Hindawi
Security and Communication Networks
Volume 2023, Article ID 3427385, 15 pages
https://doi.org/10.1155/2023/3427385

Research Article

Explore Gap between 3D DNN and Human Vision Utilizing Fooling Point Cloud Generated by MEHHO

Linkun Fan, Fazhi He, Bing Li, Xiaoxin Gao, and Jinkun Luo

School of Computer Science, Wuhan University, Wuhan 430072, Hubei, China

Correspondence should be addressed to Fazhi He; fzhe@whu.edu.cn and Bing Li; bingli@whu.edu.cn

Received 4 February 2023; Revised 4 April 2023; Accepted 6 April 2023; Published 2 May 2023

Academic Editor: Jie Cui

Deep neural network (DNN) has replaced humans to make decisions in many security-critical senses such as face recognition and automatic drive. Essentially, researchers try to teach DNN to simulate human behavior. However, many evidences show that there is a huge gap between humans and DNN, which has raised lots of security concern. Adversarial sample is a common way to show the gap between DNN and humans in recognizing objects with similar appearance. However, we argue that the difference is not limited to adversarial samples. Hence, this paper explores such differences in a new way by generating fooling samples in 3D point cloud domain. Specifically, the fooling point cloud is hardly recognized by human vision but is classified to the target class by the victim 3D point cloud DNN (3D DNN) with more than **99.99**% confidence. Furthermore, to search for the optimal fooling point cloud, a new evolutionary algorithm named Multielites Harris Hawk Optimization (MEHHO) with enhanced exploitation ability is designed. On one hand, our experiments demonstrate that: (1) 3D DNN tends to learn high-level features of one object; (2) 3D DNN that makes decisions relying on more points is more robust; and (3) the gap is hardly learned by 3D DNN. On the other hand, the comparison experiments show that the designed MEHHO outperforms the SOTA evolutionary algorithms w.r.t. statistics and convergence results.

1. Introduction

Deep Neural Network (DNN) techniques have achieved remarkable success on a variety of tasks, particularly in the fields of 2D image [1–5] and 3D shape [6–11]. In many security-critical senses, DNN has replaced humans to make decisions, such as natural language recognition, face recognition, and pedestrian reidentification. In the future, many complicated scenarios will urge to apply DNN for development, for example, level-five automatic driving and human-like service robot. Essentially, one common characteristic of these tasks is to teach DNN to simulate human behavior.

Although the prospects look bright, DNN has been proven to be vulnerable to a carefully crafted adversarial attack, no matter whether in the field of 2D images [12–20] or 3D point clouds [21–25]. The aim of adversarial attack is to generate fake samples that will not cause people to be alarmed but will mislead DNN to make wrong decision, as shown in Figure 1(b). Such a phenomenon has raised many security concerns. Szegedy et al. [12] are the pioneers to point out the vulnerability of DNN in the 2D image domain. Later, lots of attacking methods are designed [13, 16, 26]. For the 3D point cloud which is the most simple format to represent a 3D shape, adversarial samples are generated by points perturbing, attaching, and detaching [21, 22, 27, 28].

Adversarial attack takes the benign samples as the initial target and adjusts its feature to cross the decision boundary. It can be modeled as a roughly min-max problem which minimizes the probability of the original class and maximizes the probability of the targeted class. Many recent adversarial attacks on 3D point cloud are able to achieve 100% success rate [21, 27, 28]. These successes exhibit the huge difference between 3D DNN and human vision. Exploring such differences is helpful to accelerate the development of 3D DNN robustness. However, most adversarial defenses [29–36] for 3D DNN are mainly aimed at defense specific attack and do not explain this difference clearly.

Essentially, the adversarial sample itself is one way to show the gap between 3D DNN and humans in recognizing objects with similar appearance. But we argue that the gap will not be limited to adversarial samples since the disparity between DNN and the human brain. Unfortunately, the adversarial samples usually stay near the decision boundary and will not move inside the class space since the attacking purpose is already achieved. Such a characteristic prevents it from further exhibiting information about the disparity between 3D DNN and human vision.

Hence, this paper aims to explore the disparity using the newly defined fooling samples (illustrated in Figure 1(a)). The fooling samples exhibit the gap between appearances that humans and 3D DNN think category t should look like. For this purpose, we should have an extremely high level of the confidence in fooling samples. Therefore, the fooling sample is searched by an evolutionary algorithm with high exploitation ability. The formal generation process is as follows: given a 3D DNN and a target class, we generate a fooling point cloud by an evolutionary algorithm. The 3D DNN believes it belongs to the target class with more than 99.99% confidence, but human vision can hardly recognize it.

To efficiently generate fooling samples, we model the generation process as an optimization problem and design MEHHO with high exploitation ability to solve it. Specifically, Harris Hawk Optimization (HHO) [37] is an evolutionary algorithm which simulates the hunting process of harris hawk. It has dealt with a wide variety of problems ranging from operational cost optimization, engineering design, to copyright protection and data authentication [38–41]. However, HHO always stuck in the local optimum due to the nonlinearity of DNN function. Therefore, we propose the Muti-Elite Harris Hawk Optimization (MEHHO) based on group besiege and spiral-shaped attack. The experiment results show that the classification confidence of the fooling point cloud generated by MEHHO is higher than the compared algorithms.

After obtaining the fooling point cloud by MEHHO, we explore the gap between 3D DNN and human vision and further give some advice about the characteristics of 3D DNN by analyzing the fooling samples. In summary, we make the following contributions in this paper:

(i) We exhibit the difference between 3D DNN and human vision in a new way by generating fooling point cloud.

(ii) A new evolutionary algorithm MEHHO is proposed, which achieves better statistic and convergence results on the generation of fooling point cloud.

(iii) We explore the characteristics of 3D DNN by analyzing the feature of the fooling point cloud.

2. Related Works

2.1. DNNs for 3D Point Cloud. In order to perform weight sharing or other kernel optimizations, typical DNNs require regular data format as input, such as 2D image. However, a 3D point cloud is unordered and therefore cannot be directly put into typical DNNs. Most researchers transform such data to regular format by multiview projection [42–44] or 3D voxel grids [45–47], but such transformation will inevitably cause disturbances. To address this issue, Qi et al. [6] proposes PointNet which directly processes on the point cloud. It is the benchmark for many efficient 3D DNNs. Since PointNet cannot capture local structure information, Qi et al. [7] propose PointNet++, whose key is composed by the sampling layer and the grouping layer. DGCNN [48] is a typical work of graph-based methods; it transforms a point cloud to a graph and learns 3D feature using EdgeConv. LDGCNN [49] links extracted features between different layers and simplifies the transformation network of DGCNN to improve its performance. Due to the irregularity of the point cloud, it is more complicated to design the convolution kernel for 3D DNN. RS-CNN [50] implements the convolution by MLP, which aims to learn the mapping from low-level relations to high-level relations between points in the local subset. Point-Bert [10] learns 3D point cloud transformers by a Bert-style pretraining to. It achieves 93.8% accuracy on ModelNet40 for classification which surpasses most 3D DNNs.

2.2. Adversarial Attack and Defense on 3D DNN. Xiang et al. [21] propose the first work to generate adversarial point clouds; the author crafts adversarial point clouds against PointNet by point perturbation and point generation. To limit the deformation of adversarial point cloud, three perturbation metrics were introduced as the constraint. Since critical points have the greatest impact on 3D DNN when making its decisions, Zheng et al. [22] and Naderi et al. [51] focus on generating a point cloud saliency map to illustrate which points are important for 3D DNN. Then the author drops top n most important points for attack. Zhao et al. [23] propose a black-box attack and a white-box attack in 3D adversarial settings, respectively. The result shows the vulnerability of the 3D deep learning model under isometric transformations. Aiming at improving the time efficiency of adversarial attack, Zhou et al. [52] proposed LG-GAN for real-time flexible targeted point cloud attack. In the field of autonomous driving which requires higher robustness, Wang et al. [27] attack a 3D object detection task, and 3D adversarial point clouds are generated based on the idea of point cloud perturbations. To promote the robustness of 3D DNN directly, Yang et al. [28] proposed an attack and defense scheme at the same time. Otherwise, the imperceptibility of the adversarial point cloud is important for bypassing defense. Wen et al. [53] newly design geometry-aware objectives, whose solutions favor the desired surface properties of smoothness and fairness. Liu and Hu [54] shift each point in the direction of its normal vector within a strictly bounded width so as to keep geometric properties of the original point clouds. Transferability of attack method means the scope of application of one attack method; to this extent, Hamdi et al. [25] develop a point attack method by exploring the input data distribution and adding an adversarial loss.

