Hindawi
Security and Communication Networks
Volume 2021, Article ID 9930608, 9 pages
https://doi.org/10.1155/2021/9930608

*Research Article*

# Augmented Data Selector to Initiate Text-Based CAPTCHA Attack

**Aolin Che,[1] Yalin Liu,[1] Hong Xiao,[2] Hao Wang,[3] Ke Zhang,[4] and Hong-Ning Dai[1]**

[1]*Macau University of Science and Technology, Macau, China*
[2]*Guangdong University of Technology, Guangzhou 510006, China*
[3]*Department of Computer Science in Norwegian University of Science and Technology, Gjøvik, Norway*
[4]*School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China*

Correspondence should be addressed to Hong Xiao; wh_red@gdut.edu.cn

Received 4 March 2021; Accepted 31 May 2021; Published 16 June 2021

Academic Editor: Ting Chen



In the past decades, due to the low design cost and easy maintenance, text-based CAPTCHAs have been extensively used in constructing security mechanisms for user authentications. With the recent advances in machine/deep learning in recognizing CAPTCHA images, growing attack methods are presented to break text-based CAPTCHAs. These machine learning/deep learning-based attacks often rely on training models on massive volumes of training data. The poorly constructed CAPTCHA data also leads to low accuracy of attacks. To investigate this issue, we propose a simple, generic, and effective preprocessing approach to filter and enhance the original CAPTCHA data set so as to improve the accuracy of the previous attack methods. In particular, the proposed preprocessing approach consists of a data selector and a data augmentor. The data selector can automatically filter out a training data set with training significance. Meanwhile, the data augmentor uses four different image noises to generate different CAPTCHA images. The well-constructed CAPTCHA data set can better train deep learning models to further improve the accuracy rate. Extensive experiments demonstrate that the accuracy rates of five commonly used attack methods after combining our preprocessing approach are 2.62% to 8.31% higher than those without preprocessing approach. Moreover, we also discuss potential research directions for future work.

## 1. Introduction

Completely Automated Public Turing tests to tell Computers and Humans Apart, abbreviated as CAPTCHA, is a kind of test that automatically distinguishes human and robot operations. In 2003, Von Ahn et al. [1] firstly discuss the use of the dedicated-designed CAPTCHA schemes to prevent the attack from robots and thereby enhance network security. After that, researchers have invented a variety of CAPTCHA schemes, such as text-based CAPTCHAs, image-based CAPTCHAs, slider-type CAPTCHAs, and SMS CAPTCHAs. These CAPTCHA schemes are widely used in websites and mobile applications, to protect the corporations and users' passwords and data [2–6]. As shown in Figure 1, we present a statistical pie graph to count the types of the commonly used CAPTCHAs from 100 popular websites. According to our survey, nearly 55% of websites use text-based CAPTCHAs as security and identification mechanisms, far exceeding other types, due to the low development and maintenance costs. For this reason, we mainly focus on the study of text-based CAPTCHAs in this paper.

A critical design point of the text-based CAPTCHA is to prevent different computer-based attacks from recognizing the text information in CAPTCHAs, because these attacks may cause serious data leakages and economic losses for the enterprises and users [7]. To achieve this goal, researchers design many encryption methods to hinder computer recognition according to the structure of the text-based CAPTCHA. Generally, a text-based CAPTCHA image consists of three layers [8], as illustrated in Figure 2. The foreground layer ordinarily has some occluding lines, noisy spots, and other interference elements. The character layer contains characters that human users can recognize. Designers usually add some extra antisegmentation and anti-recognition characteristics, such as varied typeface, overlapping, rotation, distortion, and waving on text-based CAPTCHAs. The background layer is usually a

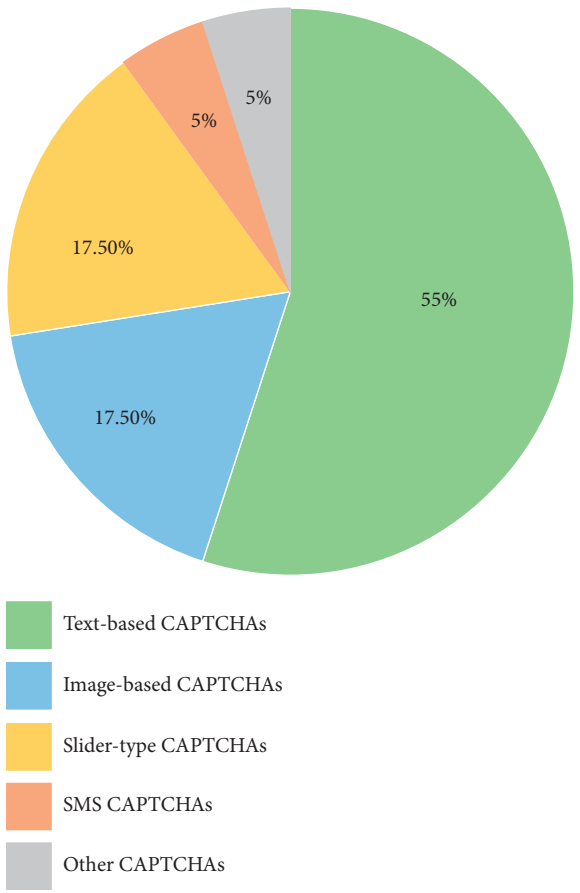



Figure 1: A statistical pie graph of the CAPTCHA types from 100 popular websites with each of them possessing more than 1 million daily visits. The chosen websites are very comprehensive, including varieties of practically applied fields (e.g., search engines, corporate websites, game websites, media-sharing sites, brand-building sites, and school websites).

