FEVA: A Federated Video Analytics Architecture for Networked Smart Cameras

Chuang Hu, Rui Lu, and Dan Wang

Abstract

Video analytics using networked smart cameras has become a core function for many applications including surveillance, object detection, AR/VR, and so on. In recent years, a number of architectures have been proposed to organize the computing and networking resources of cloud and edge cameras to collectively complete an analytics task (e.g., 3D reconstruction, multi-view re-identification). Unfortunately, in many applications, image sharing can lead to privacy concerns. One example is the high definition map (HD map) for autonomous driving. An HD map has a highly dynamic layer of real-time objects. Vehicles can collectively contribute videos from their onboard cameras to construct such a layer, but the video images can contain private information (e.g., the license plate numbers of front cars).

In this article, we propose FEVA, a new federated video analytics architecture. Intrinsically, FEVA keeps the video image data local to the edge for analytics and transmits the analytics results to the cloud for aggregation. FEVA partitions the video analytics computing tasks in a way that is privacy-preserving and maximizes the overall analytics accuracy under the computing and communication resource constraints of the edge devices. We show how FEVA can be used in practice by a case study using FEVA to support a video analytics application on multi-view vehicles 3D reconstruction. We implement FEVA by extending the open source platform TensorFlow Federated from Google. We deploy our case in an environment with four Amazon DeepLens cameras. Our evaluation shows that FEVA can protect privacy while effectively increasing the accuracy of the video analytics application.

Introduction

Recently, we have witnessed the accuracy of machine learning models in computer vision improving to become useful for real-world applications. In these applications, a deep neural network (DNN) or a convolutional neural network (CNN) model is trained for a certain video analytics task (e.g., face recognition). Then video analytics applies the pre-trained model to analyze video images for high-accuracy analytics functions.

Real-world applications have diverse requirements and resource constraints. For example, many applications require multiple cameras to collaboratively complete a video analytics task to solve the problems on limited view scopes, image missing and errors, low-resolution videos, and so on. Typical examples include 3D reconstruction [1], multi-view object re-identification [2], and others. There are also constraints on computing and communication resources. For example, edge devices (e.g., cameras) are resource-limited. Several architectures have been proposed to manage resources and support application requirements, ranging from edge-cloud video analytics [3] to collaborative video analytics [4].

Unfortunately, in many applications, image sharing can lead to privacy concerns. One example is the high-definition map (HD map) developed for autonomous driving [4]. The HD map is a map overlaid with various information such as traffic conditions and access ways with high precision at the centimeter level and updated frequently. The HD map is a key for mobility as a service, the advanced driver assistance system (ADAS), and autonomous driving. Constructing the HD map is a multi-party effort. The onboard cameras in vehicles are essential to video sources for HD map construction and updates. Nevertheless, video images can contain an HD map with irrelevant but privacy-sensitive data, such as license plate numbers.

In this article, we propose FEVA, a federated video analytics architecture. FEVA is partially inspired by the federated analytics (FA) framework proposed by Google in May 2020 [6]. FA is a new evolution following the federated learning (FL) framework. In the FA framework, individual clients collectively carry out a non-training analytic task rather than training a model in FL, and send derived insights, not weight updates in FL, to a coordinating server. Although the newly introduced FA still follows the federation paradigm, the central aggregation part and local analytics part in FA call for careful designs in specific applications.

FEVA is designed to support a specific class of applications on collaborative video analytics with privacy concerns. FEVA assumes that a DNN/CNN model has been trained. A video analytic task such as 3D reconstruction needs video images from multiple cameras; however, these cameras have privacy concerns.

We carefully study the video analytic computing task workflow and the diverse resource constraints at the edge. In our design, FEVA keeps video image data local to the edge devices, and FEVA partitions the video analytics computing workflow in a privacy-preserving way. FEVA also maximizes the video analytics accuracy with consideration of the computing and communication resources at the edge devices.

Chuang Hu, Rui Lu, and Dan Wang are with Hong Kong Polytechnic University.

