EPC: a Video Analytics System with Efficient Edge-side Privacy Control

Bihai Zhang, Siping Shi, Dan Wang*
The Hong Kong Polytechnic University
Kowloon, Hong Kong
{csbzhang,cssshi,csdwang}@comp.polyu.edu.hk

Chuang Hu
Wuhan University
Wuhan, China
handc@whu.edu.cn

ABSTRACT

Edge-cloud video analytics systems capture video streams by edge cameras and send the video streams to the cloud for analytics to support applications like video surveillance, VR/AR, autonomous driving, etc. Video streams captured at the edge may contain sensitive objects, e.g., a human being. Existing studies propose adding noise to the intermediate video analytics results, encrypting video frames, etc. In this paper, we take an orthogonal approach where we remove, a.k.a. denaturing, the sensitive objects at the edge side before sending a video frame to the cloud.

The challenge is that edge devices are highly resource-limited, and the denaturing operation has non-trivial computation costs. More specifically, before denaturing, one needs to locate the sensitive objects by object detection; such object detection computation is resource intensive. In this paper, we propose EPC, an edge-cloud video analytics system that leverages a trajectory prediction model to locate sensitive objects in video frames. We formally analyze EPC and show that EPC can guarantee privacy. We evaluate EPC with two applications, person counting and vehicle detection. Evaluation results show that EPC can prevent privacy leakage under visual data attack with 95% video analytics accuracy and a 4x speedup compared to existing privacy control mechanisms.

CCS CONCEPTS

• **Security and privacy → Human and societal aspects of security and privacy**; **Privacy protections**;

KEYWORDS

Privacy control, Trajectory prediction, Video analytics, Edge computing

ACM Reference Format:
Bihai Zhang, Siping Shi, Dan Wang and Chuang Hu. 2022. EPC: a Video Analytics System with Efficient Edge-side Privacy Control. In *17th ACM Workshop on Mobility in the Evolving Internet Architecture (MobiArch'22), October 21, 2022, Sydney, NSW, Australia.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3556548.3559635

*Dan Wang is the corresponding author.

MobiArch'22, October 21, 2022, Sydney, NSW, Australia

ACM ISBN 978-1-4503-9518-2/22/10…$15.00
https://doi.org/10.1145/3556548.3559635

1 INTRODUCTION

Nowadays, edge-cloud video analytics systems have obtained increasing attention with broad applications in video surveillance, VR/AR, Metaverse and so forth. The edge devices capture the video streams and send them to the cloud to leverage the cloud's computing resources. An essential feature of edge-cloud video analytics systems is the distrust of the cloud from the perspective of the edge because the users usually have no complete control of their sensitive data after uploading them. For example, the ADT provides intrusion detection by its edge-cloud video analytics system. Their cloud servers were hacked in 2021, and videos uploaded by hundreds of ADT home security cameras were leaked. Consequently, users choose edge devices that differ from the cloud vendor, e.g., Apple Aqara G3, to maintain certain controllability.

Many privacy protection mechanisms have been proposed for edge-cloud video analytics systems [9, 11, 17]. One category of privacy mechanisms is adding noise to the intermediate results sent to the cloud so that attackers cannot reconstruct sensitive visual data [11]. Another category of mechanisms is to encrypt video frames and perform video analytics on encrypted frames [15]. In this paper, we take an orthogonal approach where we remove the sensitive objects at the edge before sending video frames to the cloud (denaturing). Such an approach poses nil requirement for the forthcoming video analytics operations. In comparison, adding noise usually requires a co-training of the noise model and the video analytics model, whereas an encryption mechanism is limited in the number of video analytics operations supported.

A straightforward denaturing mechanism first detects sensitive objects in each frame and denatures detected objects. Yet object detection is a DNN/CNN inference operation and thus computation-intensive. This poses a challenge to resource-constrained edge devices, in particular for applications, e.g., the aforementioned illegal intrusion detection, which requires detection in a series of frames.

In this paper, we develop EPC, a video analytics system with efficient edge-side privacy control. EPC leverages *trajectory prediction*, a common technology developed in computer vision to locate sensitive objects in a video frame. Such trajectory prediction is much more resource efficient. We design a memory-based trajectory prediction approach that uses a memory to embed history trajectories and to online search a trajectory for prediction. We argue that memory-based trajectory prediction is suitable since it requires less computing resources, and more importantly, we can

control privacy leakage and computation latency. More specifically, we can manage the memory size: a larger memory size will embed more history trajectories with a higher prediction accuracy and thus less privacy leakage, yet larger memory size will also lead to greater computing cost, and vice versa for smaller memory size. We formally analyze EPC and show that EPC can guarantee privacy leakage to a threshold.

