The Best Protection is Attack: Fooling Scene Text Recognition With Minimal Pixels

Yikun Xu, Pengwen Dai, Zekun Li, Hongjun Wang, *Student Member, IEEE*, and Xiaochun Cao, *Senior Member, IEEE*

***Abstract*— Scene text recognition (STR) has witnessed tremendous progress in the era of deep learning, but it also raises concerns about privacy infringement as scene texts usually contain valuable or sensitive information. Previous works in privacy protection of scene texts mainly focus on masking out the texts from the image/video. In this work, we learn from the idea of adversarial examples and use minimal pixel perturbation to protect the privacy of text information. Although there are well-established attacking methods on non-sequential vision tasks (e.g., classification), the attack on sequential tasks (e.g., scene text recognition) has not received sufficient attention yet. Moreover, existing works mainly focus on the white-box setting, which requires complete knowledge of the target model (e.g., architecture, parameters, or gradients). These requirements limit the scope of applications for the white-box adversarial attack. Therefore, we propose a novel black-box attacking approach for the STR models, only requiring prior knowledge of the model output. Besides, instead of disturbing most pixels as in existing STR attack methods, our proposed approach only manipulates a few pixels, meaning the perturbation is more inconspicuous. To determine the location and value of the manipulated pixels, we also provide an efficient Adaptive-Discrete Differential Evolution (AD^{2}E) by narrowing down the continuous searching space to a discrete space. It can greatly reduce the queries to the target model. Experiments on several real-world benchmarks show the effectiveness of our proposed approach. Especially, when attacking the commercial STR engine, Baidu-OCR, our method achieves higher attack success rates by a large margin than existing approaches. Our work establishes an important step towards using the black-box adversarial attack with minimal pixels to protect the privacy of text information from being easily obtained by STR models.**

Manuscript received 22 February 2022; revised 26 July 2022 and 30 October 2022; accepted 24 January 2023. Date of publication 15 February 2023; date of current version 24 February 2023. This work was supported in part by the National Key R&D Program of China under Grant 2020YFB1406704, in part by the Beijing Natural Science Foundation under Grant L212004 and Grant M22006, in part by the National Natural Science Foundation of China under Grant 62025604 and Grant U2001202, and in part by the Shenzhen Science and Technology Program under Grant 202206193000001 and Grant 20220816225523001. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Lalitha Sankar. *(Corresponding author: Pengwen Dai.)*

Yikun Xu is with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China, and also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: xuyikun18@mails.ucas.ac.cn).

Pengwen Dai and Xiaochun Cao are with the School of Cyber Science and Technology, Shenzhen Campus, Sun Yat-sen University, Shenzhen 518107, China (e-mail: daipw@mail.sysu.edu.cn; caoxiaochun@mail.sysu.edu.cn).

Zekun Li is with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: li002666@umn.edu).

Hongjun Wang is with the Department of Statistics and Actuarial Science, University of Hong Kong, China (e-mail: hjwang@connect.hku.hk).

Digital Object Identifier 10.1109/TIFS.2023.3245984

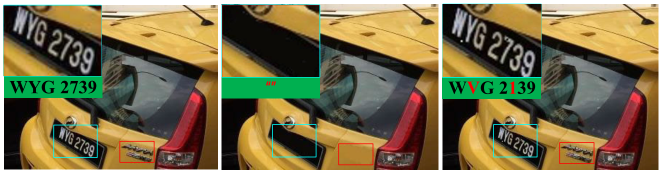
***Index Terms*— Scene text recognition, privacy protection, adversarial examples, black-box.**

I. Introduction

SCENE texts in the images/videos usually contain valuable information. They are ubiquitous in the real world and are useful in a large number of applications, such as image retrieval [1], visual captioning [2], question answering [3], hate speech detection [4], etc. However, these scene texts also contain various private information (e.g., personal names/addresses, telephone numbers, license plate numbers, etc.). Thus, the privacy protection of scene text information should be considered as important as traditional privacy protection [5], [6], [7]. When scene text privacy information is recognized by the machines used for business value (e.g., marketing) or other illegal activities (e.g., fraud, stealing), it would lead to privacy infringement [8]. With the development of deep learning, scene text recognition (STR) has become increasingly efficient and intelligent, thus bringing a greater risk of privacy infringement.

To protect the text privacy information, one intuitive solution is to remove the texts from the image/video [8], [9], [10], [11]. It can prevent privacy infringement to some extent. However, the biggest limitation of this method is that the text removal will usually sacrifice the visual quality and usability of an image as some backgrounds or other objects will also be removed, as shown in Fig. 1 (b). Another solution is to attack the STR models without harming the visual quality of the image. In recent years, adversarial machine learning has become a hot topic in the computer vision community and raises security concerns for the applications of deep learning [12], [13], [14], [15], [16], [17]. The STR model can also be easily fooled by adversarial examples which are generated by adding imperceptible perturbations to the original samples. Therefore, we turn to use adversarial attacks to protect the privacy of important text information in the natural scene images/videos without significant change to the images. Compared with text removal or background variation to protect privacy, adversarial attacks have much smaller perturbations and are inconspicuous to human observers, as shown in Fig. 1 (c).

Adversarial attacks and defenses on non-sequential vision tasks (e.g., classification) have been extensively studied in recent years [18], [19], [20]. However, the adversarial attacks on sequential vision tasks (e.g., scene text recognition),



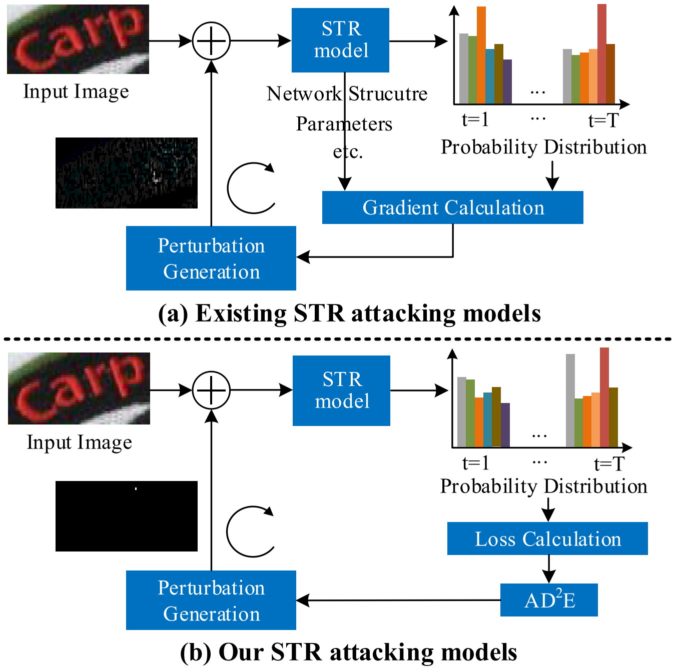
(a) Input image **(b) Text removal** **(c) Our method**

Fig. 1. Illustration of scene text information protection. (a) is the input image. (b) is the text removal results, which will sacrifice the visual quality. (c) is our adversarial results, which will retain the visual quality, and other symbols (in the red bounding box for each image) will not be removed. The top-left region denotes the corresponding zoomed region, and the green region denotes the recognition results.

have not attracted much attention yet. Non-sequential vision tasks, such as image classification [12], object detection [21], semantic segmentation [22], and source camera identification [23], do not assume temporal or spatial dependence for the input, and output one label for each input. Sequential vision tasks, however, treat the input as a sequence and output a sequence of labels (e.g., scene text recognition). The adversarial attacks on scene text recognition models are challenging due to the following reasons. Firstly, the sequential models produce arbitrary-length outputs, as words may contain different numbers of characters. This feature increases the difficulty of manipulation. Secondly, the sequential models usually utilize Recurrent Neural Networks (RNNs) or their variants for encoding the sequential context, which enhances the robustness of the representations and makes them much harder to attack [24].

Although several methods [24], [25], [26], [27], [28], [29] have been proposed to fool text image recognition models in recent years, they mainly focus on text images with homogeneous backgrounds (e.g., license plates, texts on pure-white backgrounds, etc.). Moreover, previous STR attack approaches require prior knowledge of the target model, as shown in Fig. 2 (a), including the model structure, parameters, and gradients. Besides, previous works disturb almost all pixels, which is not easily identified by people if the intensity of the perturbations for each pixel is low (invisible to human eyes). However, this kind of attack method may not be easy to perform in real-world privacy protection scenarios, as the details of the adversary model are unknown, and too many pixels are not likely to change simultaneously. Therefore, more attention should be paid to the black-box model and few-pixels-attack for STR models to protect the privacy of information in natural scene text.

Motivated by this, in this paper, we propose a novel and efficient black-box attack approach on the STR models by disturbing a minimal number of pixels, as shown in Fig. 2 (b). It requires less prior knowledge than previous adversarial methods. To find the eligible perturbation, we introduce the Adaptive-Discrete Differential Evolution (AD^{2}E) algorithm. This paper is an extended work based on [30], and it has three major improvements. Firstly, we propose the Adaptive-Discrete Differential Evolution (AD^{2}E) algorithm, which could achieve a higher success rate with fewer queries than the original Differential Evolution (DE) algorithm. Secondly, the designed Discrete Extremum Constraint reduces the number of queries to the previous model. DEC effectively transforms the continuous searching space into the discrete space, which greatly narrows down the searching space, thus reducing the number of iterations for attacking. This increases the applicability of our DEC due to the decreasing number of queries on the model. Because of the fewer constraints, it is easier to hide the information in different STR applications. Experiments demonstrate that those improvements bring a higher attack success rate but fewer queries and faster speed. Thirdly, we attack one real-world STR engine, Baidu OCR, to examine the transferability of our method. We also conduct ablation studies and experimental analyses, providing some insights into the proposed method.



(a) Existing STR attacking models

(b) Our STR attacking models

Fig. 2. Comparison of the existing STR attacking models (a) and our attacking models (b). Existing attacking models need complete knowledge of the target model and disturb a lot of pixels, while ours only needs the probability distribution and disturbs a very few pixels.

The main contributions are summarized as follows:

- We achieve text-based privacy protection by attacking the STR model with minimal pixel perturbation under black-box settings.
- We propose the Adaptive-Discrete Differential Evolution (AD^{2}E) method to adaptively find the adversarial examples with fast speed, and reduce the number of queries of attacking by discretely pushing the perturbation to the extremum value.
- Experiments on multiple benchmarks and a commercial STR engine (Baidu OCR) in the real-world setting show the superiority of our proposed method. Compared with the previous work, ours has higher transferability by a large margin.

The rest of the paper is structured as follows: Section II discusses the related work. Section III presents the architecture of the proposed attacking algorithm in detail. Section IV illustrates the experiments and analyses. Section V summarizes the paper.

II. Related Work

In this section, we first introduce some well-known STR methods based on deep neural networks. Then, we review some typical attacking methods on non-sequential vision tasks. Finally, we illustrate existing attack approaches for sequential-based STR models.

A. Scene Text Recognition

In recent years, numerous scene text recognition (STR) methods using deep neural networks (DNNs) have been proposed. These methods usually treat the STR task as a sequence recognition problem. The solution can be further divided into two ways: CTC-based methods and attention-based methods.

For the CTC-based methods [31], [32], [33], [34], CRNN [31] employs convolution neural networks (CNN) and recurrent neural networks (RNNs) to convert the input image into the feature sequences, it uses the connectionist temporal classification (CTC) [35] to align the feature sequences and output text strings by calculating the conditional probability between the prediction and the target sequences. Integrating the CTC-based paradigm, Borisyuk et al. [32] introduce curriculum learning to eliminate the need for training a separate character sequence encoding model. Hu et al. [34] propose a guided training strategy of CTC, to better learn the alignment and feature representations via a more powerful attentional guidance. To recognize irregular scene texts (i.e., oriented, perspective, or curved texts), STAR [33] integrates a spatial transformer to calibrate the irregular texts into regular texts (i.e. horizontal texts) end-to-end, before feeding into a CTC-based regular scene text recognition model.

