Poisoning and Evasion Attacks Against Deep Learning Algorithms in Autonomous Vehicles

Wenbo Jiang, *Student Member, IEEE*, Hongwei Li, *Senior Member, IEEE*, Sen Liu, *Student Member, IEEE*, Xizhao Luo, and Rongxing Lu, *Senior Member, IEEE*

***Abstract*—With the ongoing development and improvement of deep learning technology, autonomous vehicles (AVs) have made tremendous progress in recent years. Despite its great potential, AV supported by deep learning technology still faces numerous security threats, which prevent AV from being putting into large-scale practice. Aiming at this challenging situation, in this paper, we would like to exploit two attacks against deep learning algorithms in traffic sign recognition system by leveraging particle swarm optimization. Specifically, we first exploit the PAPSO (poisoning attack with particle swarm optimization) which focuses on the training process of the deep learning algorithms in the traffic sign recognition system, i.e., the attacker injects crafted samples into the training dataset, causing a reduction in classification accuracy of the traffic sign recognition system. Then, we also explore the EAPSO (evasion attack with particle swarm optimization) which on the other hand focuses on the interference process of the deep learning algorithms, i.e., the attacker adds some hardly perceptible perturbations to the targeted test sample, leading to a misclassification on it. Extensive experiments are conducted to shed light on the effectiveness of our attacks, and some corresponding defense strategies are also presented.**

***Index Terms*—Autonomous vehicles, traffic sign recognition, deep learning, evasion attack, poisoning attack.**

I. Introduction

THE pioneer study on autonomous vehicles was done by CMU Navlab Group in 1984. Since then, considerable attention [1]–[10] has been paid to the field and it has become an important research trend of vehicular technology. As deep learning technology continues to evolve and improve, autonomous vehicles have gained impressive advances. For instance, in 2010, Google developed an autonomous car that could change or maintain lanes. GM developed the EN-V series of smart cars, which used the Internet of Vehicles technology to achieve automatic driving and parking; in 2017, Tesla launched the Autopilot 8.1 system, which greatly enhanced the performance of autonomous cars.

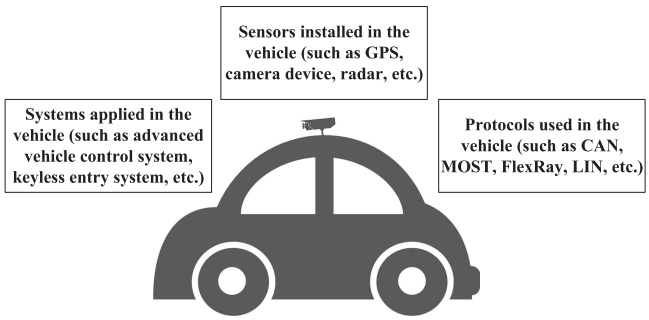


Fig. 1. Three attack surfaces of autonomous vehicle.

Despite these dramatic improvements, there is a growing recognition that autonomous vehicles expose numerous new vulnerabilities. The threats surrounding autonomous vehicles can be mainly divided into three types as illustrated in Fig. 1. For instance, Zeng *et al.* proposed an attack against the GPS [11], in which they proved that the navigation system will navigate the car to a wrong direction by shifting the GPS location slightly. Petit *et al.* [12] confirmed in their work that the MobilEye C2-270 could be confused by utilizing a laser light or LED matrix and Yan *et al.* [13] demonstrated that Tesla Model S automobile's cameras were vulnerable to the same attack; Keyless entry system is threatened by jamming attacks [14], replay attacks and relay attacks [15]; and the protocols used in the autonomous vehicle also contain many security flaws [16].

Nevertheless, little attention has been paid to the safety of deep learning algorithms in autonomous driving. In fact, autonomous vehicles heavily rely on deep learning algorithms to recognize objects and control the vehicle. For example, traffic sign recognition is an important application of deep learning in autonomous vehicles. It employs deep learning algorithms to classify the traffic sign image captured by the camera device and uses the intelligent control system to control the vehicle according to the classification results. However, deep learning is vulnerable to many malicious attacks [17]–[33]. **Poisoning attack** is performed during the training process of deep learning,

Manuscript received January 8, 2020; revised February 20, 2020; accepted February 22, 2020. Date of publication March 2, 2020; date of current version April 16, 2020. This work was supported in part by the National Key R&D Program of China under Grants 2017YFB0802300 and 2017YFB0802000, in part by the National Natural Science Foundation of China under Grants 61972454, 61802051, 61772121, 61728102, and 61472065, in part by the Peng Cheng Laboratory Project of Guangdong Province under Grant PCL2018KP004, and in part by the Guangxi Key Laboratory of Cryptography and Information Security under Grant GCIS201804. The review of this article was coordinated by Dr. Z. Ma. *(Corresponding author: Hongwei Li.)*

Wenbo Jiang and Sen Liu are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: 364730959@163.com; 893551724@qq.com).

Hongwei Li is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China, and also with the Peng Cheng Laboratory, Shenzhen 518000, China (e-mail: hongweili@uestc.edu.cn).

Xizhao Luo is with the School of Computer Science and Technology, Soochow University, Suzhou 215006, China (e-mail: xzluo@suda.edu.cn).

Rongxing Lu is with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: rlu1@unb.ca).

Digital Object Identifier 10.1109/TVT.2020.2977378

in which the attacker injects malicious samples to the training dataset, leading to a reduction in prediction accuracy of the learned model. For example, handwritten digit recognition [17], PDF malware detection [21] and recommendation systems [24] may be misled via some carefully forged malicious samples. On the other hand, **evasion attack** focuses on the predicting process of deep learning, in which the attacker generates an adversarial example and the classifier may make a wrong classification on it. For example, a misclassification may be made through adding imperceptible or hardly perceptible perturbations to the normal traffic sign image [30]. A spam may be classified as a normal e-mail by adding several positive words into the spam [31].

In this paper, we first exploit the PAPSO (poisoning attack with particle swarm optimization) and the EAPSO (evasion attack with particle swarm optimization) against deep learning algorithms. After that, we perform the PAPSO and the EAPSO on traffic sign recognition system to demonstrate the effectiveness of our attacks. Finally, we summarize some defense strategies against the two attacks. Specifically, our contributions of this paper can be elaborated in the following four aspects:

- First, we exploit the PAPSO, which degrades the performance of the learned model by injecting crafted samples into the training dataset, and we employ the particle swarm optimization to optimize the poisoned sample to gain better attack effects.
- Second, we explore the EAPSO, which attacks the interference process of the deep learning model by adding some hardly perceptible perturbations to a normal sample and causing a misclassification of it, and we utilize the particle swarm optimization to optimize the adversarial sample to improve the effect of attack.
- Third, we are the first to consider both poisoning attack and evasion attack against deep learning algorithms in traffic sign recognition system. Our two attacks are universally applicable to all deep learning algorithms or machine learning algorithms and they can be implemented through the similar attack methodology. Because these two attacks are both black-box attacks, they are more valuable in practice.
- Finally, we perform an extensive experimental evaluation on our attacks. Concretely, we first analyze the impact of hyper-parameters on attack effects and then evaluate the performance of the learned model in the case of multiple poisoned samples. Experimental results suggest that only 10% poisoned samples are enough to decrease classification accuracy of the learned model to about 33%; the probability that the targeted test sample is classified correctly can be reduced to about 22% by adding a perturbation of 12.5% of the feature value.

The remainder of this paper is organized as follows. We start by describing the details of the threat model in Section II. Then, we present the attack methodology in Section III. After that, we explain on how to implement our attacks in autonomous vehicles in Section IV. Experimental analysis is carried out in Section V, followed by related work and defense strategies in Section VII and Section VI, respectively. Finally, we draw some conclusions in Section VIII.