EPC supports applications with video analytics of a series of frames and is more efficient as compared to mechanisms based on object tracking. We evaluate EPC with two applications, person counting and vehicle detection under privacy leakage, system latency, and video analytics accuracy. Evaluation results show that EPC can prevent privacy leakage under visual data attack with 95% video analytics accuracy and a 4x speedup compared to existing privacy control mechanisms. The contributions of our paper are:

- We study edge-cloud video analytics systems with privacy concerns from the perspective of implementing edge-side privacy control.
- We develop a trajectory prediction-based system, EPC (§2) for resource-constrained edge devices. We design a memory-based trajectory prediction approach with analyzable privacy leakage control (§3).
- We present a systematic evaluation in two applications, person counting and vehicle detection (§4).

2 EPC DESIGN

2.1 Background

Edge-cloud video analytics system typically works in the following manner: an edge device (e.g., a camera enhanced with an edge box) captures video streams and performs preprocessing operations, e.g., background subtraction; then, the video frames will be sent to the cloud for video analytics using a pre-trained DNN/CNN model. Particularly, when the computation resources of the edge are available, it can optionally perform (usually partial) video analytics using the same DNN/CNN model as the cloud.

Trajectory prediction has been extensively explored in the computer vision community [12]. A *trajectory* is a sequence of positions of an object [1]. *Trajectory prediction* is the task of predicting future trajectories of the target object based on history trajectories [13]. DNN-based methods [1, 10, 13, 18] have shown good performance, and existing methods are divided into two categories: parameter-based methods [1, 13] and memory-based methods [10, 18]. The parameter-based methods, e.g., Social-LSTM [1] and SGCN [13], train a DNN model directly for trajectory prediction, whereas the memory-based methods, e.g., MANTRA [10], trains DNN models that can construct a *memory* which stores the embedding of the history trajectories. Intuitively, a trajectory will be extracted from this memory when conducting trajectory prediction. Note that the memory is substantially smaller than the DNN model itself and intuitively, trajectory prediction is basically a search of the trajectories from this memory. Parameter-based methods have high computation costs and high accuracy, and memory-based methods have low computation costs. For resource-constrained edge devices, memory-based methods are more viable, and we adopt one typical memory-based method in this paper.

Memory-based trajectory prediction. We now present a representative memory-based method, MANTRA. It has two phases, which are trajectory learning and prediction.

Trajectory learning: There are three essential components 1) a memory to store the embeddings of history trajectories, 2) encoder/decoder DNN models to build the memory, and 3) a memory controller to balance memory size and prediction accuracy. Intuitively, to achieve greater accuracy, one can construct a larger memory by embedding more history trajectories. Specifically, the memory controller is a DNN model, which takes a trajectory as input and outputs whether this trajectory should be embedded into the memory or not.

Trajectory learning performs as follows. Given history trajectories, it trains an encoder DNN model and a decoder DNN model. Then, given the desired accuracy, the encoder DNN model uses history trajectories to construct the memory by embedding the trajectories released by the memory controller. Note that a larger memory size will lead to a greater computation cost in trajectory prediction.

Trajectory prediction: given the history trajectory of an object, it uses the decoder DNN model to predict the future trajectory of this object by searching the memory. Note that larger memory size will lead to a greater computing cost.

2.2 The EPC Framework

In this paper, we develop EPC, a real-time video analytics system that protects inference privacy by restricting sensitive objects from being leaked from the edge side.

Threat model. We assume that the edge is trusted and the cloud is untrusted. The potential attacker conducts visual data attack [19] on video frames. In such an attack, the attacker can analyze the acquired video frames by a pre-trained DNN/CNN model (e.g., an object detection model for a specific sensitive object) and recognize the sensitive object.

EPC modular design. As shown in Figure 1, EPC has a privacy control module to restrict the sensitive objects from leaking at the edge side and a video analytics module to undertake various video analytics tasks. For the privacy control module, there are three key components: 1) a video stream segmentation module to partition the input video stream into groups of pictures (GOPs) by Algorithm 1, and 2) a trajectory-assisted target localization module to efficiently acquire the positions of sensitive objects in each frame, in which Algorithm 2 is executed, and 3) an object denaturing module to modify the sensitive objects based on their location in each frame. For the video analytics module, we fed the denatured video stream into the video analytics DNN model and obtain the corresponding analytics results.

