

Fighting Attacks on Large Character Set CAPTCHAs Using Transferable Adversarial Examples

Yucheng Fu, Guoheng Sun, Han Yang, Juntian Huang, Haizhou Wang*

School of Cyber Science and Engineering, Sichuan University

Chengdu, China

{fuyucheng, sunguoheng, yanghan, huangjuntian}@stu.scu.edu.cn; whzh.nc@scu.edu.cn

Abstract—Over a long period, large character set CAPTCHAs are widely used to defend against automated attack programs on the Internet. However, with the development of deep learning techniques, some attacks for large character set CAPTCHAs have been proposed, proving that they are no longer secure. To defend against black-box attacks on these CAPTCHAs, we propose a novel defense method based on transferable adversarial example techniques. On the one hand, we defend against character recognition attacks by adding adversarial perturbations to the characters of CAPTCHAs combining three strategies: *Gradient-based Attacks*, *Input Transformations* and *Attention Mechanism*. On the other hand, we defend against character detection attacks by leveraging an ensemble method to generate adversarial perturbations on the background of CAPTCHAs. To the best of our knowledge, this is the first study to improve the security of large character set CAPTCHAs against black-box attacks based on transferable adversarial example techniques. Using the eight most popular Chinese CAPTCHA schemes as examples, we conduct comprehensive experiments. Results show that our method improves the security of large character set CAPTCHAs by making the average success rate of black-box attacks significantly drop from 53.33% to 3.49%. Overall, our method can be helpful to the design of more secure large character set CAPTCHAs.

Index Terms—CAPTCHA, Adversarial Examples, Deep Learning, Large Character Set

I. INTRODUCTION

CAPTCHA (Completely Automated Public Turing Test to Tell Computers and Humans Apart) is widely used to distinguish between humans and automated computer programs. It's deployed on a large number of websites to fight against malicious behaviours on the Internet, including using web crawlers to collect information [1] and registering social bot accounts [2]. Currently, CAPTCHA schemes can be roughly classified into four categories: Text-based CAPTCHA [3], Image-based CAPTCHA [3], Audio-based CAPTCHA [4] and Behavior-based CAPTCHA [5]. Despite the availability of many advanced CAPTCHA schemes, Text-based CAPTCHAs remain the most popular due to their low implementation cost and good user experience by far.

A. Large Character Set CAPTCHAs

Unfortunately, a few works have cracked traditional Text-based CAPTCHAs composed only of English alphabetical

characters and Arabic numerals. Due to the small classification space, they can be easily cracked by well-trained deep learning models [3], [6], [7]. To address this issue, Text-based CAPTCHAs with larger character sets are designed [8], [9]. These CAPTCHAs use verification characters like Chinese, Japanese, etc, which significantly increase the number of classification targets for fighting against malicious attack models.

Although large character set CAPTCHAs are more secure than traditional Text-based CAPTCHAs, they are facing challenges from a growing number of attack threats. Using Chinese CAPTCHAs as examples, some works have performed attacks to analyze their security [7]–[10]. Meanwhile, there have been some works designing more secure Chinese CAPTCHA schemes by increasing the complexity of CAPTCHAs [8], [9], [11], [12], which are harmful to user experience, i.e. usability. However, with the development of deep learning techniques, these defenses have also been cracked by more advanced methods [7], [13]. Therefore, it is significant to propose new defense solutions for large character set CAPTCHAs.

To this end, we propose a defense method for large character set CAPTCHAs based on adversarial example techniques [14]. Instead of designing more complex CAPTCHAs, we add adversarial perturbations to improve the security of existing Input-based and Click-based large character set CAPTCHAs [9] while preserving their usability. Furthermore, our method is not limited to large character set CAPTCHAs. It also has the potential to be used for the design of Click-based CAPTCHAs with icons or images [7].

B. Challenges

Currently, there are three main challenges in enhancing the security of large character set CAPTCHAs:

1) The first challenge is how to enhance security without increasing the complexity of CAPTCHAs. To defend against the growing attacks on large character set CAPTCHAs, current methods try to make them more complex to increase the difficulty of cracking [8], [9], [12]. However, once attackers obtain high-quality datasets and build effective models, these complicated CAPTCHAs can still be cracked easily [9], [13]. Besides, increasing the complexity of CAPTCHAs usually decreases usability, which makes them hard for users to recognize but easy for automated programs to crack.

* Corresponding Author.

2) The second challenge is how to prevent attackers from locating the characters in CAPTCHAs. In order to crack large character set CAPTCHAs, attackers usually utilize advanced object detection models to locate characters in the CAPTCHAs [7], [9], [13]. However, to improve security, existing adversarial example techniques applied to CAPTCHAs only prevent attackers from recognizing characters or images [11], [15]–[17]. The perturbation added to characters can not prevent them from being located. There is still a lack of research on preventing attackers from locating objects in CAPTCHAs using adversarial example techniques.

3) The third challenge is how to defend against multiple black-box attacks effectively. It is hard to know the specific structures and parameters of models employed by the attackers in practice. The uncertainty of this black-box attack setting imposes a challenge on the implementation of defense based on adversarial example techniques. Therefore, it is necessary to improve the transferability of adversarial examples to defend against various black-box models.

C. Contributions

To address the above challenges, we propose an innovative defense method for large character set CAPTCHAs based on adversarial example techniques composed of two phases: *Dataset and Model Preparation*, *Adversarial CAPTCHA Generation*. In the *Dataset and Model Preparation* phase, we use web crawlers and synthetic CAPTCHA generator [9] to obtain the eight most popular Chinese CAPTCHA schemes as examples to construct datasets. Then, we construct and train various character detection and recognition models, which will be used to generate and evaluate adversarial CAPTCHAs. In the *Adversarial CAPTCHA Generation* phase, we propose a novel algorithm combining *Gradient-based Attacks*, *Input Transformations* and *Attention Mechanism*, namely, M-VNI-CT-FGSM, to generate transferable adversarial perturbations on the characters against recognition models. To defend against attacks from character detection models, we leverage an ensemble method, SVRE-MI-FGSM [18] combining multiple loss functions of various object detection architectures. It enables the transferability of adversarial CAPTCHAs between multiple object detection architectures and feature extraction modules. Eventually, the generated adversarial CAPTCHAs can effectively defend against multiple black-box attacks.

The source code of our defense method is now available¹. The main contributions of our work are summarized below:

- **We are the first to propose a defense method for large character set CAPTCHAs using adversarial example techniques to defend black-box attacks.** Instead of increasing the complexity of large character set CAPTCHAs, we add a limited amount of adversarial perturbations to CAPTCHAs to improve security while maintaining usability. Using Chinese CAPTCHAs as examples, we generate adversarial CAPTCHAs and conduct various experiments. Results show that our method

significantly improves the security of large character set CAPTCHAs by defending multiple black-box attacks.

- **We are the first to employ adversarial example techniques to defend against object detection attacks on CAPTCHAs.** Specifically, we leverage an ensemble method combining multiple loss functions of various object detection architectures to generate adversarial CAPTCHAs, which can prevent attackers from locating the characters in CAPTCHAs. Experimental results show that the performance of various character detection decreases when attacking our CAPTCHAs with adversarial perturbations in the backgrounds.
- **We construct a novel method with advanced strategies to generate transferable adversarial CAPTCHA.** We defend against black-box attacks from character recognition models by integrating *Gradient-based Attacks*, *Input Transformations* and *Attention Mechanism*. Specifically, we leverage *Gradient-based Attacks* to prevent local optimal and perform *Input Transformations* to augment the inputs of the algorithm. Moreover, *Attention Mechanism* is designed to perturb the fields that the models care about. To defend against black-box attacks from character detection models, we perform stochastic variance reduced ensemble generation using white-box models with different architectures. Experimental results show that our adversarial CAPTCHAs have excellent transferability to multiple black-box attacks.