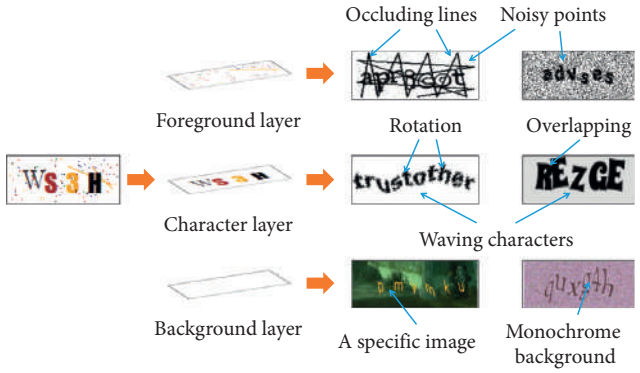



Figure 2: An example of text-based CAPTCHAs (reproduced from [8]). Generally, a text-based CAPTCHA comprises three layers: foreground layer, character layer, and background layer.

monochrome background board or a specific image that can interfere with the recognition of characters by computers.

A good attack method can identify the text-based CAPTCHA characters after adding multiple complex security features. In recent years, researchers have proposed different attack methods to solve text-based CAPTCHAs [9, 10]. One preferred attack method is using machine learning to conduct a computer-based recognition of text-based CAPTCHAs [11]. In addition, as an evolution of machine learning, deep learning is capable of making accurate decisions in the field of image recognition [12]. For this reason, recent research efforts began to explore deep learning-based attack methods to solve text-based CAPTCHAs with a high-accurate accuracy rate [7, 8]. However, the previous studies are often based on the massive volumes of unlabeled training data (e.g., millions of samples [13]), thereby bringing extra costs which are labor-intensive and time-consuming to label all training data. In fact, not all data is meaningful for training because the training on some data does not significantly improve the accuracy of the attack [14, 15]. It is not worth wasting a lot of time for us to manually label these types of data.

In this paper, we propose a preprocessing approach to effectively select CAPTCHA images so as to improve the recognition rate of machine/deep learning-based CAPTCHA attack methods. Our preprocessing approach can significantly improve the accuracy rate of commonly used attack methods. In particular, our proposed preprocessing approach contains two preprocessing modules: data selector and data augmentor. In detail, the data selector module selects some specific text-based CAPTCHAs from the original CAPTCHA data set as the training set according to the given regulations. Then, the data augmentor module randomly adds noises for each training data to obtain a higher target accuracy rate. We evaluate the proposed preprocessing approach by integrating it into five commonly used machine learning-based attack methods. These commonly used attack methods include convolutional neural network (CNN), support vector machine (SVM), decision tree (DT), random forest (RF), and logistic regression (LR). The experimental results show that, in the case of 3,319 original text-based CAPTCHAs, the accuracy rates of the five attack methods after adding our preprocessing approach are from 2.62% to 8.31% higher than the accuracy rates without adding the preprocessing module. Our experiment proves that the accuracy of text-based CAPTCHA attacks can be significantly improved by strengthening preprocessing, which however is ignored in existing studies.

In summary, this paper makes the following contributions:

(1) We present a simple and generic data-preprocessing approach to enhance the accuracy and efficiency of the text-based CAPTCHAs attacking process that utilizes machine/deep learning methods. Our preprocessing approach is composed of two modules: (i) a data selector to efficiently filter out CAPTCHAs with training significance and (ii) a data augmentor to improve the accuracy of the attack method.

(2) We perform comprehensive ablation experiments by combining our approach in five commonly used machine/deep learning methods, including CNN, SVM, DT, RF, and LR. The experiment results confirm that our approach can significantly improve

the high accuracy of the general attack methods, indicating the wide applicability of our approach.

The remainder of this paper is organized as follows: Section 2 overviews the commonly used attack methods based on machine/deep learning to break text-based CAPTCHAs. Section 3 introduces the general attacking process and the detailed design of our preprocessing approach. In Section 4, we perform a set of ablation experiments on five commonly used training methods to demonstrate that our preprocessing approach can effectively improve the accuracy rate of existing attack methods. Moreover, Section 5 discusses the future research related to man-machine verification schemes based on the current text-based CAPTCHAs. Section 6 concludes this paper.

## 2. Previous Attacking Methods

In recent years, relying on the huge magnitude of data, many training model-based attack methods are presented to recognize text-based CAPTCHAs. We can roughly divide them into two categories: machine learning and deep learning attack methods.

