

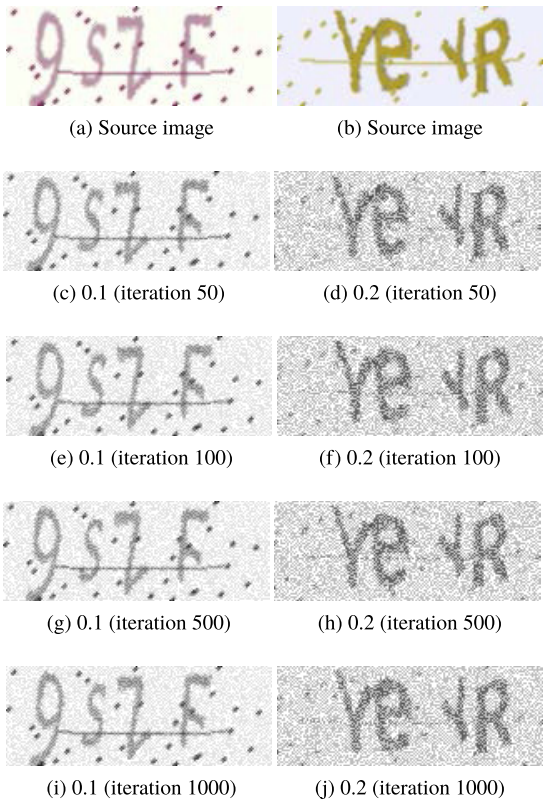


(a) Source image (b) 0.01

(c) 0.05 (d) 0.1

(e) 0.15 (f) 0.2

**Fig. 2** CAPTCHA image by applying FGSM method according to $\epsilon$.



(a) Source image (b) Source image

(c) 0.1 (iteration 50) (d) 0.2 (iteration 50)

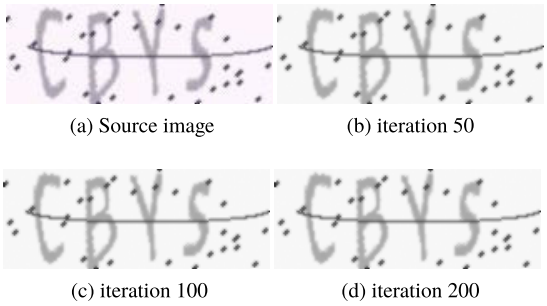
(e) 0.1 (iteration 100) (f) 0.2 (iteration 100)

(g) 0.1 (iteration 500) (h) 0.2 (iteration 500)

(i) 0.1 (iteration 1000) (j) 0.2 (iteration 1000)

**Fig. 3** CAPTCHA image by applying I-FGSM method according to $\alpha$ and iteration.

Figure 2 shows a sample CAPTCHA image by using the FGSM method and the source CAPTCHA. In the figure, we present the CAPTCHA images generated by changing the $\epsilon$ values to 0.01, 0.05, 0.1, 0.15, and 0.2. In terms of human perception, the generated CAPTCHA adds noise as a whole, but there is no difficulty in identification. Figure 3 shows a sample CAPTCHA image using the I-FGSM method and the source CAPTCHA. The $\alpha$ values were selected as 0.1 and 0.2, and the number of iterations were 50, 100, 500, and 1000. The image distortion, according



(a) Source image (b) iteration 50

(c) iteration 100 (d) iteration 200

**Fig. 4** CAPTCHA image by applying I-FGSM method according to number of iterations.

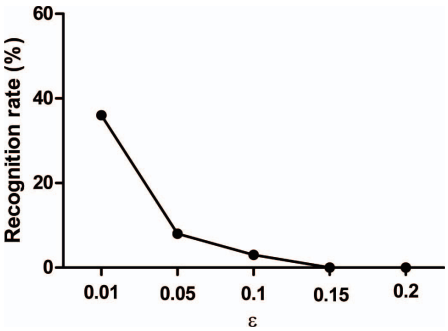



**Fig. 5** Recognition rate for CAPTCHA using FGSM method.

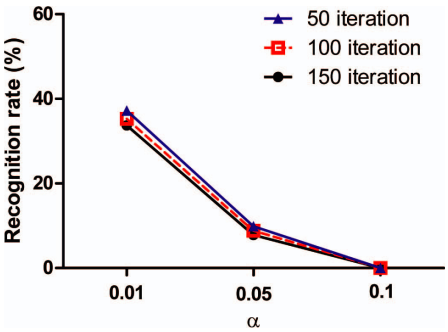



**Fig. 6** Recognition rate for CAPTCHA using I-FGSM method.

to the number of repetitions, does not change significantly. However, the image distortion by the $\alpha$ value is more pronounced. Figure 4 shows a sample CAPTCHA image, using the DeepFool method and source CAPTCHA. In terms of human perception, there is almost no difference in image distortion, according to the number of iterations.

