COMPUTERS & SECURITY 85 (2019) 89–106

Available online at www.sciencedirect.com

ScienceDirect

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

Computers & Security

POBA-GA: Perturbation optimized black-box adversarial attacks via genetic algorithm

Jinyin Chen, Mengmeng Su, Shijing Shen, Hui Xiong, Haibin Zheng*

Zhejiang University of Technology, Hangzhou 310023, China

ARTICLE INFO

Article history:
Received 3 December 2018
Revised 25 March 2019
Accepted 20 April 2019
Available online 30 April 2019

Keywords:
Deep learning
Adversarial attack
Perturbation optimization
Genetic algorithm
Defense

ABSTRACT

Most deep learning models are easily vulnerable to adversarial attacks. Various adversarial attacks are designed to evaluate the robustness of models and develop defense model. Currently, adversarial attacks are brought up to attack their own target model with their own evaluation metrics. And most of the black-box adversarial attack algorithms cannot achieve the expected success rate compared with white-box attacks. In this paper, comprehensive evaluation metrics are brought up for different adversarial attack methods. A novel perturbation optimized black-box adversarial attack based on genetic algorithm (POBA-GA) is proposed for achieving white-box comparable attack performances. Approximate optimal adversarial examples are evolved through evolutionary operations including initialization, selection, crossover and mutation. Fitness function is specifically designed to evaluate the example individual in both aspects of attack ability and perturbation control. Population diversity strategy is brought up in evolutionary process to promise the approximate optimal perturbations obtained. Comprehensive experiments are carried out to testify POBA-GA's performances. Both simulation and application results prove that our method is better than current state-of-art black-box attack methods in aspects of attack capability and perturbation control.

1. Introduction

Deep learning is the core of current machine learning and artificial intelligence (Goodfellow et al., 2016). Since it has powerful learning, feature extraction and modeling capabilities, it has been widely applied to challenging areas, such as social networks (Deng et al., 2017a), medical image analysis (Havaei et al., 2017; Kooi et al., 2017; Litjens et al., 2017) and selective classification (De Cnudde et al., 2018; Geifman and El-Yaniv, 2017). And in the area of computer vision, deep learning has become the main force for various applications such as self-driving cars (Stilgoe, 2018), image processing (Szegedy and Vanhoucke, 2017; Yosinski et al., 2014), target-driven visual navigation (Zhu et al., 2017) and scene recognition (Yuan et al., 2015).

The latest research shows that although deep learning can extract complete image features and forecast or classify it perfectly, it can be easily fooled by adversarial examples into erroneous prediction outputs by adding small perturbations on the original image (Bai et al., 2017; Liu et al., 2018; Metzen et al., 2017; Ramanathan et al., 2017; Yin et al., 2018). The adversarial attack was first proposed by Szegedy et al. (2013) and attracted more attentions, becoming a new hot topic. As long as deep models are threatened by adversarial attack, lots of deep model based applications are unable to extend. For instance, payment based on facial recognition cannot be trusted since an adversarial glass can help one imitate another

* Corresponding author.
E-mail addresses: chenjinyin@zjut.edu.cn (J. Chen), 2111703406@zjut.edu.cn (M. Su), 201407760128@zjut.edu.cn (S. Shen), 201403080329@zjut.edu.cn (H. Xiong), 2111703122@zjut.edu.cn (H. Zheng).
https://doi.org/10.1016/j.cose.2019.04.014
0167-4048/

person easily (Sharif et al., 2016), and auto-drive based on image recognition is quite dangerous if the road sign is a carefully designed adversarial example (Eykholt et al., 2017). In general, we can assess the robustness of the classifier by simulating the attacker's efforts to evade the classifier (Xu et al., 2016), so it is also possible to increase the robustness of the model by defending adversarial attacks. Lots of adversarial attack methods are brought up for understanding the attack and improving model's defensibility, such as Jacobian-based Saliency Map Attack (JSMA) (Papernot et al., 2016), DeepFool (Moosavi-Dezfooli et al., 2016), One-pixel Attack (Su et al., 2017), Limited Queries and Information Attack (Ilyas et al., 2018).

Adversarial attacks can be roughly divided into three categories: gradient-based, score-based and transfer-based attacks (Brendel et al., 2017). Gradient-based and score-based attacks are often denoted as white-box and oracle attacks respectively. Most existing gradient-based white-box attacks rely on detailed model information, such as the Basic Iterative Model (BIM) (Kurakin et al., 2016), Houdini (Cisse et al., 2017), DeepFool (Moosavi-Dezfooli et al., 2016), which are taking advantage of gradient loss. There are also black-box attacks, which use transfer across models (Papernot et al., 2017) or require access to all training data sets (Moosavidezfooli et al., 2017). Since most real-world systems do not publish the network's weight, architecture or training data sets, so white-box attacks and equivalent model attacks are not easy to implement in practice. There is still a flaw that if the attacker is capable of black-box attack without the internal configuration of the target model. That is the reason we develop a completely internal model information independent adversarial attack. It cannot only fill the gap of evolutionary based adversarial attacks with minimal perturbation, but also help evaluating and improving the defensibility of current state-of-art deep models.

Excellent adversarial examples generally have the following two characteristics. First, it is very similar to the original image. There are slight perturbations barely distinguishable by the naked eye. Second, the adversarial example can lead the target model to a high confidence misclassification. Black-box attacks are conducted without any internal model information (structure and parameter), but based on the information such as most probable class label or confidence. The target of black-box attack is to reduce the confidence of true label with limited perturbation. Therefore, in most cases black-box attack can be modeled as an optimization problem. Genetic algorithm is widely applied to various applications as a typical optimization tool, such as energy optimization (Talha et al., 2017), distribution network optimization (Syahputra, 2017), ontology alignments optimization (Gil et al., 2008) and web crawler (Goyal et al., 2016), and all of them achieve good optimization performance. In this paper, a novel adversarial perturbation optimization attack based on genetic algorithm is proposed to implement black-box attack. Fitness function is constructed on the basis of classification confidence and perturbation size, and genetic operations are designed to promise approximate optimal adversarial example.

The current adversarial attacks generally use L_0, L_2 and L_∞ as evaluation metrics for the perturbations. However, most attack algorithms use different perturbation evaluation metrics according to the characteristics of the algorithm (Akhtar and Mian, 2018). For example, L-BFGS (Szegedy et al., 2013) and FGSM (Goodfellow et al., 2014) use L_∞, JSMA (Papernot et al., 2016) and One-pixel (Su et al., 2017) use L_0, DeepFool (Moosavi-Dezfooli et al., 2016) uses L_2. Therefore, we can only evaluate the quality of the algorithm from different perspectives, and cannot judge which perturbation and algorithm are better. Based on the sensitivity of the human visual system to perturbation, a new perturbation assessment method is put forward which can comprehensively evaluate the perturbation, making the perturbation more similar to the actual sensitivity.

The main contribution of our work can be concludes as:

- *Perturbation optimization*. POBA-GA is a novel perturbation optimization method to generate black-box adversarial examples, which is capable of high successful attack rate against different deep learning models with controllable perturbations.
- *White-box comparable black-box attack*. POBA-GA can optimize the perturbations based on the confidence of the black-box output. In most cases, it can achieve high attack success rate comparable with white-box attack methods.
- *New perturbations evaluation metric*. A novel perturbation evaluation metrics is put forward. It comprehensively evaluates the perturbation and maps its size to different dimensions, which makes the evaluation result more realistic reflection of the perturbation.
- *Improve defense capability through adversarial training*. Adversarial examples generated by POBA-GA are adopted to train deep model to improve its defense capacity. Experiments prove that adversarial training based on examples from POBA-GA could defend model better than from other attack methods.

The rest of paper is organized as follows. The related works are discussed in Section 2. The main methods and strategies are introduced in Section 3. Experiments and conclusions are shown in Sections 5 and 6.

2. Related works

In this section, we introduce classic adversarial attack methods, genetic algorithms and perturbation evaluation metrics.

2.1. Attacks methods

Since Szegedy et al. (2013) proposed the concept of adversarial attack against deep model, a large number of adversarial attacks are put forward (Carlini and Wagner, 2017; Goodfellow et al., 2014; Hayes and Danezis, 2017; Moosavi-Dezfooli et al., 2017; 2016). Some researchers have launched attacks in applications, such as speech recognition systems (Carlini et al., 2016), malware detectors (Hu and Tan, 2017; Xu et al., 2016), and face recognition systems (Sharif et al., 2017). For example, Sun et al. (2018) use adversarial attacks against deep predictive models to identify susceptible locations in medical records. Sethi and Kantardzic (2018) present an adversary's view point of a classification based system. Generally, attacks are classified into white-box attacks and black-box attacks based on whether they know the internal structure of the target model. It should be noted that, because there are too many adversarial attack literatures, this paper mainly introduces some algorithms of computer vision.

2.1.1. White-box adversarial attack models

A white-box attack is a method of attacking a target model while it knows its internal structure. White-box attacks are the most common attack method, which can also be used to attack the equivalent model to achieve black-box attacks. At present, researchers have proposed a large number of white-box attacks (Carlini and Wagner, 2017; Goodfellow et al., 2014; Kurakin et al., 2016; Moosavi-Dezfooli et al., 2016; Papernot et al., 2016). For example, Bose and Aarabi (2018) proposed adversarial attacks on face detectors using neural net based constrained optimization. Yu et al. (2018) proposed a fast adversarial attack example generation framework based on adversarial saliency prediction. Chen et al. (2018) proposed robust physical adversarial attack on faster R-CNN object detector. Ramanathan et al. (2017) proposed adversarial attacks on computer vision algorithms using natural perturbations.

The perturbation optimization algorithm in this paper takes the adversarial examples generated by the white-box attack as a partial initial solution and realizes the perturbation optimization through the genetic algorithm. To ensure the diversity of the initial solution, we chose seven different attack methods to generate. The reasons for choosing this methods are described in detail in Section 3. The attack method is described in detail below.

Fast Gradient Sign Model (FGSM) (Goodfellow et al., 2014): FGSM is one of the simplest and most widely used non-target counter attacks. FGSM uses backward propagation gradient from the target DNN to generate adversarial examples. Perturbation is evaluated by $\rho = \epsilon sign(\nabla J(\theta, I_c, l))$[1], where ∇J denotes the gradient of the original image I_c around the model parameters θ. $sign(.)$ denotes the sign function, and ϵ is a small scalar value that limits the perturbation evaluation.

Basic Iterative Model (BIM) (Kurakin et al., 2016): BIM, equivalent to Projected Gradient Descent, is a standard convex optimization method. It is an extension of the single-step method, which takes multiple small step iterations while adjusting the direction after each step. After a sufficient number of iterations, BIM can successfully generate an adversarial example classified into the target label.

Jacobian-based Saliency Map Attack (JSMA) (Papernot et al., 2016): JSMA describes the input-output relationship of the target DNN by constructing a Jacobian-based saliency map. It iteratively modifies the most important pixels based on saliency mapping during iteration to fool the network. At each iteration, JSMA recalculates the saliency map and uses the DNN derivative of the input image as a modify index of the adversarial attack. This greedy search process is repeated until the number of changing pixels reaches the threshold or the deception is successful.

