Fake Node-Based Perception Poisoning Attacks against Federated Object Detection Learning in Mobile Computing Networks

Xiong Xiao[1], Mingxing Duan[1]*, Yingjie Song[1], Zhuo Tang[1], Wenjing Yang[2]*
[1]College of Computer Science and Electronic Engineering, Hunan University, Changsha, China
[2]School of Computer Science, National University of Defense Technology, Changsha, China
{xx,duanmingxing,yjsong0044,ztang}@hnu.edu.cn,wenjing.yang@nudt.edu.cn

ABSTRACT

Federated learning (FL) supports massive edge devices to collaboratively train object detection models in mobile computing scenarios. However, the distributed nature of FL exposes significant security vulnerabilities. Existing attack methods either require considerable costs to compromise the majority of participants, or suffer from poor attack success rates. Inspired by this, we devise an efficient fake node-based perception poisoning attacks strategy (FNPPA) to target such weaknesses. In particular, FNPPA poisons local data and injects multiple fake nodes to participate in aggregation, aiming to make the local poisoning model more likely to overwrite clean updates. Moreover, it can achieve greater malicious influence on target objects at a lower cost without affecting the normal detection of other objects. We demonstrate through exhaustive experiments that FNPPA exhibits superior attack impact than the state-of-the-art in terms of average precision and aggregation effect.

CCS CONCEPTS

• **Security and privacy → Mobile and wireless security**; **Distributed systems security**.

KEYWORDS

Federated learning, object detection, mobile computing, perception poisoning attacks

ACM Reference Format:
Xiong Xiao[1], Mingxing Duan[1]*, Yingjie Song[1], Zhuo Tang[1], Wenjing Yang[2]*. 2024. Fake Node-Based Perception Poisoning Attacks against Federated Object Detection Learning in Mobile Computing Networks. In *61st ACM/IEEE Design Automation Conference (DAC '24), June 23–27, 2024, San Francisco, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3649329.3655934

1 INTRODUCTION

With the vigorous development of DNN-based object detection technology, an increasing number of recognition models are deployed in applications such as smart healthcare [8] and autonomous vehicles [11]. Taking the autonomous driving system as an example, it requires a recognition model to correctly detect road conditions, pedestrian locations, obstacles, etc., to further make the correct decisions for autonomous vehicles. Otherwise, even minor errors will cause tragic traffic accidents. However, due to increasingly complex road conditions, sensitive intelligent applications such as the above are constantly placing higher requirements on the accuracy of object detection models. Previous traditional training methods [4] usually centrally aggregate and manage diverse data collected at the mobile edge to train a reliable object detection model. However, uploading a large amount of real-time data collected by sensors and cameras in these mobile edge environments to the central server not only poses enormous challenges to the network and storage, but will also result in huge costs for data transmission on millions of mobile devices [13]. More importantly, such approaches expose sensitive data on mobile devices to huge risks of leakage.

The rise of Federated Learning (FL) [3] technology has introduced innovative solutions to distributed training across massive mobile devices. In a highly interconnected FL mobile network, mobile devices (called participants) scattered at the edge only need to update local training parameters to the server to perform aggregation without having to upload their private data. Then a global model is built in continuous iterative collaborative training [18]. FL's excellent performance in distributed collaborative training and maintaining data privacy has also led to its expansion into the fields of financial, medical, and industrial Internet of Things [19].

Although FL does a good job of safeguarding data privacy in isolating data aggregation of each participant, it also exposes potential security vulnerabilities during global training. When performing federated object detection training, malicious attackers infiltrating the network can use the label flipping poisoning attacks, the bounding box poisoning attacks, and the object existence poisoning attacks to affect the global training. They aim to induce the FL systems to aggregate poisoned local models, thereby making erroneous object detection decisions [9]. Controlling more attackers can produce more obvious attack effects, but this requires huge costs. Meanwhile, in the process of joint training for a large number of mobile edge devices (such as next word prediction on virtual keyboards [1]), bottlenecks such as issues with network, signal, and power will cause many participants to interrupt or directly exit the network. Attackers can create fake nodes in such mobile networks to participate in training, making malicious participants achieve stronger aggregation effects in the Federated Average (FedAvg) [16] at lower costs, thereby achieving more reliable attack effectiveness [17].

