Privacy-Preserving Edge Video Analytics

Miao Hu, Yao Fu, and Di Wu

1 Introduction

Edge video analytics (EVA) shows great potential to be applied in artificial intelligence-driven system design, e.g., autonomous driving and smart city, and has been an area of intense research in the past few years. Given a video clip taken from cameras, edge video analytics is the process of extracting features from images/videos and applying these features to emerging applications, e.g., finding the criminal from the crowd. Apart from surveillance, it has applications in robotics, multimedia and forensics. The design and implementation of an efficient EVA model has always been an essential issue, which has received significant attention from both the computer vision research community and the computer system research community due to its wide applicability and utility.

In recent years, the privacy issues in edge video analytics aroused tremendous attentions. In summary, the privacy issues can be classified into the following categories: (1) *Privacy in video collection*. In most video analytics applications, the video collector (e.g., the camera) does not locally conduct the tasks due to resource limitation. Generally, the videos will be offloaded to other computation-powerful servers for model training and task execution. In the video collection and offloading process, video contents are inevitably at the risk of private information leakage [12, 28, 30, 33]. (2) *Privacy in video storage*. It is also a big concern to determine where should the collectors store their video contents with private information. Reliable data storage is important for video analysis tasks, especially

M. Hu · Y. Fu · D. Wu (✉)
The School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China

Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China
e-mail: humiao5@mail.sysu.edu.cn; fuyao7@mail2.sysu.edu.cn; wudi27@mail.sysu.edu.cn

W. Chang, J. Wu (eds.), *Fog/Edge Computing For Security, Privacy, and Applications*, Advances in Information Security 83,
https://doi.org/10.1007/978-3-030-57328-7_7

for a video retrieval application. Some studies (e.g., [7] and [28]) applied a trusted edge server performing video denaturing and providing data storage. (3) *Privacy in video analytics*. The privacy issues in video analytics-related applications attract more and more attentions, such as [14, 15, 18]. The video analytics applications are generally built on neural network models. Intuitively, some works persist that the privacy issues are solved as only network parameters are shared. However, network parameters can be leveraged to reveal private information of video contents. The privacy issues still cannot be ignored.

Despite the privacy issues mentioned above, the design of a privacy-preserving edge video analytics system is especially challenging: *First*, the goals of improving EVA model accuracy and protecting client privacy are somewhat conflicted with each other. Commonly, the model performance relies on the accuracy of model parameters trained on each dataset. A higher EVA model accuracy is always at the cost of more disclosure of privacy on clients' video contents. *Second*, existing crypto-based privacy-preserving approaches (e.g., homomorphic encryption) are too compute-intensive, which are not practical to be implemented in the real systems [14, 15, 18]. A light-weight privacy-preserving EVA model with low complexity is more desirable. *Third*, most popular privacy-preserving schemes (e.g., differential privacy) usually face with serious performance degradation [9, 31]. Besides, it is also unclear how the EVA model accuracy varies under different privacy conditions. *Last but not the least*, the EVA model training condition might change with time. It is expected to design a flexible EVA model training framework that can be dynamically tuned to fit all scenes. To solve the privacy issues in EVA systems, researchers have made significant progress in the past years. However, the above four key challenges have not been well addressed.

In this chapter, our objective is to address the above mentioned challenges and achieve both high accuracy and privacy preservation simultaneously. To this purpose, we propose a light-weight federated learning framework for EVA model training, which is called *FedEVA*. The main idea of *FedEVA* is to perturb clients' private model parameters with publicly available model parameters using a local perturbation operation, while the model training algorithms can still be run over perturbed parameters. There is no need for the coordinator server to recover the private model parameters. Note that, compared with compute-intensive crypto operations, both perturbation and model aggregation in our *FedEVA* framework are linear operations with low time and space complexity. Thus, *FedEVA* is light-weight and highly efficient. In addition, our framework can ensure the same recommendation accuracy on the perturbed parameters as that on the original parameters. It is possible for adversaries to capture perturbed model parameters, but they cannot easily recover the private model parameters of clients due to the lack of perturbation key. Thus, *FedEVA* can effectively prevent the leakage of user privacy.

Overall, our main contributions in this chapter can be summarized as below:

- We propose a light-weight privacy-preserving EVA framework called *FedEVA*. By conducting perturbation over private EVA model parameters, our proposed

framework can prevent the leakage of client personal information. We also verify that the model accuracy will not be affected by introducing *FedEVA* into the EVA model training systems.
- Different from previous crypto-based methods, *FedEVA* is light-weight and highly efficient. In the meanwhile, the EVA model training can still work well on the perturbed parameters. We can also dynamically change the weight parameters of the *FedEVA* framework to achieve different degree of privacy preservation, which improves the scalability and flexibility of a privacy-preserving EVA system.
- We evaluate our *FedEVA* framework through large-scale real-world datasets. The experiment results show that our *FedEVA* framework achieves a significant improvement in protecting client privacy compared to baselines, meanwhile the accuracy and efficiency of the EVA model can still be guaranteed.

The remainder of this chapter is organized as follows. Section 2 introduces the historical backgrounds for edge video analytics and standard privacy-preserving solutions. Section 3 presents the preliminary information. Section 4 proposed the system model, where the detailed algorithms are shown in Sect. 5. Section 6 evaluates the performance of the proposed algorithms. Section 7 concludes this chapter and outlooks future research directions.

2 Related Works

2.1 Edge Video Analytics

Utilization of edge computing in video analytics can help save cost, bandwidth and energy. To design an efficient edge computing system for real-time video stream analytics, characteristics of video contents should also be taken into consideration. Bilal and Erbad [3] highlighted potentials and prospects of edge computing for interactive media, and presented some preliminary works on how edge computing can be used to tackle video analytics challenges. Ran et al. [22] designed a framework tied together front-end devices with more powerful backend “helpers” to allow deep learning to be executed locally or remotely in the cloud/edge. They considered the complex interaction among model accuracy, video quality, battery constraints, network data usage, and network conditions to determine an optimal offloading strategy. Wang et al. [29] proposed an adaptive wireless video transcoding framework based on the edge computing paradigm by deploying edge transcoding servers close to base stations. It is essential to efficiently extract video characteristics, and then apply them into edge computing architecture design.