DeepFool (Moosavi-Dezfooli et al., 2016): DeepFool is a simple but very effective non-targeted attack. In each iteration, it calculates the minimum distance $d(y_1, y_0)$ required for each label $y_1 \neq y_0$ to reach the class boundary by approximating the model label with a linear label. y_1 represents the label of the adversarial example classified by deep model, and y_0 represents the true label of the adversarial example. Then make the appropriate steps in the direction of the nearest class. The image perturbation for each iteration are accumulated and the final perturbation is calculated once the output criteria changes.

Carlini and Wagner Attacks (C&W) (Carlini and Wagner, 2017): Carlini and Wagner Attacks is one of the strongest attacks. Its attack essence is a kind of refined iterative gradient attack that uses the Adam optimizer. It uses the internal configuration of the target DNN to guide the attack, and uses L_2 specification to quantify the difference between the hostile and original image.

Gaussian Blur (Chen and Ma, 2009): Gaussian Blur is a kind of linear smoothing filter, which is suitable for eliminating Gaussian noise and is widely used in the noise reduction process of image processing. Generally speaking, Gaussian filtering is the process of weighted averaging of the entire image. The value of each pixel is obtained by weighted averaging of its own and other pixel values in the neighborhood. However, the Gaussian filter also causes the image to lose certain eigenvalues, making the CNN classification error.

Salt and Pepper Noise (Varatharajan et al., 2017): Salt and pepper noise, also known as impulsive noise, randomly changes some pixel values. Salt and pepper noise appears on a binary image is to make some pixels white or black, black is pepper, and white is salt. And these salt and pepper noises have a certain probability of misclassifying the CNN label.

2.1.2. Black-box adversarial attack models

The black-box can only access the input and output of the target model, but cannot access the internal configuration of the target model. In case of image label trained by CNN, we take the image as input and produces a confidence score for each label as an output (Chen et al., 2017). In practical applications, most of the target models are black-box models, so black-box attacks have important research significance and have been studied by many researchers (Dong et al., 2018; Milton, 2018; Smith and Gal, 2018). For example, Milton (2018) proposed the evaluation of momentum diverse input iterative fast gradient sign method (M-DI2-FGSM) to attack the black-box facial recognition system. Dong et al. (2018) propose a broad class of momentum-based iterative algorithms to boost adversarial attacks. Brendel et al. (2017) introduce the Boundary Attack, open new avenues to study the robustness of machine learning models and raise new questions regarding the safety of deployed machine learning systems. Ilyas et al. (2018) proposed the black-box adversarial attacks with limited information query. In the current black-box attack, ZOO and Boundary are the state-of-art attack methods, which aim to improve the attack rate.

2.2. Genetic algorithm

The existing attack method can be regarded as the solution of the optimization problem to some extent. For example, ZOO uses a zeroth order method to optimize black-box attack (Chen et al., 2017). UPSET and ANGRI use the so-called UPSET network to optimize black-box attack (Sarkar et al., 2017).

Table 1 – Attributes of the computer vision adversarial attack methods.

Model	Black/white-box	Targeted/non-targeted	Specific/universal	Perturbation norm	Learning
L-BFGS (Szegedy et al., 2013)	White-box	Targeted	Image specific	L_∞	One shot
FGSM (Goodfellow et al., 2014)	White-box	Targeted	Image specific	L_∞	One shot
BIM & ILCM (Kurakin et al., 2016)	White-box	Non-targeted	Image specific	L_∞	Iterative
JSMA (Papernot et al., 2016)	White-box	Targeted	Image specific	L_0	Iterative
One-pixel (Su et al., 2017)	Black-box	Non-targeted	Image specific	L_0	Iterative
C&W (Carlini and Wagner, 2017)	White-box	Targeted	Image specific	L_0, L_2, L_∞	Iterative
DeepFool (Moosavi-Dezfooli et al., 2016)	White-box	Non-targeted	Image specific	L_2, L_∞	Iterative
Uni. perturbations (Moosavi-Dezfooli et al., 2017)	White-box	Non-targeted	Universal	L_2, L_∞	Iterative
UPSET (Sarkar et al., 2017)	Black-box	Targeted	Universal	L_∞	Iterative
ANGRI (Sarkar et al., 2017)	Black-box	Targeted	Image specific	L_∞	Iterative
ZOO (Chen et al., 2017)	Black-box	Non-targeted	Image specific	L_2	Iterative
Boundary (Brendel et al., 2017)	Black-box	Targeted	Image specific	L_2	Iterative
Limited (Ilyas et al., 2018)	Black-box	Non-targeted	Image specific	L_∞	Iterative
MI-FGSM (Dong et al., 2018)	Both	Both	Image specific	L_2, L_∞	Iterative
AutoZOOM (Tu et al., 2018)	Black-box	Both	Image specific	L_2	Iterative

Houdini optimized for mean per-pixel or per-class accuracy instead of mIoU for better experimental results (Cisse et al., 2017). Genetic algorithm has been widely used in various optimization problems and have achieved good results, for example, structures locations optimization (Kalajac et al.), distribution network optimization (Alencar et al., 2018) and relay related optimization (Souza et al., 2018). Therefore, we apply genetic algorithms to perturbation optimization to generate high-quality adversarial examples. Solutions are initialized as population. Fitness of each individual in population is calculated. Selection, crossover and mutation operators are carried out by certain probability to update whole population until termination condition is met. Approximate optimal solution will be found for the problem.

2.3. *Perturbation evaluation metrics*

Current adversarial attacks use L_0, L_2 and L_∞ to evaluate the size of the perturbation. L_p distance metric is used as a measure of similarity (Hayes and Danezis, 2017), and it is used to evaluate the size of perturbation. The L_0 distance measurement indicates the number of pixels changed, and the L_2 represents the Euclidean distance between the two examples, and the L_∞ represents the maximum perturbation between two pixels. Table 1 list the perturbation metrics and other attributes of the current adversarial attack methods in computer vision. Due to the recent research on adversarial attacks is very popular, we are unable to list all the literature, so the table only lists the popular ones or representative of the popular direction of computer vision adversarial attack methods.

3. Model

3.1. *Problem definition*

For a given example S, Deep Neural Model (DNN) is applied to classify S with an output label. Attack Method *AM* is adopted to generate perturbation *A* to add to S. Directly or optimally generate an adversarial example *AS* to attack target model *TM*, so the target model outputs an error label. In the field of image recognition, S represents the original image. The adversarial example *AS* is generated and used to attack method *AM*, which is added by negligible perturbation *A*.

3.1.1. *DEFINITION 1 (DNN based image label)*

Deep Neural Network (DNN) is trained by a large number of labeled images. For a given image S, DNN can output label y_1, represented as $TM(\Theta, S) = y_1$, where Θ represents parameters, y_1 is the output label of the highest confidence.

3.1.2. *Definition 2 (adversarial attack)*

Given a DNN for S, whose response output is $TM(\Theta, S) = y_0$, Attack Method *AM* generates an adversarial image *AS* to make $TM(\Theta, AS) = y_1$, and $y_0 \neq y_1$, where S and *AS* are almost indistinguishable.

Adversarial attack is triggered by adversarial examples. Most adversarial examples are generated by adding perturbations into the original image. The perturbation quality will decide the attack capacity. An effective black-attack can be considered as generating a high quality perturbation without knowing the internal structure of the *TM*, causing the *TM* to output an error class, which is illustrated in Fig. 1. And how to generated a black-box adversarial example can be assumed as an optimization problem.

3.2. *Framework*

We propose a novel Perturbation Optimization black-box Attack based on Genetic Algorithm (POBA-GA), which takes a variety of different random perturbations as the initial examples, $\phi(.)$ is the fitness function to optimize the perturbation to obtain approximate optimal adversarial example AS_{opt}. The block diagram of POBA-GA is illustrated in Fig. 2. And the symbols used in the paper are listed in Table 2.

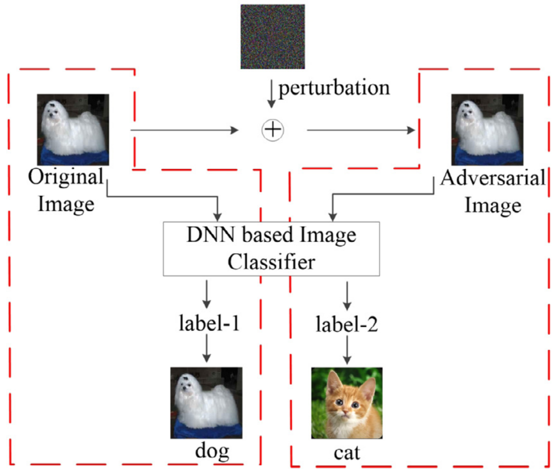


Fig. 1 – A illustration of black-box attack. Black-box attacks can be viewed as choosing the best countermeasure example generated from a white-box attack or generating a better confrontation example than the current white-box attack.

Table 2 – The symbols used in the paper.

S	The original example
AS_{opt}	The approximate optimal adversarial example of POBA-GA
AS^t	The collection of *t*th iteration adversarial examples (when $t = 0$, it represents the initial adversarial example)
AS_i^t	The *t*th iteration of the *i*th adversarial example
$AS_{i(ab)}^t$	The pixels of *a*th row and *b*th column of AS_i^t
A^t	A collection of perturbation of the *t*th iteration
A_i^t	The *t*th iteration of the *i*th perturbation
TM	The target method
L_0, L_2, L_∞	Zero/two/infinite norm
$\phi(AS_i^t)$	The fitness function of AS_i^t
$P(AS_i^t)$	The attack performance of AS_i^t in $\phi(.)$
$Z(A_i^t)$	The perturbation evaluation of A_i^t in $\phi(.)$
α	The perturbation ratio parameter in $\phi(.)$
$f(AS_i^t)$	The selected probability of AS_i^t
$fr(AS_i^t)$	The cumulative probability of AS_i^t
$p(y\|AS_i^t)$	The confidence of AS_i^t labeled as *y*
y_0	The true label of the original example S
y_1, y_2	The label for S with first and second confidence
y_{tar}	The preset label for target attack
B, C	The two-dimensional matrix in crossover/mutation
ASR	The attack success rate
P_c, P_m	The crossover/mutation probability

Fig. 2 demonstrates how a high quality adversarial example is generated through genetic algorithm. First, we generate different perturbations based on different noise point pixel thresholds, number of noise points, and noise point size. Then, add the perturbation into the original example S to generate the responding initial adversarial example $AS^{t=0}$, where $t = 0$ represents the first generation of population. Second, we define a fitness function $\phi(AS_i^t)$ to evaluate the *t*th iteration example $AS_i^t \in AS^t$, $i \in [0, n]$. Third, we judge whether the termination condition has been reached. Fourth, we apply typical operators in genetic algorithm including selection, crossover