Aiming at the security vulnerabilities when training federated object detection systems in mobile computing networks, this work

*Mingxing Duan and Wenjing Yang are the author for correspondence.

DAC '24, June 23–27, 2024, San Francisco, CA, USA

ACM ISBN 979-8-4007-0601-1/24/06…$15.00
https://doi.org/10.1145/3649329.3655934

explores a more threatening and efficient fake node-based perception poisoning attacks strategy, termed FNPPA. This is the first fake node-based perception poisoning attacks technology proposed in a mobile computing environment. Specifically, we use three perception poisoning attacks to more comprehensively and covertly contaminate training samples, making the model to make wrong object detection decisions on targeted samples without affecting other normal training samples. In addition, based on the easy disconnection characteristics of participants in the FL mobile computing environment, attackers can also inject multiple fake nodes to participate in aggregation, achieving more impressive attack effects in a simpler and lower-cost way. We demonstrate the reliability of FNPPA through multiple performance metrics under multiple datasets. Our key contributions are outlined below.

- We explore the aggregation hazards of FL networks in mobile computing environments and implement three perception poisoning attacks. Meanwhile, we make minimal prior knowledge assumptions about the attacker.
- We devise an efficient and covert perception poisoning attacks strategy based on fake nodes. FNPPA not only precisely affects the model's detection of poisoned samples, but also maintains the detection accuracy of other normal samples. In addition, fake nodes help attackers perform aggregation with greater probability and lower cost, and achieve reliable attack performance.
- We use multiple datasets to test FNPPA, and multiple performance metrics show that FNPPA exhibits more advanced attack effects than the state-of-the-art.

2 RELATED WORK

2.1 Object Detection in FL Mobile Computing Systems

Implementing reliable object detection in distributed FL networks can help intelligent applications run safely [2]. Take the autonomous driving system as an example [7]. It will decide whether the autonomous vehicle changes its driving state according to pedestrians, obstacles, etc., detected by the object detection system [5]. The execution tasks of object detection usually include confirming whether the objectness is a real object, the bounding box to be processed, and the predicted possible class of the object [10]. Building an object detection model based on the FL network can achieve more efficient and cheaper object detection in a mobile computing environment.

2.2 Perception Poisoning Attacks for Object Detection

FL supports massive mobile devices to collaboratively train object detection models. However, the potential vulnerabilities of the system give malicious attackers a wider space to launch covert attacks. They can leverage dirty labels for poisoning [15] [19], or inject trojans into training samples and activate backdoors [6] [17]. They can also pollute samples utilizing different perception poisoning attacks to get locally poisonous training models and share them in subsequent aggregations [9] [10]. However, such an attack requires controlling a significant number of participants to achieve considerable attack impact. In this paper, we analyze the issue of aggregation probabilities for malicious attackers and consider the use of fake node techniques. It has lower cost and stronger attack performance, and is more suitable for real mobile computing scenarios.

3 PRELIMINARIES AND THREAT MODELS

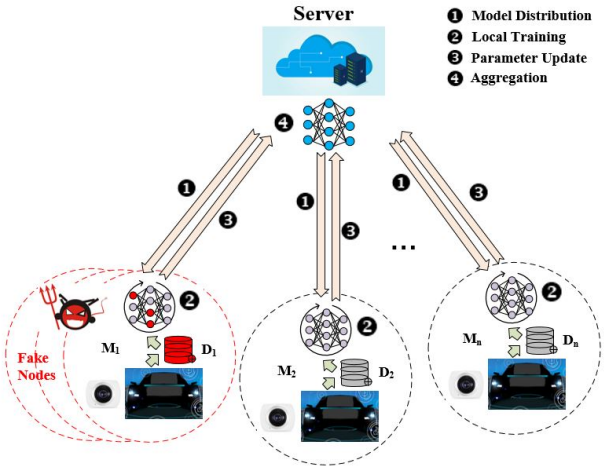


Figure 1: FL Architecture and threat models.

