Scrava: Super Resolution-based Bandwidth-Efficient Cross-Camera Video Analytics

Yu Liang, Sheng Zhang, *Senior Member, IEEE*, and Jie Wu, *Fellow, IEEE*

***Abstract*—Massively deployed cameras form a tightly connected network which generates video streams continuously. Benefiting from advances in computer vision, automated real-time analytics of video streams can be of practical value in various scenarios. As cameras become more dense, cross-camera video analytics has emerged. Combining video contents from multiple cameras for analytics is certainly more promising than single-camera analytics, which can realize cross-camera pedestrian tracking and cross-camera complex behavior recognition. Some works focused on optimization of cross-camera video analytic applications, but most of them ignore specific network situation between cameras and edge servers. Furthermore, most of them ignore the super resolution technique, which is proven to be a source of efficiency. In this paper, we first verify the potential gain of super resolution on cross-camera video analytic tasks. Then, we design and implement a cross-camera real-time video streaming analytic system, Scrava, which leverages super resolution to augment low-resolution videos and simultaneously reduce bandwidth consumption. Scrava enables real-time cross-camera video analytics and enhances video segments with the SR module under poor network conditions. We take cross-camera pedestrian tracking as an example, and experimentally verifies the effectiveness of super resolution on real-time cross-camera video analytics. Compared with using low-resolution video segments, Scrava can improve the F1 score by 47.16%, verifying the feasibility of exploiting super resolution to improve the performance of real-time cross-camera video analytic systems.**

***Index Terms*—Edge computing, video analytics, cross camera, super resolution.**

I. Introduction

ALTHOUGH real-time video stream analytics based on single camera can achieve powerful functions such as object detection, semantic segmentation, and pedestrian recognition, single camera has only a limited coverage field of view, and information caught by single camera cannot fully describe the behavior of a specific object when the object moves within a large spatial scale (e.g., campus, city). Generally, to achieve a complete surveillance of a specific area, multiple cameras are deployed with complementary views between them.

In such scenarios, cross-camera real-time video stream analytics is essential, which is capable of combining information from multiple cameras to achieve functions that cannot be achieved by single-camera video analytics. For example, when a suspect escapes the scene of an incident and roams around the city, constantly appearing in the view of different cameras, cross-camera real-time video analytics is indispensable to help the police continuously track the suspect.

Figs. 1(a) and 1(b) illustrate a classic cross-camera dataset, i.e., DukeMTMC [29], which is composed of videos captured by eight cameras deployed at Duke University, where Figure 1(a) shows the deployment of these cameras, including locations and fields of view coverage, and Figure 1(b) gives a sample image of the video captured by all eight cameras. It can be seen that multiple cameras provide a more complete coverage of the campus, thus allowing for continuous tracking of specific objects of interest at a larger spatial scale.

However, cross-camera video analytics is obviously a more difficult task than single-camera video analytics, in the sense that cross-camera video analytics requires not only analyzing video streams from each camera, but also capturing the cross-camera correlation information, such as associating the same person appearing in different cameras. To associate information across different cameras, lots of works on re-identification (ReID) have been studied, especially pedestrian re-identification [38]. Pedestrian re-identification initially emerged in conjunction with multi-camera tracking [30], and then gradually developed into an independent research subject. The development of deep learning has also sparked the development of pedestrian re-identification, and neural networks have been introduced into the field of pedestrian re-identification, becoming a popular solution in the field [20], [32].

At the same time, limited by the computing resources of the cameras, video streams are continuously transmitted from the cameras to the associated edge servers for analytics, and the more cameras deployed, the larger the amount of data to be transmitted, and the higher the requirements for the network bandwidth. In single-camera video analytics, network conditions affect the resolution of the video transmitted to the server, thus affecting the performance of the video analytic task on one single camera, while in cross-camera video streaming analytic tasks, the impact of the network conditions on the final task performance is amplified, as the network conditions not only affect video analytics on a single camera, but also affect data correlation between multiple cameras.

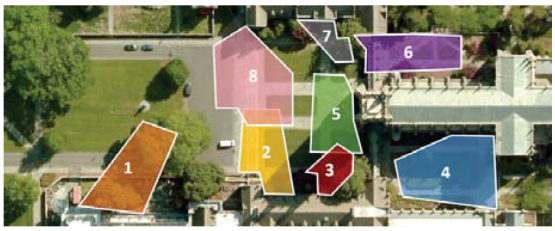
In summary, the challenges of implementing a real-time cross-camera video analytic system are as follows:

- **Changes of video resolution within one camera.** The video segments transmitted from a camera to an edge server are faced with changes in video resolution due to fluctuations in network bandwidth, which can affect real-

Y. Liang is with the School of Computer and Electronic Information and the School of Artificial Intelligence, Nanjing Normal University, Nanjing 210046, China. She is also with the State Key Laboratory for Novel Software Technology, Nanjing University. E-mail: liangyu@njnu.edu.cn.

S. Zhang is with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China. E-mail:sheng@nju.edu.cn.

J. Wu is with the Center for Networked Computing, Temple University, Philadelphia, PA 19122, USA. E-mail: jiewu@temple.edu.



(a) Camera deployment

(b) Samples

Fig. 1: DukeMTMC [29], a classic cross-camera dataset, is composed of videos captured by eight cameras deployed at Duke University. (a) shows the deployment of eight cameras, including locations and fields of view coverage, and (b) gives a sample image of the video captured by all eight cameras.

time video analytic applications within a single camera, such as object tracking. In object tracking, when video resolution switches, the coordinates and features of the tracked object will change accordingly, probably leading to tracking interruptions, errors, and other problems.

- **Variation in video resolution across cameras.** Since network conditions between different cameras and the edge server vary, video segments transmitted to the edge server for real-time video analytics would have different resolutions. When performing cross-camera data association, low-resolution video segments from some cameras would result in difficulty in cross-camera information association due to the loss of information caused by the reduced resolution.
- **High bandwidth requirements.** In cross-camera real-time video streaming, on one hand, video data captured by multiple cameras need to be transmitted simultaneously. On the one hand, video data is a highly storage-intensive data type. Nevertheless, the transmission must be achieved in real-time, which finally translates into high bandwidth requirements.

