Knowledge-Based Systems 226 (2021) 107102

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A low-query black-box adversarial attack based on transferability

Kangyi Ding [a], Xiaolei Liu [b], Weina Niu [a], Teng Hu [a,b], Yanping Wang [a], Xiaosong Zhang [a,c,*]

[a] Institute for Cyber Security, School of Computer Science and Engineering, University of Electronic Science and Technology of China, China
[b] Institute of Computer Application, China Academy of Engineering Physics, China
[c] Cyberspace Security Research Center, Peng Cheng Laboratory, China

ARTICLE INFO

Article history:
Received 17 December 2020
Received in revised form 23 April 2021
Accepted 28 April 2021
Available online 14 May 2021

Keywords:
Adversarial sample
Black-box attack
Transferability
Neural network

ABSTRACT

Artificial intelligence systems suffer from black-box adversarial attacks recently. To prevent this kind of attack, a large amount of researches that reveal the nature of this attack has emerged. However, the query count, success rate, and distortion in the existing works cannot fully satisfy the practical purposes. In this paper, we propose a low-query black-box adversarial attack based on transferability by combining the optimization-based method and the transfer-based method. Our approach aims to improve the black-box attack with a lower number of queries, higher success rate, and lower distortion. In addition, we make full use of surrogate models and optimize the objective function to further improve the performance of our algorithm. We verified our method on MNIST (Lecun and Bottou, 1998) [1], CIFAR-10 (Krizhevsky et al., 2009) [2], and ImageNet (Deng et al. 2009) [3], respectively. Experimental results demonstrate that our method can implement a black-box attack with more than 98.5% success rate and achieve specific distortion with less than 5% queries comparing with other state-of-the-art methods.

1. Introduction

In recent years, with the increased computing power and booming deep learning theory, AI (artificial intelligence) has been applied in a wide range of areas, such as image recognition [1,4–6], autonomous vehicle [7,8], natural language processing [9–11], and even human–machine confrontation [12,13]. However, the design of AI only considered the accuracy of specific tasks, while ignoring its safety and reliability to some extent. So it is challenging to provide robustness to the complex and variable AI-based applications, which undoubtedly poses a security challenge to these applications.

Unfortunately, it has been shown that the decisions of AI algorithms are not robust to some carefully crafted inputs through extensive [15–18]. An attacker can trick AI algorithms by applying a slight perturbation to the input data. For example, we change the decision of the victim model by adopting LBAT. As shown in Fig. 1, the noise we add is imperceptible. Recently, researchers have shown that an attacker can add computationally generated notes to traffic signal signs in the real world to deceive the computer vision system of autonomous vehicles [19] and an attacker can also deceive a facial recognition security system by wearing a 3D-printed eyeglass frame [20]. Besides, Liu [21] shows attackers can generate adversarial audios by adding noise to original audios which can control the existing AI speech recognition systems. It can be seen that while AI provides convenience to our lives, it also provides an opportunity for attackers with ulterior motives to take advantage of it.

Adversarial attacks work by applying a slight perturbation to the original data, which is invisible to humans but can change the classification results of the AI models dramatically. We call the modified samples adversarial samples, and the presence of an adversarial sample resembles the models' illusion.

Model robustness enhancement [22,23], input detection [24], adversarial training [25,26], denoising [27,28] and reconstruction [29] have been adopted to protect AI models from the treats of adversarial attacks. However, due to the diverse ways of generating adversarial samples, the defense method that can deal with different types of adversarial attacks has not been proposed. In addition, most of the existing defense methods can only improve a few adversarial attacks to a certain extent. For some time to come, it is unrealistic to completely resolve the threat of adversarial attacks. Therefore, adversarial attacks will be a long-term threat to AI algorithms, related applications, and systems. To enhance the robustness of the AI algorithms, we need deepen the understanding of the AI algorithms by studying adversarial attacks.

According to the known situation of the victim model, the generation methods of the adversarial samples are divided into white-box attacks and black-box attacks. The white-box attacks

[*] Corresponding author at: Institute for Cyber Security, School of Computer Science and Engineering, University of Electronic Science and Technology of China, China.
E-mail address: johnsonzxs@uestc.edu.cn (X. Zhang).

https://doi.org/10.1016/j.knosys.2021.107102
0950-7051/© 2021 Elsevier B.V. All rights reserved.

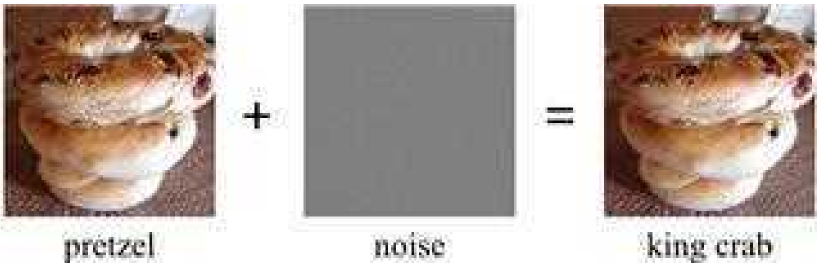

Fig. 1. LBAT was applied to Inception-V3 [14]. By adding an imperceptible noise, we can control the decision of the victim model.

mean that the attacker has a complete grasp of the model's input, output, structure, and parameters, so the attacker can easily get and make full use of the gradient of the model to implement an attack. While the black-box attacks mean the attacker can only access the model's input and output and knows nothing about the structure or parameters of the model.

With the joint efforts of researchers, a considerable amount of white-box methods [15,25,30,31] have been proposed and the white-box generation methods for generating adversarial samples can achieve excellent results. Due to the attackers can grasp all the information and gradient of the victim model, methods like [25,30,31] can implement adversarial attack with almost 100% success rate and extremely low distortions. However, in real-world attack and defense, it is difficult for an attacker to get information about the victim model. Black-box adversarial samples generation methods have become a hot topic of AI security research because it is closer to the actual application environment and more challenging.

The methods of black-box attacks are mainly divided into transfer-based methods, optimization-based methods, and decision-based methods. The adversarial samples generated by the transfer-based method have a low attack success rate and require large distortion. The optimization-based method requires a lot of queries which means a high probability to be detected and high implementation costs. The decision-based method can generate adversarial samples with a low distortion, but it still requires a large number of queries.

In this paper, we propose an approach that implements a black-box attack on deep neural networks. Our approach is named LBAT, denoting A (L)ow-query (B)lack-box Adversarial (A)ttack Based on (T)ransferability.