II. RELATED WORK

A. Attacks for Large Character Set CAPTCHAs

Some works have analyzed the security of large character set CAPTCHAs with various attack methods [3], [7]–[10]. Algwil et al. [8] analyzed Chinese CAPTCHAs for the first time. After that, Lin et al. [10] used CNN-based methods to recognize Chinese CAPTCHAs, achieving an accuracy of 95.0% for individual characters. Tang et al. [3] used deep learning techniques to crack three real-world Chinese CAPTCHAs. With the development of deep learning techniques, Wang et al. [9] conducted the first comprehensive study on the security of real-world large character set CAPTCHAs. They cracked 11 real-world Chinese CAPTCHAs with success rates ranging from 34.7% to 86.9%, which demonstrates that real-world large character set CAPTCHAs are no longer secure. Nian et al. [7] integrated feature extraction, character detection and recognition into an end-to-end model and attacked two Chinese Click-based CAPTCHAs, obtaining success rates of 99.4% and 96.0%. The research results above show that attackers have been able to crack large character set CAPTCHAs with high success rates. Although these works have exposed the vulnerability of large character set CAPTCHAs, they are still popular and serving billions of users every day [9].

B. Defense for Large Character Set CAPTCHAs

To enhance the security of large character set CAPTCHAs, some defense methods have been proposed [8], [9], [12].

¹<https://github.com/17suidas/Defense-for-LCSCaptcha/>

Algwil et al. [8] suggested that Chinese CAPTCHAs should have segmentation resistance to improve security. However, segmentation-resistant CAPTCHAs have already been cracked by end-to-end methods [9] and object detection methods [7]. Cheng et al. [12] proposed a Chinese CAPTCHA generation method using neural style transfer techniques to generate CAPTCHAs with a consistent style between characters and background, which increases the difficulty of character detection and recognition. However, Wang et al. [9] demonstrated that a well-trained model can still successfully crack such CAPTCHAs. Wang et al. [9] proposed a 3D Chinese CAPTCHA scheme based on visual reasoning and incorporated semantic information. However, Gao et al. [13] have already cracked such 3D visual reasoning CAPTCHA containing Chinese. To conclude, the majority of existing defense methods for large character set CAPTCHAs either make the CAPTCHAs more complex or add additional verification tasks for users, which may greatly decrease usability. Moreover, these defense methods were soon cracked by well-designed attacks [7], [9], [13].

C. Defense Using Adversarial CAPTCHAs

Recently, researchers have proposed to use adversarial example techniques to improve the security of CAPTCHAs [4], [11], [15]–[17]. These methods added human-imperceptible perturbations to CAPTCHAs and increase the security of CAPTCHAs without significantly decreasing usability. Osadchy et al. [15] added perturbation to images to generate adversarial CAPTCHAs. Shi et al. [16] designed Image-based and Text-based Adversarial CAPTCHAs. Shi et al. [17] studied the deployment of Text-based adversarial CAPTCHAs on a large-scale online platform. Wang et al. [4] designed Audio-based adversarial CAPTCHAs, which are friendly to people with visual impairment. Zhang et al. [11] used FGSM [14] to add adversarial perturbations to Chinese CAPTCHAs. Their method is for the white-box setting and only considers the character recognition attack, which may not be directly applied to real-world scenarios with higher security requirements.

Compared to the above works, we are the first to generate highly transferable adversarial CAPTCHAs for large character set CAPTCHAs by applying advanced adversarial example generation methods. Furthermore, our transferable adversarial CAPTCHAs can defend against both character detection and recognition attacks.

III. METHODOLOGY

Using the Chinese CAPTCHAs as examples, we introduce a defense method based on adversarial example techniques. It consists of two phases: *Dataset and Model Preparation*, *Adversarial CAPTCHA Generation*. The framework of our proposed defense method is shown in Fig. 1.

A. Dataset and Model Preparation

In order to prepare for the generation and evaluation of adversarial CAPTCHAs, we use various Chinese large character set CAPTCHAs to train attack models and construct test sets.

TABLE I
OVERVIEW OF DATASET

Scheme	Dataset for Recognition Model		Dataset for Detection Model		Total Size of Character Set
	Training Set	Test Set	Training Set	Test Set	
Yidun	786,089	4,474	1,000	1,000	1,954
Dajie	736,200	6,000	1,000	1,000	3,651
YY	284,600	6,000	1,000	1,000	1,393
Baidu	251,800	5,000	1,000	1,000	1,234
Dingxiang	272,331	4,726	1,000	1,000	1,339
Renmin	100,200	2,000	1,000	1,000	491
Sougou	3,280	3,281	1,000	1,000	20
Geetest	220,990	2,954	1,000	1,000	1,090

Yidun website: dun.163.com, Dajie website: dajie.com, YY website: yy.com, Baidu website: pan.baidu.com, Dingxiang website: dingxiang-inc.com, Renmin website: people.com.cn, Sougou website: sougou.com, Geetest website: geetest.com.

Due to the lack of public datasets for real-world large character set CAPTCHAs, we developed web crawlers to collect Chinese CAPTCHAs on the Internet, including eight popular schemes: Yidun, Dajie, YY, Baidu, Dingxiang, Renmin, Sougou and Geetest, where Renmin and Sougou are Input-based CAPTCHAs and others are Click-based. All CAPTCHAs were collected from June to July 2022. Next, the labelling works for the locations and classes of characters in CAPTCHAs were carried out by four researchers. For each scheme, we obtained and 2,000 labelled CAPTCHAs. We use 1,000 of them to train models. The other 1,000 of them are used as the test set to generate adversarial CAPTCHA and evaluate the performance of models. Note that for character recognition models, we extract the characters in these CAPTCHAs as training and test set. Detailed information including the size of the character set as well as the size of datasets for character recognition and detection models are shown in TABLE I.

We construct and train four widely used character recognition models for each CAPTCHA scheme respectively. They are Inception-ResNet-v2 (IncRes-v2) [19], Inception-v3 (Inc-v3) [20], ResNet-50 (Res-50) [21] and VGG-16 [22]. In order to train them, we leverage the CAPTCHAs generator in [9] to generate synthetic characters. For each character class, we guarantee that the same number of characters are generated. Eventually, synthetic characters and characters in labelled CAPTCHAs for training are mixed to get the training set.

We construct and train six widely used character detection models with various architectures and backbones (feature extraction modules) using the mixed dataset of all the CAPTCHAs. They are Faster-RCNN [23] with ResNet-101, ResNet-50 and VGG-16 backbones, i.e. FRCNN (Res-101), FRCNN (Res-50) and FRCNN (VGG-16), SSD [24] with MobileNet-v2 and VGG-16 backbones, i.e. SSD (Mb-v2) and SSD (VGG-16), YOLO-v5 [25]. Note that detection models give the location of characters, and class label is only ‘character’. Since the character detection task is relatively simple, we do not expand the dataset. In contrast, we mix all the labelled CAPTCHAs used for training to construct the training set with a total of 8,000 CAPTCHAs.