For the attention-based methods [36], [37], [38], [39], R^{2}AM [36] utilizes the sequence attention mechanism to learn the sequence mapping between the encoded feature sequence and the target text strings. To eliminate the attention drift problem, Chen et al. [37] propose a focusing attention network. It uses character-level annotations to supervise network learning and correct the attention sequences. To capture the orientation variations, AON [38] designs a directional attention mechanism in four directions (i.e., left-to-right, right-to-left, top-to-down, down-to-top) to enhance the feature representations. And SLOAN [39] introduces a dynamic log-polar transformation network to covert the feature from the Cartesian coordinate space into the polar space. Besides, to recognize the irregular scene texts with the sequential attention-based mechanism, Yang et al. [40] utilize the character-level annotations to learn better contexts at each spatial position via the proposed 2D attention mechanism. And ASTER [41] rectifies the irregular scene text via Thin-Plate-Spline (TPS) transformation.

Most prior works are mainly devoted to increasing the accuracy of recognition, while rare studies focus on the safety and reliability of scene text recognition models. In this paper, we aim to study the adversarial robustness of the models in the black-box scenario, which helps evaluate the reliability of the models and further contributes to the progress of scene text recognition.

B. Attacks on Non-Sequential Vision Tasks

Adversarial attack for non-sequential vision tasks is one of the most important parts of the adversarial machine learning field. The attacking methods can be roughly categorized into the white-box attack and the black-box attack. The white-box attack needs complete knowledge of the target model. A pioneering work [42] utilizes a box-constrained L-BFGS, which finds an approximate minimum solution via the line search, to construct adversarial examples. In [43], a fast gradient sign method (FGSM) is proposed to generate adversarial examples by nudging one step in the reverse gradient direction. After that, the multiple follow-up algorithms, including Iterative FGSM (I-FGSM) [44], Momentum I-FGSM (MI-FGSM) [45], are proposed to generate adversarial perturbations successfully by multi well-designed steps along the gradient directions at every tiny step. Moreover, an attacking method named deepfool [46] is proposed to compute a minimal norm adversarial perturbation by trying to find the nearest classification hyperplane. Carlini and Wagner [47] also present an efficient optimization objective by applying a change of variables and optimizing over the new variables to find the smallest perturbations during the optimization process. More attacking applications on different tasks are proposed for semantic segmentation [22], object detection [48], source camera identification [23], etc. The above-mentioned methods are all performing white-box attacks and they assume that the details of the target models are known. On the contrary, the black-box attack does not require knowledge about model structure, parameters, and gradients. Su et al. [49] show that manipulating only a few pixels could successfully attack the image classification models. Shukla et al. [50] open avenues by applying Bayesian optimization for developing black-box adversarial attacks. They show that it requires low query budgets for L_2 and L_∞ norm-constrained threat models.

C. Attacks on Sequential Tasks of Scene Text Recognition

Although lots of works have been done on attacking non-sequential vision models, sequential vision tasks (e.g., scene text recognition), on the contrary, have not attracted much attention yet. Song and Shmatikov [25] design a model to fool the traditional optical character recognition (OCR) system by adding adversarial perturbations. Chen et al. [26] propose a targeted white-box attacking method, the Fast Adversarial Watermark Attack (FAWA), which generates the perturbations as watermarks, and makes the adversarial images imperceptible to human eyes. In [51], the authors introduce the first practical adversarial attack against deep license plate recognition models, termed Robust LightMask Attacks (RoLMA), which adopts illumination technologies to create several light spots as adversarial noises. In addition to attacking the text images with coherent backgrounds, some scholars also pay attention to fooling complicated natural scene text recognition systems. Yuan et al. [27] propose an adaptive loss to speed up the attacking for sequential learning tasks. It learns the adaptive weightings by a modified binary search loss function without manually searching hyper-parameters. Meanwhile, Xu et al. in [24] and [28] make comprehensive attempts on constructing adversarial examples to fool both the CTC-based and attention-based STR models, by exploiting a generic optimization-based adversarial attack approach. Furthermore, Yang et al. [29] propose a novel and effective objective function to achieve a higher attacking success rate with less L_2 norm perturbation.

This paper proposes a black-box attacking method on STR models. Compared with the white-box attacking and the optimization objective of L_2 norm perturbation, our approach does not assume to know complete prior knowledge of the target models. Also, the attack manipulates only a few pixels (L_0 norm). We further examine the effectiveness of our attack method on the commercial Baidu OCR engine by transferring attacking, compared with the state-of-the-art white-box STR attack [24].

III. The Proposed Method

A. Threat Model Setup

Existing attacking methods on the STR model are mostly based on the white-box setting. They assume to have full knowledge of the target model, such as architecture, parameters, and gradients. However, these attacking methods are usually impractical due to strong assumptions. To overcome such weaknesses, we propose a black-box attacking method that only accesses the probability distributions of the model output. This assumption is not as strong as the white-box assumption and is more practical in real-world scenarios.

Additionally, previous attacking works for STR models usually disturb most of the pixels in the high-dimensional space, whose dimension is nearly the number of all the pixels in the input image. On the contrary, we attempt to find a feasible perturbation by altering a minimum number of pixels. Meanwhile, we also restrict the value of the perturbation to some fixed values (e.g., pure white or black), which makes it easier to carry out the attack.

The goal of this paper is to find the adversarial examples for the STR models, which can be formalized as an optimization problem:

$$\mathcal{F}(\boldsymbol{I}^{adv}) \neq \mathcal{F}(\boldsymbol{I}), \tag{1}$$

$$s.t. \;\; ||\boldsymbol{I} - \boldsymbol{I}^{adv}||_F \leq \varepsilon, \tag{2}$$

where $\mathcal{F}$ denotes the STR model. $||\boldsymbol{I} - \boldsymbol{I}^{adv}||_F$ indicates the difference of F-norm between the clean image $\boldsymbol{I}$ and the adversarial example $\boldsymbol{I}^{adv}$. The difference should not exceed a small threshold ε.

To make sure $\boldsymbol{I}$ and $\boldsymbol{I}^{adv}$ differentiate on only a few pixels and restrict the value of the perturbation to be some fixed values, our attacking STR model can be formulated as finding a solution for the following problem, mathematically:

$$\max_{\boldsymbol{I}^{adv}} \; \mathcal{H}(\boldsymbol{I}^{adv}), \tag{3}$$

$$s.t. \;\; ||\boldsymbol{I} - \boldsymbol{I}^{adv}||_0 = N_p, \tag{4}$$

$$\boldsymbol{I}^{adv}_{x,y} = \beta_k, \tag{5}$$

where $\mathcal{H}$ denotes the loss calculated by the STR model given the ground-truth label sequence string. N_p is the number of pixels that can be altered from the clean image. Eq. (5) means the value of the attacking pixel position (x, y) in the adversarial example is set to be as an appropriate fixed value $\beta_k \in \beta$. β is the fixed value set, which can narrow down the search space. β_k is the k-th value in β.

B. Framework Overview

As shown in Fig. 3, given a clean image $\boldsymbol{I} \in \mathbb{R}^{H \times W \times 3}$ (H and W are the height and width of the image) that can be correctly recognized by the STR model, we first generate the probing adversarial perturbations with the perturbations generation module. With the probing adversarial perturbations added to the clean image, we obtain the probing adversarial images. Then we feed these probing adversarial images into the STR model to get the probability distributions and calculate the losses for each adversarial perturbation/image. After that, we select the valid adversarial perturbation based on the largest loss of absolute value. If its corresponding image could confuse the STR model successfully, then it would be our final adversarial perturbation. Otherwise, we use an iterative optimization method (details illustrated in Section III-C) to find the valid adversarial perturbation.

1) ***Probing Adversarial Perturbations Generation:*** To obtain the adversarial images, we first create N_s probing adversarial perturbations $\boldsymbol{P}(g) = \{\boldsymbol{P}_i(g)\}_{i=1}^{N_s}$, where the i-th probing adversarial perturbation $\boldsymbol{P}_i(g)$ is represented by the set of pixels being attacked:

$$\boldsymbol{P}_i(g) = \{p_i^j(g) \mid p_i^j(g) = (x, y, v)\}, \;\; j = 1, \ldots, N_p, \tag{6}$$

where x and y are the coordinates of attacking pixel position, with $x \sim \mathcal{U}(0, W)$ and $y \sim \mathcal{U}(0, H)$. $\mathcal{U}$ represents the uniform distribution. v is the value of the attacking pixel, and it samples from the Gaussian distribution $\mathcal{N}(\mu, \delta)$. When adding DEC, v_a is pushed to 0 or 255 adaptively with Eq. (19) and (20) (details in Section III-D). N_p denotes the number of pixels to be attacked in an image. j is the index of the pixel to be attacked. g is the number of iterations, such that $g = 0$ in the first iteration. We set the number of max iterations as g_{max}.

Then we apply the probing adversarial perturbations $\boldsymbol{P}(g)$ to manipulate the clean image $\boldsymbol{I}$ and generate the corresponding probing adversarial images $\boldsymbol{I}'$. Each probing adversarial image $\boldsymbol{I}'_i$ can be formulated as:

$$\boldsymbol{I}'_i = \phi(\boldsymbol{I}, \boldsymbol{P}_i(g)), i = 1, \ldots, N_s, \tag{7}$$

where ϕ denotes the replacement operation that updates the pixel value in the clean image $\boldsymbol{I}$ based on the pixel position (x,y) and the adversarial value v in $\boldsymbol{P}_i(g)$.

2) ***Valid Adversarial Perturbation Selection:*** After we obtain the probing adversarial images $\{\boldsymbol{I}'_i\}_{i=1}^{N_s}$ with Eq. (7), we feed them into the STR model to output the probability distributions $\mathbf{O} \in \mathbb{R}^{N_s \times T \times C}$ of the prediction, each of which is formulated as:

$$\mathbf{O}_i = \mathcal{J}(\boldsymbol{I}'_i; \Theta), \tag{8}$$

where T is the number of sequence features for the CTC-based STR model or the predicted number of labels for the