II. Threat Model

To make a deeper investigation on the attack methodology of attackers, it is crucial for us to know the knowledge and the capability of the attacker. Thus, in this section, we use the threat model to describe the characteristics of the attacker, which mainly consists of the following aspects.

A. Attacker's Goal

Attacker's goal can be defined from the following three different perspectives:

- **Security violation:** This characteristic defines the desired security violation of the attacker and can be mainly divided into three categories: *integrity* violation, the attacker has an arbitrary objective such as causing a specific misclassification on the targeted test sample; *availability* violation, the attacker's goal is to make the functionality of the learning model unavailable by decreasing the classification accuracy; *privacy* violation, the attacker's goal is to compromise private information about the system.
- **Attack specificity:** According to attack specificity, attacker's goal can be divided into two categories: *targeted* attack, the attacker's goal is to cause a misclassification of the targeted sample; *indiscriminate* attack, the attack's goal is to cause a misclassification of any sample.
- **Error specificity:** According to error specificity, attacker's goal falls into two categories: *specific* attack, the attacker's goal is to make the classifier classify the targeted sample to a specific wrong class; *generic* attack, the attacker's goal is to make the classifier classify the targeted sample as any other class.

In this work, we consider two kinds of attacker's goals: the attacker who carries out poisoning attack focuses on decreasing the classification accuracy of the learned model; and the attacker who implements evasion attack aims at increasing the probability that the targeted test sample is misclassified.

B. Attacker's Knowledge

Attacker's knowledge of the targeted model can be mainly categorized into three levels:

- **Perfect-Knowledge:** it is also referred to as white-box attack, which means everything about the targeted model is known to the adversary. Although this scenario may be impractical in the real world, it provides an opportunity to evaluate the machine learning systems in a worst-case.
- **Limited-Knowledge:** it is also referred to as gray-box attack, which means the adversary knows some components of the targeted model, but not completely.
- **Zero-Knowledge:** it is also referred to as black-box attack, which means the adversary knows nothing about the learning model.

In this work, we assume the attacker has zero-knowledge of the targeted model, but he has the capability of using the model and knowing the test accuracy of the targeted model.

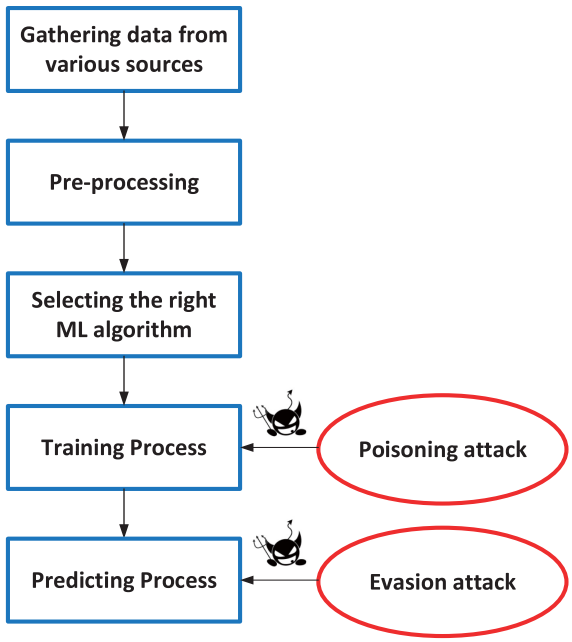


Fig. 2. Different processes targeted by poisoning attack and evasion attack.

TABLE I
NOTATIONS AND DESCRIPTION

Notations	Description
$\boldsymbol{v_i}$	velocity of the ith particle in PSO
$\boldsymbol{p_i}$	position of the ith particle in PSO
r_1, r_2	random numbers in $(0, 1)$
c_1, c_2	acceleration factors
α	inertia weight in PSO
$(\boldsymbol{x_n}, y_n)$	the normal sample and its label (unmodified data)
$(\boldsymbol{x_p}, y_n)$	the poisoned sample and its label
$(\boldsymbol{x_t}, y_t)$	the targeted sample and its label (unmodified data)
$(\boldsymbol{x_a}, y_t)$	the adversarial sample and its label
M_1, M_2	the number of particles in the swarm
$\boldsymbol{p_i^*}$	the best position of the ith particle
$\boldsymbol{p_{gb}}$	the best position of the whole swarm
T_1, T_2	the number of iteration
D_{tr}	the original training dataset
D_p	the optimal set of poisoned samples

C. Attacker's Capability

Attacker's capability means the ability of an attacker to control the training dataset or test sample. In this work, we assume that the attacker who carries out the poisoning attack has the ability of injecting corrupted samples into the training dataset and the attacker who implements evasion attack has the ability of adding perturbations to the targeted test sample.

III. Attack Methodology

In this section, we present our attack methodology. First, we begin by presenting a brief introduction to particle swarm optimization. After that, we propose the PAPSO (poisoning attack with particle swarm optimization) and the EAPSO (evasion attack with particle swarm optimization), respectively. The different processes targeted by poisoning attack and evasion attack are shown in Fig. 2. Also, the notations and symbols used in this work are listed in Table I.

A. PSO: Particle Swarm Optimization

Particle swarm optimization algorithm was originally presented by Eberhart and Kennedy in 1995 [34], whose idea was derived from birds' group foraging behavior. Specifically, in the PSO, the first step of the PSO is to generate numerous particles, which are randomly distributed in the solution space. Each particle is a possible solution in the solution space and has two properties: velocity $\boldsymbol{v_i}$ and position $\boldsymbol{p_i}$. The best position that the particle has experienced is denoted by $pbest$ and the best position that the whole group have experienced is denoted by $gbest$. In practice, the goodness of the particles is evaluated by the fitness value determined by the optimization problem. Each particle adjusts its velocity and updates its position based on $pbest$ and $gbest$, thereby generating a new generation of groups. Through information sharing among particles in the whole group and constantly update, the optimal solution can be found after several rounds of iterations.

Suppose that the optimization process is processed in N-dimensional space, the position of the ith particle can be represented as:

$$\boldsymbol{p_i} = (p_{i1}, p_{i2}, \ldots, p_{iN})^T \tag{1}$$

the velocity of the ith particle can be expressed as:

$$\boldsymbol{v_i} = (v_{i1}, v_{i2}, \ldots, v_{iN})^T \tag{2}$$

The update process of the ith particle is presented in Eq. (3) and Eq. (4):

$$v_{id}^{k+1} = \alpha v_{id}^k + c_1 r_1^k (pbest_{id}^k - p_{id}^k) + c_2 r_2^k (gbest_d^k - p_{id}^k) \tag{3}$$

$$p_{id}^{k+1} = p_{id}^k + v_{id}^{k+1} \tag{4}$$

where v_{id}^k represents the velocity of the dth-dimension of the ith particle in the kth iteration. p_{id}^k is the current position of the dth-dimension of the ith particle in the kth iteration. $pbest_{id}^k$ is the best position of the ith particle has experienced in the dth-dimension and the kth iteration. $gbest_d^k$ is the best position of the whole group has experienced in the dth-dimension and the kth iteration. α is a non-negative number, called the inertia weight, which reflects the impact of the past movement state of the particle on its current behavior. r_1, r_2 are two random numbers in (0,1). c_1, c_2 are acceleration factors. Note that, in this work, c_1, c_2 are both set to 2 and α is set to 0.8.

B. PAPSO: Poisoning Attack With Particle Swarm Optimization

Regarding PAPSO, the adversary's goal is to decrease the classification accuracy of the learned model through injecting poisoned data to the training dataset. In order to evade the outlier detection, the poisoned sample must be as stealthy as possible. Thus, our strategy is to add imperceptible or hardly perceptible perturbations to the normal sample and keep its label unchanged. Our initialization approach is to choose a normal sample and initialize random perturbations. The initialization algorithm of the PAPSO will be presented in Algorithm 1.

