The Journal of Systems & Software 170 (2020) 110767

Contents lists available at ScienceDirect

The Journal of Systems & Software

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

# Black-box adversarial sample generation based on differential evolution

Junyu Lin [a,b], Lei Xu [a,b,*], Yingqi Liu [c], Xiangyu Zhang [c]

[a] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
[b] Department of Computer Science and Technology, Nanjing University, Nanjing, China
[c] Department of Computer Science, Purdue University, West Lafayette, USA

ARTICLE INFO

*Article history:*
Received 27 September 2019
Received in revised form 19 July 2020
Accepted 28 July 2020
Available online 1 August 2020

*Keywords:*
Adversarial samples
Differential evolution
Black-box testing
Deep Neural Network

ABSTRACT

Deep Neural Networks (DNNs) are being used in various daily tasks such as object detection, speech processing, and machine translation. However, it is known that DNNs suffer from robustness problems — perturbed inputs called adversarial samples leading to misbehaviors of DNNs. In this paper, we propose a black-box technique called *Black-box Momentum Iterative Fast Gradient Sign Method* (BMI-FGSM) to test the robustness of DNN models. The technique does not require any knowledge of the structure or weights of the target DNN. Compared to existing white-box testing techniques that require accessing model internal information such as gradients, our technique approximates gradients through *Differential Evolution* and uses approximated gradients to construct adversarial samples. Experimental results show that our technique can achieve 100% success in generating adversarial samples to trigger misclassification, and over 95% success in generating samples to trigger misclassification to a specific target output label. It also demonstrates better perturbation distance and better transferability. Compared to the state-of-the-art black-box technique, our technique is more efficient. Furthermore, we conduct testing on the commercial Aliyun API and successfully trigger its misbehavior within a limited number of queries, demonstrating the feasibility of real-world black-box attack.



## 1. Introduction

In the past few years, Deep Neural Networks (DNNs) (LeCun et al., 2015) have achieved great success in many important applications, such as image classification (Krizhevsky et al., 2012), speech processing (Sak et al., 2015) and machine translation (Bahdanau et al., 2014). Their efficacy even outperforms humans. Modern software applications increasingly include DNNs as a critical component, exampled by autonomous software (Bojarski et al., 2016), Apple face ID,[1] and Amazon Echo.[2] It can be envisioned that DNN model engineering will become an essential step in the software development lifecycle. As such, testing and debugging DNN models is of importance.

However, researchers have revealed that DNNs have robustness problems. That is, they are vulnerable to adversarial samples, i.e., benign inputs that add small and imperceptible perturbation and cause DNNs to misclassify. Adversarial samples hinder the utilization of DNNs in safety-critical systems, especially those related to computer vision, including face recognition (Sharif et al., 2019), self-driving vehicles (Evtimov et al., 2018) and medical analysis (Litjens et al., 2017). To DNN based applications, adversarial samples are threats but also a method to test DNN models. Our work falls into efficiently and effectively generating adversarial samples to expose robustness problems of DNNs.

Adversarial sample generation methods (Szegedy et al., 2013; Goodfellow et al., 2015; Carlini and Wagner, 2017; Papernot et al., 2017; Su et al., 2019; Chen et al., 2017; Jere et al., 2019) have two categories: *white-box* techniques and *black-box* techniques. The former requires access to model internals such as model structure, neuron weight values, and gradients. In contrast, the latter treats the subject model as a black box and does not require access to model internals, but rather just model outputs. Black-box techniques have broader applicability. They can be used to test remote applications powered by underlying DNNs. For example, internet service providers such as Alibaba Cloud[3] and Google Cloud[4] offer pay-for-use APIs, which hide the internal details from users so that white-box techniques are not applicable. As such, black-box techniques are necessary.

In this paper, we develop a new black-box adversarial sample generation method in image classification. For a subject model,

* Corresponding author.
*E-mail address:* xlei@nju.edu.cn (L. Xu).

[1] https://machinelearning.apple.com/2017/11/16/face-detection.html
[2] https://developer.amazon.com/alexa/science
[3] https://www.alibabacloud.com/
[4] https://cloud.google.com/

https://doi.org/10.1016/j.jss.2020.110767
0164-1212/


**Fig. 1.** Our proposed black-box BMI-FGSM on ImageNet samples. All adversarial samples are misclassified by target DNN (Perturbations are magnified for better visualization).

we only assume access to model inputs and outputs. Our approach called *Black-box Momentum Iterative Fast Gradient Sign Method* (BMI-FGSM) then modifies the original inputs to trigger model misclassification. It aims to achieve the goal with as little mutation as possible, utilizing *Differential Evolution* (Storn and Price, 1997) to approximate gradient direction, and leveraging *double step size* and *candidate reuse* to improve efficiency, which will be explained later.

We compare BMI-FGSM with other state-of-the-art white-box and black-box techniques in both the untargeted setting (i.e., misclassifying to any output label) and the targeted setting (i.e., misclassifying to a specific output label). We also consider the transferability, that is, whether an adversarial sample generated for one model can be used to trigger misbehavior of another model. The experiments on a set of widely used datasets and models show that our proposed BMI-FGSM can achieve a high success rate in both untargeted and targeted settings, comparable or even better than white-box methods. Our approach can generate an adversarial sample within half a minute, faster than other black-box techniques. Finally, we apply our approach to a real-world image recognition system and expose its robustness problem.

In summary, we make the following contributions.

- We propose a novel black-box adversarial sample generation method named BMI-FGSM. This technique does not require knowledge about model architecture, weight values, or gradients.
- We propose two novel methods to improve performance, *double step size* to enlarge exploration distance and *candidate reuse* to approximate momentum that provides guidance for input perturbation, which is critical for generating effective adversarial samples.
- We compare BMI-FGSM with the state-of-the-art methods. Experimental results on the MNIST, CIFAR10, and ImageNet datasets show that our approach is more efficient than the black-box method ZOO (Chen et al., 2017) and has a comparable success rate as the white-box method MI-FGSM (Dong et al., 2018). Furthermore, we use BMI-FGSM to test the commercial Aliyun Image Recognition API and successfully trigger misbehavior.

The remainder of this paper is structured as follows. Section 2 reviews existing research related to adversarial samples. Section 3 presents our black-box test case generation technique. Section 4 posts three research questions and then answers them with experiments. Section 5 presents the conclusions and future work.

## 2. Related work

Existing works on the DNN robustness problem and adversarial sample generation method in image classification will be reviewed in the following subsections.

### 2.1. DNN robustness

Robustness is critical to the application of DNNs. A popular approach to improving model robustness is to provide additional training and validation data. There are mainly two ways of generating additional data, *data augmentation* (Zhong et al., 2017; Perez and Wang, 2017) and using Generative Adversarial Network (GAN) (Goodfellow et al., 2014; Wang et al., 2018). The former augments training dataset by transforming original data, e.g., move, rotate, flip, and scale an image to produce new ones. A GAN model is composed of a generator and a discriminator. The generator takes random input and tries to mutate it to a valid input, while the discriminator determines if the mutated one looks like a real input. The two parts compete with each other and ideally the generator would learn to generate real samples. However, existing data augmentation techniques and GANs have limited effectiveness. Hence, a more practical method to gauge and improve DNN robustness is through adversarial samples. Specifically, original input samples are perturbed to generate adversarial samples to trigger model misclassification. The training set can be enhanced with the adversarial samples to retrain the DNN model to improve robustness (Tramer et al., 2017; Madry et al., 2017). With adversarial training, DNNs are expected to be less sensitive to noises or perturbations.

