Computers, Materials & Continua CMC, vol.65, no.2, pp.1437-1452, 2020

Adversarial Attacks on License Plate Recognition Systems

Zhaoquan Gu[1], Yu Su[1], Chenwei Liu[1], Yinyu Lyu[1], Yunxiang Jian[1], Hao Li[2], Zhen Cao[3] and Le Wang[1, *]

Abstract: The license plate recognition system (LPRS) has been widely adopted in daily life due to its efficiency and high accuracy. Deep neural networks are commonly used in the LPRS to improve the recognition accuracy. However, researchers have found that deep neural networks have their own security problems that may lead to unexpected results. Specifically, they can be easily attacked by the adversarial examples that are generated by adding small perturbations to the original images, resulting in incorrect license plate recognition. There are some classic methods to generate adversarial examples, but they cannot be adopted on LPRS directly. In this paper, we modify some classic methods to generate adversarial examples that could mislead the LPRS. We conduct extensive evaluations on the HyperLPR system and the results show that the system could be easily attacked by such adversarial examples. In addition, we show that the generated images could also attack the black-box systems; we show some examples that the Baidu LPR system also makes incorrect recognitions. We hope this paper could help improve the LPRS by realizing the existence of such adversarial attacks.

Keywords: License plate recognition system, adversarial examples, deep neural networks.

1 Introduction

License plate recognition systems (LPRSs) have brought great commercial values because of the convenience and high accuracy in recognizing the license number. Nowadays, the LPRS has been widely applied in various areas. For example, the parking charge system associated with LPRS has replaced the original bookkeeping mode of time cards, which could greatly reduce labor cost and improve the efficiency. The LPRS is also an important part of electronic toll collection (ETC) which is widely used in many countries. The LPRS can help the ETC systems realize automatic toll collection without stopping the cars on the highway, which highly improves the throughput of the highway. The LPRS can also enable detecting the identity of illegal vehicles that run red lights or retrograde efficiently, which helps maintain urban public safety [Chen and Lu (2014);

[1] Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, 510006, China.

[2] Da Hengqin Science and Technology Development Company, Ltd., Zhuhai, 519000, China.

[3] Department of Computer Science, Rice University, Houston, TX 77025, USA.

[*] Corresponding Author: Le Wang. Email: wangle@gzhu.edu.cn.

Received: 31 May 2020; Accepted: 16 June 2020.

Wang, Yang, Feng et al. (2016); Wang, Yang, Chen et al. (2020)]. In constructing a smart and secure city, the LPRS technology has played an important and irreplaceable role with high accuracy and efficiency in identifying the license numbers.

Traditional license plate recognition (LPR) technologies utilize template matching, feature extraction and support vector machine (SVM) [Chen, Xu, Zuo et al. (2019); Cortes and Vapnik (1995)], but they are inefficient to recognize the license numbers and they are susceptible to environmental interference. Deep neural networks (DNNs) are widely adopted in LPRS to improve the accuracy [Gonalves, DaSilva, Menotti et al. (2016); Hendry and Chen (2019); Silva and Jung (2018); Zhang, Wang, Lu et al. (2019)]. For example, convolutional neural networks (CNNs), back propagation neural networks, and long short-term memory (LSTM) models are adopted to achieve remarkable performance in recognizing license numbers. However, DNNs lack theoretical explanation and they are vulnerable to adversarial examples that are crafted artificially. In [Szegedy, Zaremba, Sutskever et al. (2014)], the vulnerability of DNNs is firstly investigated which generates adversarial examples to fool the DNNs. These adversarial examples are crafted by adding small-magnitude perturbations to the original images; human cannot tell the difference but most DNNs would misclassify the adversarial examples. Many methods are then proposed to generate adversarial examples that have high attack success ratio [Goodfellow, Shlens and Szegedy (2015); Liu, Ye, Shang et al. (2020); McDaniel, Papernot and Celik (2016); Moosavi-Dezfooli, Fawzi and Frossard (2016); Gu, Hu, Zhang et al. (2020); Nguyen, Yosinski and Clune (2015); Simen and Wiebe (2019); Xie, Wang, Zhang et al. (2018); Xie, Wu, van der Maaten et al. (2019); Yuan, He, Zhu et al. (2019)].

However, most works generate adversarial examples against image classification systems on some well-known datasets, such as MNIST[1], CIFAR[2], and SVHN[3]; few of them extend the attack methods on the LPRS directly. In generating adversarial examples that could fool the LPRS, there are the following challenges. To begin with, the LPRS could output several numbers and characters of the license plate automatically (for example, there are seven digits on Chinese license plate), which is different from traditional image classification tasks that only classify an image to a specific label. Second, the system should first identify the location of the license plate in the image, then the system recognizes the license numbers. Adding perturbations to the locations that are outside the license plate would be meaningless. Third, the fonts of the license numbers are very standard; the generated adversarial examples by traditional attack methods could not fool the LPRS easily. For example, fast gradient sign method (FGSM) [Goodfellow, Shlens, and Szegedy (2015)], basic iterative method (BIM) [Kurakin, Goodfellow and Bengio (2017)] and projected gradient descent (PGD) [Madry, Makelov, Schmidt et al. (2017)] are three classic attack methods that could generate adversarial examples against DNNs, but applying them to the LPRS directly cannot work well.