Algorithm 1: Initialization of the PAPSO.

Input: M_1 is the number of particles in the swarm; a normal sample $(\boldsymbol{x_n}, y_n)$

1: **for** each particle $i = 1, \ldots, M_1$ **do**
2: Initialize the particle's position $\boldsymbol{p_i}$
3: Initialize the particle's velocity $\boldsymbol{v_i}$
4: Initialize $pbest$: $\boldsymbol{p_i^*} \leftarrow \boldsymbol{p_i}$
5: **end for**
6: Initialize $gbest$: Choose $acc(\boldsymbol{p_{max}})$ from $acc(\boldsymbol{p_1})$ to $acc(\boldsymbol{p_{M_1}})$ where $acc(\boldsymbol{p_{max}})$ is the maximum value.
7: $\boldsymbol{p_{gb}} \leftarrow \boldsymbol{p_{max}}$

Algorithm 2: Poisoning Attack With Particle Swarm Optimization.

Input: acceleration factors c_1, c_2; random numbers r_1, r_2; inertia weight α; T_1 is the number of iteration; M_1 is the number of particles in the swarm; a normal sample $(\boldsymbol{x_n}, y_n)$

Output: one optimal poisoned sample $(\boldsymbol{x_p}, y_n)$

1: $p_1, p_2, \ldots, p_{M_1}$ = initialization of the PAPSO$(M_1, (\boldsymbol{x_n}, y_n))$
2: **for** $t = 1$ to T_1 (iteration counter) **do**
3: **for** each particle $i = 1, \ldots, M_1$ **do**
4: $\boldsymbol{v_i} \leftarrow \alpha \boldsymbol{v_i} + c_1 r_1 (\boldsymbol{p_i^*} - \boldsymbol{p_i}) + c_2 r_2 (\boldsymbol{p_{gb}} - \boldsymbol{p_i})$
5: $\boldsymbol{p_i} \leftarrow \boldsymbol{p_i} + \boldsymbol{v_i}$
6: **if** $acc(\boldsymbol{p_i}) > acc(\boldsymbol{p_i^*})$ **then**
7: $\boldsymbol{p_i^*} \leftarrow \boldsymbol{p_i}$
8: **end if**
9: **if** $acc(\boldsymbol{p_i}) > acc(\boldsymbol{p_{gb}})$ **then**
10: $\boldsymbol{p_{gb}} \leftarrow \boldsymbol{p_i}$
11: **end if**
12: **end for**
13: **end for**
14: $(\boldsymbol{x_p}, y_n) \leftarrow (\boldsymbol{p_{gb}} + \boldsymbol{x_n}, y_n)$
15: **return** $(\boldsymbol{x_p}, y_n)$

Algorithm 3: Initialization of the EAPSO.

Input: M_2 is the number of particles in the swarm; the targeted test sample $(\boldsymbol{x_t}, y_t)$

1: **for** each particle $j = 1, \ldots, M_2$ **do**
2: Initialize the particle's position $\boldsymbol{p_j}$
3: Initialize the particle's velocity $\boldsymbol{v_j}$
4: Initialize $pbest$: $\boldsymbol{p_j^*} \leftarrow \boldsymbol{p_j}$
5: **end for**
6: Initialize $gbest$: Choose $pro(\boldsymbol{p_{min}})$ from $pro(\boldsymbol{p_1})$ to $pro(\boldsymbol{p_{M_2}})$ where $pro(\boldsymbol{p_{min}})$ is the minimum value.
7: $\boldsymbol{p_{gb}} \leftarrow \boldsymbol{p_{min}}$

Algorithm 4: Evasion Attack With Particle Swarm Optimization.

Input: acceleration factors c_1, c_2; random numbers r_1, r_2; inertia weight α; T_2 is the number of iteration; M_2 is the number of particles in the swarm; the targeted test sample $(\boldsymbol{x_t}, y_t)$

Output: the optimal adversarial sample $(\boldsymbol{x_a}, y_t)$

1: $p_1, p_2, \ldots, p_{M_2}$ = initialization of the EAPSO$(M_2, (\boldsymbol{x_t}, y_t))$
2: **for** $t = 1$ to T_2 (iteration counter) **do**
3: **for** each particle $j = 1, \ldots, M_2$ **do**
4: $\boldsymbol{v_j} \leftarrow \alpha \boldsymbol{v_j} + c_1 r_1 (\boldsymbol{p_j^*} - \boldsymbol{p_j}) + c_2 r_2 (\boldsymbol{p_{gb}} - \boldsymbol{p_j})$
5: $\boldsymbol{p_j} \leftarrow \boldsymbol{p_j} + \boldsymbol{v_j}$
6: **if** $pro(\boldsymbol{p_j}) > pro(\boldsymbol{p_j^*})$ **then**
7: $\boldsymbol{p_j^*} \leftarrow \boldsymbol{p_j}$
8: **end if**
9: **if** $pro(\boldsymbol{p_j}) > pro(\boldsymbol{p_{gb}})$ **then**
10: $\boldsymbol{p_{gb}} \leftarrow \boldsymbol{p_j}$
11: **end if**
12: **end for**
13: **end for**
14: $(\boldsymbol{x_a}, y_t) \leftarrow (\boldsymbol{p_{gb}} + \boldsymbol{x_t}, y_t)$
15: **return** $(\boldsymbol{x_a}, y_t)$

$\boldsymbol{p_i}$ $(i = 1, \ldots, M_1)$ means different perturbations which will be added to the normal sample $(\boldsymbol{x_n}, y_n)$. $acc(\boldsymbol{p_i})$ represents the classification accuracy of the learned model under $D_{tr} \cup (\boldsymbol{x_n} + \boldsymbol{p_i}, y_n)$, where $(\boldsymbol{x_n} + \boldsymbol{p_i}, y_n)$ is the poisoned sample which is combined by the normal sample $(\boldsymbol{x_n}, y_n)$ and $\boldsymbol{p_i}$. The $pbest$ of $\boldsymbol{p_i}$ is denoted by $\boldsymbol{p_i^*}$ and the $gbest$ is denoted by $\boldsymbol{p_{gb}}$.

Before delving into the details of the PAPSO, some assumptions are made: the position of each particle is limited to the allowable range during the update process; we generate and optimize one poisoned sample at a time; As for multiple poisoned samples, we inject one poisoned sample to the training dataset and carry out the PAPSO to obtain another poisoned sample, these processes will be executed repeatedly. Poisoning attack with the PSO will be described in Algorithm 2.

C. EAPSO: Evasion Attack With Particle Swarm Optimization

As for EAPSO, the adversary's goal is to decrease the probability that the targeted test sample is classified correctly by adding some hardly perceptible perturbations to the targeted test sample. The initialization algorithm of the EAPSO will be presented in Algorithm 3.

In Algorithm 3, $(\boldsymbol{x_t}, y_t)$ is the targeted test sample and $\boldsymbol{p_j}$ $(j = 1, \ldots, M_2)$ means different perturbations which will be added to the targeted test sample. $pro(\boldsymbol{p_j})$ represents the probability that the adversarial sample $(\boldsymbol{x_t} + \boldsymbol{p_j}, y_t)$ is classified correctly, where the adversarial sample is combined by the targeted test sample $(\boldsymbol{x_t}, y_t)$ and perturbation $\boldsymbol{p_j}$. The $pbest$ of $\boldsymbol{p_j}$ is denoted by $\boldsymbol{p_j^*}$ and the $gbest$ is denoted by $\boldsymbol{p_{gb}}$. Evasion attack with the PSO algorithm will be described in Algorithm 4.

It is important to mention that these two attacks are black-box attacks, which mean the adversary knows nothing about the target model when performing these attacks.