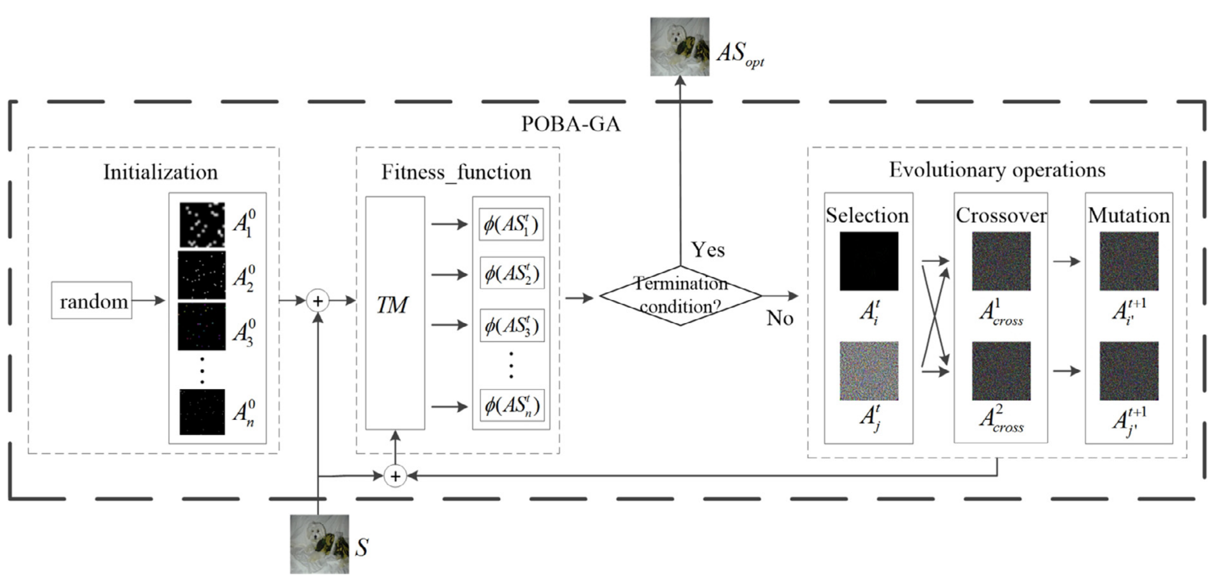


Fig. 2 – Algorithm block diagram of POBA-GA. Black-box attack is translated into the problem of finding the best perturbation. POBA-GA is established in stages: (1) Initialization. Generating a variety of different types of random perturbations. (2) Calculate fitness. Inputting the adversarial examples into the *TM* to obtain their classification result. Then, the fitness is calculated based on the classification result and the perturbation size. (3) Determine if the termination condition is reached, go to the next step or output the best adversarial example. (4) Evolutionary operations. Use the roulette selection method to select the perturbations corresponding to the adversarial examples, and then obtain the next generation perturbation by crossing and mutating.

and mutation to evolve new generation for perturbation optimization.

3.3. Initialization

Initialization is responsible for initial solution generation at the beginning. The quality of the initial solution directly affects the iterations. If the initial solution is similar to the approximate optimal solution, the algorithm will convergence quickly. Otherwise, the algorithm need more iterations to the convergence to approximate global optimal. In addition to consider the quality of each initial solution, the diversity of the initial solution is also crucial. It has been proved that diversity of initial population of genetic algorithm could promise approximate global optimal (Konak et al., 2006).

For a given example S, the optimization purpose is to evolve an approximate optimal adversarial example AS_{opt}. In random perturbation initialization, we use a search distribution of random Gaussian noise around the original image S (Ilyas et al., 2018), which is described as $AS^{t=0} = S + \delta$, where $\delta \sim \aleph(\mu, \sigma^2)$ and μ represent mathematical expectation, σ^2 represent variance. In order to increase the diversity of the initial perturbation, this paper generates different types of initial perturbation based on different variance, number of noise points, and noise point size.

3.4. Fitness function

Fitness function is defined to evaluate the quality of examples in GA. The proper fitness function can directly affect the convergence speed of genetic algorithm and whether it can find the optimal solution.

Excellent adversarial examples generally have the following two characteristics. First, it is very similar to the original image. There are only slight perturbation that are barely distinguishable by the naked eye. Second, the adversarial example may be misclassified by the target model with a highly confidence. Therefore, the fitness function designed should be related to the confidence and perturbation size, Eq. (1).

$$\phi(AS_i^t) = P(AS_i^t) - \frac{\alpha}{\max Z(A^0)} Z(A_i^t) \tag{1}$$

where $\phi(AS_i^t)$ represent the fitness function for example AS_i^t, $P(AS_i^t)$ represent the attack performance of example AS_i^t calculated by Eq. (2), $Z(A_i^t)$ represents the size of adversarial examples perturbation can calculated by Eq. (9), and $A_i^t + S = AS_i^t$. $Z(A_i^t)$ is a novel perturbation metric proposed in this paper, which will be explained and compared in Section 4. α represents the proportional coefficient and is used to adjust the proportion of attack performance and perturbation size. For example, when $\alpha = 0$, the fitness function only considers attack performance $P(AS_i^t)$. When α is larger, the optimization process pays more attention to the size of the perturbation. The optimized adversarial example may have only general attack performance, but the perturbation is very small.

$$P(AS_i^t) = \begin{cases} p(y_1|AS_i^t) - p(y_0|AS_i^t) & y_1 \neq y_0 \\ p(y_2|AS_i^t) - p(y_0|AS_i^t) & y_1 = y_0 \end{cases} \tag{2}$$

Table 3 – Perturbations evaluation metrics.

	AS_1^t	AS_2^t	AS_3^t	AS_4^t	AS_5^t	AS_6^t
$\phi(AS_i^t)$	0.9	0.45	0.6	0.79	0.95	0.85
$f(AS_i^t)$	0.20	0.10	0.13	0.17	0.21	0.19
$fr(AS_i^t)$	0.20	0.30	0.43	0.60	0.81	1

where y_0 represents the true label of the adversarial image, and y_1, y_2 represent the label with highest and second highest confidence of the *TM* output for AS_i^t, respectively. And y_1 also means the output label. $p(y|AS_i^t)$ represents the confidence that the adversarial example AS_i^t is labeled as y by the *TM*. When the output label is different from y_0 of AS_i^t, the attack is successful. The *TM* outputs the wrong label, and the attack performance is the confidence difference between the label y_1 and the true label y_0. The greater the attack capability, the stronger the attack capability. Otherwise, the output label is the same as the true label, which means the attack failed. Then we calculate the confidence difference between the highest label and the second highest label. The larger the difference, the more difficult it is to succeed.

In order to achieve the attack faster and reduce the initial number of queries to the DNN model, we will not consider the perturbation before the attack succeeds, i.e. let $\alpha = 0$. Only when the attack is successful, we will consider both attack performance $P(AS_i^t)$ and perturbation $Z(A_i^t)$. The updated fitness function is as follows.

$$\phi(AS_i^t) = \begin{cases} p(y_1|AS_i^t) - p(y_0|AS_i^t) - \frac{\alpha}{\max Z(A^{t_0})} Z(A_i^t) & y_1 \neq y_0 \\ p(y_2|AS_i^t) - p(y_0|AS_i^t) & y_1 = y_0 \end{cases} \tag{3}$$

where t_0 is the number of iterations when the initial attack succeeds, and $\frac{\alpha}{\max Z(A^{t_0})}$ is used to control the perturbation to a certain range.

3.5. Evolutionary operations

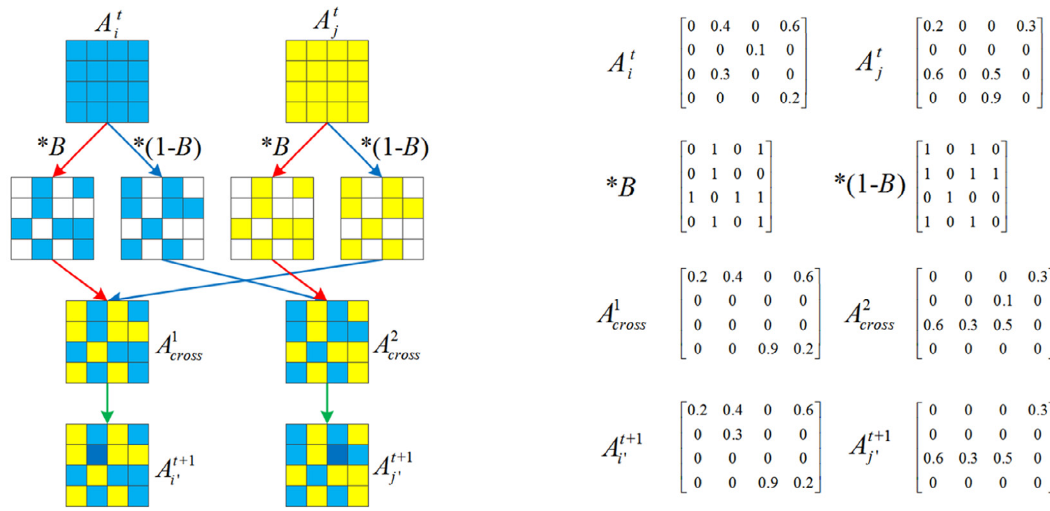
3.5.1. Selection operator

Selection operator is adopted to choose two parent examples to produce two children by crossover and mutation operators. Generally, quality examples are more likely to be selected. This paper uses roulette wheel selection (Lipowski and Lipowska, 2012). For a given example AS_i^t, its selection probability can be calculated according to Eq. (4). And the interval $(fr(AS_{i-1}^t), fr(AS_i^t)]$ for each model is calculated according to Eq. (5).

$$f(AS_i^t) = \frac{\phi(AS_i^t)}{\sum_{j=1}^{n} \phi(AS_j^t)} \tag{4}$$

$$fr(AS_i^k) = \sum_{j=1}^{i} f(AS_j^t) \tag{5}$$

For better understanding, we give a simple illustration of Roulette wheel selection. Let $n = 6$, Table 3 shows $\phi(AS_i^t)$, $f(AS_i^t)$ and $fr(AS_i^t)$ of the adversarial examples. The table



(a) The schematic of the crossover and variation. (b) The specific example of crossover and variation.

Fig. 3 – Crossing and mutating two parents to obtain new perturbations. The perturbation is divided into two complementary parts by the matrices B and $1 - B$, and then recombined with the complementary parts of the other perturbation to obtain two new perturbations. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

shows that the adversarial example AS_5^t has the largest fitness value, which means it is the best example among all. The probability of AS_5^t is selected as a parent should also be the largest, i.e. 0.21. The probability sum of all examples is 1.