There have been several works focusing on cross-camera video analytics [24], [25], [14], [11]. For example, Caesar [24] studies cross-camera complex behavior recognition, and presents a hybrid design combining rule-based and DNN-based detection to determine whether the complex behavior occurs; Spatula [14] reduces the computational overhead of cross-camera video analytics by exploiting the spatio-temporal correlation between cameras through offline statistics; CrossRoI [11] divides each frame evenly into rectangular blocks and runs the existing algorithm [19] on synchronized videos from different cameras to determine the overlapping areas of cameras by the same object that appears in the field of view of multiple cameras. However, these works hardly took super resolution into consideration, and thus lost the opportunity of utilizing the power of super resolution.

This paper is the first to use super resolution for the optimization of cross-camera video analytics, which improves the performance of cross-camera video analytic applications under bandwidth constraints and provides a possible optimization direction for future cross-camera video analytics. We first verify the effectiveness of super resolution (SR) on cross-camera video analytic tasks. Then, we design and implement Scrava (Super resolution-based Cross-camera bandwidth-efficient video analytics), which exploits super resolution to enhance low-resolution video segments at the edge server side, mitigate the impact of inadequate and unstable (wireless) network bandwidth in the edge environment on real-time cross-camera video analytics, and thus improve the performance of cross-camera video analytic tasks under poor network conditions. Finally, we evaluate the effectiveness of super resolution in cross-camera real-time video stream analytics using cross-camera pedestrian tracking as an example.

The contributions of this paper are three-fold:

- To our best knowledge, this paper applies super resolution to real-time cross-camera video analytic tasks for the first time. Through pilot experiments, we show SR can significantly improves the effectiveness of real-time video analytic applications within one single camera as well as across cameras under restricted network conditions, and improves the stability of cross-camera video analytic results.
- We design and implement a scalable, real-time cross-camera video analytic system, Scrava, which integrates an SR module into the detection/tracking module, the bandwidth prediction module, and the re-identification module. Scrava enables real-time cross-camera video analytics and enhances video segments with the SR module under poor network conditions.
- We take cross-camera pedestrian tracking as an example, and experimentally verifies the effectiveness of super resolution on real-time cross-camera video analytics. Compared with using low-resolution video segments, Scrava can improve the F1 score by 47.16%, verifying the feasibility of exploiting super resolution to improve the performance of real-time cross-camera video analytic systems.

The rest of this paper is organized as follows. We provide related works in Section II. Section III motivates the design of Scrava. Section IV provides the design of Scrava and implementation issues. Section V evaluates the performance of Scrava. Before concluding the paper in Section VII, we show limitations and future works in Section VI.

II. Related Works

Existing studies can be classified into two broad types: single-camera and cross-camera video analytics.

Single-camera video analytics. Many existing works on single-camera video analytics focus on reduce bandwidth consumption. Vigil [34] utilizes a lightweight preprocessing at mobile devices to obtain some preliminary results, which are then transmitted to edge servers for deciding the set of frames to be processed at servers. Glimpse [5] and O^3 [12]

TABLE I: Comparison between existing cross-camera video analytics methods with our work

Existing Studies	Target Environment	Optimization Goal	Main Approach
Caesar [24]	Overlapping fields of views	Support complex activity query	Rule-based and DNN-based detection
CONVINCE [25]	Overlapping fields of views	Reduce bandwidth and computation overhead	Filter out irrelevant frames
Spatula [14]	Non-overlapping fields of views	Reduce bandwidth and computation overhead	Offline statistics-based correlation
CrossRoI [11]	Overlapping fields of views	Reduce bandwidth and computation overhead	Eliminate spatial redundancy
Polly [17]	Overlapping fields of views	Reduce inference latency	Position mapping and results merging
Scrava	Overlapping fields of views	Reduce bandwidth overhead and improve cross-camera ReID	Augment low-resolution videos with SR

observe that object detection is accurate but slow, while object tracking is fast but it may accumulate errors; therefore, they intelligently switch between on-device tracking and on-server detection to filter out unnecessary frames. Taking one step further, Reducto [21] adapts filtering decisions to the time-varying correlation between feature type, filtering threshold, query accuracy, and video content, thus achieving a better filtering performance. Elf [35] offloads video analytics tasks within a frame to multiple edge servers to optimize analytics latency. Some other related studies optimize single-camera video analytics through adjusting video configuration. Chameleon [15] finds the best video configuration by maintaining a set of top-K configurations. DDS [6] uses a two-phase method: in the first phase, a client transmits low-resolution video to an edge server for analytics, and in the second phase, after receiving feedbacks from the server, the client transmits high-resolution regions, which are not identified with a high confidence level, to the server. JCAB [33] utilizes the Lyapunov optimization framework to decouple the long-term optimization problem of video configuration selection and bandwidth allocation into a series of short-term one-slot problems. These works provide efficient single-camera analytics approaches that better support Scrava.

Cross-camera video analytics. There have been several works focusing on cross-camera video analytics. Tab. I summarizes the comparison results between existing cross-camera video analytics methods with our work. Caesar [24] proposes a hybrid design combining rule-based and DNN-based detection, based on which Caesar matches the behavior definition graph by object detection and association results to determine whether the complex behavior occurs. CONVINCE [25] exploits the spatio-temporal correlation of cameras to eliminate redundant frames, thereby reducing bandwidth and computational overhead. Spatula [14] determines the spatio-temporal correlation between cameras through offline statistics and is thus more reliable. CrossRoI [11] observes that, densely deployed cameras are likely to have overlapping views between them, therefore, CrossRoI divides each frame evenly into rectangular blocks and runs the DiDi-MTMC [19] algorithm on synchronized videos from different cameras to determine the overlapping areas of cameras by the same object that appears in the field of view of multiple cameras. Polly [17] utilizes position mapping and results merging to share inference results across cameras, thus eliminating the redundant inference work for objects in the same physical area. Although these works represent substantial effort towards efficient cross-camera video analytics, most of them ignore the super resolution technique, which is proven to be a source of efficiency in improving cross-camera ReID.