Our contributions are as follows:

(1) We utilize the transferability of the adversarial samples to implement the black-box attacks. By attacking models with the same task (surrogate models), we obtain the adversarial vector which is applied for the gradient estimation. Our method has been tested on the MNIST [1], CIFAR-10 [2], and ImageNet [3]. Experiments show that our method can implement the targeted attacks on the above datasets with at least a 98.5% success rate. Compared with the existing adversarial sample generation method, the number of queries to achieve specific distortion has been greatly reduced.

(2) When the surrogate models are used to generate the adversarial vector, LBAT randomly selects the surrogate model and the parameters of the surrogate model in each step, which can ensure that the generated adversarial vector has large differences in each time. At the same time, we do not need to perform a complete white-box attack on the surrogate model, but the intermediate data generated in the first few steps, which increases the speed of generation.

(3) Due to the values of the loss function is totally different between the surrogate models and the victim model during the black-box attack optimization. We optimize the loss function of the white-box attack by adding the dynamic coefficient to guarantee the loss function consistency between the white-box attack and the black-box attack. The experiments show such measures reduce the number of queries and improve the attack success rate.

The remainder of this paper is organized as follows: Section 2 reviews the related work in the area of black-box attack. Section 3 describes how to implement our method. Section 4 shows our experimental design and results, and Section 5 concludes.

2. Related work

According to the different implementation methods, the existing black-box adversarial attack methods mainly have the problems of low success rate, a high number of required queries, and a large amount of distortion required to generate adversarial samples.

Adversarial attacks are divided into targeted and non-targeted attacks according to the goal of the task. The non-targeted attacks are used to change the decision results of the model, while the targeted attacks are used to generate the sample that leads to the specified classification or decision. Obviously, the targeted attacks are more difficult to be executed as well as more threatening. In the research field of black-box adversarial attacks, the non-target attacks can be easily achieved, but existing methods still have shortcomings to implement targeted attacks. Therefore, current research is mainly focused on targeted attacks.

Here is a brief introduction to the existing black-box attack methods.

Transfer-based method. The transferability of adversarial attacks is different from transfer learning [32,33]. The adversarial sample is transferable, that is, an adversarial sample generated for a specific model has a higher probability of being adversarial comparing to other models with the same task. This is due to the similarity of the models' boundaries towards the same task. By using the transferability of adversarial samples, researchers have proposed a variety of black-box adversarial sample generation methods. Liu [34] optimizes the loss function and generates effective adversarial samples against multiple known models, which greatly improves the transferable success rate of non-targeted attacks and targeted attacks. Eq. (1) is the loss function of the method.

$$\mathrm{argmin}_{x*} - \log\left(\left(\sum_{i=1}^{k} \alpha_i J_i\left(x^*\right)\right) \cdot \mathbf{1}_{y^*}\right) + \lambda d\left(x, x^*\right) \qquad (1)$$

Where x denotes the original data, x^* denotes the modified data, J_i denotes the softmax outputs of the surrogate model, y^* is the target label specified by the adversarial sample, α_i are the ensemble weight.

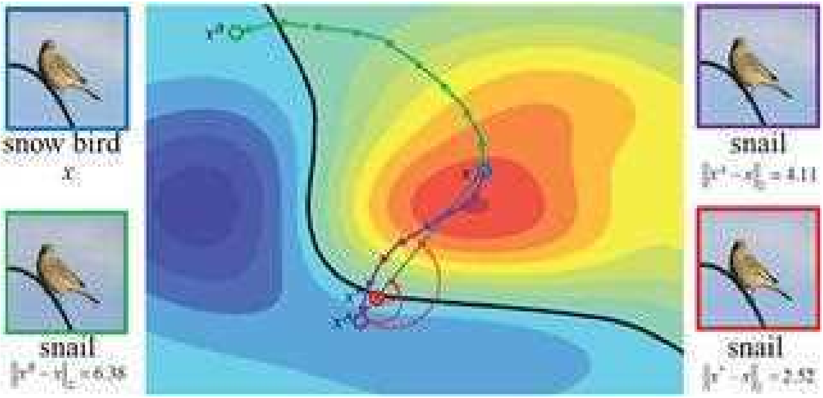

Fig. 2. Curls & Whey attack trajectory [35]. Blue ring denotes the original image, green and purple arc denote two search directions, red arc denotes the adversarial sample optimization.

Shi [35] proposed a method to improve the transferable success rate of adversarial samples between different models. The method achieves the target by attacking a surrogate model which is known to the attackers. And the attack trajectory is shown in Fig. 2. The method uses two directions to optimize the object at the same time. One of the directions is to generate an adversarial sample along the direction in which the loss function gradient descents (green arc in Fig. 2). The other direction is to first obtain the maximum value of the loss function along the gradient ascent (purple arc in Fig. 2) and then regenerate the adversarial sample along the gradient descent. Until the victim model is misled, the white-box attack stops. Then, it will choose the better adversarial sample (with lower l_2 distance, purple ring in Fig. 2). Finally, binary search and Whey Optimization (red arc in Fig. 2) are adopted to further optimize the adversarial sample. This method can increase the transferability of the adversarial sample, and reduce the distortions compared with a single search direction, but the distortion is still large and the success rate is unstable.

The transfer-based method usually has a low success rate and usually causes a lot of distortions, but it does not need to query so much on the victim model to implement an attack.

Optimization-based method. This kind of attack method [36–38] generates adversarial samples by randomly changing the input to estimate the gradient of the loss function. Chen [36] proposed the method of zeroth order stochastic coordinate descent to estimate the gradient. This method uses pixels as the unit of estimation, which requires lots of queries to the model, and when facing high-dimensional datasets, its attack accuracy will decrease seriously. Tu [37] adjusts the intensity of each pixel of the entire image randomly within a certain range, and an encoder is employed to map the entire image to a low-dimensional space, reducing the space required for estimation. Tu [37] effectively reduces the number of queries, and the accuracy of high-dimensional datasets is also relatively effective. Although the application of the encoder reduces the space for gradient estimation, it also increases the error of the calculation process, resulting in a large distortion. Liu [18] utilizes a swarm evolutionary algorithm to generate adversarial without gradient estimation, but still needs a large number of queries to implement.

Decision-based method. The Decision-based methods are able to generate adversarial samples only using the hard labels [39,40]. This kind of method uses the original image and the target label of the data as inputs and searches along the target label boundary until the perturbation satisfying the requirement. If the algorithm can be allowed to conduct with a large number of queries, the distortion can be reduced to an astonishing degree, even smaller than the distortions required by the white-box algorithm. However, the algorithm requires plenty of queries to reduce the distortion to the degree that humans cannot perceive.