Digital Object Identifier:
10.1109/MNET.001.2100261

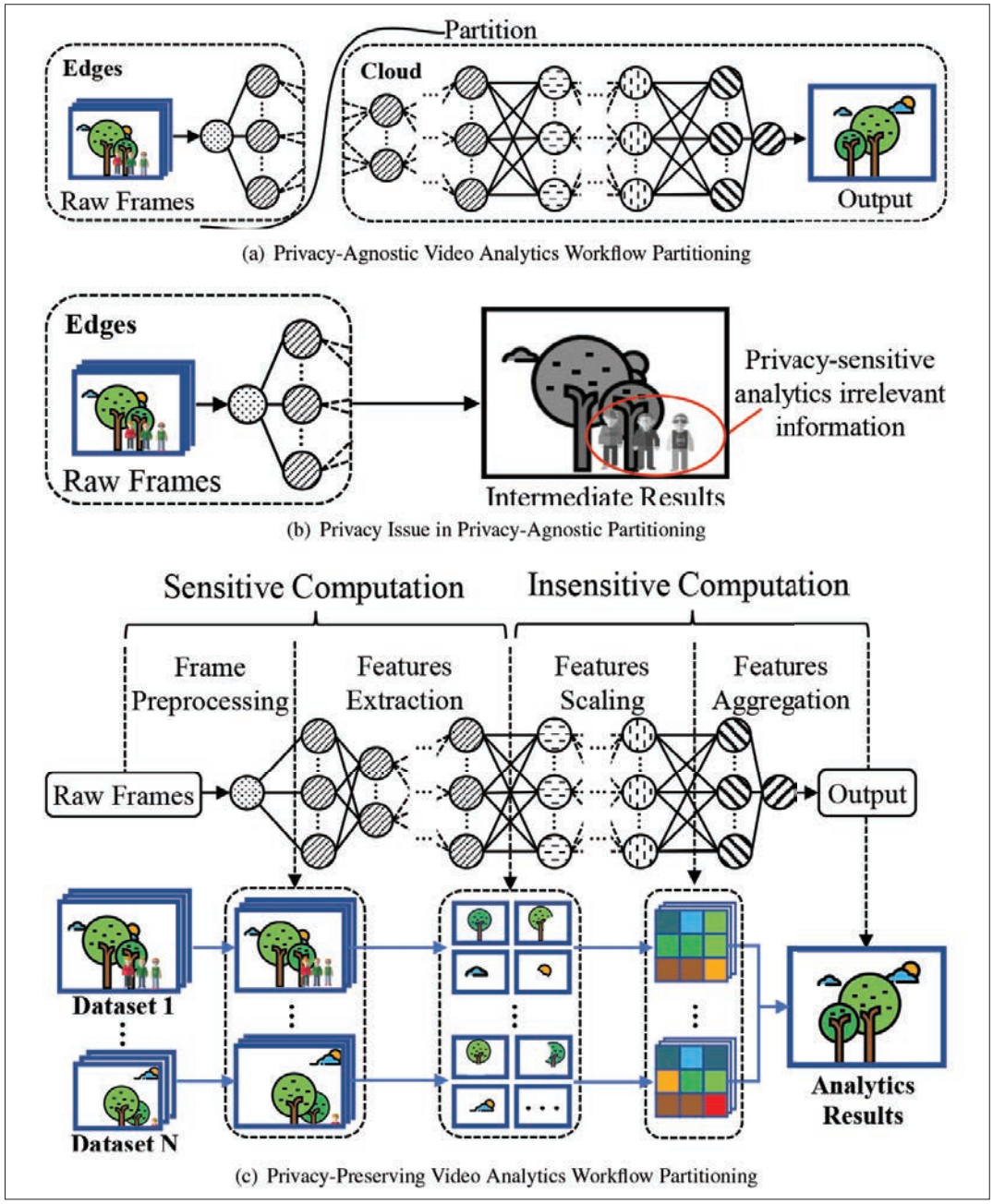

FIGURE 1. The video analytic computing workflows: a) privacy-agnostic video analytics workflow partitioning; b) privacy issue in privacy-agnostic partitioning; c) privacy-preserving video analytics workflow partitioning.

We show how FEVA can be used in practice by a case study where we implement the FEVA architecture to support a multi-view vehicle 3D reconstruction application. We implement FEVA by extending TensorFlow Federated (TFF), an open source FL platform developed by Google. We pre-train four models using two real-world traces on the cloud. Our experiment uses four Amazon DeepLens cameras for multi-view vehicle 3D reconstruction. We evaluate the FEVA performance in comparison with two methods: a collaborative video analytic method and a privacy masking video analytic method. FEVA successfully makes approaches to solve two major challenges in FA: there is no existing architecture for FA on privacy-preserving video analytics, and it is necessary to design optimization algorithms for FEVA resource management and performance. As a result, we observe that FEVA increases the video analytics accuracy up to 1.90 times and 1.34 times, respectively.

In summary, the contributions of this article are:

- We show that a new architecture is necessary by carefully studying privacy-sensitive video analytics applications and the limitation of existing architectures. We clarify the position of FEVA in the literature.
- We analyze the video analytics computing workflow and design the FEVA architecture, which is privacy-preserving and resource-efficient.
- We show how the FEVA architecture can be put into practice by a case study on multi-view vehicle 3D reconstruction with real-world implementation in TensorFlow Federated and DeepLens cameras.

Related Architecture

Video analytics applies pre-trained machine learning models (e.g., DNN/CNN) to analyze video frames for a specific analytics task. More specifically, video frames are fed into the DNN/CNN model, and the computing goes through the layers in the DNN/CNN model (Fig. 1a).