III. Motivation

In this section, we verify the improvements of super resolution in stateful computer vision tasks, taking object tracking as the example. We set the results obtained by running the object tracking algorithm on raw-resolution video segments as the ground truth for evaluation. We first present the metrics used in pilot experiments, then we show the experimental results in both the single-camera scenario and the multi-camera scenario.

A. Metrics

In multi-object tracking, the commonly used evaluation metrics are $IDF1$ and $MOTA$, where $IDF1$ emphasizes the association accuracy rather than detection accuracy, it calculates the bijection between the set of predicted trajectories $prID$ and the set of ground truth trajectories $gtID$ in order to evaluate the prediction result, while $MOTA$ is matched at the object detection level. $IDF1$ combines the IDP and IDR, where IDP (ID precision) represents the precision of recognition and it is calculated as:

$$IDP = \frac{|IDTP|}{|IDTP| + |IDFP|}, \tag{1}$$

and IDR (ID recall) represents the recall of recognition, which is calculated as:

$$IDR = \frac{|IDTP|}{|IDTP| + |IDFN|}, \tag{2}$$

besides, $IDF1$ considers both precision and recall, which is calculated as:

$$IDF1 = \frac{|IDTP|}{|IDTP| + 0.5|IDFN| + 0.5|IDFP|}. \tag{3}$$

In the above equations, $|IDTP|$ represents the number of trajectories that correctly match between $prID$ and $gtID$ (trajectory overlap greater than a threshold), $|IDFP|$ represents the number of trajectories in $prID$ that do not match any trajectory in $gtID$, and $|IDFN|$ represents the number of trajectories in $gtID$ that do not match any trajectory in $prID$.

$MOTA$ is a multi-object tracking metric at the object detection level, which calculates the bijection between the predicted bounding box set $prDets$ and the ground truth bounding box set $gtDets$ in each frame, and two bounding boxes are considered as a match (TP) if they are spatially similar enough. FN denotes the case where there is no match

(a) 4 people scenario

(b) 6 people scenario

Fig. 2: The EPFL dataset [8] is a pedestrian video dataset containing multiple cameras, all of which are located approximately 2 meters above the ground and recorded in a synchronized manner, monitoring the same area from different angles. (a) and (b) give two examples of laboratory sequences in the EPFL dataset, which are captured by four cameras with a duration of about 2.5 minutes, a resolution of (360, 288), and a frame rate of 25.

TABLE II: Comparison between four super-resolution models

SR Model	PSNR (dB)	SSIM	F1 score
EDSR X4	27.71	0.74%	42.43
EUSR X4	26.21	0.78%	41.29
MSRN X4	26.39	0.79%	45.19
RCAN X4	28.82	0.80%	43.36

in the set of predicted bounding boxes with the ground truth bounding box set, i.e., the case of missed detection. FP denotes bounding boxes are detected in the predicted bounding box set but not present in the ground truth bounding box set. The $MOTA$ metric also considers the case of Identity Switch (IDSW), including the case where the tracker mistakenly swaps the object ID or the tracking is lost and reinitialized. $MOTA$ is calculated as:

$$MOTA = 1 - \frac{|FN| + |FP| + |IDSW|}{|gtDets|}. \quad (4)$$

B. Potential Gain from Super Resolution on Single-Camera Video Analytics

We conducted experiments with laboratory sequence (6 people) from the EPFL dataset [8] to validate the enhancement of super resolution in the single-camera object tracking case. The EPFL dataset is a pedestrian video dataset containing multiple cameras, all of which are located approximately 2 meters above the ground and recorded in a synchronized manner, monitoring the same area from different angles. Figures 2(a) and 2(b) give two examples of laboratory sequences in the EPFL dataset, which are captured by four cameras with a duration of about 2.5 minutes, a resolution of (360, 288), and a frame rate of 25. In our experiments, we set the tracking interval to 10, i.e., 1 frame is processed and the next 9 frames are skipped. Besides, We use YOLOv8 [4] for object detection and ByteTrack [36] for object tracking.

Theoretically, any super-resolution model could be used in Scrava, as long as the efficiency and cost of the model could be accepted. We implemented EDSR [22] , EUSR [22], MSRN [1], and RCAN [18]. We run them on the VIPeR dataset [9], the results on average are shown in Table II. We see that, all four models perform similarly on the VIPeR dataset. We choose to use EDSR to achieve video super resolution.

The results are shown in Table III. In Table III, the 'LR' row represents the score on the low-resolution video and the 'LR+SR' row represents the score on the low-resolution video enhanced by super resolution, where the super resolution DNN is obtained by offline training on video segments left out by the four cameras. It can be seen that super resolution is effective in improving the performance of single-camera object tracking, no matter what metric ($IDF1$ or $MOTA$) is used. Note that, a super resolution model is actually predicting pixels for each input image, thus super resolution may improve the details of an object of interest as well as introduce additional noises. For example, if the quality of an input image is too low for even humans cannot identify an object in it, then super resolution probably introduces more noises than details. In such case, performing detection on the original image may be better than on the image after super resolution. In summary, although super resolution on low-resolution videos sometimes brings a reduction in precision, it can improve recall which makes up for the drawback in either metric.

C. Optimization for Cross-Camera Video Analytics

There have been several works focusing on cross-camera video analytics. Among them, Caesar [24] studies cross-camera complex behavior recognition, for which the authors propose a hybrid design combining rule-based and DNN-based detection. Caesar provides an extended rule definition language that enables users to easily define complex behaviors. Based on the proposed complex behavior definition, Caesar generates a representation, and subsequently matches the be-

TABLE III: Super resolution enhancements to single-camera object tracking

	Metric	IDP	IDR	IDF1	MOTA
Camera 1	LR	0.7697*	0.5288	0.6269	0.6685
	LR+SR	0.6985	0.5741*	0.6303*	0.7918*
Camera 2	LR	0.6386*	0.4279	0.5125	0.6394
	LR+SR	0.6292	0.5276*	0.5734*	0.7805*
Camera 3	LR	0.8261	0.6540	0.7301	0.7662
	LR+SR	0.8333*	0.7524*	0.7908*	0.8717*
Camera 4	LR	0.7060	0.4960	0.5827	0.6800
	LR+SR	0.7378*	0.6539*	0.6933*	0.8588*

havior definition graph by object detection and association results to determine whether the complex behavior occurs.