Some special characteristics from video content or edge servers can be further exploited to enhance the EVA system efficiency. Zhang et al. [34] presented Vigil, which contributed on frame selection to suppress redundancy. When multiple cameras capture different views of an object or person of interest to the user query, Vigil uploads only the frames that best capture the scene. Chowdhery and Chiang [6] proposed a model predictive compression algorithm that uses predicted drone trajectory to select and transmit the most important image frames to the ground station to maximize the application utility while minimizing the network bandwidth consumption.

The model training is one of the most important issues for EVA system design, which has also attracted tremendous attentions. Liu et al. [20] proposed a general deep neural network (DNN) architecture, and used it in their edge-based video analytics system. They also implemented a buffer management scheme, which uses Nvidia CUDA mapped memory feature to simplify the memory movement between CPU and GPU. Tarasov and Savchenko [25] proposed to apply the multi-task cascaded convolutional networks to obtain facial regions on each video frame. Then image features are extracted from each located face using preliminary trained convolutional neural networks (CNNs). Yaseen et al. [32] focused on tuning hyper-parameters associated with the deep learning algorithm to construct the video analytics model. Uddin et al. [27] proposed SIAT, which provided basic distributed video processing APIs and distributed dynamic feature extraction APIs which extract prominent information from the video data. In this chapter, we focus on the privacy-preserving EVA model training framework design.

2.2 Privacy-Preserving Video Analytics

For video analytics applications, a large number of surveillance cameras installed at different places capture video data for further analytics. The privacy issues have been focused in the video analytics framework design, which allows users to have specific control over their sensitive information, preventing from being abused by third parties.

With the advance of computer version (CV) technologies, it is possible to achieve a more granular privacy protection in the initial stage of data collection. Wang et al. [28, 30] developed OpenFace, a mechanism for privacy-preserving data collection, which denatures captured video data based on user-defined privacy policy. By applying video denaturing technology, OpenFace can selectively blur faces that occur in video frames, greatly alleviating the privacy concern. Zarepour et al. [33] blurred or eliminated the sensitive subjects with image processing technologies. Jana et al. [12] proposed DARKLY, which is integrated with OpenCV and replace the raw input feature with opaque references, which cannot be directly dereferenced by an untrusted application. However, the characteristics of the pictures or videos may be changed and cannot be applied into the video analytics.

During the data analysis phase, sensitive information may need to be processed on untrusted platforms due to the limited computing power of edge nodes. During the analysing stage, crypto technologies can reduce unauthorized data access. Fully homomorphic encryption allows data analytics being performed on encrypted data directly, instead of applying an additional decryption operation on the video. Jiang et al. [14] used level homomorphic encryption and performed privacy preserving scale-invariant feature transform on encrypted images. To enable encryption operations to run on resource constrained nodes, [15] proposed TargetFinder, which applies homomorphic encryption to search for images that include the target on encrypted data. Besides, TargetFinder used optimization technologies to reduce computation overhead of cryptographic primitives on energy and computation-constrained edge devices. Besides, [18] proposed a novel privacy preserving computing framework, where the terminal devices perform the light-weight permutation-substitution encryption and edge nodes adopt the homomorphic encryption. The edge-assisted framework can greatly reduce the computational, communication and storage burden while ensuring data security.

Federated learning facilitates the collaborative training of models without the sharing of raw data. By averaging local gradient updates, federated averaging (FedAvg) proposed by McMahan et al. [21] performs well on federated learning with non-iid data. However, [2, 17] demonstrated that simply maintaining data locality during training processes does not provide sufficient privacy guarantees. Recently, some privacy-preserving frameworks were proposed to hide clients' contributions during training, balancing the trade-off between privacy loss and model performance. Bonawitz et al. [4] designed a communication-efficient, failure-robust protocol for secure aggregation of high-dimensional data.

To obscure an individual's identity, differential privacy (DP) adds mathematical noise to a small sample of the individual's usage pattern. For example, [10] proposed an algorithm for client sided differential privacy preserving federated optimization. Wu et al. [31] used noisy differentially-private gradients to minimize the fitness cost of the federated learning model using stochastic gradient descent. However, differential privacy might lead to slow convergence on model training, and possibly low accuracy given a large number of parties with relatively small amounts of data.

It is non-trivial to design a federated learning system capable of preventing interference over training on the distributed datasets while ensuring the resulting model also has acceptable predictive accuracy. Some federated learning approaches use secure multiparty computation (SMC) to accelerate the model convergence rate, however this might introduce a high burden on the communications between clients. Moreover, SMC is vulnerable to interference. Truex et al. [26] proposed a scalable approach to protect against interference threats, which combines differential privacy with secure multiparty computation. However, it is unclear how to realize the SMC operations without introducing high computation overhead. To the authors' best knowledge, it is still a great challenge to balance the tradeoff between the training performance and the privacy-preserving level. This chapter aims to propose a privacy-preserving federated learning system without affecting the performance of video analytics model.

3 Technical Preliminary

Before presenting the system model and the proposed privacy-preserving edge video analytics scheme, we first introduce some preliminary knowledge.

3.1 Federated Optimization

Federated learning, as a federated optimization paradigm, makes multiple devices to jointly learn a global objective model without sharing their local data. The training data is stored on the local clients or edge devices, which can prevent the leakage of user privacy. Similar to distributed optimization, federated learning allows clients or edge devices to compute local updates and a coordinator server makes use of the gradients sent by user devices to aggregate global model parameters.

Following the pioneer work proposed by McMahan et al. [21], we also assume a synchronous update scheme that proceeds in rounds of communication. There is a fixed set of K clients with fixed datasets $\mathscr{D}_1, \mathscr{D}_2, \cdots, \mathscr{D}_K$ where $n_k = |\mathscr{D}_k|$. At the beginning of each round, a random fraction C of clients is selected, and the server sends the current global algorithm state to each of these clients (e.g., the current model parameters). We only select a fraction of clients for efficiency. Each selected client then performs local computation based on the global state and its local dataset, and sends an update to the server. The server then applies these updates to its global state, and the process repeats (Fig. 1).