In recent years, several architectures have been proposed to meet diverse application requirements and resource constraints. Video analytics were first conducted in the cloud to serve video analytics queries [7]. In many applications, videos are generated in edge devices such as smart cameras. The computing and communication resources of the edge devices are limited. Edge-cloud computing architectures [8] were proposed where edge devices conduct initial computing of a few DNN/CNN layers to reduce the amount of data transmission (Fig. 1a). In some recent applications, a video analytics task cannot be completed by one single edge device, and collaborative video analytics architectures were proposed to coordinate multiple edge cameras [4].

In these architectures, privacy is not an application concern. Privacy-sensitive applications are emerging. One example is the HD map for autonomous driving, and it is becoming a killer application. An HD map contains not only static objects (e.g., roads and signs) but also real-time objects (e.g., vehicles). The HD map is constructed by diverse industry players, and a critical player comprises vehicles with onboard cameras. Note that the captured videos contain not only HD map relevant information, such as real-time vehicles at an abstract object level, but also *privacy-sensitive HD map irrelevant information* such as human faces and license plate numbers. To the best of our knowledge, no existing architecture is designed to support this class of applications: collaborative video analytics with privacy concerns, such as HD maps. FEVA fills in this gap.

We comment that there are architectures for model training. In a parameter server (PS) architecture [9], distributed workers train a DNN/CNN model and exchange model updates (i.e., gradients) through a parameter server. The FL architecture [10] can be seen as a PS architecture, but raw data will not be exchanged to protect privacy. FEVA differs since FEVA targets the model inference phase. User inputs are privacy-sensitive; thus, model training (FL) and model inference (FA) have to be done by keeping data local. FEVA differs from Google FA [6] in that FEVA emphasizes video analytics applications that have unique computing workflows.

The FEVA Architecture

Privacy-Preserving Video Analytics Workflow Partitioning

In current privacy-agnostic architectures, the partition of the video analytics computing task is based on resource optimization consideration. This can lead to privacy issues. For example, Fig. 1b shows that if the DNN/CNN layers are partitioned as in Fig. 1a, the intermediate results expose analytics-task-irrelevant (sky and trees) but privacy-sensitive (to people) information. This is because the inference of the DNN/CNN layers up to this stage is still a binary image. To solve this problem, we look into the details of a video analytics computing task workflow and separate it into two parts: sensitive computation and insensitive computation. The separation position is decided by the intermediate results in each layer of the model. The sensitive computation part consists of the model layers whose intermediate results are highly similar to the raw input data, and the insensitive computation part comprises those with insight information and cannot be traced back to raw data. We continue to subdivide the sensitive computation into two steps with practical meaning: *frame preprocessing* and*features extraction*, and subdivide the insensitive computation into another two steps: *features scaling* and *features aggregation* (Fig. 1c). Frame preprocessing reshapes the images in different sizes into a regular one for later batch operations. Feature extraction extracts the features from the images. Intuitively, feature extraction collects the regions or information of interest relevant to solving the analytics task and outputs several feature fragments in different shapes and sizes containing critical information from the raw inputs. Feature scaling normalizes and standardizes the fragments. Feature aggregation aggregates the regulated features and returns the analytic results. A key observation is that the extracted features are only related to the target of the video analytics task, not to the raw video images. Consider an example video analytics task of vehicle tracking. The DNN/CNN model will be pretrained to identify vehicles. The extracted features will then be the features of the vehicles. It will not expose privacy-sensitive data (e.g., pedestrians) in the video images. Therefore, keeping the feature extraction step local can effectively preserve privacy as long as the trained DNN/CNN model does not conflict with individual privacy concerns.

In this way, privacy-preserving video analytics is transformed into ensuring the privacy policy of an edge device to have no conflict with the pretrained DNN/CNN model. If the video analytics task has a conflict with the local privacy policy of an edge device, another edge device without any privacy conflict can be chosen; we call the peers with no privacy conflict with the video analytics task the privacy-preserving peers. Clearly, if all

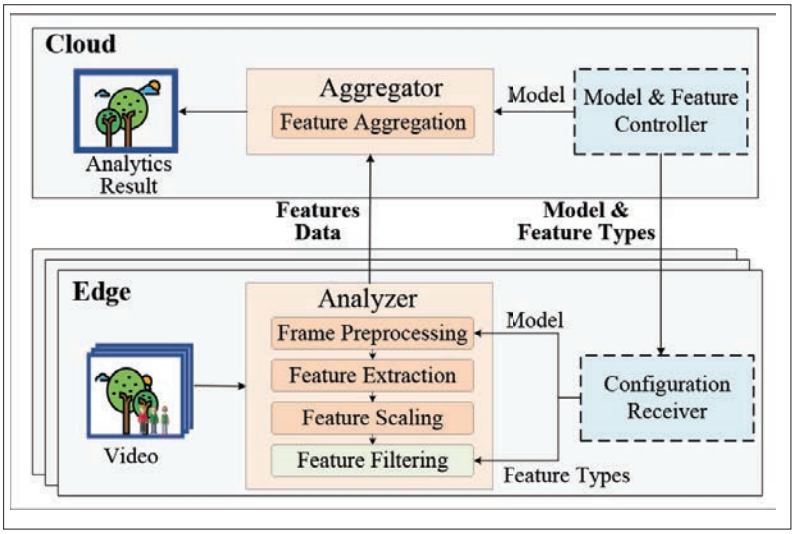

FIGURE 2. The FEVA architecture. The computing and control modules are in the solid and dashed boxes, respectively.