Hybrid attack. The hybrid attack [41–43] usually combines the transfer-based attack and optimization-based attack to increase the success rate and reduce the required queries. Suya [43] proposed a method that attackers are able to use surrogate models to find a candidate adversarial sample. After that, the attacker implements an optimization-based method based on the candidate adversarial sample. The candidate adversarial sample can be used to implement an attack from a position that is better than the original data, so Suya [43] greatly reduces the number of queries. However, due to the use of a transfer-based attack, Suya [43] will cause larger distortions.

In summary, the transfer-based method is limited by its success rate and large distortions but requires a few queries. Optimization-based methods and decision-based methods require a large number of queries, but their success rate is high enough. To maximize their strengths and avoid their weaknesses, hybrid attacks have been proposed. However, the existing hybrid attacks still require larger distortions and a lot of queries. The method we propose is a hybrid attack method that can balance distortion, number of queries, and success rate.

3. Methodology

Our attack belongs to hybrid attacks. The attack environment is the same as [42,43,43]. Our method generates adversarial samples without grasping the parameters, structure, activation functions, and other information of the victim model. We aim at using fewer queries and distortions to implement attacks. Our method needs to grasp the input data and output probability values of the victim model F. At the same time, we also need to acquire several surrogate models (F_1, F_2, $\ldots F_n$) which have the same task as the victim model F. For the surrogate models, we are able to acquire information such as parameters, structure, and activation function of these models.

Compared with the existing algorithms, LBAT can generate small-distortion adversarial samples with fewer queries, while ensuring a higher success rate. On the one hand, we improve the gradient estimation process of the previous method by adopting transferability to make the queries more efficient. On the other hand, due to the efficient queries, we can give up the strategy of using the autoencoder, so it is more likely to generate adversarial samples with small distortions. Furthermore, based on the above, we randomly select the surrogate model to improve the efficiency of adversarial vector calculation and adopt dynamic coefficients to enhance the consistency between the loss functions of the white-box attack and the black-box attack.

In this section, we first propose the framework of LBAT. Then we introduce our gradient estimation method. Furthermore, we propose our adversarial vector calculation method.

3.1. LBAT framework

As Fig. 3 shows, LBAT is divided into 2 parts, black-box gradient estimation, and white-box adversarial vector calculation. We adopt the zeroth-order optimization to estimate the gradient, different from the previous methods [36,37,43], we use an adversarial vector generated by the white-box attack, rather than the random vector, which contributes to the efficiency of our gradient estimation query.

Our algorithm is also optimized through iteration. LBAT calculates the adversarial vector at each step. We add the adversarial vector to the current sample as a query vector, estimating the gradient of the adversarial vector direction, and then update the current sample.

3.2. Black-box gradient estimation

Similar to the white-box attack [31], the goal of our method is to minimize the perturbation while changing the model decision to a specific category. We adopt the classic definition method of adversarial sample generation as Eq. (2).

$$\begin{aligned} &\min_{\mathbf{x}'} \left\|\mathbf{x}' - \mathbf{x}\right\| \\ &\text{s.t.}\, f\left(\mathbf{x}'\right) = l' \\ &f(\mathbf{x}) = l \\ &l \neq l' \\ &\mathbf{x}' \in [0, 1] \end{aligned} \tag{2}$$

Where $\mathbf{x}$ denotes the original sample, $\mathbf{x}'$ denotes the generated adversarial sample, $f(\cdot)$ denotes the output of the model, l and l' are the output categories of the model F for the original sample $\mathbf{x}$ and the adversarial sample $\mathbf{x}'$, and $\|\cdot\|$ denotes distance between the original sample and the generated sample which is usually measured by L_0, L_2, or L_∞ norms.

Further, we translate the above problem into the following formulation.

$$\begin{array}{ll} \text{minimize} & \mathcal{D}(\mathbf{x}, \mathbf{x} + \delta) + \lambda \cdot g(\mathbf{x} + \delta) \\ \text{such that} & \mathbf{x} + \delta \in [0, 1]^n \end{array} \tag{3}$$

Where $g(\cdot)$ is the objective function we defined. The definition of $g(\mathbf{x})$ is as Eq. (4)

$$g(\mathbf{x}) = \max\left\{\max_{i \neq l'} \log[f(\mathbf{x})]_i - \log[f(\mathbf{x})]_{l'}, -\kappa\right\} \tag{4}$$

Only when the model's output satisfies $\max_{i\neq l'} \log[f(\mathbf{x})]_i - \log[f(\mathbf{x})]_{l'} < -\kappa$, $g(\mathbf{x})$ will output $-\kappa$, which means that the attack is successful. λ in (3) is a constant used to control the perturbation added during optimization. The larger λ means that greater distortion can be tolerated in the optimization process. δ is the distortion adding to the original data, and κ is the confidence of class l'.

In Eq. (4), the reasons why we use $\log[f(\mathbf{x})]$ instead of $f(\mathbf{x})$ are mainly to reduce the difficulty of optimization and promote the optimization focus on classifying as a specific target, rather than minimizing the maximum probability of non-specific target, especially at the beginning of optimization.

Our method queries the point near the current data to estimate the current gradient. Unlike the previous methods [36,37,43], the coordinates of our query points are provided by the adversarial vectors by attacking the surrogate model. The gradient estimation formula is as follow:

$$\hat{\mathbf{g}} = \frac{1}{N}\sum_{i}^{N} \frac{L\left(\mathbf{x} + \delta\mathbf{u}_i\right) - L(\mathbf{x})}{\delta}\mathbf{u}_i \tag{5}$$

Where L denotes the loss function which is shown in Eq. (3), $\mathbf{u}_i$ denotes the direction of the adversarial vectors, and $\|\mathbf{u}_i\|_2 = 1$. N denotes the number of estimates around the current sample $\mathbf{x}$. In general, the larger N, the more accurate the estimation is. Since LBAT uses adversarial vectors to estimate the gradient, the $\mathbf{u}_i$ is strongly referential. And we do not need a large number of queries to estimate the gradient of one point to increase accuracy. So in our algorithm, we set $N = 1$.