While we focus on non-convex neural network objectives, the algorithm we consider is applicable to any finite-sum objective of the form

$$\min_{\omega} \quad f(\omega), \tag{1}$$

where

$$f(\omega) = \frac{1}{n}\sum_{i=1}^{n} f_i(\omega). \tag{2}$$

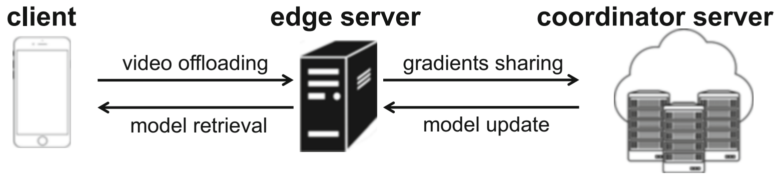

Fig. 1 Federated learning driven edge video analytics model training framework

For a machine learning problem, we have $f_i(\omega) = l(x_i, y_i; \omega)$, where the loss of the prediction on example (x_i, y_i) made with model parameters ω. Thus, we can re-write the objective in Eq. (1) as

$$f(\omega) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(\omega), \tag{3}$$

where

$$F_k(\omega) = \frac{1}{n_k} \sum_{i \in \mathscr{D}_k} f_i(\omega). \tag{4}$$

The recent video analytics applications of deep learning have almost exclusively relied on variants of stochastic gradient descent (SGD) for optimization. SGD can be applied naively to the federated optimization problem, where a single batch gradient calculation (say on a randomly selected client) is done per round of communication. This approach is computationally efficient, but requires very large numbers of rounds of training to produce good models.

3.2 Federated Averaging

Federated Averaging (FedAvg) is the most common algorithm for federated learning, by optimizing the local objective F_k on client k in each round t. T is defined to indicate the number of total rounds of client-server communications and parameter updates to produce global model. In FedAvg, when each client locally executes the procedure of gradient descent on the current model using its local data, we have

$$\omega_{t+1}^{k} = \omega_t - \eta_t \nabla F_k(\omega_t, \xi_t^k), \tag{5}$$

where ω_t are the current model parameters, η_t is the learning rate, ξ_t^k and ω_{t+1}^k are local data used and new parameters, respectively. The global iteration step in the server is to take an average of the locally updated results and obtain a new global model, namely,

$$\omega_{t+1} = \sum_{k \in S_t} \frac{n_k}{n} \omega_{t+1}^{k}. \tag{6}$$

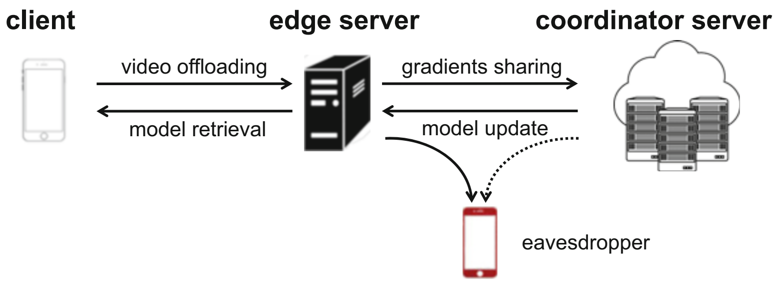

Fig. 2 The general eavesdropper attack model

3.3 Attack Model

The attacker takes advantage of unsecured network communications to access data as it is being sent or received by its user. This chapter mainly focuses on the eavesdropping attack, also known as the sniffing or snooping attack. As shown in Fig. 2, an eavesdropper might monitor the information transmitted over a network by a computer, smartphone, or another connected device. Thus some key features of the local captured videos are at risk of information disclosure.

4 *FedEVA* Framework Overview

The focus of the *FedEVA* framework is to protect the private information (e.g., video analytics results) in clients' video contents from being leaked. We utilize a helper with publicly available training datasets to guarantee the privacy-preserving property while not introducing much communication or computation overhead. To the authors' best knowledge, this concept is first introduced in this work. We will present more design details in the following.

The *FedEVA* framework contains four key components, i.e., the client, the edge server, the helper and the coordinator server.

- **Client**. The client is responsible for collecting the video contents and offloading them to the edge server. As the video source, the client can be the camera, the cellphone, the laptop, and so on. Generally, the client side does not have much computation resource for model training.
- **Edge server**. The edge servers train the EVA model based on the video contents collected from its connected clients. Generally, the edge server is in the proximity of its connected clients. We assume that the video offloading process is executed on a dedicated communication channel, which cannot cause the leakage of information. This work focuses on the possible leakage of the learned EVA model parameters from the edge server to the coordinator server.

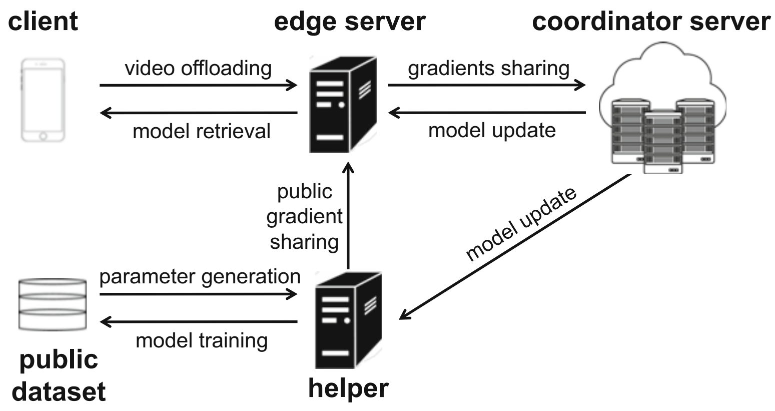

Fig. 3 *FedEVA*: a federated learning driven edge video analytics framework

- **Helper**. The helper is associated with pre-downloaded video datasets. The implementation of the *helper* is one of the key contributions of the proposed *FedEVA* framework. Since the training results from the helper will be shared to other edge servers, the helper only runs on the public datasets to prevent the disclosure of private information.
- **Coordinator server**. The coordinator assists all the edge servers to collaboratively train models using *FedAvg* or *FedSGD* algorithm.