IV. Implementing Our Attacks on Autonomous Vehicles

In this section, we first describe the processes of traffic sign recognition system. Then, we present a brief introduction to Convolutional Neural Network. After that, we explain on how

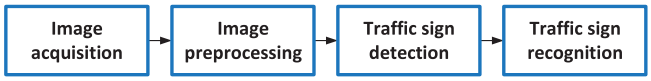


Fig. 3. The main processes of traffic sign recognition system.

to perform the PAPSO and the EAPSO on traffic sign recognition system, respectively.

A. Traffic Sign Recognition System in Autonomous Vehicles

There are many types of traffic signs, such as speed limits, prohibitions, instructions, warnings, etc. These signs are of great significance in regulating the driving behavior of drivers. Accurately detecting and identifying traffic signs can help reduce the driving pressure of drivers and avoid accidents. Nowadays, with the rapid development of intelligent transportation systems, driverless technology has received much attention and developed rapidly. As a critical part of autonomous driving system, traffic sign recognition system has been intensively explored [35], [36]. Deep learning technology plays a critical role in traffic sign recognition due to the advantages of high precision, high robustness and low cost. As shown in Fig. 3, the system mainly consists of four processes:

1) **The image acquisition process** is to collect the images captured by camera device on the car.
2) **The preprocessing process** is to remove redundant background information of the image, eliminate the effects of various noises, enhance and restore the areas containing traffic signs and provide high-quality input images for subsequent processes. For example, the image acquired in the first step may have random noise, which is called image deterioration. It will affect the accuracy of the subsequent processes. By adopting some algorithms restoring the image, image deterioration can be alleviated and the image definition can be improved.
3) **The traffic sign detection process** is to locate the image area containing the traffic sign. For instance, the input image may contain not only the region of traffic sign, but also a large number of regions that are useless for recognition. In order to provide high-quality input images for the next step, it is necessary to use some techniques to minimize those useless areas. Generally, traffic signs have a variety of features, such as color, shape, size, etc. These features are helpful for locating the location of traffic signs. By doing so, the complexity and difficulty of the recognition process can be greatly reduced, and the recognition accuracy and recognition speed can be effectively improved.
4) **The traffic sign recognition process** is to classify the image area identified in the previous stage. In recent years, neural networks, especially convolutional neural networks, have made important breakthroughs in the field of image processing, such as research on face recognition [37], [38]. Thus, we utilize convolutional neural network to classify the traffic signs in this paper.

B. A Brief Introduction to Convolutional Neural Network

CNN has shown impressive performance in the field of image processing. It is generally applied in the traffic sign recognition systems. The architecture of convolutional neural networks is similar as traditional neural networks. The difference is that CNN optimizes the training process of traditional neural networks: convolutional neural networks use convolution operations in the network layer, which is different from matrix multiplication in traditional neural networks. The convolution operation has the advantage of weight sharing and can effectively reduce the number of parameters being trained. A classic CNN architecture called LeNet-5 is depicted as in Fig. 4. In general, CNN includes the following layers:

- **The input layer** is the input to the convolutional neural network, which sends the data into the convolutional neural network in the form of a matrix.
- **The convolution layer** is the most critical part of the convolutional neural network. It consists of numerous convolution units and the back-propagation algorithm is employed to optimize the parameters of each convolution unit. The convolution layer's goal is to extract features of the input. Some features of the input can be enhanced by convolution operations and noise can be reduced.
- **The pooling layer** compresses the input feature map, which makes the feature map smaller and simplifies the network computation complexity. This also prevents overfitting to some extent.
- **The fully connected layer** connects all features and sends output values to the classifier.

C. Implementing the PAPSO on Traffic Sign Recognition System

The processes of the PAPSO against traffic sign recognition system are described as below:

1) Set the number of particles M_1, choose a normal image in the traffic signs training dataset, and carry out the initialization of the PAPSO.
2) Set the required parameters and perform the PAPSO algorithm.
3) Through the iteration of the T_1 round, select the particle that minimizes the test accuracy as a poisoned sample and put it into the original training dataset D_{tr}.
4) Repeat the procedure from step (1).
5) The poisoned sample dataset D_p generated by the above process is the optimal set of poisoned samples.

D. Implementing the EAPSO on Traffic Sign Recognition System

The processes of the EAPSO against traffic sign recognition system are described as below:

1) Set the number of particles M_2, obtain the targeted traffic sign image, and carry out the initialization of the EAPSO.
2) Set the required parameters and perform the EAPSO algorithm.

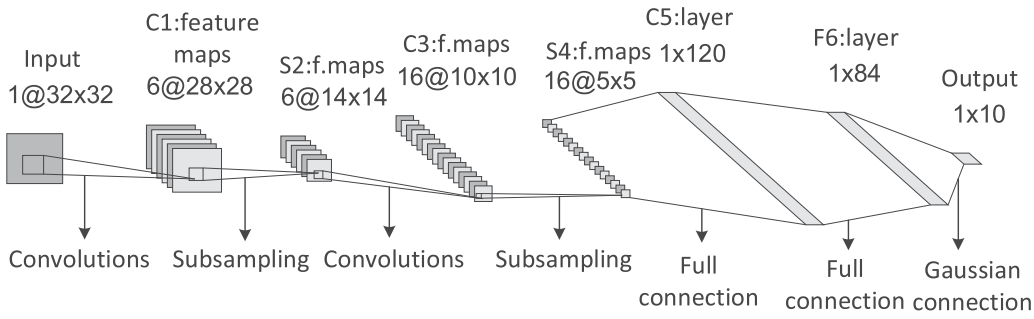


Fig. 4. The architecture of LeNet-5.

3) Through the iteration of the T_2 round, select the particle that minimizes the probability that the targeted test sample is classified correctly.
4) The optimal adversarial sample is the combination of the particle and the targeted test sample.

V. Experimental Analysis

In this section, we implement our attacks on traffic sign recognition system assisted by CNN. Specifically, we first implement the PAPSO in the case of single poisoned sample to analyze the influence of hyper-parameters. After that, we evaluate the performance of the learned model under multiple poisoned samples. Finally, we implement the EAPSO on traffic sign recognition system and evaluate the performance of the classifier on the target sample.

A. Setup

- **Datasets:** Two famous traffic sign datasets (BelgiumTS dataset[1] and GTSRB dataset)[2] are chosen as our datasets. BelgiumTS dataset contains 62 different traffic signs, whose training dataset includes about 4591 images while the test dataset has around 2534 images. The material is recorded in urban environments from Flanders region in Belgium, by GeoAutomation; GTSRB dataset contains 43 different, whose training dataset includes about 39000 images and the test dataset has around 12000 images.
- **Models:** The architecture of CNN used in our experiments is LeNet-5. The network is run over 150 epochs, with batch size 128. A classification accuracy of about 95% is reached under untainted training data on BelgiumTS dataset and a classification accuracy of about 97% is reached under untainted training data on GTSRB dataset.
- **Attack Scenarios:** As mentioned before, two attack scenarios are considered in our work: the PAPSO aims to decrease the classification accuracy of the learned model by injecting poisoned samples to the training dataset; the EAPSO focuses on minimizing the probability that the targeted test sample is classified correctly by adding some hardly perceptible perturbations to the targeted test sample.