3.5.2. *Crossover operator*

Crossover operator is executed to generate new examples from selected examples. This paper takes a uniform crossover, in which the genes at each locus of two paired individuals are exchanged with the same crossover probability, thus forming two new individuals. Fig. 3(a) shows the schematic of the crossover and variation of the example. To make the crossover process clearer, we split the step into a 2-step display. Unlike traditional chromosome crossover, our perturbation crossover can be considered as two-dimensional matrices crossover. A_i^t and A_j^t are the two parent perturbations selected, represented in Fig. 3(a) by blue and yellow matrices, respectively. Then, they are divided into two parts according to the same cross matrix. Where B is a two-dimensional matrix, and the RGB pixel value of the position is exchanged when crossing. Finally, cross perturbation to generate new perturbations A_{cross}^1 and A_{cross}^2 to pass to the next step. This process can be represented by Eqs. (6) and (7). You can find a specific example in the Fig. 3(b).

$$A_{cross}^1 = \begin{cases} A_i^t * B + A_j^t * (1 - B) & rand(0, 1) < P_c \\ A_i^t & otherwise \end{cases} \quad (6)$$

$$A_{cross}^2 = \begin{cases} A_i^t * (1 - B) + A_j^t * B & rand(0, 1) < P_c \\ A_j^t & otherwise \end{cases} \quad (7)$$

where B is a matrix of the same size of the image, each element of B is random number of 0 or 1. $rand(0, 1)$ means randomly generating a number between 0 and 1. P_c represents the probabilities of crossover. Fig. 11 in the appendix shows the effect of P_c on experimental results. The experimental results show that the larger the P_c is, the faster the fitness function converges. This is mainly because we use the parent-child hybrid method to generation updates and our mutation probability is very low. When $P_c < 1$, it is very likely to generate the same child example as the father, that is, generate duplicate examples, which increases the number of unnecessary queries and increases attack time and cost. Therefore, in order to reduce the cost of attack, this paper makes $P_c = 1$.

3.5.3. *Mutation operator*

Mutation operator indicates that some examples will be altered by a certain probability during the breeding process. In POBA-GA, we adopt multi-point mutation. According to mutation probability Pm, randomly select several pixels of AS_{cross}^q, where $q = \{1, 2\}$. The effect of variation is shown in Fig. 2, and the specific process is shown in Fig. 3. The process is defined as in the following equation.

$$A_{i'+q-1}^{t+1} = \begin{cases} A_{cross}^q * C & rand(0, 1) < P_m \\ A_{cross}^q & otherwise \end{cases} \quad (8)$$

where C is a matrix of the same size of the image. Except for a small number of elements in C, which are between 0 and 2, the remaining elements are all 1. P_m represents the probability of mutation. Through experiments we make $P_m = 0.001$ for the ImageNet data set and $P_m = 0.003$ for the MNIST and CIFAR-10 data set.

Algorithm 1 shows the pseudo code of POBA-GA, and Algorithm 2 shows the pseudo code of the function in POBA-GA.

Algorithm 1 POBA-GA.

Require: Input original example S, target model TM, parameter P_c, P_m, α, maximum number of iterations T, population size N.
Ensure: Approximate optimal adversarial example AS_{opt}

for $i = 1 \rightarrow N$ **do**
  $S_{i++}^{t=0} \leftarrow$ random initialization
end for
for $t = 1 \rightarrow T$ **do**
  $AS^t \leftarrow A^t + S$
  for $i = 1 \rightarrow N$ **do**
    if $y_1 \neq y_0$ **then**
      $\phi(AS_i^t) \leftarrow p(y_1|AS_i^t) - p(y_0|AS_i^t) - \frac{\alpha}{\max Z(A^{t_0})} Z(A_i^t)$
    else
      $\phi(AS_i^t) \leftarrow p(y_2|AS_i^t) - p(y_0|AS_i^t)$
    end if
  end for
  for $i = 1 \rightarrow N$ **do**
    $f(AS_i^t) \leftarrow \frac{\phi(AS_i^t)}{\sum_{j=1}^{n} \phi(AS_j^t)}$
    $fr(AS_{i++}^k) \leftarrow \sum_{j=1}^{i} f(AS_j^t)$
  end for
  if $\max\{\phi(AS_i^t)\} > \gamma$ **then**
    Break
  end if
  for $n = 1 \rightarrow N/2$ **do**
    $A_i^t \leftarrow$ SELECTION(AS^t)
    $A_j^t \leftarrow$ SELECTION(AS^t)
    $A_{2n-1}^{t+1}, A_{2n}^{t+1} \leftarrow$ CROSSOVER(P_c, A_i^t, A_j^{t+1})
    $A_{2n-1}^{t+1} \leftarrow$ MUTATION(P_m, A_{2n-1}^{t+1})
    $A_{2n}^{t+1} \leftarrow$ MUTATION(P_m, A_{2n}^{t+1})
  end for
end for
$AS_{opt} \leftarrow$ the AS_i^t with higerst $\phi(AS^t)$

Algorithm 2 The function in POBA-GA.

function SELECTION(AS^t)
  $k \leftarrow range(0, 1)$
  $i \leftarrow 1$
  while $k < fr(AS_i^t)$ **do**
    $i++$
  end while
  return A_i^t
end function

function CROSSOVER(P_c, A_i^t, A_j^{t+1})
  if $rand(0, 1) < P_c$ **then**
    $A_i^{t+1} \leftarrow A_i^t * B + A_j^t * (1 - B)$
    $A_j^{t+1} \leftarrow A_i^t * (1 - B) + A_j^t * B$
  else
    $A_i^{t+1} \leftarrow A_i^t$
    $A_j^{t+1} \leftarrow A_j^t$
  end if
  return A_i^{t+1}, A_j^{t+1}
end function

function MUTATION(P_m, A_i^{t+1})
  if $rand(0, 1) < P_m$ **then**
    $A_i^{t+1} \leftarrow A_i^{t+1} * C$
  else
    $A_i^{t+1} \leftarrow A_i^{t+1}$
  end if
  return A_i^{t+1}
end function

3.6. *Generation update*

Generation is updated by father-son mixed selection. The population size N perturbation with the highest fitness function in A_i^t and A_i^{t+1} are updated to A_i^{t+1}. This method is mainly used to prevent the optimal individual of the current group from being lost in the next generation, which causes the genetic algorithm cannot converge to the global optimal solution. And the termination condition of this paper is to achieve the number of cycles or the fitness function is greater than a certain value γ.

4. Evaluation metrics

In general, the researchers use L_0, L_2 and $L_\propto$ to calculate the perturbation size. However, this paper finds that there are problems in these three metric, so this paper proposes a perturbation evaluation metric improved from Sigmoid. When evaluating perturbation by the naked eye, our metric is more in line with visual assessment of perturbation. When evaluating perturbation by machine, our metric can speed up the reduction of perturbation in adversarial attack, and effectively reduce the difference between adversarial examples and original examples.

4.1. *Naked-eye evaluation*

Current adversarial attack methods often use p-norm L_p to limit perturbations (Goodfellow et al., 2016) of adversarial examples. For example, the zero-norm L_0 distance measures the number of pixels that have changed, the second-norm L_2 distance measures the Euclidean distance between two images, and the infinite-norm $L_\propto$ distance measures the maximum perturbation between two pixels.

However, no matter what kind of metric we use, we cannot express the relationship between the magnitude of the nine perturbations in Fig. 4. L_0 and $L_\propto$ cannot compare the perturbation size of these 9 pictures, and L_2 thinks that the difference between the 9th picture and the 7th picture is equal to the difference between the 1st picture and the 5th picture. However, it is difficult to distinguish the difference between the last three pictures or the difference between the first two

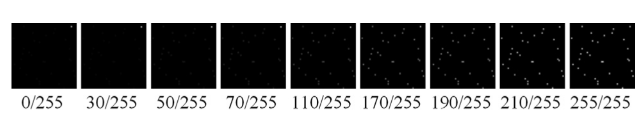


Fig. 4 – Perturbations with different pixel values of noise points. Change the 30 pixels on the 50*50 image as perturbation and normalize the changed pixel value.

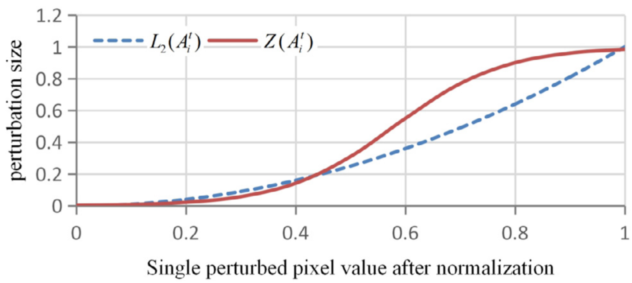


Fig. 5 – Perturbation evaluation curves of L_2 and Z. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

pictures, and the difference between the 5th and 6th picture can clearly distinguished.

The calculation of the perturbation size is more similar to the sigmoid curve distribution. Inspired by this function, we propose a new perturbation calculation indicator $Z(A_i^t)$. For a given adversarial example A_i^t, the perturbation is by Eq. (9).

$$Z(A_i^t) = \sum_{a=1}^{m_a}\sum_{b=1}^{m_b}\left(\frac{1}{1+e^{-|AS_{i(ab)}^t|*pm_1+pm_2}} - \frac{1}{1+e^{pm_2}}\right) \tag{9}$$

where m_a and m_b are the height and width of the image. $AS_{i(ab)}^t$ is the perturbation pixels in row a and column b. pm_1 and pm_2 are the parameters to adjust the perturbation pixel mapping rule. When we evaluate with the naked eye, we set $pm_1 = 10$, $pm_2 = 5.8$.

Compare new perturbation evaluation metric with the L_2, Fig. 5 is a comparison of perturbations evaluation metrics for a single perturbed pixel. Among them, the red line is the calculation result of the perturbations evaluation of our metric, and the blue line is the perturbations evaluation of the L_2. Combined Fig. 4 and Fig. 5, we can find that $Z(A_i^t)$ is more in line with the sensitivity of the human eye to perturbation. The metric proposed in this paper can effectively compensate for the vulnerabilities of L_0, L_2 and L_∞.

4.2. *Machine evaluation*

In the last section we have shown that $Z(A_i^t)$ is more in line with the sensitivity of the human eye to perturbation. And in this section we mainly prove that $Z(A_i^t)$ can better distinguish small perturbation, making the perturbation of adversarial examples generated by adversarial attacks smaller. In general, the maximum perturbation is less than 0.43(110/255), this paper set $pm_1 = 15$, $pm_2 = 3$.

In order to compare the impact of using the $Z(A_i^t)$ metric and the L_2 metric on the POBA-GA attack. We set the fitness function to $\phi(AS_i^t) = P(AS_i^t) - \frac{\alpha}{\max Z(A^0)} Z(A_i^t)$ and $\phi(AS_i^t) = P(AS_i^t) - \frac{\alpha}{\max \|A_i^0\|_2} \|A_i^t\|_2$ respectively. The experimental settings refer to Section 5, and iterating 100 generations. The experimental results are shown in Table 4. It should be noted that in order to prove the validity of our metric, we use L_2 to calculate the perturbation of adversarial examples. Table 4 shows that our metric can speed up the reduction of perturbation in adversarial attack, and effectively reduce the difference between adversarial examples and original examples. This is mainly because $Z(A_i^t)$ expands the difference between perturbation and makes the perturbation optimize faster in the direction of smaller perturbation. From the table we can find that their attack success rate is 96%, because we do not consider the perturbation before the attack is successful, so the metric does not affect the attack success rate.

Table 4 – Influence of disturbance metric on POBA-GA in VGG19.