Some existing cross-camera video analytic works analyze each video stream independently without exploiting the spatio-temporal relationships between neighboring cameras, leading to a network overhead that grows linearly with the number of cameras and an exponentially increasing computational overhead. To address these issues, CONVINCE [25] proposes a centralized, cross-camera video analytic system that exploits the spatio-temporal correlation of cameras to eliminate redundant frames, thereby reducing bandwidth and computational overhead. CONVINCE assumes that each camera has an AI chip, uses lightweight DNN preprocessing to filter out irrelevant frames to reduce bandwidth consumption. Take pedestrian counting as an example, the camera transmits the frame to the edge server for analytics only when a new object is detected in CONVINCE. In addition, CONVINCE increases the confidence level that a object is detected in the overlapping field of view of multiple cameras, enabling information sharing between neighboring cameras and thus improving the accuracy of video analytics.

CONVINCE manually sets the correlation between cameras based on observations. Different from CONVINCE, Spatula [14] aims to reduce the computational overhead of cross-camera video analytics by exploiting the spatio-temporal correlation between cameras. Spatula determines the spatio-temporal correlation between cameras through offline statistics and is thus more reliable.

CrossRoI [11] is an effort to reduce video transmission bandwidth consumption by eliminating spatial redundancy in multiple cameras. Densely deployed cameras are likely to have overlapping views between them, and when the views overlap, as long as one camera transmits and analyzes that part, the remaining cameras can spare the bandwidth consumption due to transmitting the videos monitoring the same area. Therefore, CrossRoI divides each frame evenly into rectangular blocks and runs the DiDi-MTMC [19] algorithm on synchronized videos from different cameras to determine the overlapping areas of cameras by the same object that appears in the field of view of multiple cameras. When enough information is obtained, CrossRoI solves the optimization problem: the total number of rectangular blocks to be transmitted is minimized while including all objects of interest, thus eliminating the transmission of overlapping camera views and reducing bandwidth consumption.

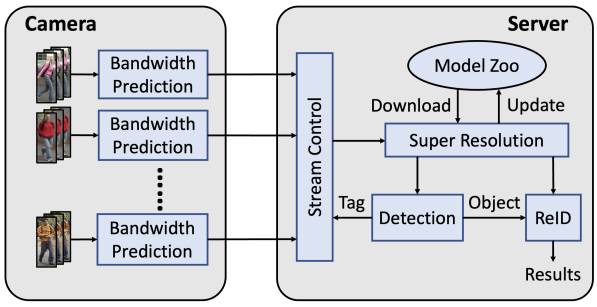


Fig. 3: Overview of Scrava

In general, cross-camera video analytics is a relatively new research area with large potential for its application. This paper is the first to use super resolution for the optimization of cross-camera video analytics, which improves the performance of cross-camera video analytic applications under bandwidth constraints and provides a possible optimization direction for future cross-camera video analytics.

IV. Design of Scrava

In an edge environment, multiple cameras are deployed in the same area and managed centrally by an organization (e.g., campus, company, and city). Usually, these cameras are connected to a cluster of edge servers via a wired or wireless network and continuously transmit video segments to the edge servers for real-time cross-camera analytics. Real-time video stream analytics can be used to meet the growing demand for public security, improve the quality of service in large shopping malls, amusement parks, etc.

Typical cross-camera video analytic applications often include the following two modules: (1) Object detection/tracking module: This module is responsible for implementing video analytics within a single camera, such as extracting, classifying, and tracking objects of interest in the video; (2) Re-identification module: Re-identification is a computer vision task that has been widely studied. Specifically, the goal of re-identification is to discriminate whether the object was captured by another camera at another location (time) for a given object of interest (query) [38]. The cross-camera real-time video analytic pipeline first applies the object detection/tracking module in each single camera to track objects, and then uses the re-identification module to achieve cross-camera tracking.

In the rest of this section, we first present the overview of Scrava, then we introduce the four main modules in Scrava one by one.

A. Overview

The poposed video analytics system, Scrava, is designed to address the problem of degraded accuracy of cross-camera real-time video analytic tasks due to network fluctuations in edge environments. Figure 3 shows the architecture of Scrava. Scrava provides a scheme that enhances the results

of cross-camera real-time video analytics even in the presence of constrained networks. Scrava contains not only the two modules mentioned above, i.e., an object detection/tracking module applied to a single camera and a re-identification module for cross-camera object matching, but also a super resolution module for the enhancement of low-resolution video segments. Note that, the example application implemented in Scrava is cross-camera object tracking, the results of which can serve many subsequent high-level applications (fugitive tracking, Amber Alert, etc.), Scrava can support many other applications that leverage super resolution.

There are a large number of re-identification works based on DNNs [31]. Typically, these works extract the features of an object q using a DNN for a given query individual, and all images in a given image set G (i.e., gallery set) are ranked according to their feature distances from q, with a smaller distance representing more similar to q.

The modeling and overall workflow of Scrava is described as follows: the set of N cameras connected to a centrally managed edge server cluster is $C = \{c_1, c_2, \cdots, c_N\}$. Each camera transmits a constant stream of video segments of corresponding resolution to the edge server based on the bandwidth prediction module, and the duration of each video segment is fixed. If there is no tracking task in the current system, no processing of the video is performed; otherwise, for a tracking object q that is tagged in camera c_q, the edge server discerns whether the received video segment is the original resolution video, and if so, the video segment will be directly processed by the object detection/tracking and re-identification module, otherwise the video segment is enhanced using the super resolution module. Subsequently, Scrava performs cross-camera tracking of q with the goal of returning frames that contain the target q in all cameras until q cannot be tracked. In the server, for each camera c_i, $\forall i \in \{1, 2, \cdots, N\}$, it performs

$$c_i \text{ runs } \begin{cases} \text{the object detection/tracking module,} & \text{if } i = q, \\ \text{the re-identification module,} & \text{o.w.} \end{cases} \tag{5}$$

Scrava performs continuous tracking on video streams from multiple cameras until q disappears from the view of all cameras.