Computer security has received a growing interest recently in diverse computer-based systems [16–25]. Traditional machine learning methods include SVM, DT, and RF, which are widely used to break text-based CAPTCHAs. Mori and Malik first attempt to attack the text-based CAPTCHA in 2003 [26]. They obtained a 33% accuracy rate on a text-based CAPTCHA called Gimpy by exploring sophisticated object recognition algorithms. Since then, researchers have proposed various attack algorithms for text-based CAPTCHAs [27]. The most representative work is that Bostik and Klecka compared five traditional machine learning methods to attack the character bubble CAPTCHA [28]. Their experimental results show that SVM had the highest accuracy rate but DT had the lowest computational cost. It is worth noting that these machine learning-based attack methods rely heavily on character segmentation. Once a character cannot be completely segmented, the accuracy rate of the machine learning-based attack method will be greatly reduced. Therefore, we summarize the limitations of using machine learning-based attack methods: (1) it is necessary to design a complex character segmentation algorithm so that the preprocessing is extremely cumbersome; (2) it needs to manually find the characteristics of each character, consequently resulting in a relatively low accuracy rate.

Deep learning-based attack methods are mainly attributed to convolutional neural networks. In contrast to traditional machine learning methods, convolutional neural networks can automatically obtain the underlying feature information of the picture through continuous convolution and pooling, and the extracted features are far more than traditional machine learning methods. Substantial studies have proved that convolutional neural networks are more suitable for image recognition than the traditional machine learning methods [9–13, 29–33]. Consequently, Gao et al. [10] took the lead in using CNN to train the segmented Hollow CAPTCHAs. Their model obtained an 89% accuracy rate on the test set, and the fastest success recognition speed was 3 seconds per text-based CAPTCHA. Among incumbent deep learning-based attacks, the accuracy rate still depends heavily on the quality of the character segmentation. As text-based CAPTCHAs designed by researchers become more complex, traditional segmentation algorithms have been unable to completely separate each character. To tackle this deficiency, Ye et al. utilized a generative adversarial network (GAN) to automatically generate a large number of labeled text-based CAPTCHAs as a training set [7, 14, 15]. For some types of text-based CAPTCHAs, this method only needs to mark about 500 original text-based CAPTCHAs to obtain a recognition rate of more than 90%.

## 3. Our Approach

*3.1. The General Attacking Process.* As illustrated in Figure 3, to break text-based CAPTCHAs by using machine/deep learning methods, the general attacking process usually requires the following steps.

*3.1.1. Data Acquisition.* This step is used to acquire the original data set, which is practically achieved by crawling a large number of text-based CAPTCHAs from real-world websites. To ensure comprehensive data samples, the acquired text-based CAPTCHAs need to have characteristics including distortion, rotation, waving, and overlapping.

*3.1.2. Data Preprocessing.* Generally speaking, data preprocessing is used to filter the original data according to a dedicated filtering operation, thus improving the efficiency of the later training process. In this paper, we present a new preprocessing approach that can also achieve a high accuracy rate of the attacking process. The detailed method design is shown in Section 3.2.

*3.1.3. Model Training.* This step is using the machine/learning training methods to generate a prediction model for recognizing text-based CAPTCHAs. Herein, we choose five commonly used machine/deep learning methods: CNN, SVM, DT, RF, and LR. Notably, CNN is a popular deep learning algorithm that can provide a relatively high accuracy rate for identifying CAPTCHA images. Particularly, in our experiment, we put every 200 images as a training batch into CNN for training after binarization. Table 1 shows the architecture and parameters of the CNN used in our experiment. In addition, we also use four machine learning methods including SVM, DT, RF, and LR in the experiment. For these methods, researchers need to artificially search some character features for computers to learn according to the characteristics of different characters.

*3.1.4. Prediction Outputting.* A prediction model is generated after the model training process, which can be used to identify the unlabeled test set. Thereby, the prediction model can output a recognition result for each test CAPTCHA.

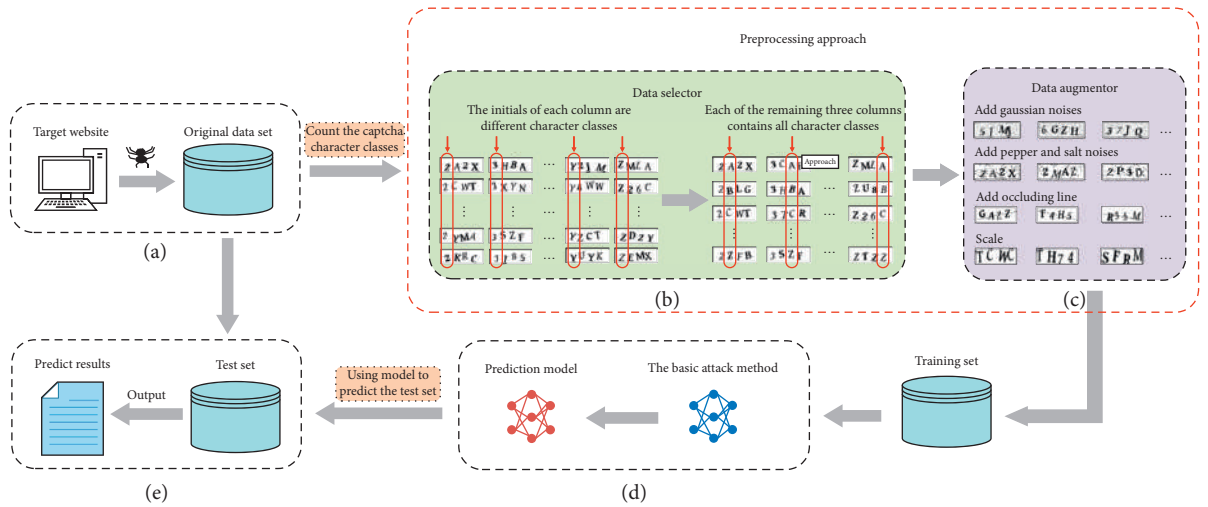



Figure 3: The general attacking process based on machine/deep learning methods to break text-based CAPTCHAs.