	Attack success rate (%)	Query count	Perturbation (per-pixel L_2)
L_2	96	5000	5.7e−06
Z	96	5000	4.3e−06

5. Experiments

In this section, we separately experiment on the parameter sensitivity, attack performance, universality of the attack algorithm, its impact on the robustness of the model and the practical application of the algorithm. In Section 5.1, we describe the platform, database, DNN models and attack implementation, making it easy for readers to reproduce the experiment. In Section 5.2, we analyzed the effect of parameter α on the experiment, so that people can quickly select different parameters α according to their needs. In Section 5.3, we compared the POBA-GA with high-quality white-box and black-box attacks, including the state of the art white-box attack C&W and black-box attack AutoZOOM (Tu et al., 2018). In Section 5.4, we study the experiment when the black-box model only return just a single binary outcome and target attack. And we want to show that our POBA-GA can achieve good attack effect and universality in such situation. In Section 5.5, we compare the defense performance of POBA-GA adversarial training and ensemble adversarial training, hoping to prove that the defense capability of the POBA-GA adversarial training is stronger than the ensemble adversarial training. In Section 5.6, we attack the face recognition system, hoping to prove that OPBA-GA not only has good experimental results in the experimental data set, but also can get good attack results in the real world and other data sets.

5.1. *Experiment setup*

Platform: The platform for all experiments is i7-7700K 4.20 GHz × 8 (CPU), TITAN Xp 12 GiB × 2 (GPU), 16 GB × 4 memory (DDR4), Ubuntu 16.04 (OS), Python 3.5, Tensorflow-gpu-1.3, Tflearn-0.3.2.[1]

Database: We use three publicly available image databases, namely, MNIST,[2] CIFAR-10,[3] ImageNet64.[4] MNIST is a

[1] Tflearn can be download at https://github.com/tflearn/tflearn.
[2] MNIST can be download at http://yann.lecun.com/exdb/mnist/.
[3] CIFAR-10 can be download at https://www.cs.toronto.edu/~kriz/cifar.html.
[4] ImageNet64 can be download at http://image-net.org/download-images.

handwritten digitally recognized data set containing 70,000 grayscale images of 28*28 size, divided into 10 classes. The CIFAR-10 data set is also a small image data set, contains 60,000 color images of size 32*32, divided into 10 classes. The Imagenet data set is a large image data set, contains about 15 million images and 22,000 classes. Before the experiment, we reshaped it to a size of 224*224.

Baseline methods: We compare POBA-GA with eight different baselines, including four white box attacks and four black box attacks, including Fast Gradient Sign Model (FGSM) (Goodfellow et al., 2014), DeepFool (Moosavi-Dezfooli et al., 2016), Basic Iterative Model (BIM) (Kurakin et al., 2016), Carlini and Wagner Attacks (C&W) (Carlini and Wagner, 2017), Boundary (Brendel et al., 2017), ZOO (Chen et al., 2017), AutoZOOM (Tu et al., 2018). These baselines are classic and efficient attack methods, including the state of the art white-box attack C&W and black-box attack AutoZOOM.

DNN models: For MNIST and CIFAR-10, we use the same DNN model as in the C&W attack (Carlini and Wagner, 2017),[5] which is also used by ZOO and Boundary (Brendel et al., 2017). On ImageNet we use the same pretrained networks VGG19 (Simonyan and Zisserman, 2014), Resnet50 (He et al., 2016) and Inception-V3 (Inc-V3) (Szegedy et al., 2016) provided by Keras[6] as Boundary.

Attack implementation: In order to make the experimental results of the comparative experiment accurate, the parameters of the comparison algorithm are taken directly from the corresponding literature. In this paper, POBA-GA performs 100 iterations on MINIST and CIFAR-10, generating 20 descendants per iteration, the variance is taken from [5,10,15,20,25], the number of noise points is taken from [50,100,150,200,250]. And POBA-GA performs 400 iterations on ImageNet, and generating 50 descendants per iteration, the variance is taken from [5,10,15,20,25], the number of noise points is taken from $[5000, 7500, 10, 000, 12, 500, 15, 000]$.

All experimental results in this paper are average values. For MNIST and CIFAR-10, we evaluated 1000 randomly examples from the validation set, for ImageNet we used 250 images.

Evaluation metric: This paper uses the attack success rate (ASR) (Dong et al., 2017), perturbation (per-pixel L_2) and query number to evaluate the performance of the experiment. The ASR is used to evaluate the attack probability of the attack method against the target model, which is calculated by the following equation.

$$ASR = \begin{cases} \frac{sumNum(AS_{opt}|y_1=y_{tar})}{sumNum(AS_{opt})} & \text{targeted attack} \\ \frac{sumNum(AS_{opt}|y_1\neq y_0)}{sumNum(AS_{opt})} & \text{non}-\text{targeted attack} \end{cases} \tag{10}$$

where $sumNum(.)$ is the number of examples.

L_2 norm (i.e. Euclidean distance) is used to quantify the difference between the adversarial and the original examples. Query number refers to how many examples the target model needs to predict before the attack method reaches the stop condition. This metric is especially important when the target model has a limit on the number of queries.

[5] https://github.com/carlini/nn_robust_attacks (commit 1193c79).

[6] https://github.com/fchollet/keras (commit 1b5d54).

5.2. *Influence of perturbation ratio parameter α*

The perturbation weight α is used to adjust the balance between perturbation $Z(A_i^t)$ and attack performance $P(AS_i^t)$. When α is larger, the example is more similar to the original one, but the success rate will be lower. When α is smaller, the perturbation of the adversarial example is larger, the attack success rate is higher. The settings of parameter α will change as people's attack requirements change, so we analyzed the effect of parameter α on the experiment so that people can quickly select different parameters α according to their needs. In addition, we also pointed out in the article that if people have clear expectations, they can also use automatic methods to adjust the parameter α, such as irace.[7]

Fig. 6 shows the influence of α on the ImageNet64 data set. From the figures, we can conclude that with the increase of α, the attack performance P and perturbations Z are reduced. This is mainly because with the increase of α, the influence of perturbation on the image gradually increases, the adversarial example is more similar to the original image, and the attack performance P gradually decreases.

According to different actual needs, we will choose a different α. In this paper, we makes $\alpha = 3$, because the perturbation almost does not decrease with the increase of α, and P is still as high as 0.9. It should be noted that although α affects the value of P, since we consider the perturbation after the attack is successful, in general, α does not affect the attack success rate.

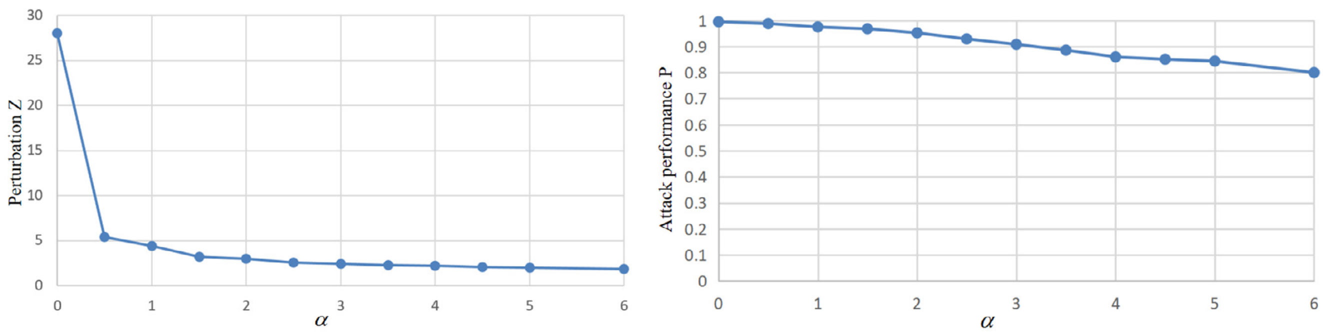
5.3. *Attack performance comparison*

To verify the effectiveness of our approach, we compared POBA-GA with the classic white-box and black-box attack methods. We found that POBA-GA has a high attack success rate, even surpassing some white-box attacks, and it can initially achieve successful attacks with a small number of queries. Table 5 shows the attack success rate, perturbations and attack time cost of the different adversarial attacks. The number of queries in parentheses represents the number of queries when the initial attack succeeds, which is used to assist in comparing the performance of our algorithm with ZOO and AutoZOOM algorithms.

5.3.1. *Perturbation*

The adversarial examples generate by POBA-GA has less perturbation with the same attack success rate. Especially for MNIST and CIFAR-10 datasets, examples generated by our method have less perturbation than most white-box and black-box attacks, mainly because the image is small and the genetic algorithm is easier to find the approximate global optimal solution. For the ImageNet64 data set, since the white-box attack method grasps the internal structure of the target model, we only compare with the black-box attack. From Table 5 we find that the perturbation of POBA-GA is significantly smaller than ZOO and AutoZOOM. Although POBA-GA's perturbation is larger than Boundary, POBA-GA has fewer queries. In addition, when $\alpha = 10$ and the number of queries

[7] Irace https://cran.r-project.org/web/packages/irace/index.html.



(a) Perturbation Z.

(b) Attack Performance P

Fig. 6 – The influence of α on the ImageNet64 data set.

is 54,000, the perturbation generated by POBA-GA can reach 8.1e−07 without affecting the attack success rate, but the attack perturbation P will decrease. Therefore, POBA-GA performs well in perturbation performance. It makes good sense that the perturbations of POBA-GA are much smaller than baselines since the initial perturbations are very small, and the optimization process of GA is capable of gradually reducing the perturbation during iterations.

5.3.2. *Attack success rate*

POBA-GA has a better attack success rate and even better than some white box attacks. Usually, white-box attack conducted based on the internal structure and parameters of the model, can perform a higher success rate attack. The attack success rate of DeepFool is low, mainly because it has strict requirements on perturbation. From Table 5 you can find that ZOO also has a higher attack success rate, however it mainly relies on a large number of queries and iterations.

Specifically, on the ImageNet64 data set, ZOO requires about 220 thousand queries for 90% attack success rate, AutoZOOM requires about 1600 queries for 100% attack success rate, while POBA-GA only need about 500 queries for 96% attack success rate. In addition, the perturbation of POBA-GA is only one-fifth of the ZOO and AutoZOOM, or even smaller. On the whole, POBA-GA achieves state-of-the-art black-box attack performances in consideration of both attack success rate and perturbation. Although there is small margin in success rate when compared with AutoZOOM, POBA-GA needs much less query times and less perturbations. We can conclude that POBA-GA is more practical in real world application when the require time is strict while we can tolerate with relatively high (less than 100%) attack success rate.

Table 5 – Attack performance comparison.