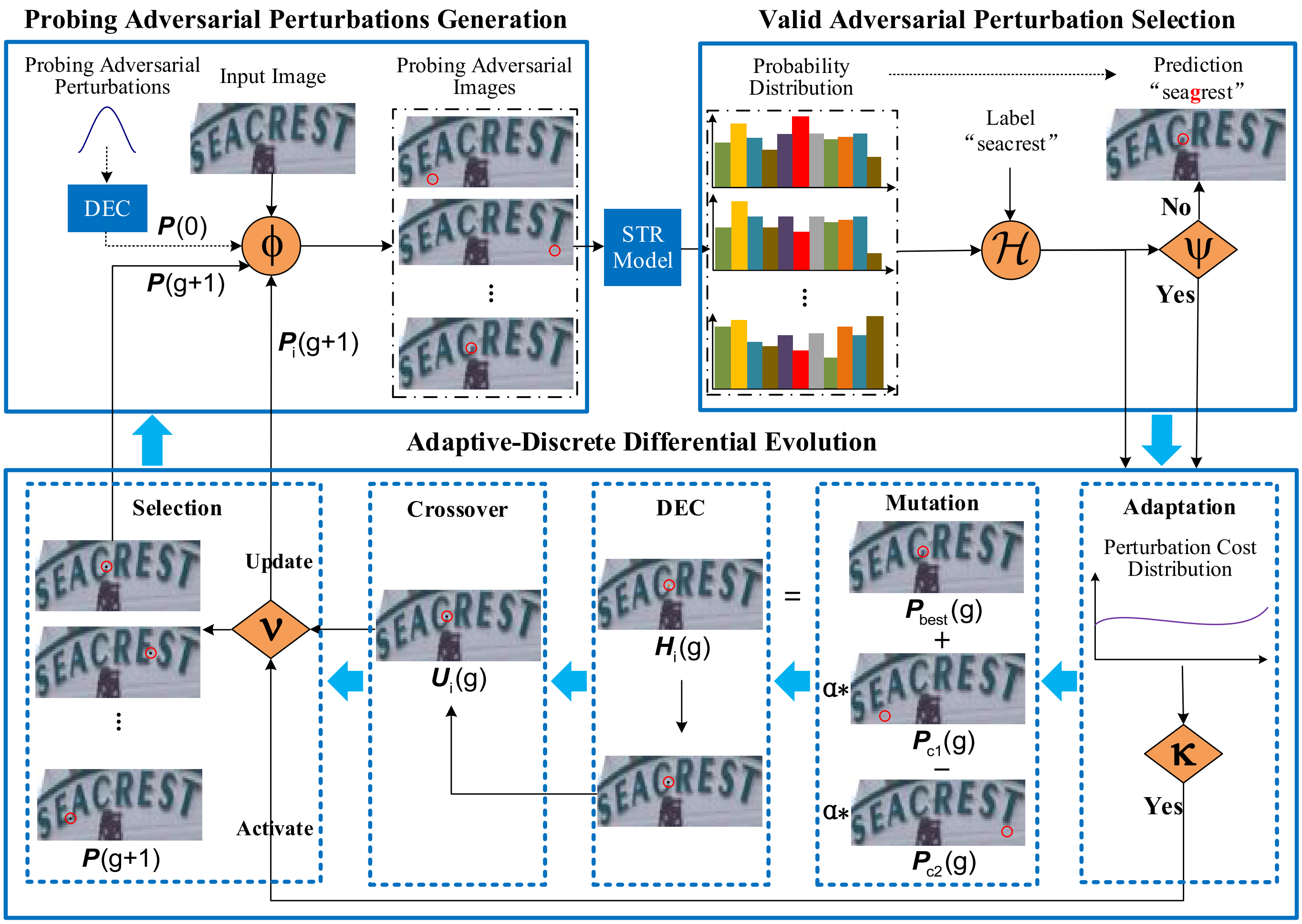


Fig. 3. Framework for constructing adversarial examples with minimum pixel perturbations using AD^{2}E. Given an input image that can be correctly recognized by the STR model, we first alter the input (clean image) with random initial perturbations by ϕ to generate multiple probing adversarial images. Then, we feed these images into the black-box STR model and obtain the losses between the predictions and ground-truth text string by $\mathcal{H}$. Next, we choose the probing image with the largest loss and feed it into the validation process denoted as ψ. If the recognized characters are not the same as the ground-truth label sequence string, then the attack is successful. Otherwise, AD^{2}E iteratively constructs new probing adversarial perturbations. In the Adaptation step, we estimate whether there is an "expectant" adversarial perturbation in the current generation by cost distribution and κ. Then we perform Mutation, DEC, Crossover, and Selection steps. The υ in the Selection step updates the current generation individual by individual or generation by generation according to the previous Adaptation step.

attention-based STR model. C denotes the character classes plus the blank; Θ is the parameters of the STR model.

In our attacking method, we only utilize the output probability distribution $\mathbf{O}$, without knowing the details of the STR model (e.g, $\mathcal{J}$, Θ, etc.).

Then we calculate the loss $\mathbf{Q}_i$ for each probing adversarial perturbation, which is expressed as:

$$\mathbf{Q}_i = \mathcal{H}(\mathbf{O}_i, \boldsymbol{l}), \tag{9}$$

where $\boldsymbol{l} \in \mathbb{R}^{N_l}$ means the ground-truth label sequence string, and N_l is the length of the text string. $\mathcal{H}$ denotes the loss calculation function.

For CTC-based STR models, which use the connectionist temporal classification (CTC) [35] to align the sequences, Eq. (9) could be instantiated as:

$$\mathbf{Q}_i = -log \sum_{\pi_i = \mathcal{B}(\boldsymbol{l})} p(\pi_i | \boldsymbol{I}'_i), \tag{10}$$

where $\boldsymbol{\pi}_i$ means the alignment path for the i-th probing adversarial image; $\mathcal{B}$ denotes the mapping function. For example, $\mathcal{B}$ maps the '-T-I-F–S-2-000-22-222-' onto 'TIFS2022' ('-' denotes the blank).

For the attention-based STR models, which utilize the sequence attention mechanism to learn the sequence mapping, Eq. (9) can be instantiated as the following equation:

$$\mathbf{Q}_i = -\sum_{t=1}^{N_l} log\ p(\boldsymbol{l}_t | \boldsymbol{I}'_i, \boldsymbol{l}_0, \ldots, \boldsymbol{l}_{t-1}), \tag{11}$$

where $\boldsymbol{l}_t$ is the t-th character in the label.

Next, we select one valid adversarial perturbation $\boldsymbol{P}''$ from the probing adversarial perturbations $\{\boldsymbol{P}'_i\}_{i=1}^{N_s}$ based on the loss $\mathbf{Q}$. It can be expressed as:

$$\boldsymbol{P}'' = \boldsymbol{P}'_{best}, \ best = \arg\max_i \{\mathbf{Q}_i\}_{i=1}^{N_s}. \tag{12}$$

Let $\boldsymbol{l}_{predict}$ denote the prediction label obtained from the probability distributions $\mathbf{O}$, and $\boldsymbol{I}''$ represent the corresponding

Algorithm 1 Attack on STR by AD^{2}E

```
Input: {$\boldsymbol{I}$: input image; $\boldsymbol{l}$: the ground-truth label sequence string; $g_{max}$: the maximum number of iterations; $N_s$: the number of probing adversarial perturbations; $N_p$: the number of pixels to be disturbed; $\epsilon$: the crossover probability.}
Output: {$\boldsymbol{I}^{adv}$: adversarial image.}
    // Initialization
    for $i = 1$ to $N_s$:
        for $k = 1$ to $N_p$:
            $p_i^k(0) = [x \sim \mathcal{U}(0, H), y \sim \mathcal{U}(0, W), v \in \{0, 255\}]$
            Re-discretize the values $v$ of attack pixels using Eq. (19) and Eq. (20);
        end for
    end for
    // Optimization
    $g = 0$
    while $g < g_{max}$ and $\boldsymbol{l}_{predict} == \boldsymbol{l}$, do
    // Adaptation
        Calculate the loss $Q_{best}$ of current best adversarial perturbation using Eq. (9) to Eq. (12) and the threshold $\tau_a$ of the current generation using Eq. (13);
        if $|Q_{best}| > |\tau_a|$:
            Set $flag = 1$;
        else if $|Q_{best}| < |\tau_a|$:
            Set $flag = 0$;
        else:
            break;
        for $i = 1$ to $N_s$, do
    // Mutation, DEC, Crossover and Selection
            $\boldsymbol{H}_i(g) = Mutation(\boldsymbol{P}_i(g))$, using Eq. (14);
            Re-discretize the values of attack pixels with Eq. (19) and Eq. (20);
            $\boldsymbol{U}_i(g) = Crossover(\boldsymbol{P}_i(g), \boldsymbol{H}_i(g))$, using Eq. (15);
            $\boldsymbol{P}_i(g + 1) = Selection(\boldsymbol{P}_i(g), \boldsymbol{U}_i(g))$, using Eq. (17);
            if $flag == 1$:
                Update $\boldsymbol{P}_i(g)$ with $\boldsymbol{P}_i(g+1)$ immediately;
                if $|\boldsymbol{Q}_i(g+1)| > |\boldsymbol{Q}_{best}(g)|$:
                    Update $\boldsymbol{Q}_{best}(g)$ and $\boldsymbol{P}_{best}(g)$ with $\boldsymbol{Q}_i(g+1)$ and $\boldsymbol{P}_i(g+1)$ immediately;
        end for
        $g = g + 1$
    end
    // Using the selected perturbation to modify the clean image
    $\boldsymbol{I}^{adv} = \phi(\boldsymbol{I}, \boldsymbol{P}_{best})$
```

valid adversarial image generated by $\boldsymbol{P}''$. When the prediction label $\boldsymbol{l}_{predict}$ of the valid adversarial image $\boldsymbol{I}''$ is unequal to the ground-truth label $\boldsymbol{l}$, $\boldsymbol{P}''$ and $\boldsymbol{I}''$ are our expected adversarial perturbation $\boldsymbol{P}^{adv}$ and adversarial image $\boldsymbol{I}^{adv}$, respectively. Then we stop the search process. Otherwise, we iteratively search the adversarial image until it achieves the goal or reaches the maximum number g_{max} of iterations (seeing the following subsection). The processing is summarized in Algorithm 1.

C. Optimization: Adaptive-Discrete Differential Evolution

Inspired by the work [49] that employs Differential Evolution (DE) [52] to generate adversarial examples, we propose the Adaptive-Discrete Differential Evolutions (AD^{2}E) to search the adversarial image in an efficient way.

The main target for replacing DE with AD^{2}E is to generate adversarial examples with higher performance (i.e. higher successful attacking rate, fewer queries, and faster speed) than DE. It is because these indicators are usually important for the practical use of private protection. DE is an evolutionary algorithm, which makes few assumptions about the target problem being optimized [49]. It does not require knowledge of gradient information, and it could search the high-dimensional spaces for candidate solutions. Besides, DE is a parallel direct search method that optimizes a problem by iteratively finding better candidate solutions compared with the previous solutions [49], [52]. However, DE usually requires a lot of queries and is likely to waste many queries on the target model because of its generation updating strategy. To improve DE, Dynamic Differential Evolution (DDE) [53] is proposed and is reported to have better efficiency and lower memory requirement with fewer queries. DE updates the probing adversarial perturbations generation by generation, that is updating the probing adversarial perturbations after all the new probing adversarial perturbations are generated. In each iteration, we only update the probing adversarial perturbations once. Differently, DDE updates the probing adversarial perturbations individual by individual, which means that we update the probing adversarial perturbations once one new probing adversarial perturbation is generated. In each iteration, we update the probing adversarial perturbations for N_s times. It could accelerate convergence. However, DDE has the tendency to be trapped in the local optimum in the early stage and usually takes much more time than DE, which searches relatively globally and is faster by parallel technique. To combine the advantages of DE and DDE, the proposed optimization method AD^{2}E adaptively employs DE or DDE to balance the performance, speed, and few queries.

Specifically, AD^{2}E includes four main steps: adaptation, mutation, crossover, and selection steps. The adaptation step is designed to estimate whether the current generation has an "expectant" adversarial perturbation. When the adaptation step estimates that the best adversarial perturbation in the current generation is likely near the solution, this best adversarial perturbation would be regarded as an "expectant" adversarial perturbation. Then the posterior selection step would automatically update the current generation individual by individual to accelerate the convergence speed. When the adaptation step estimates that there is no "expectant" adversarial perturbation in the current generation, the selection step would update generation by generation in parallel to reduce the runtime. During optimization, if the adaptation step finds the search to fall into the local optimal solution, it would directly skip the current optimization to reduce the runtime. The mutation and crossover steps are designed to explore the search space, while the selection process ensures that the promising probing adversarial images could be further generated. Although these steps are similar to those of the genetic algorithm (GA) [54], there are some differences in the specific operations [52], [55]. One is that the mutation in GA often changes a part of the genes of the current individual, while the mutation of our method uses the weighted difference vector plus the current best individual to update the individual. Another is that the bad individuals would be abandoned by probability in the selection step of GA, while our method selects the better individuals (i.e. discard the bad individuals absolutely) for the next generation.

*1) **Adaptation**:* Firstly, we try to estimate whether the current generation has an "expectant" adversarial perturbation by the threshold τ_a, which is calculated by the following equation:

$$\tau_a = max(\gamma + \lambda\xi, \eta), \tag{13}$$

where γ and ξ are the mean and standard deviation of the losses of the probing adversarial images respectively in the current generation. The factor λ is set to screen out the probing adversarial images/perturbations with "large" losses, which should be much larger than the loss distribution in the current generation. The constant η is set to exclude probing adversarial images/perturbations with very low losses. We set η and λ to be 0.4 and 11 in the experiments.