As shown in Fig. 3, the *FedEVA* framework contains data or information workflow between the modules. We introduce the key procedures as follows.

- *Video offloading*. Due to the limited computation resources, the client does not locally execute the model training function. Instead, each client periodly offloads its generated video clips regarded as the model inputs to the nearby edge server in connection.
- *Distributed model training*. The model training is conducted on the edge server with image/video inputs collected from clients. In this work, we do not propose a new model training algorithm, and any model training schemes can be applied to our *FedEVA* framework. In our *FedEVA* framework, the model training process can be divided into two categories.
  - *Model training on the private datasets*. The private clients are recruited to help enhance the performance of the trained EVA model. The incentive mechanisms are out of the scope of this work, which can be found in some other literatures, e.g., [24].
  - *Model training on the public datasets*. The helper proposed in our *FedEVA* framework is to help guarantee the privacy-preserving level of the analytics results from the private clients.

- *Public gradients sharing*. In the *FedEVA* framework, the privacy issues on the client sides are guaranteed by the setting of "public gradients sharing". Each client will receive the identical public gradients from the helper. By adding different weights to the public gradients, the actual gradient update based on the private video dataset can thus be guaranteed.
- *Model update*. In the coordinator server, the perturbed gradients will be summed to obtained a synchronized EVA model. The "model update" procedure will continue to update until the EVA model approaches a rather good level of performance.

The *FedEVA* framework can be integrated into any edge-based video analytics system. The model training framework can work well on the perturbed parameters without recovering the practical parameters from each local user. As modifications are mostly made at the user side, it is easy to rapidly deploy the *FedEVA* framework over the existing edge-based video analytics systems. One limitation is that the perturbation operation requires the use of model parameters trained on the public dataset, which may not be always available in certain scenarios. One feasible solution is to design incentive mechanism to promote some users denoting their datasets as the public ones.

5 *FedEVA* Algorithm Design and Analysis

This section will introduce the designed algorithm in our *FedEVA* framework for each module. As the client will not need to conduct a lot of operations, we will present the algorithms designed for the helper, the edge server, and the coordinator server, respectively.

5.1 Algorithm Design

5.1.1 Algorithm 1: Helper Algorithm

The operation on the helper is similar to that in standard federated learning, as illustrated in Algorithm 1.

Algorithm 1 Helper algorithm

```
1: HelperUpdate($\omega$): // run on the helper
2: $\mathscr{B} \leftarrow$ (split $\mathscr{D}'$ into batches of size $B$)
3: for batch $b \in \mathscr{B}$ do
4:     $\omega' \leftarrow \omega - \eta \nabla l(\omega; b)$
5: end for
6: return $\omega'$ to each edge server
```

5.1.2 Algorithm 2: Edge Server Algorithm

Different from the operation on the helper, the model parameter update will not be directly sent to the coordinator server due to the privacy concerns. Instead, the model update ω will be perturbed with the public update ω' returned from the helper. In detail, we will define a weight parameter α and perturb the local model update ω with a weighted public update ω', i.e.,

$$\omega = \omega + \alpha\omega', \tag{7}$$

where $\alpha \sim Lap(0, b)$ can be fit with the Laplace distribution following [8], and b is the scale parameter. The variance of parameter α is $\sigma^2 = 2b^{-2}$. Note that the privacy level increases with the value of σ. However, the value of σ cannot be increased without limit. A large value of σ will make the coordinated model tend to that trained in the helper. Afterwards, the perturbed model update will be sent to the coordinator server.

Algorithm 2 Edge server algorithm

```
1: EdgeUpdate($k$, $\omega$): // run on edge server $k$
2: $\mathscr{B} \leftarrow$ (split $\mathscr{D}_k$ into batches of size $B$)
3: for batch $b \in \mathscr{B}$ do
4:     $\omega \leftarrow \omega - \eta\nabla l(\omega; b)$
5: end for
6: randomly generate a weight $\alpha \sim Lap(0, b)$, and obtain $\omega = \omega + \alpha\omega'$
7: return $\omega$ to the coordinator server
```

5.1.3 Algorithm 3: Coordinator Server Algorithm

The global iteration step in the coordinator server is to take an average of the locally updated results and obtain a new global model. The model update is as below:

$$\omega_{t+1} = \sum_{k \in S_t} \frac{n_k}{n}\omega_{t+1}^{k}. \tag{8}$$

5.2 *Analysis on Privacy Preservation*

To quantify the level of privacy preservation provided by our *FedEVA* framework, we consider the existence of malicious attackers (e.g., eavesdroppers). According to

Algorithm 3 Coordinator server algorithm

```
1: initialize $\omega_0$
2: for each round $t$ do
3:     $\mathscr{S}_t \leftarrow$ (random set of $m$ clients)
4:     for each client $k \in \mathscr{S}_t$ in parallel do
5:         $\omega_{t+1}^k \leftarrow$ PrivateUpdate$(k, \omega_t)$
6:     end for
7:     $\omega_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{n} \omega_{t+1}^k$
8: end for
9: return $\omega$ to the coordinator server
```

[5, 13], a privacy preservation metric, called *Privacy Preserving Distance (PPD)*, is defined as below.

Definition 1 (Privacy Preserving Distance (PPD)) is defined as the difference between the real model parameters and the perturbed model parameters. That is,

$$\text{PPD} = \frac{\sum_{u=1}^{M} ||\mathbf{r}_u - \check{\mathbf{r}}_u||_2}{M}. \tag{9}$$

Specifically, a higher PPD is desirable for privacy protection, which can prevent the attackers from obtaining the original private model parameters.

Based on the distribution of PPDs, we further introduce another privacy preserving metric, called *Privacy Preserving Indicator (PPI)* as following.

Definition 2 (Privacy Preserving Indicator (PPI)) For a *FedEVA* user u, let $\mathbf{r}_u$ denote its real model parameter vector and $\check{\mathbf{r}}_u$ denote the perturbed model parameter vector. Then, we have the privacy preservation indicator defined as:

$$I_p = \Pr\left\{ \frac{\sum_{u=1}^{M} ||\mathbf{r}_u - \check{\mathbf{r}}_u||_2}{M} \le \epsilon_p \right\}, \tag{10}$$

where ϵ_p denotes a privacy preservation distance threshold.