Video analytics in EPC. As shown in Figure 1, the video frames will first be divided into GOPs by the video stream segmentation module. Next, the sensitive objects are located by the trajectory-assisted target localization module. Finally, the frame denaturing module modifies the sensitive objects in each frame based on the received locations of them and sends the modified video stream to the video analytics module, e.g., a YOLOv3 model for people counting.

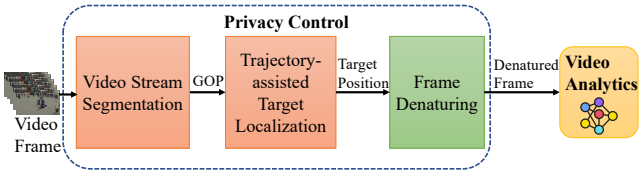


Figure 1: The modular architecture of EPC

2.3 Trajectory Learning and Prediction

There are two phases for utilizing trajectory prediction in EPC. The first phase is *trajectory learning*, where a memory-based trajectory prediction model, MANTRA, is trained offline. The second phase is *trajectory prediction*, given the edge-side resource constraints, applying object detection and MANTRA on a video stream to obtain the sensitive objects' positions in each frame with minimum privacy leakage. In this subsection, we will first introduce the training of MANTRA model, and then present how MANTRA serves in trajectory prediction-assisted target localization.

2.3.1 Trajectory learning. **Memory** acts as a database to store encodings of history trajectories in a key-value form. Let M be a memory that contains $|M|$ entries of observation-future encodings, where each entry $m_i = (\pi_i, \psi_i)$. π_i is the key and ψ_i is the value.

Encoder/decoder models aim to generate the encodings stored in the memory. Let $\Pi(\cdot)$ be the encoder of observation positions, and $\Phi(\cdot)$ be the encoder of future positions. We define $\Psi(\cdot)$ as the decoder to decode the newly observed positions. Given a sample trajectory $\mathbf{X}_i = [\mathbf{X}_O^i, \mathbf{X}_F^i]$, supposing the cosine similarity score between the encoding $\Pi(\mathbf{X}_O^i)$ and $\pi_j \in M$ is the highest, then we have: $\hat{\mathbf{X}}_F^i = \Psi(\Pi(\mathbf{X}_O^i), \phi_j)$. $\Pi(\cdot)$, $\Phi(\cdot)$ and $\Psi(\cdot)$ are implemented by Gated Recurrent Unit (GRU), and we jointly train them as an autoencoder.

Memory controller ensures prediction accuracy while maintaining a compact memory. It emits a probability P of writing a new embedding into the memory. The memory controller is trained by minimizing the following loss L_c:

$$L_c = e \times (1 - P) + (1 - e) \times P, \tag{1}$$

where e is the trajectory prediction error.

$$e = 1 - \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}_i(\hat{\mathbf{X}}_F, \mathbf{X}_F), \tag{2}$$

where

$$\mathbf{1}_i(\hat{\mathbf{X}}_F, \mathbf{X}_F) = \begin{cases} 1, & d_E((\hat{x}_i, \hat{y}_i), (x_i, y_i)) \le th_i, \\ 0, & d_E((\hat{x}_i, \hat{y}_i), (x_i, y_i)) > th_i, \end{cases} \tag{3}$$

and $d_E((\hat{x}_i, \hat{y}_i), (x_i, y_i))$ is the Euclidean distance between the i-th point of $\hat{\mathbf{X}}_F$ and $\mathbf{X}_F$, and th_i is a threshold.

Prediction error and memory size. At the beginning of training, e is close to one, indicating that the memory is too small to reconstruct future trajectory accurately. In this case, $L_c \approx 1 - P$, so the controller maximizes P to store more embeddings to reduce prediction error and the memory size increases consequently. As prediction accuracy and memory size increase, e gradually drops to zero, $L_c \approx P$, indicating that P is minimized. Since the memory controller loss is minimized in training, we can ensure that the memory achieves low prediction error with minimal memory size. Note that given different requirements of prediction error and memory size, we can construct the memory of different sizes.

