Journal of King Saud University – Computer and Information Sciences 35 (2023) 101689

HOSTED BY

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

Security in defect detection: A new one-pixel attack for fooling DNNs

Pengchuan Wang [a], Qianmu Li [a,c], Deqiang Li [b], Shunmei Meng [a], Muhammad Bilal [d,e,*], Amrit Mukherjee [f]

[a] School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China
[b] School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China
[c] School of Intelligent Manufacturing, Wuyi University, Jiangmen 529020, China
[d] School of Computing and Communications, Lancaster University, Lancaster, United Kingdom
[e] Department of Computer Engineering, Hankuk University of Foreign Studies, South Korea
[f] Department of Computer Science, University of South Bohemia, Czech Republic

ARTICLE INFO

Article history:
Received 7 January 2023
Revised 27 June 2023
Accepted 27 July 2023
Available online 15 August 2023

Keywords:
Industry 5.0
Endogenous safety
Zero-defect production
Defect detection
Network equipment
Deep neural network
TLMFO
One-pixel attack

ABSTRACT

The Industrial 5.0 Model integrates enabling technologies such as deep learning, digital twins, and the meta-universe with new development concepts. However, model and data security may pose challenges for developing zero-defect production and other industrial manufacturing industries. To address this issue, we generate adversarial examples using a one-pixel attack in adversarial machine learning, which can fool the defect detection classification model. The traditional one-pixel attack based on the Differential Evolution (DE) algorithm has limited global search ability. Therefore, we use a novel algorithm called *Teaching and Learning-based Moth-Flame Optimization* (TLMFO), which enhances the global search performance and improves the attack effectiveness. We evaluate TLMFO on benchmark functions and attacks on Cifar10 and ImageNet datasets, and compare it with MFO and DE. The results show that TLMFO outperforms both MFO and DE in terms of accuracy and speed of convergence. Moreover, TLMFO achieves notably better attack effectiveness than DE under targeted and untargeted attacks on the Cifar10 dataset and under-targeted attacks on the ImageNet dataset. Our research confirms that safety prevention is a link worth considering in developing Industry 5.0.

1. Introduction

Over the past decade, Industry 4.0 (Ghobakhloo, 2020; Xu et al., 2022b) has integrated advanced technologies such as AI, deep learning, digital twins, metaverse, and the Internet of Things into manufacturing practices and applications. Building upon the innovations pioneered by Industry 4.0, Industry 5.0 (Leng et al., 2022) delves deeper into the potential of the Internet and AI to provide robust support for sustainable development. Industry 5.0 remains in its nascent stages, facing challenges related to future implementation strategies, potential applications, and practical, real-world scenarios.

Zero-defect production (Powell et al., 2022) is a paradigm that strives to eliminate defects by detecting and correcting defective products and process parameters and predicting and preventing defects. This innovative concept transcends traditional quality methods and plays a significant role in the sustainable development direction of both Industry 4.0 and 5.0. The term "zero defect" was first coined in the United States during the 1960s and has since been embraced by numerous large-scale industrial manufacturing enterprises as an essential component of their strategic development (Qi et al., 2022b; Zhang et al., 2022). With the advent of Industry 4.0 and continuous advancements in digital technology (Xu et al., 2021), zero-defect production has gained considerable momentum in recent years (Psarommatis et al., 2018). Furthermore, the sustainable development vision of Industry 5.0 will create even more opportunities for zero-defect production to thrive.

The advent of Industry 5.0 has brought about a deeper integration of artificial intelligence technology and the efficiency and accuracy of machines in industrial production (Akundi et al., 2022). However, data and technical security (Xu et al., 2022d) are major challenges that prevent enterprises from adopting zero-defect production (Mao et al., 2021). Software protection and data maintenance in enterprise production are often vulnerable to

* Corresponding author.
E-mail addresses: wangpc@njust.edu.cn (P. Wang), qianmu@njust.edu.cn (Q. Li), lideqiang@njupt.edu.cn (D. Li), mengshunmei@njust.edu.cn (S. Meng), m.bilal@ieee.org (M. Bilal), amrit1460@ieee.org (A. Mukherjee).

Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2023.101689
1319-1578/

malicious software that can launch destructive cyberattacks (Xu et al., 2020). Therefore, it is evident that security is essential for zero-defect production to realize its full potential. Consequently, the research on security in Industry 5.0 has experienced unprecedented growth in recent years.

Powell et al. (2022) conducted a structured literature review (Powell et al., 2022) examining recent advances in zero-defect manufacturing (ZDM). The authors classified the current ZDM strategies into defect prevention and defect compensation. They also highlighted that the future global challenge for zero-defect production is to overcome the network security issues that may limit the application scope of defect prevention and compensation strategies.

In the Industry 5.0 manufacturing industry context, implementing ZDM solutions, especially at the shop floor level, requires data security and other considerations to ensure the safety and reliability of data collection and processing from different sources during the manufacturing process. However, most of the existing ZDM solutions proposed by researchers need to consider the reliability of data security and machine learning algorithms in practical applications. Therefore, the feasibility of ZDM solutions depends on whether data and model security can be guaranteed, which is a problem that needs to be explored further.

Deep Neural Networks (DNNs), powered by large-scale datasets, can outperform humans in image processing and ZDM tasks.However, DNNs are vulnerable to adversarial examples, which are crafted by adding subtle perturbations to the original examples (e.g., injecting imperceptible noises into an image).

Adversarial example attacks belong to the research field of Adversarial Machine Learning (AML). Essentially, it is a constrained optimization problem that aims to fool the DNN models by using the smallest perturbations according to certain distance metrics. Fig. 1 illustrates the three common types of attacks in the AML scenario: privacy attacks, attacks against training data, and attacks against DNN models. Privacy attacks on machine learning usually happen in the inference stage of the model, where the DNNs are observed to predict or recover some sensitive information. Attacks against training data are also known as poisoning attacks, and the strategy is to inject maliciously distributed data to corrupt the training dataset. Attacks against the algorithm models seek adversarial examples that cause the model to produce a wrong prediction.

This paper addresses the data security issues (Xu et al., 2022c) that arise in applying AI technology for zero-defect detection in product manufacturing. Specifically, we focus on the black-box attack scenario, where an adversary can modify a pixel in the input image to fool the DNN model and cause false detection results, leading to increased product costs (Su et al., 2019; Khan et al., 2019). We propose using the number of perturbation vectors instead of their length to measure attack effectiveness. Moreover, we employ an improved Evolutionary Algorithm (EA) that does not require gradient information and only relies on the input–output behavior of the defect detection model to generate one-pixel attacks. Our method enhances the realism and feasibility of the attack scenario.

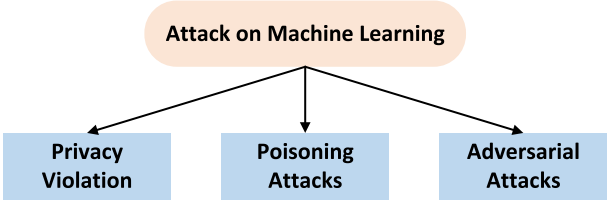

Fig. 1. Attacks on machine learning.

Khan et al. demonstrated the effectiveness of one-pixel attacks on various neural networks and revealed the limitations of DE in exploiting the full potential of such attacks (Khan et al., 2019). DE often fails to find the optimal one-pixel perturbation that can cause the highest misclassification rate. Despite this, recent studies have not explored alternative optimization methods to EA for one-pixel attack tasks.

Our contributions. This paper explores the potential threat of generative adversarial example attacks on zero-defect production in Industry 5.0 scenarios. We present a case study that shows the vulnerability of a defect detection model to one-pixel attacks and underscores the importance of model safety and data security in zero-defect manufacturing. To enhance the attack effectiveness of one-pixel attacks, we propose a novel EA, TLMFO, which integrates a teaching mechanism into Moth-Flame Optimization (MFO) (Nayak et al., 2020; Mirjalili, 2015) to improve its optimization performance and balance its exploration and exploitation capabilities. The detailed contributions of this paper are summarized as follows:

- We introduce a unique learning mechanism that enhances the global optimization capability of MFO by introducing a teaching phase from the best moth and a mutual learning phase among moths. We call this method TLMFO. The teaching and learning phases are integrated into different stages of MFO, enabling more frequent communication among moth individuals and achieving a better balance between exploration and exploitation than MFO.
- We develop a new method for finding the optimal one-pixel attack using the TLMFO core instead of the DE core. This method fills the gaps in the research field of one-pixel attacks and demonstrates the high potential of EA in adversarial machine learning.
- To evaluate the generalization performance of TLMFO, we first compare it with other mainstream metaheuristics for global optimization on the CEC2015 and 2019 function datasets. Moreover, to verify that adversarial attacks can cause errors in different defect detection models, we apply our method to various datasets against diverse models and show that adversarial attacks can compromise model security.

The structure of this article is as follows. Section 2 reviews the previous research on one-pixel attacks and defect detection. Section 3 gives some preliminary knowledge. The new method is proposed in Section 4 and describes the process of the TLMFO for one-pixel attack tasks. Section 5 verifies the TLMFO by global optimization and one-pixel attack experiments, following with results and discussion. Conclusions and future research are given in the last section.

2. Related works

2.1. One-pixel attack

The one-pixel adversarial attack is a type of malicious attack that occurs in the model testing stage. The attacker crafts adversarial examples by adding carefully designed small perturbations to the original data, which can fool the deep learning model and make it misclassify them with high confidence. We provide a brief overview of the literature on the one-pixel attack, its applications, and the limitations of using DE to perform one-pixel attacks. Fig. 2 illustrates the one-pixel attack and its applications. Various variants of the one-pixel attack have been applied in real scenarios with improved performance. Conversely, the existence of real

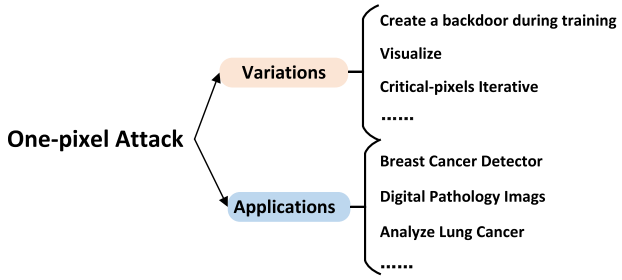

Fig. 2. One-pixel attack and its applications.

applications has motivated researchers to enhance the performance of the one-pixel attack.

2.1.1. Variations of one-pixel attack

Some studies have suggested that embedding a backdoor during training can enhance attack performance. Alberti et al. devised an attack that can induce abnormal behavior of the neural network in the testing stage by modifying only one pixel in the training image when the original network structure is unknown (Alberti et al., 2018). Similarly, some studies have shown that visualizing receptive fields can help solve many problems. In 2020, Vargas observed that attacking nearby pixels of the perturbed pixel in a one-pixel attack often produces the same effect. Therefore, Vargas et al. introduced a propagation map, which displays the influence of each layer in the DNNs structure on the disturbance of feature points (Vargas and Su, 2020). In 2020, Wang et al. employed the adversarial map technique to visualize the distribution of one-pixel attacks (Wang et al., 2020). Sinha et al. visualized the activation map and heat map of each layer of the DNN model in a one-pixel attack to generate adversarial perturbation for one pixel (Sinha and Saranya, 2021).

On the other hand, some research focused on adding a Critical Pixel Iteration (CriPI) algorithm for better performance. In 2021, Quan et al. (Quan et al., 2021) proposed a new Critical Pixel Iteration (CriPI) algorithm to identify pixels with higher importance scores and generate one-pixel attack perturbations with higher success rates. Zhou et al. (Zhou et al., 2022) applied the strategy from the dataset to the optimization process of the one-pixel attack. They leveraged the image structure information to optimize the initial generation scheme, thereby improving the attack performance. They called this method Image Structure based One-pixel Attack (ISOA).

Chen et al. developed a defense technique using the one-pixel attack's vulnerability, Patch Selection Denoiser (PSD). This technique does not change most pixels in the whole image but removes a few potential attack pixels in a partial patch (Chen et al., 2019). Li et al. used the one-pixel attack to embed additional information and create an adversarial steganographic image based on data-hiding technology to achieve a black box attack (Li et al., 2023).