Given a specific ϵ_p, the value of I_p indicates the per-rating distortion probability between the perturbed model parameter vector and the original model parameter vector. A lower I_p indicates a better privacy preservation degree, and vice versa.

The private model parameters are perturbed by individual users before transmitting them to the coordinator server. Thus, only perturbed model parameters $\check{\mathbf{r}}_u$ can be captured by attackers. For a local *FedEVA* user u, the privacy preservation indicator can be rewritten as:

$$I_{p,u} = \Pr\left\{||\mathbf{r}_u - \check{\mathbf{r}}_u||_2 \le \epsilon_p\right\}. \tag{11}$$

Note that the perturbed model parameter vector is not only related to real model parameters $\mathbf{r}_u$ but also related to the model parameters trained on the public dataset.

To prove the privacy level of our *FedEVA* framework, we compare it with the standard differential privacy (DP) scheme whose noise w is drawn from Laplace distribution, i.e., $Lap(0, \lambda_{DP}) = \frac{1}{2\lambda_{DP}} \exp\left(-\frac{|x|}{\lambda_{DP}}\right)$, where $\lambda_{DP} = \frac{\Delta \mathbf{r}_u}{\epsilon}$ presented by Dwork and Roth [8]. We obtain the following proposition.

Proposition 1 *For a local FedEVA user u, when the adding noise α follows the Laplace distribution $\alpha \sim Lap(0, \lambda_0)$, we can achieve*

$$I_{p,u}^{FedEVA} = I_{p,u}^{DP}, \quad \forall u \in \mathcal{U}, \tag{12}$$

where $\lambda_0 = \frac{\lambda_{DP}}{\|\mathbf{r}'\|_2} = \frac{\Delta \mathbf{r}_u}{\epsilon \|\mathbf{r}'\|_2}$.

Proof Starting from the Algorithm 3, we have

$$\|\mathbf{r}_u - \check{\mathbf{r}}_u\|_2 = |\alpha| \|\mathbf{r}'\|_2.$$

Therefore, we have

$$I_{p,u}^{FedEVA} = \Pr\{|\alpha| \le \frac{\epsilon_p}{\|\mathbf{r}'\|_2}\} = 1 - \exp(-\frac{\epsilon_p}{\lambda_0 \|\mathbf{r}'\|_2}). \tag{13}$$

Similarly, for ϵ-DP, we have

$$I_{p,u}^{DP} = \Pr\{|w| \le \epsilon_p\} = 1 - \exp\left(-\frac{\epsilon_p}{\lambda_{DP}}\right). \tag{14}$$

Combining (13) and (14), we obtain

$$I_{p,u}^{FedEVA} = I_{p,u}^{DP}$$
$$\exp(-\frac{\epsilon_p}{\lambda_0 \|\mathbf{r}'\|_2}) = \exp\left(-\frac{\epsilon_p}{\lambda_{DP}}\right).$$

Hence to ensure same privacy level, we set $\lambda_0 = \frac{\lambda_{DP}}{\|\mathbf{r}'\|_2}$.

6 Performance Evaluation

In this section, we conduct field measurements to evaluate the performance of the proposed *FedEVA* scheme.

6.1 Experiment Settings

We reform the YOLOv3-tiny model [23] into the federated learning setting based on an open source implementation[1] and train the model on the COCO dataset [19]. To evaluate the performance of the *FedEVA* scheme, we compare it with other state-of-the-art algorithms.

- **Central YOLOv3-tiny algorithm**. YOLO is a real-time object detection algorithm that has been widely applied in video analytics.
- **Standard FedAvg algorithm**. *FegAvg* is a classical algorithm in FL which allows many clients to train a model collaboratively without sharing private data between clients or with the server, which can provide a certain level of privacy.
- **FedAvg+LDP algorithm**. Differential privacy (DP) describes the patterns of the dataset while withholding information about individuals in the dataset. Local DP (LDP) adds noise to each client's update before sharing with the server, and guarantees much stronger protection to clients' privacy.

We set the batch size as 8 and use multi-scale training. Following [16], we use Adam optimizer and gradient accumulations. In other words, the model is updated every two batches. The standard YOLOv3-tiny model is trained based on [23]. In the FL setting, we equally partition the COCO dataset into $N = 10$ parts and give each client one part. We carry out experiments on both IID and non-IID data, where the IID assumption is typically made by distributed optimization algorithms and the non-IID data is assumed by the federated optimization algorithms. On the IID case, we randomly shuffle the data before assigning to clients. On the non-IID case, we sort all data in the order of image type, which is accessible in the COCO dataset. After that, the data of the same type are aggregated together and each client will only have a few types of data.

In each global epoch, all clients participate the training process by receiving the model from the server and training it locally. We set local epoch $E = 1$. In *FedAvg*, all clients' updates are averaged with the weight of their data count. In *FedEVA*, one client is presumed as the public edge server; while other nine clients are private edge servers. The value of α in *FedEVA* is generated by Laplace distribution $Lap(0, b)$, where $b = 0.15$. In local DP setting, to guarantee the privacy, we clip the update with $C = 30$, which was introduced by Abadi et al. [1].

We achieve $(200, 10^{-9})$-DP using Laplace mechanism. Holohan et al. [11] proved if $b \geq \frac{C}{\epsilon - \log(1-\delta)}$, $Lap(0, b)$ satisfies (ϵ, δ)-DP. In our experiment, we have $b = \frac{30}{200 - \log(1-10^{-9})} \simeq 0.15$ so that we can compare *FedEVA* and *FL+LDP* fairly. The Laplace function we use is shown in Fig. 4. We generate 10^6 random numbers with respect to $Lap(0, 0.15)$ and count the number of points falling in every interval.

The experiment results are compared with two performance metrics, including:

[1] https://github.com/eriklindernoren/PyTorch-YOLOv3.