Algorithm 1: SSeg

Input: Predefined threshold τ. Video Stream S of length T_s.
Output: Group of Pictures G

```
1 for t = 1 to T_s do
2     f = rgb2gray(downsampling(f_t))
3     diff_A(f, f_ref) = Difference(f, f_ref)
4     if diff_A(f, f_ref) ≤ τ then
5         G.append(f_t)
6     else
7         Output G
8         G = f_t, f_ref = f
```

2.3.2 Trajectory prediction. Due to resource constraints, we must decide whether the sensitive objects in video frames are located by object detection or trajectory prediction to minimize privacy leakage under latency constraints. We first introduce video stream segmentation. Then, we propose the privacy leakage and latency constraint. Finally, we formulate the trajectory prediction frame scheduling problem and propose an algorithm to solve it.

Video stream segmentation. A trajectory prediction model cannot discover a new object. So, we may discover an object after it has existed for a period, leading to privacy leakage. In video streams, the adjacent frames are similar in terms of content. If a new object emerges, a high frame difference will be detected. Thus, we propose an algorithm called SSeg to divide a video stream into a set of GOPs, where the inter-frame difference is under a threshold. We propose to use *Area* metric [8] because it is sensitive to the arrival of a new object but not the motion of existing objects.

SSeg is summarized in Algorithm 1. We first downsample a new frame to 320×180 and convert it to grayscale (line 2). Next, we compute the difference between it and the reference frame (line 4). If the difference is within a threshold, the new frame is appended to the current GOP. Otherwise, it is set as the reference frame, and a new GOP starts from it (line 5-9).

Latency of trajectory prediction-assisted denaturing. We formulate the time t_j to denature frame f_j below:

$$t_j = x_j \cdot t_{det}^j + (1 - x_j) \cdot t_{pre}^j + t_{de}^j \tag{4}$$

where $x_j \in \{0, 1\}$. x_j equals to one when object detection and recognition execute on f_j. Otherwise, x_j equals to zero. t_{det}^j, t_{pre}^j and t_{de}^j are the object detection and recognition latency, the trajectory prediction latency and the denaturing latency respectively. Suppose the upper bound of latency for a given GOP of length T is βT, we have:

$$T_{de} = \sum_{i=1}^{T} t_i \le \beta T \tag{5}$$

Privacy leakage of trajectory prediction-assisted denaturing. Privacy refers to the private information encoded in the pixels covering a sensitive object O_k. Each pixel p_i carries part of the privacy.

Algorithm 2: TPFSche

Input: $\theta_{tra}, G, \beta T$
Output: $\mathbf{x}$: Scheduling policy of GOP G

$\mathbf{x} = [1, 1, 0, 0, \ldots, 0]$
for $t = 3$ *to* T **do**
  $L = L_G(\mathbf{x}, \theta_{tra})$
  $x_t = 1$
  if $L_G(\mathbf{x}, \theta_{tra}) \leq L$ *and* $T_{de} \leq \beta T$ **then**
    $L = L_G(\mathbf{x}, \theta_{tra})$
  else
    $x_t = 0$
    Output $\mathbf{x}$

Thus, we propose to define privacy leakage as the remaining private information in the denatured object O_k^d. The privacy leakage L_O of O_k^d is the summation of undenatured pixels' CAM values, i.e.,

$$L_O(O_k^d) = \sum_U CAM(p_i) \tag{6}$$

where U is the undenatured region of O_k^d, and $CAM()$ is the Class Activation Map (CAM) [20] for images.

Problem formulation. Given a GOP G of length T, latency constraint βT, we need to determine the scheduling policy $\mathbf{x}$ to minimize the privacy leakage under Constraint (5). We formulate the **Trajectory Prediction Frame Scheduling Problem** as below:

$$\min_{\mathbf{x}} \sum_{t=1}^{T} (1 - x_t) \sum_{k=1}^{K_t} L_O(O_k^d) \tag{7}$$

$$\begin{aligned} s.t. \quad & \sum_{i=1}^{T} t_i \leq \beta T, \sum_{t=1}^{T} x_t = s, \sum_{t=1}^{s} x_t = s, \forall s = 2, \ldots, T \\ & x_t \in \{0, 1\}, \quad \forall t = 1, \ldots, T. \end{aligned} \tag{8}$$

To solve this problem, we propose a greedy algorithm shown in Algorithm 2. The main idea is to search all viable solutions and select the solution which minimizes privacy leakage. Since the searching space is $O(T)$ and we can finish each search in $O(1)$ time, this search terminates in $O(T)$ time.

3 PRIVACY ANALYSIS

This section provides a formal analysis of the privacy leakage in EPC. We first formally model the privacy leakage. Then, we analyze how our design of EPC ensures that the privacy leakage in EPC can be bounded by a threshold.