2.1.2. Applications of one-pixel attack

One-pixel attacks have been successfully applied to many different domains as a black-box adversarial attack paradigm:

Korpihalkola et al. used one-pixel perturbed adversarial images to attack the IBM CODAIT's MAX breast cancer detector and showed that one-pixel attacks pose a threat to cyber security (Korpihalkola et al., 2021). Alatalo et al. analyzed the chromaticity and spatial distribution methods of one-pixel attacks and studied their success and failure in more detail (Alatalo et al., 2022).

Paul et al. proposed a multi-initialized Convolutional Neural Network (CNN) based defense strategy that used the Fast Gradient Sign Method (FGSM) and one-pixel attack techniques to minimize the misclassification rate and make the CNN model more robust against adversarial attacks (Paul et al., 2020).

In 2021, Korpihalkola et al. attempted to reduce the misdiagnosis rate of AI-based medical imaging methods under attack (Korpihalkola et al., 2021). They presented a one-pixel attack method for color optimization of medical imaging and devised a new one-pixel attack that has the least impact on the color value of pixels and reduces the unnatural coloring of the original non-pixel attack. Guo et al. applied the one-pixel attack technique to a Continuously Variable Quantum Key Distribution (CV-QKD) system and directly attacked the CV-QKD defense countermeasures based on the DNNs classification in 2023 (Guo et al., 2023).

2.2. Defect detection based on machine vision

The rapid advancement of computer storage capacity and computing power has enabled the integration of artificial intelligence technologies (Xu et al., 2022a), such as machine vision and deep learning, for high-quality surface defect detection. Combining these two technologies can improve detection accuracy and reduce design costs, which is a goal pursued by many researchers (Qi et al., 2022a; Xu et al., 2021). This subsection reviews some of the recent mainstream research on defect classification detection based on machine vision.

2.2.1. Traditional image processing methods

Raheja et al. (2013) developed a new scheme of automatic fabric defect detection systems based on a Grey-level Cooccurrence Matrix (GLCM) (Zhang et al., 2022) and a windowed short-time Fourier transform (Gabor filter). Moreover, Zhang et al. (Zhang et al., 2015) combined the Local Binary Pattern (LBP) with the GLCM method to extract the feature information of the defective fabric image from both local and global perspectives. Dunderdale et al. (Dunderdale et al., 2020) applied Scale-invariant Feature Transform (SIFT) descriptors to detect local feature points and integrated them with a random forest classifier to identify defective photovoltaic modules.

2.2.2. Intelligent methods based on deep learning

Masci et al. (Masci et al., 2012) developed a maximum pool CNN method for the supervised classification of steel defects, which can accurately distinguish different types of defects. Park et al. (Park et al., 2016) presented a general approach based on CNN to inspect the surface abnormality of parts automatically. Nguyen et al. (Nguyen et al., 2021) devised an inspection system based on CNN to achieve defect classification of casting products. Yang and Jiang (Yang and Jiang, 2021) adopted an unsupervised pre-training method and introduced a unified DNN with multi-level features to detect weld seam anomalies from images.

In conclusion, there is a lack of safety research in machine vision defect detection based on traditional and deep learning methods. If a trusted technical method is attacked, it will cause irreparable economic and reputation losses to the enterprise organization (Maddikunta et al., 2022). It will also hinder the achievement of the zero-defect production goal. Therefore, this paper adopts the traditional idea of attack as defense and integrates attack and defense to investigate whether defect detection is vulnerable to adversarial attacks.

3. Preliminaries

3.1. One-pixel attack

The one-pixel attack was first introduced by Su et al. in 2019 (Su et al., 2019), who attempted to create an adversarial example by

modifying a single pixel. Fig. 3 shows a schematic diagram of how the one-pixel attack can fool DNNs. In the original problem, $f$is the target image classifier, $x$is the original image sample correctly classified as t, and $f_t(x)$ is the probability of correct classification. The one-pixel attack is a targeted black-box attack that can be formulated by the following Eq. (1):

$$e^*(x) = argmax_{e(x)} f_{adv}(x + e(x)) \tag{1}$$

$$subject to \|e(x)\|_0 \leqslant d \tag{2}$$

The vector $e(x)$ in the problem is an added adversarial perturbation. The one-pixel attack aims to find the optimal solution $e^*(x)$ that can misclassify $x$into the target class advunder the maximum perturbation limit d.

Unlike full-pixel attacks and multi-pixel attacks, one-pixel attacks modify only one pixel and replace the perturbation vector of that pixel with the corresponding position of the zero matrix $e(x)$. Each perturbation vector group (for color RGB image) consists of five elements: the x-y coordinate value of the pixel in the original image and the RGB value of the pixel itself. Eq. (3) is the iterative formula for generating candidate solutions using the DE algorithm in the original method:

$$x_i(g+1) = x_{r_1}(g) + F\left(x_{r_2}(g) + x_{r_3}(g)\right), r_1 \neq r_2 \neq r_3 \tag{3}$$

Here x_i is a candidate solution, and other parameters are consistent with the DE algorithm. Each generated candidate solution is compared with corresponding parents and puts the best candidate set into the next iteration. The iteration is terminated when the trigger stop criterion.

3.2. MFO algorithm

The optimization mechanism of MFO ensures the efficiency, flexibility and portability of the algorithm. MFO also has a more competitive performance in processing and optimizing search spaces with unknown or less known information. In this subsection, we briefly introduce the mathematical mechanism of MFO. Based on the biological characteristics of the transverse orientation of the moth, MFO defines the spiral function as the main iterative formula, as shown in Eq. (4):

$$M_i = S(M_i, F_j) = D_i \cdot e^{bt} \cdot cos(2\pi t) + F_j \tag{4}$$

where M_i denotes the $i$moth, F_j denotes the $j$flame, and S is a spiral function.

Here the distance between the $i$moth and the $j$flame is denoted by D_i. $b$is generally chosen as 1, and $t$is an arbitrary number in the interval from −1 to 1. The specific calculation formula is as following Eq. (5):

$$D_i = |F_j - M_i| \tag{5}$$

The MFO algorithm makes the number of flames adaptively decrease with the iteration and finally retains the optimal flame as the optimal solution. The specific calculation formula is as Eq. (6):

$$flame\,no = round\left(N - l * \frac{N-1}{T}\right), \tag{6}$$

where l, N, T are the current iteration number, maximum flame number, and maximum iteration number.

However, MFO still suffers from low accuracy when solving high-dimensional and complex problems. In the next section, we will introduce our proposed method and apply it to a one-pixel attack.

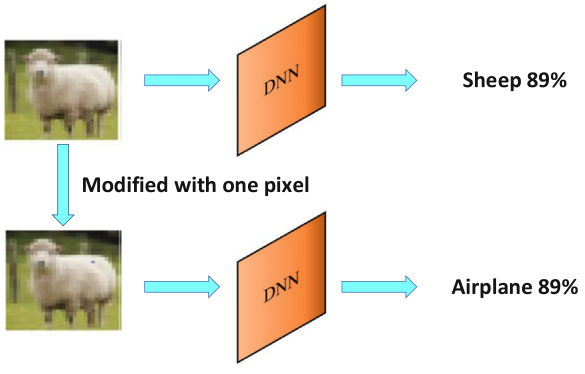

Fig. 3. Attacks on machine learning (Yang and Jiang, 2021).

4. TLMFO

Self-learning is a concept that has gained attention in artificial intelligence technology in recent years. Some EAs with specific learning capabilities are called learning-based intelligent optimization algorithms (LIOAs) (Park et al., 2016). LIOAs incorporate some learning operators and mechanisms that bring some advantages to traditional intelligent algorithms, such as: (i) Enhancing the specific learning abilities of traditional intelligent algorithms; (ii) Facilitating more frequent and effective information exchange between individuals and the algorithm; (iii) Improving the convergence efficiency of the algorithm. In this paper, we propose the TLMFO, which integrates the teaching and learning mechanism with the moth flame optimization (MFO) algorithm to overcome the limitations of the original MFO and better exploit the application potential of the one-pixel attack problem.

4.1. TLMFO algorithm

In biological evolution, organisms constantly adapt their behaviors to the environment to enhance their survival chances. Moths have evolved a remarkable ability to navigate under moonlight and starlight, which suggests that they are a group of organisms with cognitive, learning, and decision-making abilities.

The Teaching and Learning-based Optimization algorithm (TLBO) mimics two mechanisms of information exchange in a class: (i) teacher teaching and (ii) student peer learning. The teacher phase simulates the teaching process of the teacher to the students, and the learner phase simulates the cooperative learning process among the students (Wang et al., 2021; Zou et al., 2019). In this section, we introduce TLBO (Rao et al., 2011) into MFO, using the teacher and learner phases to balance the exploration and exploitation processes of MFO. On the one hand, this enables the moths to have the ability to learn independently and improves the optimization ability and the quality of candidate solutions. On the other hand, the improved algorithm has strong self-organization and collaboration capabilities. Fig. 4 shows the flowchart of the TLMFO algorithm. The blue dashed boxes indicate the teacher and learner mechanisms.

To describe the optimization process of TLMFO in detail, we numbered the main steps in the flowchart. In the initialization phase, we set the same constant parameter values (such as the spiral factor) as the original MFO. Moreover, we randomly initialize the moths according to the specific problem and generate the corresponding flame positions using the fitness function. These steps correspond to 1–3 in Fig. 4.

Algorithm 1. The pseudo code of TLMFO algorithm

```
   /* Initial Phase                                          */
1  Initialize: The spiral factor b = 1, the maximum number
     of iterations: Maxiter, Initial position of moth
     population: M, the number of dimensions: d;
2  Up flame no using Eq. (6);
3  OM = FintessFunction(M)
4  if iteration = 1 then
5  |  F = sort(M);
6  |  OF = sort(OM);
7  else
8  |  F = sort(M_{t-1}, M_t);
9  |  OF = sort(OM_{t-1}, M_t);
10 end
   /* Teacher Phase                                          */
11 Donate the best Moth as the Teacher;
12 for each moth of the population do
13 |  T_f = rand(1 + rand(0,1){2 - 1});
14 |  for j = 1 → d do
15 |  |  Calculate D_difference using Eq. (7);
16 |  end
17 |  Update M using Eq. (9) if M^i_new is better than M^i_old;
18 end
   /* Best Moth individual update Phase                      */
19 for j = 1 → N do
20 |  for j = 1 → d do
21 |  |  update r and t;
22 |  |  Calculate D using Eq. (5) with respect to the
   |  |    corresponding moth;
23 |  |  Update M using Eq. (10) with respect to the
   |  |    corresponding moth;
24 |  end
25 end
   /* Learner Phase                                          */
26 Donate the mean of all moths as Mean;
27 Randomly select one moth as learner M_k, (i ≠ k);
28 if M_i betterthan M_k then
29 |  for j = 1 → d do
30 |  |  M^i_new = M^i_old + round(0,1)(M_k - M_i);
31 |  end
32 else
33 |  for j = 1 → d do
34 |  |  M^i_new = M^i_old + round(0,1)(M_i - M_k);
35 |  end
36 |  Update M using Eq. (9) if M^i_new is better than M^i_old;
37 end
38 return the Best OM.
```

4.1.1. Teacher phase

In the exploration process of MFO, the teacher phase selects the moth with the best current objective function value as a teacher and simulates the teaching behavior to improve the learning level of other moths in the population. This improves the overall learning level of the population and helps to find the optimal solution. The difference between the overall average (mean value) of the students (moths) and the teacher (the best individual) is given by Eq. (7):

$$D_{difference} = rand(0,1)\left(M_{teacher} - T_f M_{mean}\right) \tag{7}$$

$M_{teacher}$ is the current optimal moth individual, and M_{mean} is the average level of the objective function values of all moths. T_f is the teaching factor that reduces the difference between $M_{teacher}$ and M_{mean}. The value of T_f is either 1 or 2, which is randomly chosen, following the original paper as shown in Eq. (8):

$$T_f = rand(1 + rand(0,1)\{2-1\}) \tag{8}$$

In the teacher phase, the ith moth in the population updates its position according to the Eq. (9):

$$M^i_{new} = M^i_{old} + D_{difference} \tag{9}$$

In the teacher phase, the moth population has gone through a round of teacher teaching. According to the principle of survival of the fittest, a new moth population M_{new} is obtained, corresponding to step 4 to 6 in Fig. 4.