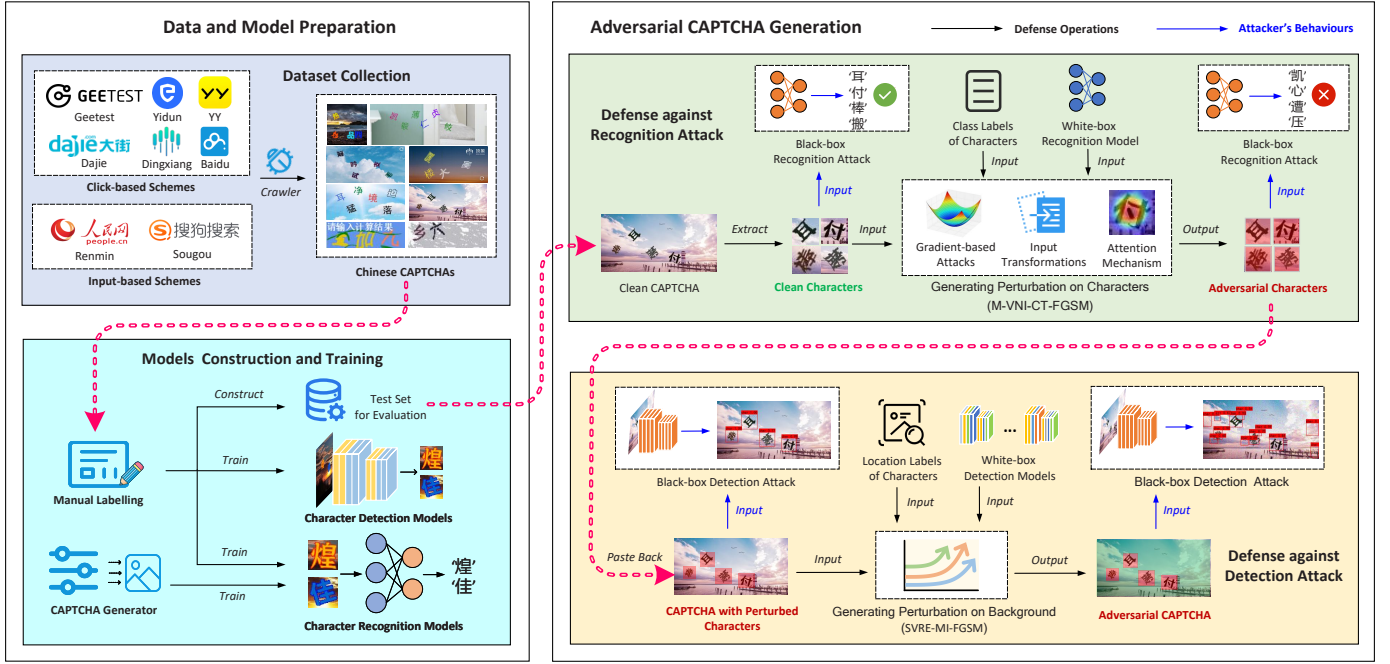


Fig. 1. Framework of Proposed Defense Method

We also reproduce two state-of-the-art attacks for large character set CAPTCHAs, which construct the Attention-based Model [9] and the End-to-end Model based on Mask R-CNN [7] respectively for evaluation.

B. Adversarial CAPTCHAs Generation

1) *Defense against Recognition Attacks*: In this phase, we use the character as input image x and its character class as label y . For each CAPTCHA scheme, we use a character recognition model as the white-box model f to generate the adversarial character. The output is the adversarial character x^{adv} . In order to constrain the maximum perturbation on each pixel, we use L_∞ norm method, which guarantees $\|x - x^{adv}\|_\infty \leq \epsilon$ and misleads the model, i.e. $f(x^{adv}; \theta) \neq y$ by maximizing model's loss function $J(x^{adv}, y; \theta)$.

Here we use IncRes-v2 as the white-box model f since it integrates residual connection and inception block, which are widely used feature extraction networks [19]. Moreover, IncRes-v2 shows good transferability in various works [26]–[28]. Therefore, the adversarial characters generated by IncRes-v2 can be highly transferable. Then, we integrate three strategies including *Gradient-based Attacks* [28], [29], *Input Transformations* [29]–[31], *Attention Mechanism* [26], [32] with the basic FGSM [14] to construct our adversarial character generation algorithm, namely, M-VNI-CT-FGSM.

Gradient-based Attacks. One popular class of strategies to improve the transferability of the adversarial examples is perturbing the input with well-calculated gradient information [28], [29] to alleviate local optima. Here we leverage two Gradient-based Attacks, i.e. *Gradient Variance Tuning Fast Gradient Sign Method* [28] and *Nesterov Iterative Fast Gradient Sign Method* [29] to fine-tune the direction of gradients,

i.e. VNI-FGSM. With the initialization of gradient $g_0 = 0$ and gradient variance $v_0 = 0$, we iteratively update the x_t^{adv} . Let μ be the decay factor of momentum and α be the perturbation step size. In each iteration, x_t^{adv} will first be updated by $x_t^{nes} = x_t^{adv} + \alpha \cdot \mu \cdot g_t$ [29]. Next, the gradient $\nabla_{x_t^{nes}} J(x_t^{nes}, y; \theta)$ and the gradient variance v_{t+1} obtained previously are added up [28]. The updated g_{t+1} is obtained by adopting momentum [26] as

$$g_{t+1} = \alpha \cdot \mu \cdot g_t + \frac{\nabla_{x_t^{nes}} J(x_t^{nes}, y; \theta) + v_t}{\|\nabla_{x_t^{nes}} J(x_t^{nes}, y; \theta) + v_t\|_1}, \quad (1)$$

then the gradient variance v_t for next iteration is updated as

$$v_{t+1} = \frac{1}{N} \sum_{i=1}^N \nabla_{x_{t,i}^{adv}} J(x_{t,i}^{adv}, y; \theta) - \nabla_x J(x, y; \theta), \quad (2)$$

where N is the number of sampled neighbour points of $x_{t,i}^{adv}$, $x_{t,i}^{adv} = x_t^{adv} + r_i$ is the i -th sampled point, and $r_i \sim U[-\beta \cdot \epsilon, \beta \cdot \epsilon]$ consists of uniformly distributed values and has the same shape as x_t^{adv} , β is used to control sampling range.

Input Transformations. Another class of strategy we considered to improve the transferability of the adversarial examples is Input Transformations, which increases the diversity of input image to make the direction of optimization more generalized [29]–[31]. In the generation process of adversarial characters, we perform Composite Transformation (CT) combining three types of Input Transformations. 1) *Diverse Input Method* [30]: Before being input to the gradient calculation, the image x will be performed random resizing and padding transformation with a probability p to alleviate overfitting, which is denoted as $D(x; p)$; 2) *Translation-Invariant Method* [31]: This strategy makes translation-based data augmentation

by using a kernel matrix W to convolve with the gradient g , which is denoted as $T(g) = W * g$; 3) *Scale-Invariant Method* [29]: It increases the input diversity by performing the Scale-Invariant transformation to input x . Thus x is expanded to a set of m scaled copies for further gradient calculation. This process is denoted as $S(x; m) = \{x/2^0, x/2^1, \dots, x/2^{m-1}\}$.