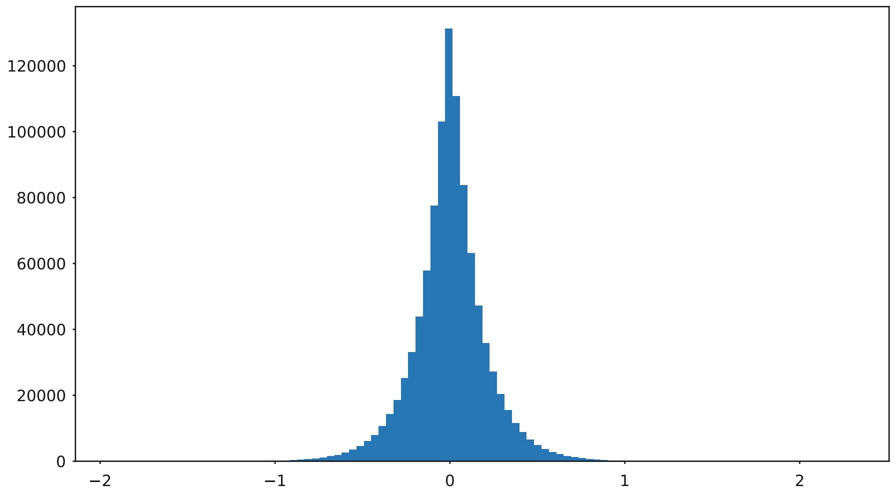

Fig. 4 PDF of Laplace function $Lap(0, 0.15)$

- *Model accuracy*. We use the mean Average Precision (mAP) as the metric to evaluate the EVA model accuracy.
- *Privacy level*. We use the privacy preserving distance (PPD) in Proposition 1 as the metric for evaluating the privacy level. A larger PPD indicates a better privacy level, and vice versa. We assume the central training model is noise-free, i.e., the actual model.

We train YOLO model for 60 epochs and evaluate the model after each epoch.

6.2 Experiment Results

The experiments are conducted under both IID case and non-IID case.

6.2.1 IID Case

Figure 5 compares the mAP during training process with different FL models and central training. The x-axis represents the epoch number, while the y-axis represents the mAP of the aggregated model in each epoch. Note that the mAP metric of *FedAvg+LDP* scheme is 0 during 60 epochs, i.e., the *FedAvg+LDP* scheme diverges in the EVA model training. From this experiment, we can observe the mAP performance of all distributed versions of the YOLO algorithm is worse than that of the central version. We also find that the mAP metric of our *FedEVA* scheme is similar to that with the standard *FedAvg* scheme. This experiment indicates the

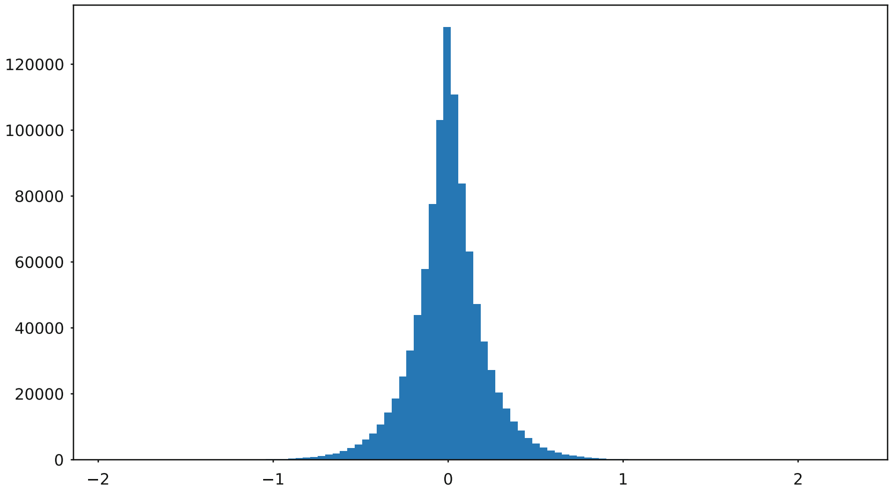

Fig. 5 Comparison of mAP on the IID COCO dataset

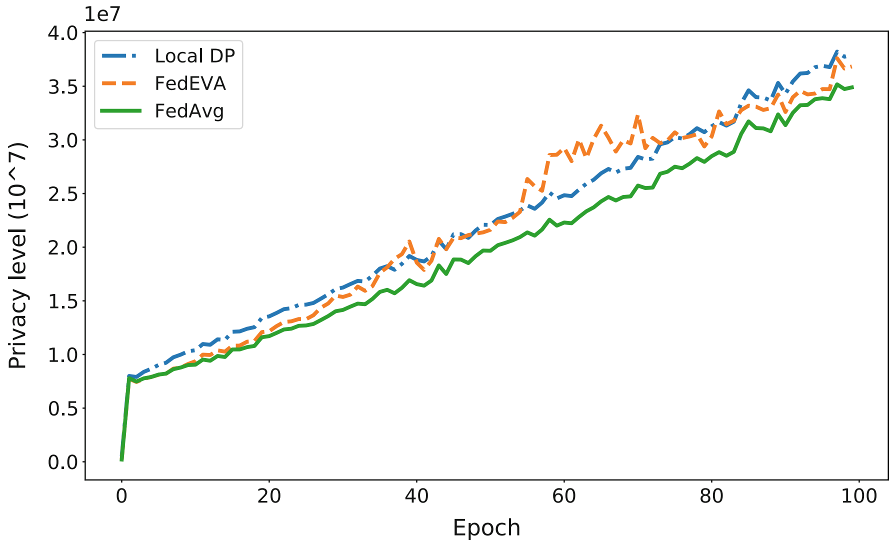

Fig. 6 Comparison of privacy level on the IID COCO dataset

effectiveness of our *FedEVA* scheme, which greatly outperforms the *FedAvg+LDP* scheme under our experimental settings.

Figure 6 compares the privacy level of *FedAvg*, *FedEVA*, and *FedAvg+LDP*. The x-axis represents the epoch number, while the y-axis represents the privacy level. Specifically, the privacy level of the central version of YOLO is zero, i.e., the privacy issue is not considered in the standard version. By comparison, our

FedEVA scheme outperforms the standard *FedAvg* scheme, which implies that the design in this chapter can achieve a better privacy preservation degree. Although the *FedAvg+DLP* scheme guarantees a stronger privacy protection, the EVA model training performance under the *FedAvg+DLP* scheme is invalid.