Privacy leakage. In EPC, privacy leakage mainly depends on the accuracy of locating the regions for denaturing. The localization error comes from the trajectory prediction error. We use the same metric defined in Equation (6) to measure the privacy leakage of a denatured sensitive object O_k^d. Since trajectory prediction error contributes to the localization error and corresponding undenatured region, we define the privacy leakage L_{EPC} of a denatured sensitive object O_k^d in frame f_t as follows:

$$L_{EPC}(O_k^d) = (1 - x_t) \sum_U CAM(p_i) \tag{9}$$

where U is the undenatured region due to error in trajectory prediction, and x_t is the decision variable in scheduling policy $\mathbf{x}$ corresponding to the frame which contains O_k^d.

Given a GOP G of length T and the privacy definition in Equation (9), we can compute the privacy leakage of G as follows:

$$L_{EPC}(G) = \sum_{t=1}^{T} \sum_{k=1}^{K_t} L_{EPC}(O_k^d) \tag{10}$$

Inserting Equation (9) into Equation (10), we can rewrite the privacy leakage of GOP G as:

$$L_{EPC}(G) = \sum_{t=1}^{T} \sum_{k=1}^{K_t} (1 - x_t) \sum_U CAM(p_i) \tag{11}$$

Theorem 1. *If $L_c \leq 0.5$ & $P < 0.5$ or $L_c > 0.5$ & $P > 0.5$, then $e \leq L_c$.*

Proof. Based on Equation (1), we can get $e = \frac{L_c - P}{1 - 2P} (P \neq 0.5)$. If $L_c \leq 0.5$, then we have $(1 - 2P)L_c \geq L_c - P$. If $P < 0.5$, then we can get $e = \frac{L_c - P}{1 - 2P} \leq L_c$. Similarly, we can prove $e \leq L_c$ when $L_c > 0.5$ & $P > 0.5$. □

Theorem 2. *Given the constraint $e \leq \alpha$, for a GOP G of length T, we have:*

$$L_{EPC}(G) \leq \sum_{p=1}^{N_{tra}} \Big(\sum_{i=1}^{N(1-\alpha)} \epsilon_i^p + \sum_{i=1}^{N\alpha} \mu_i^p \Big) \tag{12}$$

where N_{tra} is number of trajectories and $N = T - \sum_{t=1}^{T} x_t$.

Proof. Theorem 1 ensures that we can bound the trajectory prediction error e by minimizing controller loss L_c. If the training process ensures $e \leq \alpha$, then, for a prediction $\hat{\mathbf{X}}_F^i$ of length N, according to Equation (2) and (3), there are at least $N(1-\alpha)$ objects at position $(\hat{x}_i, \hat{y}_i)$ satisfying the condition that $d_E((\hat{x}_i, \hat{y}_i), (x_i, y_i)) \leq th_i$. Thus, there are at least $N(1 - \alpha)$ denatured objects satisfying:

$$\sum_U CAM(p_i) \leq \epsilon_i^p, \tag{13}$$

where $\epsilon_i^p = \max_U \sum_U CAM(p_i)$ and the undenatured region U satisfies the condition $d_E((\hat{x}_i, \hat{y}_i), (x_i, y_i)) \leq th_i$. For the other at most $N\alpha$ objects, their privacy leakage cannot exceed all the private information they contains, so these at most $N\alpha$ objects satisfies:

$$\sum_U CAM(p_i) \leq \mu_i^p, \tag{14}$$

where $\mu_i^p = \sum_{R^i} CAM(p_i)$ and R_i is the region covered by corresponding sensitive object. Therefore, for each prediction $\hat{\mathbf{X}}_F^p$, its privacy leakage cannot exceed $\sum_{i=1}^{N(1-\alpha)} \epsilon_i^p + \sum_{i=1}^{N\alpha} \mu_i^p$. Finally, we can generalize it to all predictions of a GOP to get Equation (12). Thus, we can prove Theorem 2. □

Theorem 2 ensures a privacy leakage guarantee for a GOP. Since a video stream consists of GOPs, it can easily generalize to the conclusion that EPC can bound privacy leakage.

4 EVALUATION

4.1 Experimental Setup

We use a desktop with an Intel i9 CPU and an Nvidia RTX 3090 GPU to simulate an edge box. An AWS DeepLens camera functions as the IP-camera connected to an edge box.