As for optimization, we use the ADAM optimizer [44]. ADAM optimizer is one of the most popular optimizers used in deep learning optimization. Unlike other deep learning applications, our black-box attack cannot obtain accurate gradient information of the current position. It only estimates the gradient of the adversarial vector direction, which probably be inaccurate. So we adjust the learning rate according to the gradient estimation of $\hat{\mathbf{g}}$.

$$\text{LR} = \begin{cases} \eta, & \text{if } \hat{\mathbf{g}} >= 0 \\ \eta/3, & \text{if } \hat{\mathbf{g}} < 0 \end{cases} \tag{6}$$

Where η denotes the initial learning rate value we set. When $\hat{\mathbf{g}} >= 0$, we consider that the direction of the adversarial vector is highly likely to be the correct optimization direction. While $\hat{\mathbf{g}} < 0$, we reduce the learning rate and consider that direction is untrusted, due to we do not query the direction of $-\mathbf{u}$.

3.3. Adversarial vector calculation

The previous methods [36,37,43] add a random vector to estimate the gradient of the current data in the direction of the random vector, and the efficiency of queries along the gradients in random directions is extremely low. In this paper, we perform a white-box attack on the surrogate models and add the obtained adversarial vector to the current data to estimate the gradient.

In the specific implementation of our method, the white-box attack against the victim model only uses a small number of iterations to obtain an adversarial vector, instead of generating an adversarial sample that is effective for the selected surrogate model. This generation method not only reduces the time for generating adversarial samples but also increases the diversity of adversarial vectors. In our experiment, we set the number of iterations of the white-box attack to 5.

If only one surrogate model is used to generate the adversarial vectors, the direction of each query will be similar. And the single direction of the query is harmful to optimization. The strategy we adopt is to randomly select one surrogate model in each step for implementing the white-box attack. In order to further improve the difference of the adversarial vector of each query, we set the parameters in the white-box attack to be within a certain range randomly.

In this paper, we use the current sample as the input of the white-box attack. Carlini [31] with L_2 norm is adopted to implement the white-box attack. The loss function of the Carlini [31] method is the same as Eq. (3) and the objective function is as follow:

$$g(\mathbf{x}) = \max\left\{\max_{i \neq l'}[f(\mathbf{x})]_i - [f(\mathbf{x})]_{l'}, -\kappa\right\} \tag{7}$$

Where κ denotes confidence of our adversarial sample and we reset κ to a random value each iteration. However, due to the different boundary between the victim model and the surrogate models, the optimization focus of the objective function of the white-box attack and the black-box attack is quite different. For example, since the goal of our algorithm is to achieve the black-box attack on the victim model, it often appears that the value of the black-box attack objective function is much smaller than the value of the white-box attack objective function. Especially when the loss function of black-box attack is close to 0 (the attack is

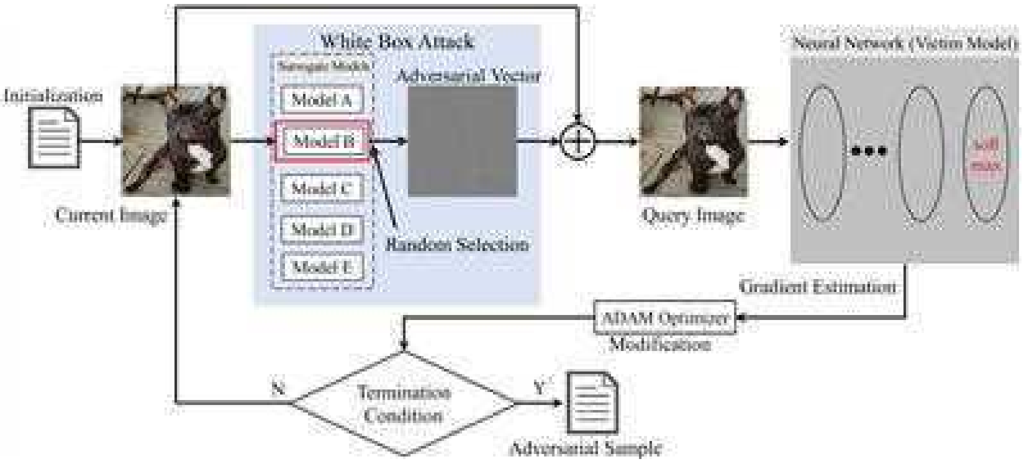

Fig. 3. Framework of LBAT. In each step, we conduct the white-box attack on a random selection model to calculate the adversarial vector. Then, the adversarial vector is added to the current image to get the query image. Next, we evaluate the gradient along the direction of the adversarial vector and use ADAM optimizer to update current data. Until the victim model classifies the current data into the specified category, the iteration exits.

almost successful), the white-box attack loss function non-target part ($\max_{i\neq l'}[f(\mathbf{x})]_i$) may not contribute to the optimization. The loss function uses Eq. (7) may cause optimization difficulties.

To solve the above problem, we control the optimization focus by adding a dynamic coefficient dc to the white-box attack objective function.

$$g(\mathbf{x}) = \max\left\{\max_{i\neq l'}[f(\mathbf{x})]_i - dc\cdot[f(\mathbf{x})]_{l'}, -\kappa\right\} \tag{8}$$

In order to ensure that Eq. (8) can be optimized at any condition, we set $\kappa = 2000$. Here we propose 2 calculation methods of the dynamic coefficient.

$$dc = \min(\frac{\text{obj}_{\text{max}}}{\text{obj}_{\text{current}}}, 3) \tag{9}$$

$$dc = \min(\sqrt{\frac{\text{obj}_{\text{max}}}{\text{obj}_{\text{current}}}}, 3) \tag{10}$$

Where obj is the objective function value of black-box attack. obj_{max} denotes the maximum value that appears in the iteration. $\text{obj}_{\text{current}}$ denotes the objective function value of the current data. We show the experiment result of our algorithm with different dynamic coefficients in Section 4.

After implementing the white-box attack, we need to transfer it to the format which can be the input of the query. So, we need to transfer the white-box adversarial sample to the adversarial vector. The transfer formulas are as follows:

$$\begin{aligned}\eta &= \text{adv}_{\text{white}} - \mathbf{x}_{\text{current}}\\ \eta &= \tfrac{\eta}{\|\eta\|}\end{aligned} \tag{11}$$

The white-box attack of LBAT is shown in algorithm 1.

Where **adv** denotes the intermediate data generated in the iterative process of the white-box attack. And the intermediate data is not necessarily to be an adversarial sample but is partly adversarial (the objective function value of the intermediate data is more likely less than the data that is input to the white-box attack). C&W($F(\cdot)_j$, $\mathbf{adv}_{\text{white}}$) denotes the white-box attack method we adopt.

Our method uses the adversarial vector to evaluate gradient rather than the random vector. This kind of strategy can improve the efficiency of queries, and reduce the query count to generate the adversarial samples. However, calculating the adversarial vector requires more queries to the surrogate models, and gradient calculation of the surrogate models.