### 2.2. Adversarial sample generation

Adversarial sample generation methods can be categorized into white-box methods and black-box methods.

White-box methods assume full knowledge of the target DNN, such as model architecture and neuron weights. Szegedy et al. (2013) observed that adding small perturbations to input images can cause DNN model misclassification and converted the generation of adversarial samples to a constrained minimization problem. Goodfellow et al. (2015) proposed "fast gradient sign method" (FGSM), a gradient-based method aiming at a very short generation time. Kurakin et al. (2016) proposed a basic iterative method to generate more powerful samples. Dong et al. (2018) introduced momentum and proposed "momentum iterative fast gradient sign method" (MI-FGSM) to further balance the trade-off between success rate and transferability. Carlini and Wagner (2017) proposed the C&W method, an optimization-based technique systematically builds examples by directly optimizing

the perturbation with an Adam optimizer. Additional proposed mechanisms include binary search and change of variable space.

Black-box methods assume no access to model internals. Instead, they can query the target model by sending inputs and observe the corresponding outputs. Compared with white-box techniques, black-box methods can be used to perform testing to third-party models and have much broader applicability. Papernot et al. (2017) assumed no internal information and insufficient training data, and proposed to train a substitute model with a small synthetic training dataset. Narodytska and Kasiviswanathan (2017) discovered the phenomenon that is merely modifying a single pixel might lead to model misclassification. Su et al. (2019) exposed DNN robustness problems by leveraging Differential Evolution to search for this kind of pixel. Chen et al. (2017) proposed Zeroth Order Optimization (ZOO), a derivative-free generation method. The authors exploited a finite differencing method to calculate the approximate gradient by analyzing two very close points in the loss function.

Another type of DNN robustness testing is the transferability testing (or no-box method) (Szegedy et al., 2013; Moosavi-Dezfooli et al., 2017). These techniques do not query the target model, neither do they generate any samples. Instead, they use adversarial samples produced by another model to test the target DNN. The underlying assumption is that if an adversarial sample can confuse a model, it is likely that it is equally confusing for another model. Therefore, transferability can be considered an important quality metric of generalization for the adversarial test cases.

We aim at developing a black-box adversarial sample generation method that features high success rate, high transferability, and cost-effectiveness. Many parallel works have also studied the problem of black-box adversarial generation, but our work remains unique in the approach. Ilyas et al. (2018) uses a natural evolution strategy (which can be seen as a finite differences estimate on a random gaussian basis) to estimate the gradients for use in the projected gradient descent method. Following this work, Ilyas et al. (2019) formalizes the gradient estimation problem and develop a bandit optimization framework incorporating time and data-dependent information, to generate black-box adversarial samples. In the same threat model, we leverage Differential Evolution to approximate gradient sign to convert a white-box iterative gradient-based method to its black-box version that only requires accessing model outputs. Alzantot et al. (2019) develops a gradient-free approach for generating adversarial examples by leveraging genetic optimization, where the fitness function is defined similarly to CW loss, using prediction scores from the black-box model. The authors adopt dimensionality reduction and adaptive parameter scaling for boosting gradient-free optimization. In contrast, our approach models the gradient sign, combining with our *double step size* and *candidate reuse* strategies enables attacks that can reliably generate adversarial samples. Another line of work aims to generate adversarial samples in different scenarios. Cheng et al. (2019) focuses on the label-only setting and propose a generic optimization algorithm, which can be applied to discrete and non-continuous models other than neural networks, such as the decision tree. Suya et al. (2019) simulates a scenario where the attacker has access to a large pool of seed inputs and proposes a hybrid strategy that combines optimization-based and transferability-based methods.

### 2.3. Testing DNNs

Compared with traditional software, the behavior of a Deep Learning (DL) system is determined by the structure and weights of DNNs. The dimension and test space of DNN is often larger. DeepXplore (Pei et al., 2017) proposes a white-box differential testing algorithm for systematically finding inputs that can trigger different behaviors between multiple DNNs. They propose neuron coverage for systematically measuring the parts of a DNN exercised by test inputs. Tensorfuzz (Odena and Goodfellow, 2018) provides coverage-guided fuzzing methods for neural network by using the approximate nearest neighbor algorithm. DeepTest (Tian et al., 2018) generates test cases that maximize the numbers of activated neurons and finds erroneous behaviors under different realistic driving conditions (e.g., blurring, rain, and fog). DeepGauge (Ma et al., 2018a) proposes multi-granularity testing coverage for DL systems based on the internal state of DNN. Their testing criteria are scalable to complex DNNs. DeepCT (Ma et al., 2019) adapts the notion of combinatorial testing and introduces a set of coverage criteria based on neuron input interactions for each layer of DNNs to guide test generation towards balancing the defect detection ability and the number of tests. DeepCover (Sun et al., 2018) proposes a set of four test criteria for DNNs, inspired by the MC/DC test criteria in traditional software. They evaluate the proposed test criteria on small scale neural networks and show a higher defect detection ability than random testing. DeepMutation (Ma et al., 2018b) adapts the concept of mutation testing and proposes a mutation testing framework specialized for DNNs to measure the quality of test data. They believe that mutation testing is a promising technique that helps to generate higher quality test data.

Existing works on testing the DL system detect erroneous behaviors of DNNs under realistic circumstances (e.g., lighting, occlusion), and mostly focus on testing criteria, which requires DNNs to be transparent. In contrast, adversarial sample generation demonstrates a particular type of erroneous behavior of DNNs. Our proposed method tests black-box DL systems with adversarial samples that are indistinguishable from the original ones, to expose misbehaviors from the attacker's perspective.

## 3. BMI-FGSM algorithm

Our technique, called *Black-box Momentum Iterative Fast Gradient Sign Method* (BMI-FGSM), is inspired by iterative gradient-based methods and *Differential Evolution*. It features mechanisms to improve the efficiency and effectiveness of generation. Fig. 2 is the basic framework of our method. Given a clean input image and a pre-trained DNN, Differential Evolution first derives a gradient sign population, children candidates compete with their parents using the corresponding perturbed inputs. Then, we perturb the inputs with the approximate gradient signs. The process repeats until an adversarial sample very similar to the benign input is produced.

In the following sections, iterative gradient-based methods are first introduced, and Differential Evolution is presented, followed by the complete algorithm with two improvement mechanisms *double step size* and *candidate reuse*.