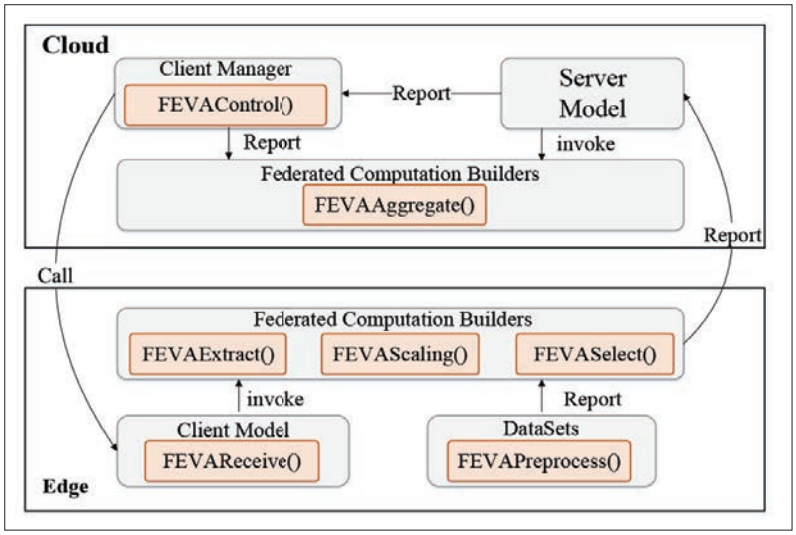

FIGURE 3. FEVA implementation based on TFF.

peers have privacy conflicts with the video analytics task, this video analytics task itself would be very sensitive and thus cannot be completed. We present the FEVA architecture given privacy-preserving peers and leave it for future work how such peers can be selected.

The FEVA Architecture

We show the FEVA modular architecture in Fig. 2. FEVA has computing modules that undertake video analytics tasks and control modules that manage and optimize the resource constraints in diverse applications.

Intrinsically, the FEVA computing modules partition the video analytics computing task by a Frame Preprocessing module, a Feature Extraction module, a Feature Scaling module in the edge, and a Feature Aggregation module in the cloud.

The FEVA control modules maximize video analytics accuracy under computing and communication resource constraints of the edge devices. Note that a specific video analytics task can be achieved by multiple machine learning models, leading to diverse computing requirements and generating different amounts of intermediate data for communication. FEVA has a *Model & Feature Controller* module on the cloud side for model selection and feature filtering, as well as a *Configuration Receiver* in the edge to configure the model and the feature types from the *Model & Feature Controller* module. A *Feature Filtering computing* module is added to drop the features as instructed by the *Configuration Receiver* module.

High Accuracy Video Analytics with Resource Constraints

Different applications can have diverse resource constraints and a number of pre-trained models. We maximize the video analytics accuracy under resource constraints.

The FEVA Resource Optimization (FEVA-RO) Problem: Given the computation capacity and the communication capacity of the edge devices and the cloud, the pre-trained models, the features, and the required delay determine a model and a set of features for the video analytics task to maximize the analytic accuracy.

The models vary in accuracy and needed computation time under the given computation resource. The features vary in "importance" for accuracy and needed communication time. *Feature importance* is defined as the accuracy degradation incurred by dropping the feature [11]. To adapt the privacy restrictions in a FEVA application, a privacy negative correlation parameter is also included in each feature importance computation, which can reduce the importance if the features have high privacy risk. It is set up according to variant FEVA application requirements and local law restrictions. Please note that the accuracy and computation time of a model can be derived through measurement, and the feature importance can be derived through measurement on a small dataset in advance. The FEVA-RO problem has a Knapsack structure, as the required delay, the models and features, the model accuracy and the feature importance, the computation time, and the communication time for a given model and features can be regarded as the capacity of the knapsack, the weight of the item, and the value of the item, respectively. The FEVA-RO problem is equivalent to a Knapsack problem, which has been proven as NP-hard.

The FEVA-RO problem is NP-hard. It is unrealistic to find a globally optimal solution within polynomial time. We design the Maximize Analytics Accuracy (MAA) algorithm, which divides the FEVA-RO problem into two subproblems: the *Model Selection Problem* and the *Feature Filtering problem*:

The Model Selection Problem: Given the pre-trained models with the accuracy and the computation time of each model, determine a model for video analytics, subject to the constraint that the computation time is within the required delay, to maximize the analytics accuracy.