	Black/white-box	Attack method	MNIST	CIFAR-10	ImageNet64		
					VGG19	Resnet50	Inc-V3
Perturbation (per-pixel L_2)	White-box Adversarial attack	FGSM	6.5e−02	7.3e−05	3.4e−06	4.0e−06	2.6e−06
		DeepFool	3.2e−03	4.1e−06	2.4e−07	9.3e−08	9.1e−08
		BIM	8.2e−03	1.2e−05	8.5e−07	8.2e−07	6.4e−07
		C&W	3.1e−03	6.9e−06	5.7e−07	2.2e−07	7.6e−08
	Black-box Adversarial attack	Boundary	4.0e−03	6.4e−06	3.5e−07	2.1e−07	4.2e−07
		ZOO	4.3e−03	5.8e−04	3.9e−05	3.2e−05	2.8e−05
		AutoZOOM	6.4e−03	7.2e−04	6.2e−05	5.1e−05	5.4e−05
		POBA-GA	3.0e−03	6.8e−05	1.5e−05	1.4e−05	1.7e−05
Attack success rate	White-box Adversarial attack	FGSM	86%	89%	77%	80%	82%
		DeepFool	90%	87%	75%	79%	75%
		BIM	98%	98%	97%	96%	94%
		C&W	100%	100%	99%	100%	99%
	Black-box Adversarial attack	Boundary	78%	76%	72%	70%	70%
		ZOO	100%	100%	90%	90%	88%
		AutoZOOM	100%	100%	100%	100%	100%
		POBA-GA	100%	100%	96%	98%	95%
Query count	Black-box Adversarial attack	Boundary	12,500	12,500	125,000	125,000	125,000
		ZOO	9250	4324	235,272	223,143	264,170
		AutoZOOM	445(100)	103(86)	4647(1686)	4256(1695)	4051(1701)
		POBA-GA	423(94)	381(78)	3786(536)	3614(492)	3573(471)

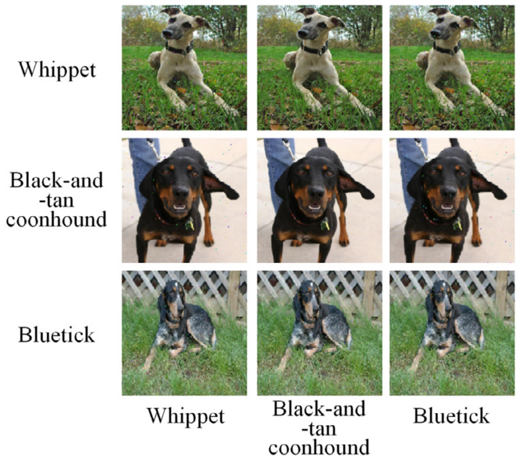


Fig. 7 – Target attack. The rows represent the original label and the column represents the output label of the target model.

The reason why we have such a high attack success rate is mainly because we do not consider the effects of the perturbation on the experiment before the attack is successful. But it should be noted that even if we do not consider the effects of the perturbation, the perturbation will not be very large, as we have limited the perturbation at initialization.

5.3.3. *Query count*

POBA-GA has fewer queries than other black-box attacks. Since white-box attacks are conducted based on the internal structure of the model, we only analyze black-box attacks. From Table 5, we can find that POBA-GA and AutoZOOM need significantly fewer queries than Boundary and ZOO. AutoZOOM reduces the query time by adopting an adaptive random full gradient estimation strategy to strike a balance between query counts and estimation errors, and features a decoder (AE or BiLIN) for attack dimension reduction and algorithm acceleration. POBA-GA reduces the number of queries by genetic algorithm. The main reasons are as follows: (1) when selecting examples, it is more likely to choose examples with high adaptability. (2) The high probability of crossover and variation increases the example diversity. (3) Use father and son mixed selection to retain the best example. (4) The effects of the perturbations are not considered before the attack is successful.

5.4. *Universality*

Targeted attack: Non-targeted attack only needs to make the *TM* output label different from the correct one, and targeted attack need the target model output specified label. In order to verify the universality of the POBA-GA algorithm, we also achieve a target attack on *TM*. The fitness function value is designed as $\phi(AS_i^t) = p(y_{tar}|AS_i^t) - p(y_0|AS_i^t) - \alpha Z(AS_i^t)$, where $p(y_{tar}|AS_i^t)$ is the probability that the target model will classify the input picture as label y_{tar}. Fig. 7 shows the targeted attack. Each line represents the original label, and each column represents the output label of the target model. The adversarial example average fitness function $\phi(AS_{pot}) = 0.53$, perturbation $Z(A_{pot}) = 3.846$. Therefore, the POBA-GA method can implement targeted attacks. We randomly selected 50 examples from the ImageNet64 data set to attack VGG19, inc-V3 and Resnet50. The attack success rates were 82%, 80% and 84%, respectively.

Table 6 – The performance of BOPA-GA on different outcome models.

	Attack success rate	Query count (initial success)
RCC	98%	536
RSB	73%	6276

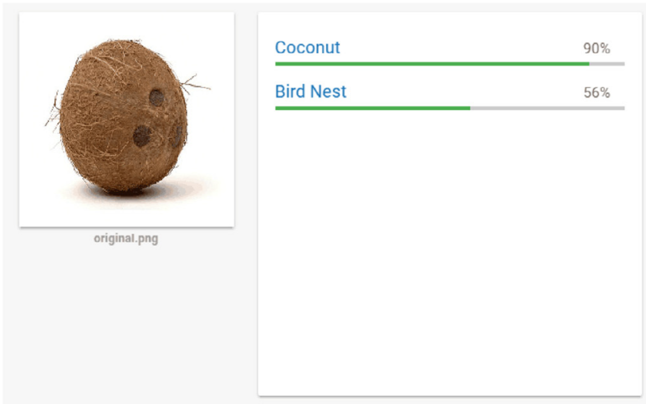
A single binary outcome: In general, the DNN model will give the confidence of the classification, but there are also some depth models that only return a single binary. Therefore, we experiment with models that just a single binary outcome. Different with RCC (Return confidence of classification), we cannot calculate the attack performance $P(AS_i^t)$ of RSB (Return a single binary). Therefore, we use Monte Carlo approximation to estimate the confidence of the classification, and then optimize the perturbation with POBA-GA. Estimating the confidence of AS_i^t with Monte Carlo approximation can be expressed by the following equation.

$$\widehat{P}(AS_i^t) = \frac{1}{N'}\sum_{i=1}^{N'} R(AS_i^t + \delta) \tag{11}$$

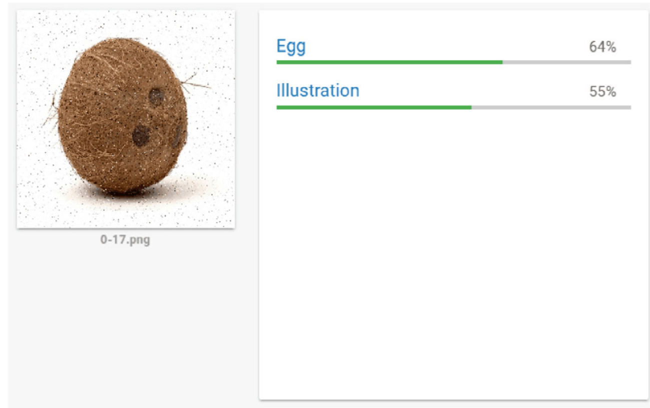
where $\delta \sim \aleph(0, 30)$, $N' = 100$ is the number of examples used to estimating the confidence of AS_i^t. $R(AS_i^t + \delta)$ is the binary outcome of $AS_i^t + \delta$, for example, when $AS_i^t + \delta$ is predicted to the be the second class, then $R(AS_i^t + \delta) = [0, 1, 0, \ldots, 0]$.

Table 6 shows the comparison using the RCC(Return confidence of classification) and RSB (Return a single binary) on the VGG19. From the Table 6 we can find that if the model just return a single binary, the number of queries will increase by about 100 times, mainly because we need to estimate the confidence of the example through 100 queries. The attack success rate will be reduced from 98% to 73%. This is mainly because, although we have estimated the confidence by Monte Carlo approximation, there is a certain difference from the actual situation. If you want to increase the attack success rate, you can increase the N' or the number of iterations.

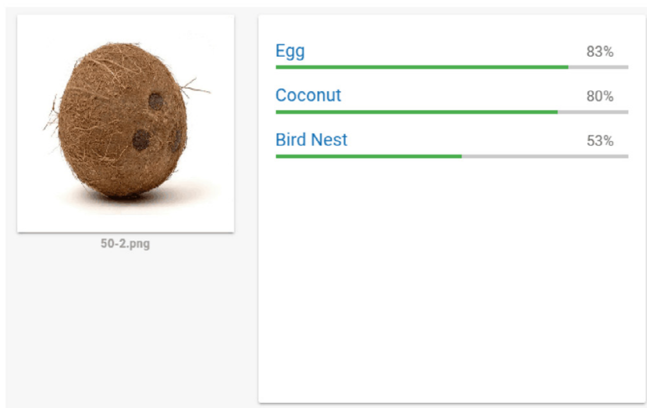
Real world experimentation: In this section, we perform real world experimentation on GCP (which has a black box model), to demonstrate the applicability of the proposed methodologies. Considering the limited number of model queries in the actual scene, this paper only queries 1000 times, i.e. iterative 50 generations, generating 20 examples per generation. The experimental results are shown in Fig. 8. Fig. 8(a) is the prediction result of the original example, Fig. 8(b) is the prediction result of a certain initial adversarial example, and Fig. 8(c) is the prediction result of a 50th generation adversarial example. The experimental results show that if we do not consider the perturbation, we can quickly achieve the attack on GCP. During the optimization process, we optimized the perturbation while ensuring that the attack was successful, and increased the confidence of the eggs from 64% to 83%.



(a) The prediction result of the original example



(b) The prediction result of an initial adversarial example



(c) The prediction result of a 50^{th} generation adversarial example

Fig. 8 – The prediction results of GCP platform.

Considering the limited number of model queries in the actual scene, this paper only queries 1000 times. If the query continues, the confidence of the real class label will further decrease. Therefore, POBA-GA can achieve the attack on the real world experimentation of GCP and proves the applicability of the proposed method.

5.5. *Defensiveness from POBA-GA adversarial training*

In this section, we compare the defense performance of POBA-GA adversarial training and ensemble adversarial training, and prove that the defense capability of the ensemble adversarial training is weak. Defender need add the adversarial examples generate by POBA-GA into the training data set of defense model to improve the robustness of the model. The adversarial training in this paper is achieved by retraining the model after adding the adversarial examples to the training data set. The retraining data set has 20 different classes, each class consisting of 900 normal examples and 100 adversarial examples. In the ensemble adversarial training, we use FGSM, SJMA, Deep-Fool, BIM and C&W generated 5*20 adversarial examples for each class. In the POBA-GA adversarial training, we added 100 POBA-GA adversarial examples to the training data set. Both POBA-GA adversarial training

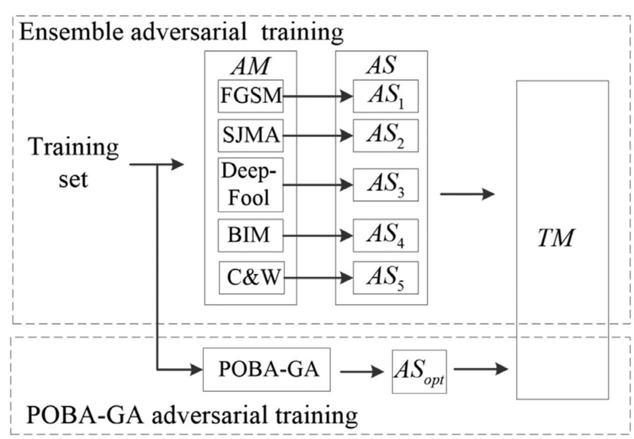


Fig. 9 – The block diagram of the defense. Ensemble adversarial training is mainly used to train and defend against the adversarial examples generated by FGSM, SJMA, DeepFool, BIM and C&W. POBA-GA adversarial training is the defense method proposed in this paper.

and ensemble adversarial training use a batch size of 100 and *STEPS* (iterations number) of 10,000. The *LR*(Learning Rate) of

Table 7 – The defensive comparison of ensemble adversarial training and POBA-GA adversarial training.