Attention Mechanism. We construct a matrix *mask* (M) to determine the perturbation updating magnitude in each pixel according to the attention of model. Specifically, visualized observations in [27], [31] indicate that models with different structures and weights would have similar Class Activation Maps (CAM) [32], [33] for the correctly classified images. Inspired by this, we believe that attackers' models also capture shared features for the ground true character class y . Thus, we propose an attention mechanism based on Gradient-weighted Class Activation Mapping (Grad-CAM) [32] to perturb characters. Firstly, the spatially pooled gradients are calculated as

$$a_k^c[y] = \frac{1}{Z} \sum_m \sum_n \frac{\partial f(x)[y]}{\partial A_k^c[m, n]}, \quad (3)$$

where A_k^c is the c -th feature map in the k -th layer, Z is a factor to normalize $a_k^c \in [-1, 1]$, $a_k^c[y]$ is the attention weight of c -th feature map in k -th convolution layer for class y . Next, instead of using ReLU function to remove the negative points [32], we take the absolute value of the weighted sum $\sum_c a_k^c[y] \cdot A_k^c$ to keep values in the areas with negative contributions to the correct prediction. We do this because both positive and negative points in $\sum_c a_k^c[y] \cdot A_k^c$ can affect the prediction probability of ground true class y . Moreover, the larger the absolute value of the points in $\sum_c a_k^c[y] \cdot A_k^c$, the more they affect the prediction. We believe that negative points are also meaningful since increasing them can reduce the correct prediction probability. Thus, we take the absolute value of points in $\sum_c a_k^c[y] \cdot A_k^c$ and let our FGSM-based algorithm determine the update direction to sufficiently utilize the contribution of the feature map. Then, $\sum_c a_k^c[y] \cdot A_k^c$ is normalized to $[0, 1]$ and resized to the same shape as the input x by (4). According to the translation-invariance property of Convolutional Neural Network (CNN), the resized feature maps can represent the attention weights in each area.

$$\hat{H}_k^t = \text{Resize}(\text{Norm}(|\sum_c a_k^c[y] \cdot A_k^c|)). \quad (4)$$

We then construct a *mask* (M) based on \hat{H}_k^t and α , which is a matrix to determine the perturbation magnitude of each pixel that can be updated in each iteration during optimization.

$$\text{mask} = \alpha + \hat{H}_k^t \cdot \gamma, \quad (5)$$

where $\alpha = \epsilon/T$, ϵ is the maximum perturbation, T is the number of iterations, and γ is the parameter used to enlarge the step size. *mask* enables the pixels with more attention weights to be updated with larger step sizes. Note that we still clip x^{adv} to bound it with $\|x - x^{adv}\|_\infty < \epsilon$.

The adversarial character generation algorithm M-VNI-CT-FGSM with the above strategies is shown in Algorithm 1.

Algorithm 1 M-VNI-CT-FGSM

Input: Character classifier f with parameters θ , category cross-entropy loss function J

Input: Clean character image x and label y

Input: Maximum perturbations ϵ , additional perturbations within each iteration γ , number of iterations T , momentum decay factor μ , factor β to control sampling range, number of samples N for variance calculation, translation invariance kernel W , number of scaling invariance samples m , diverse input probability p

Output: An adversarial example of character image x_T^{adv}

- 1: $a = \epsilon/T$
 - 2: $g_0 = 0; v_0 = 0; x_0^{adv} = x$
 - 3: Derive *mask* by (3),(4),(5)
 - 4: **for** $t = 0$ to $T - 1$ **do**
 - 5: $x_t^{nes} = x_t^{adv} + a \cdot \mu \cdot g_t$
 - 6: Scale x_t^{nes} to a set of images $x_t^s = S(x_t^{nes}; m)$
 - 7: Random transformation $x_t^d = D(x_t^s; p)$
 - 8: Update gradient $g_{t+1} = \mu \cdot g_t + \frac{W * (\nabla_{x_t^d} J(x_t^d, y; \theta) + v_t)}{\|W * (\nabla_{x_t^d} J(x_t^d, y; \theta) + v_t)\|_1}$
 - 9: Update v_{t+1} by (2)
 - 10: Update x_{t+1}^{adv} with gradient sign and *mask*, then clip x_{t+1}^{adv} by ϵ $x_{t+1}^{adv} = \text{Clip}_{x, \epsilon}(x_t^{adv} + \text{mask} \cdot \text{sign}(g_{t+1}))$
 - 11: **end for**
 - 12: **return** x_T^{adv}
-

2) *Defense against Detection Attacks:* In the real world, the detection models used by attackers may have different object detection architectures, for example, they might use one-stage object detectors like YOLO [30] or two-stage object detectors like Faster-RCNN [28]. However, it is difficult to generate adversarial examples that can transfer among multiple detection architectures. To address this challenge, we combine the loss functions of multiple white-box models and use an effective ensemble adversarial example generation method based on stochastic gradient variance reduction [18]. It can achieve more generalized optimization by considering the gradient direction of multiple loss functions.

Instead of generating adversarial examples against object recognition models [18], we perform ensemble generation against object detection models f_1, f_2, \dots, f_n with various object detection architectures. We optimize the weighted sum of loss functions as (6) to generate adversarial perturbation.

$$J_{ens}(x, y; \theta, w) = \sum_{i=0}^n w_i J_i(x, y; \theta_i), \quad (6)$$

where J_i is the loss function of model f_i with parameters θ_i , w_i is the weight of f_i , x is the input CAPTCHA, y is the location label. In practice, we can tune w_i for each model f_i to adjust the defense effectiveness against white-box models. We use the SVRE-MI-FGSM in [18] to generate adversarial CAPTCHA by modifying its loss function to the $J_{ens}(x, y; \theta, w)$ in (6). The algorithm can search the direction to increase the loss function of multiple object detection

architecture, which enables the transferability among models with various model architectures. Here we use FRCNN (Res-101), SSD (Mb-v2) and YOLO-v5 as white box models f_1, f_2, f_3 since they represent three of the most popular object detection architecture [34].

Note that in practice, we do not update perturbations in the areas containing characters to avoid confusion. The experiments in Section IV-B and IV-C also show that without perturbations against detection models in the character, our method can still show good effectiveness against detection attacks. Ultimately, after adding adversarial perturbations against detection models, the adversarial CAPTCHA can already mislead multiple character recognition and detection models.

IV. EXPERIMENTS

A. Experimental Setup

Environment. We conducted experiments on four servers with 43 GB random-access memory, Intel (R) Xeon (R) Platinum 8255C CPU and NVIDIA GeForce RTX 3090 GPU.

Metrics. We define three metrics to evaluate the performance of attacking CAPTCHAs: Detection Success Rate (DSR), Recognition Success Rate (RSR) and Attack Success Rate (ASR) in (7). CAPTCHA schemes with lower values on these three metrics are considered to be more effective in defense.

$$DSR = D/N, RSR = R/N, ASR = A/N, \quad (7)$$

where D is the number of CAPTCHAs in which all characters within the label are correctly located; R is the number of CAPTCHAs in which all characters within the label are accurately recognized, given that their positions are already known; A is the number of successful attacks; N is the total number of CAPTCHAs. Note that we have different definitions of the successful attack for Click-based CAPTCHAs and Input-based CAPTCHAs according to their characteristics. For Click-based CAPTCHAs, if attack models can correctly give all the characters contained in the label of CAPTCHA, we regard it as a successful attack. For Input-based CAPTCHAs, users are required to enter all the characters in CAPTCHAs and any additional answer will result in an error. Therefore, if the output characters of an attack model are exactly the same as the characters in the label of CAPTCHA, we regard it as a successful attack. The definition of successful detection is that for each box (a rectangle that covers the character) b within the label of a CAPTCHA, there is a box b' in the prediction result satisfying their Intersection over Union $IOU_{b,b'} < 0.5$.