[1]https://btsd.ethz.ch/shareddata/
[2]http://benchmark.ini.rub.de/?section=gtsdb&subsection=dataset

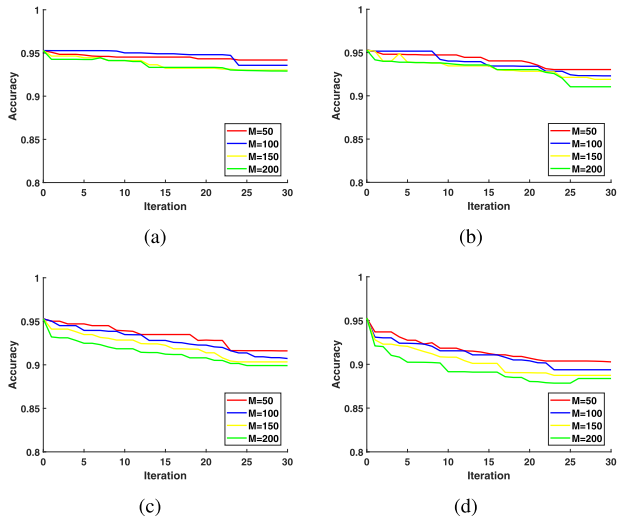


Fig. 5. Results of the PAPSO on BelgiumTS dataset. (a) The modified range = 5%. (b) The modified range = 7.5%. (c) The modified range = 10%. (d) The modified range = 12.5%.

B. Experimental Results of the PAPSO

1) The Influence of Hyper-Parameters (in the Case of Single Poisoned Sample): We first analyze the attack effect of single poisoned sample with different hyper-parameters. Concretely, we use modified range to describe the allowable variation range of the sample's feature value. For example, the modified range means the range of the feature value v is limited to $(0.95v, 1.05v)$ in the case that the modified range is 5%. Then we set the modified range of the image to 5%, 7.5%, 10%, 12.5% and set the population number M to 50, 100, 150, 200. The influence of the two hyper-parameters on the effect of the PAPSO is shown in Fig. 5 and Fig. 6.

From the results we have obtained, we observe that the larger the number of particle swarms, the more effective the attack is. That's because the larger number of particle swarms enables the particles to be distributed in the solution space more randomly, which is easier for the algorithm to get a better solution. Moreover, a large modified range can also improve the attack effect, but it may also reduce the stealth of the attack.

2) Multiple Poisoned Samples: After analyzing the influence of the two hyper-parameters on the effect of the PAPSO, we evaluate the performance of the learned model under multiple

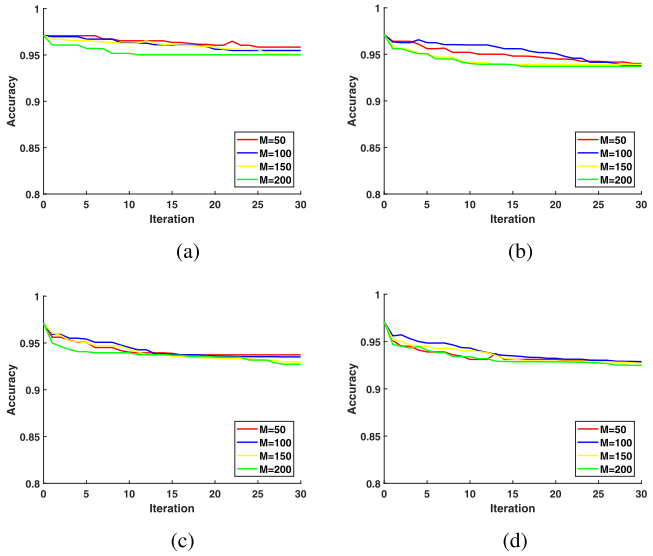


Fig. 6. Results of the PAPSO on GTSRB dataset. (a) The modified range = 5%. (b) The modified range = 7.5%. (c) The modified range = 10%. (d) The modified range = 12.5%.

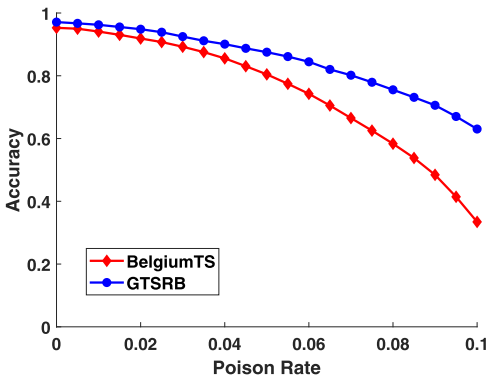


Fig. 7. Results of the PAPSO in the case of multiple poisoned samples.

poisoned samples. Specifically, we set the modified range to 12.5%, the population number M to 100, the number of iterations T to 25 and carry out the PAPSO repeatedly. The results are shown in Fig. 7, where poison rate means the proportion of the poisoned samples to the total number of samples.

On the basis of our results, it can be concluded that the performance of the learned model degrades significantly by injecting poisoned samples into the training dataset. More concretely, the red line represents the classification accuracy of the learned model under BelgiumTS dataset, which has dropped from 95% percent to 33% by injecting only 10% poisoned samples. The blue line represents the classification accuracy of the learned model under GTSRB dataset, which has declined from 97% to 63% by injecting only 10% poisoned samples.

C. Experimental Results of the EAPSO

As for BelgiumTS dataset, we select the test sample shown in Fig. 8(a) below as the target sample, the probability that the sample is classified correctly is about 93% without attack. In terms of GTSRB dataset, we select the test sample

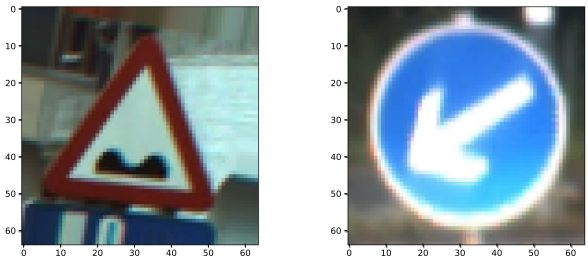


Fig. 8. The two target test samples. (a) The target sample in BelgiumTS dataset. (b) The target sample in GTSRB dataset.

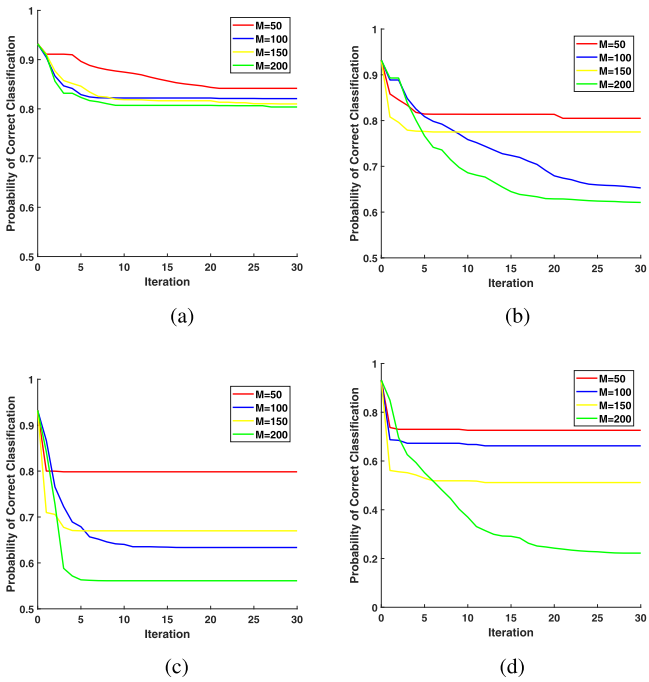


Fig. 9. Results of the EAPSO on BelgiumTS dataset. (a) The modified range = 5%. (b) The modified range = 7.5%. (c) The modified range = 10%. (d) The modified range = 12.5%.

shown in Fig. 8(b) below as the target sample, the probability that the sample is classified correctly is about 99% without attack. Similarly, we evaluate the effect of the EAPSO when the modified range of the image is set to 5%, 7.5%, 10% and 12.5%, the population number M is set to 50, 100, 150, 200. The experimental results of the EAPSO are shown in Fig. 9 and Fig. 10.

From the results we have obtained, we find that there is a distinct decrease in the probability that the targeted sample is classified correctly after several iterations. Specifically, in the case of the modified range is 12.5% and the population number is 200, that probability falls from 93% to 22% in terms of BelgiumTS dataset and that probability falls from 99% to 24% in terms of GTSRB dataset.