### 3.1. Iterative gradient-based methods

Throughout this paper, we focus on a $n$-classifier DNN model (LeCun et al., 1998) of image classification task where $I \in \mathbb{R}^m$ is the input image and $F(I) \in \mathbb{R}^n$ is the $n$-dimension output that denotes the predicted labels. More formally, we define a DNN as follows

$$F(I) = \text{softmax}(Z(I)) \tag{1}$$

$$C(I) = \text{argmax}_i\{F_i(I)\} \tag{2}$$

Here, $Z(I)$ is computed by hidden layers and a set of neuron weight matrices, known as the *logits*. The softmax function normalizes the output so $F(I)$ denotes the probability distribution of the predicted labels i.e. $F_i(I) \in [0, 1] \wedge \sum_i F_i(I) = 1$, where $F_i(I)$

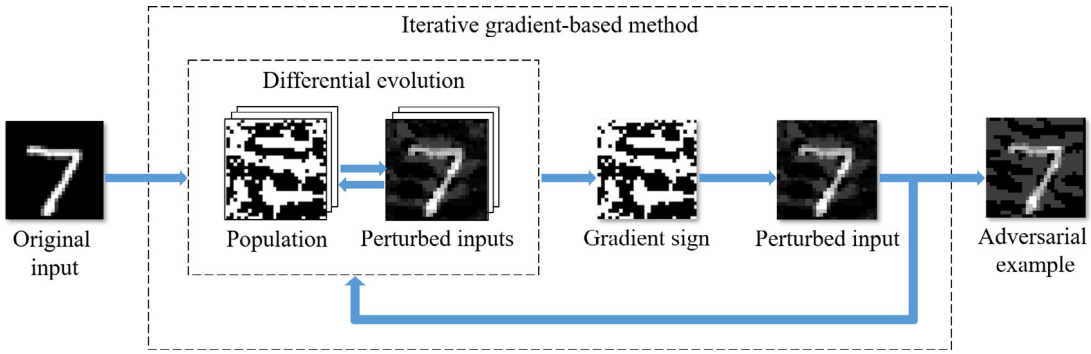



**Fig. 2.** Basic framework of the Black-box Momentum Iterative Fast Gradient Sign Method.

is the probability that input $I$ belongs to label $i$. Thus the final classification label $C(I)$ is the label with the highest probability.

During the black-box adversarial sample generation, we assume the DNN model is pre-trained and fixed (i.e., neuron weight values are fixed). We assume access to the input image, output label, and output confidence, which is common for black-box methods. We do not assume any access to the structure, neuron weights, or intermediate outputs. Given a clean input $I$ and its ground-truth label $y$, we call $I_{adv}$ an adversarial sample if

$$C(I_{adv}) \neq y \wedge \|I_{adv} - I\|_p \leq \epsilon \quad \text{(untargeted)} \tag{3}$$

$$C(I_{adv}) = q \wedge q \neq y \wedge \|I_{adv} - I\|_p \leq \epsilon \quad \text{(targeted)} \tag{4}$$

where in untargeted setting sample is misclassified to any false label and in targeted setting sample misclassified to a specific false label $q$. A threshold $\epsilon$ controls the perturbation magnitude by limiting the $L_p$-norm distance.

Iterative gradient-based methods are white-box generation methods that iteratively calculate the gradient and perturb the input. These methods feature high efficiency, success rate, and extensibility, usually require a small number of back propagations to craft adversarial samples in the $L_\infty$ neighborhood of the original input. For example, the *basic iterative method* (Kurakin et al., 2016) produces adversarial sample by applying gradient $T$ times with a small step size:

$$I_t = I_{t-1} + \frac{\epsilon}{T} \cdot \text{sign}(\nabla_{I_{t-1}} J(F(I_{t-1}), y)) \tag{5}$$

where $\nabla$ represents the gradient and $J(\cdot)$ is the cost function. It can finish the generation fast and yield superior results, also known as the "iterative fast gradient sign method". To further balance the trade-off between success rate and transferability, an improved version of the basic iterative method introduces momentum (Dong et al., 2018) by replacing the raw gradient with an accumulated gradient $g_t$:

$$g_t = \mu \cdot g_{t-1} + \frac{\nabla_{I_{t-1}} J(F(I_{t-1}), y)}{\left\| \nabla_{I_{t-1}} J(F(I_{t-1}), y) \right\|_1} \tag{6}$$

The critical step of iterative gradient-based methods is the gradient computation. In white-box attacks, gradients are usually computed by back propagation.

### 3.2. Differential evolution

Differential Evolution (DE) (Storn and Price, 1997) is an evolutionary algorithm to solve optimization problems. DE tries to find the input that best fits the objective through input mutation. It has three phases: evolution, crossover, and selection. During the phases of evolution and crossover, new children candidate solutions are generated. In selection, by comparing children with their parents, the ones with better fitness survive and are chosen for the next iteration of evolution. DE has the following advantages: (i) DE algorithm is independent of DNN, and hence any improvements of DE (Das and Suganthan, 2011; Das et al., 2009; Zhang and Sanderson, 2009) are also directly applicable. (ii) DE solves non-differentiable, noisy, and dynamic problems such that it can handle both continuous and discontinuous problems without requiring understanding internal structure of the problems. (iii) DE has a heuristic-based diversity insurance mechanism which can prevent from being trapped in local maxima and minima while standard greedy search and gradient descent often cannot.

In our algorithm, we employ DE to search for the gradient sign. Denote $I \in \mathbb{R}^m$ as the input image, $y$ as the ground-truth label, $F(I) \in \mathbb{R}^n$ as the outputs of DNN. We initialize the first generation $x(0)$ as follows.

$$x(0) = \{x_i(0) | x_{ij}(0) = \text{random}\{-1, 1\}; 1 \leq i \leq N; 1 \leq j \leq m\} \tag{7}$$

where $x_i(0)$ denotes the gradient signs with the same size as $I$, and is the $i$th candidate of the 0th generation. Variable $x_{ij}(0)$ is the $j$th feature randomly initialized with 1 or $-1$. $N$ is the size of population (i.e, the number of candidates in a generation). Fig. 3(a) visualizes a candidate in $x(0)$. The candidate is essentially a matrix of the same size as the input image.

The evolution and crossover at the $g$th generation is defined as follows.

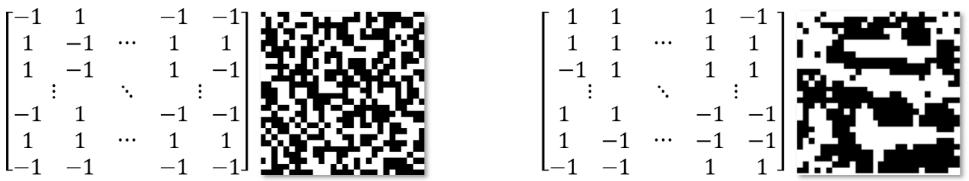
$$v_i(g+1) = \text{sign}(x_{r_1}(g) + DR \cdot (x_{r_2}(g) - x_{r_3}(g))) \quad (i \neq r_1 \neq r_2 \neq r_3) \tag{8}$$

$$u_{ij}(g+1) = \begin{cases} v_{ij}(g+1) & \text{rand}(0, 1) < CR \\ x_{ij}(g) & \text{otherwise} \end{cases} \tag{9}$$

where $DR$ denotes the scaling factor of differential vector, $CR \in [0, 1]$ denotes the crossover probability to control crossover variants. Intuitively, Eq. (8) denotes that a mutant $v_i(g+1)$ is derived from three randomly selected candidates from the previous generation $x(g)$, denoted by the three random numbers $r_1$, $r_2$, and $r_3$. We assign a sign function because it returns only two valid values representing two different gradient directions, and leads to fast convergence. Eq. (9) denotes that an offspring $u_i(g+1)$ is formed by recombining the mutant with the previous candidate, and $u_{ij}(g+1)$ is the $j$th feature of $u_i(g+1)$.

Next, compare fitness of $x(g)$ and $u(g+1)$ to determine the next generation $x(g+1)$. However, it is hard to set a fitness function for candidates since they represent gradient signs, so we convert candidates to a set of perturbed images first (Eq. (5), replace sign$(\cdot)$ with candidate), and then apply the fitness function. For the DNNs we test, although making the confidence of $y$ as fitness works acceptably well, we ultimately settle on this following fitness function.

$$f(I') = \begin{cases} F_y(I') - \max_{i \neq y}\{F_i(I')\} & \text{untargeted} \\ \max_{i \neq q}\{F_i(I')\} - F_q(I') & \text{targeted} \end{cases} \tag{10}$$



(a) (b)

**Fig. 3.** Visualization of two candidates in Differential Evolution. White pixels denote the corresponding matrix elements are 1. (a) a candidate in the initial population $x(0)$; (b) a candidate in the last population $x(G)$.