[1] The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/.
[2] The CIFAR-10 and CIFAR-100 datasets. http://www.cs.toronto.edu/~kriz/cifar.html.
[3] The Street View House Numbers (SVHN) dataset. http://ufldl.stanford.edu/housenumbers/.

In this paper, we present adversarial attacks on the LPRS. First of all, we choose HyperLPR[1] as the LPRS and we propose two attack methods that add perturbations on the basis of random noise and Gaussian noise respectively. Then, we modify three classic attack methods (FGSM, BIM and PGD) to generate adversarial examples that could have good attack performance on the LPRS. Finally, we show that the generated images against the HyperLPR system could also fool a black-box system (Baidu LPR system[2]). The evaluation results show that the generated images could attack the HyperLPR system easily, and they can also fool the Baidu LPR system without knowing the architecture and the parameters beforehand. We summarize the contributions of this paper as follows:

1) We propose two attack methods against the HyperLPR system that generate adversarial examples by adding random noise and Gaussian noise respectively;

2) We modified three traditional attack methods such that these methods could generate adversarial examples to fool the HyperLPR system;

3) We verify the transferability of the generated adversarial examples as they can also fool the black-box neural networks.

The rest of the paper is organized as follows. The next section introduces some background of the license plate recognition system. In Section 3, we introduce different types of adversarial attacks against deep neural networks. We propose the attack methods on the license plate recognition system in Section 4. Evaluation results are shown in Section 5 and we summarize the paper in Section 6.

2 The process of license plate recognition system

In this section, we introduce the process of the license plate recognition system (LPRS). Generally, the LPRS consists of the following four modules: image collection, image preprocessing, license plate location, and license plate recognition [Du, Ibrahim, Shehata et al. (2013); Laroca, Severo, Zanlorensi et al. (2018); Redmon and Farhadi (2017); Xu, Yang, Meng et al. (2018)]. The whole process is illustrated in Fig. 1.

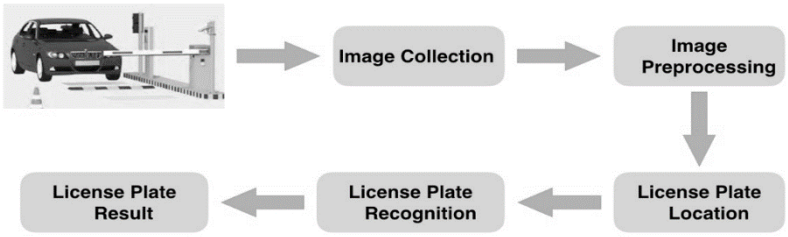

Figure 1: License plate recognition process

As shown in the figure, the image collection module collects the vehicle images by cameras that are placed at the parking lots, highway intersections, etc. The quality of the captured images is greatly affected by many environmental interferences, such as weather,

[1] HyperLPR LPR. https://github.com/zeusees/HyperLPR.
[2] Baidu LPR system. https://ai.baidu.com/tech/ocr_cars/plate.

light intensity, angel, distance, etc. Hence, the image preprocessing module removes the unnecessary background and noise information. The commonly adopted methods include graying, binarization, and edge detection. After that, the system identifies the location of the license plate in the third module. There are many traditional identification methods, such as texture analysis-based methods, edge detection-based methods, wavelet transform based methods and neural network-based methods [Chen and Lu (2014); Viola and Jones (2001)]. This module could find out the position of the license plate for further identification. The license plate recognition module enables recognizing the license numbers on the license plate. Traditional methods include template matching, feature extraction and support vector machine (SVM) [Cortes and Vapnik (1995)], while deep neural networks (DNNs) are widely adopted in recent years to achieve high recognition accuracy, such as in multiple target recognition [Feng, Arshad, Zhou et al. (2019)].

In this paper, we select the HyperLPR system and propose attack methods against the system. The HyperLPR system constructs an end-to-end recognition neural network which is composed of a convolution layer, a max-pooling layer, two filtering layers, four gated recurrent units of 256 hidden units, a dropout layer, and the output layer. The HyperLPR system can recognize each character on the license plate efficiently, and we choose the HyperLPR system as the model to attack.

3 Adversarial attacks against deep neural networks

In this section, we introduce the adversarial attacks against deep neural networks (DNNs). Although significant progress has been made by DNNs, it is still unsecure to adopt DNNs in many critical applications due to the adversarial attacks to the DNNs. The attackers could modify an input image slightly such that the image is misclassified by the DNNs while humans cannot tell the difference. Those modifications added to the input image are called *perturbations*, and the generated images are called *adversarial examples*.