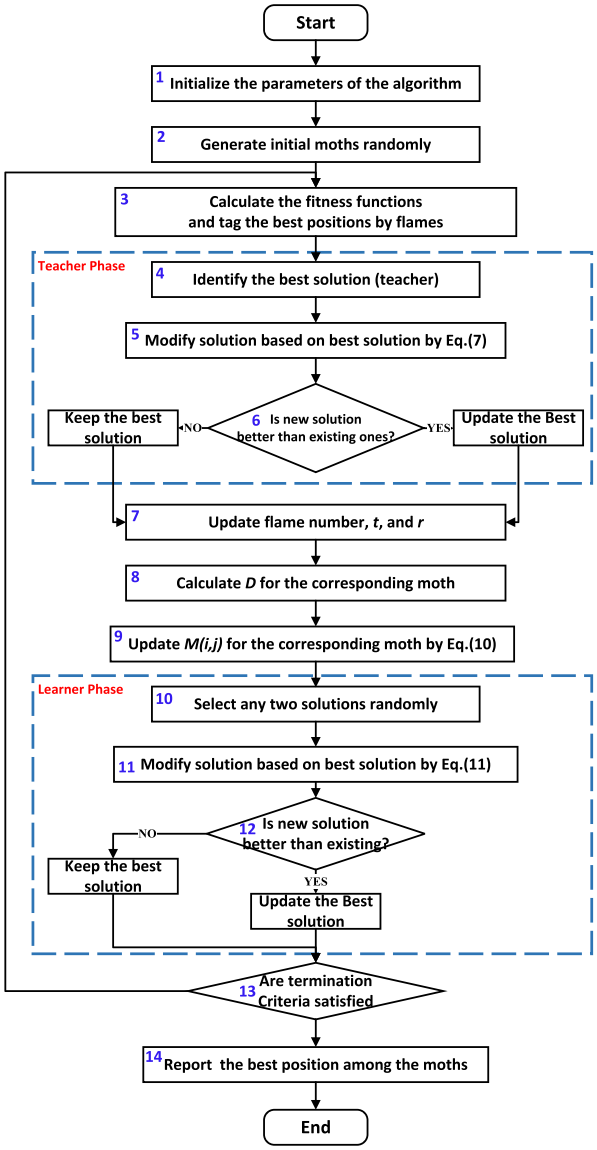

Fig. 4. The flowchart of TLMFO algorithm.

4.1.2. Individual update phase

After a round of teacher phase, the moth population gets better moths than the initial ones. According to the logarithmic spiral function, the moth's update is as follows:

$$M_i = S\left(M^i_{new}, F_j\right) = D_i \cdot e^{bt} \cdot cos(2\pi t) + F_j \tag{10}$$

Here M_i denotes the $i$moth (up to date), the D_i is updated according to Eq. 5, and the adaptive flame mechanism is consistent with MFO, corresponding to step 7 to 9 in Fig. 4. To enhance the exploitation ability of the moths after the individual update, we introduce a learner mechanism in the exploitation phase to increase the exploration rate in the field space around flames.

4.1.3. Learner phase

In the actual teaching process, the teacher's input and the mutual communication between students are the main ways to increase knowledge. Random interaction between students will give students with poorer knowledge obtains more knowledge in the learner phase. In the exploitation phase of MFO, we introduce the mutual learning mechanism between moths and randomly select two students (moths) M_i and M_j from the class, and the $i-th$ student (moth) updates its position according to the following Eq. 11:

$$M^i_{new} = \begin{cases} M^i_{old} + round(0,1)(M_i - M_j) & f(M_i) < f(M_j) \\ M^i_{old} + round(0,1)(M_j - M_i) & f(M_i) > f(M_j) \end{cases} \tag{11}$$

In principle, the knowledge reserves of the student (moth) M^i_{new} will continue to increase due to cooperative learning.

After the two-phase optimization process of teacher teaching and individual update, we added a mutual learning stage in the second half of TLMFO. In this stage, the optimization difference between different moths is reduced to balance the exploration and exploitation process of the moths and further explore the optimal solution. This phase corresponds to steps 10 to 12 in Fig. 4. The final algorithm outputs the optimal solution after reaching the termination condition set for the problem, corresponding to steps 13 to 14.

4.1.4. Computational complexity

To better illustrate the iterative process of the TLMFO algorithm solving problems, we present the pseudo-code of TLMFO: Algorithm 1.

The computational complexity (CC) generally defines the key indicators of its running time according to the implementation process of the algorithm. As shown in Algorithm 1, the CC of TLMFO is calculated as follows:

1) Initialize the population $O(n)$.
2) Loop iteration. Iteratively compute the fitness of the best moth in the population $O(t)$.
3) The selection of the best teacher: quick sorting method $O(n^2)$;
4) Checking the change of fitness value of the local optimal individual $O(1)$;
5) The ranking mechanism of flames $O(nd)$.
6) The mutual learning mechanism among students $O\left(\frac{n(n-1)}{2}\right)$;
7) Updating populations $O(d)$.
8) Obtain the best individual $O(1)$.

Therefore, the CC of TLMFO is given by Eq. 12:

$$\begin{aligned} O(TLMFO) = & \; O\left(n + t\left(O\left(t + n^2 + n \times d + 1 + \tfrac{n(n-1)}{2}\right) + d\right) + 1\right) \\ = & \; O\left(\frac{3}{2}tn^2 + tnd\right) \end{aligned} \tag{12}$$

According to the rules of time complexity analysis, the time complexity of the TLMFO algorithm increases within an acceptable range.

4.2. TLMFO algorithm application to one-pixel attack

This section elaborates the TLMFO-based one-pixel attack. Fig. 5 depicts the high-level overview of our attack, which consists of three phases: prediction, perturbation, and attack.

4.2.1. Prediction phase

The one-pixel attacks are a type of black-box attack that does not rely on the internal information of the victim models. We therefore use evolutionary computation to perform this attack. We use an image dataset with correct classification labels in the initial phase to evaluate the attack effect. This process corresponds to the prediction phase in Fig. 5. Suppose we have a test dataset $I_{test} = \{I_1, I_2, \cdots, I_N\}$ for launching attacks, belonging to L different categories $Y = \{y_1, y_2, \cdots, y_L\}$. The input and output mapping can be obtained through the DNN classifiers $y' = f(I_{test})$. We can screen out the target images I_{true} with the ground truth label by comparing the initial labels. In the next phase, we use the target image set to verify the success rate of one-pixel attacks. The 2nd to 8th step of

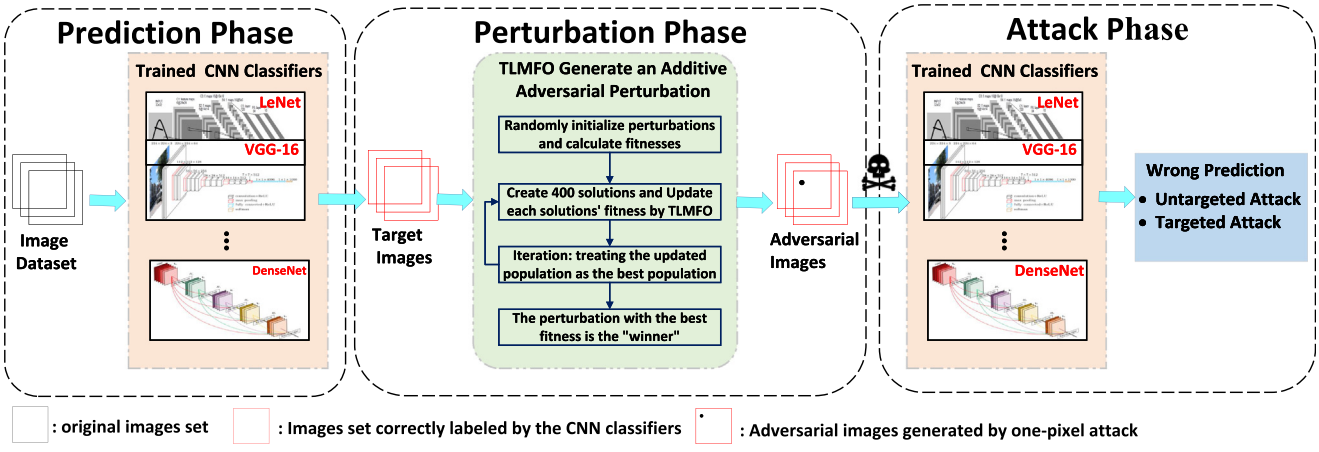

Fig. 5. Three phases (prediction, perturbation, and attack) of the TLMFO-based one-pixel attack.

Algorithm 2 indicates the entire process of our prediction phase. We will briefly introduce the three main steps in the prediction phase.

Algorithm 2. The proposed TLMFO-based one-pixel attack

Input: Image dataset I, Five CNN models $f(x)$.
Output: Set of perturbed factors P_{Adv}, Attack success rate.

1 Initialize: The spiral factor $b = 1$, the maximum number of iterations: $Maxiter = 100$, the initial adversarial example $P_{Adv} = (x_1, x_2, ..., x_i)$, $p_i = (x_i, y_i, R_i, G_i, B_i)$;
`/* Datasets Test Screening Phase */`
2 Upload image datasets: CIFAR10, ImageNet
3 Upload Trained CNN model: Lenet, Pure-cnn, Net-in-Net, Resnet, Capsnet
4 **for** *index to range*($len(I\_test)$) **do**
5 Save correctly classified images I_ture:
6 $result = model.predict\_classes(I\_test[index])$;
7 Calculate the classification accuracy of the model;
8 **end**
`/* Definition of the Objective Function */`
9 Calculate the optimal solution to the perturbation: $e^*(x) = argmax_{e(x)} f_{adv}(x + e(x))$, subject to $\|e(x)\|_0 \leq d$;
10 Targeted attacks: $f(X_{Adv})! = f(X)$;
11 Untargeted attacks: $f(X_{Adv}) = target\_class$;
`/* TLMFO-based Attacks Phase */`
12 Attacked image: $I = I\_ture$, current iteration count: $t = 0$, $I_{label} = f(I)$;
13 **for** t *to* $Maxiter$ **do**
14 **for** $i = 1 \rightarrow Maxiter$ **do**
15 $p_i = (randn(0, len, 2), randn(0, 255, 3))$;
16 Use TLMFO algorithm to explore the optimal solution;
17 Append (P, p_i)
18 **end**
19 $P_c = perturb(P, I)$, $t++$;
20 *Untargeted attacks:*
21 **If** $f(P_c) \neq I_{label}$ **then** break
22 *Targeted attacks:*
23 **If** $f(P_c) = Targeted_{label}$ **then** break
24 **end**
25 **Return** P_{Adv}, $Attacksuccess\_rate$.

Load image dataset For this attack, we used the common CIFAR-10 dataset and ImageNet dataset to verify the effectiveness of the attack (Khan et al., 2019). The main task of a one-pixel attack on the CIFAR-10 dataset is to correctly classify a 32x32 pixel image into one of 10 categories (such as cat, frog, bear), and each category has 6000 images. Here we select 5000 images in each category as the training set and 1000 as the test set.

The ImageNet dataset is currently the most widely used in deep learning for images. The ImageNet dataset covers 20,000 categories with more than 14 million images. The search space of ImageNet is 50 times that of CIFAR-10. The task of the ImageNet dataset is to verify whether this minor modification can fool a larger image.

Load CNNs Models We train several different convolutional neural networks as the target classifiers on the CIFAR-10 dataset. We omit the details of the specific network architectures.

Calculate Model Accuracies After loading the models, we evaluate all test images with each model to ensure that we only attack the images that are classified correctly.