Algorithm 1 White-Box Attack of LBAT

```
Input: current data = $\mathbf{x}_{\text{current}}$; Target label, $\mathbf{y}$; Surrogate models,
    $F(\cdot)_1, F(\cdot)_2 \cdots F(\cdot)_n$; Max iteration, $m(m <= 5)$;
Output: adversarial vector $\mathbf{u}$;
randomly select a model from $(F(\cdot)_1, F(\cdot)_2 \cdots F(\cdot)_n)$, suppose the model $j$ is selected;
randomly set the white-box attack loss function constant $c\lambda$ in Eq. (3);
$\mathbf{adv}_{\text{white}} \leftarrow \mathbf{x}_{\text{current}}$;
for $i = 0$ to $m$ do
    $\mathbf{adv}_{\text{white}}$ $\leftarrow$C&W($F(\cdot)_j$,$\mathbf{adv}_{\text{white}}$) attack [31] with objective function equation (10);
    if $\mathbf{adv}_{\text{white}}$ is adversarial sample of $F_j$ then
        break;
    end if
end for
$\eta \leftarrow \mathbf{adv}_{\text{white}} - \mathbf{x}_{\text{current}}$;
$\mathbf{u} \leftarrow \frac{\eta}{\|\eta\|}$;
return  $\mathbf{u}$
```

If we set the computational complexity of the black-box attack with the random vector as $O(m)$, the white-box attack gradient calculation and query cost of each time as $O(n)$. The computational complexity of LBAT of each step $O(LBAT)$ can be evaluated as follows:

$$O(LBAT) = 5\cdot O(n) + O(m) \tag{12}$$

5 means maximum iteration used in LBAT. According to Eq. (12), our method requires more calculation in each step. For the black-box adversarial attack, the main cost is the query to victim model, while the costs of the query to the surrogate models and the gradient calculation are relatively low. Besides, the maximum iteration we use is still little. So, compared with the black-box attack with random vector, the cost of LBAT is acceptable.

4. Experiment

This section presents the experiment setup, evaluation, method setting, and performance compared with state-of-the-art black-box attack methods.

4.1. Experiment setup

We use several representative benchmark datasets, including MNIST [1], CIFAR-10 [2], ImageNet [3].

The MNIST is composed of grayscale images composed of handwritten digits. The size of the image is (28 × 28 × 1). The training set contains 60,000 images, and the test set contains 10,000 images. The surrogate models and the victim model used in our experiment are trained by the same training set, but the structure is different. The data for the black-box attack comes from the test set. MNIST [1] has 10 categories.

The CIFAR-10 [2] is composed of relatively unclear RGB images, and the size of the image is (32 × 32 × 3). The training set contains 50,000 images, and the test set contains 10,000 images. The surrogate models and the victim model used in our experiment are all trained from the training set, but the structure is different. The data for the black-box attack comes from the test set.

ImageNet [3] is an RGB dataset. The dataset consists of more than 10 million images that belong to 1000 categories (there are also 1001 categories ImageNet [3] dataset). The classification accuracy of this dataset has long been used as the performance evaluation standard for image classification models. The dataset has multiple input sizes, and our experiment unified the data to 299 × 299 × 3. We use Inception-V3 [14] (require input: 299 × 299 × 3) as the victim model, InceptionResNet-V2 [45] (require input: 299 × 299 × 3), NASNetLarge [46] (require input: 331 × 331 × 3), Xception [47] (require input: 299 × 299 × 3), DenseNet121 [48] (require input: 224 × 224 × 3), MobileNetV2 [49] (input: 224 × 224 × 3) are the surrogate models. For surrogate models with different input requirements, we modify the image size to fit the models. The above models are all selected to be trained by the ImageNet [3] training set, and the black-box attack data comes from the test set.

We compare our attack with Tu [37], Chen [40] and Suya [43]. For [37], we use the Autoencoders provided in their Github. For Chen [40], we randomly pick the data whose label is different from the attacked ones from the training set as target images. For Suya [43], we use the surrogate models the same as we use in ours.

Since the difficulty of solving non-targeted attacks has been well resolved, our experiments mainly focus on targeted attacks. For MNIST [1] and CIFAR-10 [2] which have 10 categories of labels, we randomly select 10 images for each category to attack, and the target label includes all categories except their original labels. For ImageNet [3] which has 1000 categories of labels, we randomly select 100 images to attack. Meanwhile, we randomly select 2 labels as our target labels.

4.2. Evaluation index

We use the following indexes to evaluate the performance of the algorithm.

(1) Attack success rate. We use the attack success rate to evaluate the reliability of the attack. The query upper bound we set is different among the evaluated methods. The number of maximum queries varies with the methods' capabilities and characteristics. We set the maximum number of queries to 100,000 for Tu [37], Chen [40], Suya [43] and 5000 for LBAT for the reason that LBAT is more efficient for each query, but it takes longer time to generate.

(2) Mean query count of the initial attack. We record the number of queries required by the initial success attack each time and calculate the average of different attack methods. For decision-based attacks like Chen [40] which is different from the others, their number of queries of initial attack is not meaningful because of their large distortion, so we do not record this index.

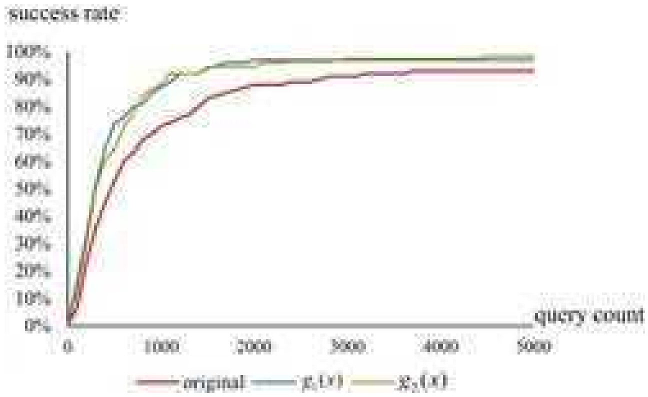

Fig. 4. LBAT Performance evaluation with different dynamic coefficient.

Table 1
Performance evaluation of black-box targeted attacks with different objective function on ImageNet [3].

Objective function	Successful rate	Distortion	Query count
Original	93.00%	6.36E−05	674.68
$g_1(x)$	97.00%	7.92E−05	413.27
$g_2(x)$	98.50%	7.18E−05	499.67

(3) Mean query count with per-pixel specific L_2 distortion. The attack stop condition we set is to satisfy a certain distortion requirement, instead of implementing an adversarial attack.