**Algorithm 1** Approximate gradient signs

**Input:** base sample $I$, generation $x(0)$, perturbation distance $\alpha$, DE iterations $G$;
**Output:** last generation $x(G)$;
1: $I(0) \Leftarrow I + \alpha \cdot x(0)$;
2: **for** $g = 0 : G - 1$ **do**
3: Generate next generation $u(g+1)$ from $x(g)$;
4: $I(g+1) \Leftarrow I + \alpha \cdot u(g+1)$;
5: Compare $f(I(g))$ with $f(I(g+1))$ and select $x(g+1)$;
6: **end for**
7: **return** last generation $x(G)$;

**Algorithm 2** BMI-FGSM

**Input:** clean example $I_0$, population size $N$, perturbation distance $\epsilon$, DE iterations $G$, iterative gradient-based method iterations $T$, candidate keeping rate $KR$;
**Output:** adversarial sample $I_T$;
1: Initialize $N$ candidates $x(0)$
2: $\alpha \Leftarrow \epsilon$
3: $\beta \Leftarrow \epsilon / T$
4: **for** $t = 0 : T - 1$ **do**
5: $x(G) \Leftarrow ApproxGradientSigns(I_t, x(0), \alpha, G)$;
6: $I(G) \Leftarrow I_t + \beta \cdot x(G)$;
7: $I_{t+1} \Leftarrow \underset{i \in I(G)}{\operatorname{argmin}} f(i)$;
8: Keep candidates of best $KR * 100\% * N$ fitness in $x(G)$ and initialize other candidates;
9: $x(0) \Leftarrow x(G)$;
10: $\alpha \Leftarrow \alpha - \beta$;
11: **end for**
12: **return** adversarial sample $I_T$;

The lower the $f(I')$, the higher fitness value an input $I'$ has. This fitness function aims to suppress the probability of the ground-truth label while enhancing the maximum probability among other false labels. Fig. 3(b) visualizes one of the candidates which have evolved $G$ generations. We can observe the pattern that serves as approximate gradient signs.

### 3.3. Black-box momentum iterative fast gradient sign method

Using DE, we leverage Algorithm 1 to approximate the gradient signs. Children candidates are generated by Eqs. (8) and (9). We cannot directly compare candidates, so we convert candidates to perturbed images (line 4). The perturbed images' fitness values are compared to determine the offspring using Eq. (10). Candidate with the lowest fitness score of $x(G)$ is the final approximate gradient signs. Note that in black-box testing, the additional overhead by DE is inevitable because in white-box testing the "exact gradient signs" can be computed by back propagation in milliseconds. Note that in the black-box attack context, precise estimation of gradient signs is neither possible nor necessary.

We then extend our work to BMI-FGSM, an algorithm inheriting the advantages of both DE and iterative gradient-based method. As shown in Algorithm 2, the loop denotes the main procedure of the iterative gradient-based method. After approximating gradient signs with a base sample $I_t$, we apply $x(G)$ to $I_t$ again and generate perturbed images $I(G)$, the next base sample $I_{t+1}$ will be the fittest one of $I(G)$. After $T$ iterations, image $I_T$ is returned as the adversarial sample. The most challenging part of BMI-FGSM is to escape local optimums and re-gain momentum for better results. Hence, we design two mechanisms called *double step size* and *candidate reuse*.

#### 3.3.1. Double step size

There are two types of perturbations in our proposed BMI-FGSM. The first one happens in approximate gradient signs (Algorithm 1, line 4). It is a "temporary perturbation" in which we always perturb the same base sample $I$, and generate a set of temporary images $I(g+1)$ for fitness evaluation. The other one is in the loop of iterative gradient-based method (Algorithm 2, line 6). It is the "permanent perturbation" in which we iteratively perturb the clean image from $I_0$ to $I_T$.

Either perturbation requires a perturbation distance, usually it is $\epsilon/T$, but there would be a problem if we just use the same distance — the DE loop would evaluate candidates using temporary images that are perturbed with a very small distance. The DE procedure hence runs the risk of being stuck in some local optima. Note that the white-box version of iterative gradient-based method does not suffer from this step size problem as they always calculate exact gradient signs by back propagation.

As a result, we design the *double step size*: exploiting two distances for different perturbations. Specifically, $\alpha$ initialized with $\epsilon$ is for gradient signs calculation, and $\beta$ initialized with $\epsilon/T$ is for the permanent perturbation. At the end of iteration $t$, we have perturbed $I_t$ to $I_{t+1}$ with a distance $\beta$, and hence decreased the exploration distance by updating $\alpha \leftarrow \alpha - \beta$ to satisfy $\alpha + t \cdot \beta \equiv \epsilon$, where parameter $\epsilon$ is still the only overall perturbation distance of adversarial sample. Fig. 4 shows the difference with and without double step size. The circle area denotes local optima. Points $I_*$ represent the movement of permanent perturbations. Points $I'_*$ denote the best position that DE can find. Without double step size, perturbations are short-sighted so that the path would be trapped for a while. With double step size, we can explore further and hence guide the procedure to escape from the local optima area.

Intuitively, the double step size enables us to use a more considerable perturbation distance when approximating gradient signs. Dynamic exploration distance enlarges the potential to achieve better fitness. With the assistance of double step size, we improve the success rate of sample generation on ImageNet.

#### 3.3.2. Candidate reuse

In the training of DNNs, momentum (Polyak, 1964) is introduced to improve gradient descent. Using accumulated gradients rather than the raw gradients helps escape local optima and converge fast. Researchers in the white-box MI-FGSM project (Dong et al., 2018) leverage this idea and integrate momentum in their techniques. Accumulated gradient balances the trade-off between transferability and perturbation distance, producing high-quality adversarial samples. But according to our tests, the way of constructing an accumulated gradient cannot be directly ported to our method for two reasons: (i) The accumulation of approximate

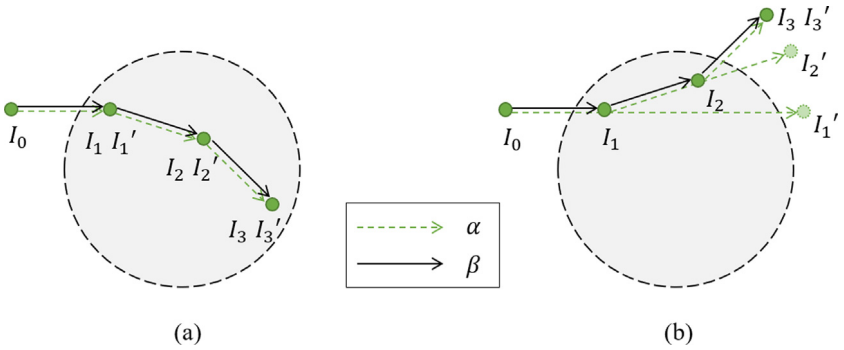



**Fig. 4.** An example of double step size. (a) no double step size; (b) double step size.

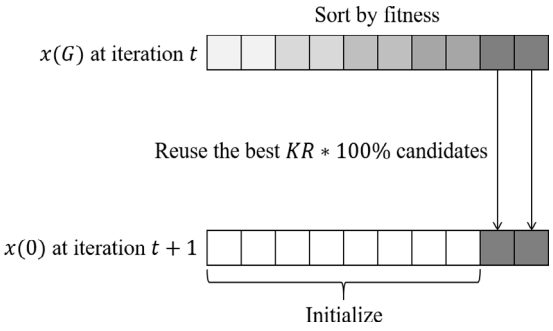



**Fig. 5.** An example of candidate reuse.

gradient signs also means the accumulation of errors. (ii) Candidates have only two valid values 1 and −1, so the accumulated gradient Eq. (6) is not appropriate.