4.2.2. Perturbation phase

Against the one-pixel attack, we only need to consider the model output feedback (classification probability) given by the model input (image) rather than gradients and network structures. We chose TLMFO as a tool for generating adversarial examples, hoping to effectively find solutions that have more attack potential than traditional gradient methods or DE. The perturbation phase uses the TLMFO algorithm to find an adversarial example that causes the model to misclassify. This process corresponds to the perturbation phase in Fig. 5.

Prediction Function In the perturbation phase, we formulate the perturbation of the optimal pixel as a variational optimization problem. The basic variable initialization is shown in the first line of Algorithm 2. We designed an objective function in lines 9 to 11 of Algorithm 2. This function runs multiple perturbed images on a given model, hoping that the adversarial perturbed image sample X_{Adv} can return a lower probability of the true label while maintaining a low perturbation cost. We have provided the success conditions for targeted and untargeted attacks according to different attack scenarios.

TLMFO Core First, we preset the same problem description, and use Eq. 1 to find the optimal solution on which dimensions need to be perturbed and the perturbation strength of each dimension.

The second step is to use the TLMFO described in the previous section to optimize the perturbed coding group (candidate solution). The candidate solution also contains five elements: x-y coordinates and perturbed RGB values. For the fairness of subsequent tests, we set the number of initial candidate solutions to 400. We use Eq. 10 in TLMFO to generate the initial sub-items and use the TLMFO update strategy (Eq. 7 and Eq. 11) to iteratively update the sub-items to get the best solution. The specific implementation steps are shown in Fig. 6:

The third step is to modify the dataset using the optimal perturbation coding set. By perturbing only one pixel in the image, we make the CNNs output incorrect classification labels.

Starting from line 12 of Algorithm 2, we present the pseudo-code for TLMFO optimization. The best perturbation factor p_i' can be obtained by sequential iterations. Line 16 of Algorithm 2 shows how TLMFO optimizes the process of the five-element group p_i.

4.2.3. Attack phase

Here we demonstrate two variants of the one-pixel attack: untargeted and targeted. The goal of an untargeted attack is to misclassify an image. We only need to ensure that the perturbed image is classified into any other category except its correct one to maximize the sum of the probabilities of all different categories. The goal of a targeted attack is to make a model classify an image as a given target class. We only need to make the perturbed image have a higher probability of being classified into the target category than the correct one. This process corresponds to the attack phase in Fig. 5.

How do we find the pixels that will result in a successful attack? First, we formulate it as an optimization problem: we use the TLMFO algorithm to iteratively test the perturbed image samples on the model. In an untargeted attack, we minimize the output probability of the correct category. In a targeted attack, we maximize the output probability of the target category. Lines 20 to 23 of Algorithm 2 indicate the entire process of our attack phase.

5. Experimental results and analysis

5.1. Global optimization

In this section, we first verify the optimization potential of TLMFO in the field of function optimization and compare it with DE. Then, we demonstrate the superiority of TLMFO over the current mainstream meta-heuristic methods.

5.1.1. Parameter settings

We conducted simulation experiments of function optimization on a computer equipped with a 64-bit Windows 11 operating system, Intel Core i7-10875H, CPU@ 2.30 GHz, 16 GB RAM, and MATLAB version R2020b. We ran each Algorithm 30 times with 1000 iterations per run and compared the performance of TLMFO with eight other algorithms: MFO, DE (Das and Suganthan, 2011), Gray Wolf Optimizer (GWO) (Faris et al., 2018), Particle Swarm Optimization (PSO) (Kennedy and Eberhart, 1995), Seagull Optimization Algorithm (SOA) (Dhiman and Kumar, 2019), Gradient-based Optimizer (GBO) (Ahmadianfar et al., 2020), rithmetic Optimization Algorithm (AOA) (Abualigah et al., 2021), and Reptile Search Algorithm (RSA) (Abualigah et al., 2022). Table 1 shows the parameters of these algorithms.

5.1.2. Benchmark test functions

We selected nine benchmark functions from the CEC2015 and CEC2019 benchmark (Mohammadi-Balani et al., 2021) sets to compare and verify the optimization performance of TLMFO, MFO, and DE (Mohammadi-Balani et al., 2021). The benchmark functions can be classified into three types: unimodal functions (f_1-f_3), which have only one global optimum and test the exploration ability and convergence speed of the algorithms; multimodal functions (f_4-f_6), which have many local optima and test the exploitation ability and local search ability of the algorithms; and fixed-dimensional functions (f_7-f_9), which are similar to multimodal functions but test the global search ability in low-dimensional spaces. Table 2 shows the details of the benchmark functions, including the dimension (*Dim*), the theoretical optimal value (f_{min}), and the search range (*Range*) of each function.

5.1.3. Experimental results and analysis

We performed the experiments under consistent testing conditions for all algorithms. Tables 3–5 show the experimental results of the benchmark functions, where the Best and Std columns represent the best value and standard deviation obtained by each algorithm. The algorithms are ranked according to the Best value, and the Rank column shows the rank of TLMFO. The best-ranked algorithm for each function is highlighted in bold and underlined.

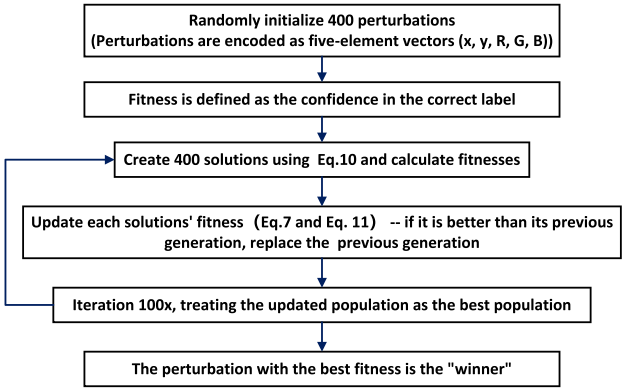

Fig. 6. A flowchart describing TLMFO.

Table 3 lists the results of all algorithms on the unimodal benchmark function. The table shows that the TLMFO algorithm outperforms other algorithms in terms of overall performance. In particular, for f_1 and f_3, the TLMFO algorithm achieves the theoretical optimal value in 500 dimensions. For f_2, although the TLMFO algorithm does not reach the theoretical optimal value, it has a higher accuracy than other algorithms. Moreover, the table indicates that the variance value obtained by the TLMFO algorithm is closer to 0 than that of other comparison algorithms, which demonstrates its better robustness.

Furthermore, DE performs poorly on such high-dimensional problems, as it fails to converge to the theoretical optimal value and has the median best value and the lowest variance value among the rankings of several algorithms. This indicates that DE has low robustness on such high-dimensional problems. In contrast, TLMFO performs well on the high-dimensional unimodal test function, proving that TLMFO is effective and feasible for solving high-dimensional unimodal problems compared to DE and MFO.

Fig. 7–9 show the mean convergence and variance plots of TLMFO, MFO, DE, GWO, PSO, AOA, SOA, GBO, and RSA on the high-dimensional unimodal function after 30 runs. From Fig. 7–9, we can see that although RSA achieves the same theoretical optimal value as TLMFO, TLMFO has the fastest convergence speed and the highest accuracy among the eight algorithms. The convergence and variance plots also reveal that DE does not converge to the optimal value in $f_1 - f_3$. In summary, the TLMFO algorithm outperforms DE, MFO, and other algorithms regarding accuracy, robustness, and stability on unimodal functions in high-dimensional spaces.

Table 4 shows the results of the multimodal benchmark function. The rank column in Table 4 shows that the TLMFO algorithm has the best performance among the eight comparison algorithms, ranking first in terms of optimization. Moreover, the TLMFO, GBO, and RSA algorithms achieve the same excellent performance on f_4 and f_5, reaching the theoretical optimal value on f_4. However, from the convergence curves (Fig. 10 and 11) below, we can observe that the TLMFO algorithm converges faster than GBO and RSA. From the data analysis, in the multimodal benchmark test function, TLMFO outperforms DE in every aspect, which makes it a convincing alternative to DE for one-pixel attacks. Therefore, we can confirm that TLMFO is more competitive in optimizing high-dimensional multimodal functions.

Fig. 10–12 show all algorithms' mean convergence and variance plots on high-dimensional multimodal functions after 30 independent runs. The mean convergence plots show that the TLMFO algorithm (green curve) can converge faster than other algorithms and maintain a high accuracy close to the theoretical value. According to the variance plots and Table 4, the variance value of the TLMFO

Table 1
Parameter setting.

Algorithm	Parameter	Value
TLMFO	Spiral factor b	1
MFO	Spiral factor b	1
DE	Crossover Probability	0.2
	Lower Bound of Scaling Factor	0.2
	Bound of Scaling Factor	0.8
GWO	$\vec{\alpha}$	linear descent 2 to 0
PSO	$W_{Max}, W_{Min}, C_1, C_2$	0.9, 0.6, 2, 2
SOA	Control Parameter (A)	[-2,0]
	f_c	2
GBO	β_{min}, β_{max}	0.2,1.2
	Probability pr	0.5
AOA	control parameter μ	0.5
	sensitive parameter α	5
RSA	α, β	0.1, 0.1

Table 2
Information and features of benchmark test functions.

Unimodal Benchmark Functions				
Name	Function	*Dim*	Range	f_{min}
Sphere	$f_1(x)=\sum_{i=1}^{n}x_i^2$	500	$x_i\in[-100,100]$	0
Schwefel2.22	$f_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	500	$x_i\in[-10,10]$	0
Schwefel1.2	$f_3(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}x_j\right)^2$	500	$x_i\in[-100,100]$	0
Multimodal Benchmark Functions				
Name	Function	*Dim*	Range	f_{min}
Rastrigin	$f_4(x)=\sum_{i=1}^{n}[x_i^2-10cos(2\pi x_i)+10]$	500	$x_i\in[-5.12,5.12]$	0
Ackley	$f_5(x)=-20exp\left[\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-exp(\frac{1}{n}\sum_{i=1}^{n}cos2\pi x_i)\right]$	500	$x_i\in[-32,32]$	0
Alpine	$f_6(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}\lvert x_j sin(x_i)+0.1x_j\rvert\right)$	500	$x_i\in[-10,10]$	0
Fixed-dimension Multimodal Benchmark Functions				
Name	Function	*Dim*	Range	f_{min}
Drop Wave	$f_7(x)=-\frac{1+cos\left(12\sqrt{x_1^2+x_2^2}\right)}{0.5\left(x_1^2+x_2^2\right)+2}$	2	$x_i\in[-5.12,5.12]$	-1
Storn's Chebyshev Polynomial Fitting Problem-$f_8(x)$		9	$x_i\in[-8192,8192]$	1
Happy Cat Function-$f_9(x)$		10	$x_i\in[-100,100]$	1

Table 3
Comparison results of unimodal benchmark functions.

Function		TLMFO	MFO	DE	GWO	PSO	SOA	GBO	AOA	RSA	Rank
f_1	Best	**0.00E+00**	8.74E+05	5.68E+03	4.17E-13	6.69E+03	7.54E-11	6.70E-253	4.84E-01	0.00E+00	1
	Std	0.00E+00	3.17E+04	1.96E+02	1.09E-12	3.24E+02	1.30E-08	0.00E+00	4.57E-02	0.00E+00	
f_2	Best	**8.02E-174**	2.07E+03	3.22E+104	3.06E-08	2.25E+64	2.75E-08	1.14E-125	2.30E-34	4.32E-64	1
	Std	7.18E-140	8.40E+01	3.22E+104	1.38E-08	4.80E+145	1.40E-07	3.27E-119	1.04E-03	8.28E-01	
f_3	Best	**0.00E+00**	2.55E+06	7.42E+06	5.88E+04	3.74E+05	1.26E+02	1.63E-200	1.36E+01	0.00E+00	1
	Std	0.00E+00	9.53E+05	9.23E+05	5.56E+04	1.59E+05	2.56E+04	0.00E+00	2.79E+01	0.00E+00	

Table 4
Comparison results of multi-modal benchmark functions.