6.2.2 Non-IID Case

Figure 7 compares the mAP during training process with different FL models and central training on the non-IID dataset. The x-axis represents the epoch number, while the y-axis represents the mAP of the aggregated model in each epoch. Note that the mAP metric of *FedAvg+LDP* is 0 during 60 epochs. From this experiment, we can observe the mAP performance of *FedEVA* is still similar to that of standard *FedAvg*, though both *FedEVA* and *FedAvg* perform worse on the non-IID dataset than that on the IID dataset. This experiment indicates our *FedEVA* scheme is as robust as *FedAvg* and outperforms *FedAvg+LDP* under the non-IID case.

Figure 8 compares the privacy level of *FedAvg*, *FedEVA*, and *FedAvg+LDP* versus the training epoch. The x-axis represents the epoch number, while the y-axis represents the privacy level. As the privacy issue is not considered in the standard version, the privacy level of the central version of YOLO is zero in default. Again, although the *FedAvg+DLP* scheme guarantees a stronger privacy protection, the performance of EVA model training under the *FedAvg+DLP* scheme is invalid. By comparison, the *FedEVA* scheme achieves a similar privacy level as that in the standard *FedAvg* scheme, however, our *FedEVA* scheme achieves much stronger protection on clients' privacy.

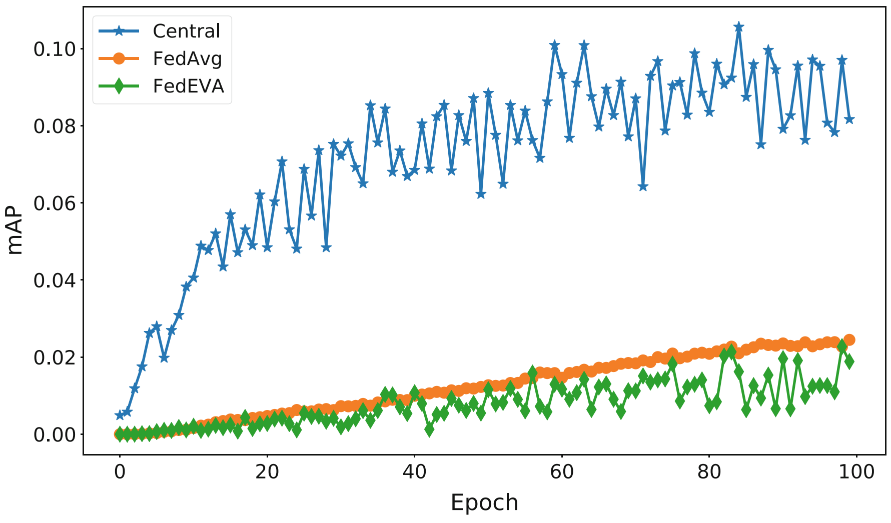

Fig. 7 Comparison of mAP on the non-IID COCO dataset

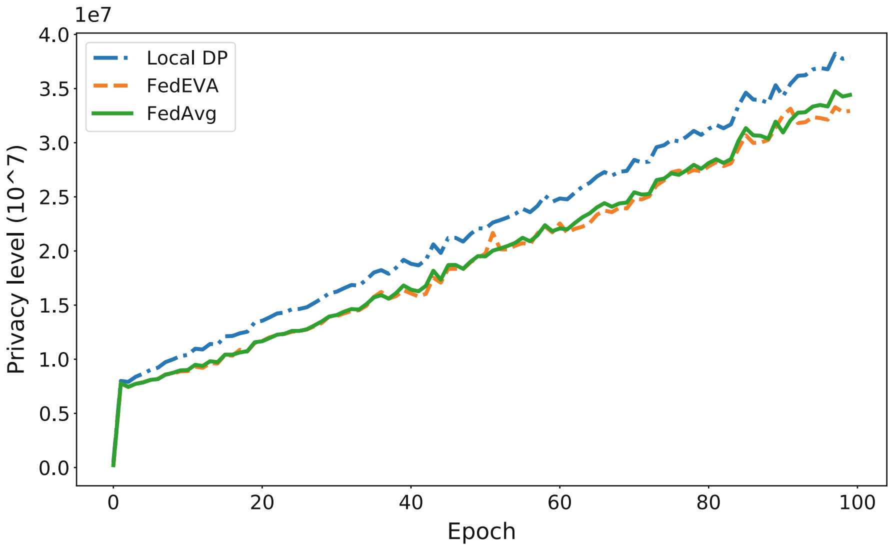

Fig. 8 Comparison of privacy level on the non-IID COCO dataset

7 Conclusions

This chapter proposed a federated learning driven privacy-preserving edge video analytics model training system, named as *FedEVA*. By locally perturbing the private rating, the model parameters can still guarantee the model training accuracy, while preventing eavesdroppers from tapping clients' video contents information. The key idea lies in that the design of the perturbation method will not change the model training and updating structure. Besides, we also verify with experiments that our *FedEVA* framework outperforms the standard *FedAvg* scheme in the degree of privacy preservation. The *FedEVA* framework is practical and efficient as the perturbation operation is a linear operation with low time and space complexity.