3.1 Overview of adversarial example

The concept of adversarial example is initiated in [Szegedy, Zaremba, Sutskever et al. (2014)], which investigated the vulnerability of DNNs. After that, FGSM was proposed in [Goodfellow, Shlens and Szegedy (2015)], which generates adversarial examples in an efficient way. This method is a pioneering work that draws the attention from a large number of researchers.

The DNNs lack theoretical explanation, neither the adversarial examples could be explained in a theoretical method. There are two main reasons that may lead to the vulnerability of DNNs. The first reason comes from the architecture of the DNNs. Some works assume the architecture and the training steps are unstable to the crafted adversarial examples. The second reason might be the incomplete training data. Since the training data cannot cover all possibilities and all features, it is very difficult to train a neural network that satisfies the data distribution perfectly. Then, the decision boundary of the trained DNN might be inconsistent with the real data distributions. Fig. 2 shows a simple example where the middle-dashed curves imply the real distribution of three classes (A, B and C), while the classification boundaries of a trained model are depicted

as middle solid lines. Adversarial examples might exist in the areas between these curves and these lines.

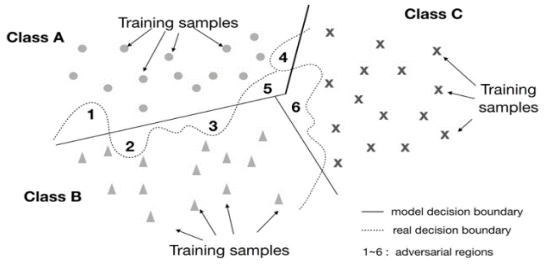

Figure 2: An example of adversarial examples space

3.2 Classification of adversarial attacks

The methods of generating the adversarial examples are referred to *adversarial attacks*. In this part, we introduce different types of adversarial attacks.

3.2.1 White-box and black-box attacks

Regarding whether the attackers could obtain the information of the DNNs, the adversarial attacks could be classified into white-box attacks and black-box attacks.

White-box attacks assume that attackers have full access to the information of the DNNs, including the architecture of the DNNs, the training data, the trained parameters, activation functions, etc. The mainstreams of white-box attacks are gradient-based attacks and optimization-based attacks. In contrast, *black-box attacks* assume the attackers cannot obtain the detailed information of the DNNs, and they can only utilize the output (such as the classification label and confidence) of the DNNs to generate adversarial examples. The mainstreams of black-box attacks are based on substitute neural networks and the transferability of adversarial examples that are generated by white-box attacks.

White-box attacks are fully studied in the extant works, but these methods are inapplicable in practice since it is difficult to obtain the system's information. Black-box attacks could cause security problems to real recognition systems, and they draw more attention recently.

3.2.2 Targeted and non-targeted attacks

Regarding the goal of the attackers, the adversarial attacks could be classified into targeted attacks and non-targeted attacks. *Non-targeted attacks* only try to fool the DNNs. For example, non-targeted attacks against the LPRS assumes the system could not recognize the license numbers correctly. In contrast, *targeted attacks* aim at generating the adversarial examples that should be recognized as a pre-selected label. Targeted attacks against the LPRS aim at making the system recognizing the adversarial example as a pre-defined license number. It is obvious that non-targeted attacks are much easier than targeted attacks.

3.2.3 Single-step and iterative attacks

Single-step attacks generate the adversarial examples against DNNs with just one step of optimization or gradient updating. This kind of attacks could generate adversarial examples efficiently, but the attack success ratio is not high. Many improved methods are proposed to generate the images in multiple iterations; these methods are referred to *iterative attacks*.

3.2.4 Image domain and physical domain attacks

Image domain attacks generate adversarial examples against image classification DNNs by modifying the pixels of the images. However, the crafted images might not exist in practice. Most of the extant adversarial attacks belong to the image domain attacks; they might modify the whole image but the modification might not exist. In Kurakin et al. [Kurakin, Goodfellow and Bengio (2017); Brown, Mané, Roy et al. (2018)], the generated images are printed on the paper and the recognition system also misclassifies the printed image. This work initiates the study of *physical domain attacks*. In Athalye et al. [Athalye, Engstrom, Ilyas et al. (2017)], the generated turtle is 3D printed and the DNN would misclassify it as a rifle. In Sharif et al. [Sharif, Bhagavatula, Bauer et al. (2016)], glasses are designed on purpose such that the person wearing the glasses would be misidentified. The physical domain attacks could incur critical security problems against a real recognition system.

3.3 Classic attack methods

We introduce some classic adversarial attack methods and their intuitive ideas.