3.1 FL Architecture in Mobile Computing Systems

The architecture of the federated object detection model based on a mobile computing environment can be found in Fig. 1. Each mobile edge device ($M = \{M_i\}_{i=1}^{n}$) with storage capability utilizes its local private data ($D = \{D_i\}_{i=1}^{n}$) for model training. The training samples held by all devices are different from each other, but they all have similar data features. One can see from Fig. 1 that the key execution flow of the entire FL architecture includes four main steps.

(1). The server distributes the initial global object detection model $G_{od-g}^{(0)}$.

(2). Participants perform local training. They implement local training through the Stochastic Gradient Descent (SGD) strategy for each round (r) of acquiring $G_{od-g}^{(0)}$, denoted as:

$$G_{od-l}^{(r+1)} = G_{od-g}^{(r)} - \eta \cdot \nabla L(G_{od-g}^{(r)}, D_i) \tag{1}$$

Where η and ∇L refer to the learning rate and gradient respectively.

(3). Participants update the trained local model $G_{od-l}^{(r+1)}$ to the server.

(4). The server implements the following aggregation through FedAvg.

$$G_{od-g}^{(r)} = \frac{1}{s} \sum_{i=1}^{s} G_{od-l_i}^{(r)} \tag{2}$$

It should be noted that the server will randomly select s participants in each round r of the aggregation stage, which is a compromise between training efficiency and communication optimization. When all R rounds of training complete iterations, the final converged global model G_{od-g} can be obtained.

3.2 Threat Models and Attacker Knowledge

Threat models: The threat models we consider are that malicious attackers lurking in mobile computing systems can launch covert perception poisoning attacks at any time. One can observe from Fig. 1 that the attacker is marked in red and contains injected fake nodes. They jointly participate in model updating and aggregation. Specifically, they only receive the original attacker's poisoning model without doing data preprocessing and execution training, and have no interference with the FL architecture and execution algorithm. Additionally, the server is honest and cannot be compromised or hacked. We set the strictest knowledge assumptions about malicious attackers and their fake nodes.

Attacker objectives and capabilities: The explicit objectives of the attackers are to use multiple perception poisoning attacks to poison the local model, while using fake nodes technology to achieve more obvious attack performance during the global aggregation process. They are designed to cause a decrease in accuracy on the specified samples of the attack, without affecting other normal training samples. The attacker's capabilities include the following two key cores. First, each attacker can locally and covertly manipulate the object class label, tamper with the size of the bounding box, or eliminate object boxes in the sample to implement perception poisoning attacks, and then train the poisoned local model based on the destroyed objects. Second, attackers can inject a large number of fake nodes into the mobile computing system and manipulate them to participate in aggregation. It is important to emphasize that they do not reveal the data and models of honest participants, nor do they destroy the global network and alter the aggregation rules.

4 FAKE NODE-BASED PERCEPTION POISONING ATTACKS

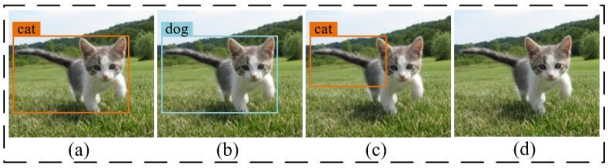


Figure 2: An implementation example of perception poisoning attacks. (a) Clean sample without poisoning. (b) Label flipping poisoning. (c) Bounding box poisoning. (d) Object existence poisoning.

4.1 Implementation Details of Perception Poisoning Attacks

Although the federated object detection model trained based on the mobile computing environment can perform perception tasks well, and can support the intelligent operation of systems such as autonomous driving and smart robots, making safer driving decisions for them. However, this also gives malicious attackers more room to manipulate the execution of perception tasks and launch different attack strategies based on perception tasks. As illustrated in Fig. 2, such perception poisoning attacks encompass the following three threats. (1) **Label flipping poisoning**. One

Algorithm 1: FNPPA Algorithm

Require: Local private data D_i; Total iteration rounds of training ($r \in R$); Global model of r-rounds iteration $G_{od-g}^{(r)}$; Fake nodes fn; Loss function L and Learning rate η.