Each module of Scrava is described in details in the following subsections.

B. Super Resolution Module

Generally speaking, since features of the video streams captured by different cameras differ significantly from each other, training a dedicated super resolution DNN for each camera and then enhancing the video streams in that camera would achieve significantly more accurate results on that specific camera than using a pre-trained model on a uniform dataset. Therefore, we train a specific super resolution DNN for each camera in Scrava, which constitutes a set of super resolution model parameters

$$SR_{para} = \{para_1, para_2, \cdots, para_N\}.$$

If Scrava detects that a video segment from camera c_i is downsampled, it uses the corresponding parameter $para_i$ to super-resolve the video segment, reconstructs it to obtain a high-resolution video segment, and uses the reconstructed segment for subsequent cross-camera object tracking. Besides, Scrava employs online training to fine tune the super resolution models to adapt to varying video contents.

1) Workflow: The SR module in Scrava works as follows. In the initialization phase, when a new camera connects to Scrava, it first sends a message to Scrava to indicate its arrival, and Scrava then adds it into the camera list. After that, the camera would send high-resolution video segments to Scrava for initialize SR model parameters.

In the online training phase, Scrava tracks the F1 score of computer vision tasks on high-resolution video frames generated by SR models, then decides whether starts the online training phase.

2) Initialization Phase: Videos from different cameras have different characteristics, since they are deployed at different locations and with various angles. Therefore, one super resolution model for all cameras probably lead to a low accuracy in re-identification or object detection. We need to train separate SR model for each camera, so as to improve the quality of super resolution, which finally translates into a high accuracy.

When a new camera connects to Scrava, it first sends a message to Scrava to indicate its arrival, and Scrava then adds it into the camera list. After that, the camera would send high-resolution video segments to Scrava for initialize SR model parameters. Here are a few design considerations.

Firstly, when transmitting high-resolution video segments, Scrava chooses to transmit several video frames, instead of just regions of interest (RoIs) of several frames. Although it saves bandwidth to transmit RoIs, we find the backgrounds in video frames are important for a SR model to achieve good performance. The main reason is that, RoIs represent only a small part of a frame, lacking enough details for a SR model to reconstruct high-resolution frames.

Secondly, a video segment is set to 2 seconds in Scrava. Due to the similarity between consecutive video frames from a camera, consecutive video frames can form a group of frames that can be encoded with a smaller size. Thus, when the length of a video segment increases, we can save bandwidth by removing redundancies between frames; however, if the segment length is set to a too large value, then it would lead to an unacceptable latency in obtaining analytics results. Scrava makes a trade-off between realtime and bandwidth consumption, and sets the segment length to 2 seconds, which is proven to be good in our extensive experiments.

3) Online Training Phase: Scrava should adapt to varying video contents. To achieve this, Scrava tracks the F1 score of computer vision tasks on high-resolution video frames generated by SR models, then decides whether starts the online training phase. There are several design considerations.

Firstly, how often does a camera transmit original high-resolution video segments to Scrava? Scrava needs the accuracy of computer vision tasks on the original high-resolution video segments as the ground-truth of comparison. When a camera transmits original high-resolution video segments at

a high frequency, Scrava can detect video content change more quickly, however, these original high-resolution video segments may occupy too much bandwidth. On the other side, when a camera transmits original high-resolution video segments at a low frequency, bandwidth is saved, but Scrava may use a out-dated SR model for a long time since it cannot detect content changes in time. In Scrava, a camera sends an original high-resolution video segment every 120 video segments, which only leads to 2.6-5.7% more traffic compared with just transmitting low-quality video.

Secondly, when should Scrava start the online training phase? Every 120 video segments, Scrava receives an original high-resolution video segment from each camera. Then, for each camera, Scrava can compare the analytics results obtained from high-resolution video frames generated by SR models and original high-resolution video frames, after which Scrava computes the F1 score of the current SR model of that camera. If the difference between current score and the exponentially weighted moving average (EWMA) of previous scores exceeds a threshold, Scrava considers that the video content of this camera changes significantly and this camera requires the online training phase to adapt to the content change. Note that, historical frames probably help enhance SR models. Instead of using historical frames, Scrava chooses to utilize current and future frames to adapt to varying video contents.

Thirdly, when should the online training phase terminate? As we know, as online training phase proceeds, the marginal gain of online training would decrease. Hence, Scrava can monitor the marginal gain and terminates the training phase if the gain is below a threshold. Besides, if the online training phase takes a too long time, it may overfit over a stale video. To avoid this, Scrava also sets a maximum training steps for each online training phase, that is, Scrava would terminate an online training phase if the marginal gain is below a threshold or the training steps reaches the maximum steps.

C. Object Detection/Tracking Module

Object detection as well as object tracking, both of which are popular tasks in computer vision. Among them, object detection is to detect the position and category of one or more objects from an image, and many deep learning models for target detection exist, such as Faster R-CNN [28], YOLO [27], and SSD [23], which can accurately locate object positions and give the category they belong to. Object tracking refers to tracking one or more objects in a video sequence and continuously updating the position of the objects throughout the video. Since object tracking usually requires detecting objects in each frame in order to track them, object detection is the basis of object tracking and is an important part of the object tracking process. At the same time, the result of object detection can also be used for initializing object tracking to obtain more stable and accurate tracking results.

Scrava contains an integrated object detection/tracking module. When a video segment arrives, the server extracts frames from the video and applies object detection to them, and performs object tracking based on the results of object detection. It is worth noting that the input to the existing object tracking algorithms is typically a complete video with the same resolution of each frame in the video. However, in real-time video stream analytic applications, the video stream transmitted from a camera to the edge server is probably composed of video segments of short duration (e.g., 2∼4 seconds), and the resolution of video segments from the same camera may change over time due to network fluctuations, and the coordinates of an object may also change with the resolution of the video segment. In order to adapt the existing object tracking algorithm to the scenario of real-time video stream analytics, Scrava modifies the existing object tracking algorithm by adjusting all video segments from the same camera to a uniform size, which is the highest resolution from that camera. When a video is converted from a low resolution to a high resolution, interpolation introduces a lot of noise that may lead to a decrease in the accuracy of the subsequent analytics, and the super resolution module mentioned in Section IV-B can be optimized in this step to improve the accuracy of subsequent video streaming analytic tasks.