Applications and datasets. We use two typical applications to evaluate EPC: 1)*Vehicle Detection (VD)* marks the bounding boxes of vehicles in video streams of nuScenes dataset [2] with YOLOv5s model, 2)*Person Counting (PC)* counts the number of persons in each frame of the MOT15 dataset [7] with YOLO5Face model.

Baselines. We compare EPC with two baselines: 1) *Naive Video Denaturing (NVD)* [14] denatures objects for each frame in a detect-modify manner, 2) *Amadeus* [4] denatures the sensitive objects based on the locations which are obtained by object detection and KCF tracking. Particularly, for fair comparison with *Amadeus*, we test EPC on various frequencies of object detection, e.g., 1:3 indicates object detection is performed on one-fourth of all frames.

Evaluation Metrics. We use three metrics to compare the performance of EPC with the baselines: 1) *Defense Accuracy* is used to show the privacy protection performance and calculated as the proportion of frames that are protected w.r.t all frames. Specifically, the protected frame refers to from which the visual data attacker can not recognize a sensitive object, 2) *Latency per Frame* represents the time spent in converting a raw frame to a denatured frame, 3) *Analytics Accuracy* shows the utility of the denatured video frame and is computed by the accuracy metric of each application.

4.2 Experiment Results

4.2.1 Performance in Privacy Protection. Figure 2(a) depicts defense accuracy of EPC, Amadeus and NVD. Overall, EPC outperforms Amadeus on both applications. Given the same fraction of object detection, EPC towers over Amadeus in 19.05%-23.49% and 3.55%-5.69% on VD application and PC application, respectively. Compared to NVD, EPC can reach even 96.40% of its accuracy, which indicates a very small performance gap. We need to note that EPC executes under the resource constraints which hardly support real-time execution of NVD. Therefore, Figure 2(a) validates the effectiveness of integrating trajectory prediction into privacy control for video analytics systems.

4.2.2 Improvement on System Latency. According to Figure 2(b), in VD application, NVD shows a latency of 21.0ms. In PC application, NVD takes 48ms to denature a frame, i.e., the frame rate is 20.8fps, but a real-time system usually requires frame rate to be 30fps, presenting that NVD is not suitable for resource-constrained edge devices. Compared to NVD, EPC significantly outperforms it in latency metric. For EPC-1:1, its latency is only 51.81% and 50.83% of NVD on VD and PC applications, respectively. EPC-1:4 even achieves up to 78.92% latency reduction on PC application. EPC also demonstrates its advantage over Amadeus. The latency of EPC-1:1 and EPC-1:4 are 92.95% and 76.67% of Amadeus-1:1 and Amadeus-1:4 on PC application, respectively. The ratios drop to 78.56% and 49.27% on VD application. The main reason is that trajectory prediction is an order of magnitude faster than object detection. And trajectory prediction requires only history trajectories as input, instead of complex visual inputs, e.g., video frames.

4.2.3 Performance in Analytics Accuracy. Figure 2(c) displays the analytics accuracy of VD and PC applications. As shown in Figure 2(c), running video analytics with frames denatured by EPC is only slightly affected. For VD application, the analytics accuracy of EPC is 97.65%-98.75%. For PC application, the result is 98.17%-98.73%. The analytics accuracy of NVD is 99.2% for VD application and 99.3% for PC application. EPC's performance is very close to NVD's performance, but EPC executes much faster than NVD. EPC also demonstrates its advantage over Amadeus in analytics accuracy. Different from protection accuracy, analytics accuracy varies a little under different latency constraints. We argue that this is because mis-tracking or mis-predicting an object's position will directly lead to failure of privacy protection, while video analytics is not very sensitive to such error.