Function		TLMFO	MFO	DE	GWO	PSO	SOA	GBO	AOA	RSA	Rank
f_4	Best	**0.00E+00**	6.09E+03	6.71E+03	4.73E-11	7.39E+03	5.46E-12	0.00E+00	0.00E+00	0.00E+00	1
	Std	0.00E+00	1.59E+02	5.99E+01	6.84E+00	2.78E+02	3.78E+00	0.00E+00	2.48E-06	0.00E+00	
f_5	Best	**8.88E-16**	2.00E+01	2.04E+01	2.59E-08	1.24E+01	2.00E+01	8.88E-16	6.30E-03	8.88E-16	1
	Std	0.00E+00	1.27E-01	1.01E-01	1.67E-08	1.90E-01	7.67E-05	0.00E+00	4.77E-04	0.00E+00	
f_6	Best	**7.52E-175**	7.20E+02	9.60E+02	3.49E-08	7.27E+02	3.04E-07	1.44E-125	1.34E-04	2.25E-147	1
	Std	9.56E-128	3.77E+01	1.65E+01	3.26E-03	5.06E+01	6.13E-04	3.34E-119	4.23E-04	1.47E-121	

Table 5
Comparison results of multi-modal benchmark functions.

Function		TLMFO	MFO	DE	GWO	PSO	SOA	GBO	AOA	RSA	Rank
f_7	Best	**−1**	−1	−1	−1	−1	−1	−1	−1	−1	1
	Std	0	3.24E-02	0	1.16E-02	0	0	0	0	−1	
f_8	Best	**3.47E+04**	5.48E+08	1.97E+09	4.91E+04	7.40E+04	4.14E+04	3.56E+04	5.06E+05	3.87E+04	1
	Std	1.03E+03	1.78E+10	4.42E+09	3.09E+08	6.05E+05	9.88E+07	8.39E+02	2.15E+09	6.70E+03	
f_9	Best	8.1615	19.9993	13.9171	20.3341	20.0717	20.1838	**3.8858**	19.9741	1.12E+01	2
	Std	4.16E+00	1.32E-01	1.17E+00	7.06E-02	1.01E-01	8.33E-02	2.9825	3.86E-02	4.98E+00	

algorithm is 0, and there are no outliers in the data. Compared with the large number of outliers in DE and MFO, the TLMFO algorithm is more robust. In summary, the TLMFO algorithm exhibits faster convergence speed, stronger optimization ability, and higher robustness in this type of benchmark function optimization.

Table 5 presents the results of the fixed-dimensional multi-modal functions. TLMFO ranks first among the three functions, and GBO achieves the optimal value on f_9. For f_9, from the variance plot in Fig. 13, we can see that GBO is an outlier at the optimal value, and TLMFO has better robustness. Similarly, TLMFO, MFO, and DE achieve the theoretical optimal value on f_7, but TLMFO

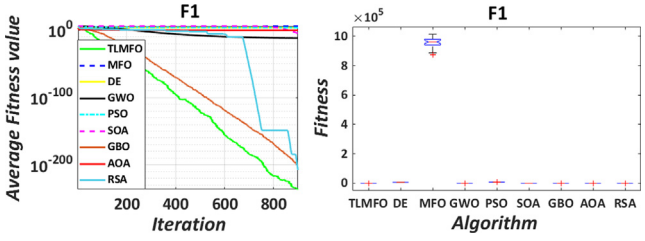

Fig. 7. D = 500, convergence curves, variance diagram for f_1.

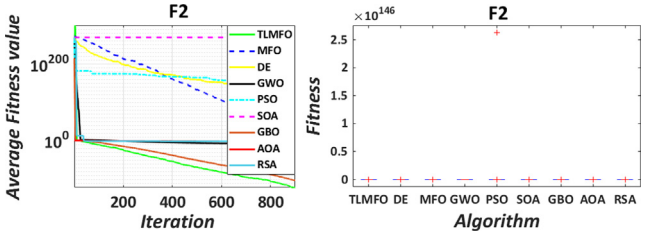

Fig. 8. D = 500, convergence curves, variance diagram for f_2.

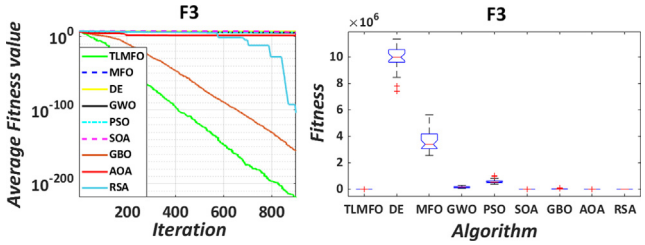

Fig. 9. D = 500, convergence curves, variance diagram for f_3.

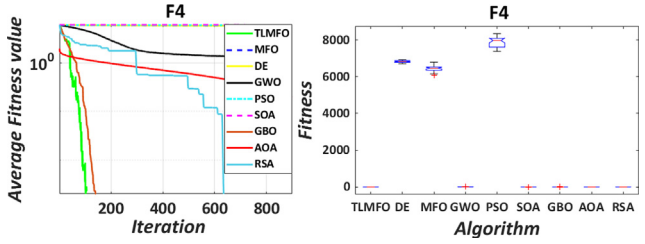

Fig. 10. D = 500, convergence curves, variance diagram for f_4.

has better results in the other tasks where DE and MFO perform poorly. Therefore, the TLMFO algorithm outperforms the DE and MFO algorithms in optimizing fixed-dimensional multimodal function problems.

Fig. 13–15 show all algorithms' mean convergence and variance plots on fixed-dimensional multimodal functions after 30 independent runs. The three plots show that TLMFO has higher convergence speed, accuracy, and robustness than other algorithms. The plots also show that the TLMFO algorithm has more potential for such problems than the DE algorithm.

5.1.4. Wilcoxon rank-sum test

We use the Wilcoxon rank-sum test to verify further the performance to compare the above experiments. We conduct statistical tests of TLMFO's performance against MFO, DE, GWO, PSO, AOA, SOA, GBO, and RSA, and obtain a *p*-value output at the end of the test. If the *p*-value is lower than 0.05, the difference between TLMFO and the comparison algorithm is statistically significant. If the *p*-value exceeds 0.05, the difference between TLMFO and the comparison algorithm is not statistically significant. The *NaN*value indicates that TLMFO is highly superior to the comparison algorithm. The following table shows the p-value of the statistical test:

Table 6 shows that the p-value of TLMFO on the test function f_1 to f_6 is much lower than 0.05, which confirms its superiority on unimodal and multimodal functions. In f_7-f_9, due to the lower dimensionality, the algorithms have similar convergence performance, resulting in some non-significant differences with different algorithms. However, it still demonstrates its effectiveness and

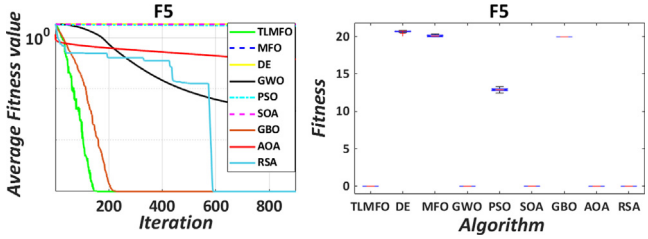

Fig. 11. D = 500, convergence curves, variance diagram for f_5.

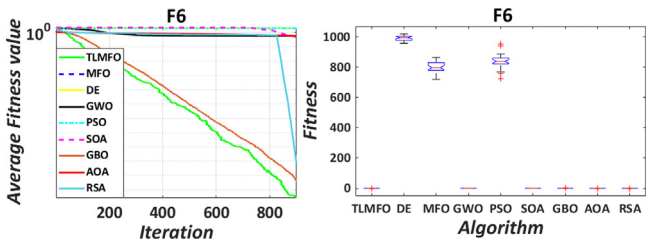

Fig. 12. D = 500, convergence curves, variance diagram for f_6.

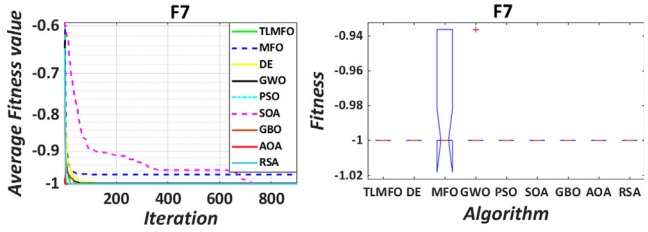

Fig. 13. D = 2, convergence curves, variance diagram for f_7.

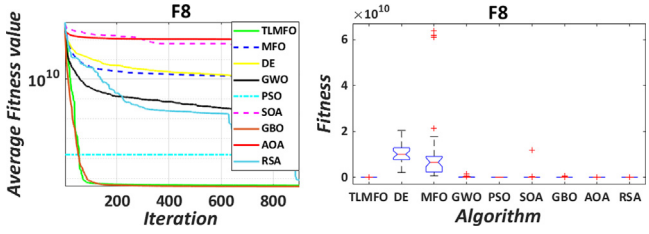

Fig. 14. D = 9, convergence curves, variance diagram for f_8.

robustness. Therefore, from the statistical perspective, we also verify our conclusions.

Through the summary analysis of the function optimization problem, TLMFO shows more potential than the DE and MFO algorithm in the benchmark function optimization problem, indicating its suitability for one-pixel attacks. In the next part, we will verify the application potential of TLMFO for one-pixel attacks in the AML field through experiments and detailed comparative analysis with DE.

5.2. One-pixel attack test

5.2.1. Model accuracy

We use the CIFAR10 dataset and load five CNN models, as shown in Table 7. We test all the images on each model and attack those correctly classified. We measure the perturbation strength by the number of pixels changed rather than the length of the perturbation vector. We compare the results of one-pixel, 3-pixel, and

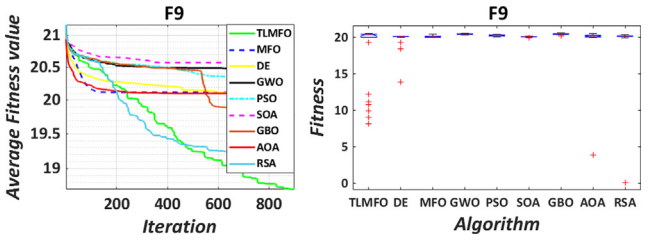

Fig. 15. D = 10, convergence curves, variance diagram for f_9.

5-pixel modifications. We focus on improving the success rate of the one-pixel attack algorithm and do not consider other attack strategies. Table 7 shows the accuracy and number of parameters for each model. Resnet (He et al., 2016) has the highest accuracy with the second largest number of parameters, while Lenet (Springenberg et al., 2015) has the lowest. We also plot the training loss and accuracy curves of Pure-cnn at Epoch = 100 (see Figure 16). The results are consistent with the data in Table 7.

5.2.2. One-pixel attack statistics analysis

We introduce **attack success_rate** metrics to evaluate the effectiveness of our attack method on the CIFAR-10 and ImageNet datasets. In an untargeted attack, the success rate is the percentage of perturbed images classified into other categories different from the original. In a targeted attack, it is the probability of perturbing an image into a specified target category. The success rate is the main criterion for judging whether an attack is successful. We perform targeted and untargeted attacks on the correctly classified images in the CIFAR-10 dataset for the five models following the steps in the attack phase above.

The experiment collects statistics on targeted and untargeted one-pixel attacks using TLMFO core, MFO core, and DE core for each pair of models. The data points include the success rate of attacking a given model, the number of pixels perturbed (1, 3, or 5 pixels), and the target category (for targeted or untargeted attacks). The data in bold and underlined indicate better performance.

We performed the attack tests in the same experimental environment to ensure fairness. This paper compared and validated the one-pixel attack results of the DE core with the existing experimental data from other researchers, and the results were consistent.

The experiment evaluated the success rates of DE and TLMFO's attacks on five networks using the CIFAR-10 dataset (Table 8). These results demonstrate the general effectiveness of this type of attack against different DNN models. Lenet was the most vulnerable network among the five for untargeted attacks, with the highest attack success rate for all three cores (DE, MFO, and TLMFO). In contrast, Pure-cnn was the most robust network among the five. TLMFO achieved the best attack success rate for targeted attacks

Table 7
Model parameters and training accuracy.