Algorithm 1: Re-identification Algorithm

Input: candidate object $o \in G$,
feature representation of current target q: $\mathbf{F_q}$,
all instances identified as q: Q
Output: True or False: indicating whether object o is identified as an instance of q

```
result ← False;
f_o ← Obtain the feature representation of the candidate object o with the feature extractor;
for f_i ∈ F_q do
    distance ← dist(f_o, f_i);
    if distance < match_threshold then
        result ← True;
        Q ← Q ⋃ {f_o};
        F_q ← Uniformly sampled from Q;
        break;
return result;
```

D. Re-Identification Module

Scrava aims to achieve real-time object tracking across cameras, while the object detection/tracking module mentioned in Section IV-C is performed on the video stream of a single camera. In order to achieve real-time tracking across cameras, we need to correlate the video stream information from multiple cameras. The function of this module is to correlate the information between multiple cameras to achieve cross-camera real-time object tracking.

In order to achieve fast and accurate re-identification, feature extraction of the tracking object q and its candidate match objects in the search space is required. DNNs are capable of automatically learning feature representations of images from large amounts of data and extracting high-level semantic features. CNN (Convolutional Neural Network), one of the

most widely used deep learning models in image processing, is capable of extracting feature information at different levels layer by layer through a hierarchical structure of multiple convolutional and pooling layers. Therefore, CNNs are used for feature extraction in the proposed Scrava.

Recall that, typical re-identification approaches extract the features of an object q using a DNN for a given query, and all images in a given image set G (i.e., gallery set) are ranked according to their feature distances from q, with a smaller distance representing more similar to q.

In our module, by feature extraction, object q can be represented as a vector $\mathbf{f_q}$ consisting of floating point numbers. We adopt the Euclidean distance to measure the variability between features. For example, the distance between $\mathbf{f_q}$ and $\mathbf{f_i}$ is:

$$\text{dist}(\boldsymbol{f_q}, \boldsymbol{f_i}) = \sqrt{\sum_{j=1}^{|\boldsymbol{f_q}|}(f_{qj} - f_{ij})^2}, \tag{6}$$

in which f_{qj} denotes the j-th dimension of vector $\mathbf{f_q}$. If $\text{dist}(\boldsymbol{f_q}, \boldsymbol{f_i})$ is less than a threshold, then we can say q is an instance of i.

However, as re-identification goes on, the number of instances of object q increases. For example, as one person walks within the coverage of a camera, his or her poses may change significantly, leading to varying features. If we only keep one feature for an object, then the re-identification algorithm may miss many true instances of it. Therefore, we should keep more than one feature for an object. Specifically, we extend the feature $\mathbf{f_q}$ of object q to $\mathbf{F_q}$, which is

$$\mathbf{F_q} = \{\mathbf{f_{q1}}, \mathbf{f_{q2}}, ..., \mathbf{f_{qL}}\}, \tag{7}$$

in which L is a hyper-parameter controlling the cardinality of $\mathbf{F_q}$. When L increases, $\mathbf{F_q}$ becomes more likely to cover all possible features of q; however, it also leads to more cost when matching objects.

Alg. 1 shows the re-identification algorithm used in Scrava. It maintains $\mathbf{F_q}$ for an object q. For a new instance o emerges, if the distance between o and any feature in $\mathbf{F_q}$ is less than a threshold, object o is then identified as an instance of q. That is, for those candidate match objects whose distance from object q is less than a threshold $match\_threshold$, we consider it as the same entity with object q and add it to the returned results of Scrava, and use it to update the feature representation of object q.

E. Bandwidth Prediction Module

Since the bandwidth prediction module is deployed on each camera, we choose a lightweight design considering the weak computational capability of each camera. On a specific camera c, a video segment i of size $S_{c,i}$ is transmitted; after this video segment is transmitted, the delay $L_{c,i}$ could be obtained; then, the available bandwidth, $B_{c,i}$, between the camera c and the server at the time of transmitting the video segment i is estimated by

$$B_{c,i} = \frac{S_{c,i}}{L_{c,i}}. \tag{8}$$

On a camera side, consecutive video segments are also close in time, and we assume a certain continuity of bandwidth when transmitting adjacent video segments. For a given sliding window size N, the available bandwidth, $B'_{c,i+1}$, when transmitting video segment $i+1$ on camera c is estimated by the following equation:

$$B'_{c,i+1} = \frac{1}{N}\sum_{j=i-N+1}^{i} B_{c,j}. \tag{9}$$

V. Evaluation

The goal of this section is to validate the proposed and implemented system through extensive experiments. We want to evaluate the effectiveness of super resolution in cross-camera real-time video stream analytics, using cross-camera pedestrian tracking as an example.

A. Implementation

We implemented the prototype of the cross-camera real-time video analytic system, Scrava, mainly using Python 3.8 under Ubuntu 18.04. We used Pytorch [26] as the deep learning framework supporting super resolution, object detection/tracking and re-identification.

Video transmission. To implement video transmission between multiple cameras and the server, Scrava uses Flask [10] framework as the data transmission framework between the cameras and the server, and WonderShaper [2] to simulate different network bandwidth traces.

DNN model. We use the Pytorch framework to implement DNNs for super resolution, object detection/tracking, and re-identification. Super resolution DNNs for each camera are trained from randomly initialized super resolution DNNs, and the training and validation dataset are obtained from historical video data of that camera. Note that, for the object detection/tracking and re-identification modules, we use models pre-trained on a large training set, since our work focuses on improving cross-camera real-time video streaming analytic applications in the presence of network limitations, rather than application-specific DNN performance.

Video encoding and decoding. We use the OpenCV library [3] to implement video compression, decompression, and other image transformation operations.