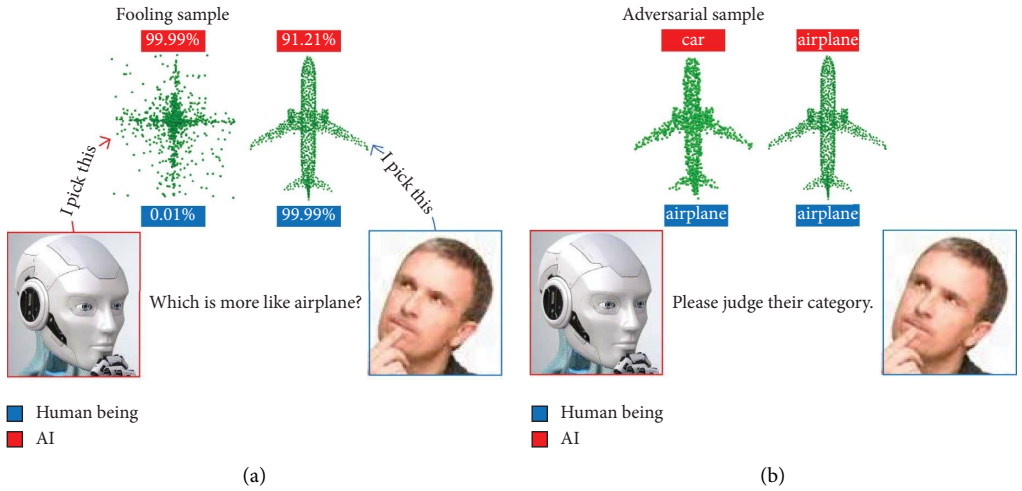

FIGURE 1: Illustration of the proposed fooling sample (a) and adversarial sample. (b) Adversarial sample has a slight difference from the benign sample but is able to mislead DNN to make a wrong decision. By contrast, the fooling sample is very different from the benign, but DNN believes it belongs to the target class with more than 99.99% confidence.

Compared with adversarial attack, there are only a few methods proposed for 3D point cloud adversarial defense. Adversarial training retrains DNN with adversarial samples which is an effective way to defend against an adversarial attack. Liu et al. [30] extend this defense method to 3D DNN and achieve promising results. Restoring the adversarial point cloud is another way to defend against an adversarial attack. Zhou et al. [31] design a denoiser network to remove the outlier points to defend the attacks. Meanwhile, the smoothness of the point cloud surface is guaranteed. Dong et al. [32] extracted the global feature to confuse the benign point cloud and the adversarial point cloud by the designed vector. Liang et al. [34] propose a novel perturbation adaptation generation network. This model can adaptively generate adversarial point clouds according to the victim point cloud. Different from improving the robustness of the whole point cloud, Wicker and Kwiatkowska [35] analyzed the pointwise robustness of 3D DNN in an adversarial setting.

As we can see, most attack methods achieve a nearly 100% success rate by crafting visually similar adversarial point cloud. The gap between 3D DNN and human vision is obvious. In this paper, we further explore such a gap by generating the fooling point cloud as the supplement of adversarial "attack-defense" of 3D DNN.

2.3. Evolutionary Algorithm. The evolutionary algorithm simulates the behavior of animals in nature and is a kind of typical global optimization technique which has dealt with lots of engineering problems [55, 56]. The initial evolutionary algorithm is inspired by the biology and the natural selection principle to optimize the problems. After that, many efficient evolutionary algorithms are designed, and they can be grouped in the three main categories which include swarm-based, evolution-based, and physics-based methods. In detail, swarm-based methods [38, 57–59] mimic the cooperative mechanism of group animals. Evolution-based methods [60–66] are inspired by Darwinian evolution law, which follows the saying: survival of the fittest and elimination of the unfit. Physics-based methods [56, 67–69] simulate the physical laws of how things work in the universe. Although the thriving of the evolutionary algorithm, it is impossible that one evolutionary algorithm is able to adapt to all optimization problems due to the unique characteristics of each problem. For the above point cloud generation problem, we design a new evolutionary algorithm named MEHHO based on group besiege and spiral-shaped attack to enhance the exploitation ability.

3. Method to Generate a Fooling Point Cloud

3.1. Problem Definition. A 3D point cloud $X \in R^{N*3}$ is defined as a set of N 3D points, where each point $x_i \in R^3$ is represented by its 3D coordinates (x_i, y_i, and z_i). Meanwhile, the 3D DNN is defined as $F: X \longrightarrow y$, which maps an input point cloud X to its corresponding class label $y \in Y$. The probability that X is classified as target class t is indicated by $P(t \mid X, \theta)$ with the well trained parameters θ.

For the ease of understanding, we first illustrate the process of an adversarial attack. It aims to mislead a 3D DNN F to classify an adversarial example X' as the target class t, by adding perturbations that are imperceptible to humans. Adversarial example shows the disparity of human and DNN when recognizing objects with similar appearance. The formal description is

$$\min D\left(X, X^{\text{adv}}\right) \quad \text{s.t.} F\left(X^{\text{adv}}\right) = t, \tag{1}$$

where $D(X, X^{\text{adv}})$ means the distance between adversarial point cloud X^{adv} and benign point cloud X. The common metrics include L_p normal, chamfer distance, and hausdorff distance.

Different from adversarial attack, we mainly explore the gap between appearance that human and 3D DNN think category t should look like. We first randomly generate

a point cloud as the initial, and then maximize its probability w.r.t. category t to generate fooling sample X^{fool}:

$$\max P\left(t \mid \theta, X^{\text{fool}}\right) \quad \text{s.t.} F\left(X^{\text{fool}}\right) = t. \tag{2}$$

Here, we utilize $\exp\left(F_t\left(X^{\text{fool}}\right)\right)/\sum_{i=1}^{|\mathbb{X}|}\exp F_i\left(X^{\text{fool}}\right))$ to quantify probability of category t, where $\mathbb{X}$ is the inference set. It is obvious that higher probability will better reflect the inside of 3D DNN when it recognizes category t. In the next section, we design a new evolutionary algorithm to solve equation (2). The generation process is as shown in Figure 2.

3.2. Harris Hawks Optimization (HHO). Given the number of points and target class, there are massive permutations to compose into a point cloud. We urge to find the optimal one with the highest confidence efficiently. Evolutionary algorithm (EA) [70] is a good kind of global optimization technique and has dealt with a wide variety of problems.

HHO is one of the most efficient EAs which mimic the hunting mechanism of harris hawks. It is simple and straightforward and can be divided into exploration phase and exploitation phase. HHO has the advantages of well the convergence rate, competitive exploration ability, and good time efficiency. Meanwhile, it has shown to be good at handling high-dimensional data despite its weak exploitation ability. In lots of practical engineering applications, HHO has been proven to be very effective and stable.

3.2.1. Exploration Phase. Since the rabbit (global optimum) is not easy to be found, the hawks wait, observe, and monitor the desert site by two ways, which as shown in the following equation:

$$X(t+1) = \begin{cases} X_{\text{rand}}(t) - r_1\left|X_{\text{rand}}(t) - 2r_2X(t)\right|, & q \geq 0.5, \\ \left(X_{\text{rabbit}}(t) - X_m(t)\right) - r_3\left(LB + r_4(UB - LB)\right), & q < 0.5, \end{cases} \tag{3}$$

where $X(t)$ is the position of hawk in t iteration, X_{rand} is the position of a random hawk in population, $X_m = (1/N)\sum_{i=1}^{N}(X_i)$ is the average position of the population, X_{rabbit} means the current optimal hawk, UB and LB means the upper bound and lower bound of variables, and r_1, r_2, r_3, r_4 and q are random numbers inside (0, 1).

3.2.2. Exploitation Phase. There are abundant search strategies in the exploitation phase. The hawks exchange search strategy according to the random number q and search energy $E = 2E_0(1 - t/T)$ with E_0 decreases from 0 to 1, t the current iteration and T the total iteration times.

(1) Soft Besiege. When $r \geq 0.5$ and $|E| \geq 0.5$, the rabbit is energetic and tries to escape by misleading jumps. To make the rabbit exhausted, the hawks encircle it softly by

$$\begin{aligned} X(t+1) &= \Delta X(t) - E\left|JX_{\text{rabbit}}(t) - X(t)\right|, \\ \Delta X(t) &= X_{\text{rabbit}}(t) - X(t), \end{aligned} \tag{4}$$

where $J = 2(1 - r_5)$ means the random jump strength of the rabbit throughout the escaping procedure and r_5 is a random number inside (0, 1).

(2) Hard Besiege. When $r \geq 0.5$ and $|E| < 0.5$ the rabbit is tired and the hawks attack it by hardly encircle:

$$X(t+1) = X_{\text{rabbit}}(t) - E|\Delta X(t)|. \tag{5}$$

(3) Soft Besiege with Progressive Rapid Dives. When $r < 0.5$ and $|E| \geq 0.5$, the hawks progressively select the best possible dive to attack the rabbit. The new site of hawks is calculated by

$$X(t+1) = \begin{cases} Y, & if F(Y) < F(X(t)), \\ Z, & if F(Z) < F(X(t)), \end{cases} \tag{6}$$

where

$$\begin{aligned} Y &= X_{\text{rabbit}}(t) - E\left|JX_{\text{rabbit}}(t) - X(t)\right|, \\ Z &= Y + S \cdot \text{LF}(D). \end{aligned} \tag{7}$$

here S is a random vector and $\text{LF}(\cdot)$ means levy flight function which can be computed as follows:

$$\text{LF}(x) = 0.01 \cdot \alpha \cdot \beta/|\nu|^{1/\gamma}, \tag{8}$$

where α, ν are two random number, and β is obtained by

$$\beta = \left(\frac{\Gamma(1+\gamma) \cdot \sin(\pi\gamma/2)}{\Gamma(1+\gamma/2) \cdot \gamma \cdot 2^{(\gamma-1/2)}}\right)^{1/\gamma}. \tag{9}$$

where γ is a constant number which equal to 1.5.