3.3.1 Fast sign gradient method (FGSM)

FGSM is proposed in Goodfellow et al. [Goodfellow, Shlens and Szegedy (2015)], which can generate the adversarial examples efficiently. FGSM assumes the perturbations are added according to the gradient direction and the adversarial example X' is composed of the original image and the added perturbations. As shown in Fig. 3, $X' = X + \eta$ where X represents the original image and η implies the added perturbations (noise). Define $L(\theta, x, y^{true})$ as the loss function of training the neural network that the input image x is recognized as label y^{true}, and θ represents the model parameters. The perturbations are generated as:

$$\eta = \epsilon \cdot \text{sign}\left(\nabla_x L(\theta, x, y^{true})\right) \quad (1)$$

where ϵ is the added threshold, $\text{sign}(\cdot)$ indicates whether the value is positive or not, and $\nabla_x L(\cdot)$ represents the gradient of the loss function.

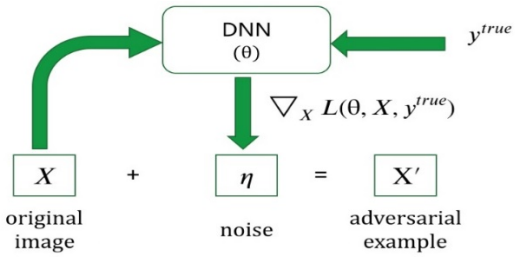

Figure 3: Generating adversarial examples using FGSM

3.3.2 Basic iteration method (BIM)

FGSM could generate the adversarial example efficiently since it only updates the image in one iteration. However, this method cannot achieve high attack success ratio. BIM can be considered as an iteration version of FGSM, and the calculation process is similar. As shown in the following equations, the adversarial example is generated in multiple iterations where α is a similar parameter that restricts the perturbations.

$$\begin{cases} X'_{t+1} = X'_t + \eta_t \\ \eta_t = \alpha \cdot \mathrm{sign}\left(\nabla_x L(\theta, X'_t, y^{true})\right) \end{cases} \tag{2}$$

3.3.3 Project gradient descent (PGD)

The PGD method is also an iterative attack method of FGSM. PGD adds random noise before generating the adversarial example iteratively. The method first initializes the search from a random point within the allowed restriction, then it adopts the iterative FGSM and generate the adversarial example. The process is formulated as the following equation:

$$\begin{cases} \eta_t = \alpha \cdot \mathrm{sign}\left(\nabla_x L(\theta, X'_t, y^{true})\right) \\ X'_{t+1} = \prod_S (X'_t + \eta_t) \end{cases} \tag{3}$$

where $\prod_S(\cdot)$ implies clipping the adversarial example within the restriction.

4 Adversarial attacks on LPRS

In this section, we introduce the adversarial attacks on the license plate recognition system. To begin with, we show the method of adding noise to the original images; then we show how the classic attack methods could be modified to attack the LPRS.

4.1 Random noise based adversarial attack

The random noise based adversarial attack is simple; we add random noise during the recognition of the HyperLPR system. After the system identifies the location of the license plate, we add random noise to each pixel of the image. Although the noise is generated randomly, the experiment results show that some generated adversarial example could attack the system successfully. This is because traditional LPR systems are mainly trained on the standard license plate; the trained neural networks cannot handle such noise well.

Fig. 4 shows an example that the generated adversarial example attacks the HyperLPR system successfully. Fig. 4(a) shows the original image after identifying the license plate,

while Fig. 4(b) shows the generated image by adding random noise. It is clear that we can still recognize the license number of both images as "陕 A 8AX58"[1]. However, when we input the adversarial example (Fig. 4(b)) to the HyperLPR system, it was misidentified as "陕 A8AAX58" and the confidence value of the result is high (0.9299).

(a) The original image (b) the adversarial example

Figure 4: An adversarial example generated by random noise

Actually, generating adversarial examples by adding random noise could not attack the recognition DNNs easily. In this paper, we show that the license plate recognition system is vulnerable to such a simple attack method. It also implies the incompleteness of the training data could cause adversarial examples. Hence, training the DNNs with more license plate data, this kind of attack might be defended.

4.2 Gaussian filter based adversarial attack

Adding random noise to the original image might not attack the HyperLPR system with high success ratio. We propose the Gaussian filter based adversarial attack.

Gaussian blur, also known as Gaussian smoothing, is commonly used to reduce the image noise and the detail information of the image [Zhang and Ma (2019)]. The image produced by the Gaussian blur technology has good visual effect such that the generated image looks quite similar as the original image through a translucent screen. From the mathematical perspective, the Gaussian blur process of an image can be considered as the convolution of the image with a normal distribution. Since the Fourier transform of Gaussian function is another Gaussian function, Gaussian blur is also regarded as a low-pass filter on the image.

Gaussian filter is a kind of liner filtering which has been widely adopted. The noise in many images obey the Gaussian distribution, and Gaussian filters are commonly utilized in image processing. The normal distribution is used to calculate the transformation of each pixel in the image to smooth the image. One-dimensional gaussian distribution is formulated as:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}, \tag{4}$$

and two-dimensional gaussian distribution is formulated as：

$$G(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2+y^2}{2\sigma^2}}. \tag{5}$$

In Gaussian filter, a user-specified template (also called convolution or mask) is used to scan each pixel of the image, and the pixel value is replaced by the weighted average gray value of the pixels in the template area.