Ensure: $G_{od-g}^{(r+1)}$.

Initialization: (For Server) the global model $G_{od-g}^{(0)}$, (For Participants) the attackers m

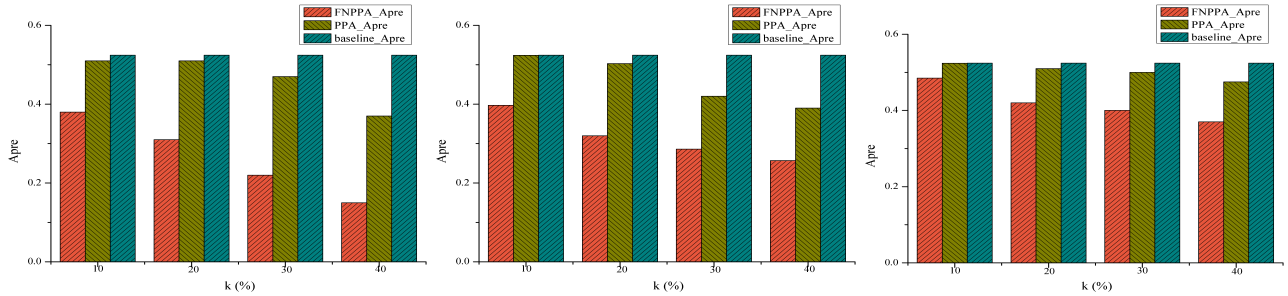
//(1): At Server-side
Function *Agg(r+1)*
for $i \in s$ **do**
  $G_{od-l_i}^{(r+1)} = Participant\_Update(G_{od-g}^{(r)})$
end for
$G_{od-g}^{(r+1)} = \frac{1}{s}\sum_{i=1}^{s} G_{od-l_i}^{(r)}$

//(2): At Participant-side
Function Participant_Update($G_{od-g}^{(r)}$)
//Clean update
Clean participants $i \in (n-m)$
for $epoch\_e = (1, 2...E)$ **do**
  for $batch-size\_b \in D_i$ **do**
    $G_{od-l_i}^{(r+1)} = G_{od-g}^{(r)} - \eta \cdot \nabla L(G_{od-g}^{(r)}, b)$
  end for
end for
$G_{od-l_i}^{(r+1)} \leftarrow G_{od-g}^{(r)}$

//Tainted update
Attackers $i \in m$
for $epoch\_e = (1, 2...E)$ **do**
  for $batch-size\_b \in D_{i_m}$ **do**
    $G_{od-l_i}^{(r+1)} = G_{od-g}^{(r)} - \eta \cdot \nabla L(G_{od-g}^{(r)}, b)$
  end for
end for
Attackers i_m distribute tainted model to fake nodes fn
$G_{od-l_{fn}}^{(r+1)} \leftarrow G_{od-l_m}^{(r+1)}$
The malicious ($m\&fn$) models: $G_{od-l_{mal}}^{(r+1)} \leftarrow G_{od-g}^{(r)}$

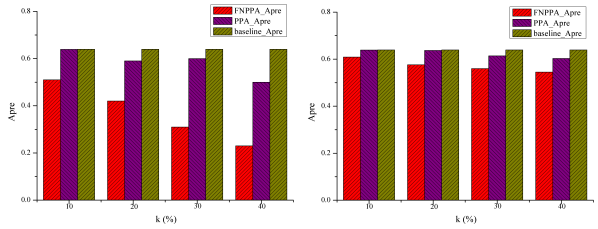
return $G_{od-g}^{(r+1)}$

can find that from Fig. 2(b), compared with Fig. 2(a), it can be seen that the malicious adversary flips the source class label ("*cat*") to the selected target attack class label ("*dog*") to covertly achieve perception poisoning at the class label level. They pursue the goal of enabling the object detection model to misclassify the source class object in the input sample ("classify the object with the class label *cat* as *dog*"), but they will not tamper with the bounding box and its coordinates of the object. (2) **Bounding box poisoning**. One can find from Fig. 2(c) that the malicious adversary aims to tamper with the bounding box position of the source object of the local sample and redefine its size, without flipping the class label of the object. Once such poisoned samples with arbitrarily marked bounding boxes are trained, the object detection model



(a) Label flipping poisoning of VOC (b) Bounding box poisoning of VOC (c) Object existence poisoning of VOC

Figure 3: Evaluation *Apre* of perception poisoning attacks on VOC dataset, for different k, the fn is set to 5.