The Feature Filtering Problem: Given the selected model, the features with feature size and feature importance, the required delay, and the bandwidth determine a set of features to maximize the total feature importance.

Accordingly, we develop two subalgorithms, specifically, the *Model Selection Algorithm* and a *Feature Filtering Algorithm*, to solve the above two subproblems, respectively.

The Model Selection Algorithm simply traverses the pre-trained models to find out the model with the maximum accuracy while the computation time of the model is within the required delay. This algorithm outputs the selected model and the computation time. With the computation time, the algorithm can get the required communication time by subtracting the computation time from the required delay.

The Feature Filtering Algorithm is a simple greedy algorithm, which selects the feature with the maximum ratio of the feature importance and the feature size one by one, as long as the communication delay constraint holds.

The developed MAA algorithm consisting of the two subalgorithms works as follows: When a video analytics task generates at the edge, it takes the computation capacity and communication capacity of the edge devices and the cloud, the pre-trained models, the features, and the required delay as inputs, runs the model selection algorithms to determine the model for video

analytics first, and then runs a feature filtering algorithm to determine a set of features. In this way, the model and the features are determined to maximize the accuracy. Our algorithm decouples model selection and feature filtering. We admit that joint optimization of model selection and feature filtering can achieve better performance. However, the evaluation shows that our decoupled approach achieves acceptable performance, and a joint optimization requires a more complex algorithm and implementation, which is left for future investigation.

A Case Study on Multi-View Vehicles 3D Reconstruction

In this section, we apply the FEVA architecture to support a multi-view vehicles 3D reconstruction (MV3DR) application [12]. MV3DR constructs 3D models for vehicles through captured videos. Video analytics using 3D models can help to establish the real-time objects of the HD map. Clearly, constructing the 3D models needs videos from collaborative cameras since the visual appearance of vehicles varies greatly from different viewpoints (e.g., the front and rear views of the vehicle). Consequently, multiple video images are exploited for the multi-view vehicle 3D reconstruction via the 3D reconstruction algorithm [1].

Implementation

FEVA Implementation: We implement FEVA by extending Google's TensorFlow Federated (TFF) [13]. The codes are publicly available on the GitHub site.[1]

Briefing of the TFF Architecture: Figure 3 shows the architecture of TFF. It works as follows:

1. The `Client Manager` on the cloud monitors the states of the edge devices (e.g., the bandwidth) and selects a set of edge devices for the next round of model training. It sends an FL plan including the global model and the training parameters (e.g., the learning steps) to the selected edge devices.
2. The `Client Model` on the edge updates the local model and training parameters according to the received FL plan and further reforms these as a TensorFlow graph file that is the execution description file and is sent the federated computation builders (FCBs) at the edge.
3. The `DataSets` at the edge preprocess the data to the required format and send it to the `FCB`.
4. The edge-side `FCB` takes the TensorFlow graph file and the reformed data as inputs and runs the training algorithm to update the local model, which is then reported to the `Server Model` on the cloud.
5. After receiving the local models from all edges, the `Server Model` invokes the `FCB` to aggregate these local models.
6. Then the cloud-side `FCB` runs the model aggregation algorithm to update the global model, which is then reported to the `Client Manager`. The above steps repeat until the global model converges.

FEVA Modules Implementation: We implement FEVA modules by revising the modules in TFF as shown in Fig. 3. For the Model & Feature Controller module, we add a `FEVAControl` function into the Client Manager, which runs the model selection and feature filtering algorithm to compute the model and the feature types. The `FEVAControl` gets the bandwidth of the edge devices by calling the state monitoring application programming interface (API) provided by TFF and encapsulates the results as an FL plan to send to the edge. We add the `FEVAReciever` function into the `Client Model` to reform the received models as a TensorFlow graph file so that the model can be run in the TensorFlow environment. We modify the `DataSets` to reform the frames by adding a `FEVAPreprocess` function.

In TFF, FCB modules provide the execution environment for all operations, and especially provide lots of functions for model testing (model inference). To exploit the mature model execution functions and environment, we add a `FEVA-Aggregate` function to the cloud-side FCB to run the models for feature aggregation. We also add `FEVAExtract` and `FEVAScaling` functions into edge-side FCB to run models for feature extraction and feature scaling, respectively. The `FEVASelect` function is added to filter our given features to reduce the feature size.

FIGURE 4. A FEVA-supported multi-view vehicles 3D reconstruction system implementation.

MV3DR Implementation: We implement MV3DR on FEVA. We pre-train four models for MV3DR with different levels of accuracy on the cloud: an LSTM model, a generic access network (GAN)-based model, and two CNN-based models. We derive the inputs of function `FEVAControl` in advance:

1. We measure the computation and communication time by running the models on the deployed cameras and the computer.
2. We can derive the feature sizes directly from the model definition.
3. For feature importance, we apply an adaptive multi-view features selection (AMFS) [11] method to leverage all the features extracted by a weight matrix, which is trained by a small valid dataset, VeRi [14], to rank the features according to their importance. We reform the frames to a 256 × 256 size in 8-bit color by resampling using pixel area relation methods to get better performance for our case study.