	Name	Accuracy	Param_count
1	Lenet	0.4395	62006
2	Pure_cnn	0.8513	1369738
3	Net_in_Net	0.9074	972658
4	Resnet	0.9231	470218
5	Capsnet	0.7982	12384576

on Lenet, Net-in-Net, and Resnet. On Capsnet and Pure-cnn, TLMFO showed attack success rates close to the optimal ones. We also found that DE and MFO cores had similar attack efficiency on one-pixel attacks. Although they achieved the best success rate on one network each, TLMFO generally outperformed them.

TLMFO demonstrates superior attack potential for targeted attacks on all classifiers except for the Capsnet classifier, which remains impervious to attack. As shown in Table 8, both MFO and DE have their respective strengths and weaknesses in terms of attack efficiency but still need to catch up when compared to our proposed method. In summary, our TLMFO-based attack exhibits improved performance in the one-pixel attack problem and is more effective in identifying target features that can lead to image misclassification.

Following the one-pixel attack experiment on the CIFAR10 dataset, we conducted a similar attack experiment on the ImageNet dataset. This experiment evaluated the effectiveness of minor modifications in fooling classification models on larger images. We performed one-pixel attack tests using three methods - TLMFO, MFO, and DE - against the AlexNet classifier. The results of these one-pixel attacks on the ImageNet dataset are presented in Table 9. TLMFO continues to exhibit superior attack potential compared to DE and MFO on larger images.

Moreover, the success rate of evolutionary computation-based attacks on the AlexNet network demonstrates that one-pixel attacks can be effectively extended to larger images and fool interconnected DNNs. However, it should be noted that larger images, such as 227 × 227 photos, are less susceptible to attack compared to smaller 32 × 32 CIFAR10 images. Optimizing one-pixel attacks using the TLMFO algorithm provides a practical and low-cost tool for launching attacks against neural networks.

5.2.3. Effect of the perturbation size on attack potential

To further assess the attack potential of the TLMFO-based one-pixel attack method, we conducted additional experiments to determine whether changes in the perturbation scale would affect the search efficiency of TLMFO without limiting the attack depth. For this purpose, we selected the same 500 CIFAR10 image samples used in the one-pixel attack experiment and increased the pixel value of the attack to 3 or 5. We then compared the perturbation capabilities of the three EAs against the classifier. The results of these experiments are presented in Tables 10 and 11.

Table 6
The p-values of the Wilcoxon rank-sum test over all runs ((TLMFO VS **), $p \succeq 0.05$ are underlined).

Function	MFO	DE	GWO	PSO	SOA	GBO	AOA	RSA
f_1	1.7344e-06	3.0123E-11	3.0123E-11	3.0123E-11	3.0123E-11	3.0123E-11	3.0123E-11	3.0123E-11
f_2	1.7344e-06	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11
f_3	1.7344e-06	3.0180E-11	3.0180E-11	3.0180E-11	3.0180E-11	3.0180E-11	3.0180E-11	3.0180E-11
f_4	1.7344e-06	1.2118E-12	1.2118E-12	1.2118E-12	1.2078E-12	NaN	5.6000E-03	1.2118E-12
f_5	1.7344e-06	1.2118E-12	1.2118E-12	1.2118E-12	4.6147E-13	NaN	1.2118E-12	1.2118E-12
f_6	1.7344e-06	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11	3.0199E-11
f_7	1.2207e-04	NaN	**3.3371E-01**	NaN	NaN	NaN	NaN	NaN
f_8	1.7344e-06	3.0199E-11	3.0199E-11	3.0199E-11	2.0338E-09	3.0199E-11	3.0199E-11	3.0199E-11
f_9	1.4412E-02	7.0000E-03	4.9818E-04	4.9818E-04	1.4932E-04	**1.0547E-01**	7.2884E-03	3.6147E-04

Table 8
Results of **0ne-pixel** attack on five networks with different core.

Model	Core	Target Category	Attack Success_rate
Lenet	DE	untarget	0.768
		target	1.444E-01
	MFO	untarget	0.716
		target	1.167E-01
	TLMFO	untarget	**0.794**
		target	**2.015E-01**
Pure_cnn	DE	untarget	**0.138**
		target	6.667E-02
	MFO	untarget	0.12
		target	6.444E-02
	TLMFO	untarget	0.12
		target	**8.178E-02**
Net-in-Net	DE	untarget	0.36
		target	1.000E-01
	MFO	untarget	0.36
		target	9.667E-02
	TLMFO	untarget	**0.384**
		target	**1.444E-01**
Resnet	DE	untarget	0.302
		target	1.122E-01
	MFO	untarget	0.306
		target	1.333E-01
	TLMFO	untarget	**0.312**
		target	**1.560E-01**
Capsnet	DE	untarget	0.126
		target	0.000E+00
	MFO	untarget	**0.167**
		target	0.000E+00
	TLMFO	untarget	0.162
		target	0.000E+00

By analyzing the results of the one-pixel attack experiment presented in Table 8, we observe that as the perturbation size increases, the probability of misclassification of multiple networks increases, with all three EAs exhibiting a similar upward trend. Compared to DE and MFO, TLMFO achieved higher attack success rates on all five classifiers and demonstrated superior attack potential overall. When the perturbation pixel value increased from 1 to 3, the attack success rate of all three methods improved. However, when the perturbation pixel value increased from 3 to 5, we observed that DE and MFO did not exhibit any improvement on some classifiers. These results suggest that the search potential of the DE and MFO algorithms may have been exhausted during the later stages of optimization. In contrast, TLMFO consistently demonstrated robustness as the number of pixels increased, with its attack success rate gradually increasing.

5.2.4. Wilcoxon signed-ranks test

To validate further the conclusion that the TLMFO-based one-pixel attack exhibits the best attack potential, as derived from our previous experiments, we employed the Wilcoxon signed-rank test to determine whether the differences between the three attack methods were statistically significant when perturbing the five classifiers.

Table 9
Results of one-pixel attack on AlexNet networks with ImageNet dataset.

Model Accuracy	Core	Attack Auccess_rate
57.3%	DE	16.04%
	MFO	17.17%
	TLMFO	**17.88%**

We used TLMFO, MFO, and DE to generate adversarial images for 50 CIFAR10 images and obtained the percentage of predicted classifications for each of the 50 groups through the classifier. The difference d_i between the performance scores of TLMFO and DE or MFO was calculated using the obtained ratios and sorted in ascending order. The p-value was then determined based on the rank of the difference variable. Our initial assumption was that there was no difference between the three attack methods. If $p < 0.05$, we would reject the null hypothesis. The results of the Wilcoxon signed-rank test are presented in Table 12. When conducting adversarial attack experiments on different classifiers and pixel values, the statistical test results for TLMFO compared to MFO and DE were far less than 0.05, indicating a significant difference between TLMFO and the other two methods. These results demonstrate the superior performance of the TLMFO-based one-pixel attack.

5.2.5. Comparison of attack effectiveness

To further evaluate the attack effectiveness and efficiency of our proposed attack method, we conducted a comparison experiment against the original iterative method (Su et al., 2019) and four other mainstream attack methods - iterative FGSM (Goodfellow et al., 2014), jacobian-based saliency map attack (1-pixel) (Papernot et al., 2016), SIOA, and LSA (Narodytska and Kasiviswanathan, 2017)- on the Kaggle CIFAR10 dataset. The experiment used the Network in Network (NIN) and VGG16 networks, which had been trained on the CIFAR-10 dataset. The results of this comparison are presented in Table 13.

When comparing the attack effectiveness of our proposed method against five previous works, our approach demonstrates a higher attack success rate when attacking only one pixel. As shown in Table 13, our proposed method still exhibits comparable

Table 10
Results of **3-pixel attack** on five networks with different core.

Model	Core	Target Category	Attack Success_rate
Lenet	DE	untarget	0.92
		target	1.667E-01
	MFO	untarget	0.92
		target	2.111E-01
	TLMFO	untarget	**0.93**
		target	**2.778E-01**
Pure_cnn	DE	untarget	0.436
		target	1.333E-01
	MFO	untarget	**0.44**
		target	1.367E-01
	TLMFO	untarget	0.43
		target	**1.667E-01**
Net-in-Net	DE	untarget	0.772
		target	1.444E-01
	MFO	untarget	0.75
		target	1.333E-01
	TLMFO	untarget	**0.802**
		target	**1.575E-01**
Resnet	DE	untarget	0.582
		target	1.500E-01
	MFO	untarget	0.612
		target	2.222E-01
	TLMFO	untarget	**0.622**
		target	**2.889E-01**
Capsnet	DE	untarget	0.296
		target	**4.444E-02**
	MFO	untarget	0.316
		target	1.889E-02
	TLMFO	untarget	**0.44**
		target	1.667E-02

Table 11
Results of **5-pixel attack** on five networks with different core.

Model	Core	Target Category	Attack Success_rate
Lenet	DE	untarget	0.922
		target	2.444E-01
	MFO	untarget	0.930
		target	2.577E-01
	TLMFO	untarget	**0.932**
		target	**3.778E-01**
Pure_cnn	DE	untarget	0.508
		target	1.889E-01
	MFO	untarget	0.513
		target	1.911E-01
	TLMFO	untarget	**0.526**
		target	**2.111E-01**
Net-in-Net	DE	untarget	0.786
		target	2.111E-01
	MFO	untarget	0.750
		target	1.967E-01
	TLMFO	untarget	**0.8**
		target	**3.444E-01**
Resnet	DE	untarget	0.586
		target	1.444E-01
	MFO	untarget	0.612
		target	2.333E-01
	TLMFO	untarget	**0.708**
		target	**2.889E-01**
Capsnet	DE	untarget	0.328
		target	4.444E-02
	MFO	untarget	0.382
		target	4.111E-02
	TLMFO	untarget	**0.404**
		target	**6.667E-02**

effectiveness to previous works that employ full (or more) pixel attacks, even under more restrictive conditions.

5.2.6. Perturbed images

To provide a more intuitive analysis of our experimental results, we present attack maps for 1, 3, and 5-pixel attacks on the five models. An attack is considered successful if the correct category differs from the perturbed category and unsuccessful if they are the same. For ease of observation, we have marked successful attacks in red. As shown in Fig. 17, there is a positive correlation between the perturbed pixel value and the attack success rate, corroborating our previous statistical analysis of the attacks. However, it should be noted that there are still instances of unsuccessful attacks. Additionally, our analysis revealed several interesting observations:

Table 12
The *p*-values of the Wilcoxon signed-ranks test over all runs.

Models	Pixel	TLMFO VS DE	TLMFO VS MFO
Lenet	1-pixel	2.13E-02	1.46E-04
	3-pixel	8.31E-04	1.40E-04
	5-pixel	1.00E-03	4.35E-05
Pure-cnn	1-pixel	5.20E-06	2.75E-05
	3-pixel	4.23E-07	6.08E-07
	5-pixel	2.31E-05	3.92E-06
Net-in-Net	1-pixel	9.60E-06	2.91E-06
	3-pixel	3.12E-05	6.31E-04
	5-pixel	2.60E-06	4.86E-06
Resnet	1-pixel	3.37E-07	1.54E-07
	3-pixel	7.21E-05	1.12E-04
	5-pixel	1.55E-04	2.30E-05
Capsnet	1-pixel	1.06E-04	2.91E-06
	3-pixel	8.35E-05	1.01E-04
	5-pixel	3.01E-05	3.65E-06

Table 13
Comparison of attack effectiveness.