Moreover, we use Top-1 accuracy to evaluate the impact of adversarial perturbation when recognizing characters. To further investigate the defense effect of adversarial perturbation on detection models, we define the Mean Misdetecion Number (MMN). It is the average number of predicted boxes that do not contain characters in the label per CAPTCHA. Higher MMNs mean better misleading effects of the adversarial example on detection models.

Algorithm Parameters. We set the maximum perturbation ϵ to 0.1, which means that the change of each pixel in







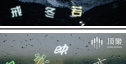
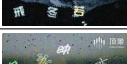

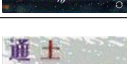
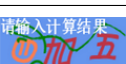



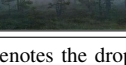

CAPTCHAs is less than 10%. In M-VNI-CT-FGSM, we set the maximum perturbation γ added by the attention *mask* in (5) to 0.03. Other parameters in *Gradient-based Attacks* and *Input Transformations* are the same as the original papers [28]–[31]. Moreover, the parameters of SVRE-MI-FGSM are the same as the original paper [18] except that we use the loss function introduced in (6) and set $w_1 = w_2 = w_3 = 1$ for three white-box models f_1, f_2, f_3 .

B. Defense Effectiveness of Adversarial CAPTCHAs

In this experiment, we show the effectiveness of our adversarial CAPTCHAs against black-box attacks. For each of the eight CAPTCHA schemes, we generate adversarial CAPTCHAs using 1,000 clean CAPTCHAs from the test set. Then we input clean CAPTCHAs and adversarial CAPTCHAs into three black-box attacks: **SR**, i.e. using SSD (VGG-16) [24] to detect characters and using Res-50 [21] as a character recognition model; **FA**, i.e. using FRCNN (Res-50) [23] to detect characters and using the Attention-based Models in [9] to recognize characters. Note that for the Input-based scheme Renmin and Sougou, characters are directly recognized using the End-to-end Attention-based Model [9]. In this case, the recognition and detection processes cannot be separated, so we only measure their ASR metric; **EM**, i.e. using the End-to-end Model with Mask-RCNN architecture in [7]. In this case, recognition and detection also cannot be separated, we only measure the ASR metric.

From Table II, the following observations can be made. Firstly, the DSR, RSR and ASR of clean CAPTCHAs are relatively high. Specifically, the most intuitive metric, i.e. ASRs are ranging from 6.5% to 99.0%, which proves that the commonly used attack models are already sufficient to crack large character set CAPTCHAs. Secondly, adversarial CAPTCHAs can significantly reduce these three metrics. We calculate the average ASR in clean CAPTCHAs and the average ASR in adversarial CAPTCHAs. Results show that after adding adversarial perturbation, the average ASR drop from 53.33% to 3.49%. In particular, the End-to-end Model in [7] and the Attention-based Model in [9] with great differences from the white-box models can also be successfully defended. Moreover, the RSR of these eight schemes is significantly reduced to near 0.0%. The DSR of most schemes is also reduced. However, the DSR drop of Geetest and Sougou is not much significant. We analyze the reasons as follows. As for the Geetest CAPTCHA scheme, the difference between the characters and the background is quite obvious, thus it is hard to make detection models “believe” that the characters in perturbed CAPTCHAs are background areas. As for the Sougou scheme, the character locations of this Input-based CAPTCHA are quite fixed. It is easy for models to learn this characteristic. Therefore, it is harder to mislead the attack models for them. Thirdly, as we can see in Table II, the perturbations in Adversarial CAPTCHAs are not much obvious, which means the usability of these CAPTCHAs is maintained. In summary, our adversarial CAPTCHAs generation method

TABLE II
DEFENSIVE EFFECTIVENESS OF ADVERSARIAL CAPTCHA UNDER THREE BLACK BOX ATTACKS SR, FA, EM

Scheme	Clean CAPTCHA					Adversarial CAPTCHA				
	Sample	Metric	SR [21], [24]	FA [9]	EM [7]	Sample	Metric	SR [21], [24]	FA [9]	EM [7]
Yidun		DSR	95.5%	92.7%	—		DSR	39.5% (↓ 56.0%)	26.5% (↓ 66.2%)	—
		RSR	50.2%	33.7%	—		RSR	0.0% (↓ 50.2%)	0.1% (↓ 33.6%)	—
		ASR	43.8%	21.1%	6.5%		ASR	0.0% (↓ 43.8%)	0.0% (↓ 21.1%)	0.0% (↓ 6.5%)
Dajie		DSR	99.6%	97.6%	—		DSR	50.4% (↓ 49.2%)	49.5% (↓ 48.1%)	—
		RSR	57.1%	38.1%	—		RSR	0.0% (↓ 57.1%)	0.1% (↓ 38.0%)	—
		ASR	42.5%	8.0%	26.8%		ASR	0.0% (↓ 42.5%)	0.0% (↓ 8.0%)	0.0% (↓ 26.8%)
YY		DSR	99.2%	99.2%	—		DSR	27.1% (↓ 72.1%)	65.5% (↓ 33.7%)	—
		RSR	70.7%	57.6%	—		RSR	0.0% (↓ 70.7%)	0.0% (↓ 57.6%)	—
		ASR	77.5%	55.0%	21.2%		ASR	0.0% (↓ 77.5%)	0.0% (↓ 55.0%)	0.0% (↓ 21.2%)
Baidu		DSR	99.8%	99.3%	—		DSR	35.4% (↓ 64.4%)	25.6% (↓ 73.7%)	—
		RSR	87.8%	65.9%	—		RSR	0.0% (↓ 87.8%)	0.0% (↓ 65.9%)	—
		ASR	83.0%	47.1%	60.1%		ASR	0.0% (↓ 83.0%)	0.0% (↓ 47.1%)	0.0% (↓ 60.1%)
Dingxiang		DSR	98.8%	97.6%	—		DSR	26.9% (↓ 71.9%)	4.8% (↓ 92.8%)	—
		RSR	53.5%	40.3%	—		RSR	0.0% (↓ 53.5%)	0.0% (↓ 40.3%)	—
		ASR	57.0%	39.1%	21.0%		ASR	0.0% (↓ 57.0%)	0.0% (↓ 39.1%)	0.0% (↓ 21.0%)
Renmin		DSR	100.0%	—	—		DSR	47.2% (↓ 52.8%)	—	—
		RSR	73.7%	—	—		RSR	3.5% (↓ 70.2%)	—	—
		ASR	70.7%	66.2%	38.3%		ASR	1.6% (↓ 69.1%)	7.5% (↓ 58.7%)	1.9% (↓ 36.4%)
Sougou		DSR	98.5%	—	—		DSR	95.8% (↓ 2.7%)	—	—
		RSR	99.4%	—	—		RSR	7.6% (↓ 91.8%)	—	—
		ASR	99.0%	87.2%	78.2%		ASR	9.5% (↓ 89.5%)	39.3% (↓ 47.9%)	21.1% (↓ 57.1%)
Geetest		DSR	99.9%	100.0%	—		DSR	85.8% (↓ 14.1%)	97.2% (↓ 2.8%)	—
		RSR	94.2%	64.0%	—		RSR	1.5% (↓ 92.7%)	1.6% (↓ 62.4%)	—
		ASR	93.9%	59.5%	77.3%		ASR	1.2% (↓ 92.7%)	0.9% (↓ 58.6%)	0.8% (↓ 76.5%)

(↓ x%) denotes the drop of adversarial CAPTCHA compared to clean CAPTCHA in metrics

is able to decrease the success rate of black-box models while maintaining the usability of CAPTCHAs.