Figure 5 shows the recognition rate of the target model for CAPTCHA using the FGSM method. As $\epsilon$ increases, the recognition rate decreases in the target model. When $\epsilon$ is 1.5, the recognition rate is zero. Figure 6 shows the recognition rate on a CAPTCHA image using the I-FGSM method. In the figure, as the value of $\alpha$ increases, the recognition rate for the target model decreases. When the value of $\alpha$ is approximately 0.1, the recognition rate is zero, demonstrating that the I-FGSM is better than FGSM. On the contrary, the number of iterations was found to have little effect on the recognition rate. Figure 7 shows the recognition rate for

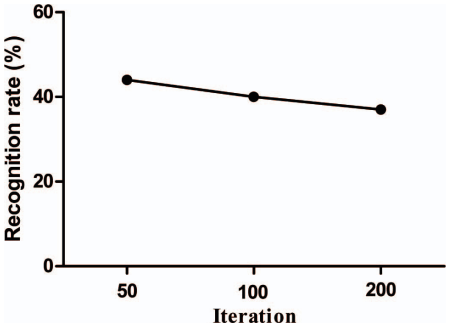



**Fig. 7** Recognition rate for CAPTCHA using DeepFool method.

a CAPTCHA image, using the DeepFool method. As the number of iterations increase, the recognition rate for the model decreases slightly. In terms of recognition rate, the effect of the number of iterations is insignificant.

## 4. Discussion

By applying adversarial example methods, such as the FGSM, I-FGSM, and DeepFool, to existing CAPTCHA, we can generate robust CAPTCHA images to inhibit machine learning service attack models such as DNNs. Based on this assumption, a transformer creates an adversarial CAPTCHA sample that can trick the target model by extracting the content of the result value for the input value and calculating it based on the loss function.

The recognition rate decreased significantly as the distortion coefficient increased, as seen in Figs. 5 and 6. In the case of FGSM, the recognition rate reached 0% when $\epsilon$ was 0.15. In the case of I-FGSM, the recognition rate reached 0% when $\alpha$ was 0.1 for 50 iterations. On the other hand, the number of iterations do not affect the recognition rate significantly.

In terms of human perception, the larger the distortion coefficients, as seen in Figs. 2, 3, and 4, the more distorted the image was; however, human perception was maintained. Also, even if the number of iterations increased, the image distortion was hardly recognized in terms of human perception. There are differences between the CAPTCHA domain and the existing image domain. First, in the existing image domain, the adversarial example includes little noise, which is not recognized by humans. However, CAPTCHA is easy in terms of generation because it does not have to degrade the human recognition rate, even in the presence of some noise. Second, as the false values in the CAPTCHA system are incorrect, there is no need to create targeted adversarial examples.

## 5. Conclusion

This paper presents a robust method for CAPTCHA image generation in a machine learning model by using the adversarial example method. In this method, we applied adversarial example methods, i.e., FGSM, I-FGSM, and DeepFool, to generate a CAPTCHA image that prevented the recognition via machine model while maintaining the human recognition rate. Based on the experimental results, the CAPTCHA exhibited a recognition rate of 0% with $\epsilon$ = 0.15 for FGSM, a recognition rate of 0% with $\alpha$ = 0.1 over 50 iterations for I-FGSM, and a recognition rate of 45% over 150 iterations for the DeepFool method. The application of the adversarial example method has proved the possibility of improving the performance of CAPTCHA.

Future research scopes include the application of text-based CAPTCHA as well as voice and video CAPTCHAs. In addition, future research should focus on studying the generation of multiple CAPTCHA images by applying methods such as the generative adversarial network [11]. Lastly, usability evaluation supported by data and comparing robustness of our CAPTCHA with that of other similar CAPTCHAs should also be the focus of future studies.

### References

[1] L.V. Ahn, M. Blum, N.J. Hopper, and J. Langford, "Captcha: Using hard ai problems for security," International Conference on the Theory and Applications of Cryptographic Techniques, pp.294–311, Springer, 2003.
[2] E. Bursztein, M. Martin, and J. Mitchell, "Text-based captcha strengths and weaknesses," Proceedings of the 18th ACM conference on Computer and communications security, pp.125–138, ACM, 2011.
[3] A. Hindle, M.W. Godfrey, and R.C. Holt, "Reverse engineering captchas," 2008 15th Working Conference on Reverse Engineering, pp.59–68, IEEE, 2008.
[4] R. Hussain, K. Kumar, H. Gao, and I. Khan, "Recognition of merged characters in text based captchas," 2016 IEEE 3rd International Conference on Computing for Sustainable Global Development (INDIACom), pp.3917–3921, 2016.
[5] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," arXiv preprint arXiv:1312.6199, 2013.
[6] I.J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," arXiv preprint arXiv:1412.6572, 2014.
[7] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," arXiv preprint arXiv:1607.02533, 2016.
[8] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," Proceedings of the IEEE conference on computer vision and pattern recognition, pp.2574–2582, 2016.
[9] N. Ketkar, "Introduction to pytorch," Deep Learning with Python, pp.195–208, Springer, 2017.
[10] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol.86, no.11, pp.2278–2324, 1998.
[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," Advances in Neural Information Processing Systems, pp.2672–2680, 2014.