[1] https://github.com/polyuDLab/FEVA-DEMO

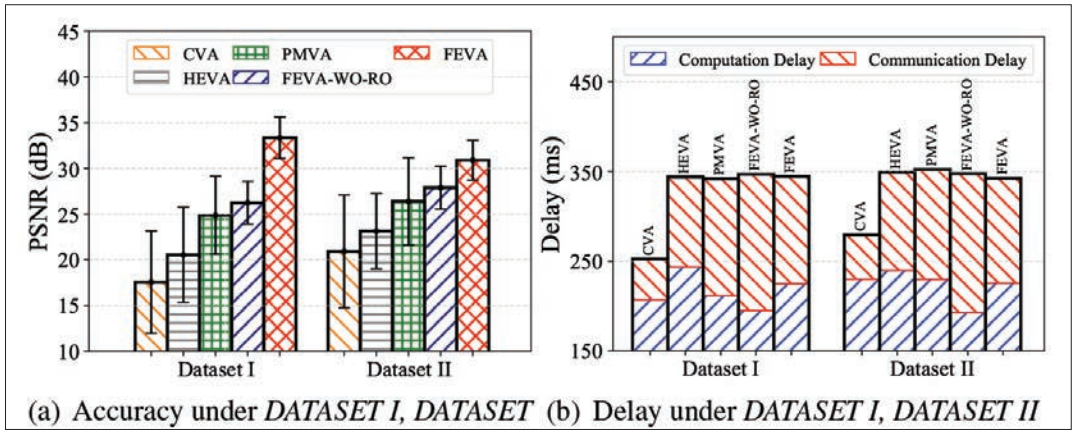

FIGURE 5. The performance of CVA, HEVA, PMVA, FEVA-WO-RO and FEVA under the different datasets: a) accuracy under *DATASET I*, *DATASET II*; b) delay under *DATASET I*, *DATASET II*.

Evaluation

Evaluation Setup: We evaluate the performance of our FEVA-supported MV3DR in an environment with four Amazon DeepLens cameras acting as edge devices and a laptop acting as a cloud server (Fig. 4). We use two popular benchmark vehicle images datasets for model training:

1. VeRi [14], denoted as *DATASET I*, contains over 50,000 images of 776 vehicles.
2. CityFlow [15], denoted as *DATASET II*, contains more than 10,000 images of 666 vehicles.

Each vehicle is captured in both datasets from different viewpoints (e.g., front, front-side, side, rear-side, rear) under different illumination and resolution conditions. We train the models with the dataset on a computation server with an RTX-2080Ti GPU and an Intel i7 CPU. The DeepLens captures the video streams from the dataset in 10 fps and connects to the cloud server by 2.4 GHz WiFi. The required delay is set to 350 ms. The system conducts the MV3DR task once per second.

Evaluation Criteria: We evaluate the accuracy and the delay performance of the video analytics application supported by FEVA. We use peak signal-to-noise ratio (PSNR) as the evaluation metrics for accuracy:

$$PSNR = 10\lg\frac{M^2}{MSE} \quad (1)$$

where M = 255 since our image is coded in 8 bits, and *MSE* represents the mean square error between the imputted frames and the ground truth.

Baselines for Comparison: There is currently no architecture for collaborative video analytics with privacy concerns. We design two straightforward privacy-preserving schemes as baselines for comparison:

- Collaborative Video Analytics (CVA): The edges automatically ignore the frames including private information, such as human faces and licenses plates, and refuse to upload those to the cloud server.
- Homomorphic Encryption Video Analytics (HEVA): The edges automatically encrypt the data before uploading to the cloud server, preventing privacy leaking. The cloud receives and applies complex homomorphic analytics algorithms without decrypting them.
- Privacy Masking Video Analytics (PMVA): The edges detect and circle the outlines of this private information and replace it with null values. The cloud receives these processed frames with blanks.

We also explore FEVA's internal components (i.e., the MAA algorithm for the FEVA resource optimization) to better understand its contribution to the performance of the system. Thus, we implement FEVA without resource optimization (FEVA-WO-RO); that is, FEVA selects a model with maximum accuracy from the set of models whose delay meets the required delay without running the MAA algorithms.

Experiment Results: In this section, we present our experimental results.

Performance Improvement: Figure 5a shows the PSNR of different methods under both datasets. We also measure the computation delay, the communication delay, and the overall delay to complete a video analytics task in Fig. 5b. There are several key observations.