(a) Bounding box poisoning of INRIA (b) Object existence poisoning of INRIA

Figure 4: Evaluation *Apre* of perception poisoning attacks on INRIA dataset, for different k, the fn is set to 5.

will be unable to estimate the size and specific location of real objects in actual mobile computing scenarios. (3) **Object existence poisoning**. One can find from Fig. 2(d) that the malicious adversary implements the poisoning attack at the object existence level by eliminating the object class label and bounding box marked in the original sample. Such training samples will cause the object detection model to weaken its ability to identify the source object after the collaborative training is completed.

4.2 FNPPA Strategy

After all malicious adversaries have produced malicious samples, they will further utilize these contaminated samples D_{i_m} to implement the next round of local training and strengthen the poisoned joint model in continuous iterations. What is even more prominent is that malicious adversaries take advantage of the vulnerability of participants in the mobile computing network to easily go offline to create fake nodes (fn) to directly participate in model aggregation. These fake nodes do not pose a destructive threat to other participants and server, but only receive poisoned local models $G_{od-l_m}^{(r+1)}$ from the original malicious attacker to implement updates. They collude to manipulate these local poisoning models to enable malicious participants to achieve stronger aggregation effects in the FedAvg through lower costs, thereby achieving more reliable attack effectiveness. The attack details of malicious adversaries and fake nodes are implemented as follows:

$$G_{od-l_{mal}}^{(r+1)} = G_{od-g}^{(r)} - \eta \cdot \nabla L(G_{od-g}^{(r)}, D_{i_m}) \tag{3}$$

where the original malicious adversaries (m) and the injected fake nodes (fn) belong to malicious attackers (mal), ($m\&fn \in mal$).

Due to the strong aggregation effectiveness exhibited by malicious attackers in mobile computing networks. To a large extent, once the randomly selected participants to execute FedAvg are malicious attackers, this will result in the positive updates of many honest participants being compromised. We present the algorithm details and implementation flow of FNPPA more intuitively in Algorithm 1.

5 EXPERIMENTS STUDY

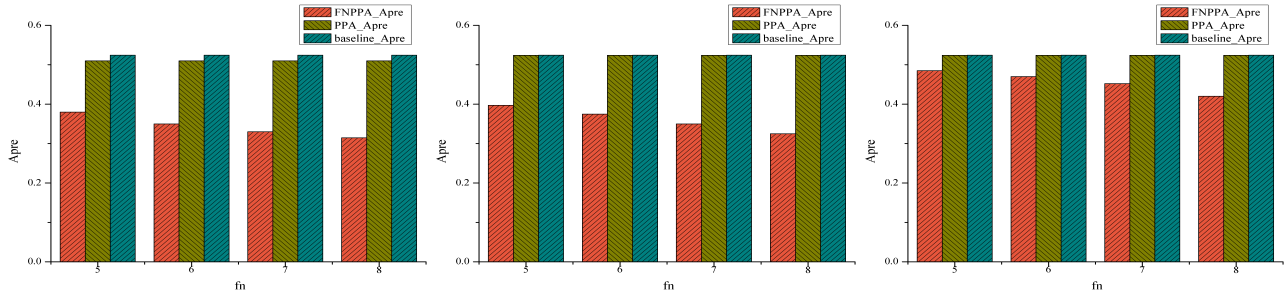
5.1 Experimental Setup

Datasets and Model: We utilize two widely-used object detection benchmark datasets to evaluate the attack effectiveness of the FNPPA strategy. One is the PASCAL VOC dataset [14], which contains 20 object classes and a total of 5,011 data samples, of which the first 1,000 samples are used for testing. The other is the INRIA Person dataset [20], which contains 614 samples for training and 288 samples for testing. All samples are evenly distributed to all participants to perform local training. In addition, this work utilizes Faster R-CNN [12] as the main object detection algorithm and VGG16 as the backbone to explore the effectiveness of attacks against the above two datasets.