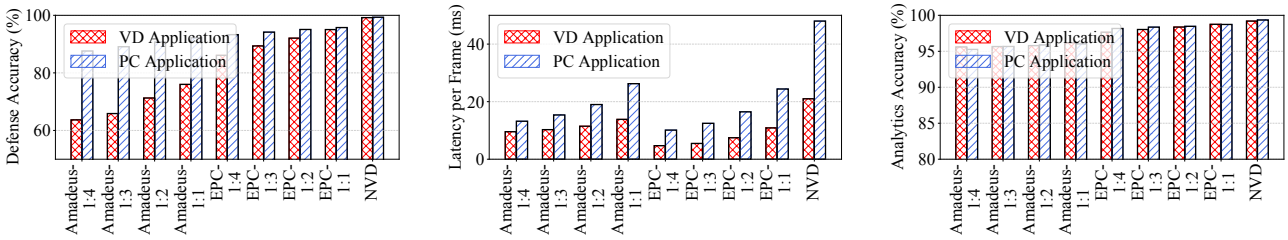
4.2.4 Analysis of System Latency. Figure 3 presents the decomposition of latency for Amadeus and EPC. As we can see from Figure 3, the latency for each case is decomposed into time for target localization and time for tracking/trajectory prediction. Obviously, target localization dominates the latency for both Amadeus and EPC. For Amadeus, the fraction of target localization is 43.93%-75.81% on VD application and 72.73%-91.43% on PC application. For EPC, the fraction of target localization is 89.17%-96.15% on VD application and 94.86%-98.36%. Thus, on both VD and PC applications, EPC's fraction of target localization in latency is higher than Amadeus. Therefore, apart from target localization, EPC requires less time to locate a sensitive object, explaining why EPC can outperform Amadeus in latency metric. Besides, for Amadeus, as the fraction of object localization drops, the benefit due to KCF tracking is gradually encroached by its cost. But for EPC, the fraction of trajectory prediction in latency constantly stays at a low level.

5 RELATED WORK

Edge-cloud video analytics systems have been widely used nowadays. Our work falls into the edge-cloud video analytics systems with trusted edge and untrusted cloud. One major challenge is that edge devices are resource-constrained. This leads to research studies on mechanisms on hardware acceleration [3], frame filtering [8], etc. It is also typical that video analytics systems periodically adjust the resolution of a group of future frames given dynamic resource constraints. Our EPC, with trajectory prediction, can predetermine future resources to be consumed by itself.

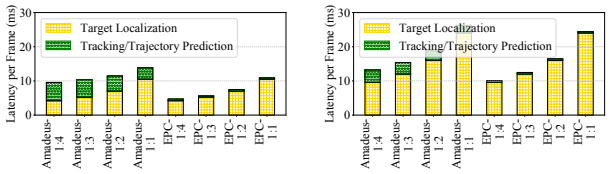
Trajectory prediction methods can be classified according to the way they describe target motion and formulate the causes [12]. There are physics-based methods [5], planning-based methods [6], and pattern-based methods [1]. Our trajectory prediction method falls into pattern-based methods. Pattern-based methods are usually supported by DNN models and can be further classified into parameter- and memory-based methods. EPC is based on the memory-based trajectory prediction method for its resource efficiency and analyzable privacy control.

Privacy-preserving edge-cloud video analytics systems have been developed due to visual data attacks [19] and privacy concerns. Existing privacy control mechanisms include adding noise [11], image transformation [9, 17], encryption [15], etc. Our work falls into the denaturing [4, 14, 16]. Denaturing is suitable for broad applications since it only removes sensitive objects and requires no modification of video analytics model. There exist denaturing



(a) Comparison in defense accuracy (b) Comparison in latency per frame (c) Comparison in analytics accuracy

Figure 2: The overall performance of EPC and baselines



(a) VD Application (b) PC Application

Figure 3: Decomposition of latency per frame for Amadeus and EPC

mechanisms [4, 16] based on object tracking. We show in our experiments that EPC outperforms these mechanisms. Yet a more important advantage is that object tracking-based mechanisms perform denaturing on the current frame. Consequently, the system may not know how much resource is available for future video analytics and cannot prepare adjustments.

6 CONCLUSION

This paper presents EPC, a new edge-cloud video analytics system with efficient edge-side privacy control. EPC can protect the privacy of video streams from visual data attacks by locating and denaturing sensitive objects before sending them to the cloud. To fit resource-constrained edge devices, EPC adopts a lightweight trajectory prediction method to efficiently locate sensitive objects in frames. Theoretical analysis and experimental results show EPC can achieve valid privacy control with negligible impact on utility.

ACKNOWLEDGMENTS

Dan Wang's work is supported by GRF 15210119, 15209220, 15200321, 15201322, ITF-ITSP ITS/070/19FP, CRF C5018-20G.

REFERENCES

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. 2016. Social LSTM: Human Trajectory Prediction in Crowded Spaces. In *Proc. of IEEE/CVF CVPR'16.* Las Vegas, NV, USA.

[2] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, et al. 2020. nuScenes: A Multimodal Dataset for Autonomous Driving. In *Proc. of IEEE/CFV CVPR'20.* Virtual Event.