(4) Hard Besiege with Progressive Rapid Dives. When $r < 0.5$ and $|E| < 0.5$, the rabbit is exhausted and be trapped. The hawks dive and kill the rabbit:

$$X(t+1) = \begin{cases} Y, & if F(Y) < F(X(t)), \\ Z, & if F(Z) < F(X(t)), \end{cases} \tag{10}$$

where Y and Z are obtained by

$$\begin{aligned} Y &= X_{\text{rabbit}}(t) - E\left|JX_{\text{rabbit}}(t) - X_m(t)\right|, \\ Z &= Y + S \cdot LF(D). \end{aligned} \tag{11}$$

3.3. Multielites HHO. Our purpose is to excavate the inside of 3D DNN by generating a fooling point cloud which 3D DNN sincerely believes belongs to the target class. Therefore, we push the initial point cloud away from the decision

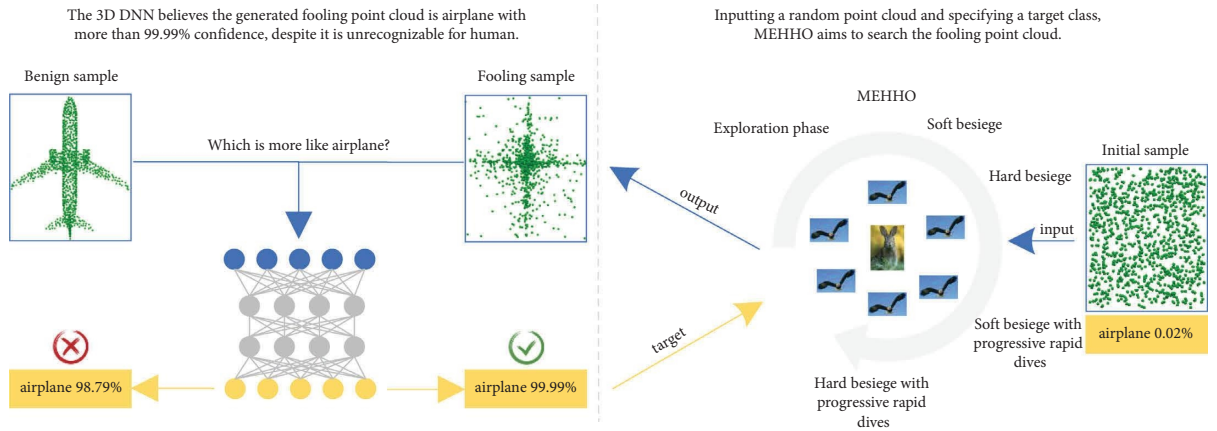

FIGURE 2: The framework to generate a fooling point cloud. Inputting a random point cloud to MEHHO and specifying a target class, MEHHO will generate a point cloud that can hardly be recognized by human vision. But the victim 3D DNN will classify it to the target class with more than 99.99% confidence which is higher than the benign point cloud.

boundary as far as possible. That is, we search for fooling point cloud $X^{'}$ whose probability $P(t^{'} \mid \theta, X^{'}) \longrightarrow 100\%$. Thus, it requires an evolutionary algorithm with excellent exploitation ability.

Though HHO boosts the exploitation behavior through abundant searching strategies and different Levy flight-based patterns, it is hard to impend over 100% due to several local optimum of 3D DNN. Thus, we propose the multielites HHO (MEHHO) with better exploitation ability by the designed group besiege and spiral-shaped attack. Its pseudocode is shown as Algorithm 1.

3.3.1. Group Besiege. In HHO, the rabbit in each iteration is decided by one elite hawk who has the best position as most evolutionary algorithms do. Although this mechanism accelerates the convergence around the current optimum, it will move along zigzag when close to the optimum position and will increase time cost. Therefore, we propose the group besiege stage which utilizes the top k best individual to represent X_{rabbit}. By this way, the hawks will search for the rabbit based on abundant information from multielites:

$$X_{\text{rabbit}}(t) = \frac{X_1^* + X_2^* + \cdots + X_k^*}{k}, \tag{12}$$

where $X_1^*, X_2^*, \cdots, X_k^*$ are the hawks with the top k best positions.

3.3.2. Spiral-Shaped Attack. Logarithmic spiral is a well-known attack stage in nature, and thus many mathematical optimization techniques based on logarithmic spirals are designed, such as WOA and MFO. Here, we design a prey factor m to leverage the movement toward the rabbit and the jump combined of HHO with logarithmic spiral. Therefore, the hawk position Y in stage soft besiege with progressive rapid dives and stage hard besiege with progressive rapid dives is reformulated by equation (13), as shown in Figure 3.

$$Y = (1 - m) \cdot X_{\text{rabbit}} - m\left(E\left|X_{\text{rabbit}} - X(t)\right|\right) \cdot e^{\text{bl}} \cdot \cos(2\pi l), \tag{13}$$

where m is the prey factor which randomly selected in $[0.3, 0.9]$, l is a random number in $[-1, 1]$, b is a constant number which is set to 1.

3.4. Generate a Fooling Point Cloud by MEHHO. The generation process for fooling point cloud is modeled as a maximization problem. Then, we solve the maximization problem by MEHHO. In detail, each hawk of one population is represented by one point cloud. The fitness of one hawk is calculated by equation (2). During evolution, to limit the search space, we constrain the points of the point cloud in a bounding box. Besides, the initial population is composed by the point clouds with random points in the bounding box.

4. Results and Analysis

We conduct experiments to answer the following three questions: (1) whether fooling samples can be found? (Subsection 4.2) (2) what characteristic of 3D DNN does a fooling sample reveals? (Subsection 4.3) (3) does the improvement of MEHHO work in generating fooling samples? (Subsection 4.4).

4.1. 3D DNNs and Datasets

4.1.1. 3D DNNs. The results are mainly obtained on PointNet [6]. Besides, PointNet++ [7], DGCNN [48], and RS-CNN [50] are also selected to prove the transferability. They are typical works of pointwise MLP methods, convolution-based methods, and graph-based methods, respectively [11]. In the experiments, we follow their default setting for fairness.

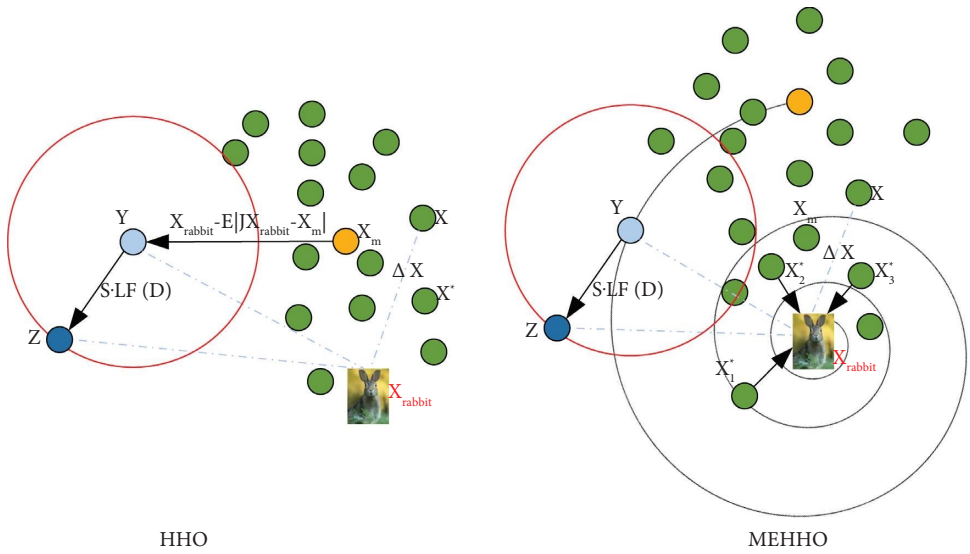

Figure 3: The stage of soft besiege with progressive rapid dives and stage of hard besiege with progressive rapid dives in MEHHO.

Initialize the population;
Calculate the fitness of each hawk in $\mathcal{X}$;
Find top k best hawks $X_1^*, X_2^*, \cdots, X_k^*$;
while $t < T$ **do**
  Obtain $X_{\text{rabbit}}(t) = X_1^* + X_2^* + \cdots + X_k^*/k$.
  for $i = 1$ to $|\mathcal{X}|$ **do**
    for $j = 1$ to N_s **do**
      Update $E = 2E_0(1 - t/T)$
      if $|E| \geq 1$ **then**
        update $X(t+1)$ by equation (3)
      else if $|E| < 1$ **then**
        if $r \geq 0.5$ and $|E| \geq 0.5$ **then**
          $X(t+1) = \Delta X(t) - E|JX_{\text{rabbit}}(t) - X(t)|$
        else if $r \geq 0.5$ and $|E| < 0.5$ **then**
          $X(t+1) = X_{\text{rabbit}}(t) - E|\Delta X(t)|$
        else if $r < 0.5$ and $|E| \geq 0.5$ **then**
          if $F(Y) < F(X(t))$ **then**
            update $X(t+1)$ by equation (13)
          else if $F(Y) < F(X(t))$ **then**
            $X(t+1) = Y + S \cdot \text{LF}(D)$
          else if $r < 0.5$ and $|E| < 0.5$ **then**
            if $F(Y) < F(X(t))$ **then**
            update $X(t+1)$ by equation (13)
          else if $F(Y) < F(X(t))$ **then**
            $X(t+1) = Y + S \cdot \text{LF}(D)$
    end
    Update X_1^*, X_2^*, X_3^*
    $t+=1$
  end
end

Algorithm 1: Pseudo-code of MEHHO.

4.1.2. Datasets. The experiments are conducted on 3D Minist and ModelNet40, respectively [46], which are widely used in adversarial attack experiments [21–23, 25, 27, 28, 52, 53]. In detail, there are 12,311 objects from 40 categories in ModleNet40, where 9,843 are used for training and the other 2,468 for testing. 3D Mnist includes 6000 3D handwritten digits from 10 categories, of which 5,000 are used for training and 1,000 for testing. For convenience of comparison, we uniformly sample 1,024 points from each object and rescale them into a unit cube.