FL Training and Attack Setups: For the training setups, we first simulate the FL system with n = 100 participants to perform collaborative training. Second, the server randomly selects s = 10 ($0.1 \times n$) participants in each round to execute FedAvg. Third, based on the convergence trend, we set the iteration rounds of both datasets to 200. Fourth, we set the number of malicious attackers m in the computing environment to $k\% \times n$, where $k\%$ is the expansion factor of the number of attackers. In addition, attackers can inject fn fake nodes to participate in updates. All experimental test results are implemented through the PyTorch library and obtained by repeating the test 5 times. For the attack setups, we consider implementing all perception poisoning attacks on VOC. We tamper with the object label whose source class is "person" to the target class "pottedplant" for label flipping poisoning, and shrink the bounding box to 0.1 times its original size. Since the INRIA dataset only contains objects of the "person" class, we focus on bounding box poisoning and object existence poisoning.

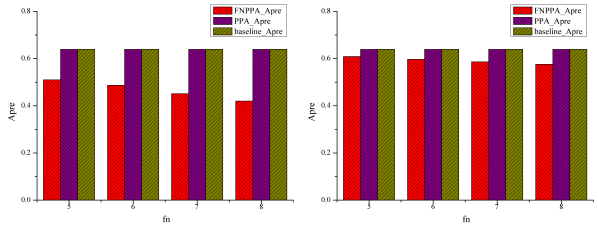
5.2 Evaluation Metric and Baseline

We utilize three perception poisoning attacks on the VOC dataset, and count the average precision ($APre$) of their source class object



(a) Label flipping poisoning of VOC (b) Bounding box poisoning of VOC (c) Object existence poisoning of VOC

Figure 5: Evaluation *Apre* of perception poisoning attacks on VOC dataset, for different fn, the k is set to 10%.



(a) Bounding box poisoning of INRIA (b) Object existence poisoning of INRIA

Figure 6: Evaluation *Apre* of perception poisoning attacks on INRIA dataset, for different fn, the k is set to 10%.

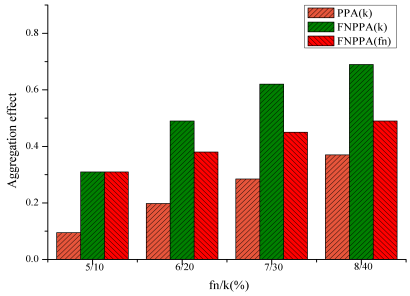


Figure 7: Evaluation of aggregation effect of two datasets.

detection as our evaluation metric. The INRIA dataset counts the latter two. We conduct clean testing on two datasets with untampered samples, and record the precision of their source class as baseline. Secondly, we compare the performance of FNPPA with the state-of-the-art perception poisoning attacks technology PPA proposed in [9]. To demonstrate the fairness of the comparison of performance results, the baseline, PPA, and FNPPA all utilize the same datasets and model, FL training setups, attack setups, evaluation metric, and FL aggregation rule previously detailed.

5.3 Performance Analysis of FNPPA

5.3.1 Evaluation of the effect of k. In this part, we explore the impact of k on the attack performance, where each malicious attacker only injects 5 fake nodes. One can see from Fig. 3, which reports the *APre* of source classes for the VOC dataset under three perception poisoning attacks states as k increases. It can be observed that when k is at the minimum setting, FNPPA has a significant decline compared with PPA and baseline. When k is at the maximum, the effect is most obvious. The reason is that our FNPPA controls more malicious adversaries to participate in the aggregation, achieving more intuitive attack effectiveness. Fig. 4 shows the effectiveness of the latter two perception poisonings of the INRIA dataset. We can obtain similar performance effects to those of the VOC dataset.