Table 1: Architecture of the training model.

| Type/stride | Filter shape | Input size | Output size |
|---|---|---|---|
| Conv2D_1/S1 | $3 \times 3 \times 32$ | $200 \times 60 \times 160 \times 1$ | $200 \times 58 \times 158 \times 32$ |
| MaxPool2D_1/S2 | $2 \times 2$ | $200 \times 58 \times 158 \times 32$ | $200 \times 29 \times 79 \times 32$ |
| Conv2D_2/S1 | $5 \times 5 \times 64$ | $200 \times 29 \times 79 \times 32$ | $200 \times 25 \times 75 \times 64$ |
| MaxPool2D_2/S2 | $2 \times 2$ | $200 \times 25 \times 75 \times 64$ | $200 \times 12 \times 37 \times 64$ |
| Conv2D_3/S1 | $5 \times 5 \times 128$ | $200 \times 12 \times 37 \times 64$ | $200 \times 8 \times 33 \times 128$ |
| MaxPool2D_3/S2 | $2 \times 2$ | $200 \times 8 \times 33 \times 128$ | $200 \times 4 \times 16 \times 128$ |
| Flatte | — | $200 \times 4 \times 16 \times 128$ | $200 \times 8192$ |
| Dense | — | $200 \times 8192$ | $200 \times 128$ |
| Reshape | — | $200 \times 128$ | $200 \times 4 \times 32$ |

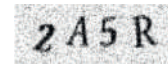
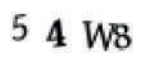
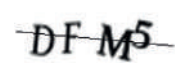
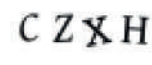
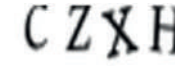
Then, we count whether the prediction result given by the prediction model is consistent with the characters on the corresponding CAPTCHA. Finally, we give the accuracy rate of the prediction model on the test set.

*3.2. Our Preprocessing Approach.* We can clearly see that, in a general attacking process, not all text-based CAPTCHAs in the original data samples can affect the final accuracy rate. This motivates us to choose only a small number of training data that have a significant influence on prediction results and can also lead to a high accuracy rate. To achieve this goal, we design a new preprocessing approach composed of two modules: (i) a data selector that invokes a set of regulations to filter out a high-quality text-based CAPTCHA training set and (ii) a data augmentor that is used to expand the data set and enhance the robustness of the prediction model.

*3.2.1. Data Selector.* We aim to design a data selector that can generate a clean and effective training set by filtering the original data set, thereby improving the prediction accuracy of the whole model. For the detailed design of the data selector, we formulate a set of regulations that can filter out a high-training quality training set of text-based CAPTCHAs. The filtering regulations of the data selector are also shown in Part B of Figure 3. First, we count the character classes on the original CAPTCHA image set. There are a total of 32 character classes: 2 ~ 9, A ~ Z except for "O" and "I". That is to say, each CAPTCHA on the website is composed of four characters (repeatable) that are randomly selected from the 32 character classes. In this case, each class of characters can appear with the probability of 1/32 in one bit of the text-based CAPTCHA string. Secondly, we divide the training set pictures into 32 columns in alphabetical order and arrange the images with the same first CAPTCHA character into one column. For the remaining three columns, each column must contain all character classes. This will ensure that 32 different characters in each position will be trained by the neural network.

*3.2.2. Data Augmentor.* Data augmentation is used to expand the data set, thus enhancing the robustness of the prediction model. It can also prevent the neural network from learning irrelevant features, thereby fundamentally improving the overall performance of the model. We propose a data augmentor using four different image noises such as Gaussian noise, pepper and salt noise, scale, and occluding line. Table 2 shows the comparison of the four types of noise images with the original image. In our experiment, the data augmentor randomly allocates an augmentation method for each picture or decides not to perform any processing. Then, the data augmentor automatically puts the noise-added images into CNN for

Table 2: Comparison after adding different noises.

| Scheme | Original example | Noised example |
|---|---|---|
| Gaussian noise |  |  |
| Pepper and salt noise |  |  |
| Occluding line |  |  |
| Scale |  |  |

training. The workflow of the data augmentor is illustrated in part C of Figure 3. Each image noise is described in detail as follows:

(i) Gaussian noise: Gaussian noise is named after Carl Friedrich Gauss because its probability density function (PDF) is equal to that of the normal distribution. In our experiment, we first use the `random.randn()` function in Python to randomly return an array with a standard normal distribution. Then, we multiply the array by the Gaussian noise probability to generate a noise matrix. Finally, we add the noise matrix to the matrix of the normal text-based CAPTCHA to generate a Gaussian noise image. The function of Gaussian noise can be expressed as

$$G = I + N_g, \tag{1}$$

where $G$ denotes the noisy image, $I$ denotes the unprocessed image, and $N_g$ represents the Gaussian noise with a variance $\sigma = 50$ and a mean value of 0.