Then we could estimate the state of the current generation by comparing the loss of the best probing adversarial perturbation with the threshold by Eq. (13). We can obtain the best perturbation $\boldsymbol{P}_{best}(g)$ on the g-th iteration, and meanwhile get its loss $\boldsymbol{Q}_{best}(g)$ using Eq. (9) to Eq. (12). Then we examine whether the $\boldsymbol{Q}_{best}(g)$ is an "expectant" adversarial perturbation. If it is an "expectant" adversarial perturbation (i.e., its loss $\boldsymbol{Q}_{best}(g)$ is larger than the threshold τ_a), we activate the dynamic module and update the probing adversarial perturbations individual by individual. If $|\boldsymbol{Q}_{best}(g)| < |\tau_a|$, we update the probing adversarial perturbations generation by generation in parallel to save the runtime. If $|\boldsymbol{Q}_{best}(g)| = |\tau_a|$, it means that we get stuck in a local extremum and could not find the adversarial perturbation during optimization, then we skip the process to save the runtime.

*2) **Mutation**:* We then randomly select two perturbations $\boldsymbol{P}_{c1}(g)$ and $\boldsymbol{P}_{c2}(g)$ on the g-th iteration to obtain the difference vector. $c1$ and $c2$ are randomly selected to bring in more choices for subsequent crossover and selection steps.

The intuition for mutation is to ensure that the new mutation perturbations $\boldsymbol{H}(g)$ are selected around the best perturbation $\boldsymbol{P}_{best}(g)$. The random $c1$ and $c2$ are used to generate otherness for each new mutation perturbation $\boldsymbol{H}_i(g)$. A hyper-parameter α is used as the scaling factor, which controls the relative distance between the new mutation perturbations and the current best probing adversarial perturbation.

After mutation, the probing adversarial perturbations $\boldsymbol{P}(g)$ are replaced by mutation perturbations $\boldsymbol{H}(g)$. Each mutation vector $\boldsymbol{H}_i(g)$ could be written as follows:

$$\boldsymbol{H}_i(g) = \boldsymbol{P}_{best}(g) + \alpha(\boldsymbol{P}_{c1}(g) - \boldsymbol{P}_{c2}(g)), \tag{14}$$

where i, $c1$, and $c2$ represent the i-th, $c1$-th, and $c2$-th mutation perturbations in $\boldsymbol{H}(g)$. $c1 \neq c2 \neq i$. The scaling factor α is set to be 0.5 in the experiments.

To reduce the number of queries to the model, we introduce DEC (see section III-D for details) to find the appropriate attacking pixel position on the image in a discrete searching space.

*3) **Crossover**:* To increase the diversity of the probing adversarial perturbations, we update the perturbations according to the crossover step [52]:

$$u_i^j(g) = \begin{cases} h_i^j(g), & if \;\; rand(0,1) \le \epsilon, \\ p_i^j(g), & else, \end{cases} \tag{15}$$

where $rand(0, 1)$ represents that the random number is chosen from the uniform distribution of $[0, 1]$; $\epsilon \in [0, 1]$ denotes the probability of crossover. ϵ is set to be 1 in this work, meaning we allow all mutation perturbations $\boldsymbol{H}(g)$ to replace the probing adversarial perturbations $\boldsymbol{P}(g)$; $p_i^j(g)$ is the same meaning as that in Eq. (6); just as $p_i^j(g)$, $h_i^j(g)$ means the j-th pixel perturbation for $\boldsymbol{H}_i(g)$ as follows:

$$\boldsymbol{H}_i(g) = \{h_i^j(g) | h_i^j(g) = (x, y, v)\}, \tag{16}$$

where $\boldsymbol{H}_i(g)$ is calculated by Eq. (14).

*4) **Selection**:* This step guarantees that the probing adversarial perturbations $\boldsymbol{P}_i(g+1)$ in the $(g+1)$-th iteration is better than or equal to $\boldsymbol{P}_i(g)$ in the g-th iteration. It is formulated as follows:

$$\boldsymbol{P}_i(g+1) = \begin{cases} \boldsymbol{U}_i(g), & if \;\; \mathcal{H}(\boldsymbol{U}_i(g)) > \mathcal{H}(\boldsymbol{P}_i(g)), \\ \boldsymbol{P}_i(g), & else, \end{cases} \tag{17}$$

where $\boldsymbol{U}_i(g)$ is comprised of all $u_i^j(g)$ as follows:

$$\boldsymbol{U}_i(g) = \{u_i^j(g) | u_i^j(g) = (x, y, v)\}, \tag{18}$$

where similar to $p_i^j(g)$, $u_i^j(g)$ means the j-th pixel perturbation in $\boldsymbol{U}_i(g)$.

When the adaptation step finds that there is an "expectant" probing adversarial perturbation in the current generation, the selection step starts to search and update the generation individual by individual. It could help to decrease the number of queries and memory usage of our method. Otherwise, if the adaptation step estimates that all the probing adversarial perturbations in the current generation are trivial, the selection step will perform to update generation by generation to help the algorithm search fast and parallel.

It should be pointed out that before we have generated a new generation $\boldsymbol{P}(g+1)$, the mutation perturbations $\boldsymbol{H}(g)$ and the crossover perturbations $\boldsymbol{U}(g)$ could only be regarded as temporary perturbations within the iteration, not a new generation. After the selection step, we just obtain a new generation $\boldsymbol{P}(g+1)$. Therefore, the mutation, crossover, and selection steps iteratively search on the previous probing adversarial perturbations, not on the newly generated perturbations or images. Our method could guarantee that the new generation is no worse than the previous generation, that is our method iteratively searches for the new better adversarial perturbations than the previous perturbations.

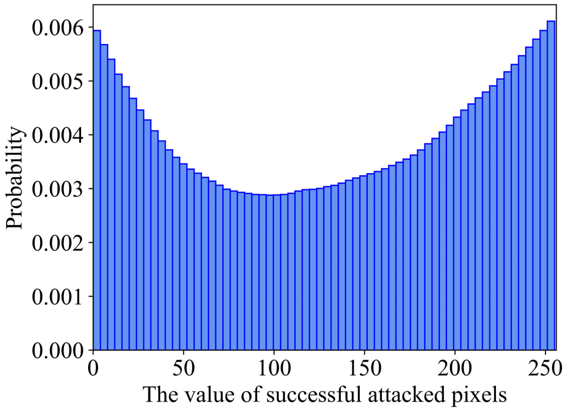


Fig. 4. The successful attacked pixels distribution for CUTE in the only one-pixel setting. There are 288 images in the CUTE dataset. We firstly screen out the images which could be correctly recognized by CRNN. Then we enumerate all the attack positions in the screened-out images and all possible integer attacking values in [0, 255]. We can conclude that 0 and 255 are the most frequent values that attack the dataset successfully.

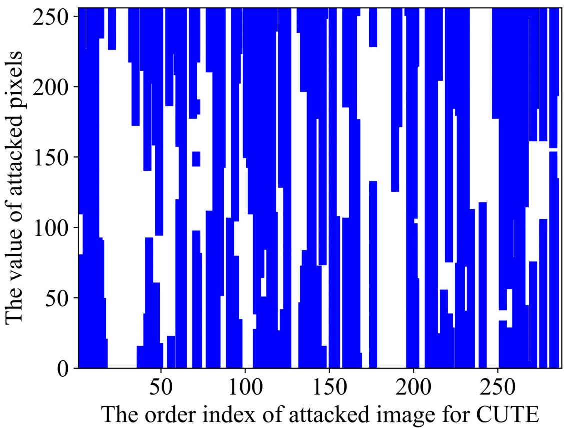


Fig. 5. The success attack value for every image. The blue points represent that the corresponding value can attack the image successfully. The index means the order of the image in the dataset. It can be found that sometimes when 0 fails to attack, 255 will realize to attack successfully.

D. Discrete Extremum Constraint

In the optimization step above, the candidate solutions span over a large search space. The optimization process not only iterates through all the pixel locations in the target image but also all the possible values of the pixels. To find which gray value is easy to attack the STR models, which could help to narrow down the searching space, we perform the following brute-force searching algorithm. Specifically, we apply the brute-force searching algorithm to enumerate all the attack positions in the image and all possible integer attacking values in [0, 255]. We use the dataset CUTE [56] and the STR model CRNN to perform the experiment. We firstly screen out the images which could be correctly recognized by CRNN. Then we search the screened-out images, whose pixel positions are between 0 and (H-1) for height and 0 and (W-1) for the width of the image. Besides, we search all the gray values from 0 to 255 of the attacking point in the image to examine whether it could change the output label by CRNN. As the brute-force searching algorithm is very time-consuming, we only perform the experiment of the one-pixel attack. Then we use the statistical results to analyze which gray value is more likely to attack images.

As shown in Fig. 4 and 5, we can observe that 0 and 255 are the most frequent values that can attack the STR model successfully. Moreover, for most images that can be successfully attacked, we find that using the attack value v of 0 or 255 can fulfill the adversarial attack. However, we also notice that a few images can not be attacked successfully for all the positions with attack value v of 0 or 255. For example, using $v = 0$ fails to attack some images successfully in all positions. Fortunately, it can be found in Fig. 5 that when 0 fails to attack, 255 will be able to attack successfully, and vice versa. Especially, using attack values 0 and 255 can successfully attack almost all images at some appropriate position. It indicates that discretizing the values of attacked pixels to 0 and 255 is reasonable.

Therefore, to reduce the number of queries/iterations and narrow down the search space, we propose the Discrete Extremum Constraint (DEC). It can effectively push the continuous searching space to a discrete space. Specifically, during the generation of probing adversarial perturbations $P(0)$, instead of choosing the value v of the attacked pixel based on the Gaussian distribution $\mathcal{N}(\mu, \delta)$, we randomly change v from a set of extremum $\{\beta_{min}, \beta_{max}\}$ (β_{min} and β_{max} are two fixed attack values), such that $\beta_{max} \gg \beta_{min}$. During mutation, we re-discretize the value v in $\boldsymbol{H}_i(g)$ in Eq. (14) to achieve fewer queries and better performance with the following equation:

$$v = \mathbb{I}(w \geq \mu) * (\beta_{max} - \beta_{min}) + \beta_{min}, \tag{19}$$

where $\mathbb{I}$ means the indicator function. w samples from the Gaussian distribution $\mathcal{N}(\mu, \delta)$.

In our attacking method, we fix β_{min} and β_{max} to be 0 and 255, respectively. As each gray value in an image is between 0 and 255, the setting of the attacking value as 0 or 255 can make the attacking values the farthest distance to those values of the same positions in the clean image. Then there will be a higher probability of increasing the loss in Eq. (9). Besides, the above values of our proposed DEC are easy to implement and can greatly increase the feasibility of the algorithm. For example, in real-world attacking scenarios, our DEC can directly cover the attacking pixel position with pure white and pure black colors, which can be accessed easily in the real world. After we set the value of the attacking value, then we can concentrate to find which pixels to be replaced with pure white and pure black that can change the prediction of the STR model. The brute-force searching algorithm can validate the effectiveness of our DEC.

The above DEC could efficiently decrease the search space of our method. However, the above constraint does not consider the information of the original gray value of the attacking pixel position in the clean image. Therefore, we push it furthermore to let it adaptively choose β_{min} or β_{max} according to the gray value of attacking pixel position in the target image instead of randomly. That is, the (adaptive) DEC further chooses the gray value farthest from the gray value of the attacking pixel position in the clean image. Specifically, it performs as follows:

$$w = \beta_{max} - \rho(x, y), \tag{20}$$

where $\rho(x, y)$ means the gray value of the clean image of the pixel position (x, y).

The experiments in the next section demonstrate the effectiveness of our (adaptive) DEC.

IV. Experiments

A. Experimental Settings

In this paper, following [24], we attempt to attack the CTC-based STR models (i.e., CRNN [31], Rosetta [32], and STAR [33]) and the attention-bated STR models (i.e., TRBA [58]). We select the images that can be correctly recognized by the STR model and apply our attacking method to them. If any character in the adversarial images is recognized incorrectly by the STR model, we regard it as a successful attack.

We evaluate our proposed AD^{2}E on four benchmarks:

IC13 [59] dataset contains 229 images with 1,095 text regions. After removing words that include non-alphanumeric characters and the ones with less than three characters, this dataset has 857 cropped images and each image contains one text region.