We design a novel mechanism *candidate reuse* to achieve effect similar to gradient accumulation. Specifically, while generation $x(G)$ holds $N$ candidate gradients, our technique saves the best $KR * 100\% * N$ candidates as part of the initial population of next round, where $KR$ is the keeping rate. Fig. 5 gives an example of how candidate reuse works. The reused candidates that carry previous iterations' gradient information, participate in the evolution and play the role of "momentum". The candidate reuse method is more practical in the black-box attack context than the accumulated gradient. Applying candidate reuse, we make BMI-FGSM available on ImageNet.

## 4. Experiments

We use an Intel Xeon E5 CPU and an Nvidia Tesla K80 GPU for all model trainings and sample generations. We aim to answer the following three research questions:

- RQ1: How does BMI-FGSM perform compared with existing methods?
- RQ2: How is the transferability of BMI-FGSM?
- RQ3: How do double step size and candidate reuse improve BMI-FGSM on large model and dataset?
- RQ4: How dose BMI-FGSM perform testing third-party applications?

We first report that BMI-FGSM is substantially faster than existing black-box techniques and achieves comparable performance with existing white-box methods. Then, we show that the two novel mechanisms do improve performance. Finally, we present the results of generating adversarial samples on a third-party system: the Aliyun Image Recognition API.

**Table 1**
Evaluation of adversarial sample generation methods on MNIST.

| Methods | Untargeted | | | Targeted | | |
|---|---|---|---|---|---|---|
| | Success | Avg. dist | Avg. time | Success | Avg. dist | Avg. time |
| C&W | 100.0% | – | 33.4 s | 100.0% | – | 34.0 s |
| MI-FGSM | 100.0% | 0.195 | 0.1 s | 100.0% | 0.247 | 0.2 s |
| ZOO | 100.0% | – | 68.4 s | 97.8% | – | 87.4 s |
| BMI-FGSM | 100.0% | 0.227 | 16.7 s | 98.2% | 0.314 | 23.7 s |

### 4.1. RQ1: Performance comparison

#### 4.1.1. Dataset and models

For the MNIST (LeCun et al., 1998) and CIFAR10 (Krizhevsky and Hinton, 2009) datasets. We randomly choose 100 images for evaluation and sample 10 images from each class. For each image, we apply one untargeted attack and nine targeted generations for the nine other classes. Thus there are 100 untargeted samples and 900 targeted samples in total for each dataset.

The target DNN models we used are based on the settings of Carlini and Wagner (2017, Table 1). We train and get 99% accuracy for MNIST and 84% accuracy for CIFAR10. Data were normalized before classification. Misclassified clean images are ignored during generation.

#### 4.1.2. Comparison methods

On MNIST and CIFAR10, our proposed BMI-FGSM is compared with three state-of-the-art adversarial sample generation techniques C&W (Carlini and Wagner, 2017), ZOO (Chen et al., 2017) and MI-FGSM (Dong et al., 2018) in both untargeted and targeted settings. For all comparison methods, we refer to the code implementation of Foolbox[5] with necessary modifications.

- C&W (Carlini and Wagner, 2017, white-box, optimization-based, $L_2$-norm) builds examples by directly optimizing the perturbation with an optimizer.
- ZOO (Chen et al., 2017, black-box, optimization-based, $L_2$-norm) exploits finite difference method to calculate approximate gradient.
- MI-FGSM (Dong et al., 2018, white-box, gradient-based, $L_\infty$-norm) introduces momentum to improve the basic iterative method.

#### 4.1.3. Parameters

For C&W, we conduct 9 binary searches for the best scale parameter $c$ starting from 0.01. We use $\eta = 0.01$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ for the Adam optimizer. The confidence $k = 10$ for robustness and the max perturbation distance is set to 20. We run

[5] https://github.com/bethgelab/foolbox

**Table 2**
Evaluation of adversarial sample generation methods on CIFAR10.

| Methods | Untargeted | | | Targeted | | |
|---|---|---|---|---|---|---|
| | Success | Avg. dist | Avg. time | Success | Avg. dist | Avg. time |
| C&W | 100.0% | – | 11.3 s | 100.0% | – | 12.7 s |
| MI-FGSM | 100.0% | 0.020 | <0.1 s | 100.0% | 0.035 | <0.1 s |
| ZOO | 100.0% | – | 143.9 s | 94.8% | – | 189.4 s |
| BMI-FGSM | 100.0% | 0.034 | 20.6 s | 96.3% | 0.047 | 30.4 s |

**Table 3**
Evaluation of transfer rate on MNIST.

| Methods | Untargeted | | Targeted | |
|---|---|---|---|---|
| | FC | LeNet5 | FC | LeNet5 |
| C&W | 2.0% | 5.0% | 6.0% | 11.0% |
| MI-FGSM | 30.0% | 20.0% | 42.2% | 45.4% |
| ZOO | 3.0% | 1.0% | 3.1% | 0.2% |
| BMI-FGSM | 38.0% | 36.0% | 39.2% | 24.0% |

**Table 4**
Evaluation of transfer rate on CIFAR10.

| Methods | Untargeted | | | Targeted | | |
|---|---|---|---|---|---|---|
| | AllConv | NiN | VGG16 | AllConv | NiN | VGG16 |
| C&W | 12.9% | 8.2% | 16.5% | 18.8% | 11.8% | 18.8% |
| MI-FGSM | 26.8% | 19.4% | 31.6% | 46.7% | 38.3% | 55.2% |
| ZOO | 11.8% | 9.4% | 10.6% | 22.1% | 11.7% | 14.3% |
| BMI-FGSM | 21.5% | 14.5% | 18.4% | 25.4% | 22.8% | 20.1% |

**Table 5**
Adversarial sample generation on ImageNet.

| Methods | Untargeted | | | Targeted | | |
|---|---|---|---|---|---|---|
| | Success | Avg. dist | Avg. time | Success | Avg. dist | Avg. time |
| C&W | 100.0% | – | 8.2 min | 98.6% | – | 12.0 min |
| MI-FGSM | 98.6% | 0.003 | <0.1 min | 98.6% | 0.005 | 0.1 min |
| ZOO | 98.6% | – | 18.1 min | 21.6% | – | 230.9 min |
| BMI-FGSM | 98.6% | 0.025 | 7.4 min | 93.2% | 0.046 | 19.5 min |

1000 iterations for both MNIST and CIFAR10. The optimization terminates if the loss does not decrease after each tenth of the iterations. For ZOO, we use a batch size of 128 i.e., evaluate 128 gradients and update 128 pixels at each step. The ZOO attack is based on C&W's framework, so we keep the setting consistent with C&W except running 3000 iterations for MNIST and 1000 iterations for CIFAR10.