(ii) Pepper and salt noise: in essence, both pepper and salt noises are some black and white pixels that randomly appear on images. Salt noise is also called white noise and pepper noise is also called black noise. Concretely, in our method, we randomly select some pixels on the image and change the pixels' RGB values to the $(255, 255, 255)$ or $(0, 0, 0)$ colour on the standard RGB colour comparison table. The function of pepper (or salt) noise can be expressed as

$$P = I + N_p, \tag{2}$$

where $P$ denotes the noisy image, $I$ means the unprocessed image, and $N_p$ represents the pepper (or salt) noise with a noise probability of 0.5.

(iii) Occluding line: we implement a method to generate an occluding line. The principle of this method is shown in Figure 4. The formula of this method is shown in equation (3). First, we assume that the coordinates of the left and right endpoints $A$ and $B$ of the occluding line are $A(x_1, x_2)$ and $B(x_2, y_2)$, respectively. Furthermore, we found that when $A_x \in (0, 10)$, $A_y \in (10, 50)$ and $B_x \in (150, 160)$, $B_y \in (10, 50)$, the occluding line can cross the four characters on the CAPTCHA image. Therefore, we use the function to generate the endpoints of the occluding line. Then, we call the function `draw.line()` to generate a line segment linking the two endpoints $A$ and $B$. The function of generating an occluding line is defined as

$$L = I + N_l, \tag{3}$$

where $L$ means the observed image of occluding line, $I$ denotes the unprocessed image, and $N_l$ represents the occluding line.

(iv) Scale: the fourth data augmentation method used in data augmentor is scale. We carefully analyze the characteristics of the original CAPTCHA images then proposed an image scale method. First, we translate a $60 \times 160$ size image to the left by 10 pixels and then upwards by 10 pixels. Secondly, we call the pan and zoom functions to adjust the image to $50 \times 140$. Finally, we call the resize function to restore the image to $60 \times 160$. It can be seen from the example in Table 2 that, compared with the original CAPTCHA image, the characters after enlarged still retain their basic character features. This method can effectively prevent the neural network from learning irrelevant or nonkey features of characters, fundamentally improving the accuracy of the model. In particular, we have

$$S = \Phi(I), \tag{4}$$

where $S$ denotes the observed image of occluding line, $\Phi$ means the scale operator, and $I$ represents the unprocessed image.

## 4. Experiments

In this section, we perform the complete attacking experiment by using our approach presented in Section 3. We particularly conduct a set of ablation experiments by applying our data-preprocessing approach in different training methods, including CNN and four commonly used machine learning methods.

### 4.1. Experiment Settings

*4.1.1. Data Preparation.* We used the crawled 9,955 unlabeled original text-based CAPTCHAs from E-ZPass New York website as the attack target of this experiment. The

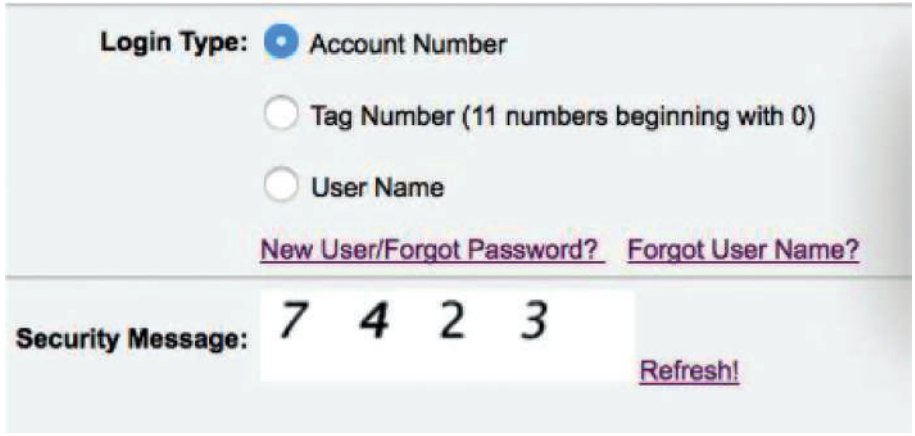



Figure 4: The text-based CAPTCHA of the E-ZPass New York login interface.

principle of this method is shown in Figure 5. The characteristics of the CAPTCHA of this website are as follows: no extra interference noise, the characters are distorted, and only two visual colours of white and black. These features significantly reduce our workload because it saves us the tedious preprocessing step of denoising. It is worth noting that similar characters may cause a negative impact on the success rate of recognition, such as "0" and "O", "I" and "1". To avoid this predicament, we manually delete the pictures containing these four characters in the web crawler data set. Therefore, the original CAPTCHA image set contains a total of 32 character classes: 2–9, A–Z except for "O" and "I".

*4.1.2. Implementation Details.* Our implementation environment is PyCharm 2020.2.2, the programming language is Python 3.7, and we test it on a laptop with 1.6 GHz Internet core i5-8250CPU, 4 GB RAM, and Windows $10 \times 64$. Meanwhile, we uniformly use `findContours()` function to segment training images, which is built-in OpenCV 4.1. For CNN, we use TensorFlow 2.0 framework to write the training code.