C. Transferability Comparison of Adversarial CAPTCHA

We select four representative schemes, Yidun (distorted characters), YY (hollow characters), Dajie (thin font) and Geetest (multiple fonts), to conduct experiments. We use our selected white-box model to generate adversarial characters and adversarial CAPTCHAs. Then we attack them with multiple black-box models to investigate their transferability.

1) Transferability against Character Recognition Attacks:

For each scheme, we randomly select 1,000 characters in the test set to generate adversarial characters. Then we attack these characters using the white-box model and five black-box models. They are IncRes-v2, Inc-v3, Res-50, VGG-16, ensemble models which average the prediction probabilities of Inc-v3, Res-50, VGG-16 and the Attention-based Model in [9], which are denoted as **R1**, **R2**, **R3**, **R4**, **R5** and **R6**. We use five baseline methods for transferability comparison. They are VNI-CT-FGSM [28], VNI-FGSM [28], FGSM [14] and DeepFool [35]. This experiment also gives the ablation analysis of our method. Specifically, the VNI-CT-FGSM is the ablation of our method with the *Attention Mechanism* removed. The VNI-FGSM is the ablation with *Input Transformations* removed. The FSGM is the basic Gradient Signed Method, and the DeepFool is another classical adversarial examples generation method. Meanwhile, we use clean characters and add random Gaussian noise with the same perturbation magnitude ϵ to the characters for comparison.

According to the results in Table III, we have the following observations. Firstly, all character recognition models achieve high accuracy on both clean characters and characters with Random Noise. Among them, IncRes-v2 has the highest character recognition accuracy, while VGG-16 is the lowest. Secondly, The adversarial Characters generated by the white-box model IncRes-v2 using our method can significantly reduce the accuracy of multiple white-box and black-box models. Thirdly, *Gradient-based Attacks*, *Input Transformations* and *Attention Mechanism* can all improve the transferability of adversarial characters. After integrating these strategies, our method achieves the best defense result.

2) *Transferability against Character Detection Attack*: For each scheme, we use the 1,000 CAPTCHAs in the test set to generate adversarial CAPTCHAs. As mentioned in Section III-B, we first add perturbations on characters and then generate perturbations on the background. We attack these adversarial CAPTCHAs with three white-box models and three black-box models. The white-box models are FRCNN (Res-101), SSD (Mb-v2) and YOLO-v5, which are denoted as **D1**, **D2** and **D3**. The black-box models are FRCNN(Res-50), FRCNN(VGG-16), SSD(VGG-16), which are denoted as **D4**, **D5** and **D6**. We use clean CAPTCHAs and add three types of perturbation for comparison, i.e. random Gaussian noise, perturbations generated by the ENS-MI-FGSM [26] with stochastic variance reduction ablated, and noise generated by SVRE-MI-FGSM [18]. Both ENS-MI-FGSM and SVRE-MI-FGSM use the loss function in (6). Then we input the perturbed test set to white-box and black-box detection models.

TABLE III
TRANSFERABILITY EVALUATION OF ADVERSARIAL CHARACTERS AGAINST RECOGNITION MODELS

Scheme	Generation Method	Character Recognition Attack					
		White-box Model	Black-box Model				
		R1 → R1	R1 → R2	R1 → R3	R1 → R4	R1 → R5	R1 → R6
Yidun	Clean Character	90.4%	82.3%	81.7%	64.0%	93.1%	76.3%
	Random Noise	85.9%	73.5%	72.6%	54.3%	87.2%	69.4%
	DeepFool [35]	35.0%	78.9%	78.9%	63.0%	89.1%	69.3%
	FGSM [14]	12.1%	21.8%	22.3%	24.7%	27.4%	40.6%
	VNI-FGSM [28]	0.4%	14.1%	17.9%	21.9%	11.7%	39.8%
	VNI-CT-FGSM [28]	0.1%	0.8%	1.6%	1.1%	0.1%	17.6%
	Our Method	0.0%	0.0%	0.3%	0.5%	0.0%	6.0%
Dajie	Clean Character	96.1%	94.3%	91.1%	75.5%	98.2%	82.8%
	Random Noise	92.0%	90.9%	86.0%	67.4%	97.3%	76.2%
	DeepFool [35]	46.8%	93.1%	90.8%	73.4%	97.6%	78.0%
	FGSM [14]	13.4%	28.2%	45.5%	37.3%	41.2%	47.7%
	VNI-FGSM [28]	2.6%	21.9%	41.9%	31.7%	29.9%	37.4%
	VNI-CT-FGSM [28]	0.0%	0.8%	3.5%	8.3%	2.2%	17.6%
	Our Method	0.0%	0.2%	1.0%	3.8%	0.1%	9.4%
YY	Clean Character	92.0%	87.5%	92.6%	55.2%	95.0%	90.1%
	Random Noise	89.0%	84.4%	90.0%	51.3%	93.9%	88.4%
	DeepFool [35]	42.5%	85.7%	92.1%	54.1%	93.4%	88.7%
	FGSM [14]	8.4%	27.1%	40.6%	24.2%	32.0%	38.6%
	VNI-FGSM [28]	0.4%	14.8%	31.1%	20.6%	12.7%	54.2%
	VNI-CT-FGSM [28]	0.0%	0.8%	1.1%	0.9%	0.4%	5.5%
	Our Method	0.0%	0.3%	0.3%	0.4%	0.1%	0.9%
Geetest	Clean Character	98.3%	94.5%	97.9%	61.6%	98.2%	85.0%
	Random Noise	97.7%	93.1%	97.6%	60.3%	98.2%	82.2%
	DeepFool [35]	47.8%	93.3%	98.0%	61.3%	98.0%	83.7%
	FGSM [14]	16.6%	54.0%	76.8%	48.4%	62.2%	46.9%
	VNI-FGSM [28]	1.3%	47.4%	78.0%	46.5%	47.1%	51.0%
	VNI-CT-FGSM [28]	2.3%	9.6%	22.8%	17.8%	14.5%	11.3%
	Our Method	1.2%	7.5%	10.7%	17.9%	7.2%	9.0%

R1: IncRes-v2 [19], R2: Inc-v3 [20], R3: Res-50 [21], R4: VGG-16 [22], R5: Ensemble model [19]–[22], R6: Attention-based Model in [9]

A → B: Using characters generated by white-box model A to defend attack model B

The DSR and MMN of these detection models are measured.