Method	Success rate (%)	Number (percentage) of pixels	Network
FGSM	87.11	1024(100%)	NIN
	81.84	1024(100%)	VGG
LSA	70.48	33 (3.24%)	NIN
	75.29	30 (2.99%)	VGG
JSM	59.63	5 (0.48%)	NIN
	43.67	5 (0.48%)	VGG
ISOA	56.61	1 (0.098%)	NIN
	40.83	1 (0.098%)	VGG
DE	54.65	1 (0.098%)	NIN
	41.45	1 (0.098%)	VGG
TLMFO	59.84	1 (0.098%)	NIN
	43.48	1 (0.098%)	VGG

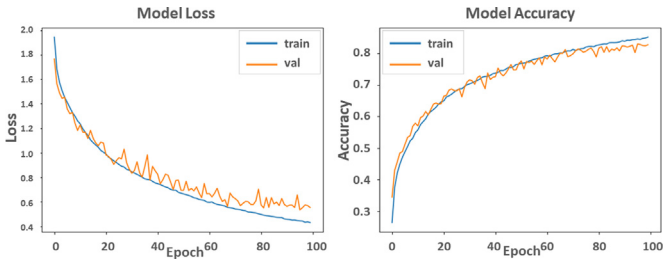

Fig. 16. Model loss curve and accuracy curve of Pure-cnn.

- An increase in the number of pixels does not necessarily guarantee an increase in the attack success rate. For example, when attacking the Pure-cnn model with 3 pixels, a horse was incorrectly classified as a deer, but when attacked with 5 pixels, it was correctly classified as a horse.
- The same image can be classified into different categories by perturbing pixels at different locations. For instance, when perturbing the Net-in-Net model with 1 and 3 pixels, a dog was incorrectly classified as a frog and a bear, respectively.

5.2.7. A use case in defect detection

Despite numerous security studies on ZDM, our literature review did not uncover any specific cases demonstrating the security threats faced by ZDM. To highlight the importance of the model and data security protection in the ZDM field, we employed a one-pixel attack to induce misclassification by the detection model.

When extracting copper in a mining area, electrolytic copper is used to removing it through adsorption onto the original steel plate via electrolysis. As more copper is adsorbed, the surface copper must be peeled off once it reaches a particular value. In the process of electrolytic operation, the identification model needs to be used to observe the drum on the surface of the copper plate occasionally, and alarm instructions need to be issued if obvious bubbles are identified. We employed a one-pixel attack to generate adversarial examples that fooled the defect detection recognition model into incorrectly identifying normal copper plates without bubbles on their surface as abnormal. This resulted in high labor costs when many normal copper plates were erroneously flagged during anomaly detection.

These results demonstrate that if data and model security cannot be ensured, the practicality of ZDM schemes becomes a critical issue that warrants further investigation.

6. Conclusion

This paper discusses the challenges of model and data security in defect detection and investigates the potential impact of adver-

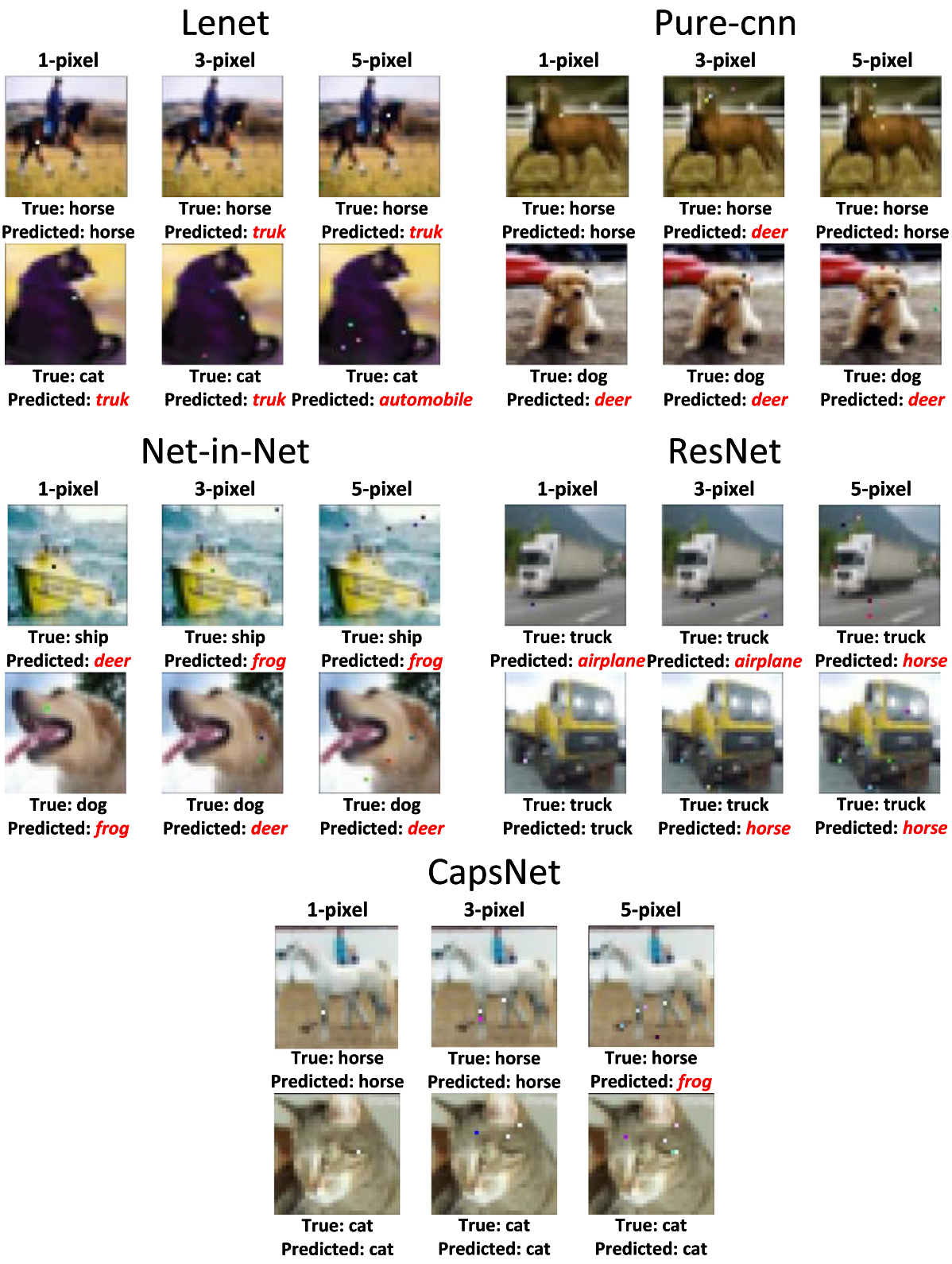

Fig. 17. One-pixel attack on the CIFAR10 dataset.

sarial attacks, based on one-pixel attack technology, on the accuracy of defect classification. We introduced a teaching and learning mechanism into the MFO algorithm to develop the TLMFO algorithm, which exhibits enhanced learning capabilities. Simulation results using benchmark functions demonstrate that TLMFO possesses superior robustness and optimization accuracy. Experiments on the CIFAR10 and ImageNet datasets confirm that TLMFO-based one-pixel attacks can induce classification errors in deep learning models at minimal cost without requiring access to the model's internal information. Furthermore, comparing attack effectiveness on the Kaggle CIFAR10 dataset reveals that one-pixel attacks can still produce misclassification results even under

more restrictive conditions. A case study in defect detection highlights the importance of model and data security protection in the ZDM field. Our findings indicate that adversarial attacks can compromise defect detection accuracy, thereby increasing the cost of industrial manufacturing and reducing cost-effectiveness. The paper aims to promote safety awareness in the development of industrial 5.0 and zero-defect production systems and to inspire further research in this area.

In the future, we will continue to pay attention to the threats posed by security issues to the sustainable development of Industry 5.0 and explore a series of security challenges brought by advanced intelligent technologies such as data security, model security, and private security. We will take offense as defense and continue to tap into the attack potential of adversarial attacks in adversarial machine learning to improve the defense capability of machine learning. At the same time, we will research the explainability of deep neural networks to realize the visualization, quantification, and verification of model decisions. Ultimately, our goal is to achieve security and trust in the deployment model of Industry 5.0.

Declaration of Competing Interest

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

Acknowledgements

This work was supported in part by The 4th project "Research on the Key Technology of Endogenous Security Switches" (2020YFB1804604) of the National Key R&D Program, the Special Fund for Transformation of Scientific and Technological Achievements of Jiangsu Province (No. BA2022011), the Special Fund for transformation and upgrading of Industrial and information Industry of Jiangsu Province (Tackling and Industrialization of Threat Detection and response System for Industrial Internet Terminals), the National Natural Science Foundation of P.R. China (62272244) and the National Nature Science Foundation of P.R. China under grant #62302236.

References

Abualigah, Laith et al., 2022. Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer. Expert Syst. Appl. 191, 116158.

Abualigah, Laith, Diabat, Ali, Mirjalili, Seyedali, Elaziz, Mohamed Abd, Gandomi, Amir H., 2021. The Arithmetic Optimization Algorithm. Comput. Methods Appl. Mech. Eng. 376, 113609. https://doi.org/10.1016/j.cma.2020.113609. ISSN 0045-7825, https://doi.org/10.1016/j.cma.2020.113609.

Ahmadianfar, Iman, Bozorg-Haddad, Omid, Chu, Xuefeng, 2020. Gradient-based optimizer: A new metaheuristic optimization algorithm. Inf. Sci. 540, 131–159. https://doi.org/10.1016/j.ins.2020.06.037. ISSN 0020-0255.

Akundi, Aditya, Euresti, Daniel, Luna, Sergio, Ankobiah, Wilma, Lopes, Amit, Edinbarough, Immanuel, 2022. State of Industry 5.0–analysis and identification of current research trends. Appl. Syst. Innovat. 5 (1), 27. https://doi.org/10.3390/asi5010027.

Alatalo, J., Korpihalkola, J., Sipola, T., Kokkonen, T., 2022. Chromatic and spatial analysis of one-pixel attacks against an image classifier. In: Koulali, MA., Mezini, M. (Eds.), Networked Systems. NETYS 2022. Lecture Notes in Computer Science, vol 13464. Springer, Cham. https://doi.org/10.1007/978-3-031-17436-0_20.

Alberti, M., Pondenkandath, V., Wursch, M., Bouillon, M., Seuret, M., Ingold, R., Liwicki, M., 2018. Are you tampering with my data? In: Proceedings of the European Conference on Computer Vision (ECCV) Workshops, pp. 1–18. https://link.springer.com/conference/eccv.

Chen, D., Xu, R., Han, B., 2019. Patch selection denoiser: an effective approach defending against one-pixel attacks. In: Gedeon, T., Wong, K., Lee, M. (Eds.), Neural Information Processing. ICONIP 2019. Communications in Computer and Information Science, vol. 1143. Springer, Cham. https://doi.org/10.1007/978-3-030-36802-9_31.

Das, S., Suganthan, P.N., 2011. Differential evolution: a survey of the state-of-the-art. IEEE Trans. Evol. Comput. 15 (1), 4–31. https://doi.org/10.1109/TEVC.2010.2059031.

Dhiman, Gaurav, Kumar, Vijay, 2019. Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. Knowl.-Based Syst. 165, 169–196. https://doi.org/10.1016/j.knosys.2018.11.024. ISSN 0950-7051.

Dunderdale, C., Brettenny, W., Clohessy, C., van Dyk, E.E., 2020. Photovoltaic defect classification through thermal infrared imaging using a machine learning approach. Prog. Photovolt. Res. Appl. 28, 177–188. https://doi.org/10.1002/pip.3191.

Faris, H., Aljarah, I., Al-Betar, M.A., et al., 2018. Grey wolf optimizer: a review of recent variants and applications. Neural. Comput. Appl. 30, 413–435. https://doi.org/10.1007/s00521-017-3272-5.

Ghobakhloo, Morteza, 2020. Industry 4.0, digitization, and opportunities for sustainability. J. Clean. Prod. 252, 119869. https://doi.org/10.1016/j.jclepro.2019.119869. ISSN 0959-6526.

Goodfellow, Ian J., Jonathon Shlens, Christian Szegedy, 2014. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572.

Guo, Y., Yin, P., Huang, D., 2023. One-pixel attack for continuous-variable quantum key distribution systems. Photonics 10, 129. https://doi.org/10.3390/photonics10020129.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27–30, 2016, IEEE Computer Society, pp. 770–778, https://doi.org/10.1109/CVPR.2016.90.