Defense method	Attack category	Algorithm	MNIST	CIFAR-10	ImageNet64		
					VGG19	Resnet50	Inc-V3
Ensemble adversarial training	White-box Adversarial attack	FGSM	16%	15%	20%	22%	22%
		DeepFool	15%	10%	10%	8%	10%
		BIM	14%	14%	18%	18%	20%
		C&W	18%	18%	22%	20%	20%
	Black-box Adversarial attack	Boundary	38%	36%	52%	50%	50%
		ZOO	64%	60%	78%	80%	78%
		AutoZOOM	60%	59%	79%	76%	78%
		POBA-GA	**70%**	**72%**	**88%**	**86%**	**85%**
POBA-GA adversarial training	White-box Adversarial attack	FGSM	32%	35%	50%	54%	52%
		DeepFool	33%	31%	36%	34%	38%
		BIM	35%	33%	42%	44%	40%
		C&W	32%	35%	52%	50%	51%
	Black-box Adversarial attack	Boundary	37%	38%	51%	52%	54%
		ZOO	44%	42%	74%	74%	72%
		AutoZOOM	46%	40%	76%	73%	76%
		POBA-GA	**26%**	**28%**	**34%**	**32%**	**32%**

Fig. 10 – Facial recognition application. The photos of first row are the original images, the photos of second row are adversarial examples, and the photos of third row are the faces corresponding to the wrong label.

this paper is calculated by the following equation.

$$LR = BLR * e^{-\frac{i * in(0.1/MLR)}{STEPS}} \qquad (12)$$

where BLR is the Base Learning Rate, MLR is the Minimum Learning Rate and i is the current iterations number. In this paper we make $BLR = 0.1$, $MLR = 0.001$. The adversarial training process is shown in Fig. 9.

Ensemble adversarial training: Generate adversarial examples to classifier the adversarial examples generated by ensemble attacks and normal examples. In our case, FGSM, SJMA, DeepFool, BIM, C&W are adopted as ensemble model to attack *TM*.

POBA-GA adversarial training: Adversarial training is applied to *TM* based on adversarial examples generated by POBA-GA. POBA-GA can provide high-quality adversarial examples for training which could improve defensibility of the target model.

Table 7 shows the target model's defensibility comparison after ensemble adversarial training and POBA-GA adversarial training. Attack success rate of almost all attacks are decreased significantly compared with before adversarial training (in Table 5). From the experiment result, we can also find that the attack success rate of MNIST and CIFAR-10 are lower than ImageNet64. This is mainly because they have fewer pixels and similar adversarial examples, while ImageNet64 has more pixels and can choose more ways to change. We can find that the adversarial training based on POBA-GA attacks is less effective than ensemble adversarial training on white-box attack. This is mainly because the ensemble adversarial training is implemented based on the adversarial examples generated by the white-box attack. However, the adversarial defense against black-box attacks is better than the ensemble defense. The most important thing is that adversarial defense reduce the attack success rate of POBA-GA to about 30%.

5.6. *Facial recognition application*

Our experiments were mainly carried out on data sets such as VGG19, inc-V3 and Resnet50. However, such detectors are not widely used in practical applications. Therefore, we expand the application scenario of POBA-GA. In recent years, facial recognition-based identity authentication systems have become popular and widely used in life (Deng et al., 2017b; Zhou and Lam, 2018), so the security of face recognition systems has become increasingly important.

The experiment selected the Wild dataset (LFW) as the experimental data set.The Labeled Faces in the Wild dataset (LFW) (Huang et al., 2008) containing more than 5000 faces and more than 10,000 images. We use POBA-GA to attack LFW faces and generate corresponding adversarial examples. Fig. 10 shows the original image and their true labels, and the adversarial examples generated by POBA-GA and the target model predicted label. From the figure we can see that POBA-GA can indeed achieve face attacks through perturbation optimization, which has strong applicability.

6. Conclusion

Adversarial attack against deep model can cause fundamental errors. We focus on black-box attacks since it is more operable and harder to defend. Different from current black-box adversarial attacks, we propose a novel evolutionary algorithm based adversarial example generation method for black-box attack implementation. A perturbation optimized black-box attacks (POBA-GA) is put forward against deep neural networks. Abundant experiments are carried out compared with classic white-box and black-box adversarial attack methods. The results prove that POBA-GA has higher attack success rate than other attack methods. It can achieve 100% attack success rate in CIFAR-10 and MNIST classification models, and it can achieve 96% attack success rate on ImageNet64 black-box method. In both attack success rate and perturbation control, POBA-GA has better performance than existing black-box attack. For further study, we will study on POBA-GA's attack transferability on different target models.

Declaration of interests

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.

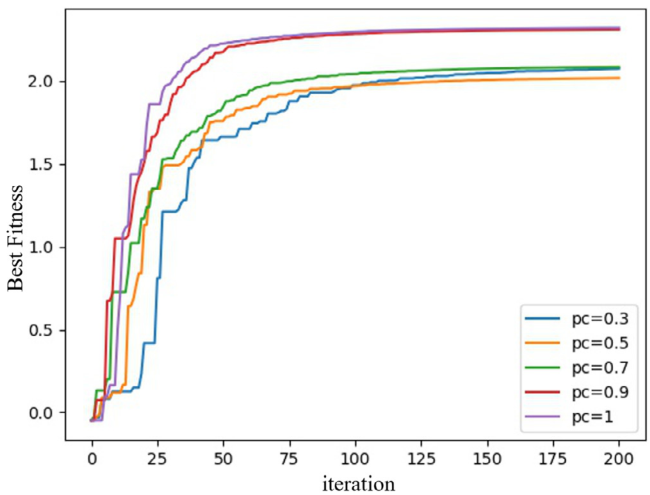
Acknowledgment

This work is partially supported by National Natural Science Foundation of China (61502423 and 61572439), Zhejiang Natural Science Foundation (LY19F020025), Signal Recognition Based on GAN, Deep Learning for Enhancement Recognition Project, Zhejiang University Open Fund (2018KFJJ07), and Zhejiang Science and Technology Plan Project (2017C33149).

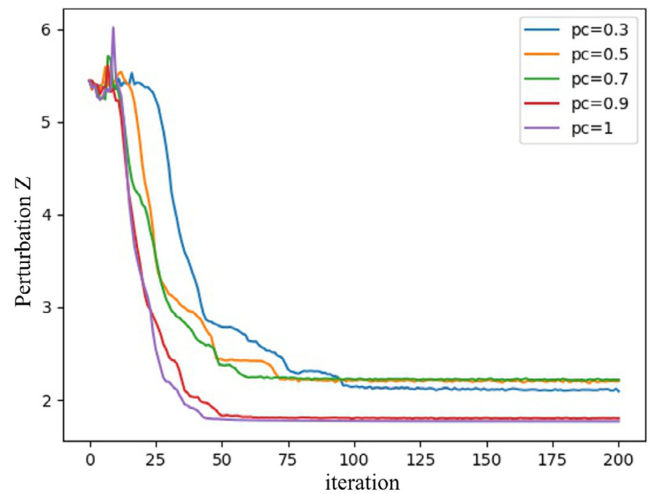
Appendix

Parameter adjustment

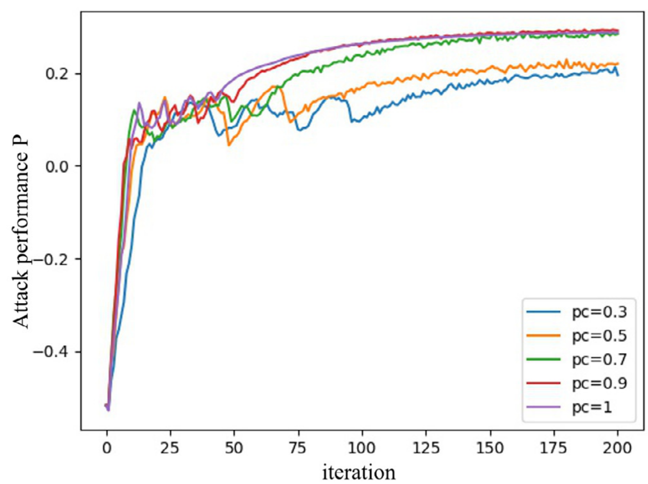
Fig. 11 shows the effect of P_c on experimental results. Fig. 11(a) shows the best of the fitness function of the parent adversarial the example. Fig. 11(b) and (c) shows the perturbation and attack performance P of examples with the best fitness function value.



(a) Best fitness function value



(b) Perturbation



(c) Attack performance P

Fig. 11 – The effect of P_c on experimental results.

REFERENCES

Akhtar N., Mian A. Threat of adversarial attacks on deep learning in computer vision: a survey. arXiv:180100553 2018.

Alencar M, Souza J, Souza B, Neves W. Optimal allocation of photovoltaic panels in distribution network applying genetic algorithm. Proceedings of the 2018 simposio Brasileiro de sistemas eletricos (SBSE). IEEE, 2018.

Bai W, Quan C, Luo Z. Alleviating adversarial attacks via convolutional autoencoder. In: Proceedings of the eighteenth IEEE/ACIS international conference on software engineering, artificial intelligence, networking and parallel/distributed computing (SNPD). IEEE; 2017. p. 53–8.

Bose A.J., Aarabi P. Adversarial attacks on face detectors using neural net based constrained optimization. arXiv:180512302 2018.

Brendel W., Rauber J., Bethge M. Decision-based adversarial attacks: reliable attacks against black-box machine learning models. arXiv:171204248 2017.

Carlini N, Mishra P, Vaidya T, Zhang Y, Sherr M, Shields C, Wagner D, Zhou W. Hidden voice commands.. In: Proceedings of the 2016 USENIX security symposium; 2016. p. 513–30.

Carlini N, Wagner D. Towards evaluating the robustness of neural networks. In: Proceedings of the 2017 IEEE symposium on security and privacy (SP). IEEE; 2017. p. 39–57.

Chen F, Ma J. An empirical identification method of Gaussian blur parameter for image deblurring. IEEE Trans Signal Process 2009;57(7):2467–78.

Chen PY, Zhang H, Sharma Y, Yi J, Hsieh CJ. ZOO: zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In: Proceedings of the tenth ACM workshop on artificial intelligence and security. ACM; 2017. p. 15–26.

Chen ST, Cornelius C, Martin J, Chau DH. Robust physical adversarial attack on faster R-CNN object detector, 2018. arXiv: 180405810.

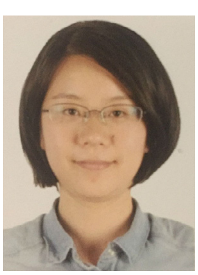
Cisse M, Adi Y, Neverova N, Keshet J. Houdini: fooling deep structured prediction models, 2017. arXiv: 170705373.