(4) Attack success rate in specific queries. We use the attack success rate under a specific number of queries. Similar to the maximum number of queries, the specific numbers set for different methods are also different.

(5) Attack success rate with a specific L_2 distortion. This index shows the percentage of attacks that satisfy the termination condition within the maximum number of queries.

(6) Mean per-pixel L_2 distortion. This index evaluates the invisibly of the adversarial attack.

4.3. Objective function selection

In Section 3.3, we have introduced our dynamic coefficients. In this section, we test the effect of dynamic coefficient through experiments and choose the best dynamic coefficient. The experiment is performed on the ImageNet [3].

The $g_1(x)$ and $g_2(x)$ in Fig. 4 and Table 1 denote the objective function adopted the dynamic coefficient equation (9) and Eq. (10) respectively. As shown in Fig. 4 and Table 1, we are able to find that dynamic coefficients can indeed reduce the number of queries and increase the success rate of adversarial attacks. However, dynamic coefficients will increase distortion slightly.

Both objective functions with dynamic coefficients have similar performance. Comparing $g_2(x)$ with $g_1(x)$, $g_2(x)$ has higher success rate and lower distortion, while requires 20% more queries. For the sake of balance, we use $g_2(x)$ as objective function in the following experiments.

4.4. Black-box attacks on MNIST and CIFAR-10

Tables 2 and 3 show the performance evaluation on MNIST [1] and CIFAR-10 [2] with different kinds of algorithms. Tu [37], Suya [43] and our algorithm use the zeroth-order optimization to optimize the algorithm, and we set the λ = 10 in Eq. (3).

Table 2
Performance evaluation of black-box targeted attacks on MNIST [1].

Method	Autozoom-AE [37]	Hop Skip Jump [40]	Hybrid batch attack [43]	Ours
Attack success rate (ASR)	100%	100%	100%	100%
Mean query count (initial success)	1279.64	–	603.39	62.26
Mean query count with per-pixel L_2 distortion<1E-4	23778.68	4612.24	2716.05	93.34
Attack success rate within 5000 queries	17.6%	–	63.9%	90.0%
Attack success rate L_2 distortion<1E-4	41.7%	100%	62.7%	100%
Mean per-pixel L_2 distortion (initial success)	7.8E−3	–	6.4E−3	3.7E−3

Table 3
Performance evaluation of black-box targeted attacks on CIFAR-10 [2].

Method	Autozoom-AE [37]	Hop Skip Jump [40]	Hybrid batch attack [43]	Ours
Attack success rate (ASR)	100%	100%	100%	100%
Mean query count (initial success)	213.71	–	37.14	30.98
Mean query count with per-pixel L_2 distortion<1E-4	5141.95	968.55	4150.92	41.91
Attack success rate within 5000 queries	59.1%	–	95.4%	98.2%
Attack success rate L_2 distortion<1E-4	100%	100%	24.4%	100%
Mean per-pixel L_2 distortion (initial success)	1.9E−3	–	1.49E−3	6.14E−4

As Tables 2 and 3 show, all algorithms can achieve a 100% success rate to generate an adversarial sample. We can find our algorithm needs much fewer queries than the others to generate adversaries. For example, the mean query counts required by our algorithm to get an initial success are reduced by 95.3%, 89.7% compared with Tu [37] and Suya [43] respectively on MNIST [1]. And on CIFAR-10 [2], the initial success is reduced by 85.5% and 16.6% respectively.

Our algorithm has a higher probability to generate adversarial samples successfully in specific queries. Tables 2 and 3 show that LBAT has 90% and 98.2% success rate to generate adversaries on MNIST [1] and CIFAR-10 [2] respectively.

Besides, LBAT can generate an initial adversarial sample with smaller distortion than others. For example, our initial success distortion is 47.4% of Tu [37] and 57.8% of Suya [43] on MNIST and 14.50% of Tu [37] and 83.41% Suya [43] on CIFAR-10 [2]. Not only our algorithm can generate an adversarial sample within the specific distortion with a higher success rate, but also need much fewer queries to reach the specific distortion.

According to the number of queries required for the initial success of different kinds of black-box attacks, we can infer that it is much easier to implement a black-box attack on CIFAR-10 [2] than MNIST [1]. We think the reason lies in that under the same number of labels of images, compared with MNIST [1], CIFAR-10 [2] has more dimensions to be modified to generate an adversarial sample.

It is easy to know that LBAT has better performance in almost all evaluation aspects applied to small datasets.

4.5. Black-box attacks on ImageNet

Table 4 shows the performance evaluation on ImageNet [3] with different kinds of black-box attack methods. For ImageNet [3], we set the λ = 10 and N = 1 (N denotes the number of vectors we use to evaluate gradient each step) for Tu [37] and Suya [43]. And we set λ = 100 for our algorithm. A larger λ means fewer generation queries, but it also produces greater distortion. [37,43] have a better balance effect when λ = 10. Our algorithm does not use auto-encoders and requires smaller distortion to generate adversarial samples. Therefore, we can set a larger λ to facilitate the generation of adversarial samples using fewer iterations. We show different λ performance evaluations in Table 5 and Fig. 5.

Fig. 5 shows the relationship between the objective function value and the number of iterations. When the iteration stops, a larger objective function value means a lower attack success rate and greater distortion. Fig. 5 logarithmizes the ordinate axis, using the logarithm to base 2. As shown in Fig. 5 and Table 5, when λ = 100, the balance of attack success rate, number of queries, and distortion of our algorithm is better than others. While λ = 10 requires much more queries and λ = 1000 cannot reduce the number of queries and produce larger distortion.

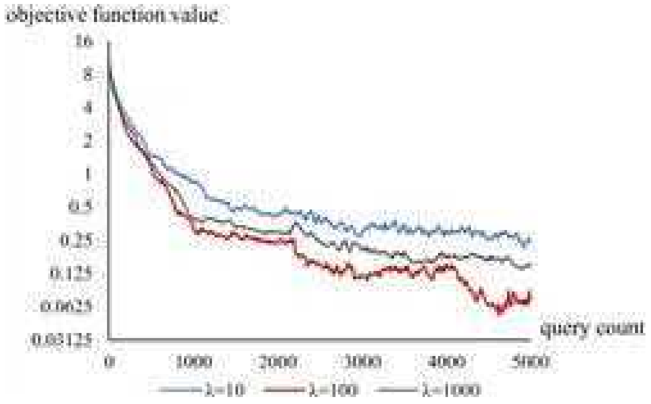

Fig. 5. LBAT performance evaluation with different λ.