For MI-FGSM, we set the maximum iterations $T = 40$ for MNIST and $T = 20$ for CIFAR10, and use a step size $\epsilon/T = 0.01$ per iteration. The upper bounds of perturbation distance are 0.4 and 0.05, respectively. The decay factor $\mu = 1$. We check the adversarial sample at the end of each iteration and return early if it successfully causes misclassification. For our BMI-FGSM, we set $DR = 1, CR = 0.9, N = 100, KR = 0.2$. In the case of MNIST, we set $\epsilon = 0.4, T = 40, G = 50$. In the case of CIFAR10, we set $\epsilon = 0.2, T = 20, G = 100$. Early return is activated.

#### 4.1.4. Results

We report the success rate, average perturbation distance, and time for the four methods on MNIST and CIFAR10 in Table 1 and Table 2, respectively. From column 2, we can observe that all methods achieve 100% success rates in the untargeted setting. From column 5, The success rate of our approach in the targeted setting is over 95%, which is higher than the black-box ZOO. In terms of the perturbation distance in columns 3 and 6, we ignore the results of C&W and ZOO because they optimize $L_2$-norm distance while MI-FGSM and BMI-FGSM use $L_\infty$-norm. BMI-FGSM shows very close perturbation distance to the MI-FGSM, that is, our black-box method generates samples with similar quality as the white-box method. Columns 4 and column 7 illustrate the time cost. Our black-box approach can generates a valid adversarial sample within about 20 s for untargeted attacks and 30 s for targeted attacks, which is more efficient than the black-box ZOO. Note that our approach is an iterative gradient-based method, which is originally designed for fast sample generation.

Overall, our approach features high success rate and high efficiency. Fig. 6 visualizes some generated adversarial samples. We can observe that BMI-FGSM is able to generate images that show a similar visual impression as original images. These malicious samples with imperceptible distortion to human eyes confuse DNNs with high accuracy.

### 4.2. RQ2: Transferability

Transferability indicates the cross-model usability of adversarial samples. We evaluate the transferability on MNIST and CIFAR10. We define the transfer rate to be the percentage of transferable adversarial samples generated in Section 4.1, i.e., samples that can trigger the misbehavior of another model.

In the case of MNIST, we train a simple FC network (3x fully connected layers) and a LeNet5 (LeCun et al., 1998) as targets. In the case of CIFAR10, we train an All Convolution Network (AllConv) (Springenberg et al., 2014), a Network in Network (NiN) (Lin et al., 2013) and a VGG16 (Simonyan and Zisserman, 2015) network as targets. All models above are kept as similar as possible to their original framework with minor modifications at the softmax layer to fit the output.

#### 4.2.1. Results

We report the transferability in Tables 3 and 4. Values in the table denote the transfer success rate, represent the generalization of the adversarial samples generated by different methods. We can observe that adversarial samples created by gradient-based methods (MI-FGSM, BMI-FGSM) have a higher transfer rate than those by optimization-based methods (C&W, ZOO). The optimization-based methods must tolerate larger perturbation distance in order to obtain higher transferability. Besides, our proposed BMI-FGSM takes advantage of momentum, which is introduced for balancing success rate and transferability, achieving a comparable score to white-box MI-FGSM. The transferability of our approach is higher than the black-box ZOO method on both MNIST and CIFAR10.

### 4.3. RQ3: Strategies on large dataset

Modern image classification applications may have a larger dataset and a complex model. Attacks in such settings are challenging and expensive due to the large input space. We evaluate the performance and transferability of four methods on the large ImageNet dataset. Then, we study the efficacy of our proposed two mechanisms, double step size and candidate reuse, by generating an image with a hard limit of iterations.

We randomly choose 100 images from ImageNet validation set (Krizhevsky et al., 2012) and apply one untargeted and one targeted generation for each image. Thus there are 100 untargeted samples and 100 targeted samples in total. We use VGG16 (Simonyan and Zisserman, 2015) as the target model, InceptionV3 (Szegedy et al., 2016) and ResNet101 (He et al., 2016) as the transfer models. We adapt the CIFAR10 parameter settings in Section 4.1. For untargeted setting, an adversarial sample is valid only when its ground-truth label not in the top-5 prediction.

#### 4.3.1. Results

Table 5 demonstrates the performance and Table 6 shows the transferability. Our proposed BMI-FGSM achieves a success rate of 98.6% in the untargeted setting, 93.2% in targeted setting.

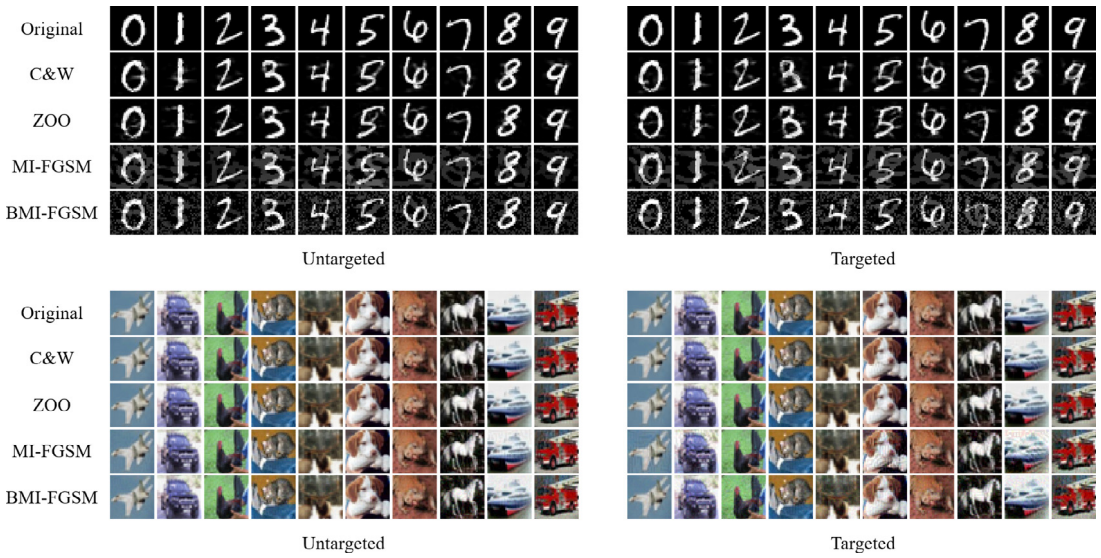



**Fig. 6.** Visualization of the adversarial samples on MNIST and CIFAR10.

**Table 6**
Transfer rate on ImageNet.

| Methods | Untargeted | | Targeted | |
|---|---|---|---|---|
| | InceptionV3 | ResNet101 | InceptionV3 | ResNet101 |
| C&W | 16.2% | 14.9% | 15.1% | 13.7% |
| MI-FGSM | 11.0% | 12.3% | 11.0% | 9.6% |
| ZOO | 37.0% | 37.0% | 25.0% | 31.3% |
| BMI-FGSM | 28.8% | 30.1% | 29.0% | 33.3% |

**Table 7**
Different strategies comparison on ImageNet.

| Strategy | Success rate | Avg. dist | First valid |
|---|---|---|---|
| BMI-FGSM | 98.6% | 0.025 | 861 |
| No double step size | 70.3% | 0.021 | 808 |
| No candidate reuse | 0% | – | – |
| Perturbed images as population | 63.5% | 0.020 | 797 |

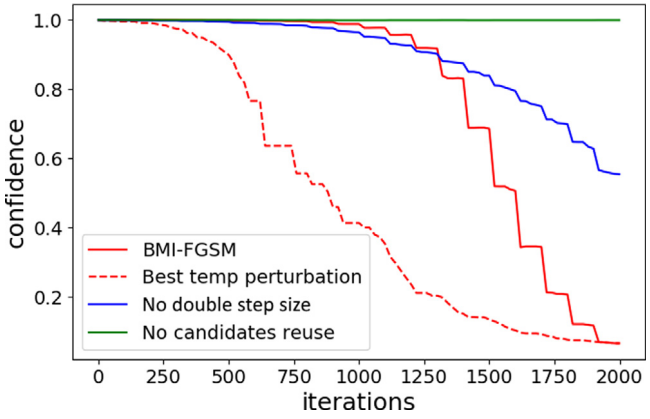



**Fig. 7.** Confidence of ground-truth versus iterations.

Compared with the white-box technique MI-FGSM, our approach has a higher perturbation distance due to large scale input images, and hence more gradients need to be approximated. Compared with the black-box ZOO, our approach significantly reduces time consumption and generates more transferable samples. We plot some of our generated samples in Fig. 1. All are hard to distinguish from the original samples.