Kennedy, J., Eberhart, R., 1995. Particle swarm optimization. In: Proceedings of ICNN'95 - International Conference on Neural Networks, vol. 4, pp. 1942–1948, https://doi.org/10.1109/ICNN.1995.488968.

Khan, U., Woods, W., Teuscher, C., 2019. Exploring and expanding the one-pixel attack. https://archives.pdx.edu/ds/psu/28613.

Korpihalkola, J., Sipola, T., Puuska, S., Kokkonen, T., 2021, August. One-pixel attack deceives computer-assisted diagnosis of cancer. In: 2021 4th International Conference on Signal Processing and Machine Learning, pp. 100–106. https://doi.org/10.1145/3483207.3483224.

Korpihalkola, J., Sipola, T., Kokkonen, T., 2021. Color-optimized one-pixel attack against digital pathology images. In: 2021 29th Conference of Open Innovations Association (FRUCT), pp. 206–213, https://doi.org/10.23919/FRUCT52173.2021.9435562.

Leng, Jiewu, Sha, Weinan, Wang, Baicun, Zheng, Pai, Zhuang, Cunbo, Liu, Qiang, Wuest, Thorsten, Mourtzis, Dimitris, Wang, Lihui, 2022. Industry 5.0: Prospect and retrospect. J. Manuf. Syst. 65, 279–295. https://doi.org/10.1016/j.jmsy.2022.09.017. ISSN 0278-6125.

Li, M., Wang, X., Cui, Q., et al., 2023. Adversarial data hiding with only one pixel. Infr. Proces. Manage. 60 (2), 103222.

Maddikunta Praveen Kumar Reddy, Pham Quoc-Viet, Prabadevi, B., Deepa, N., Dev Kapal, Gadekallu Thippa Reddy, Ruby Rukhsana, Liyanage Madhusanka, 2022. Industry 5.0: A survey on enabling technologies and potential applications. J. Ind. Infr. Integrat. 26, 100257, ISSN 2452-414X, https://doi.org/10.1016/j.jii.2021.100257.

Mao, X., Chen, Y., Wang, S., Su, H., He, Y., Xue, H., 2021. Composite adversarial attacks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 10, pp. 8884–8892. https://doi.org/10.1609/aaai.v35i10.17075.

Masci, J., Meier, U., Ciresan, D., Schmidhuber, J., Fricout, G., 2012. Steel defect classification with max-pooling convolutional neural networks. In: The 2012 International Joint Conference on Neural Networks (IJCNN), pp. 1–6. https://doi.org/10.1109/IJCNN.2012.6252468.

Mirjalili, Seyedali, 2015. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. Knowl.-Based Syst. 89, 228–249. https://doi.org/10.1016/j.knosys.2015.07.006. ISSN 0950-7051.

Mohammadi-Balani, Abdolkarim et al., 2021. Golden eagle optimizer: A nature-inspired metaheuristic algorithm. Comput. Ind. Eng. 152, 107050.

Mohammadi-Balani, Abdolkarim, Nayeri, Mahmoud Dehghan, Azar, Adel, Taghizadeh-Yazdi, Mohammadreza, 2021. Golden eagle optimizer: A nature-inspired metaheuristic algorithm. Comput. Ind. Eng. 152, 107050. https://doi.org/10.1016/j.cie.2020.107050. ISSN 0360-8352.

Narodytska, Nina, Kasiviswanathan, Shiva Prasad, 2017. Simple black-box adversarial attacks on deep neural networks. CVPR Workshops 2, 2.

Nayak, J., Vakula, K., Dinesh, P., Naik, B., 2020. Moth flame optimization: developments and challenges up to 2020. In: Das, A., Nayak, J., Naik, B., Dutta, S., Pelusi, D. (Eds.), Computational Intelligence in Pattern Recognition. Advances in Intelligent Systems and Computing, vol. 1120. Springer, Singapore. https://doi.org/10.1007/978-981-15-2449-3_40.

Nguyen, T.P., Choi, S., Park, S.J., et al., 2021. Inspecting Method for Defective Casting Products with Convolutional Neural Network (CNN). Int. J. Precis. Eng. Manuf.-Green Tech. 8, 583–594. https://doi.org/10.1007/s40684-020-00197-4.

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A., Limitations, The, 2016. The limitations of deep learning in adversarial settings. 2016 IEEE European Symposium on Security and Privacy (EuroS&P), Saarbruecken, Germany 2016, 372–387. https://doi.org/10.1109/EuroSP.2016.36.

Park, J.K., Kwon, B.K., Park, J.H., et al., 2016. Machine learning-based imaging system for surface defect inspection. Int. J. Precis. Eng. Manuf.-Green Tech. 3, 303–310. https://doi.org/10.1007/s40684-016-0039-x.

Paul, R., Schabath, M., Gillies, R., Hall, L., Goldgof, D., 2020. Mitigating adversarial attacks on medical image understanding systems. In: 2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI), pp. 1517–1521, https://doi.org/10.1109/ISBI45749.2020.9098740.

Powell, Daryl, Magnanini, Maria Chiara, Colledani, Marcello, Myklebust, Odd, 2022. Advancing zero defect manufacturing: A state-of-the-art perspective and future

research directions. Comput. Ind. 136. https://doi.org/10.1016/j.compind.2021.103596. ISSN 0166-3615.

Psarommatis, F., Kiritsis, D., 2018. A Scheduling Tool for Achieving Zero Defect Manufacturing (ZDM): a conceptual framework. In: Moon, I., Lee, G., Park, J., Kiritsis, D., von Cieminski, G. (Eds.), Advances in Production Management Systems. Smart Manufacturing for Industry 4.0. APMS 2018. IFIP Advances in Information and Communication Technology, vol 536. Springer, Cham. https://doi.org/10.1007/978-3-319-99707-0_34.

Qi, L. et al., 2022a. Privacy-aware data fusion and prediction for smart city services in edge computing environment. In: 2022 IEEE International Conferences on Internet of Things (iThings) and IEEE Green Computing & Communications (GreenCom) and IEEE Cyber, Physical & Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics), pp. 9–16, https://doi.org/10.1109/iThings-GreenCom-CPSCom-SmartData-Cybermatics55523.2022.00043.

Qi, L., Lin, W., Zhang, X., Dou, W., Xu, X., Chen, J., 2022b. A correlation graph based approach for personalized and compatible web APIs recommendation in mobile APP development. In: IEEE Transactions on Knowledge and Data Engineering, https://doi.org/10.1109/TKDE.2022.3168611.

Quan, W., Nagothu, D., Poredi, N., Chen, Y., 2021. Cripi: an efficient critical pixels identification algorithm for fast one-pixel attacks. In: Sensors and Systems for Space Applications XIV, vol. 11755, SPIE, pp. 83–99, https://doi.org/10.1117/12.2581377.

Raheja, Jagdish Lal, Kumar, Sunil, Chaudhary, Ankit, 2013. Fabric defect detection based on GLCM and Gabor filter: A comparison. Optik 124 (23), 6469–6474. https://doi.org/10.1016/j.ijleo.2013.05.004. ISSN 0030-4026.

Rao, R.V., Savsani, V.J., Vakharia, D.P., 2011. Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. Comput. Aided Des. 43 (3), 303–315. https://doi.org/10.1016/j.cad.2010.12.015. ISSN 0010-4485.

Sinha Shubham, Saranya, S.S., 2021. One Pixel Attack Analysis Using Activation Maps. Annals of the Romanian Society for Cell Biology, 8397–8404. Retrieved from https://www.annalsofrscb.ro/index.php/journal/article/view/2382.

Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.A., 2015. Striving for simplicity: The all convolutional net. In: Bengio, Y., LeCun, Y. (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Workshop Track Proceedings. URL: http://arxiv.org/abs/1412.6806.

Su, J., Vargas, D.V., Sakurai, K., 2019. One pixel attack for fooling deep neural networks. IEEE Trans. Evol. Comput. 23 (5), 828–841. https://doi.org/10.1109/TEVC.2019.2890858.

Vargas, D.V., Su, J., 2020. Understanding the one-pixel attack: Propagation maps and locality analysis. In: CEUR Workshop Proceedings, vol. 2640. CEUR-WS.

Wang, W., Sun, J., Wang, G., 2020. Visualizing one pixel attack using adversarial maps. In: 2020 Chinese Automation Congress (CAC), pp. 924–929. https://doi.org/10.1109/CAC51589.2020.9327603.

Wang, P., Cai, Z., Kim, D., Li, W., 2021. Detection mechanisms of one-pixel attack. Wireless Commun. Mobile Comput. 2021. https://doi.org/10.1155/2021/8891204.

Xu, X. et al., 2022a. Game theory for distributed IoV task offloading with fuzzy neural network in edge computing. IEEE Trans. Fuzzy Syst. 30 (11), 4593–4604. https://doi.org/10.1109/TFUZZ.2022.3158000.

Xu, X. et al., 2021. Edge server quantification and placement for offloading social media services in industrial cognitive IoV. IEEE Trans. Industr. Inf. 17 (4), 2910–2918. https://doi.org/10.1109/TII.2020.2987994.

Xu Han, Li Yaxin, Jin Wei, Tang Jiliang, 2020. Adversarial attacks and defenses: frontiers, advances and practice. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20). Association for Computing Machinery, New York, NY, USA, pp. 3541–3542. https://doi.org/10.1145/3394486.3406467.

Xu Xiaolong, Fang Zijie, Zhang Jie, He Qiang, Yu Dongxiao, Qi Lianyong, Dou Wanchun, 2021. Edge Content Caching with Deep Spatiotemporal Residual Network for IoV in Smart City. ACM Trans. Sen. Netw. 17, 3, Article 29 (August 2021), 33 pages. https://doi.org/10.1145/3447032.

Xu, X., Gu, J., Yan, H., Liu, W., Qi, L., Zhou, X., 2022b. Reputation-aware supplier assessment for blockchain-enabled supply chain in industry 4.0. In: IEEE Transactions on Industrial Informatics, https://doi.org/10.1109/TII.2022.3190380.

Xu, X., Tian, H., Zhang, X., Qi, L., He, Q., Dou, W., 2022c. DisCOV: Distributed COVID-19 Detection on X-ray images with edge-cloud collaboration. IEEE Trans. Serv. Comput. 15 (3), 1206–1219. https://doi.org/10.1109/TSC.2022.3142265.

Xu Xiaolong, Liu Wentao, Zhang Yulan, Zhang Xuyun, Dou Wanchun, Qi Lianyong, Bhuiyan Md Zakirul Alam. 2022d. PSDF: Privacy-aware IoV Service deployment with federated learning in cloud-edge computing. ACM Trans. Intell. Syst. Technol. 13(5), 22. https://doi.org/10.1145/3501810, Article 70.

Yang, L., Jiang, H., 2021. Weld defect classification in radiographic images using unified deep neural network with multi-level features. J. Intell. Manuf. 32, 459–469. https://doi.org/10.1007/s10845-020-01581-2.

Zhang, L., Jing, J., Zhang, H., 2015. Fabric defect classification based on LBP and GLCM. J. Fiber Bioeng. Informat. 8 (1), 81–89.

Zhang, Y.D., Wang, W., Zhang, X., et al., 2022. Secondary pulmonary tuberculosis recognition by 4-direction varying-distance GLCM and fuzzy SVM. Mobile Netw. Appl.. https://doi.org/10.1007/s11036-021-01901-7.

Zhang, Yuqing, Xie, Min, He, Yihai, Han, Xiao, 2022. Capability-based remaining useful life prediction of machining tools considering non-geometry and tolerancing features with a hybrid model. Int. J. Prod. Res., 1–17.

Zhou, T., Agrawal, S., Manocha, P., 2022. Optimizing one-pixel black-box adversarial attacks. arXiv preprint arXiv:2205.02116.

Zou, Feng, Chen, Debao, Qingzheng, Xu., 2019. A survey of teaching–learning-based optimization. Neurocomputing 335, 366–383. https://doi.org/10.1016/j.neucom.2018.06.076. ISSN 0925-2312.