De Cnudde S, Martens D, Provost F, et al. An exploratory study towards applying and demystifying deep learning classification on behavioral big data. Faculty of Applied Economics; Technical Report, 2018.

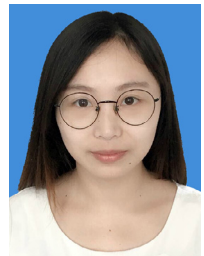
Deng S, Huang L, Xu G, Wu X, Wu Z. On deep learning for trust-aware recommendations in social networks. IEEE Trans Neural Netw Learn Syst 2017;28(5):1164–77.

Deng W, Hu J, Zhang N, Chen B, Guo J. Fine-grained face verification: FGLFW database, baselines, and human-DCMN partnership. Pattern Recognit 2017;66:63–73.
Dong Y, Liao F, Pang T, Su H, Zhu J, Hu X, Li J. Boosting adversarial attacks with momentum. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018. p. 9185–93.
Dong Y., Liao F., Pang T., Su H., Zhu J., Hu X., Li J. Boosting adversarial attacks with momentum. arXiv preprint 2018.
Eykholt K., Evtimov I., Fernandes E., Li B., Rahmati A., Xiao C., Prakash A., Kohno T., Song D. Robust physical-world attacks on deep learning models. arXiv:170708945 2017.
Geifman Y, El-Yaniv R. Selective classification for deep neural networks. In: Proceedings of the advances in neural information processing systems; 2017. p. 4885–94.
Goodfellow I, Bengio Y, Courville A, Bengio Y, 1. Cambridge: MIT Press; 2016.
Goodfellow I.J., Shlens J., Szegedy C. Explaining and harnessing adversarial examples. arXiv:14126572 2014.
Goyal N, Bhatia R, Kumar M. A genetic algorithm based focused Web crawler for automatic webpage classification; 2016. p. 1–6.
Havaei M, Davy A, Warde-Farley D, Biard A, Courville A, Bengio Y, Pal C, Jodoin PM, Larochelle H. Brain tumor segmentation with deep neural networks. Med Image Anal 2017;35:18–31.
Hayes J., Danezis G. Machine learning as an adversarial service: learning black-box adversarial examples. arXiv:170805207 2017.
He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 770–8.
Hu W., Tan Y. Black-box attacks against RNN based malware detection algorithms. arXiv:170508131 2017.
Huang GB, Mattar M, Berg T, Learned-Miller E. Labeled faces in the wild: a database forstudying face recognition in unconstrained environments. Proceedings of the workshop on faces in 'Real-Life' images: detection, alignment, and recognition, 2008.
Ilyas A., Engstrom L., Athalye A., Lin J. Black-box adversarial attacks with limited queries and information. arXiv:180408598 2018.
Kalajac E., Karabegović A., Ponjavić M. Optimization of the structures locations using a genetic algorithm in the transmission line design. In: Proceedings of the 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). Opatija, Croatia. IEEE; 2018.
Konak A, Coit DW, Smith AE. Multi-objective optimization using genetic algorithms: a tutorial. Reliabil Eng Syst Saf 2006;91(9):992–1007.
Kooi T, Litjens G, van Ginneken B, Gubern-Mérida A, Sánchez CI, Mann R, den Heeten A, Karssemeijer N. Large scale deep learning for computer aided detection of mammographic lesions. Med Image Anal 2017;35:303–12.
Kurakin A., Goodfellow I., Bengio S. Adversarial examples in the physical world. arXiv:160702533 2016.
Lipowski A, Lipowska D. Roulette-wheel selection via stochastic acceptance. Phys A Stat Mech Appl 2012;391(6):2193–6.
Litjens G, Kooi T, Bejnordi BE, Setio AAA, Ciompi F, Ghafoorian M, van der Laak JA, van Ginneken B, Sánchez CI. A survey on deep learning in medical image analysis. Med Image Anal 2017;42:60–88.
Liu Q, Liu T, Liu Z, Wang Y, Jin Y, Wen W. Security analysis and enhancement of model compressed deep learning systems under adversarial attacks. In: Proceedings of the twenty-third Asia and south pacific design automation conference. IEEE Press; 2018. p. 721–6.
Martinez-Gil J, Alba E, Aldana-Montes JF. Optimizing ontology alignments by using genetic algorithms. In: Proceedings of the workshop on nature based reasoning for the semantic Web. Germany: Karlsruhe, 2008.
Metzen JH, Kumar MC, Brox T, et al. Universal adversarial perturbations against semantic image segmentation. IEEE International Conference on Computer Vision (ICCV). IEEE; 2017. p. 2774–83.
Milton M.A.A. Evaluation of momentum diverse input iterative fast gradient sign method (M-DI2-FGSM) based attack method on MCS 2018 adversarial attacks on black box face recognition system. arXiv:180608970 2018.
Moosavi-Dezfooli SM, Fawzi A, Fawzi O, et al. Analysis of universal adversarial perturbations. arXiv preprint arXiv:1705.09554, 2017.
Moosavi-Dezfooli SM, Fawzi A, Frossard P. DeepFool: a simple and accurate method to fool deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 2574–82.
Moosavidezfooli SM, Fawzi A, Fawzi O, Frossard P. Universal adversarial perturbations. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2017. p. 86–94.
Papernot N, McDaniel P, Goodfellow I, Jha S, Celik ZB, Swami A. Practical black-box attacks against machine learning. In: Proceedings of the 2017 ACM on Asia conference on computer and communications security. ACM; 2017. p. 506–19.
Papernot N, McDaniel P, Jha S, Fredrikson M, Celik ZB, Swami A. The limitations of deep learning in adversarial settings. In: Proceedings of the 2016 IEEE European symposium on security and privacy (EuroS&P). IEEE; 2016. p. 372–87.
Ramanathan A, Pullum L, Husein Z, Raj S, Torosdagli N, Pattanaik S, Jha SK. Adversarial attacks on computer vision algorithms using natural perturbations. In: Proceedings of the tenth international conference on contemporary computing (IC3). IEEE; 2017. p. 1–6.
Sarkar S., Bansal A., Mahbub U., Chellappa R. UPSET and ANGRI: breaking high performance image classifiers. arXiv:170701159 2017.
Sethi TS, Kantardzic M. Data driven exploratory attacks on black box classifiers in adversarial domains. Neurocomputing 2018;289:129–43.
Sharif M, Bhagavatula S, Bauer L, Reiter MK. 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. ACM; 2016. p. 1528–40.
Sharif M., Bhagavatula S., Bauer L., Reiter M.K. Adversarial generative nets: neural network attacks on state-of-the-art face recognition. arXiv:180100349 2017.
Simonyan K., Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv:14091556 2014.
Smith L., Gal Y. Understanding measures of uncertainty for adversarial example detection. arXiv:180308533 2018.
Souza LA, Silva CL, Calixto WP. Optimized setting of directional overcurrent relays via genetic algorithm. Proceedings of the 2018 simposio Brasileiro de sistemas eletricos (SBSE). IEEE, 2018.
Stilgoe J. Machine learning, social learning and the governance of self-driving cars. Soc Stud Sci 2018;48(1):25–56.
Su J., Vargas D.V., Kouichi S. One pixel attack for fooling deep neural networks. arXiv:171008864 2017.
Sun M, Tang F, Yi J, Wang F, Zhou J. Identify susceptible locations in medical records via adversarial attacks on deep predictive models. In: Proceedings of the twenty-fourth ACM SIGKDD international conference on knowledge discovery & data mining. ACM; 2018. p. 793–801.
Syahputra R. Distribution network optimization based on genetic algorithm. J Electr Technol UMY 2017;1(1):1–9.

Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 2818–26.
Szegedy C, Vanhoucke VO. Processing images using deep neural networks: U.S. Patent; 2017. 9,715,642.
Szegedy C., Zaremba W., Sutskever I., Bruna J., Erhan D., Goodfellow I., Fergus R. Intriguing properties of neural networks. arXiv:13126199 2013.
Talha M, Saeed MS, Mohiuddin G, Ahmad M, Nazar MJ, Javaid N. Energy optimization in home energy management system using artificial fish swarm algorithm and genetic algorithm. In: Proceedings of the international conference on intelligent networking and collaborative systems. Springer; 2017. p. 203–13.
Tu CC, Ting P, Chen PY, et al. Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks. arXiv preprint arXiv:1805.11770, 2018.
Varatharajan R, Vasanth K, Gunasekaran M, Priyan M, Gao X. An adaptive decision based Kriging interpolation algorithm for the removal of high density salt and pepper noise in images. Comput Electr Eng 2018;70:447–61.
Xu W, Qi Y, Evans D. Automatically evading classifiers. Proceedings of the 2016 network and distributed systems symposium, 2016.
Yin Z, Wang F, Liu W, Chawla S. Sparse feature attacks in adversarial learning. IEEE Trans Knowl Data Eng 2018;30(6):1164–77.
Yosinski J, Clune J, Bengio Y, Lipson H. How transferable are features in deep neural networks?. In: Proceedings of the advances in neural information processing systems; 2014. p. 3320–8.
Yu F, Dong Q, Chen X. Asp: A fast adversarial attack example generation framework based on adversarial saliency prediction, 2018. arXiv: 180205763.
Yuan Y, Mou L, Lu X. Scene recognition by manifold regularized deep learning architecture. IEEE Trans Neural Netw Learn Syst 2015;26(10):2222–33.
Zhou H, Lam KM. Age-invariant face recognition based on identity inference from appearance age. Pattern Recognit 2018;76:191–202.
Zhu Y, Mottaghi R, Kolve E, Lim JJ, Gupta A, Fei-Fei L, Farhadi A. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In: Proceedings of the 2017 IEEE international conference on robotics and automation (ICRA). IEEE; 2017. p. 3357–64.



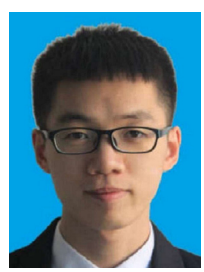
Jinyin Chen received B.S. and Ph.D. degrees from Zhejiang University of Technology, Hangzhou, China, in 2004 and 2009, respectively. She studied evolutionary computing in Ashikaga Institute of Technology, Japan in 2005 and 2006. She is currently an associate professor in college of information engineering, Zhejiang University of Technology. Her research interests include evolutionary computing, data mining and deep learning algorithm.



Mengmeng Su is a Master student at the college of Information engineering, Zhejiang University of Technology. She received her bachelor degree from Zhejiang University of Technology in 2017. Her research interest covers intelligent computing, artificial immune and industrial safety.

Shijing Shen is a Master student at the college of Information engineering, Zhejiang University of Technology. She research interests include deep learning and computer vision.

Hui Xiong is a Master student at the college of Information engineering, Zhejiang University of Technology. His research interests are image processing and artificial intelligence.



Haibin Zheng is a Master student at the college of Information engineering, Zhejiang University of Technology. He received his bachelor degree from Zhejiang University of Technology in 2017. His research interests include data mining and applications, and bioinformatics.