5.3.2 Evaluation of the effect of fn. Edge smart devices participating in training in a mobile computing environment are at great risk of going offline, which gives malicious attackers greater attack space when injecting fake nodes. Therefore, in this part, we explore the impact of fake node fn on the attack effect, of which only 10% is retained by the malicious attacker k. One can see from Fig. 5, which reports the *APre* of the source classes of three perception poisoning attacks for the VOC dataset under the condition of increasing fn. It can be clearly noticed that although both PPA and FNPPA show a lower *APre* than the baseline, the decreasing effect of FNPPA is more significant. Similar performance is also demonstrated in Fig. 6. Because fn helps malicious attackers participate in poisoning aggregation on a larger scale and cover clean updates as much as possible. We can also find from Fig. 7, where $PPA(k)$ represents the aggregation effect only when k gradually increases, $FNPPA(k)$ and $FNPPA(fn)$ respectively represent the aggregation effect under the condition that as k gradually increases, fn is set to 5 and as fn gradually increases, k is set to 10%. With the assistance of fn, FNPPA shows a more powerful aggregation effect than PPA, so such a more prominent attack effectiveness is also foreseeable.

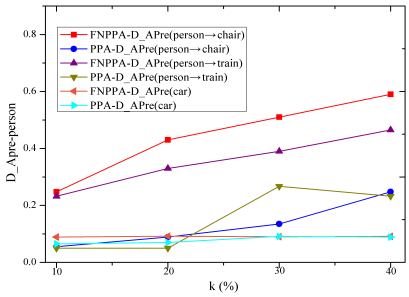


Figure 8: Evaluate the impact of different target classes.

5.3.3 Evaluation of different target classes. In this part, we explore the impact of implementing perception poisoning attacks of different target classes on object detection systems. We designate "person" as the source class, and set the target class as the one that is most likely to be mislabeled ("chair") and the target class that is most difficult to be mislabeled ("train"). One can see from Fig. 8, that as k increases, the $D_{APre-person}$ of FNPPA gradually increases, where D_{APre} represents the difference between $APre$ in the poisoned state and the non-poisoned state. FNPPA shows a more obvious difference change than PPA, which is an intuitive effect of the $Apre$ reduction caused by our injected fn. In addition, $D_{APre-car}$ is relatively stable, where we use "car" to represent the precision changes of other classes in the poisoning state. It shows that both FNPPA and PPA exhibit similar performance, and they will have almost no negative impact on the normal detection of other classes. Experimental results show that our attack is targeted enough to have a strong negative effect on the source class without harming the normal detection of objects in other classes.

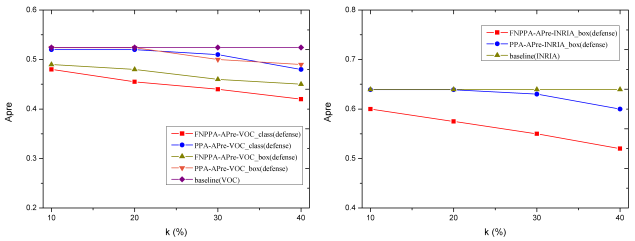
5.3.4 Evaluation of the effectiveness against defense strategy. In this part, we explore the attack effectiveness of our proposed FNPPA under the influence of defensive measures. We utilize an advanced robust clustering strategy [9] as a defense mechanism for the system, which primarily identifies malicious effects through statistical properties in the parameter contributions of malicious and clean participants and further eliminates malicious updates. One can see from Fig. 9, that under the influence of the defense strategy, the poisoning effects of PPA and FNPPA are alleviated to varying degrees under various settings, which is expected. We also noticed that FNPPA still has a higher $APre$ than PPA. The reason for maintaining such attack performance is that our FNPPA always maintains higher malicious updates than PPA under the same settings and covers shared clean parameters as much as possible.

6 CONCLUSION

This paper explores a more threatening and efficient fake node-based perception poisoning attacks strategy (FNPPA) to target security vulnerabilities in training federated object detection systems in mobile computing networks. Malicious attackers in the system implement covert poisoning attacks in local private data, and inject multiple fake nodes to participate in aggregation during the training phase, aiming to make the local poisoning model more likely to cover clean updates. They pursue greater malicious influence on target objects without affecting the normal detection of other objects. We demonstrate through extensive experiments under multiple datasets that the proposed FNPPA exhibits better attack impact than the state-of-the-art on multiple evaluation dimensions.