B. Experimental Setup

Our prototype system consists of two parts: the server and multiple clients (cameras). The cameras are responsible for encoding the video to the corresponding resolution according to the bandwidth prediction module and transmitting it to the server; the server is responsible for most of the computational work, including the training of super resolution DNNs, super resolution of images (videos), object detection/tracking, and re-identification. We use Nvidia RTX 3090 to support deep learning related computations. In our experiments, the offline training contains 15,000 epochs and lasts roughly 30 minutes on Nvidia RTX 3090. The training data is the selected frames

Fig. 4: Samples from the VIPeR dataset [9]

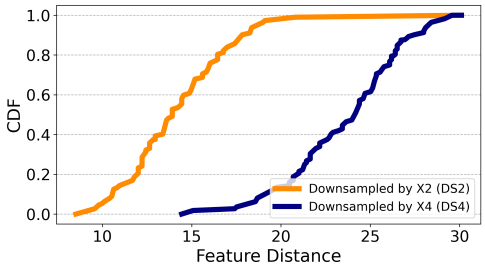


Fig. 5: CDF of the feature distance between instances of the same entity within camera a

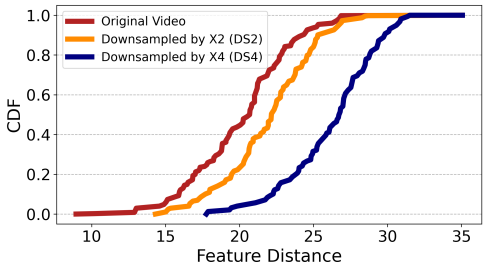


Fig. 6: CDF of the feature distance between instances of the same entity across cameras a and b

from the first 5 minutes of a video stream. The maximum step is set to 15,000 to 25,000.

For each camera, the server assigns a dedicated tracker to it. To achieve object tracking within a single camera, we use YOLOv8 [4] for object detection, ByteTrack [36] for object tracking and EDSR [22] for video super resolution, just the same as in Section III-B. To extract the features of a object (e.g., a pedestrian), we use ResNet50 [13] as a feature extractor, which is able to convert the object image into a vector of length 2,048.

We used the VIPeR [9] and EPFL datasets [8] mentioned in Section III-B to verify the effectiveness of super resolution in cross-camera pedestrian tracking. In the VIPeR dataset, each image contains only one person, thus it does not need detection and tracking. Each image is 128x48, and it needs 12 - 16 ms and about 10 ms to perform EDSR X4 and re-identification, respectively. Each image from the EPFL dataset may contain more than one person. Performing detection, tracking, super resolution, and re-identification on an image from EPFL needs 22-34ms, 10-12ms, 20ms, and 12ms, respectively.

In our experiments, the offline training contains 15,000 epochs and lasts roughly 30 minutes on Nvidia RTX 3090. As we mentioned in previous sections, the training data is the selected frames from the first 5 minutes of a video stream. The maximum step is set to 15,000 to 25,000.

C. Results on Cross-Camera Pedestrian Re-Identification

Many datasets have been collected for pedestrian re-identification, such as VIPeR [9], CUHK03 [20], Market-1501 [37]. To validate the effectiveness of super resolution on cross-camera pedestrian re-identification, we performed validation experiments on the VIPeR dataset. VIPeR consists of 632 pairs of the same pedestrian, with two images in each example pair from two different cameras (noted as cameras a and b). Different images of the same pedestrian have relatively large variations in viewpoint, pose, and illumination. Figure 4 gives an example of the images in VIPeR.

The image resolution reduction blurs the features of the objects, resulting in larger feature distances between different instances belonging to the same entity, making re-identification more difficult. We selected 100 example pairs from VIPeR and verified the effect of resolution reduction on the feature distance between instances of the same entity in both single-camera and cross-camera scenarios, respectively, and the results are shown in Figures 5 and 6.

Figure 5 shows the distance between the same instance with different resolutions in the same camera, where DS2 and DS4 represent the case where the images in camera a are downsampled by a factor of 2 and 4, respectively. We calculate the distance of downsampled feature with the original image features from camera a. It can be seen that the more the resolution is reduced, the greater the distance between the same instance. Figure 6, on the other hand, shows the effect of resolution reduction on different instance features of the same entity across cameras, in which ori represents the original images from cameras a and b, and DS2 and DS4 represent the cases that images from camera b are downsampled by a factor of 2 and 4, respectively, which shares a similar trend with Figure 5.

Figures 5 and 6 illustrate the loss of information due to res-

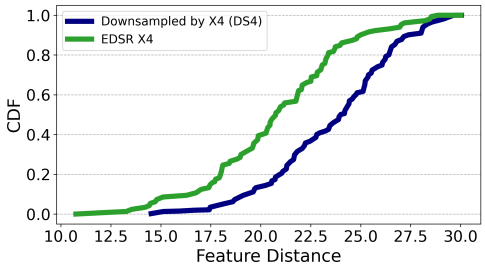


Fig. 7: Enhancement of super resolution on pedestrian re-identification within camera a

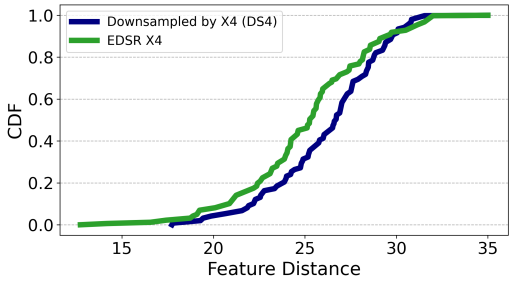
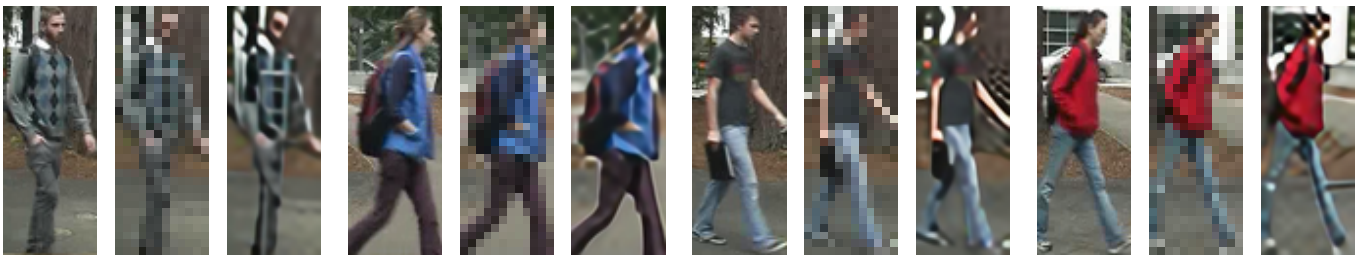