The effect of different strategies is illustrated in Table 7. Observe that the average distance and the first valid iteration (the iteration where the first successful adversarial sample was found) have no evident distinction, while the success rates change noticeably — the success rate decreases by about 28% without double step size. The algorithm is almost unusable without candidate reuse. We also evaluate an alternative that uses the perturbed images as the population rather than gradient sign, while keeping everything else the same. Observe that the success rate decrease as well, suggesting incompatibility. This result indicates that new strategies are surely needed to apply this alternative.

To further investigate the advantages of double step size and candidate reuse, we set a hard $T \times G = 2000$ iterations and report the confidence of ground-truth label versus iterations in Fig. 7. The lower ground-truth label confidence, the higher attack ability of sample. The curve of no candidate reuse ends at about 0.99 shows the importance of momentum information. The curve of no double step size ends at about 0.55 because a single short perturbation distance runs the risk of being trapped in some local optima. In contrast, the dashed curve that indicates the best temporary perturbed images when approximate gradient signs drop quickly, and guide BMI-FGSM to achieve lower confidence of 0.05. With all the techniques applied, BMI-FGSM is able to generate visually undetectable adversarial samples that effectively suppress the probability of ground-truth label.

Adversarial samples represent a particular type of misbehavior of the DL system. To illustrate the ability of BMI-FGSM to test the robustness of DNN, we use random testing without coverage guidance and Tensorfuzz[6] with coverage guidance as baselines of DNN testing techniques. Table 8 reports the performance of different methods in generating untargeted adversarial samples. Observe that our approach outperforms random testing and Tensorfuzz, which means BMI-FGSM is more appropriate for robustness testing of adversarial attacks.

### 4.4. RQ4: Testing third-party API

In order to validate the reliability and applicability of our method to third-party applications, we test the Aliyun Image Recognition API,[7] a commercial computer vision toolkit powered

[6] https://github.com/brain-research/tensorfuzz
[7] https://help.aliyun.com/product/53258.html


**Fig. 8.** Aliyun Image Recognition API. Left: clean image. Right: adversarial image.

**Table 8**
Comparison of testing techniques on ImageNet.

| Strategy | Success rate | Avg. dist |
|---|---|---|
| Random | 0% | – |
| Tensorfuzz | 31.1% | 0.046 |
| BMI-FGSM | 98.6% | 0.025 |

by Alibaba Cloud. The API is a n-way classifier that can be queried and outputs label-score pairs for a given image, without any internal details. Our goal is to perform adversarial sample generation to trigger black-box API misbehavior. However, testing Aliyun API is more challenging than that on a common black-box model because of the following reasons: (i) The Aliyun API provides a free trial of 5000 queries. After that, it costs about $1.5 per 5000 queries. From the perspective of the attacker, a query-efficient adversarial generation algorithm is highly desirable. (ii) The number of classes is even larger, but we do not know how many and cannot enumerate all labels. (iii) The API only outputs scores for up to 5 top labels. The scores do not sum to one. They are neither probabilities nor logits.

In this experiment, we adapt the parameter settings in Section 4.3. We set the perturbation distance bound to be $\epsilon = 0.1$ and perform an untargeted generation with a budget of 5000 queries to the target API. We use API's top-1 prediction of the original image as the fitness. The algorithm early terminates when the top-1 prediction change.

#### 4.4.1. Results

Our approach achieves a 92% success rate against the Aliyun API on an ImageNet sample set of 25 images. Some adversarial samples against the image recognition API are given in Fig. 8. The left part is the original predictions, showing that the API correctly classifies clean images with high confidence. The right part is the adversarial predictions, showing adversarial samples mislabeled by the API. For example, the sign image with a 97% confidence score is misclassified as a restaurant, while the two images look very similar. Overall, we successfully trigger the Aliyun API misbehavior with perturbed images.

## 5. Conclusion

In this paper, we introduce Differential Evolution and develop a new type of black-box adversarial sample generation method called BMI-FGSM. We generate adversarial samples that are hard to detect and successfully attack DNNs on MNIST, CIFAR10, and ImageNet. We propose two mechanisms, double step size and candidate reuse, to be the essential part of our black-box method and conduct experiments to validate their efficacy. Experimental results show that our approach obtains success rate, perturbation distance, and transferability comparable to the state-of-the-art white-box techniques while reducing the time consumption of black-box method considerably. It even achieves similar performance to a white-box method on a large dataset. Finally, we craft adversarial samples for the Aliyun Image Recognition API and expose its robustness problem, demonstrating that our approach

is able to test real-world third-party systems from the perspective of the attacker.

Future work includes testing models with defense techniques or models in other areas. This work also opens up new opportunities on how to combine adversarial techniques with evolutionary algorithms like Differential Evolution.

## CRediT authorship contribution statement

**Junyu Lin:** Conceptualization, Methodology, Software, Validation, Writing - original draft. **Lei Xu:** Project administration, Writing - review & editing. **Yingqi Liu:** Supervision, Writing - review & editing. **Xiangyu Zhang:** Supervision, 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

We thank the anonymous reviewers for their constructive comments. This research was supported, in part by NSFC 61832009, NSF 1901242 and 1910300, ONR N000141410468 and N000141712947, IARPA TrojAI W911NF-19-S-0012, Sandia National Lab under award 1701331. Any opinions, findings, and conclusions in this paper are those of the authors only and do not necessarily reflect the views of our sponsors.

## References

Alzantot, M., Sharma, Y., Chakraborty, S., et al., 2019. GenAttack: Practical black-box attacks with gradient-free optimization. In: Proceedings of the Genetic and Evolutionary Computation Conference. ACM, pp. 1111–1119.

Bahdanau, D., Cho, K., Bengio, Y., 2014. Neural machine translation by jointly learning to align and translate. In: 2nd International Conference on Learning Representations. presentation.

Bojarski, M., Testa, D.D., Dworakowski, D., et al., 2016. End to end learning for self-driving cars. arXiv:1604.07316.

Carlini, N., Wagner, D., 2017. Towards evaluating the robustness of neural networks. In: 38th IEEE Symposium on Security and Privacy (SP). IEEE, pp. 39–57.

Chen, P.Y., Zhang, H., Sharma, Y., Yi, J.F., Hsieh, C.J., 2017. 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. AISec '17, ACM, pp. 15–26.

Cheng, M., Le, T., Chen, P.Y., et al., 2019. Query-efficient hard-label black-box attack: An optimization-based approach. In: 7th International Conference on Learning Representations. Poster.