4.2. Generate a Fooling Point Cloud

4.2.1. Fooling Point Clouds on 3D Minist. To evaluate whether the fooling point clouds able to fool 3D DNN, we first generate fooling point clouds on 3D Mnist by PointNet, as shown in Figure 4. The results show that the proposed MEHHO can find a fooling point cloud for all classes of 3D Mnist. At the same time, all fooling point clouds are hardly recognized by human vision, but PointNet labels them as the target class with more than 99.99% confidence which are higher than the benign point cloud.

Although the fooling point cloud is unrecognizable by human vision, the fooling point clouds do contain a similar outline to the victim class. For example, the fooling point clouds w.r.t. class 0 tend to appear an oval outline. Fooling point clouds classified to 3, 5, 6, and 9 all tend to exhibit outline like the benign digit class. Such a result suggests that MEHHO explores the global feature that PointNet used to recognize a point cloud and utilizes the explored feature to generate fooling point clouds.

4.2.2. Fooling Point Clouds on ModelNet40. Furthermore, to verify whether PointNet is overfitting on 3D Mnist, we generate fooling point clouds on ModelNet40, which has bigger training data and more categories. The generated fooling point clouds are shown in Figure 5. The result suggests that it is feasible to generate fooling point clouds on all 40 classes. In detail, 32 classes of them only need 100 iterations, and the rest 8 classes need more than 1000 iterations and a larger population. The hypothesis is that MEHHO stuck in the local optimum. In sum, Figures 4 and 5 illustrate that MEHHO has the ability to fool PointNet with 100% success rate on both 3D Mnist and ModelNet40.

4.2.3. Generate Fooling Samples by Other 3D DNNs. In this section, we further generate fooling samples on other 3D DNNs. The chosen 3D DNNs are PointNet++, DGCNN, and RS-CNN. The result illustrates that MEHHO can generate point clouds that fool PointNet++, DGCNN, and RS-CNN on all 40 classes of ModelNet40. Portion results are shown in Figure 6.

4.3. Further Analysis of Fooling Samples. For each fooling sample, the confidence of the victim 3D DNN on it is 100%. In this section, we explore characteristics of 3D DNN by analyzing the feature of fooling point clouds.

4.3.1. Influence of Iteration Times. We first visualize the process to generate fooling point clouds in Figure 7. The result suggests that the confidence of the fooling point cloud goes up obviously, and the outline gradually approaches to the benign point cloud at the same time, according to the increase in iterations t. Meanwhile, we find that the points of fooling point cloud tend to gather to the center of the bounding box. But the benign point cloud whose points are evenly distributed in the bounding box will slow the gathering tendency. The phenomenon suggests that the global structure mainly guides the generation of the fooling point cloud.

To further explore the influence of iterations times T, we compare fooling point clouds from different T as shown in Figure 8. The result indicates that the points in the point cloud are more concentrated toward the center, and their outline change slightly with the increase of T which further exhibits the importance of the global feature.

4.3.2. Which Points Are Matter in the Fooling Point Cloud? We find that lots of points in the fooling point cloud tend to gather toward the center rather than the surface. However, psychophysical evidence shows that humans perceive 3D object shapes from combined sources of boundary contour, motion, texture, and shading [71]. For human vision, the gathered points are redundant to represent object shape. When PointNet makes decisions, it does not regard each point equally. To evaluate whether the gathered points are crucial for PointNet, we visualize point cloud by the saliency map technology which aims to evaluate point-wise importance [22]. The saliency map of fooling point clouds and the corresponding benign point cloud are shown in Figure 9. The results show that the gathered points contain proportional high score points which exhibit their importance for PointNet.

At the same time, Zheng et al. [22] reduce the accuracy of PointNet from 89.20% to 44.30% by dropping 200 high score points. For comparison, we drop the same number of high-score points of the fooling point cloud, causing the accuracy to reduce to 21.05%. The phenomenon illustrates that the fooling point cloud is more fragile in spite of its high confidence. The possible reason is that PointNet makes decisions based on fewer points of fooling point cloud than the benign. In sum, the result suggests that more points a 3D DNN relies on when making decisions, the harder it is for 3D DNN to be fooled.

4.3.3. What Type of Feature PointNet Learns? To explore what feature the PointNet learns, we generate several fooling point clouds from independent runs which are shown as Figure 10. The basis is that the generated fooling point cloud is what PointNet thinks one class should be. In Figure 10, we observe that the fooling point clouds belonging to the same class are similar to the benign. Specifically, for class "vase," the five fooling point clouds all exhibit narrow bottleneck, base, and a wide body. For class "tent," the five fooling point clouds all exhibit the shape of a triangle like the victim point cloud. The result exhibits the similarity between human vision and PointNet. It is infeasible to generate objects recognizable by the human vision only through the evolutionary algorithm in the 2D image domain unless adding specific constraints [72].

These results suggest that PointNet tends to learn high-level features rather than low- or middle-level feature. If PointNet properly learns low- or middle-level features, the generated fooling point cloud should contain repetitions of object subcomponents [72] such as bottleneck.

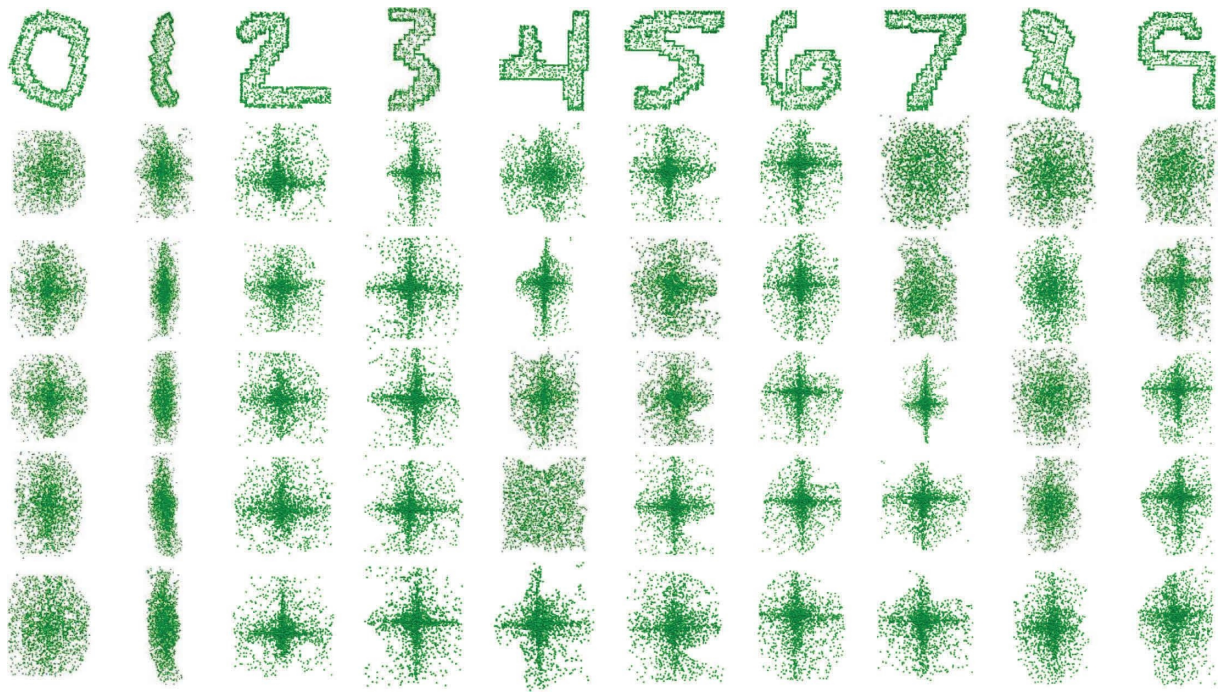

Figure 4: The generated fooling point clouds on 3D Mnist. Each column denotes the same digit class. The first row is the benign point clouds and the rest rows are fooling point clouds generated by independent runs. MEHHO explores the main feature that PointNet uses to recognize a point cloud. Therefore, most fooling point clouds exhibit a similar outline to the corresponding benign point cloud.