IC15 [60] dataset contains 2,077 test images. These images are cropped from 500 incidental scene images that are captured by Google Glasses without careful focusing, thus some images are blurry or of low resolution.

IIIT5K [61] dataset is collected from the internet using Google Image searches. It contains 2,000 training images and 3,000 testing images.

CUTE [56] dataset is proposed for curved text recognition. 288 testing images are cropped from 80 high-resolution natural images via the annotated bounding boxes.

We adopt two evaluation metrics: (i) success rate (**SR**), which represents the attack success rate. A successful attack happens when the original image could be recognized correctly while the adversarial image can not be recognized correctly; (ii) the average $\mathbf{L}_0$ distance, which means the number of perturbed pixels. Higher **SR** and lower $\mathbf{L}_0$ represent the better performance of the proposed method.

In our experiments, the height (H) and width (W) of the input image are set to be 32 and 100 respectively (other heights and widths can also use our method). The number of probing adversarial perturbations N_s is set as 600 and the number of the maximum iterations g_{max} is set as 300.

The proposed model is implemented in Pytorch v1.2 framework. Experiments are mainly conducted on a workstation with a 1.70 GHz Intel(R) Xeon(R) E5-2609 CPU, a single NVIDIA Titan Xp GPU, and 64G RAM.

B. Comparison With State-of-the-Art Methods

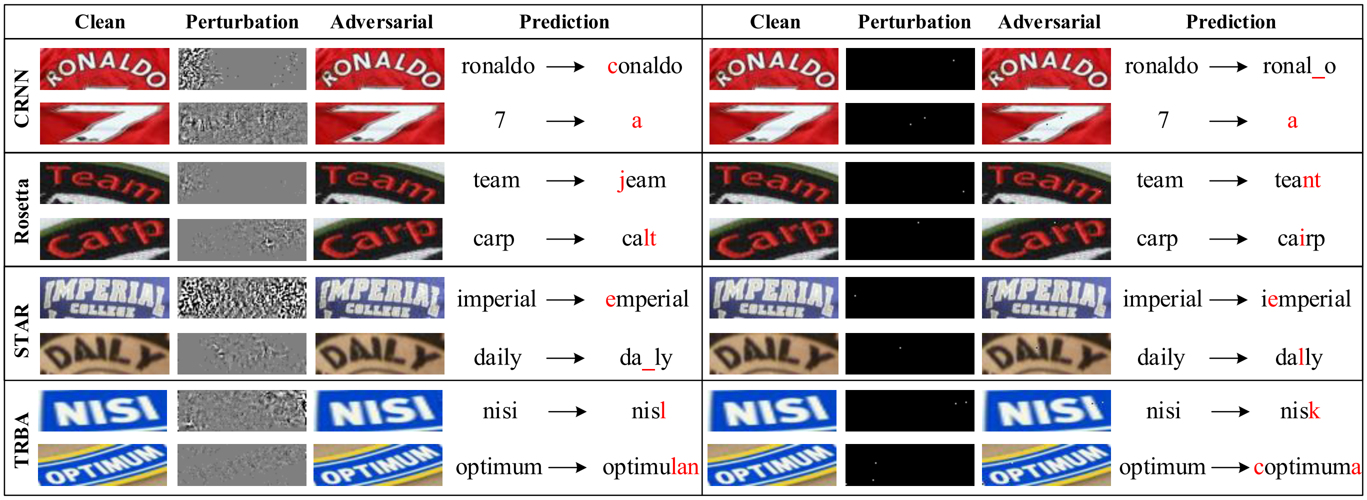
Table I shows the quantitative comparisons. Compared with Basic-0.1, BasicBinary-10, and an adaptive method [27], our proposed one achieves a comparable **SR** on the CUTE dataset. For example, our algorithm without adaptive method and DEC (termed as "Baseline" [30]) achieves a comparable success rate (**SR**), i.e., 100% vs. 99.5%, for CRNN. After applying AD^{2}E, the success rate of our method achieves 100%. Compared with the attack method in [29] when attacking Rosetta [32], our method also achieves 100% **SR** on IC13, IC15, and CUTE datasets. Compared with the attack method in [24], our method also achieves the comparable **SR** with different recognition models (e.g., CRNN [31], Rosetta [32], STAR [33], and TRBA [58]). However, the $\mathbf{L}_0$ distance of our attack approach is much smaller than [24], as the adversarial perturbations in [24] manipulate most pixels in the input image. Specifically, the $\mathbf{L_0}$ of our method with AD^{2}E for TRBA is 6 on IC13, which is much lower than that of 2,244 by [24]. Above is the comparison with the existing work of white-box attacks. As there are no black-box attacking methods for STR models, we introduce one state-of-the-art black-box method [57] of attacking face recognition. Following the work in [57], which also uses an evolutionary method and tries to perform a search in a lower dimensional space [57], we use its algorithm to apply to the STR task and provide a comparison of our method with the black-box attack. As shown in Table I, our method also achieves a comparable **SR**, while the $\mathbf{L}_0$ distance of our attack approach is much smaller than [57], which disturbs also almost all pixels in the image. The poor performance of [57] may be attributed to two reasons: Firstly, the target model of the work [57] is based on non-sequential vision tasks, which is not so fitted for the STR task. Secondly, the gray values of their search space range from 0 to 255. Although the work also uses the technique to narrow down the search space, the very large search space of the gray value would not let the method make full use of some critical pixels to attack. Therefore, they have to disturb much more pixels than our method, which uses DEC to search the discrete value to perform more efficiently. Besides, we also add a baseline (termed as "Baseline-R") that randomly perturbs 7-10 pixels in each image. The coordinates of attacked pixel position (x, y) and the attacked pixel value v are all random such that $x \sim \mathcal{U}(0, W)$, $y \sim \mathcal{U}(0, H)$, and $v \sim \mathcal{U}(0, 255)$, where $\mathcal{U}$ represents the uniform distribution. For a fair comparison, the number of iterations for Baseline-R is the same as that in our proposed method, i.e. 300. We find that 10 pixels random perturbations perform best in the baselines with regard to **SR**, and we report the results in Table I. As shown in Table I, the Baseline-R achieves a much lower **SR** with disturbing even more pixels than our method. The results also verify the efficacy of our proposed algorithm.

Some qualitative results are shown in Fig. 6. We can observe that the adversarial perturbations in [24] cover a large number of pixels while our method only alters a few pixels. The adversarial examples generated by our method are also hard to be perceived by humans in complicated scenarios. Although our attacking method does not rely on the knowledge of the target model, it can achieve comparable **SR** compared with existing white-box STR attacking methods. It is because our method tends to find the most important pixels that can change the recognition outputs.

TABLE I

QUANTITATIVE COMPARISONS OF STR ATTACK APPROACHES ON FOUR BENCHMARK DATASETS WITH STATE-OF-THE-ART METHODS. * MEANS RESULTS WITH OUR REIMPLEMENTATION. SR MEANS THE ATTACK SUCCESS RATE. L_0 MEANS THE L_0 DISTANCE BETWEEN THE ORIGINAL IMAGE AND THE ADVERSARIAL IMAGE. THE RESULTS SHOW THAT WE CAN ACHIEVE NEAR **SR** WITH MUCH FEWER PIXELS AND LESS KNOWLEDGE OF THE TARGET MODEL

Method		IC13		IC15		IIIT5K		CUTE	
		SR ↑	L_0 ↓	SR ↑	L_0 ↓	SR ↑	L_0 ↓	SR ↑	L_0 ↓
White-box	Basic_0.1 [27]	–	–	–	–	99.7	–	–	–
	Basic_1 [27]	–	–	–	–	95.4	–	–	–
	Basic_10 [27]	–	–	–	–	39.1	–	–	–
	BasicBinary_3 [27]	–	–	–	–	**100**	–	–	–
	BasicBinary_5 [27]	–	–	–	–	**100**	–	–	–
	BasicBinary_10 [27]	–	–	–	–	**100**	–	–	–
	Adaptive (CRNN) [27]	–	–	–	–	**100**	–	–	–
	Yang et al. (ASTER) [29]	**100**	–	**100**	–	**100**	–	**100**	–
	Xu et al. (CRNN) [24]	99.9	–	**100**	–	99.9	–	**100**	–
	Xu et al. (Rosetta) [24]	99.8	–	99.9	–	99.6	–	99.2	–
	Xu et al. (STAR) [24]	99.9	–	**100**	–	99.8	–	**100**	–
	Xu et al. (TRBA) [24]	99.8	–	99.9	–	99.4	–	99.2	–
	Xu et al. (CRNN) * [24]	99.5	2,076	99.9	1,134	99.9	1,845	**100**	2,054
	Xu et al. (Rosetta) * [24]	99.7	1,899	**100**	1,131	**100**	1,709	**100**	2,135
	Xu et al. (STAR) * [24]	96.2	2,176	99.7	2,290	98.1	1,916	**100**	2,290
	Xu et al. (TRBA) * [24]	93.5	2,244	99.6	2,346	97.1	1,941	99.1	2,346
Black-box	Dong et al. (CRNN) * [57]	**100**	3,192	**100**	3,180	**100**	3,196	**100**	3,183
	Dong et al. (Rosetta) * [57]	**100**	3,200	**100**	3,168	**100**	3,195	**100**	3,200
	Dong et al. (STAR) * [57]	**100**	3,196	**100**	3,189	**100**	3,194	**100**	3,200
	Dong et al. (TRBA) * [57]	99.9	3,189	99.8	3,188	99.7	3,195	99.5	3,200
	Baseline-R (CRNN)	41.5	10	64.4	10	45.7	10	57.8	10
	Baseline-R (Rosetta)	39.7	10	57.1	10	43.9	10	57.7	10
	Baseline-R (STAR)	34.5	10	52.0	10	37.7	10	52.7	10
	Baseline-R (TRBA)	26.3	10	44.4	10	29.4	10	42.1	10
	Baseline (CRNN) [30]	98.0	7	99.4	7	98.3	**7**	99.5	6
	Baseline (Rosetta) [30]	99.6	6	**100**	7	98.8	**7**	**100**	6
	Baseline (STAR) [30]	97.7	7	97.6	7	96.7	**7**	98.5	**4**
	Baseline (TRBA) [30]	95.7	7	97.2	7	94.7	**7**	97.2	7
	Ours (CRNN) w/ AD^{2}E	99.7	**5**	**100**	**5**	99.8	**7**	**100**	7
	Ours (Rosetta) w/ AD^{2}E	**100**	**5**	**100**	7	99.7	**7**	**100**	**4**
	Ours (STAR) w/ AD^{2}E	98.5	6	99.0	7	98.1	**7**	99.5	6
	Ours (TRBA) w/ AD^{2}E	97.5	6	98.5	7	97.2	**7**	**100**	7



(a) Results from Xu et al. [24]

(b) Ours

Fig. 6. Qualitative comparisons between the attacking method in [24] (a) and ours (b). Note that we scaled the perturbation for better visualization by multiplying a value of 20. '_' represents that there is no word. We can observe that the adversarial perturbations in [24] cover a large number of pixels while our method only alters a few pixels.