*4.1.3. Evaluation Metric.* To evaluate the quality of text-based CAPTCHA attacks, we use a metric of the prediction accuracy, that is, the accuracy rate ACC. A prediction model with a higher accuracy rate will have an accuracy value closer to 100%. We can use the following formula to calculate the accuracy rate:

$$\mathrm{ACC} = \frac{K}{T} \times 100\%, \tag{5}$$

where $K$ represents the number of text-based CAPTCHAs successfully identified by the prediction model on the test set and $T$ represents the size of the test set.

*4.2. Ablation Experiments.* In our experiment, we conduct a set of ablation experiments to verify the two preprocessing approaches we propose. Figure 6 reports the accuracy rate of each basic attack method in different preprocessing


Figure 5: Principle of adding the occluding line. The CAPTCHA image size is $60 \times 160$. The abscissa represents the length, the ordinate represents the width, and the shading area represents the area where the left and right endpoints A and B of the occluding line are generated by the method.

environments. In the benchmark experiment, we input a set of unprocessed data sets into five commonly used attack methods for training. Moreover, we prepare the same amount of labeled training images and the same basic attack method for each experiment to ensure the fairness of the experiment. The performance improvement shows that our preprocessing approach is advantageous in character recognition, especially when using two preprocessing modules simultaneously.

We first add the data selector and data augmentor modules separately. The accuracy rate of the CNN-based attack method increases by 2.61% and 0.66% compared to the benchmark experiment after adding data selector and data augmentor, respectively. This shows that even adding one preprocessing module can also be advantageous in recognition. When we add these two preprocessing modules to the CNN-based attack at the same time, the accuracy rate of this model is 95.5%, which is 4.51% higher than the case without any preprocessing module. Overall, the preprocessing approach can further improve the accuracy rate of the CNN-based attack method.

To verify the applicability of the preprocessing approach, we conduct a set of ablation experiments on four machine learning-based attack methods. We choose the four most commonly used traditional machine learning methods, including SVM, DT, RF, and LR as basic training models. Then,

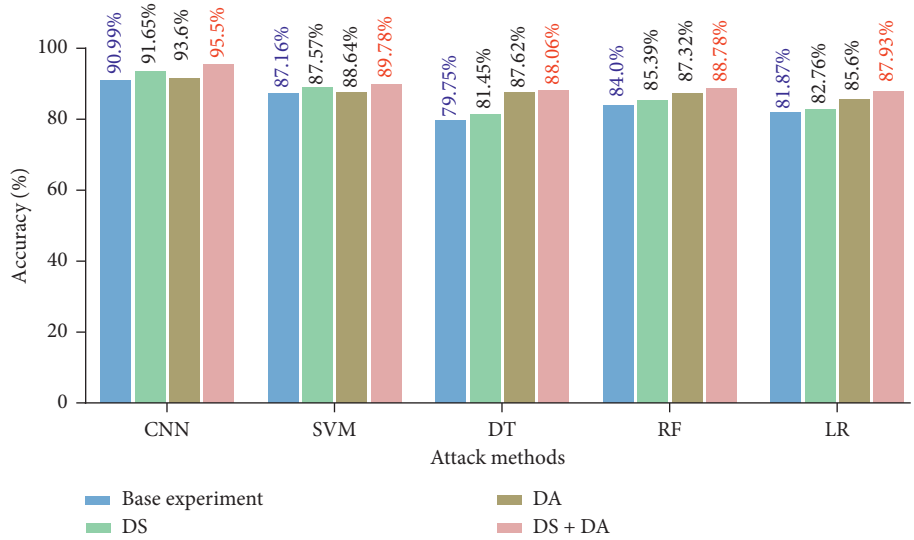



Figure 6: Combining our preprocessing approach with five commonly used attack methods, it shows that even adding a single preprocessing module can improve the accuracy of the basic attack method. Adding two attack modules at the same time can significantly improve the accuracy of each method. Furthermore, compared with machine learning methods, CNN has more advantages in text-based CAPTCHA recognition.

we do a set of ablation experiments on each of the machine learning methods to demonstrate the wide applicability of our proposed preprocessing modules. Each set of ablation experiments consists of a basic control group, an experiment adding data selector, an experiment adding data augmentor, and an experiment adding two preprocessing modules. The training sets in the basic control group contain 3,319 randomly marked original CAPTCHA images. For the remaining two experiments with adding a data selector, the module selects 3,319 CAPTCHAs according to the actual situation.

From Figure 6, we can see that after adding data selector (DS) and data augmentor (DA) modules to different attack methods, the accuracy rate has improved in varied degrees. In detail, after adding two preprocessing modules simultaneously, the accuracy rate for SVM increases from 87.16% to 89.78%, DT from 79.75% to 88.06%, RF from 84.0% to 88.78%, and LR from 81.87% to 87.93%. The extensive contrast experiments show that our preprocessing approach has strong applicability, which can be easily deployed to other attack methods.

Notably, in the five commonly used attack methods, the CNN attack method has the highest accuracy rate under the same amount of training images, even in the case without adding any preprocessing modules. The fundamental reason is that the advantage of CNN lies in feature extraction. The neural network can even extract some underlying features after multiple convolutions. In the case of large-scale training data, the deep neural network can extract more features than traditional machine learning methods. Since people usually use ImageNet or other large-scale data sets in current image recognition algorithms, many traditional methods such as SVM and DT are difficult to meet practical requirements in terms of computational complexity. Therefore, when the training set is large and complex, the effect of using a neural network with low computational complexity and high parallelism is better than traditional machine learning methods. The outstanding performance of CNN shows that using CNN to identify text-based CAPTCHA is more accurate than traditional machine learning methods.