First, FEVA outperforms CVA in both datasets. For example, in *DATASET I*, the PSNR of CVA and FEVA reaches up to 17.56 dB and 33.34 dB, respectively. FEVA gets 1.90 times PSNR than that of CVA. This is because CVA streams raw frames to the cloud without privacy protection. Edges may drop important frames instead of uploading them to the cloud because of their private information. The cloud cannot get sufficient data to get better performance. However, in FEVA, all the features the cloud needs are permitted to upload. In exchange, it takes longer and increases the computation delay, as shown in Fig. 5b.

Second, FEVA outperforms HEVA and PMVA in both datasets. We take *DATASET I* as an example; the PSNR of HEVA and PMVA reaches up to 20.55 dB and 24.87 dB. FEVA gets 1.34 and 1.62 times improvement over HEVA and PMVA, respectively. This is because the encryption computation and masking operations in edges are highly time-consuming, limiting the selection of analytics models running on the cloud since the total time is constant. The computation delay of PMVA is larger than that of FEVA since the detection of privacy information still requires another neural network running at edges. In contrast, FEVA runs only some of the layers.

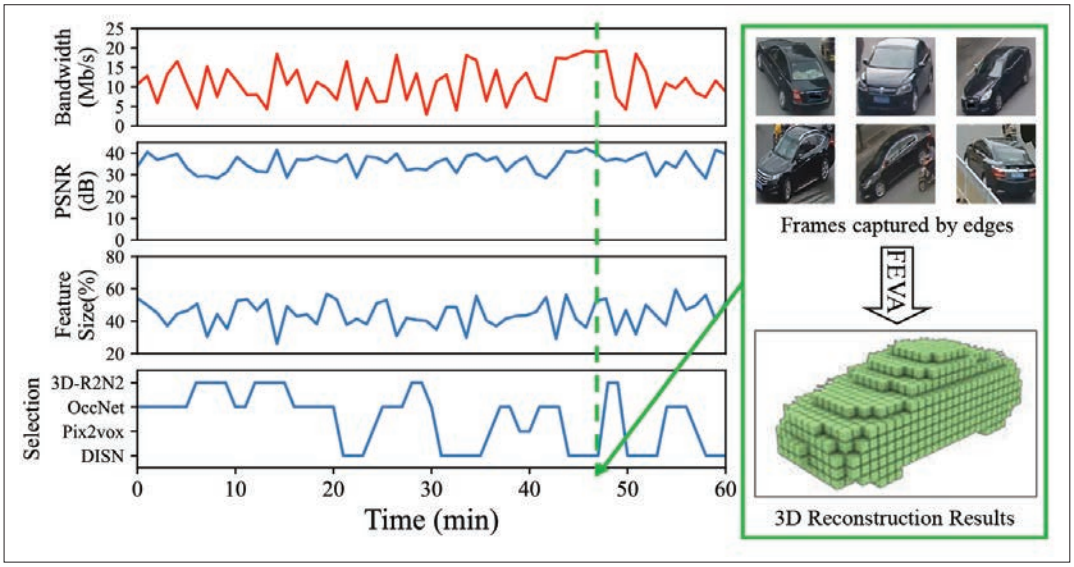

FIGURE 6. The end-to-end operations of FEVA in the field.

Figure 5a also shows the PSNRs of FEVA and FEVA-WO-RO in both datasets. We notice that FEVA has an accuracy improvement of 1.23 times compared to FEVA-WO-RO in *DATASET I*. These results illustrate that the model selection and feature filtering algorithms of FEVA work effectively.

Moreover, we compare FEVA to FEVA-WO-RO in terms of the computation time and communication time. We observe that FEVA and FEVA-WO-RO have similar overall delay and meet the delay requirement. We can see that FEVA outperforms FEVA-WO-RO with a 16.80 percent reduction in communication time.

This illustrates that our resource optimization algorithm, MAA, can significantly reduce the size of features streamed to the cloud. We also notice that FEVA consumes more computation time compared to FEVA-WO-RO. It means more computation time can be allocated for running a complex model with higher accuracy in FEVA. These results confirm that the proposed algorithms can improve the analytics accuracy.

The End-to-End Operations of FEVA: Figure 6 shows the end-to-end operations in the field of FEVA in 60 minutes. The top graph shows the bandwidth changing under the 2.4 GHz WiFi connection. The second and third graphs show the average PSNR and the percentages of uploading features of the 3D reconstruction in each minute. The bottom graph displays the model selected by FEVA. On the right of Fig. 6, we display a 3D reconstruction result at about 45 min. It indicates that FEVA adjusts the model and feature filtering algorithm to successfully achieve high analytics quality in runtime. For example, at about 42 minutes, the bandwidth rises, and FEVA rapidly switches to the highest-accuracy model and uploads more features to the cloud. It successfully starts to maintain the highest PSNR at 45 minutes until the bandwidth drops at 48 minutes. These validate that our proposed method is feasible and effective in practice.