Das, S., Abraham, A., Chakraborty, U., et al., 2009. Differential evolution using a neighborhood-based mutation operator. IEEE Trans. Evol. Comput. 13 (3), 526–553.

Das, S., Suganthan, P.N., 2011. Differential evolution: A survey of the state-of-the-art. IEEE Trans. Evol. Comput. 15 (1), 4–31.

Dong, Y.P., Liao, F.Z., Pang, T.Y., et al., 2018. Boosting adversarial attacks with momentum. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. IEEE, pp. 9185–9193.

Evtimov, I., Eykholt, K., Fernandes, E., et al., 2018. Robust physical-world attacks on deep learning models. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. IEEE, pp. 1625–1634.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., et al., 2014. Generative adversarial nets. Adv. Neural Inf. Process. Syst. 27 (2), 2672–2680.

Goodfellow, I., Shlens, J., Szegedy, C., 2015. Explaining and harnessing adversarial examples. In: 3rd International Conference on Learning Representations. presentation.

He, K.M., Zhang, X.Y., Ren, S.Q., et al., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, pp. 770–778.

Ilyas, A., Engstrom, L., Athalye, A., et al., 2018. Black-box adversarial attacks with limited queries and information. In: Proceedings of the 35th International Conference on Machine Learning. IMLS, pp. 2142–2151.

Ilyas, A., Engstrom, L., Madry, A., 2019. Prior convictions: Black-box adversarial attacks with bandits and priors. In: 7th International Conference on Learning Representations. Poster.

Jere, M., Rossi, L., Hitaj, B., et al., 2019. Scratch that! an evolution-based adversarial attack against neural networks. arXiv:1912.02316.

Krizhevsky, A., Hinton, G., 2009. Learning Multiple Layers of Features from Tiny Images. Technical Report.

Krizhevsky, A., Sutskever, I., Hinton, G., 2012. Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. 25 (1), 1097–1105.

Kurakin, A., Goodfellow, I., Bengio, S., 2016. Adversarial examples in the physical world. arXiv:1607.02533.

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521 (7553), 436–444.

LeCun, Y., Bottou, L., Bengio, Y., et al., 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86 (1), 2278–2324.

Lin, M., Chen, Q., Yan, S.C., 2013. Network in network. arXiv:1312.4400.

Litjens, G., Kooi, T., Bejnordi, B.E., et al., 2017. A survey on deep learning in medical image analysis. Med. Image Anal. 42 (1), 60–88.

Ma, L., Juefei-Xu, F., Xue, M.H., et al., 2019. DeepCT: Tomographic combinatorial testing for deep learning systems. In: 26th IEEE International Conference on Software Analysis, Evolution and Reengineering. SANER 2019, IEEE, pp. 614–618.

Ma, L., Juefei-Xu, F., Zhang, F.Y., et al., 2018a. Deepgauge: Multi-granularity testing criteria for deep learning systems. In: Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering. ASE 2018, ACM, pp. 120–131.

Ma, L., Zhang, F.Y., Sun, J.Y., et al., 2018b. Deepmutation: Mutation testing of deep learning systems. In: 29th IEEE International Symposium on Software Reliability Engineering. ISSRE 2018, IEEE, pp. 100–111.

Madry, A., Makelov, A., Schmidt, L., et al., 2017. Towards deep learning models resistant to adversarial attacks. arXiv:1706.06083.

Moosavi-Dezfooli, S.M., Fawzi, A., Fawzi, O., et al., 2017. Analysis of universal adversarial perturbations. arXiv:1705.09554.

Narodytska, N., Kasiviswanathan, S.P., 2017. Simple black-box adversarial attacks on deep neural networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops. CVPRW, IEEE, pp. 1310–1318.

Odena, A., Goodfellow, I., 2018. Tensorfuzz: Debugging neural networks with coverage-guided fuzzing. arXiv:1807.10875.

Papernot, N., McDaniel, P., Goodfellow, I., et al., 2017. Practical black-box attacks against machine learning. In: Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security. ASIA CCS '17, ACM, pp. 506–519.

Pei, K.X., Cao, Y.Z., Yang, J.F., et al., 2017. Deepxplore: Automated whitebox testing of deep learning systems. In: Proceedings of the 26th Symposium on Operating Systems Principles. SOSP '17, ACM, pp. 1–18.

Perez, L., Wang, J., 2017. The effectiveness of data augmentation in image classification using deep learning. arXiv:1712.04621.

Polyak, B., 1964. Some methods of speeding up the convergence of iteration methods. USSR Comput. Math. Math. Phys. 4 (5), 1–17.

Sak, H., Senior, A., Rao, K., et al., 2015. Fast and accurate recurrent neural network acoustic models for speech recognition. In: 16th Annual Conference of the International Speech Communication Association. pp. 1468–1472.

Sharif, M., Bhagavatula, S., Bauer, L., et al., 2019. A general framework for adversarial examples with objectives. ACM Trans. Priv. Secur. 22 (6), 16.

Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations. presentation.

Springenberg, J.T., Dosovitskiy, A., Brox, T., et al., 2014. Striving for simplicity: The all convolutional net. arXiv:1412.6806.

Storn, R., Price, K., 1997. Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. 11 (4), 341–359.

Su, J.W., Vargas, D.V., Sakurai, K., 2019. One pixel attack for fooling deep neural networks. IEEE Trans. Evol. Comput. 23 (1), 1.

Sun, Y.C., Huang, X.W., Kroening, D., 2018. Testing deep neural networks. arXiv:1803.04792.

Suya, F., Chi, J., Evans, D., et al., 2019. Hybrid batch attacks: Finding black-box adversarial examples with limited queries. arXiv:1908.07000.

Szegedy, C., Vanhoucke, V., Ioffe, S., et al., 2016. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, pp. 2818–2826.

Szegedy, C., Zaremba, W., Sutskever, I., et al., 2013. Intriguing properties of neural networks. arXiv:1312.6199.

Tian, Y.C., Pei, K.X., Jana, S., et al., 2018. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In: Proceedings of the 40th International Conference on Software Engineering. ICSE 2018, AC<, pp. 303–314.

Tramer, F., Kurakin, A., Papernot, N., et al., 2017. Ensemble adversarial training: Attacks and defenses. arXiv:1705.07204.

Wang, T.C., Liu, M.Y., Zhu, J.Y., et al., 2018. High-resolution image synthesis and semantic manipulation with conditional GANs. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. IEEE, pp. 8798–8807.

Zhang, J.Q., Sanderson, A.C., 2009. JADE: Adaptive differential evolution with optional external archive. IEEE Trans. Evol. Comput. 13 (5), 945–958.

Zhong, Z., Zheng, L., Kang, G.L., et al., 2017. Random erasing data augmentation. arXiv:1708.04896.

**Junyu Lin** is a postgraduate student at the Nanjing University, Department of Computer Science and Technology. His research interest includes AI in software engineering, deep learning security and testing.

**Lei Xu** is an associate professor at the Nanjing University, Department of Computer Science and Technology. Her research interest includes Web software analysis and testing, big code mining.

**Yingqi Liu** is a research assistant at the Purdue University, Department of Computer Science. His research interest includes adversarial machine learning, AI-assisted computer reasoning and other AI related projects.

**Xiangyu Zhang** is a profess or at the Purdue University, Department of Computer Science. His research interest includes program analysis, security, deep learning security, dependability and interpretability.