## 5. Discussion

Reviewing the current research studies of CAPTCHAs, lots of efforts are made to improve the accuracy rate of attack methods, including varieties of machine/deep learning-based training models as well as our data-preprocessing approach. With such an increasing evolution of attack methods, the more sophisticated design of CAPTCHAs is required to enhance security. The sophisticated design, however, may also cause recognition difficulties of CAPTCHAs for humans and thus leads to a poor user experience. Toward this end, how to balance the design complexity of CAPTCHAs and the user experience is a significant research topic in the future [18, 34, 35].

Taking text-based CAPTCHAs as an example, we investigate a real user experience in terms of different security strengths of CAPTCHA designs. In particular, we perform a questionnaire survey for 20 participants who are required to identify the 500 text-based CAPTCHA images that are crawled from real-world websites. These CAPTCHA images have different security strengths according to different combinations of several basic image features including noise, overlapping, rotation, distortion, and waving. Then, participants evaluate the user experience of the CAPTCHAs by giving the usability score from 1 to 5. The statistical results of our questionnaire survey are shown in Table 3, in which we consider five combinations of image features to compare five different security strengths of CAPTCHA designs. For each combination of image

Table 3: User experience against five different security strengths of text-based CAPTCHAs.

| Image examples | Image features | | | | | Humans | Usability score |
|---|---|---|---|---|---|---|---|
| | Noises | Overlapping | Rotation | Distortion | Waving | Accuracy rate (%) | |
|  | | | √ | | √ | 99 | 4.8 |
|  | | | √ | √ | √ | 93 | 3.7 |
|  | | √ | √ | √ | √ | 92 | 3.4 |
|  | √ | √ | √ | | √ | 85 | 3 |
|  | √ | √ | √ | √ | √ | 33 | 2 |

features, we count the average accuracy rate and the average usability rating given by all participants.

In Table 3, we can clearly observe that the difficulty of human identification is increasing with the increasing combination of the image features. For the CAPTCHA including only two image features (i.e., rotation and waving), it has the lowest security strength, thereby the average accuracy rate of humans can reach 99%. However, the average accuracy rate of humans is only 66% when the CAPTCHA has five image features, meaning that around one-third of users would fail identification. In addition, as the used image features increase, the usability score given by the participants gets lower and lower. This is because although complex CAPTCHA images can better prevent malicious bot attacks, it also increases the difficulty of human identification and deteriorates the user experience. All in all, to address the unbalanced problem between the security strengths and user experience, the future work is required to design a new type of text-based CAPTCHA that can effectively resist robot attacks while also being user-friendly.

## 6. Conclusion

As the most widely used CAPTCHA scheme, text-based CAPTCHA is an important security mechanism to distinguish between humans and computers. However, our attack poses a threat to real-world text-based CAPTCHA. We present a new preprocessing approach for text-based CAPTCHA attacks. The preprocessing approach includes two modules: data selector and data augmentor. First of all, the data selector module automatically selects some data with training significance as the training set in the original data set according to the rules set by us. Secondly, the data augmentor module increases the amount of training data to increase the robustness of the model. We successfully deployed this preprocessing approach on five different commonly used attack methods, and the accuracy rate was significantly improved. The key advantage of our preprocessing approach is that it requires less human involvement and for any basic attack method, by adding this preprocessing step, a higher accuracy rate can be achieved with relatively less training data for any basic attack method. Finally, we further discussed the existing problems in the current text-based CAPTCHAs design. Our work can also be applied to many practical scenarios [20, 22, 24], such as ID card character recognition, bank card number recognition, and handwritten numeral recognition. How to expand to the application field is also a part of our ongoing work.

## Data Availability

Data are available on request to the corresponding author.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## References

[1] L. Von Ahn, M. Blum, N. J. Hopper, and J. Langford, "CAPTCHA: using hard AI problems for security," in *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 294–311, Springer, Wersaw, Poland, May 2003.

[2] L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum, "Recaptcha: human-based character recognition via web security measures," *Science*, vol. 321, no. 5895, pp. 1465–1468, 2008.

[3] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. M. Voelker, and S. Savage, "Re: captchas-understanding captcha-solving services in an economic context," in *Proceedings of the USENIX Security Symposium*, vol. 10, p. 3, Washington, DC, USA, August 2010.

[4] M. Shirali-Shahreza and S. Shirali-Shahreza, "Captcha for blind people," in *Proceedings of the 2007 IEEE International Symposium on Signal Processing and Information Technology*, pp. 995–998, IEEE, Cairo, Egypt, December 2007.

[5] J. Lazar, J. Feng, T. Brooks et al., "The soundsright captcha: an improved approach to audio human interaction proofs for blind users," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2267–2276, Austin, TX, USA, May 2012.

[6] V. P. Singh and P. Pal, "Survey of different types of captcha," *International Journal of Computer Science and Information Technologies*, vol. 5, no. 2, pp. 2242–2245, 2014.