As shown in Table 4, our algorithm is the only one that cannot generate an adversarial sample with a 100% successful rate, but 98.5% is still a very high success rate. We think the reason why we cannot achieve a 100% success rate is that the limitation of our reference models. We use 5 surrogate models in our algorithm, we can consider that we have 5 alternative optimization directions for each iteration. Although the alternative directions have a high probability to be a good optimization direction, they may still be bad optimization directions in a certain space, causing optimization shocks. Besides, it is obvious that the more surrogates models we use, the higher the success rate we can get. We can improve our success rate by adding more surrogate models.

As for the number of queries and the size of the distortion, LBAT is significantly less than others. At first, our queries for initial success are only 3.27% of the Tu [37] and 11.4% of the Suya [43]. Then our attack requires 1.11% and 2.58% number of the queries to generate attack within $1E - 4$ L_2 distortion compared with Tu [37] and Chen [40] respectively, and the Suya [43] even cannot generate an adversarial sample within such distortion, due to the large distortion caused by the ensemble attacks and autoencoder.

Our success rate in 5000 queries is the same as our initial success rate and nearly 9 times than Tu [37] and 150% than

Table 4
Performance evaluation of black-box targeted attacks on ImageNet [3].

Method	Autozoom-AE [37]	Hop Skip Jump [40]	Hybrid batch attack [43]	Ours
Attack success rate (ASR)	100%	100%	100%	98.50%
Mean query count (initial success)	15270.54	–	4376.02	499.67
Mean query count with per-pixel L_2 distortion<1E-4	44973.18	19393.63	None	499.67
Attack success rate within 5000 queries	11.50%	–	67.50%	98.50%
Attack success rate L_2 distortion<1E-4	90%	100%	0	98.50%
Mean per-pixel L_2 distortion (initial success)	1.4E−4	–	5.6E−4	7.18E−05

Table 5
Performance evaluation of black-box targeted attacks on ImageNet [3] with different λ.

λ	Attack success rate (ASR)	Mean query count (initial success)	Attack success rate L2 distortion <0.0001	Mean per-pixel L2 distortion (initial success)
λ = 10	95.0%	637.83	95.0%	5.92E−5
λ = 100	98.5%	499.67	98.5%	7.18E−5
λ = 1000	97.5%	578.14	89.0%	8.07E−5

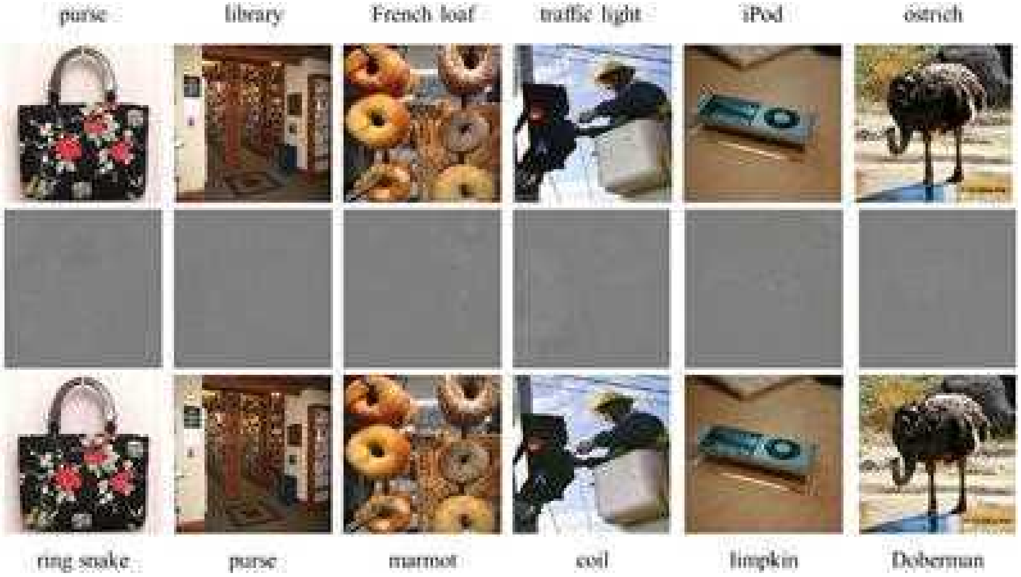

Fig. 6. Adversarial samples generated by LBAT.

Suya [43]. In addition, our initial success distortion is 51.3% of Tu [37] and 12.84% of Suya [43].

Compared with MNIST [1] and CIFAR-10 [2], adversarial attack on ImageNet [3] requires much more queries, due to the large search optimization space. However, LBAT can implement adversarial attacks on ImageNet [3] with a much lower query count.

The Fig. 6 shows a case study of LBAT on ImageNet [3]. The first row shows the original image, the second row shows the noise generated by the adversarial samples, and the third row is the generated adversarial samples. It can be seen that the distortion caused by our algorithm for adversarial samples generating is almost invisible.

Implementing black-box attacks on ImageNet [3] is challenging. However, our algorithm has excellent performance on attack tasks against ImageNet [3]. By comparing with the state-of-the-art methods, LBAT can reduce at least 88.6% query count to achieve the initial success and 97.42% query count to implement adversarial attack within specific L_2 distortion. Besides, our method has the highest success rate to reach specific distortion.

5. Conclusion

In this paper, we focus on deepening the understanding of the security of AI algorithms. Like other hybrid attack methods, our approach combines transfer-based and optimization-based attacks. Distinctively, we adopt the random model, random parameters, and dynamic coefficient strategies to calculate the adversarial vector to improve the efficiency of queries. By evaluating on multiple datasets, LBAT demonstrates its efficient query ability, that is, it can achieve the black-box attack with low distortion, low query counts, and high success rate. Our future work about the black-box adversarial attack will focus on performing our method with the surrogate models which are not training on the same dataset.

CRediT authorship contribution statement

Kangyi Ding: Methodology, Software, Writing - original draft. **Xiaolei Liu:** Data curation, Software. **Weina Niu:** Validation, Writing - review & editing. **Teng Hu:** Formal analysis. **Yanping Wang:** Validation. **Xiaosong Zhang:** Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was funded by the National Key Research and Development Plan 2016QY13Z2302, the National Natural Science Foundation of China under Grant No. 61902262, the Major Science and Technology Special Projects of Sichuan, China 19ZDZX0038, and the Director of Computer application Research Institute Foundation, China SJ2020A08.