From the results in Table IV, we can have the following observations. Firstly, all character detection models achieve high DSR and low MMN on both noise-free and random-noise CAPTCHA images. This implies that character detection models seldom make mistakes for CAPTCHAs without adversarial perturbations. Secondly, detection models with multiple architectures and feature extraction modules have lower DSR and higher MMN when attacking adversarial CAPTCHAs. Lower DSR means that our defense makes both the black-box and white-box models significantly less capable to detect characters, thus there are only a small number of adversarial CAPTCHAs with all characters located. Higher MMN means that the adversarial examples make the detection model give more prediction locations, i.e. they mistake parts of the background as characters. These results demonstrate that the ensemble method considering different object detection architectures can generate adversarial CAPTCHAs with transferable defense ability against detection attack models. Thirdly, compared to Random Noise and ENS-MI-FGSM, using the SVRE-MI-FGSM [18] can further improve the defense ability against character detection models.

D. Robustness of Adversarial CAPTCHA

In this experiment, we consider that attackers use one of the most effective countermeasures, i.e. adversarial training [36],

which uses adversarial examples to train models. With Yidun, Dajie, YY, and Geetest CAPTCHA schemes as examples, we generate adversarial perturbations on 1,000 CAPTCHAs from training sets with our method. Then, we extract the characters in all the perturbed CAPTCHAs to fine-tune the character recognition model Res-50 that has been trained on normal characters. Next, we use these perturbed CAPTCHAs to fine-tune the character detection model SSD (VGG-16) that has been trained on normal CAPTCHAs. We measure the ASR metric in three settings for comparison: 1) **NC**: Using SSD (VGG-16) and Res-50 with normal training to attack clean CAPTCHAs. 2) **AC**: Using SSD (VGG-16) and Res-50 fine-tuned by adversarial training to attack clean CAPTCHAs. 3) **AA**: Using SSD (VGG-16) and Res-50 fine-tuned by adversarial training to attack adversarial CAPTCHAs.

Results in Fig. 2 show that after fine-tuning models with adversarial CAPTCHAs, the ASR metrics of the attack models increase. Using Geetest scheme as an example, its ASR metric improves from 1.2% (shown in Table II) to 37.8%. However, the 37.8% ASR metric of **AA** is still significantly lower than the 93.9% ASR metric of **NC** and the 93.5% of **AC**. It means the denfense ability of our adversarial CAPTCHAs is still much better than the CAPTCHAs without adversarial perturbation. Therefore, even though attackers obtain our CAPTCHAs to fine-tune their models, the adversarial CAPTCHAs are robust to defense the fine-tuned models.

TABLE IV
TRANSFERABILITY EVALUATION OF ADVERSARIAL CAPTCHA AGAINST DETECTION MODELS

Scheme	Metric	Attack	Character Detection Attack					
			White-box Model			Black-box Model		
			D1+D2+D3 → D1	D1+D2+D3 → D2	D1+D2+D3 → D3	D1+D2+D3 → D4	D1+D2+D3 → D5	D1+D2+D3 → D6
Yidun	DSR	Clean CAPTCHA	94.1%	76.4%	86.4%	91.3%	94.8%	95.5%
		Random Noise	91.0%	74.2%	87.4%	88.3%	89.6%	91.8%
		ENS-MI-FGSM [26]	16.5%	4.7%	40.1%	38.7%	38.9%	48.1%
		Our Method	7.7%	1.3%	31.7%	28.3%	30.7%	39.5%
	MMN	Clean CAPTCHA	0.457	0.032	0.036	0.458	0.145	0.040
		Random Noise	0.421	0.274	0.242	0.532	0.364	0.235
		ENS-MI-FGSM [26]	5.112	13.177	4.123	5.479	4.289	2.988
		Our Method	6.337	16.522	6.237	6.987	6.808	5.254
Dajie	DSR	Clean CAPTCHA	99.1%	99.4%	99.8%	99.4%	99.6%	99.6%
		Random Noise	98.1%	97.2%	95.1%	97.6%	92.8%	96.8%
		ENS-MI-FGSM [26]	48.9%	20.8%	40.8%	58.8%	39.6%	54.0%
		Our Method	38.2%	6.2%	33.7%	50.8%	33.2%	50.4%
	MMN	Clean CAPTCHA	0.045	0.000	0.003	0.039	0.020	0.003
		Random Noise	0.046	0.026	0.006	0.064	0.023	0.020
		ENS-MI-FGSM [26]	3.293	8.139	2.732	4.946	0.821	0.928
		Our Method	3.836	10.644	3.841	5.779	1.476	1.983
YY	DSR	Clean CAPTCHA	99.5%	89.9%	97.7%	99.2%	99.5%	99.2%
		Random Noise	98.1%	86.4%	93.8%	98.4%	96.3%	97.2%
		ENS-MI-FGSM [26]	57.4%	8.0%	57.0%	83.3%	46.9%	50.7%
		Our Method	29.0%	1.7%	33.0%	65.8%	27.8%	27.1%
	MMN	Clean CAPTCHA	0.077	0.004	0.005	0.021	0.030	0.005
		Random Noise	0.101	0.100	0.015	0.184	0.074	0.049
		ENS-MI-FGSM [26]	2.227	10.720	1.977	2.520	1.878	1.420
		Our Method	3.416	15.082	3.425	3.775	4.096	3.585
Geetest	DSR	Clean CAPTCHA	100.0%	99.4%	99.9%	100.0%	100.0%	99.9%
		Random Noise	100.0%	99.9%	99.8%	100.0%	100.0%	99.9%
		ENS-MI-FGSM [26]	92.5%	0.9%	97.0%	98.2%	89.6%	92.4%
		Our Method	86.6%	0.1%	95.4%	97.2%	83.6%	85.8%
	MMN	Clean CAPTCHA	0.057	0.005	0.007	0.191	0.023	0.013
		Random Noise	0.061	0.185	0.041	0.351	0.205	0.176
		ENS-MI-FGSM [26]	5.032	21.311	8.474	5.833	6.778	6.552
		Our Method	9.111	24.306	14.661	9.331	10.863	11.078

D1: FRCNN (Res-101) [23], D2: SSD (Mb-v2) [24], D3: YOLO-v5 [25], D4: FRCNN(Res-50) [23], D5: FRCNN(VGG-16) [23], D6: SSD(VGG-16) [24]
A → B: Using CAPTCHAs generated by white-box model A to defend attack model B

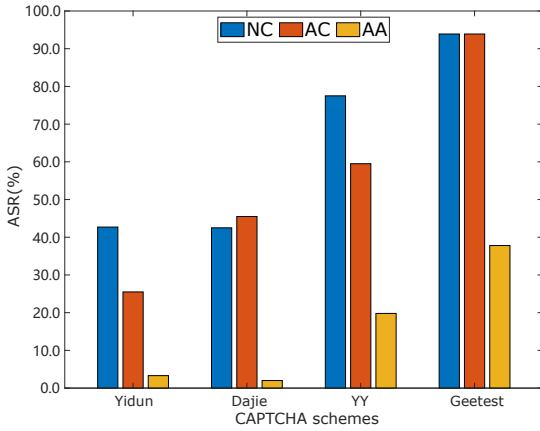


Fig. 2. ASR of four CAPTCHA schemes under NC, AC and AA settings

V. CONCLUSION

This paper proposes a defense method using adversarial example techniques to fight black-box attacks on large character set CAPTCHAs. To defend against character recognition attacks, we add adversarial perturbations to the charac-

ters by combining three strategies: *Gradient-based Attacks*, *Input Transformations* and *Attention Mechanism* so as to increase the transferability of adversarial CAPTCHAs. To fight character detection attacks, we leverage an ensemble method combining loss functions of multiple architectures to perturb the background of the CAPTCHAs. To the best of our knowledge, this is the first defense method in the large character set CAPTCHAs based on transferable adversarial example techniques. We conduct various experiments on eight popular Chinese CAPTCHA schemes. Our results show that the adversarial CAPTCHAs generated by our method significantly decrease the success rate of multiple black-box attacks. Moreover, the adversarial CAPTCHAs are robust since they can maintain defense ability when they are attacked by models using adversarial training. Since the real-world deployment of CAPTCHAs is quite complicated and more advanced attack methods will be used, making our defense more adaptable and robust will be our future work.