[7] G. Ye, Z. Tang, D. Fang et al., "Yet another text captcha solver: a generative adversarial network based approach," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 332–348, Toronto, Canada, October 2018.

[8] S. Tian and T. Xiong, "A generic solver combining unsupervised learning and representation learning for breaking text-based captchas," in *Proceedings of the Web Conference*, pp. 860–871, Taipei, Taiwan, 2020.

[9] S. Sachdev, "Breaking captcha characters using multi-task learning cnn and svm," in *Proceedings of the 2020 4th International Conference on Computational Intelligence and Networks (CINE)*, pp. 1–6, IEEE, Kolkata, India, February 2020.

[10] H. Gao, W. Wang, J. Qi, X. Wang, X. Liu, and J. Yan, "The robustness of hollow captchas," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, pp. 1075–1086, Berlin, Germany, November 2013.

[11] J. Yan and A. S. El Ahmad, "A low-cost attack on a microsoft captcha," in *Proceedings of the 15th ACM Conference on Computer and Communications Security*, pp. 543–554, Virginia, VA, USA, October 2008.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[13] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud, and V. Shet, "Multi-digit number recognition from street view imagery using deep convolutional neural networks," 2013, https://arxiv.org/abs/1312.6082.

[14] X.-W. Chen and X. Lin, "Big data deep learning: challenges and perspectives," *IEEE Access*, vol. 2, pp. 514–525, 2014.

[15] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *Journal of Big Data*, vol. 2, no. 1, p. 1, 2015.

[16] T. Chen, X. Li, X. Luo, and X. Zhang, "Under-optimized smart contracts devour your money," in *Proccedings of the 2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pp. 442–446, IEEE, Austria, TX, USA, February 2017.

[17] T. Chen, Z. Li, Y. Zhu et al., "Understanding ethereum via graph analysis," *ACM Transactions on Internet Technology*, vol. 20, no. 2, pp. 1–32, 2020.

[18] E. Bursztein, M. Martin, and J. Mitchell, "Text-based captcha strengths and weaknesses," in *Proceedings of the 18th ACM Conference on Computer and Communications Security*, pp. 125–138, Illinois, IL, USA, October 2011.

[19] T. Chen, Y. Zhang, Z. Li et al., "Tokenscope: automatically detecting inconsistent behaviors of cryptocurrency tokens in ethereum," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1503–1520, London, UK, November 2019.

[20] A. K. Sangaiah, D. V. Medhane, T. Han, M. S. Hossain, and G. Muhammad, "Enforcing position-based confidentiality with machine learning paradigm through mobile edge computing in real-time industrial informatics," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4189–4196, 2019.

[21] X. Zhou, W. Liang, S. Shimizu, J. Ma, and Q. Jin, "Siamese neural network based few-shot learning for anomaly detection in industrial cyber-physical systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5790–5798, 2020.

[22] A. K. Sangaiah, D. V. Medhane, G.-B. Bian, A. Ghoneim, M. Alrashoud, and M. S. Hossain, "Energy-aware green adversary model for cyberphysical security in industrial system," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 3322–3329, 2019.

[23] W. Liang, L. Xiao, K. Zhang, M. Tang, D. He, and K.-C. Li, "Data fusion approach for collaborative anomaly intrusion detection in blockchain-based systems," *IEEE Internet of Things Journal*, 2021.

[24] A. K. Sangaiah, A. S. Rostami, A. A. R. Hosseinabadi et al., "Energy-aware geographic routing for real time workforce monitoring in industrial informatics," *IEEE Internet of Things Journal*, 2021.

[25] X. Zhou, Y. Hu, W. Liang, J. Ma, and Q. Jin, "Variational lstm enhanced anomaly detection for industrial big data," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 5, pp. 3469–3477, 2020.

[26] G. Mori and J. Malik, "Recognizing objects in adversarial clutter: breaking a visual captcha," in *Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, IEEE, Madison, WI, USA, June 2003.

[27] Y. Zi, H. Gao, Z. Cheng, and Y. Liu, "An end-to-end attack on text captchas," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 753–766, 2019.

[28] O. Bostik and J. Klecka, "Recognition of captcha characters by supervised machine learning algorithms," *IFAC-PapersOnLine*, vol. 51, no. 6, pp. 208–213, 2018.

[29] M. Tang, H. Gao, Y. Zhang, Y. Liu, P. Zhang, and P. Wang, "Research on deep learning techniques in breaking text-based captchas and designing image-based captcha," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, pp. 2522–2537, 2018.

[30] A. Dionysiou and E. Athanasopoulos, "Sok: Machine vs. machine–a systematic classification of automated machine learning-based captcha solvers," *Computers & Security*, vol. 97, Article ID 101947, 2020.

[31] J. Yan, "A simple generic attack on text captchas," in *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, USA, February 2016.

[32] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.

[34] J. Yan and A. S. El Ahmad, "Usability of captchas or usability issues in captcha design," in *Proceedings of the 4th Symposium on Usable Privacy and Security*, pp. 44–52, Pittsburgh, PA, USA, August 2008.

[35] C. A. Fidas, A. G. Voyiatzis, and N. M. Avouris, "On the necessity of user-friendly captcha," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2623–2626, Vancouver, BC, Canada, May 2011.