[3] Stephen Cass. 2019. Taking AI to the edge: Google's TPU now comes in a maker-friendly package. *IEEE Spectrum* 56, 5 (2019), 16–17.

[4] Sandeep D'souza, Victor Bahl, Lixiang Ao, and Landon P. Cox. 2020. Amadeus: Scalable, Privacy-Preserving Live Video Analytics. *arXiv:2011.05163* (2020).

[5] A. Elnagar. 2001. Prediction of moving objects in dynamic environments using Kalman filters. In *Proc. of CIRA'01.* Banff, Canada.

[6] Junru Gu, Chen Sun, and Hang Zhao. 2021. DenseTNT: End-to-end Trajectory Prediction from Dense Goal Sets. In *Proc. of IEEE/CVF ICCV'21.* Virtual Event.

[7] Laura Leal-Taixé, Anton Milan, Ian Reid, Stefan Roth, and Konrad Schindler. 2015. MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking. *arXiv:1504.01942* (2015).

[8] Yuanqi Li, Arthi Padmanabhan, Pengzhan Zhao, Yufei Wang, Guoqing Harry Xu, et al. 2020. Reducto: On-Camera Filtering for Resource-Efficient Real-Time Video Analytics. In *Proc. of ACM SIGCOMM'20.* Virtual Event.

[9] Rui Lu, Siping Shi, Dan Wang, Chuang Hu, and Bihai Zhang. 2022. Preva: Protecting Inference Privacy through Policy-based Video-frame Transformation. In *Proc. of ACM/IEEE SEC'22.* Seattle, WA, USA.

[10] Francesco Marchetti, Federico Becattini, Lorenzo Seidenari, and Alberto Del Bimbo. 2020. MANTRA: Memory Augmented Networks for Multiple Trajectory Prediction. In *Proc. of IEEE/CFV CVPR'20.* Virtual Event.

[11] Fatemehsadat Mireshghallah, Mohammadkazem Taram, Prakash Ramrakhyani, et al. 2020. Shredder: Learning Noise Distributions to Protect Inference Privacy. In *Proc. of ACM ASPLOS'20.* Lausanne, Switzerland.

[12] Andrey Rudenko, Luigi Palmieri, et al. 2020. Human Motion Trajectory Prediction: A Survey. *IJRR* 39, 8 (2020), 895–935.

[13] Liushuai Shi, Le Wang, Chengjiang Long, Sanping Zhou, Mo Zhou, Zhenxing Niu, et al. 2021. SGCN: Sparse Graph Convolution Network for Pedestrian Trajectory Prediction. In *Proc. of IEEE/CVF CVPR'21.* Virtual Event.

[14] Pieter Simoens, Yu Xiao, Padmanabhan Pillai, Zhuo Chen, Kiryong Ha, and Mahadev Satyanarayanan. 2013. Scalable Crowd-sourcing of Video from Mobile Devices. In *Proc. of ACM MobiSys'13.* Taipei, Taiwan.

[15] Kimia Tajik, Akshith Gunasekaran, Rhea Dutta, et al. 2019. Balancing Image Privacy and Usability with Thumbnail-Preserving Encryption. In *Proc. of NDSS Symposium'19.* San Diego, CA, USA.

[16] Junjue Wang, Brandon Amos, Anupam Das, Padmanabhan Pillai, Norman Sadeh, et al. 2017. A scalable and privacy-aware IoT service for live video analytics. In *Proc. of ACM MMSys'17.* Taipei, Taiwan.

[17] Hao Wu, Xuejin Tian, et al. 2021. PECAM: Privacy-Enhanced Video Streaming and Analytics via Securely-Reversible Transformation. In *Proc. of ACM MobiCom'21.* New Orleans, LA, USA.

[18] Chenxin Xu, Weibo Mao, Wenjun Zhang, and Siheng Chen. 2022. Remember Intentions: Retrospective-Memory-based Trajectory Prediction. In *Proc. of IEEE/CFV CVPR'22.* New Orleans, LA, USA.

[19] Guangsheng Zhang, Bo Liu, Tianqing Zhu, Andi Zhou, and Wanlei Zhou. 2022. Visual privacy attacks and defenses in deep learning: a survey. *Artificial Intelligence Review* 55, 6 (01 Aug 2022), 4347–4401.

[20] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. 2016. Learning Deep Features for Discriminative Localization. In *Proc. of IEEE/CFV CVPR'16.* Las Vegas, NV, USA.