ACKNOWLEDGMENT

This work is supported by the Key Research and Development Program of Science and Technology Department of Sichuan Province under grant No. 2023YFG0145 and the

National Key Research and Development Program of China under grant No. 2022YFC3303101. In addition, this work is also partially supported by Key Research and Development Program of Science and Technology Department of Sichuan Province (nos. 2020YFS0575, 2021YFG0159, 2021KJT0012-2021YFS0067), Science and Engineering Connotation Development Project of Sichuan University (No. 2020SCUNG129), Sichuan University and Yibin Municipal People's Government University and City Strategic Cooperation Special Fund Project (No. 2020CDYB-29), and Science and Technology Plan Transfer Payment Project of Sichuan Province (No. 2021ZYSF007).

REFERENCES

- [1] D. Kim and L. Sample, "Search prevention with captcha against web indexing: A proof of concept," in *Proceedings of the 22th International Conference on Computational Science and Engineering and the 17th International Conference on Embedded and Ubiquitous Computing*, 2019, pp. 219–224.
- [2] M. Torky, A. Meligy, and H. Ibrahim, "Securing online social networks against bad bots based on a necklace captcha approach," in *Proceedings of the 12th International Computer Engineering Conference*, 2016, pp. 158–163.
- [3] 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.
- [4] P. Wang, H. Gao, X. Guo, Z. Yuan, and J. Nian, "Improving the security of audio captchas with adversarial examples," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–18, 2023.
- [5] Y. Xu, G. Reynaga, S. Chiasson, J.-M. Frahm, F. Monrose, and P. C. van Oorschot, "Security and usability challenges of moving-object captchas: decoding codewords in motion," in *Proceedings of the 21st USENIX Security Symposium*, 2012, pp. 49–64.
- [6] C. Li, X. Chen, H. Wang, P. Wang, Y. Zhang, and W. Wang, "End-to-end attack on text-based captchas based on cycle-consistent generative adversarial network," *Neurocomputing*, vol. 433, pp. 223–236, 2021.
- [7] J. Nian, P. Wang, H. Gao, and X. Guo, "A deep learning-based attack on text captchas by using object detection techniques," *IET Information Security*, vol. 16, no. 2, pp. 97–110, 2022.
- [8] A. Algwil, D. Ciresan, B. Liu, and J. Yan, "A security analysis of automated chinese turing tests," in *Proceedings of the 32nd Annual Conference on Computer Security Applications*, 2016, pp. 520–532.
- [9] P. Wang, H. Gao, Q. Rao, S. Luo, Z. Yuan, and Z. Shi, "A security analysis of captchas with large character sets," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 6, pp. 2953–2968, 2020.
- [10] D. Lin, F. Lin, Y. Lv, F. Cai, and D. Cao, "Chinese character captcha recognition and performance estimation via deep neural network," *Neurocomputing*, vol. 288, pp. 11–19, 2018.
- [11] Y. Zhang, H. Gao, G. Pei, S. Kang, and X. Zhou, "Effect of adversarial examples on the robustness of captcha," in *Proceedings of the 10th International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, 2018, pp. 1–109.
- [12] Z. Cheng, H. Gao, Z. Liu, H. Wu, Y. Zi, and G. Pei, "Image-based captchas based on neural style transfer," *IET Information Security*, vol. 13, no. 6, pp. 519–529, 2019.
- [13] Y. Gao, H. Gao, S. Luo, Y. Zi, S. Zhang, W. Mao, P. Wang, Y. Shen, and J. Yan, "Research on the security of visual reasoning captcha," in *Proceedings of the 30th USENIX Security Symposium*, 2021, pp. 3291–3308.
- [14] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proceedings of the 3rd International Conference on Learning Representations*, 2015, pp. 1–11.
- [15] M. Osadchy, J. Hernandez-Castro, S. Gibson, O. Dunkelman, and D. Pérez-Cabo, "No bot expects the deepcaptcha! introducing immutable adversarial examples, with applications to captcha generation," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2640–2653, 2017.
- [16] C. Shi, X. Xu, S. Ji, K. Bu, J. Chen, R. Beyah, and T. Wang, "Adversarial captchas," *IEEE Transactions on Cybernetics*, 2021.
- [17] C. Shi, S. Ji, Q. Liu, C. Liu, Y. Chen, Y. He, Z. Liu, R. Beyah, and T. Wang, "Text captcha is dead? a large scale deployment and empirical study," in *Proceedings of the 27th ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 1391–1406.
- [18] Y. Xiong, J. Lin, M. Zhang, J. E. Hopcroft, and K. He, "Stochastic variance reduced ensemble adversarial attack for boosting the adversarial transferability," in *Proceedings of the 31st IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14983–14992.
- [19] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, 2017.
- [20] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the 25th IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.
- [21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the 25th IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
- [22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the 3rd International Conference on Learning Representations*, 2015, pp. 1–14.
- [23] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Proceedings of the 29th Advances in Neural Information Processing Systems*, 2015, pp. 91–99.
- [24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Proceedings of the 14th European Conference on Computer Vision*, 2016, pp. 21–37.
- [25] G. Jocher et al., "Ultralytics/yolov5: v6.2," 2022. Available online: <https://doi.org/10.5281/zenodo.3908559> (accessed on 26 August 2022).
- [26] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," in *Proceedings of the 27th IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9185–9193.
- [27] W. Wu, Y. Su, X. Chen, S. Zhao, I. King, M. R. Lyu, and Y.-W. Tai, "Boosting the transferability of adversarial samples via attention," in *Proceedings of the 29th IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1161–1170.
- [28] X. Wang and K. He, "Enhancing the transferability of adversarial attacks through variance tuning," in *Proceedings of the 30th IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1924–1933.
- [29] J. Lin, C. Song, K. He, L. Wang, and J. E. Hopcroft, "Nesterov accelerated gradient and scale invariance for adversarial attacks," in *Proceedings of the 8th International Conference on Learning Representations*, 2020, pp. 1–12.
- [30] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, and A. L. Yuille, "Improving transferability of adversarial examples with input diversity," in *Proceedings of the 28th IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2730–2739.
- [31] Y. Dong, T. Pang, H. Su, and J. Zhu, "Evading defenses to transferable adversarial examples by translation-invariant attacks," in *Proceedings of the 28th IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4312–4321.
- [32] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the 16th IEEE International Conference on Computer Vision*, 2017, pp. 618–626.
- [33] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the 25th IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2921–2929.
- [34] D. Wang, C. Li, S. Wen, Q.-L. Han, S. Nepal, X. Zhang, and Y. Xiang, "Daedalus: Breaking nonmaximum suppression in object detection via adversarial examples," *IEEE Transactions on Cybernetics*, vol. 52, no. 8, pp. 7427–7440, 2021.
- [35] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the 25th IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2574–2582.
- [36] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," in *Proceedings of the 5th International Conference on Learning Representations*, 2017, pp. 1–17.