The adversarial samples with different modified ranges are shown in Fig. 11 and Fig. 12. These adversarial samples are normal samples combined with hardly perceptible perturbations.

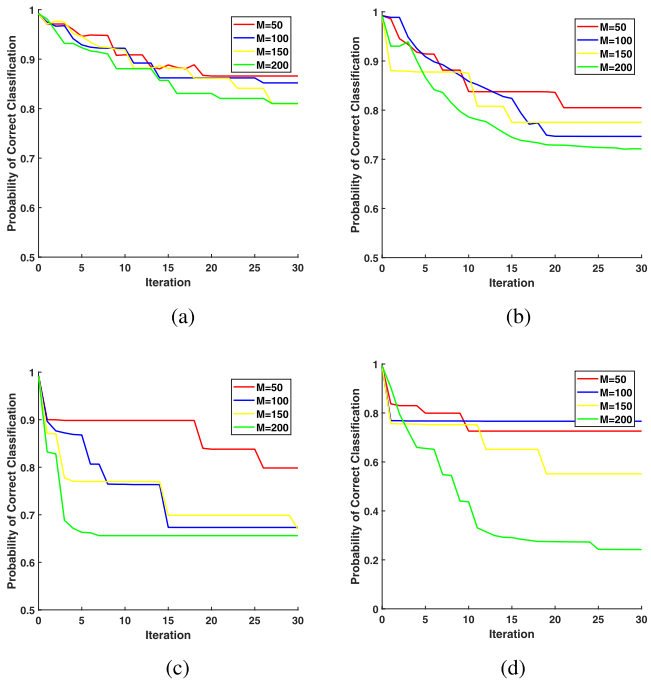


Fig. 10. Results of the EAPSO on GTSRB dataset. (a) The modified range = 5%. (b) The modified range = 7.5%. (c) The modified range = 10%. (d) The modified range = 12.5%.

They can still be correctly classified under the recognition of the human eyes, but may be misclassified under the recognition of the deep learning model.

VI. Defense Strategies

A. Defense Strategies Against Poisoning Attack

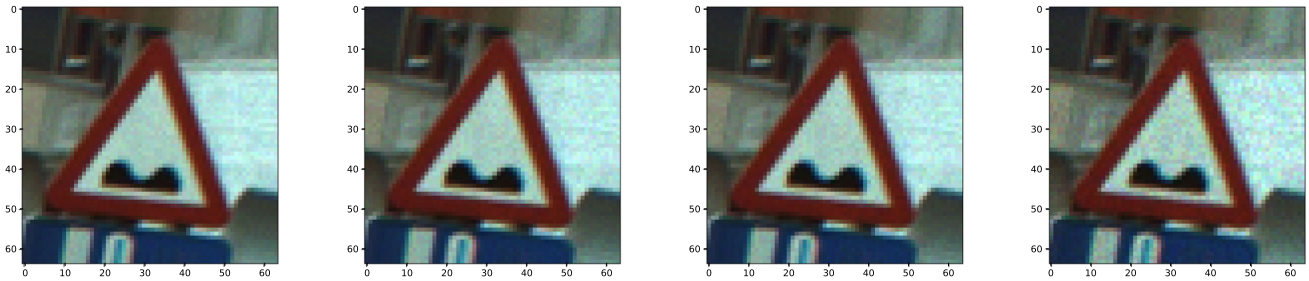
Defense technologies against poisoning attack mainly fall into two categories: data sanitization and robustness improvement. Nelson *et al.* [39] proposed a defense algorithm called reject on negative impact to remove malicious samples: firstly, they add a suspicious sample to the original training dataset and get a new training dataset. After that, a new classifier is trained under the new training dataset. Then they evaluate and compare the performance between new classifier and the original classifier (trained under the original training dataset) under the same test dataset. If the error rate of the new classifier is higher than the original classifier, the suspicious sample will be recognized as a poisoned sample and cleaned out of the training dataset. Otherwise, it will be considered as an untainted sample. However, this method requires a set of untainted samples and other samples will be judged based on this untainted data set. In practice, it is often difficult to obtain enough untainted samples. In addition, the operations of this method are computationally intensive which are not applicable to large-scale datasets. Biggio *et al.* [40] proposed a defense technology which employed the method of bagging ensemble construction. It can reduce the effect of the poisoned samples in training dataset by improving the robustness of the model.

B. Defense Strategies Against Evasion Attack

Defense strategies against evasion attack focus on improving the robustness of learning algorithms. Papernot *et al.* [41] proposed an algorithm called distillation to improve the robustness of the machine learning model. Specifically, they train an initial deep neural network based on the original training data X and labels Y to obtain the probability vector predictions $F(X)$. Then they use the training data X and the output result $F(X)$ as a new label to train a similar distillation network to obtain a new probability vector predictions $F^d(X)$. After that, they use the new distillation network to classify or predict. The sensitivity of the model to small disturbances is reduced and the resistance to the adversarial sample is improved through this method. Szegedy *et al.* [42] proposed a method called adversarial training to mitigate evasion attacks. Specifically, they introduced legalized adversarial samples into the training dataset, which mimicked the possible adversarial samples during the prediction process. Adversarial training can gradually improve the classification performance of the model on legal samples and adversarial samples and make the learned model more robust against adversaries. Tramèr *et al.* [43] proposed ensemble adversarial training, which augments training data with adversarial samples from many other models. Song *et al.* [44] proposed Multi-strength Adversarial Training (MAT), which combined the adversarial training examples with different adversarial strengths to defend evasion attacks. In fact, although adversarial training is effective, it still cannot completely resist adversarial samples. Mainly because it is unrealistic to include all possible adversarial samples in the training set during the adversarial training process.

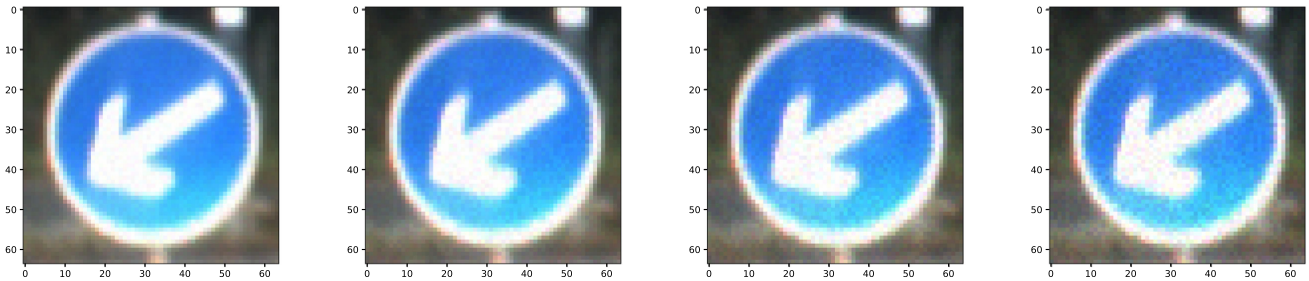
VII. Related Work

In terms of poisoning attack, Mei *et al.* [45] proposed a poisoning attack framework employing the idea of *machine teaching* [46], which is the inverse problem of machine learning. In *machine teaching*, the learning algorithm L and the target model θ^* are known, the optimal training set D needs to be found so that $L(D) = \theta^*$. [47] extended the target of poisoning attacks to multi-classification problems for the first time, they proposed a new poisoning attack algorithm leveraging on the method of back-gradient optimization. This algorithm can be performed on a wider range of learning algorithms with lower complexity. Jagielski *et al.* [48] proposed a poisoning attack framework on regression algorithms. Specifically, they proposed two methods of poisoning attacks: the first one is based on optimization, the attacker's strategy can be formulated as a two-level optimization problem. The gradient descent method is used to optimize this problem to obtain a better poisoning sample; the second one is based on statistics, poisoning samples can be quickly generated by statistically analyzing the position characteristics of the poisoning samples that have performed well in the past. Suciu *et al.* [49] defined the knowledge and capabilities of the attacker in detail from four dimensions and proposed a targeted poisoning algorithm StingRay, which can be applied to many different learning algorithms and can bypass two existing defense algorithm.