[1] The license belongs to one author. This example is only used for research.

Different from random noise based attack method, the perturbations are added by the Gaussian filter. For example, we utilize a kernel of 15×15 to generate the adversarial examples. By choosing different σ values, the attack performance could be different.

Table 1: Recognition results with different σ values

σ	Result	Confidence
15	赣 A8AX58	0.8819
19	津 A8AY58	0.7946
21	青 A8AY58	0.7332

As shown in Tab. 1, we choose different σ values to generate the adversarial examples, and the results show that the HyperLPR system can be attacked easily.

4.3 Modified FGSM on LPRS

FGSM cannot be applied directly to generate adversarial examples against PLRS. Hence, we improve FGSM as follows:

Step 1: Identify the position of the license plate;

Step 2: Resize the license plate image to 160×40;

Step 3: Generate the adversarial example within T iteration;

Step 4: Initialize $\epsilon = 0.001$ and generating adversarial example X_t' by Eq. (1);

Step 5: If X_t' is identified correctly by the system, update ϵ by the loss function $L(\theta, x, y^{true})$ and generate X_{t+1}' in the next iteration; else return the generated image.

Since the HyperLPR system recognizes the license numbers with input size 160×40, we resize the license plate image to suit the system. Then we generate the adversarial example by modifying the restriction threshold ϵ, which is initialized as 0.001. FGSM belongs to single-step attack, but it is difficult to choose an appropriate value for the threshold ϵ. Hence, we modify the parameter in multiple iterations until the generated adversarial example fools the system or the number of iterations reaches T.

As shown in Fig. 5, the HyperLPR system could identify the input image as the correct license number "陕 A 8AX58" with high confidence value 0.9907, while the generated adversarial example (the right image) is identified incorrectly as "赣 A8AX58" with high confidence 0.8417. The perturbations are also depicted and the threshold is $\epsilon = 0.022$.

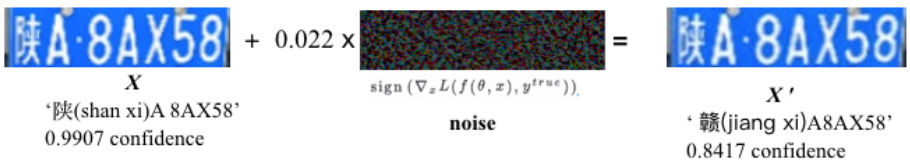

Figure 5: A generated adversarial example by the modified FGSM

Compared with the noise based attack methods, the generated adversarial images by the modified FGSM could fool the HyperLPR system more easily, while the generated image is more similar as the input image, compared with the generated image in Fig. 4(b).

4.4 Modified BIM on LPRS

FGSM cannot achieve targeted attack easily, and we propose the modified BIM that can achieve both non-targeted and targeted attacks.

4.4.1 Modified BIM for non-targeted attack

Similar as the modified FGSM, we introduced the non-targeted attack against the HyperLPR system as follows:

Step 1: Identify the position of the license plate;
Step 2: Resize the license plate image to 160×40;
Step 3: Define the loss function $\mathrm{L}(\theta, x, y^{true})$ as the cost that the image x is classified as the correct label y^{true} by the HyperLPR system;
Step 4: Initialize $\alpha = 0.01$, and generate the adversarial example within T iterations. The adversarial example is generated by Eq. (2).

The method defines the loss function $\mathrm{L}(\theta, x, y^{true})$ such that the image x is classified as the correct label y^{true}; the goal of the non-targeted attack is to compute the adversarial example X_t' that maximizes the loss function. This is because non-targeted attack aims at fooling the system by identifying the image as an incorrect label. The generated adversarial examples with different numbers of iterations could be misidentified as different license numbers. We show the results in Section 5.2.

4.4.2 Modified BIM for targeted attack

Targeted attack against the HyperLPR system is very different from non-targeted attack. As shown in previous figures (such as Figs. 4 and 5), the adversarial examples are misidentified as some unreasonable license numbers. For example, the generated image in Fig. 4(b) is recognized as "陕 A8AAX58", which contains 8 characters but a reasonable license number only has 7 characters. We show how to modify BIM for targeted attack against the HyperLPR system. Define the target license number is y^{adv} and the goal is to generate the adversarial example X' such that it is recognized as y^{adv} by the system. The process is similar as the modified BIM for non-targeted attack. The difference is that we define the loss function $\mathrm{L}\left(\theta, x, y^{adv}\right)$ as the cost that the HyperLPR system recognizes the image as y^{adv} in Step 3. In Step 4, we generate the adversarial example in T iterations and the image in each iteration is $X'_{t+1} = X'_t + \alpha \cdot \mathrm{sign}\left(\nabla_{\mathrm{x}} \mathrm{L}\left(\theta, X'_t, y^{adv}\right)\right)$ where $\alpha = 0.01$. The goal is to generate the adversarial example that minimizes the loss function.