Conclusion

In this article, we propose FEVA, a federated video analytics architecture, for privacy-persevering and resource-efficient video analytics applications. Intrinsically, FEVA keeps the video image data local to the edge for analytics and transmits the analytics results to the cloud for aggregation. We develop algorithms that can partition the video analytics computing tasks in a way that is privacy-preserving and maximizes the overall analytics accuracy under the computing and communication resource constraints of the edge devices. We implement a FEVA-supported multi-view vehicle 3D reconstruction application. Evaluation results show substantial improvements in video analytics accuracy.

Acknowledgments

Our sincere thanks to the following grants for supporting this work: GRF 15210119, 15209220, 15200321, ITF-ITSP ITS/070/19FP, CRF C5026-18G, C5018-20G, PolyU 1-ZVPZ, 1-ZVUG, and a Huawei Collaborative Project.

The authors would like to thank Xiaoyi Wang, who kindly and bravely shared seven of his toy cars (Fig. 4), without which the prototype schematic of our experiments could not have been constructed.

References

[1] H. Xie *et al.*, "Pix2vox: Contextaware 3d Reconstruction from Single and Multi-View Images," *Proc. IEEE ICCV '19*, Seoul, Korea, Oct. 2019.
[2] Y. Zhou and L. Shao, "Aware Attentive Multi-View Inference for Vehicle Re-Identification," *Proc. IEEE CVPR '18*, Salt Lake City, UT, June 2018.
[3] S.-W. Kim *et al.*, "Edge-Network-Assisted Real-Time Object Detection Framework for Autonomous Driving," *IEEE Network*, vol. 35, no. 1, Jan./Feb. 2021, pp. 177–83.
4] C. Wang *et al.*, "Joint Configuration Adaptation and Bandwidth Allocation for Edge-Based Real-Time Video Analytics," *Proc. IEEE INFOCOM '20*, virtual, July 2020.
[5] Q. Yuan *et al.*, "Toward Efficient Content Delivery for Automated Driving Services: An Edge Computing Solution," *IEEE Network*, vol. 32, no. 1, Jan./Feb. 2018, pp. 80–86.
[6] D. Ramage, "Federated Analytics: Collaborative Data Science Without Data Collection"; https://ai.googleblog.com/2020/05/federated-analytics-collaborative-data.html, accessed May 2020.
[7] X. Ran *et al.*, "Deepdecision: A Mobile Deep Learning Framework for Edge Video Analytics," *Proc. IEEE INFOCOM '18*, Honolulu, HI, Apr. 2018.
[8] C. Hu *et al.*, "Dynamic Adaptive DNN Surgery for Inference Acceleration on the Edge," *Proc. IEEE INFOCOM '19*, Paris, France, Apr. 2019.
[9] M. Li *et al.*, "Scaling Distributed Machine Learning with the Parameter Server," *Proc. USENIX OSDI '14*, Broomfield, CO, Oct. 2014.

[10] Q. Yang *et al.*, "Federated Learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 13, no. 3, 2019, pp. 1–207.
[11] Z. Wang *et al.*, "Adaptive Multiview Feature Selection for Human Motion Retrieval," *Signal Processing*, vol. 120, 2016, pp. 691–701.
[12] N. D. Reddy, M. Vo, and S. G. Narasimhan, "Carfusion: Combining Point Tracking and Part Detection for Dynamic 3d Reconstruction of Vehicles," *Proc. IEEE CVPR '18*, Salt Lake City, UT, June 2018.
[13] "TensorFlow Federated: Machine Learning on Decentralized Data"; https://www.tensorflow.org/federated, accessed Oct. 2019.
[14] X. Liu *et al.*, "Large-Scale Vehicle Re-Identification in Urban Surveillance Videos," *Proc. IEEE ICME '16*, Seattle, WA, July 2016.
[15] Z. Tang *et al.*, "Cityflow: A Cityscale Benchmark For Multi-Target Multi-Camera Vehicle Tracking and Reidentification," *Proc. IEEE CVPR '19*, Seoul, Korea, Oct. 2019.

Biographies

Chuang Hu (cschu@comp.polyu.edu.hk) received his Ph.D. degree from Hong Kong Polytechnic University in 2019. He is currently a research assistant professor in the Department of Computing, Hong Kong Polytechnic University. His research interests include edge learning, distributed machine learning, and IoT.

Rui Lu (csrlu@comp.polyu.edu.hk) received his B.S. degree from the Department of Computing Science and Engineering, Southern University of Science and Technology in 2019. He is currently a Ph.D. candidate at Hong Kong Polytechnic University. His research interests include edge computing and edge learning.

Dan Wang (csdwang@comp.polyu.edu.hk) received his Ph.D. degree in computer science from Simon Fraser University, Canada, in 2007. He is currently an associate professor in the Department of Computing, Hong Kong Polytechnic University. He was a TPC Co-Chair of IEEE/ACM IWQoS 2020 and a TPC Co-Chair of ACM e-Energy 2020.