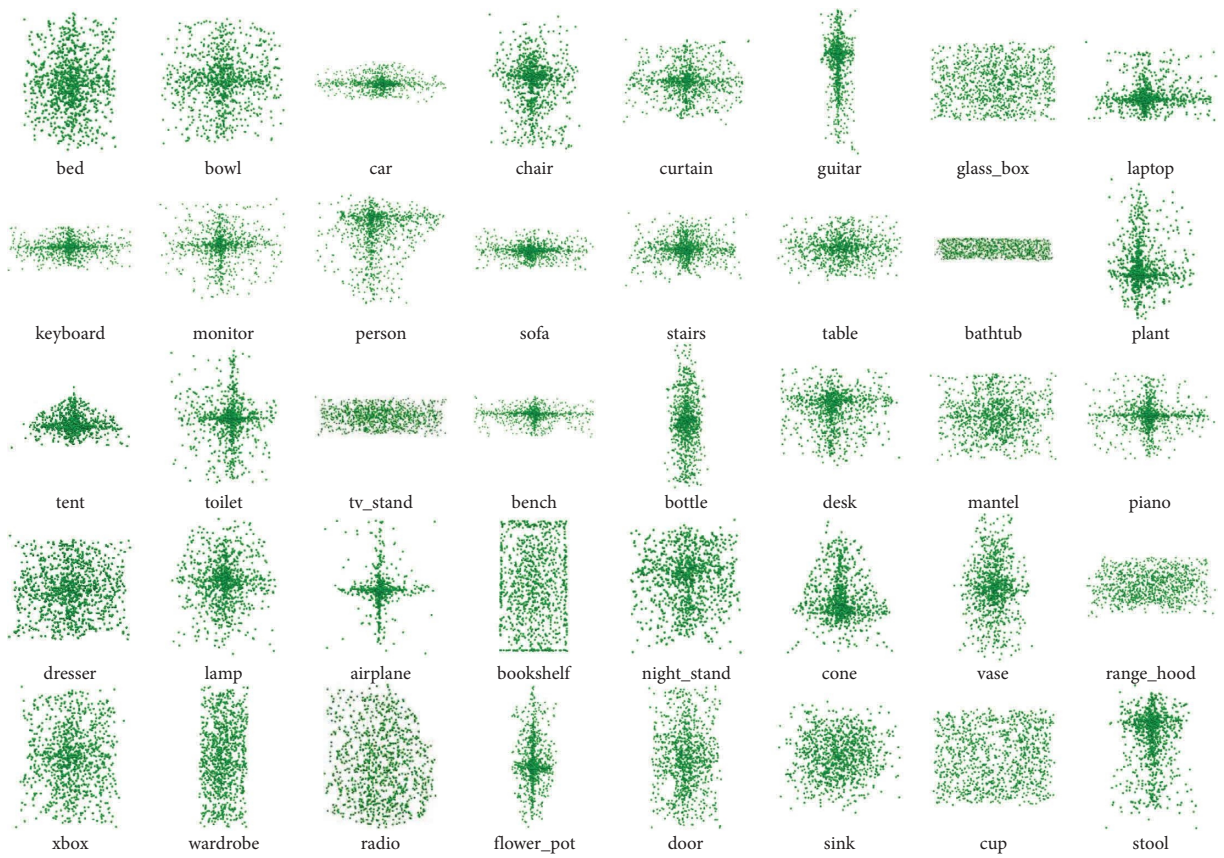

Figure 5: The generated fooling point clouds on ModelNet40. PointNet believes that they should be classified as the target class with more than 99.99% confidence.

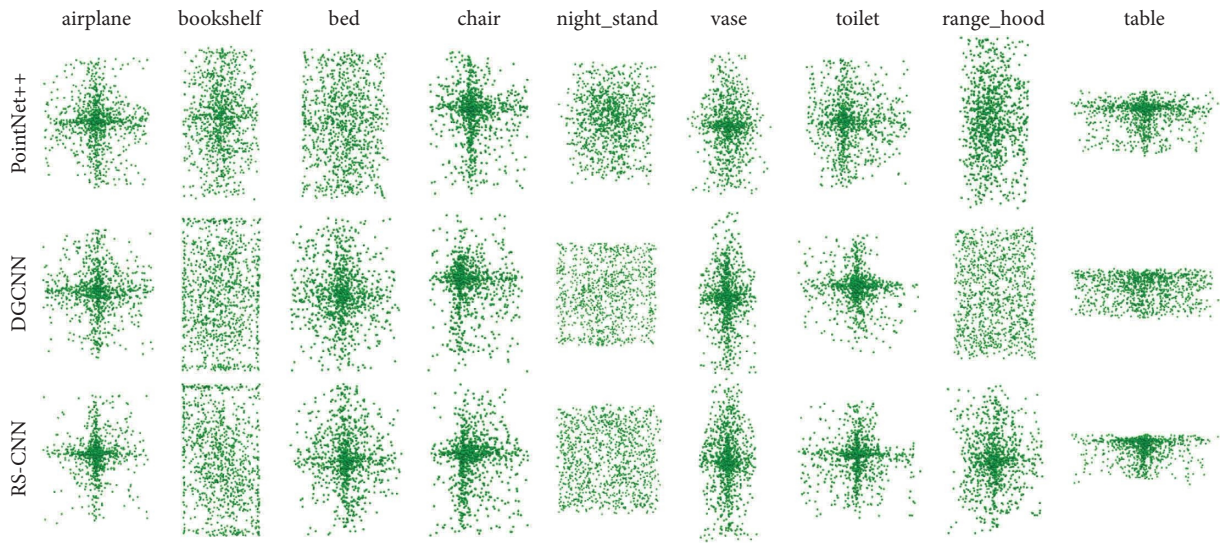

Figure 6: The generated fooling point clouds by PointNet++, DGCNN, and RS-CNN. Each fooling point cloud achieves more than 99.99% confidence on corresponding 3D DNN.

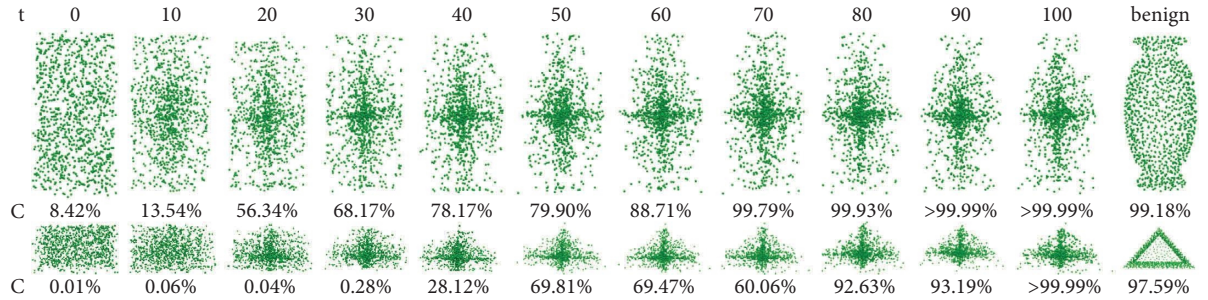

Figure 7: The process to generate the fooling point clouds, t means iteration steps and C means confidence of generated point cloud. Result suggests that the global structure mainly guides the generation of fooling point cloud.

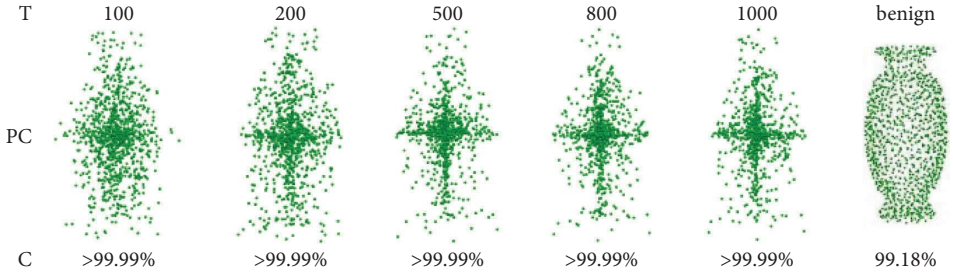

Figure 8: Fooling point clouds from five independent runs with different iterations times T. Their outlines are similar to the benign, and more points gather to the center as iteration number increases.

4.3.4. Features Extracted by Different 3D DNN. Discussing the similarity of features extracted from different 3D DNNs is helpful to understand the disparity between different 3D DNNs. In this section, we feed the fooling point clouds generated by one DNN to other DNNs and evaluate the inference accuracy. The chosen 3D DNNs are PointNet, PointNet++, and DGCNN.

In Table 1, we find that PointNet and PointNet++ both able to recognize the fooling point clouds generated by each other more efficiently than DGCNN. The possible reason is that the two 3D DNNs tend to extract similar high-level features due to their similar structures.

Comparing each column of Table 1, we find that PointNet and PointNet++ both achieve higher test accuracy than DGCNN, which means that DGCNN is harder to be fooled than PointNet and PointNet++. Furthermore, it is reasonable to say that the features extracted from one class by one 3D DNN are not unique, since one 3D DNN is

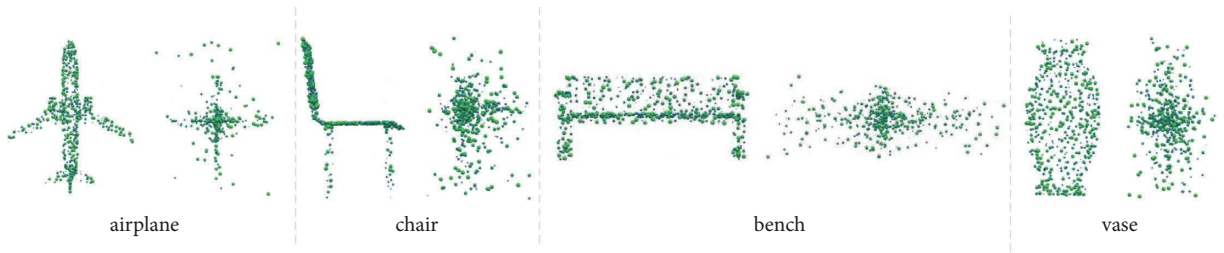

Figure 9: Saliency map of point cloud. The more important the point is, the more big and green the point will be. The result shows that the gathered points are useful for PointNet, while they are redundant for human vision.