BIM targeted attack has the similar process as the non-target attack, but the judgment of the targeted attack method is different. In the targeted attack, the iteration stops until the generated image is recognized as the target. Considering the example in Fig. 4, the target license number is chosen as "陕 A 8AM58" (the fifth character should be misidentified from X to M). After 45 iterations, the generated adversarial example is identified as "陕 A 8AM58" with high confidence (0.865) by the HyperLPR system. Specifically speaking, the system would recognize each character in the image separately and the fifth character is recognized as 'M'. The examples are shown in Tab. 4 (please see Section 5.3).

4.5 Modified PGD on LPRS

We modify the PGD method against the HyperLPR system. Similar as the modified FGSM and BIM, we first identify the location of the license plate and resize the image to 160*40. We define the loss function $L(\theta, x, y^{true})$ as the cost that the image x is classified as the correct label y^{true}. The method generates the adversarial example in multiple iterations and the whole image could be modified. Since the procedures are similar as the modified BIM, we do not describe the details. Fig. 6 shows the adversarial example generated by the modified PGD method. The HyperLPR system misidentifies the generated image incorrectly as "A 贛 A8AX58" with confidence value 0.9508.

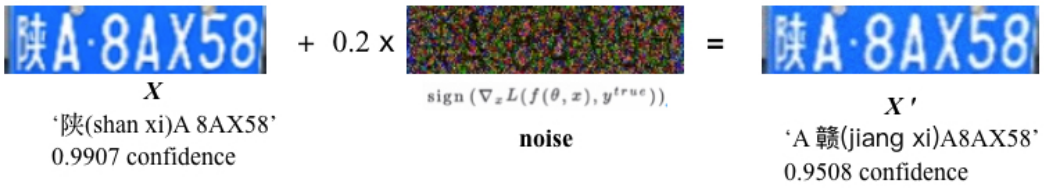

Figure 6: the adversarial example generated by the modified PGD method

5 Evaluation results

In this section, we present the evaluation results of the proposed methods on the license plate recognition systems. We implemented the adversarial attack methods against the HyperPLR system. In order to show the attack performance against some black-box LPRS, we choose Baidu LPRS to show the attack performance.

We implemented the HyperLPR system, which determines the boarders of the license plate by OpenCV and identifies the license number precisely. We collected 1150 images; some of them are taken from the physical world and the others are from website. We implemented our proposed methods in Python and run these methods with four GPU cards (GeForce RTX 2080 Ti). We first show the impact of parameters in several attack methods.

5.1 Modified FGSM

As shown in Section 4.3, the parameter ϵ is computed by the loss function in different iterations. Different ϵ values might lead to different recognition results and different confidence values. Considering the license example "陕 A 8AX58", we show that the HyperLPR system would misrecognize the generated adversarial example as "贛 A8AX58" when $\epsilon = 0.022$. We also show some recognition results for different ϵ values in Tab. 2.

Table 2: License plate recognition results for different ε values

Epsilon	Result	Confidence
0.022	贛 A8AX58	0.8417
0.058	A 贛 A8AX58	0.8377
0.069	A 新 A8AX58	0.8357
0.092	鄂 A 新 A8AX58	0.7691

5.2 Modified BIM for non-targeted attack

The modified BIM could attack the HyperLPR system easily. For the non-targeted attack, we show that the generated images in different iterations would be recognized as different license numbers incorrectly. As shown in Tab. 3, when the number of iterations reaches 2, 13, 15, 23,29, the generated images are misrecognized as "A 贛 A8AX58", "A 贛 B8AX56", "A 贛 C8AX56", "A 贛 C8AM56", and "A 贛 C8AAM56" respectively.

Table 3: The modified BIM for non-targeted attack

Iteration	Images	Result	Confidence
2		A 贛 A8AX58	0.9508
13		A 贛 B8AX56	0.8928
15		A 贛 C8AX56	0.8984
23		A 贛 C8AM56	0.9121
29		A 贛 C8AAM56	0.9660

5.3 Modified BIM for targeted attack

We show the process of the modified BIM for targeted attack. We set two different targets "陕 A 8AM58" and "陕 A 8AX56"; the first one assumes the fifth bit is misrecognized from "X" to "M" while the second one assumes the last bit is misrecognized from "8" to "6". In Tab. 4, we show the generated license images that lead to the target attacks.

Table 4: The modified BIM for targeted attack

Iteration	Images	Result	Confidence
45		陕 A 8AM58	0.9865
100		陕 A 8AX56	0.9983

5.4 Modified PGD

We evaluated the modified PGD attack method. We show some examples in Tab. 5 and conclude that the generated images could fool the HyperLPR system with high confidence, but human can still identify the correct license numbers easily. The other two examples are taken from the image set randomly and these images are only generated for research.