TABLE II

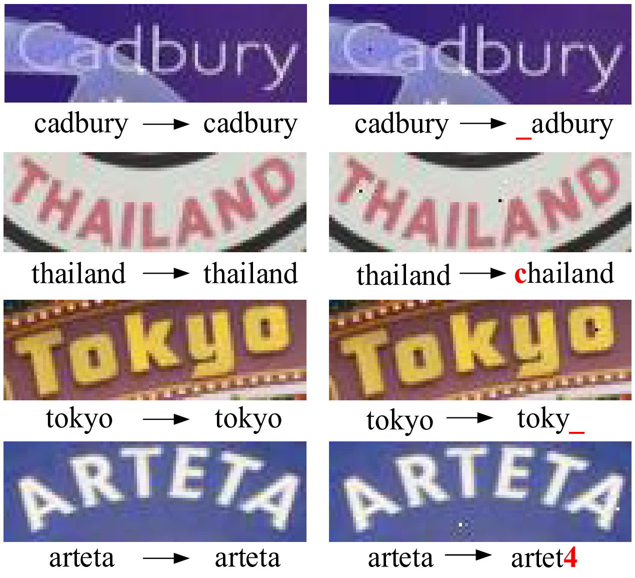
PERFORMANCE COMPARISON ON ATTACKING REAL-WORLD COMMERCIAL SYSTEM (BAIDU OCR). THE ADVERSARIAL EXAMPLES ARE GENERATED TO ATTACK THE OFF-LINE STR MODEL OF ROSETTA AND TRBA. THE LAST ROW (i.e., OURS ($N_p = [1, 7]$)) MEANS THE SUCCESS RATE OF THE UNION SET OF THE FIRST SEVEN ROWS FOR OURS. THE RESULTS SHOW THAT THE TRANSFERABILITY OF OUR METHOD IS MUCH BETTER THAN THE WORK [24]

Method	Rosetta		TRBA	
	SR (↑)	L_0 (↓)	SR (↑)	L_0 (↓)
Xu et al. [24]	1.8	2,135	6.3	2,346
Ours (N_p = 1)	6.4	**1**	3.6	**1**
Ours (N_p = 2)	12.7	2	9.9	2
Ours (N_p = 3)	10	3	9.9	3
Ours (N_p = 4)	17.3	4	13.5	4
Ours (N_p = 5)	20.9	5	13.5	5
Ours (N_p = 6)	8.2	6	20.7	6
Ours (N_p = 7)	11.8	7	18	7
Ours (N_p = [1, 7])	**43.6**	3.6	**40.5**	4

C. Attack on Commercial Scene Text Recognition System

To verify the transferability of our method, we further investigate the attacking performance on a real-world commercial STR system, Baidu OCR, via the generated adversarial samples. Specifically, we select the images that can be correctly recognized by the Baidu OCR engine[1] in the CUTE dataset. Then, we utilize our proposed method to generate adversarial examples that can fool the off-line STR models Rosetta [32] and TRBA [58] from the above-selected images. We call the set of adversarial samples which are generated by our attack method as $\mathcal{S}_a$. Meanwhile, we also test the attack method in [24], and name the set of the adversarial samples generated by [24] as $\mathcal{S}_b$. Next, we obtain the common image name set, via $\mathcal{S} = \mathcal{S}_a \cap \mathcal{S}_b$, to ensure the same number of total adversarial examples for a fair comparison. Finally, we use Baidu OCR to recognize these adversarial examples to test whether they could transfer attacking Baidu OCR successfully.

As shown in Table II, with the CTC-based STR model Rosetta [32], the success rate (**SR**) of our adversarial samples reaches the best value of 20.9% when $N_p = 5$, which has an improvement of 14.0% compared with the state-of-the-art attacking method [24]. When the number of perturbed pixels increases from 1 to 7, the **SR** can increase to 43.6%, which is about 41.8% higher than [24]. Meanwhile, the weighted average L_0 distance of the adversarial perturbations from our method is lower than that in [24] (3.6 vs. 2,135). With the attention-based STR model TRBA, the success rate (**SR**) of our method achieves the best value of 20.7% when $N_p = 6$, which has an improvement of 14.6% compared with [24]. When the number of perturbed pixels increases from 1 to 7, the **SR** can achieve 40.5%, which is about 38.7% higher than [24]. The results of attacking the two different models show that our model has quite good transferability.



(a) Results from Xu et al. [24] (b) Ours

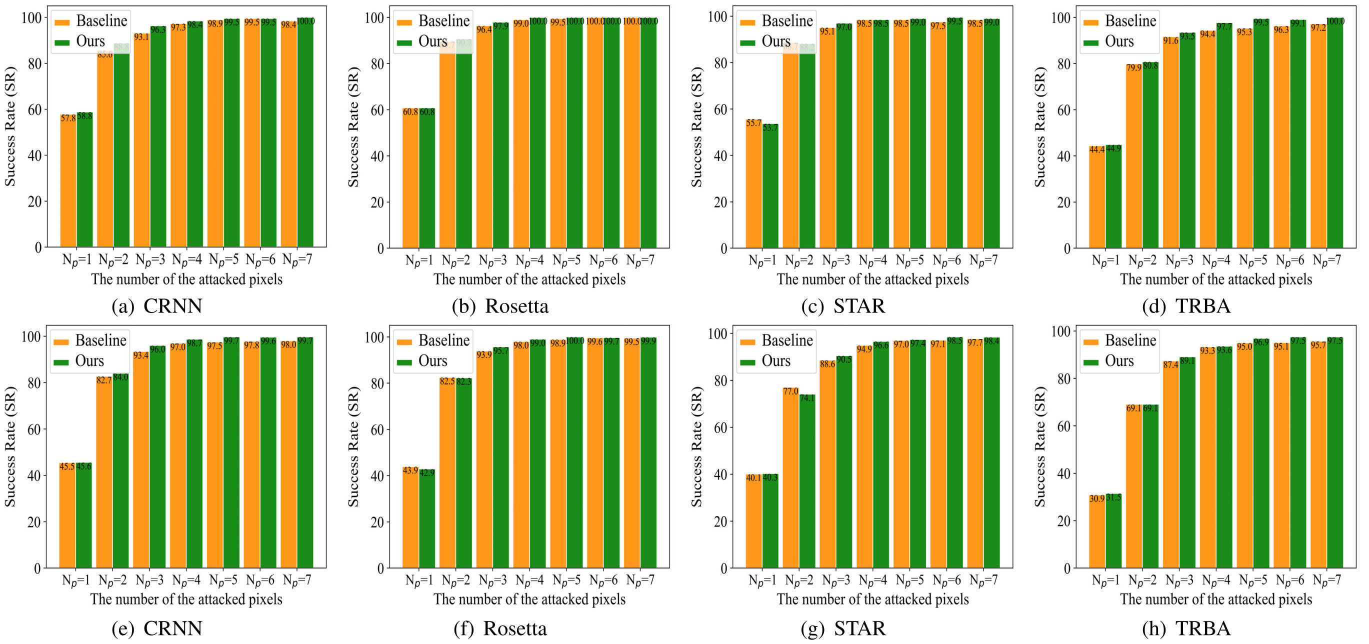
Fig. 7. Adversarial attacking on Baidu OCR. The left columns are the adversarial results generated from Xu et al. [24], while the right columns are ours. We can observe that some adversarial images generated by [24] fail to transfer attacking Baidu OCR while our model is able to attack successfully.

Some qualitative results are shown in Fig. 7 to demonstrate the model transferability on Baidu OCR. The superiority lies that we can modify a very few pixels to transfer attacking Baidu OCR while the work [24] modifies much more pixels to fail the task. Specifically, the white-box algorithm disturbs many pixels and each pixel has a very small change. Thus the contribution from each pixel is quite small. It may thus fail to fool the new model when it is greatly different from the previous model. Only some of the disturbing pixels may help to change the results (we call these pixels "positive pixels") while others may not. These small number of positive pixels are not able to change the results. Besides, although the contributions of all the positive pixels are enough, some pixels may help to let the results get back (we call these pixels "negative pixels"). It fails to transfer attacking if the negative contributions of the negative pixels are stronger than those of positive pixels. On the contrary, our method tries to find the most important pixels to change the recognition results and the extreme value constraint guarantees the farthest distance between the new value and the original value. Thus transfer attacking success rate of our method is higher as expected.

D. Ablation Studies

*1) **Influence of the Number of Perturbed Pixels:*** As shown in Fig. 8, with the increase of the number of perturbed pixels, the success rate (**SR**) of our attack method also increases until the plateau. Experimental results on four STR models (CRNN, Rosetta, STAR, and TRBA) show that when altering only one pixel, the **SR** of our method is still quite satisfactory on multiple datasets. For example, it is easy to attack CUTE which has complicated backgrounds or layouts by over 40%. When the number of perturbed pixels increases to 7, the **SR** can reach about 99% on all models. It indicates that altering 7 pixels is sufficient to generate decent adversarial examples.

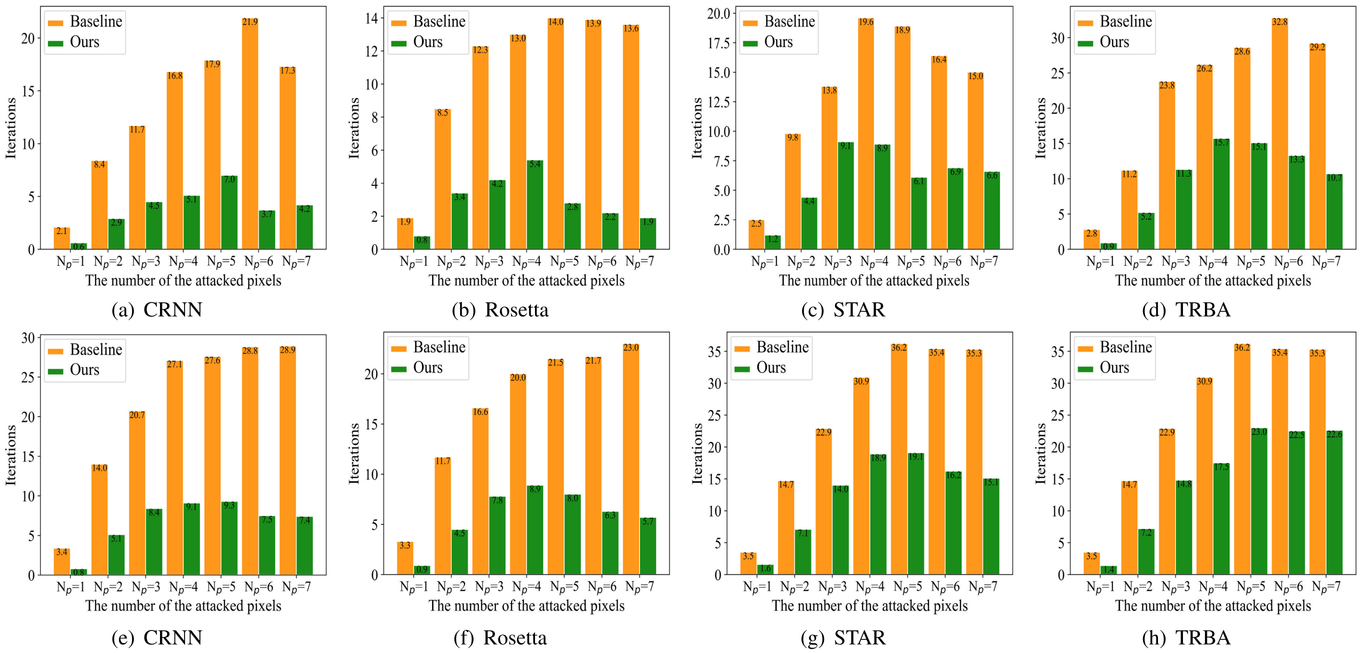
[1] The API toolkit is available at *https://cloud.baidu.com/doc/OCR/OCR-API.html*



(a) CRNN (b) Rosetta (c) STAR (d) TRBA

(e) CRNN (f) Rosetta (g) STAR (h) TRBA

Fig. 8. The success rate (**SR**) against the number of the attacked pixels for the dataset CUTE (top row) and IC13 (bottom row). With the increasing number of perturbed pixels, the **SR** of our attack method keeps increasing until the plateau. Besides, our method always achieves much better **SR** on multiple datasets with different numbers of perturbation pixels than the Baseline.



(a) CRNN (b) Rosetta (c) STAR (d) TRBA

(e) CRNN (f) Rosetta (g) STAR (h) TRBA

Fig. 9. The average number of iterations against the number of the attacked pixels for the dataset CUTE (top row) and IC13 (bottom row). When the number of attacked pixels increases from 1 to 3(4, 5), the number of iterations of our method also increases stably. Then the average number of iterations decreases when the number of attacked pixels increases from 3(4, 5) to 7. It indicates that the number of perturbed pixels in AD^{2}E can influence the optimization, and thus result in different iterations. Besides, our method could also reduce the number of iterations compared with the Baseline.