(a) (b) (c) (d)

Fig. 11. The adversarial samples for BelgiumTS dataset. (a) The modified range = 5%. (b) The modified range = 7.5%. (c) The modified range = 10%. (d) The modified range = 12.5%.



(a) (b) (c) (d)

Fig. 12. The adversarial samples for GTSRB dataset. (a) The modified range = 5%.(b) The modified range = 7.5%. (c) The modified range = 10%. (d) The modified range = 12.5%.

As for evasion attack, it was first proposed for some applications of machine learning in the field of security. The attacker can generate adversarial samples that can successfully evade the detection of the security system to achieve a malicious attack on the system, which poses a serious threat to the security of the system. For example, the attacker may evade spam detection by adding some positive words to spam [31] or imitate the identities of other users in the face recognition system [50]. As for the method to generate adversarial sample, Szegedy *et al.* [42] employed L-BFGS algorithm to search the adversarial samples in the input space and obtain the optimal adversarial samples by adding small perturbations to the correctly classified input sample. Goodfellow *et al.* [51] employed the FGSM (fast gradient sign method) to obtain the minimum perturbations r, so that the perturbed image can deceive the deep learning model and make it misclassified. Moosavi-Dezfooli *et al.* [52] proposed the DeepFool method to generate adversarial samples with minimal perturbations. Their experiments demonstrated that the adversarial samples produced by DeepFool have less perturbation than FGSM under the similar deception rates. Papernot *et al.* [53] employed Jacobian Matrix to evaluate the sensitivity of the model to each input feature and used adversarial salience map to select the perturbation. After that, they ranked the contribution of each input feature to the misclassified target and generated an optimal adversarial sample. Szegedy *et al.* [42] found transferability of the adversarial samples, which means an adversarial sample against a machine learning model may still be effective against a different machine learning model. After that, considerable research efforts have been devoted to the transferability of the adversarial sample [30], [54].

VIII. Conclusion

In this paper, we have proposed two attacks against traffic sign recognition system in autonomous vehicles by leveraging particle swarm optimization. Specifically, we first propose the PAPSO (poisoning attack with particle swarm optimization) which attacks the training process of the deep learning model by injecting crafted samples into the training dataset. Then we propose the EAPSO (evasion attack with particle swarm optimization) which attacks the interference process of the deep learning model by adding some hardly perceptible perturbations to a normal sample and causing a misclassification of it. After that, the PAPSO and the EAPSO are performed on traffic sign recognition system. Experimental results suggest that the classification accuracy of the learned model has dropped from 95% to 33% by injecting only 10% poisoned samples; the probability that the targeted sample is classified correctly falls from 93% to 22% by adding some hardly perceptible perturbations. Finally, some defense technologies against these two attacks are presented.

References

[1] N. Kato *et al.*, "The deep learning vision for heterogeneous network traffic control: Proposal, challenges, and future perspective," *IEEE Wireless Commun.*, vol. 24, no. 3, pp. 146–153, Jun. 2017.

[2] F. Tang *et al.*, "On removing routing protocol from future wireless networks: A real-time deep learning approach for intelligent traffic control," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 154–160, Feb. 2018.

[3] F. Tang, Y. Kawamoto, N. Kato, and J. Liu, "Future intelligent and secure vehicular network toward 6G: Machine-learning approaches," *IEEE Proc.*, vol. 108, no. 2, pp. 292–307, Feb. 2020.

[4] J. Ning, J. Wang, J. Liu, and N. Kato, "Attacker identification and intrusion detection for in-vehicle networks," *IEEE Commun. Lett.*, vol. 23, no. 11, pp. 1927–1930, Nov. 2019.

[5] H. Nakayama, S. Kurosawa, A. Jamalipour, Y. Nemoto, and N. Kato, "A dynamic anomaly detection scheme for AODV-based mobile ad hoc networks," *IEEE Trans. Veh. Technol.*, vol. 58, no. 5, pp. 2471–2481, Jun. 2009.

[6] Z. Ma *et al.*, "Fine-grained vehicle classification with channel max pooling modified CNNs," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3224–3233, Apr. 2019.

[7] Y. Li, Q. Luo, J. Liu, H. Guo, and N. Kato, "TSP security in intelligent and connected vehicles: Challenges and solutions," *IEEE Wireless Commun.*, vol. 26, no. 3, pp. 125–131, Jun. 2019.

[8] Z. Ma, H. Yu, W. Chen, and J. Guo, "Short utterance based speech language identification in intelligent vehicles with time-scale modifications and deep bottleneck features," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 121–128, Jan. 2019.

[9] G. Xu, H. Li, S. Liu, M. Wen, and R. Lu, "Efficient and privacy-preserving truth discovery in mobile crowd sensing systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3854–3865, Apr. 2019.

[10] Z. Ma, J. Xie, Y. Lai, J. Taghia, J.-H. Xue, and J. Guo, "Insights into multiple/single lower bound approximation for extended variational inference in non-Gaussian structured data modeling," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published, doi: 10.1109/TNNLS.2019.2899613.

[11] K. C. Zeng *et al.*, "All your GPS are belong to us: Towards stealthy manipulation of road navigation systems," in *Proc. USENIX Secur. Symp.*, 2018, pp. 1527–1544.

[12] J. Petit, B. Stottelaar, M. Feiri, and F. Kargl, "Remote attacks on automated vehicles sensors: Experiments on camera and LiDAR," in *Black Hat Eur.*, vol. 11, pp. 1–13, 2015.

[13] C. Yan, W. Xu, and J. Liu, "Can you trust autonomous vehicles: Contactless attacks against sensors of self-driving vehicle," in *Def. Con.*, vol. 24, pp. 1–13, 2016.

[14] S. van de Beek, R. Vogt-Ardatjew, and F. Leferink, "Robustness of remote keyless entry systems to intentional electromagnetic interference," in *Proc. Int. Symp. Electromagn. Compat.*, 2014, pp. 1242–1245.

[15] A. Francillon, B. Danev, and S. Capkun, "Relay attacks on passive keyless entry and start systems in modern cars," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2011, pp. 431–439.

[16] S. Woo, H. J. Jo, I. S. Kim, and D. H. Lee, "A practical security architecture for in-vehicle CAN-FD," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 8, pp. 2248–2261, Aug. 2016.

[17] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," in *Proc. Int. Conf. Mach. Learn.*, 2012, pp. 1807–1814.

[18] G. Xu, H. Li, Y. Dai, K. Yang, and X. Lin, "Enabling efficient and geometric range query with access control over encrypted spatial data," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 4, pp. 870–885, Apr. 2019.

[19] H. Ren, H. Li, Y. Dai, K. Yang, and X. Lin, "Querying in internet of things with privacy preserving: Challenges, solutions, and opportunities," *IEEE Network*, vol. 32, no. 6, pp. 144–151, Nov./Dec. 2018.

[20] W. Jiang, H. Li, S. Liu, Y. Ren, and M. He, "A flexible poisoning attack against machine learning," in *Proc. IEEE Int. Conf. Commun.*, 2019, pp. 1–6.

[21] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert, and F. Roli, "Is feature selection secure against training data poisoning," in *Proc. Int. Conf. Mach. Learn.*, 2015, vol. 2, pp. 1689–1698.