References

[1] Y. Lecun, L. Bottou, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[2] A. Krizhevsky, G. Hinton, et al., Learning Multiple Layers of Features from Tiny Images, Citeseer, 2009.
[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, Ieee, 2009, pp. 248–255.
[4] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, 2014.
[6] K. He, X. Zhang, S. Ren, S. Jian, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision & Pattern Recognition, 2016.
[7] E. Yurtsever, J. Lambert, A. Carballo, K. Takeda, A survey of autonomous driving: Common practices and emerging technologies, IEEE Access 8 (2020) 58443–58469.
[8] Y. Huang, Y. Chen, Autonomous driving with deep learning: A survey of state-of-art technologies, 2020, arXiv preprint arXiv:2006.06091.
[9] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[10] R. Collobert, J. Weston, A unified architecture for natural language processing: Deep neural networks with multitask learning, in: Proceedings of the 25th International Conference on Machine Learning, 2008, pp. 160–167.
[11] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, 2014, arXiv preprint arXiv:1409.0473.
[12] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., Mastering the game of go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489.
[13] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al., Dota 2 with large scale deep reinforcement learning, 2019, arXiv preprint arXiv:1912.06680.
[14] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818–2826.
[15] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks, 2013, arXiv preprint arXiv:1312.6199.
[16] I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, 2014, arXiv preprint arXiv:1412.6572.
[17] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z.B. Celik, A. Swami, The limitations of deep learning in adversarial settings, in: 2016 IEEE European Symposium on Security and Privacy (EuroS&P), IEEE, 2016, pp. 372–387.
[18] X. Liu, T. Hu, K. Ding, Y. Bai, W. Niu, J. Lu, A black-box attack on neural networks based on swarm evolutionary algorithm, in: Australasian Conference on Information Security and Privacy, Springer, 2020, pp. 268–284.
[19] R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-cam: Visual explanations from deep networks via gradient-based localization, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 618–626.
[20] M. Sharif, S. Bhagavatula, L. Bauer, M.K. Reiter, Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition, in: Proceedings of the 2016 Acm Sigsac Conference on Computer and Communications Security, 2016, pp. 1528–1540.
[21] X. Liu, K. Wan, Y. Ding, X. Zhang, Q. Zhu, Weighted-sampling audio adversarial example attack, in: AAAI, 2020, pp. 4908–4915.
[22] N. Papernot, P. McDaniel, X. Wu, S. Jha, A. Swami, Distillation as a defense to adversarial perturbations against deep neural networks, in: 2016 IEEE Symposium on Security and Privacy (SP), IEEE, 2016, pp. 582–597.
[23] Y. Yu, P. Yu, W. Li, Auxblocks: Defense adversarial examples via auxiliary blocks, in: 2019 International Joint Conference on Neural Networks (IJCNN), IEEE, 2019, pp. 1–8.
[24] D. Gopinath, G. Katz, C.S. Pasareanu, C. Barrett, Deepsafe: A data-driven approach for checking adversarial robustness in neural networks, 2017, arXiv preprint arXiv:1710.00486.
[25] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, A. Vladu, Towards deep learning models resistant to adversarial attacks, 2017, arXiv preprint arXiv:1706.06083.
[26] H. Kannan, A. Kurakin, I. Goodfellow, Adversarial logit pairing, 2018, arXiv preprint arXiv:1803.06373.
[27] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, J. Zhu, Defense against adversarial attacks using high-level representation guided denoiser, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1778–1787.
[28] X. Jia, X. Wei, X. Cao, H. Foroosh, Comdefend: An efficient image compression model to defend adversarial examples, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6084–6092.
[29] N. Frosst, S. Sabour, G. Hinton, Darccc: Detecting adversaries by reconstruction from class conditional capsules, 2018, arXiv preprint arXiv:1811.06969.
[30] S.-M. Moosavi-Dezfooli, A. Fawzi, P. Frossard, Deepfool: a simple and accurate method to fool deep neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2574–2582.
[31] N. Carlini, D. Wagner, Towards evaluating the robustness of neural networks, in: 2017 Ieee Symposium on Security and Privacy (Sp), IEEE, 2017, pp. 39–57.
[32] I. Chaturvedi, Y.-S. Ong, R.V. Arumugam, Deep transfer learning for classification of time-delayed Gaussian networks, Signal Process. 110 (2015) 250–262.
[33] K. Weiss, T.M. Khoshgoftaar, D. Wang, A survey of transfer learning, J. Big data 3 (1) (2016) 1–40.
[34] Y. Liu, X. Chen, C. Liu, D. Song, Delving into transferable adversarial examples and black-box attacks, 2016, arXiv preprint arXiv:1611.02770.
[35] Y. Shi, S. Wang, Y. Han, Curls & whey: Boosting black-box adversarial attacks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 6519–6527.
[36] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, C.-J. Hsieh, Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models, in: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, 2017, pp. 15–26.
[37] C.-C. Tu, P. Ting, P.-Y. Chen, S. Liu, H. Zhang, J. Yi, C.-J. Hsieh, S.-M. Cheng, Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 742–749.
[38] A. Ilyas, L. Engstrom, A. Athalye, J. Lin, Black-box adversarial attacks with limited queries and information, 2018, arXiv preprint arXiv:1804.08598.
[39] W. Brendel, J. Rauber, M. Bethge, Decision-based adversarial attacks: Reliable attacks against black-box machine learning models, 2017, arXiv preprint arXiv:1712.04248.
[40] J. Chen, M.I. Jordan, M.J. Wainwright, Hopskipjumpattack: A query-efficient decision-based attack, in: 2020 IEEE Symposium on Security and Privacy (SP), IEEE, 2020, pp. 1277–1294.
[41] T. Brunner, F. Diehl, M.T. Le, A. Knoll, Guessing smart: Biased sampling for efficient black-box adversarial attacks, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 4958–4966.
[42] S. Cheng, Y. Dong, T. Pang, H. Su, J. Zhu, Improving black-box adversarial attacks with a transfer-based prior, in: Advances in Neural Information Processing Systems, 2019, pp. 10934–10944.
[43] F. Suya, J. Chi, D. Evans, Y. Tian, Hybrid batch attacks: Finding black-box adversarial examples with limited queries, in: 29th {USENIX} Security Symposium ({USENIX} Security 20), 2020, pp. 1327–1344.
[44] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.
[45] C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31, 2017.
[46] B. Zoph, V. Vasudevan, J. Shlens, Q.V. Le, Learning transferable architectures for scalable image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8697–8710.
[47] F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1251–1258.
[48] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.
[49] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, Mobilenetv2: Inverted residuals and linear bottlenecks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4510–4520.