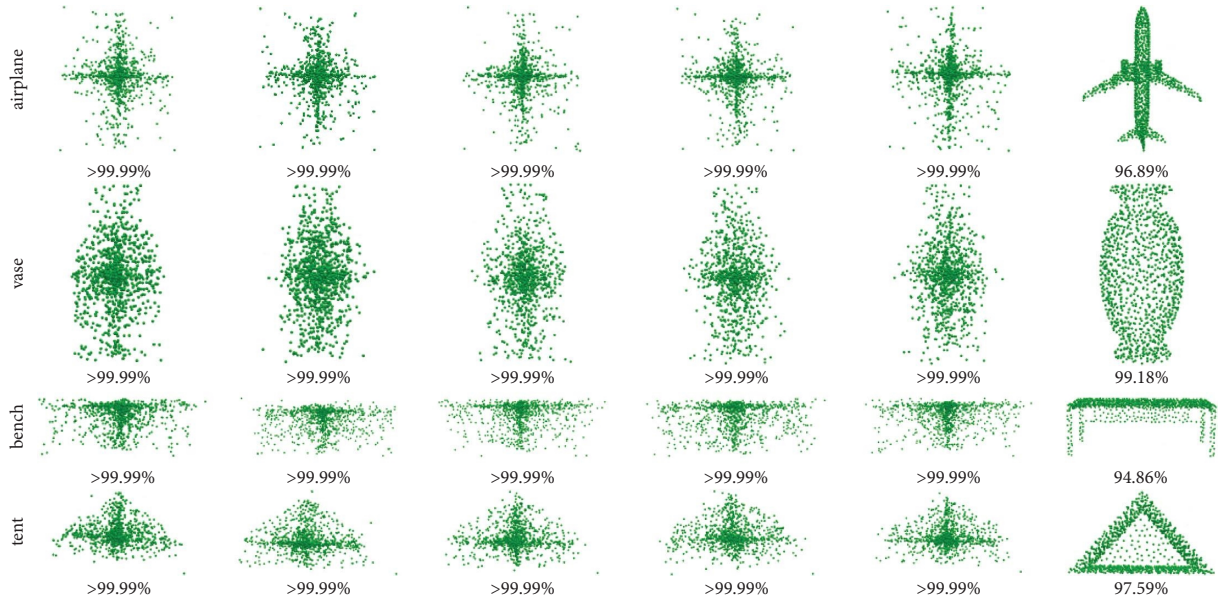

Figure 10: The generated fooling point clouds and their confidence from five independent runs of MEHHO. 3D DNN tends to learn high-level feature rather than low- or middle-level feature.

Table 1: The accuracy of the 3D DNN tested on fooling point clouds generated by another 3D DNN. For example, 30.56% is the accuracy of PointNet++ tested on fooling point clouds generated by PointNet. We find that DGCNN is harder to fool than PointNet and PointNet++. Otherwise, PointNet and PointNet++ tend to extract similar high-level feature due to similar network structures, which illustrates that 3D DNN can be fooled in different ways.

	Pointnet	Pointnet++	DGCNN
Pointnet	100%	30.56%	5.56%
Pointnet++	37.93%	100%	3.45%
DGCNN	10.71%	17.86%	100%

vulnerable to fooling samples generated by another 3D DNN. Thus, we can draw the conclusion that 3D DNN can be fooled in different ways.

To verify whether the same 3D DNN from independent training learns discriminative features, we separately train two PointNets named PointNet_1 and PointNet_2. The two PointNets have the same network structure and are trained on the same dataset, but the initialization parameters are different. We then test the accuracy of each PointNet on the fooling point clouds generated by another PointNet. The results are shown in Table 2 which illustrates that the features extracted by the same 3D DNN are similar to a large extent. The difference may come from the randomness of independent training.

4.3.5. Try to Eliminate Fooling Sample. To explore the defense against fooling samples, we regard the fooling point clouds as a new class and add them to the training dataset as an "$n+1$" class. After that, we retrain PointNet using the created dataset to find whether PointNet is able to recognize the fooling samples. The result shows that the retrained 3D DNN can recognize the fooling point clouds with accuracy of 96.88%.

Table 2: The accuracy of 3D DNN tested on fooling point clouds generated by the same 3D DNN that from independent training. Features learned by the same 3D DNN are similar to a large extent.

	Pointnet_1	Pointnet_2
Pointnet_1	100%	89.86%
Pointnet_2	88.75%	100%

Table 3: Parameter settings for evolutionary algorithms.

Algorithms	Parameters	Values
MEHHO	$r_1, r_2, r_3, r_4, r_5, q; \gamma, k, m$	Random in (0, 1); 1.5; 3; random in [0.3, 0.9]
WOA	$a; r; l$	Liner reduction from 2 to 0; random in [0, 1]; random in [−1, 1]
BWO	$m; p$	Random in [0.4, 0.9]; 0.3
SSA	$a; ST$	Random in (0, 1]; random in [0.5, 1]
SO	$c_1; c_2; c_3$	0.5; 0.05; 2
I-GWO	$a; r1, r2$	Liner reduction from 2 to 0, random in [0, 1]
HHO	$r_1, r_2, r_3, r_4, r_5, q; \gamma$	Random in (0, 1); 1.5

Table 4: The probability of fooling point cloud generated by seven evolutionary algorithms, bigger is better. Max, avg, and min means maximum value, average value, and minimum value obtained by running the algorithm five times, respectively.

Model	Value	WOA	BWO	SSA	SO	I-GWO	HHO	MEHHO
Bathtub	Max	$3.38E-05$	***0.9998***	0.0092	*0.9956*	0.9923	0.0003	**0.9999**
	Avg	$3.31E-05$	*0.9579*	0.0030	***0.9937***	0.0001	0.0003	**0.9999**
	min	$2.84E-05$	**0.9999**	$1.63E-08$	*0.9868*	0.2093	0.0003	***0.9992***
Bed	Max	***0.9998***	*0.9997*	0.9772	0.9689	0.9499	0.9991	**0.9999**
	Avg	0.6915	0.9779	0.9625	***0.9996***	*0.9987*	0.9985	**0.9999**
	min	0.4911	0.9849	0.5453	***0.9995***	0.9889	*0.9976*	**0.9999**
Bench	Max	0.20	0.9949	0.9922	0.9984	***0.9994***	*0.9989*	**0.9999**
	Avg	0.1554	***0.9991***	0.3178	0.9942	*0.9984*	0.8868	**0.9999**
	min	0.1200	0.9998	0.0508	0.9933	*0.9949*	0.8293	**0.9999**
Bookshelf	Max	0.7559	*0.9932*	0.9888	***0.9980***	0.9886	0.9621	**0.9999**
	Avg	0.6654	***0.9981***	0.4457	*0.9968*	0.969170	0.8501	**0.9999**
	min	0.6069	*0.9886*	0.2135	***0.9958***	0.8389	0.7031	**0.9999**
Bottle	Max	0.2946	*0.9978*	0.9959	0.9649	**0.9999**	***0.9999***	0.9968
	Avg	0.2295	*0.9976*	0.8927	***0.9986***	**0.9989**	0.9910	0.9942
	min	0.0280	**0.9987**	0.5923	0.9901	***0.9978***	0.9891	*0.9935*
Bowl	Max	0.5162	0.9779	0.2730	0.9834	*0.9989*	***0.9999***	**0.9999**
	Avg	0.0129	*0.9699*	0.2684	0.9041	***0.9709***	0.9549	**0.9999**
	min	0.0075	*0.9989*	0.0002	0.8415	**0.9999**	0.9136	***0.9990***

The bold number means the best result. Besides, the number with bold and italics and italics represent the second and third best value, respectively.

However, we find that MEHHO can explore new fooling point clouds to fool the retrained PointNet even after ten retrainings. This phenomenon indicates that the difference between PointNet and human vision is hardly learned.

4.4. Evaluation of MEHHO

4.4.1. Comparison Evolutionary Algorithms. To evaluate the advantages of MEHHO w.r.t. searching for fooling point cloud, we compare the statistical results and the convergence results with WOA [73], BWO [74], SSA [75], SO [76], I-GWO [77], and HHO [37]. The victim 3D DNN and dataset is PointNet and ModelNet40, respectively.

4.4.2. Parameters Setting. We set the iteration times $T = 50$ and the population size to 100. The hyper-parameters of each evolutionary algorithm are shown in Table 3.

Since the softmax layer always assigns probability to all classes, the probability of the target class will not achieve to 100%, but will close to 100% as possible. The results in Table 4 suggest that the generated fooling point cloud of MEHHO is closer to 100% than other evolutionary algorithms. In detail, WOA always fails to generate fooling point cloud since it falls into the local optimum, when attacking 'bathtub'. For model *bed*, *bench*, *bookshelf*, MEHHO is able to generate fooling point cloud with more than 99.99% confidence. For model *bottle* and *bowl*, I-GWO is the better one to generate fooling samples, but it is inferior to MEHHO on other models.