[22] W. Jiang, H. Li, G. Xu, M. Wen, G. Dong, and X. Lin, "PTAS: Privacy-preserving thin-client authentication scheme in blockchain-based PKI," *Future Gener. Comput. Syst.*, vol. 96, pp. 185–195, 2019.

[23] H. Li, Y. Yang, T. H. Luan, X. Liang, L. Zhou, and X. Shen, "Enabling fine-grained multi-keyword search supporting classified sub-dictionaries over encrypted cloud data," *IEEE Trans. Dependable Secure Comput.*, vol. 13, no. 3, pp. 312–325, May-Jun. 2016.

[24] M. Fang, G. Yang, N. Z. Gong, and J. Liu, "Poisoning attacks to graph-based recommender systems," in *Proc. 34th Annu. Comput. Secur. Appl. Conf.*, 2018, pp. 381–392.

[25] H. Li, D. Liu, Y. Dai, T. H. Luan, and X. Shen, "Enabling efficient multi-keyword ranked search over encrypted mobile cloud data through blind storage," *IEEE Trans. Emerg. Topics Comput.*, vol. 3, no. 1, pp. 127–138, Mar. 2015.

[26] H. Li, X. Lin, H. Yang, X. Liang, R. Lu, and X. Shen, "EPPDR: An efficient privacy-preserving demand response scheme with adaptive key evolution in smart grid," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 8, pp. 2053–2064, Aug. 2014.

[27] H. Li, R. Lu, L. Zhou, B. Yang, and X. Shen, "An efficient merkle-tree-based authentication scheme for smart grid," *IEEE Syst. J.*, vol. 8, no. 2, pp. 655–663, Jun. 2014.

[28] G. Xu, H. Li, H. Ren, K. Yang, and R. H. Deng, "Data security issues in deep learning: Attacks, countermeasures and opportunities," *IEEE Commun. Mag.*, vol. 57, no. 11, pp. 116–122, Nov. 2019.

[29] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Trans. Ind. Informat.*, to be published, doi: 10.1109/TII.2019.2945367.

[30] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proc. Conf. Comput. Commun.*, 2017, pp. 506–519.

[31] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. Tygar, "Adversarial machine learning," in *Proc. 4th ACM Workshop Secur. artif. Intell.*, 2011, pp. 43–58.

[32] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "VerifyNet: Secure and verifiable federated learning," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, no. 1, pp. 911–926, Jul. 2020.

[33] H. Li, D. Liu, Y. Dai, T. H. Luan, and S. Yu, "Personalized search over encrypted data with efficient and secure updates in mobile clouds," *IEEE Trans. Emerg. Topics Comput.*, vol. 6, no. 1, pp. 97–109, Jan.–Mar. 2018.

[34] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Mach. Human Sci.*, 1995, pp. 39–43.

[35] J. Li and Z. Wang, "Real-time traffic sign recognition based on efficient CNNs in the wild," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 3, pp. 975–984, Mar. 2019.

[36] J. Kim, S. Lee, T.-H. Oh, and I. S. Kweon, "Co-domain embedding using deep quadruplet networks for unseen traffic sign recognition," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 6975–6982.

[37] Q. Wang and G. Guo, "LS-CNN: Characterizing local patches at multiple scales for face recognition," *IEEE Trans. Inf. Forensics Sec.*, vol. 15, no. 1, pp. 1640–1653, Oct. 2020.

[38] W. Hu, Y. Huang, F. Zhang, and R. Li, "Noise-tolerant paradigm for training face recognition CNNs," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 11 887–11 896.

[39] B. Nelson *et al.*, "Misleading learners: Co-opting your spam filter," in *Proc. Mach. Learn. Cyber Trust*, 2009, pp. 17–51.

[40] B. Biggio, I. Corona, G. Fumera, G. Giacinto, and F. Roli, "Bagging classifiers for fighting poisoning attacks in adversarial classification tasks," in *Proc. Int. Workshop Multiple Classifier Syst.*, 2011, pp. 350–359.

[41] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proc. IEEE Symp. Secur. Privacy*, 2016, pp. 582–597.

[42] C. Szegedy *et al.*, "Intriguing properties of neural networks," in *Int. Conf. Learn. Representations (ICLR)*, 2014.

[43] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," in *Int. Conf. Learn. Representations (ICLR)*, 2018.

[44] C. Song *et al.*, "MAT: A multi-strength adversarial training method to mitigate adversarial attacks," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI*, 2018, pp. 476–481.

[45] S. Mei and X. Zhu, "Using machine teaching to identify optimal training-set attacks on machine learners," in *Proc. Nat. Conf. Artif. Intell.*, 2015, pp. 2871–2877.

[46] X. Zhu, "Machine teaching: An inverse problem to machine learning and an approach toward optimal education," in *Proc. Nat. Conf. Artif. Intell.*, 2015, pp. 4083–4087.

[47] L. M.-González *et al.*, "Towards poisoning of deep learning algorithms with back-gradient optimization," in *Proc. 10th ACM Workshop Artif. Intell. Secur.*, 2017, pp. 27–38.

[48] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nitarotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *IEEE Proc. Symp. Secur. Privacy*, 2018, pp. 931–947.

[49] O. Suciu, R. Marginean, Y. Kaya, H. Daume III, and T. Dumitras, "When does machine learning fail? Generalized transferability for evasion and poisoning attacks," in *Proc. USENIX Secur. Symp.*, 2018, pp. 1299–1316.

[50] K. W. Bowyer, "Face recognition technology: Security versus privacy," *IEEE Technol. Soc. Mag.*, vol. 23, no. 1, pp. 9–19, Spring 2004.

[51] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Int. Conf. Learn. Representations (ICLR)*, 2015.

[52] S.-M. M.-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: A simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 2574–2582.
[53] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *IEEE Proc. Eur. Symp. Secur. Privacy*, 2016, pp. 372–387.
[54] S.-M. M.-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 1765–1773.

Wenbo Jiang (Student Member, IEEE) received the B.S. degree in information security from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2017, where he is currently working toward the master's degree with the School of Computer Science and Engineering. His research interests include cryptography, blockchain, and the machine learning.

Hongwei Li (Senior Member, IEEE) received the Ph.D. degree from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in June 2008. He is currently the Head and a Professor with Department of Information Security, School of Computer Science and Engineering, UESTC. He worked as a Postdoctoral Fellow with the University of Waterloo from October 2011 to October 2012. His research interests include network security and applied cryptography. He is the Distinguished Lecturer of IEEE Vehicular Technology Society.

Sen Liu (Student Member, IEEE) received the B.S. degree in information security from Guizhou University, Guiyang, China, in 2017. He is currently working toward the master's degree with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China. His research interests include cryptography and searchable encryption.

Xizhao Luo received the B.S. and M.S. degrees from the Xi'an University of Technology, Xi'an, China, in 2000 and 2003, respectively, and the Ph.D. degree from Soochow University, Suzhou, China, in 2010. He held a Postdoctoral position with the Center of Cryptography and Code, School of Mathematical Science, Soochow University, for three years. His main fields of interest are cryptography and computational complexity.

Rongxing Lu (Senior Member, IEEE) received the Ph.D. degree from the University of Waterloo, Waterloo, ON, Canada, in 2012. He is an Associate Professor with the Faculty of Computer Science, University of New Brunswick (UNB), Fredericton, NB, Canada. Before that, he worked as an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore from April 2013 to August 2016. He worked as a Postdoctoral Fellow with the University of Waterloo from May 2012 to April 2013. He was awarded the most prestigious "Governor Generals Gold Medal," He won the 8th IEEE Communications Society (ComSoc) Asia Pacific (AP) Outstanding Young Researcher Award, in 2013. He is presently a Senior Member of IEEE Communications Society. He currently serves as the Vice-Chair (Conferences) of IEEE ComSoc CIS-TC. He is the winner of 2016–2017 Excellence in Teaching Award, FCS, UNB.