Fig. 9 also shows that with the change of the number of perturbed pixels, the average number of **iterations** for the successful attack will also change. For example, for the STR model of Rosetta on the dataset CUTE, the average iterations of our method first increase when the number of attacked pixels increases from 1 to 4. Then the average iterations decrease with the number of attacked pixels increasing from 4 to 7. It indicates that the number of perturbed pixels in our method influences the optimization, and thus results in different iterations. Although the average number of the iterations is the lowest when $N_p = 1$, most adversarial examples generated by the maximum iterations can not fool the STR model, indicating few pixels attack might not be sufficient for complicated scene text images.

2) ***Influence of Adaptive-Discrete Differential Evolution:*** We can compare the Influence of AD^{2}E with Baseline [30] in Fig. 8. As shown in Fig. 8, when we use the AD^{2}E method through optimization, our method always achieves

TABLE III

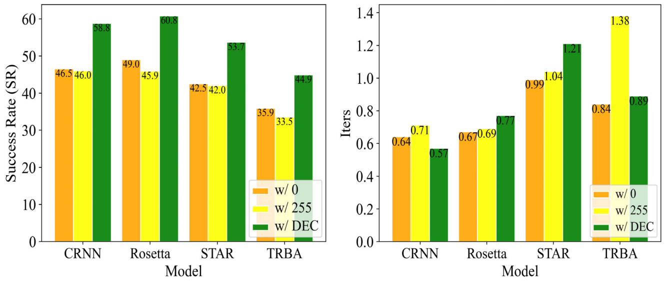
INFLUENCE OF THE ADAPTIVE METHOD. SR MEANS THE ATTACK SUCCESS RATE. ITERATIONS MEAN THE AVERAGE ITERATIONS FOR THE SUCCESSFUL ATTACKING IMAGE. RUNTIME MEANS THE WHOLE SECONDS FOR ATTACKING THE DATASET CUTE. THE RESULTS SHOW THAT OUR ADAPTIVE METHOD CAN ACHIEVE NEAR **SR** WITH MUCH LESS TIME

Model	**Only DDE**			**Only DE**			**Ours**		
	SR (↑)	Iterations (↓)	Runtime (↓)	SR (↑)	Iterations (↓)	Runtime (↓)	SR (↑)	Iterations (↓)	Runtime (↓)
CRNN	**100**	5.01	$1.85*10^4$	**100**	6.63	$1.02*10^4$	**100**	**4.16**	$\mathbf{1.61*10^3}$
Rosetta	**100**	**1.11**	$4.45*10^3$	**100**	2.69	$4.83*10^3$	**100**	1.86	$\mathbf{1.29*10^3}$
STAR	99	7.23	$4.71*10^4$	**99.5**	8.07	$1.69*10^4$	99	**6.61**	$\mathbf{7.22*10^3}$
TRBA	99.5	**9.87**	$1.79*10^5$	**100**	15.82	$2.78*10^4$	**100**	10.69	$\mathbf{1.63*10^4}$

much better **SR** on multiple datasets with different numbers of perturbation pixels than the Baseline. Specifically, for CUTE, our method obtains 100% **SR** while the Baseline gets 98.4% with $N_p = 7$ on CRNN. On Rosetta, our method achieves 100% **SR**, which is 1.0% better than that of the Baseline with $N_p = 4$. Besides, on STAR, our model has an improvement of 2.0% in **SR** than the Baseline, which achieves 97.5% with $N_p = 6$. Moreover, on attention-based TRBA, the **SR** of our method achieves 100% **SR**, exceeding the Baseline by 2.8% with $N_p = 7$. As for IC13, our approach achieves 100% **SR** on Rosetta, while the Baseline only achieves 98.9% with $N_p = 5$.

Additionally, our method can also reduce the number of iterations, as shown in Fig. 9. In most settings, compared with Baseline, the proposed method could always reduce the number of iterations by a factor of over 2/3. For example, for CUTE on CRNN, our average iterations of successful attacks reduce from 21.9 to 3.1 with 6 pixels. Compared with the Baseline, our approach decreases the iterations by a factor of over 3/4. On Rosetta, the average iterations decline from 13.6 to 1.9 with 7 pixels by over 86%. For STAR, the average iterations of successful attacks reduce from 18.9 to 6.1 with 5 pixels. Meanwhile, the average iterations of successful attacking decrease from 2.8 to 0.9 with 1 pixel for TRBA. As for IC13, the average iterations of successful attacking for our method are 5.7 on Rosetta with $N_p = 7$, while the Baseline uses 23.0 iterations.

3) Influence of the Adaptive Method: We perform experiments to verify the influence of the adaptive method to choose DE and DDE. We use the dataset CUTE and attack 7 pixels with only DDE, only DE, and our AD^{2}E. All three experiments utilize DEC for a fair comparison. As shown in Table III, for success rate, the three optimization methods are near. For iterations, compared with only DE, ours requires fewer queries. It is because the adaptive method could accelerate the convergence speed when performing the DDE. Compared with only DDE, the average iterations of our method for CRNN and STAR are even better. The reason may be that our method is not easy to trap into local optimum as we have the DE choice compared with only DDE. The average iterations of our method for Rosetta and TRBA are only a little higher than only DDE, with the runtime of our method saving more than 70% and 90%, respectively. For the runtime, we could find that the runtime of our method is the lowest of all four models. The reason is three folds: (1) When the adaptive step finds that there is an "expectant" probing adversarial perturbation, it chooses to use DDE for accelerating convergence speed. (2) When the adaptive step finds that the generation traps in the extreme local extremum, it lets the optimization skip immediately to save the runtime. (3) When the adaptive step finds that the current generation may be far from finding the adversarial example, it chooses to use DE, which could take advantage of the parallel technique to accelerate the speed. Because of (1) and (2), our method would be faster than that only with DE, as we could use fewer iterations to find the solution and skip unnecessary iterations. Because of (2) and (3), our adaptive method will be much faster than that only with DDE, whose strategy of updating the probing adversarial perturbations individual by individual is very time-consuming.



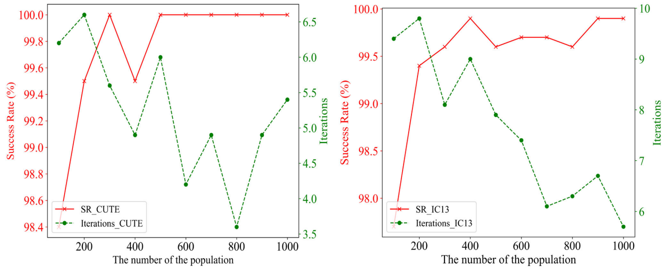
(a) Success rate (**SR**)

(b) Iterations

Fig. 10. The average success rate (**SR**) and iterations for whether adding 0 or 255 constraints on the dataset CUTE. It implies that DEC is reasonable for getting a better SR than that only with 0 or 255.

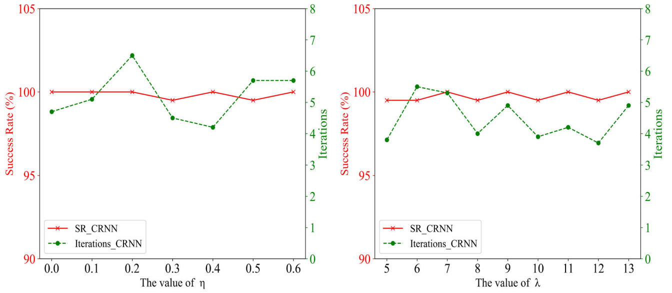
4) Influence of the Discrete Extremum Constraint: In DEC, when we fix the value of attack pixels to 0 or 255, it yields slightly worse **SR** and requires more iterations to perform the attack, as shown in Fig. 10. Specifically, our method with DEC (adaptively to choose 0 or 255) can achieve **SR** of 58.8%, while 0 or 255 can only get **SR** of 49.0% and 45.9% for Rosetta in Fig. 10 (a). It indicates that 0 and 255 are complementary for a successful attack. Besides, iterations for these three settings are quite close. As shown in Fig. 10 (b), the iterations with the constraint of 0, 255, and DEC for CRNN are 0.64, 0.71, and 0.57, respectively. It implies that the density of successful attacking pixel positions for 0 or 255 is close. Therefore, DEC is reasonable for getting a reasonable **SR** and an acceptable number of iterations.

5) Influence of the Number of the Probing Adversarial Perturbations: In the process of optimization, the number of the probing adversarial perturbations, N_s, plays an important role. As shown in Fig. 11, experimental results indicate that the **SR** of our attack method can be promoted with the increase of N_s, until it achieves a stable **SR**. For instance, the **SR** reaches 100% after N_s increases to 500 for the dataset CUTE. Meanwhile, the **SR** reaches 99.6% when $N_s = 400$ for the



(a) CUTE (b) IC13

Fig. 11. The **SR** and iterations against the number of probing adversarial perturbations. The target model is CRNN [31] with datasets CUTE and IC13. The results indicate that the **SR** of our method can be improved with the increase of N_s before it plateaus. Besides, the iterations first increase and then decrease with the increase of N_s.



(a) (b)

Fig. 12. (a) The effect of η on success rate (**SR**) and the number of iterations on dataset CUTE for model CRNN (λ is set as 11). (b) The effect of λ (from 5 to 13) on success rate (**SR**) and the number of iterations on the dataset CUTE for the model CRNN (η is set as 0.4). The results show that our method is robust to the values of η and λ.

dataset IC13. Besides, we can further observe that it first increases after N_s increases from 100 to 300, then decreases after N_s increases from 300 to 400, and finally increases to convergence after N_s increases from 400 to 1,000 for the dataset CUTE. Besides, the number of iterations first increases when N_s increases from 100 to 200, then decreases when the number of probing adversarial perturbations increases from 400 to 700 for the dataset IC13. The reason for increasing is that some images that are harder to attack are attacked successfully. And the reason for decreasing is that the search range increasing by one generation helps to reduce the number of iterations. Therefore, we set N_s to be 600 in our experiments, because the **SR** and the number of iterations have become stable after N_s increases to 600.

E. The Parameter Sensitivity Analysis

To further verify the effectiveness and analyze the parameter sensitivity, we conduct the seven-pixel-attacking experiments by changing the value of parameters η and λ. Fig. 12 (a) shows the success rate and the number of iterations with changing values of η. We observe that the **SR** and iterations perform the best when setting η as 0.4. Meanwhile, the **SR** and iterations have not changed so much. Similarly, as shown in Fig. 12 (b), we observe that $\lambda = 11$ leads to the optimal **SR** and iterations. Meanwhile, the **SR** and iterations have not changed so much for these values. The experiments demonstrate that our method is robust to the values of η and λ.

V. Conclusion

In this paper, we propose a novel black-box attacking method AD^{2}E for the CTC-based and attention-based STR to protect the privacy of text information in natural scene images. This approach does not require prior knowledge of network configurations (e.g., network structure, gradients, etc.), it only employs the probability distribution of the network output to perform the sequential vision attack via perturbing a very few pixels. Extensive experiments on four real-world datasets have demonstrated the effectiveness of our proposed attack method. Especially, by pushing the continuous searching space to a discrete space using DEC, we can greatly reduce the number of iterations and the number of queries to the model. Experimental results also show that the adversarial examples generated by our method can better fool the commercial STR system (i.e., Baidu OCR), compared with existing STR attack methods. In the future, we will work on designing adversarial attack approaches without knowledge of probability distribution, to better protect the private text information in images and videos. We hope this work could raise public concerns about using a few pixels to protect the free scene text data from unauthorized or even illegal privacy infringement.

References

[1] S. Karaoglu, R. Tao, T. Gevers, and A. W. M. Smeulders, "Words matter: Scene text for image classification and retrieval," *IEEE Trans. Multimedia*, vol. 19, no. 5, pp. 1063–1076, May 2017.

[2] J. Wang, J. Tang, and J. Luo, "Multimodal attention with image text spatial relationship for OCR-based image captioning," in *Proc. 28th ACM Int. Conf. Multimedia*, Oct. 2020, pp. 4337–4345.