Table 5: The modified PGD attack

Images	Recognition Result	Confidence
	川陕 AG0F29	0.9265
	JJAY888	0.95506
	赣 A8AX58	0.9326

5.5 Efficiency comparison of different attack methods

We conduct the adversarial attack methods on the license plate dataset; each image can generate a corresponding adversarial example that fools the HyperLPR system by setting different parameters. We show the efficiency comparison of these attacks, including the random noise and Gaussian noise based attacks. The average time cost to generate an adversarial example that fools the system is depicted in Fig. 7. For the random noise and Gaussian noise based attacks, we only compute the average time of successful attacks, and the average time is short. The other three attack methods can generate the adversarial examples that fool the system and we compute the average time. From the figure, PGD method works more efficiently than the other two methods, while FGSM spends more time because the method only adds perturbations according to the sign of the gradient.

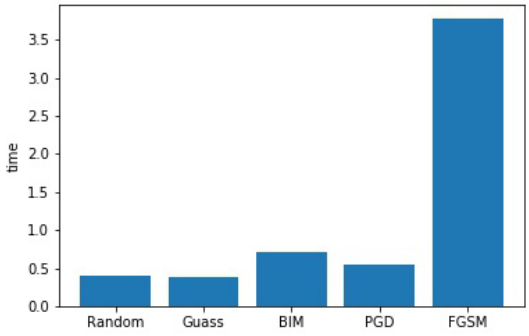

Figure 7: Efficiency comparison of the attack methods

5.5 Attack performance on the black-box LPRS

We choose the Baidu LPRS as the black-box recognition system. As we do not know the architecture and the trained parameters of the recognition system, the attack performance against the system could show the transferability of the generated adversarial examples. Taking the license number "陕 A 8AX58" as the example, we show the recognition results of Baidu LPR system as Tab. 6. From the table, the generated adversarial examples could also be recognized as incorrect results by the Baidu LPR system,

Table 6: Attack performance against HyperLPR and Baidu LPR

	Images	HyperLPR	Baidu LPR
Input Image		陕 A 8AX58	陕 A8AX58
Random Noise Based Attack		陕 A8AAX58	陕 A8XA58
Gaussian Filter Based Attack		贛 A8AX58	Unable Identify
Modified FGSM		贛 A8AX58	陕 A8XX58
Modified BIM (non-targeted)		A 贛 A8AX58	陕 A8XX58
Modified BIM (targeted)		陕 A 8AX56	陕 A8XA58
Modified PGD		贛 A8AX58	陕 A8XX58

6 Conclusion and future works

Deep neural networks have been widely adopted in many intelligent systems, but the adversarial examples that misleading the deep neural networks could incur security problems on the systems. In this paper, we show that the license plate recognition system could be attacked by the generated adversarial examples. Specifically, we show different attack methods that could make the systems make incorrect recognitions. In addition, the generated adversarial examples could also attack black-box systems without obtaining the network information beforehand, which implies a serious security problem on the real LPR systems. We hope this work could draw the attention of the intelligent LPR systems and it is necessary to design robust algorithms against adversarial attacks.

In the future, we will study the defense methods that could improve the security of the intelligent systems. One interesting direction is to modify the architecture of the deep neural networks that could defend such adversarial examples, and another further direction is to collect and generate more useful image in the training dataset.

Acknowledgement: We would like to thank anonymous reviewers and the associate editor for the constructive comments in improving this work.

Funding Statement: This work is supported by the National Natural Science Foundation of China under Grant Nos. U1636215, 61902082, the Guangdong Key R&D Program of China 2019B010136003, and National Key R&D Program of China 2019YFB1706003.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

Athalye, A.; Engstrom, L.; Ilyas, A.; Kwok, K. (2018): Synthesizing robust adversarial examples. *International Conference on Machine Learning*, pp. 284-293.

Brown, T. B.; Mané, D.; Roy, A.; Abadi, M.; Gilmer, J. (2017): Adversarial patch. http://arxiv.org/abs/1712.09665.

Chen Y.; Lu C. (2014): The business value of the license plates recognition system. *Electronic Test*, vol. 22, pp. 110-112.

Chen, Y. T.; Xu, W. H.; Zuo, J. W.; Yang, K. (2019): The fire recognition algorithm using dynamic feature fusion and IV-SVM classifier. *Cluster Computing*, vol. 22, no. 3, pp. 7665-7675.

Cortes, C.; Vapnik, V. (1995): Support-vector networks. *Machine Learning*, vol. 20, no. 3, pp. 273-297.

Du, S.; Ibrahim, M.; Shehata, M. S.; Badawy, W. M. (2013): Automatic license plate recognition (ALPR): a state-of-the-art review. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 2, pp. 311-325.