References

1. Abadi, M., Chu, A., Goodfellow, I., McMahan, H.B., Mironov, I., Talwar, K., Zhang, L.: Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 308–318. (2016)
2. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V.: How to backdoor federated learning. In: Cryptography and Security (2018). arXiv:1807.00459
3. Bilal, K., Erbad, A.: Edge computing for interactive media and video streaming. In: Proceedings of the International Conference on Fog and Mobile Edge Computing (FMEC), pp. 68–73 (2017)

4. Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H.B., Patel, S., Ramage, D., Segal, A., Seth, K.: Practical secure aggregation for privacy-preserving machine learning. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS), pp. 1175–1191 (2017)
5. Bringer, J., Chabanne, H., Favre, M., Patey, A., Schneider, T., Zohner, M.: GSHADE: Faster privacy-preserving distance computation and biometric identification. In: Proceedings of the ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec), pp. 187–198 (2014)
6. Chowdhery, A., Chiang, M.: Model predictive compression for drone video analytics. In: 2018 IEEE International Conference on Sensing, Communication and Networking (SECON Workshops), pp. 1–5. IEEE, Piscataway (2018)
7. Davies, N., Taft, N., Satyanarayanan, M., Clinch, S., Amos, B.: Privacy mediators: Helping IoT cross the chasm. In: Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications, pp. 39–44. ACM, New York (2016)
8. Dwork, C., Roth, A.: The Algorithmic Foundations of Differential Privacy. Now Foundations and Trends, Hanover (2014)
9. Fan, W., He, J., Guo, M., Li, P., Han, Z., Wang, R.: Privacy preserving classification on local differential privacy in data centers. J. Parall. Distrib. Comput. **135**, 70–82 (2020)
10. Geyer, R.C., Klein, T., Nabi, M.: Differentially private federated learning: A client level perspective. In: Proceedings of NIPS Workshop: Machine Learning on the Phone and other Consumer Devices, pp. 1–7 (2017)
11. Holohan, N., Leith, D.J., Mason, O.: Differential privacy in metric spaces: numerical, categorical and functional data under the one roof. Inf. Sci. **305**, 256–268 (2015)
12. Jana, S., Narayanan, A., Shmatikov, V.: A scanner darkly: Protecting user privacy from perceptual applications. In: 2013 IEEE Symposium on Security and Privacy, pp. 349–363. IEEE, Piscataway (2013)
13. Järvinen, K., Kiss, Á., Schneider, T., Tkachenko, O., Yang, Z.: Faster privacy-preserving location proximity schemes. In: Cryptology and Network Security, pp. 3–22. Springer, Berlin (2018)
14. Jiang, L., Xu, C., Wang, X., Luo, B., Wang, H.: Secure outsourcing SIFT: efficient and privacy-preserving image feature extraction in the encrypted domain. IEEE Trans. Depend. Secure Comput. **17**(1), 179–193 (2020)
15. Khazbak, Y., Qiu, J., Tan, T., Cao, G.: Targetfinder: privacy preserving target search through IoT cameras. In: Proceedings of the International Conference on Internet of Things Design and Implementation, pp. 213–224. ACM, New York (2019)
16. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2014). 1412.6980
17. Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: Challenges, methods, and future directions (2019). 1908.07873
18. Li, X., Li, J., Yiu, S., Gao, C., Xiong, J.: Privacy-preserving edge-assisted image retrieval and classification in IoT. Front. Comput. Sci. **13**(5), 1136–1147 (2019)
19. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European Conference on Computer Vision, pp. 740–755. Springer, Berlin (2014)
20. Liu, P., Qi, B., Banerjee, S.: Edgeeye: An edge service framework for real-time intelligent video analytics. In: Proceedings of the 1st International Workshop on Edge Systems, Analytics and Networking, pp. 1–6. ACM, New York (2018)
21. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, pp. 1273–1282 (2017)
22. Ran, X., Chen, H., Zhu, X., Liu, Z., Chen, J.: DeepDecision: A mobile deep learning framework for edge video analytics. In: Proceedings of the IEEE Conference on Computer Communications (INFOCOM), pp. 1–9 (2018)
23. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement (2018). Preprint arXiv:180402767

24. Sarikaya, Y., Ercetin, O.: Motivating workers in federated learning: A stackelberg game perspective. IEEE Netw. Lett. **2**(1), 23–27 (2020)
25. Tarasov, A.V., Savchenko, A.V.: Emotion recognition of a group of people in video analytics using deep off-the-shelf image embeddings. In: International Conference on Analysis of Images, Social Networks and Texts, pp. 191–198. Springer, Berlin (2018)
26. Truex, S., Baracaldo, N., Anwar, A., Steinke, T., Ludwig, H., Zhang, R., Zhou, Y.: A hybrid approach to privacy-preserving federated learning. In: Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security (AISec), pp. 1–11 (2019)
27. Uddin, M.A., Alam, A., Tu, N.A., Islam, M.S., Lee, Y.K.: Siat: a distributed video analytics framework for intelligent video surveillance. Symmetry **11**(7), 911 (2019)
28. Wang, J., Amos, B., Das, A., Pillai, P., Sadeh, N., Satyanarayanan, M.: A scalable and privacy-aware IoT service for live video analytics. In: Proceedings of the 8th ACM on Multimedia Systems Conference, pp. 38–49. ACM, New York (2017)
29. Wang, D., Peng, Y., Ma, X., Ding, W., Jiang, H., Chen, F., Liu, J.: Adaptive wireless video streaming based on edge computing: Opportunities and approaches. IEEE Trans. Serv. Comput. **12**, 1–12 (2018)
30. Wang, J., Amos, B., Das, A., Pillai, P., Sadeh, N., Satyanarayanan, M.: Enabling live video analytics with a scalable and privacy-aware framework. ACM Trans. Multimedia Comput. Commun. Appl. **14**(3), 64 (2018)
31. Wu, N., Farokhi, F., Smith, D., Kaafar, M.: The value of collaboration in convex machine learning with differential privacy. In: Proceedings of the IEEE Symposium on Security and Privacy (SP), Los Alamitos, pp. 485–498 (2020)
32. Yaseen, M.U., Anjum, A., Rana, O., Antonopoulos, N.: Deep learning hyper-parameter optimization for video analytics in clouds. IEEE Trans. Syst. Man Cyb. Syst. **49**(1), 253–264 (2018)
33. Zarepour, E., Hosseini, M., Kanhere, S.S., Sowmya, A.: A context-based privacy preserving framework for wearable visual lifeloggers. In: 2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops), pp. 1–4. IEEE, Piscataway (2016)
34. Zhang, T., Chowdhery, A., Bahl, P.V., Jamieson, K., Banerjee, S.: The design and implementation of a wireless video surveillance system. In: Proceedings of the 21st Annual International Conference on Mobile Computing and Networking, pp. 426–438. ACM, New York (2015)