[3] F. Liu, G. Xu, Q. Wu, Q. Du, W. Jia, and M. Tan, "Cascade reasoning network for text-based visual question answering," in *Proc. 28th ACM Int. Conf. Multimedia*, Oct. 2020, pp. 4060–4069.

[4] D. Kiela et al., "The hateful memes challenge: Detecting hate speech in multimodal memes," in *Proc. NeurIPS*, vol. 33, 2020, pp. 1–14.

[5] J. Yu, B. Zhang, Z. Kuang, D. Lin, and J. Fan, "iPrivacy: Image privacy protection by identifying sensitive objects via deep multi-task learning," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 5, pp. 1005–1016, May 2017.

[6] J. Du, C. Jiang, K. C. Chen, Y. Ren, and H. V. Poor, "Community-structured evolutionary game for privacy protection in social networks," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 3, pp. 574–589, Mar. 2017.

[7] X. Tian, P. Zheng, and J. Huang, "Robust privacy-preserving motion detection and object tracking in encrypted streaming video," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 5381–5396, 2021.

[8] S. Zhang, Y. Liu, L. Jin, Y. Huang, and S. Lai, "EnsNet: Ensconce text in the wild," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 801–808.

[9] C. Liu, Y. Liu, L. Jin, S. Zhang, C. Luo, and Y. Wang, "EraseNet: End-to-end text removal in the wild," *IEEE Trans. Image Process.*, vol. 29, pp. 8760–8775, 2020.

[10] B. Conrad and P.-I. Chen, "Two-stage seamless text erasing on real-world scene images," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2021, pp. 1309–1313.

[11] K. Inai, M. Pålsson, V. Frinken, Y. Feng, and S. Uchida, "Selective concealment of characters for privacy protection," in *Proc. 22nd Int. Conf. Pattern Recognit.*, Aug. 2014, pp. 333–338.

[12] Y. Zhong and W. Deng, "Towards transferable adversarial attack against deep face recognition," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 1452–1466, 2020.

[13] H. Kwon, Y. Kim, H. Yoon, and D. Choi, "Selective audio adversarial example in evasion attack on speech recognition system," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 526–538, 2019.

[14] H. Zhang, Y. Avrithis, T. Furon, and L. Amsaleg, "Walking on the edge: Fast, low-distortion adversarial examples," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 701–713, 2020.

[15] M. Shen, H. Yu, L. Zhu, K. Xu, Q. Li, and J. Hu, "Effective and robust physical-world attacks on deep learning face recognition systems," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 4063–4077, 2021.

[16] H. Wang, G. Wang, Y. Li, D. Zhang, and L. Lin, "Transferable, controllable, and inconspicuous adversarial attacks on person re-identification with deep mis-ranking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1–10.

[17] H. Wang, G. Li, X. Liu, and L. Lin, "A Hamiltonian Monte Carlo method for probabilistic adversarial attack and learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 4, pp. 1725–1737, Apr. 2020.

[18] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. ICLR*, 2018, pp. 1–23.

[19] H. Zhang, Y. Yu, J. Jiao, E. P. Xing, L. E. Ghaoui, and M. I. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *Proc. ICML*, 2019, pp. 1–11.

[20] H. Wang and Y. Wang, "Self-ensemble adversarial training for improved robustness," in *Proc. ICLR*, 2022, pp. 1–8.

[21] H. Zhang and J. Wang, "Towards adversarially robust object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 421–430.

[22] J. H. Metzen, M. C. Kumar, T. Brox, and V. Fischer, "Universal adversarial perturbations against semantic image segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2774–2783.

[23] B. Wang, M. Zhao, W. Wang, X. Dai, Y. Li, and Y. Guo, "Adversarial analysis for source camera identification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 11, pp. 4174–4186, Nov. 2020.

[24] X. Xu, J. Chen, J. Xiao, L. Gao, F. Shen, and H. T. Shen, "What machines see is not what they get: Fooling scene text recognition models with adversarial text images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 12301–12311.

[25] C. Song and V. Shmatikov, "Fooling OCR systems with adversarial text images," 2018, *arXiv:1802.05385*.

[26] L. Chen, J. Sun, and W. Xu, "FAWA: Fast adversarial watermark attack on optical character recognition (OCR) systems," in *Proc. ECML-PKDD*, 2020, pp. 547–563.

[27] X. Yuan, P. He, X. Lit, and D. Wu, "Adaptive adversarial attack on scene text recognition," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Jul. 2020, pp. 358–363.

[28] X. Xu, J. Chen, J. Xiao, Z. Wang, Y. Yang, and H. T. Shen, "Learning optimization-based adversarial perturbations for attacking sequential recognition models," in *Proc. 28th ACM Int. Conf. Multimedia*, 2020, pp. 2802–2822.

[29] M. Yang, H. Zheng, X. Bai, and J. Luo, "Cost-effective adversarial attacks against scene text recognition," in *Proc. ICPR*, 2021, pp. 2368–2374.

[30] Y. Xu, P. Dai, and X. Cao, "Less is better: Fooling scene text recognition with minimal perturbations," in *Proc. ICONIP*, 2021, pp. 537–544.

[31] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 11, pp. 2298–2304, Nov. 2017.

[32] F. Borisyuk, A. Gordo, and V. Sivakumar, "Rosetta: Large scale system for text detection and recognition in images," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 71–79.

[33] W. Liu, C. Chen, K.-Y. K. Wong, Z. Su, and J. Han, "STAR-Net: A spatial attention residue network for scene text recognition," in *Proc. BMVC*, 2016, p. 7.

[34] W. Hu, X. Cai, J. Hou, S. Yi, and Z. Lin, "GTC: Guided training of CTC towards efficient and accurate scene text recognition," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 7, 2020, pp. 11005–11012.

[35] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. ICML*, 2006, pp. 369–376.

[36] C.-Y. Lee and S. Osindero, "Recursive recurrent nets with attention modeling for OCR in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2231–2239.

[37] Z. Cheng, F. Bai, Y. Xu, G. Zheng, S. Pu, and S. Zhou, "Focusing attention: Towards accurate text recognition in natural images," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5086–5094.

[38] Z. Cheng, Y. Xu, F. Bai, Y. Niu, S. Pu, and S. Zhou, "AON: Towards arbitrarily-oriented text recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5571–5579.

[39] P. Dai, H. Zhang, and X. Cao, "SLOAN: Scale-adaptive orientation attention network for scene text recognition," *IEEE Trans. Image Process.*, vol. 30, pp. 1687–1701, 2021.

[40] X. Yang, D. He, Z. Zhou, D. Kifer, and C. L. Giles, "Learning to read irregular text with attention mechanisms," in *Proc. IJCAI*, 2017, pp. 3280–3286.

[41] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai, "ASTER: An attentional scene text recognizer with flexible rectification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2035–2048, Sep. 2019.

[42] C. Szegedy et al., "Intriguing properties of neural networks," in *Proc. ICLR*, 2014, pp. 1–10.

[43] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *Stat*, vol. 1050, p. 20, Dec. 2015.

[44] I. G. Alexey Kurakin and S. Bengio, "Adversarial examples in the physical world," in *Proc. ICLR Workshop*, 2017, pp. 1–14.

[45] Y. Dong et al., "Boosting adversarial attacks with momentum," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9185–9193.

[46] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2574–2582.

[47] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 39–57.

[48] Y. Zhao, H. Yan, and X. Wei, "Object hider: Adversarial patch attack against object detectors," 2020, *arXiv:2010.14974*.

[49] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 828–841, Oct. 2017.

[50] S. N. Shukla, A. K. Sahu, D. Willmott, and Z. Kolter, "Simple and efficient hard label black-box adversarial attacks in low query budget regimes," in *Proc. SIGKDD*, 2021, pp. 1461–1469.

[51] M. Zha, G. Meng, C. Lin, Z. Zhou, and K. Chen, "RoLMA: A practical adversarial attack against deep learning-based LPR systems," in *Proc. Int. Conf. Inf. Secur. Cryptol.*, 2019, pp. 101–117.

[52] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[53] A. Qing, "Dynamic differential evolution strategy and applications in electromagnetic inverse scattering problems," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 1, pp. 116–125, Jan. 2005.

[54] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: Past, present, and future," *Multimedia Tools Appl.*, vol. 80, no. 5, pp. 8091–8126, 2021.

[55] B. Hegerty, C.-C. Hung, and K. Kasprak, "A comparative study on differential evolution and genetic algorithms for some combinatorial problems," in *Proc. Mexican Int. Conf. Artif. Intell.*, vol. 9, 2009, p. 13.

[56] A. Risnumawan, P. Shivakumara, C. S. Chan, and C. Tan, "A robust arbitrary text detection system for natural scene images," *Exp. Syst. Appl.*, vol. 41, no. 18, pp. 8027–8048, 2014.

[57] Y. Dong et al., "Efficient decision-based black-box adversarial attacks on face recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7714–7722.

[58] J. Baek et al., "What is wrong with scene text recognition model comparisons? Dataset and model analysis," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4714–4722.

[59] D. Karatzas et al., "ICDAR 2013 robust reading competition," in *Proc. 12th Int. Conf. Document Anal. Recognit.*, Aug. 2013, pp. 1484–1493.

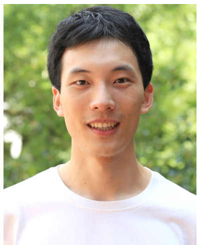
[60] D. Karatzas et al., "ICDAR 2015 competition on robust reading," in *Proc. 13th Int. Conf. Document Anal. Recognit. (ICDAR)*, Aug. 2015, pp. 1156–1160.

[61] A. Mishra, K. Alahari, and C. Jawahar, "Scene text recognition using higher order language priors," in *Proc. Brit. Mach. Vis. Conf.*, 2012, pp. 1–11.

Yikun Xu received the Ph.D. degree from the Institute of Information Engineering, Chinese Academy of Sciences. His current research interests include AI security, adversarial learning, and privacy protection. He is a reviewer for conferences and journals such as AAAI, CVPR, ICCV, and IEEE TRANSACTIONS ON MULTIMEDIA.

Pengwen Dai received the Ph.D. degree from the Institute of Information Engineering, Chinese Academy of Sciences. He is currently an Assistant Professor with the School of Cyber Science and Technology, Sun Yat-sen University. His current research interests include pattern recognition, multimedia understanding, and AI security.

Zekun Li is currently pursuing the Ph.D. degree in computer science with the University of Minnesota (UMN). Her research interests include the automatic understanding of historical maps with computer vision and natural language processing approaches. She has worked on text detection of historical map labels, connecting separated text labels, linking recognized place names to existing knowledge bases (entity linking), and inferencing label types (entity typing).



Hongjun Wang (Student Member, IEEE) received the B.E. and M.E. degrees from Sun Yat-sen University in 2018 and 2021, respectively. He is currently pursuing the Ph.D. degree in statistics with the University of Hong Kong. His research interests include open-set recognition, out-of-distribution detection, domain adaptation, and trustworthy machine learning. He has been serving as a reviewer for numerous academic journals and conferences such as AAAI, NeurIPS, ICML, ICLR, CVPR, *Machine Learning*, IEEE TRANSACTIONS ON IMAGE PROCESSING, and *Neurocomputing*.

Xiaochun Cao (Senior Member, IEEE) received the B.E. and M.E. degrees in computer science from Beihang University, China, and the Ph.D. degree in computer science from the University of Central Florida, Orlando, USA. After graduation, he spent more than three years at ObjectVideo Inc., as a Research Scientist. From 2008 to 2012, he was a Professor with Tianjin University, Tianjin, China. He has been a Professor with the Institute of Information Engineering, Chinese Academy of Sciences, since 2012. He is currently with the School of Cyber Security, Sun Yat-sen University, China. He is on the Editorial Boards of the IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON MULTIMEDIA, and IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY. From 2004 and 2010, he was a recipient of the Piero Zamperoni Best Student Paper Award from the International Conference on Pattern Recognition.