Feng, C.; Arshad, S.; Zhou, S.; Cao, D.; Liu, Y. (2019): Wi-Multi: A three-phase system for multiple human activity recognition with commercial WiFi devices. *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 7293-7304.

Gonalves, G. R.; DaSilva, S. P. G., Menotti, D., Schwartz, W. R. (2016): Benchmark for license plate character segmentation. *Journal of Electronic Imaging*, vol. 25, no. 5, pp. 053034.

Goodfellow, I. J.; Shlens, J.; Szegedy, C. (2015): Explaining and harnessing adversarial examples. *International Conference on Learning Representations*.

Gu, Z.; Hu, W.; Zhang, C.; Lu, H.; Yin, L. et al. (2020): Gradient shielding: towards understanding vulnerability of deep neural networks. *IEEE Transactions on Network Science and Engineering*, 10.1109/TNSE.2020.2996738.

Hendry; Chen, R. C. (2019): Automatic license plate recognition via sliding-window darknet-YOLO deep learning. *Image and Vision Computing*, vol. 87, pp. 47-56.

Kurakin, A.; Goodfellow, I. J.; Bengio, S. (2017): Adversarial examples in the physical world. In *International Conference on Learning Representations*.

Laroca, R.; Severo, E.; Zanlorensi, L. A.; Oliveira, L. S.; Gonçalves, G. R. et al. (2018): A robust real-time automatic license plate recognition based on the YOLO detector. In *International Joint Conference on Neural Network*.

Liu, C.; Ye, D.; Shang, Y.; Jiang, S.; Li, S. et al. (2020). Defend against adversarial samples by using perceptual hash. *Computers, Materials & Continua*, vol. 62, no. 3, pp. 1365-1386.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. (2017): Towards deep learning models resistant to adversarial attacks. https://arxiv.org/abs/*1706.06083*.

McDaniel, P. D.; Papernot, N.; Celik, Z. B. (2016): Machine learning in adversarial settings. *IEEE Security & Privacy*, vol. 14, pp. 68-72.

Moosavi-Dezfooli, S. M.; Fawzi, A.; Frossard, P. (2016): DeepFool: a simple and

accurate method to fool deep neural networks. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2574-2582

Nguyen, A. M.; Yosinski, J.; Clune, J. (2015): Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 427-436.

Redmon, J.; Farhadi, A. (2017): YOLO9000: better, faster, stronger. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7263-7271.

Sharif, M.; Bhagavatula, S.; Bauer, L.; Reiter, M. K. (2016): Accessorize to a crime: real and stealthy attacks on state-of-the-art face recognition. *ACM Conference on Computer and Communications Security*, pp. 1528-1540.

Silva, S. M.; Jung, C. R. (2018): License plate detection and recognition in unconstrained scenarios. *European Conference on Computer Vision*, pp. 580-596.

Simen T.; Wiebe V. R. (2019): Fooling automated surveillance cameras: adversarial patches to attack person detection. https://arxiv.org/abs/1904.08653v1.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D. et al. (2014): Intriguing properties of neural networks. *International Conference on Learning Representations*, arXiv preprint arXiv:1312.6199.

Viola, P.; Jones, M. (2001): Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. I-I.

Wang, X.; Yang, L. T.; Chen, X.; Deen, M. J.; Jin, J. (2018): Improved multi-order distributed HOSVD with its incremental computing for smart city services. *IEEE Transactions on Sustainable Computing*, doi: 10.1109/TSUSC.2018.2881439.

Wang, X.; Yang, L. T.; Feng, J.; Chen, X.; Deen, M. J. (2016). A Tensor-based big service framework for enhanced living environments. *IEEE Cloud Computing Magazine*, vol. 3, no. 6, pp. 36-43.

Xie, C.; Wang, J.; Zhang, Z.; Ren, Z.; Yuille, A. L. (2018): Mitigating adversarial effects through randomization. *International Conference on Learning Representations*, arXiv preprint arXiv:1711.01991.

Xie, C.; Wu, Y.; Der Maaten, L. V.; Yuille, A. L.; He, K. (2019): Feature denoising for improving adversarial robustness. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 501-509.

Xu, Z.; Yang, W.; Meng, A.; Lu, N.; Huang, H. et al. (2018): Towards end-to-end license plate detection and recognition: a large dataset and baseline. *European Conference on Computer Vision*, pp. 255-271.

Yuan, X.; He, P., Zhu, Q.; Li, X. (2019): Adversarial examples: attacks and defenses for deep learning. *IEEE Transactions on Neural Networks*, vol. 30, no. 9, pp. 2805-2824.

Zhang, J. M.; Wang, W.; Lu, C. Q.; Wang, J.; Sangaiah, A. K. (2019): Lightweight deep network for traffic sign classification. *Annals of Telecommunications*, pp. 1-11.

Zhang, Y.; Ma, F. (2019): Application and optimization of image fuzzy control algorithm based on gaussian blur in TensorFlow training. *International Conference on Mechatronics*.