ACKNOWLEDGMENTS

The work is supported by the National Natural Science Foundation of China (62225205, 92055213, 62302157, U23A20317, 62172146), Natural Science Foundation of Hunan Province of China (2021JJ10023), the Science and Technology Program of Changsha (kh2301011), Shenzhen Basic Research Project (Natural Science Foundation) (JCYJ20210324140002006), the China Postdoctoral Science Foundation (No. 2023M741124), Natural Science Foundation of Changsha of China (kq2208042).



(a) Defense on VOC (class & box) (b) Defense on INRIA (box)

Figure 9: Evaluation $Apre$ against defense strategy on VOC and INRIA dataset, for different k, the fn is set to 5.

REFERENCES

[1] Andrew Hard et al. 2018. Federated learning for mobile keyboard prediction. *arXiv:1811.03604* (2018).
[2] Bingyan Liu et al. 2021. DistFL: Distribution-aware federated learning for mobile scenarios. In *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*. 1–26.
[3] Brendan McMahan et al. 2017. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. 1273–1282.
[4] Chen Zhang et al. 2021. A survey on federated learning. *Knowledge-Based Systems* 216, 106775 (2021).
[5] Deepthi Jallepalli et al. 2021. Federated learning for object detection in autonomous vehicles. In *Proceedings of the 2021 IEEE Seventh International Conference on Big Data Computing Service and Applications (BigDataService)*. 107–114.
[6] Eugene Bagdasaryan et al. 2020. How to backdoor federated learning. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. 2938–2948.
[7] Jin-Hua Chen et al. 2021. BDFL: A byzantine-fault-tolerance decentralized federated learning method for autonomous vehicle. *IEEE Transactions on Vehicular Technology* 70, 9 (2021), 8639–8652.
[8] Jiachun Li et al. 2022. A federated learning based privacy-preserving smart healthcare system. *IEEE Transactions on Industrial Informatics* 18, 3 (2022), 2021–2031.
[9] Ka-Ho Chow et al. 2021. Perception poisoning attacks in federated learning. In *Proceedings of 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*.
[10] Ka-Ho Chow et al. 2023. STDLens: Model hijacking-resilient federated learning for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16343–16351.
[11] Kan Xie et al. 2022. Efficient federated learning with spike neural networks for traffic sign recognition. *IEEE Transactions on Vehicular Technology* 71, 9 (2022), 9980–9992.
[12] Limeng Qiao et al. 2021. DeFRCN: Decoupled Faster R-CNN for few-shot object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 8681–8690.
[13] Mingzhe Chen et al. 2020. Wireless communications for collaborative federated learning. *IEEE Communications Magazine* 58, 12 (2020), 48–54.
[14] Tao Kong et al. 2017. RON: Reverse connection with objectness prior networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 5936–5944.
[15] Vale Tolpegin et al. 2020. Data poisoning attacks against federated learning systems. In *Proceedings of the European Symposium on Research in Computer Security*. 480–501.
[16] Wei Yang Bryan Lim et al. 2020. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials* 22, 3 (2020), 2031–2063.
[17] Xiong Xiao et al. 2022. SBPA: Sybil-based backdoor poisoning attacks for distributed big data in AIoT-based federated learning system. *IEEE Transactions on Big Data* (2022), 1–12.
[18] Xiong Xiao et al. 2023. FDSFL: Filtering Defense Strategies Towards Targeted Poisoning Attacks in IIoT-based Federated Learning Networking System. *IEEE Network* 37, 4 (2023), 153–160.
[19] Xiong Xiao et al. 2023. SCA: Sybil-based collusion attacks of IIoT data poisoning in federated learning. *IEEE Transactions on Industrial Informatics* 19, 3 (2023), 2608–2618.
[20] Youngjoon Yu et al. 2022. Defending person detection against adversarial patch attack by using universal defensive frame. *IEEE Transactions on Image Processing* 31 (2022), 6976–6990.