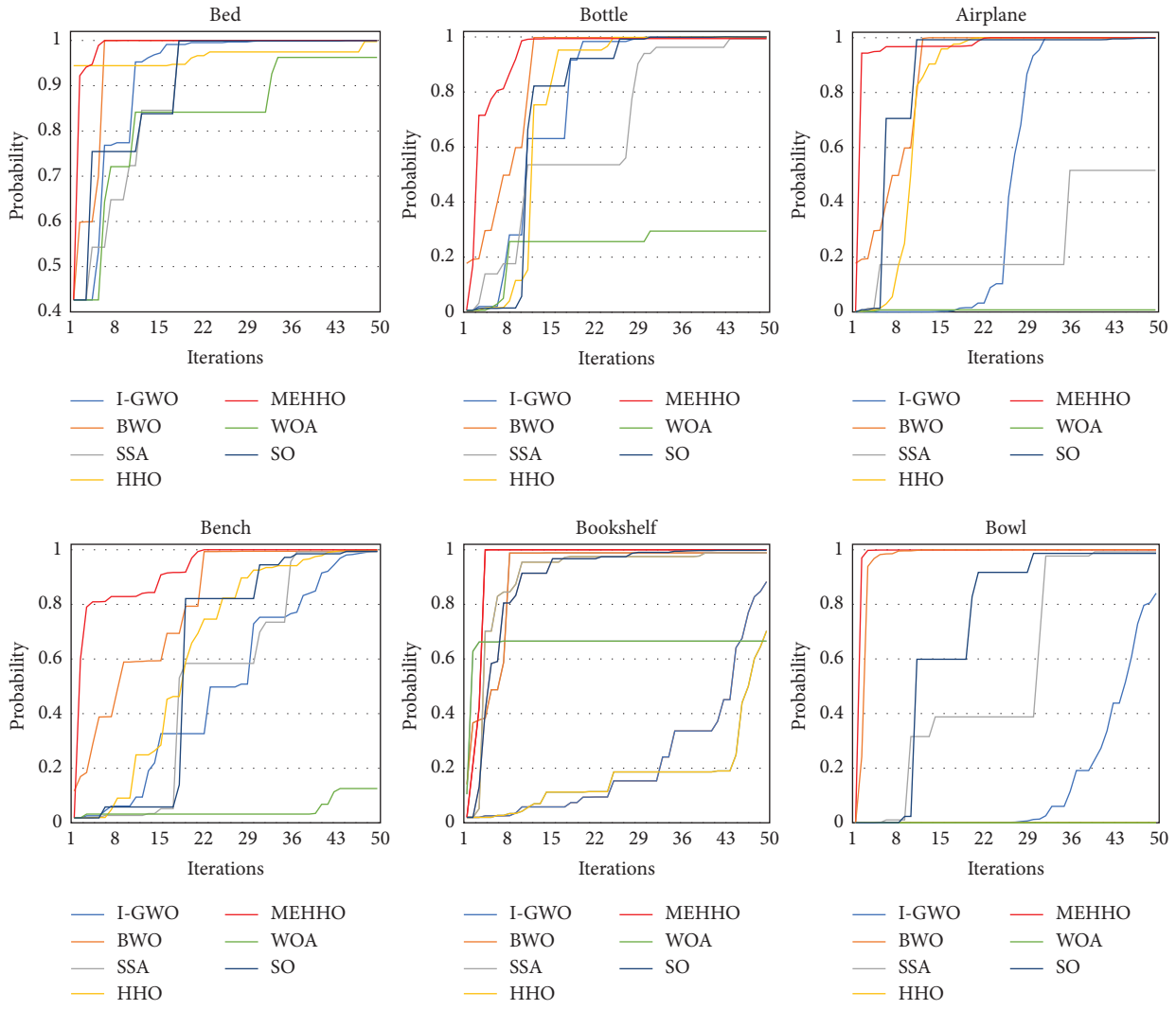

Figure 11: Convergence results of the comparison algorithms. The closer the curve is to the upper left, the better. MEHHO achieves the best convergence performance.

Figure 11 shows the convergence curves on classes *bed*, *bottle*, *airplane*, *bench*, *bookshelf*, and *bowl*. The closer the curve is to the upper left, the better. For the model *bench*, the probability of MEHHO increases rapidly when $t < 8$ and converges when $t > 18$. Meanwhile, we argue that the model *bed* is easily fooled since the probability of all evolutionary algorithms increases at the early stage of iteration and all achieve probabilities greater than 90%. Furthermore, for model *Bookshelf* and *Bottle*, MEHHO is able to generate fooling point cloud using the minimum iteration times. Therefore, the results suggest that MEHHO outperforms other algorithms w.r.t. convergence performance.

5. Conclusions and Future Works

To research the difference between 3D DNN and human vision, we generate a fooling point cloud using MEHHO. The fooling point cloud is unrecognizable to humans but is classified as target class by 3D DNN firmly. We analyze the characteristic of 3D DNN by analyzing features of the fooling point cloud and give the following main conclusions: (1) 3D DNN tends to learn high-level feature of one object; (2) 3D DNN that makes decision relying on more points is more robust; (3) The gap is hardly learned by 3D DNN. In the end, we evaluated the proposed MEHHO on several models, and the results suggest that MEHHO achieves better statistics and convergence results.

In the future, we will extend the research from the following two aspects: First, explore the feature extracted by each layer of 3D DNN to learn more about it. Second, some generated fooling point clouds exhibit similar shapes as the victims in our experiments. Thus, it is meaningful to study whether such a method can generate a point cloud that conforms to human vision.

Data Availability

The ModelNet40 is a public dataset widely used for academic research.

Conflicts of Interest

The authors declare that they have no conflicts of interests.

Authors' Contributions

Linkun Fan, Xiaoxin Gao, and Jinkun Luo contributed equally to this work.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant no. 62072348, the Science and Technology Major Project of Hubei Province (Next-Generation AI Technologies) under Grant no. 2019AEA170 and China Yunnan Province Major Science and Technology Special Plan Project no. 202202AF080004. The numerical calculations in this paper have been done on the supercomputing system in the Supercomputing Center of Wuhan University.

References

[1] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," *International Conference on Machine Learning*, vol. 37, pp. 448–456, 2015.

[2] J. Zhang, F. He, Y. Duan, and S. Yang, "AIDEDNet anti-interference and detail enhancement dehazing," *Frontiers of Computer Science*, vol. 17, no. 2, Article ID 172703, 2023.

[3] B. Zoph, V. Vasudevan, and J. Shlens, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8697–8710, Salt Lake City, UT, USA, June 2018.

[4] M. Tan and Q. Le, "Efficientnet: rethinking model scaling for convolutional neural networks," in *Proceedings of the International Conference on Machine Learning*, pp. 6105–6114, Long Beach, CA, USA, June 2019.

[5] W. Tang, F. He, Y. Liu, and Y. Duan, "MATR: multimodal medical image fusion via multiscale adaptive transformer," *IEEE Transactions on Image Processing*, vol. 31, pp. 5134–5149, 2022.

[6] C. R. Qi, H. Su, and K. Mo, "Pointnet: deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 652–660, Honolulu, HI, USA, July 2017.

[7] C. R. Qi, L. Yi, and H. Su, "Pointnet++: deep hierarchical feature learning on point sets in a metric space," in *Proceedings of the 31st Conference on Neural Information Processing Systems*, Long Beach, CA, USA, December 2017.

[8] Y. Liang, F. He, X. Zeng, and J. Luo, "An improved Loop subdivision to coordinate the smoothness and the number of faces via multi-objective optimization," *Integrated Computer-Aided Engineering*, vol. 29, no. 1, pp. 23–41, 2021.

[9] Y. Song, F. He, Y. Duan, T. Si, and J. Bai, "LSLPCT: an enhanced local semantic learning transformer for 3D point cloud analysis," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–13, 2022.

[10] X. Yu, L. Tang, and Y. Rao, "Point-BERT: pre-training 3D point cloud transformers with masked point modeling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, New Orleans, LA, USA, June 2022.

[11] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 12, pp. 4338–4364, 2021.

[12] C. Szegedy, W. Zaremba, and I. Sutskever, "Intriguing properties of neural networks," in *Proceedings of the International Conference on Learning Representations*, New Orleans, LA, USA, June 2014.

[13] J. Zhang, B. Li, and J. Xu, "Towards efficient data free black-box adversarial attack," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15115–15125, New Orleans, LA, USA, June 2022.

[14] H. Wu, F. He, Y. Duan, and X. Yan, "Perceptual metric-guided human image generation," *Integrated Computer-Aided Engineering*, vol. 29, no. 2, pp. 141–151, 2022.

[15] S. Chen, Z. He, and C. Sun, "Universal adversarial attack on attention and the resulting dataset damagenet," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, 2020.

[16] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 39–57, San Jose, CA, USA, May 2017.

[17] M. Zhou, Z. Niu, and L. Wang, "Adversarial ranking attack and defense," in *Proceedings of the European Conference on Computer Vision*, pp. 781–799, Glasgow, UK, August 2020.

[18] Y. Jiang and D. Ye, "Black-box adversarial attacks against audio forensics models," *Security and Communication Networks*, vol. 2022, Article ID 6410478, 8 pages, 2022.

[19] H. Yin, H. Zhang, and J. Wang, "Boosting adversarial attacks on neural networks with better optimizer," *Security and Communication Networks*, vol. 2021, Article ID 9983309, 9 pages, 2021.

[20] A. N. Bhagoji, W. He, and B. Li, "Practical black-box attacks on deep neural networks using efficient query mechanisms," in *Proceedings of the European Conference on Computer Vision*, pp. 154–169, Munich, Germany, September 2018.

[21] C. Xiang, C. R. Qi, and B. Li, "Generating 3d adversarial point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9136–9144, Long Beach, CA, USA, June 2019.

[22] T. Zheng, C. Chen, and J. Yuan, "Pointcloud saliency maps," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1598–1606, Long Beach, CA, USA, June 2019.

[23] Y. Zhao, Y. Wu, and C. Chen, "On isometry robustness of deep 3d point cloud models under adversarial attacks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1201–1210, Seattle, WA, USA, June 2020.

[24] H. Xu, F. He, L. Fan, and J. Bai, "D3AdvM: a direct 3D adversarial sample attack inside mesh data," *Computer Aided Geometric Design*, vol. 97, Article ID 102122, 2022.

[25] A. Hamdi, S. Rojas, and A. Thabet, "Advpc: transferable adversarial perturbations on 3d point clouds," in *Proceedings of the European Conference on Computer Vision*, pp. 241–257, Glasgow, UK, August 2021.

[26] J. Mao, B. Weng, and T. Huang, "Research on multimodality face antispoofing model based on adversarial attacks. Security and communication networks," *Security and Communication Networks*, vol. 2021, Article ID 3670339, 12 pages, 2021.

[27] X. Wang, M. Cai, F. Sohel, N. Sang, and Z. Chang, "Adversarial point cloud perturbations against 3D object

detection in autonomous driving systems," *Neurocomputing*, vol. 466, pp. 27–36, 2021.

[28] J. Yang, Q. Zhang, and R. Fang, "Adversarial attack and defense on point sets," 2019, https://arxiv.org/abs/1902.10899.

[29] Y. Zhang, G. Liang, and T. Salem, "Defense-pointnet: protecting pointnet against adversarial attacks," in *Proceedings of the 2019 IEEE International Conference on Big Data*, pp. 5654–5660, Los Angeles, CA, USA, December 2019.

[30] D. Liu, R. Yu, and H. Su, "Extending adversarial attacks and defenses to deep 3d point cloud classifiers," in *Proceedings of the 2019 IEEE International Conference on Image Processing*, pp. 2279–2283, Taipei, Taiwan, September 2019.

[31] H. Zhou, K. Chen, and W. Zhang, "Dup-net: denoiser and upsampler network for 3d adversarial point clouds defense," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1961–1970, Seoul, Korea (South), October 2019.

[32] X. Dong, D. Chen, and H. Zhou, "Self-robust 3d point recognition via gather-vector guidance," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11513–11521, Seattle, WA, USA, June 2020.

[33] F. Yu, L. Wang, and X. Fang, "The defense of adversarial example with conditional generative adversarial networks," *Security and Communication Networks*, vol. 2022, Article ID 3932584, 12 pages, 2020.

[34] Q. Liang, Q. Li, and W. Nie, "PAGN: perturbation adaption generation network for point cloud adversarial defense," *Multimedia Systems*, vol. 28, pp. 1–9, 2022.

[35] M. Wicker and M. Kwiatkowska, "Robustness of 3d deep learning in an adversarial setting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11767–11775, Long Beach, CA, USA, June 2019.

[36] L. Jiang, K. Qiao, and R. Qin, "Cycle-Consistent Adversarial GAN: the integration of adversarial attack and defense," *Security and Communication Networks*, vol. 2020, Article ID 3608173, 9 pages, 2020.

[37] A. A. Heidari, S. Mirjalili, and H. Faris, "Harris hawks optimization: algorithm and applications," *Future Generation Computer Systems*, vol. 97, pp. 849–872, 2019.

[38] H. Gholizadeh and H. Fazlollahtabar, "Robust optimization and modified genetic algorithm for a closed loop green supply chain under uncertainty: case study in melting industry," *Computers and Industrial Engineering and Industrial Engineering*, vol. 147, Article ID 106653, 2020.

[39] J. Too, A. R. Abdullah, and N. Mohd Saad, "A new quadratic binary Harris hawk optimization for feature selection," *Electronics*, vol. 8, no. 10, p. 1130, 2019.

[40] A. Ramadan, S. Kamel, A. Korashy, A. Almalaq, and J. L. Domínguez-García, "An enhanced Harris Hawk optimization algorithm for parameter estimation of single, double and triple diode photovoltaic models," *Soft Computing*, vol. 26, no. 15, pp. 7233–7257, 2022.

[41] Y. Su, Y. Dai, and Y. Liu, "A hybrid parallel Harris hawks optimization algorithm for reusable launch vehicle reentry trajectory optimization with no-fly zones," *Soft Computing*, vol. 25, no. 23, pp. 14597–14617, 2021.

[42] H. Su, S. Maji, and E. Kalogerakis, "Multi-view convolutional neural networks for 3d shape recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 945–953, Santiago, Chile, December 2015.

[43] Z. Yang and L. Wang, "Learning relationships for multi-view 3D object recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7505–7514, Seoul, Korea (South), November 2019.

[44] X. Wei, R. Yu, and J. Sun, "View-gcn: view-based graph convolutional network for 3d shape analysis," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1850–1859, Seattle, WA, USA, June 2020.

[45] D. Maturana and S. Scherer, "Voxnet: a 3d convolutional neural network for real-time object recognition," in *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 922–928, Hamburg, Germany, September 2015.

[46] Z. Wu, S. Song, and A. Khosla, "3d shapenets: a deep representation for volumetric shapes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1912–1920, Boston, MA, USA, June 2015.

[47] T. Le and Y. Duan, "Pointgrid: a deep network for 3d shape understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9204–9214, Salt Lake City, UT, USA, June 2018.

[48] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics*, vol. 38, no. 5, pp. 1–12, Salt Lake City, UT, USA, June 2019.

[49] K. Zhang, M. Hao, and J. Wang, "Linked dynamic graph cnn: learning on point cloud via linking hierarchical features," 2019, https://arxiv.org/abs/1904.10014.

[50] Y. Liu, B. Fan, and S. Xiang, "Relation-shape convolutional neural network for point cloud analysis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8895–8904, Long Beach, CA, USA, June 2019.

[51] H. Naderi, C. Dinesh, and I. V. Bajic, "Model-free prediction of adversarial drop points in 3D point clouds," *IEEE Transactions on Multimedia*, 2022.

[52] H. Zhou, D. Chen, J. Liao, and Lg-gan, "Label guided adversarial network for flexible targeted attack of point cloud based deep networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10356–10365, Seattle, WA, USA, June 2020.

[53] Y. Wen, J. Lin, and K. Chen, "Geometry-aware generation of adversarial point clouds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, 2020.

[54] D. Liu and W. Hu, "Imperceptible transfer attack and defense on 3D point cloud classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, 2022.

[55] H. Güllü, "Prediction of peak ground acceleration by genetic expression programming and regression: a comparison using likelihood-based measure," *Engineering Geology*, vol. 141-142, pp. 92–113, 2012.

[56] H. Güllü and A. Ali Agha, "The rheological, fresh and strength effects of cold-bonded geopolymer made with metakaolin and slag for grouting," *Construction and Building Materials*, vol. 274, Article ID 122091, 2021.

[57] J. Luo, F. He, and X. Gao, "An enhanced grey wolf optimizer with fusion strategies for identifying the parameters of photovoltaic models," *Integrated Computer-Aided Engineering*, vol. 30, no. 1, pp. 89–104, 2022.

[58] N. Razavi, H. Gholizadeh, S. Nayeri, and T. A. Ashrafi, "A robust optimization model of the field hospitals in the sustainable blood supply chain in crisis logistics," *Journal of the Operational Research Society*, vol. 72, no. 12, pp. 2804–2828, 2021.

[59] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm Intelligence*, vol. 1, no. 1, pp. 33–57, 2007.

[60] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.

[61] Y. Zhou, S. Zhang, Q. Luo, and M. Abdel-Baset, "CCEO: cultural cognitive evolution optimization algorithm," *Soft Computing*, vol. 23, no. 23, pp. 12561–12583, 2019.
[62] G. Jeyakumar and C. Shunmuga Velayutham, "Distributed heterogeneous mixing of differential and dynamic differential evolution variants for unconstrained global optimization," *Soft Computing*, vol. 18, no. 10, pp. 1949–1965, 2014.
[63] J. R. Koza and R. Poli, *Genetic Programming. Search Methodologies*, Springer, Boston, MA, USA, 2005.
[64] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66–72, 1992.
[65] I. Rechenberg, *Evolutionsstrategien*, Springer, Berlin Germany, 1978.
[66] D. Dasgupta and M. Zbigniew, *Evolutionary Algorithms in Engineering Applications*, Springer Science and Business Media, Berlin Germany, 2013.
[67] H. Du, X. Wu, and J. Zhuang, *Small-world Optimization Algorithm for Function Optimization*, International conference on natural computation, Berlin, Heidelberg, 2006.
[68] H. S. Hosseini, "Principal components analysis by the galaxy-based search algorithm: a novel meta-heuristic for continuous optimisation," *International Journal of Computational Science and Engineering*, vol. 6, no. 1/2, pp. 132–140, 2011.
[69] J. Luo, F. He, H. Li, X. T. Zeng, and Y. Liang, "A novel whale optimisation algorithm with filtering disturbance and nonlinear step," *International Journal of Bio-Inspired Computation*, vol. 20, no. 2, pp. 71–81, 2022.
[70] D. B. Fogel, "Evolutionary computation: toward a new philosophy of machine intelligence," *Complexity*, vol. 2, no. 4, pp. 28–30, 1997.
[71] S. Lappin, J. F. Norman, and F. Phillips, "Fechner, information, and shape perception," *Attention, Perception, and Psychophysics*, vol. 73, no. 8, pp. 2353–2378, 2011.
[72] J. Yosinski, J. Clune, and A. Nguyen, "Understanding neural networks through deep visualization," in *Proceedings of the International Conference on Machine Learning*, Guangzhou, China, July 2015.
[73] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.
[74] V. Hayyolalam and A. A. Pourhaji Kazem, "Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems," *Engineering Applications of Artificial Intelligence*, vol. 87, Article ID 103249, 2020.
[75] J. Xue and B. Shen, "A novel swarm intelligence optimization approach: sparrow search algorithm," *Systems Science and Control Engineering*, vol. 8, no. 1, pp. 22–34, 2020.
[76] F. A. Hashim and A. G. Hussien, "Snake Optimizer: a novel meta-heuristic optimization algorithm," *Knowledge-Based Systems*, vol. 242, Article ID 108320, 2022.
[77] M. H. Nadimi-Shahraki, S. Taghian, and S. Mirjalili, "An improved grey wolf optimizer for solving engineering problems," *Expert Systems with Applications*, vol. 166, Article ID 113917, 2021.