Fig. 8: Enhancement of super resolution on pedestrian re-identification across cameras a and b



(a) Original (b) DS X4 (c)EDSR X4 (d) Original (e) DS X4 (f)EDSR X4 (g) Original (h) DS X4 (i)EDSR X4 (j) Original (k) DS X4 (l)EDSR X4

Fig. 9: Visualization samples of super resolution enhancement to low-resolution pedestrian images from the VIPeR dataset, in which "DS X4" denotes downsampled by a factor of 4

olution reduction, which can make pedestrian re-identification more difficult. Super resolution is a common method for low-resolution image enhancement. Taking the scene in DS4 as an example, we use EDSR to reconstruct the downsampled images in camera a and b, and then compare their distances with the original images in camera a. The obtained results are displayed in Figures 7 and 8. We can see that in most cases, super resolution can reduce the distance between different instances of images belonging to the same entity, making their features more similar and thus enhancing the pedestrian re-identification application. On average, super resolution reduces the feature distance in the single-camera and cross-camera scenarios by 2.7767 and 1.1014, respectively, compared with the low-resolution images.

Furthermore, Figure 9 gives four visualization samples of super resolution enhancement to low-resolution pedestrian image, in which "DS X4" denotes an image downsampled by a factor of 4. Intuitively, compared with downsampled images, upsampled images (by EDSR X4) have clearer contours and richer details. These results explain the source of the improved effectiveness of the super resolution for the pedestrian re-identification module from another perspective.

D. Results on Cross-Camera Pedestrian Tracking

We have verified the potential gain of super resolution on single-camera and cross-camera pedestrian re-identification in the presence of reduced resolution. However, in order to evaluate the effect of super resolution more comprehensively, this subsection validates its enhancement effect on the cross-camera pedestrian tracking, an end-to-end cross-camera application.

The dataset used for validation is the EPFL dataset, laboratory sequence mentioned above. In this subsection, it is assumed that the tracking always starts from camera 1, and the four cameras synchronize their video streams to the server, which receives video segments and runs cross-camera pedestrian tracking. Besides, we assume that there is ample bandwidth between camera 1 and the server to transmit the raw resolution video; the remaining three cameras are bandwidth-constrained and thus transmit low-resolution video segments (downsampled by a factor of 4), and the server applies super resolution to these low-resolution videos for reconstruction. Since the original dataset did not provide full annotation, we manually selected and annotated several tracks to verify the enhancement brought by super resolution.

Super resolution enhancement. The ultimate goal of the application of cross-camera pedestrian tracking is to return the frames and annotations containing the target pedestrians in all cameras, so in this subsection we set accuracy, recall, and the corresponding F1 scores as evaluation metrics. Table IV demonstrates the improvement in video analytic accuracy after super resolution compared to low-resolution videos. As shown in Table IV, the use of super resolution is effective in improving the score of cross-camera pedestrian tracking (13.59-73.17%) in all trajectories and video streams, and the recall improves in all cases, which is pivotal in some significant tasks (e.g., Amber Alert).

It is worth mentioning that, the precision metric decreases in some cases in Table IV. The main reason is that, a super resolution model is actually predicting pixels for each input image, thus super resolution may improve the details of an object of interest as well as introduce additional noises. For example, if the quality of an input image is too low for even

TABLE IV: Super resolution enhancements to cross-camera pedestrian tracking

	Camera ID	Precision	Recall	F1 score
Trajectory 1	Camera 2	36.36%↑	19.05%↑	25.01%↑
	Camera 3	57.14%↓	9.68%↑	13.59%↑
	Camera 4	30%↓	20.69%↑	29.22%↑
Trajectory 2	Camera 2	100%↑	57.28%↑	72.84%↑
	Camera 3	2.9%↓	65.86%↑	61.38%↑
	Camera 4	12.79%↓	61.65%↑	58.54%↑
Trajectory 3	Camera 2	87.5%↑	29.17%↑	43.75%↑
	Camera 3	0%	33.33%↑	46.92%↑
	Camera 4	100%↑	57.69%↑	73.17%↑

humans cannot identify an object in it, then super resolution probably introduces more noises than details. In such case, performing detection on the original image may be better than on the image after super resolution.

Bandwidth consumption. Comparing the case where all four cameras transmit the original resolution video stream, the reduction of bandwidth consumption in our experiments is shown in Figure 10, where the cx-yp besides x-axis represents video streams from the y people scenario in camera x. The results show that transmitting low-resolution video significantly reduces bandwidth consumption compared to transmitting the original video stream, reducing bandwidth overhead by 91.34% on average, which significantly facilitates the scalability of the camera network. Overall, the experimental results show that super resolution can effectively reduce the information loss caused by the resolution reduction.

VI. Discussions

In this section, we discuss several potential limitations and future research directions.

Inter-frame and inner-frame encoding. Varying and limited bandwidth between cameras and edge servers is usually the bottleneck of performance. To decrease the transmission latency of video streams, several studies proposed to eliminate redundancy between frames (e.g., Reducto [21]) as well as within a frame (e.g., DDS [6]). For the former, frame filtering is often used to filter out frames that contribute little useful information, while for the latter, regions of interest (RoIs) within a frame are cropped out to send to edge servers. Both approaches are effective especially when the bandwidth is the bottleneck while the computing resources are adequate, and they are orthotropic to our work Scrava.

Cross-camera contents for enhancement. Scrava only uses video frames within a camera to online train a SR model for the camera itself, and does not make full use of the video frames from other cameras. However, these are several challenges to achieve this. Firstly, how to select video frames from other cameras for online training? Using all video frames from other cameras is not practical or efficient, since most frames from other cameras may contain no similar objects/views and provide no benefit to online training. Secondly, how to efficiently share selected video frames among SR modules in Scrava? Thirdly, if cameras can share video frames for online training, then could them share SR models and fine-tune them? Incorporating cross-camera contents for enhancement is left as future work.

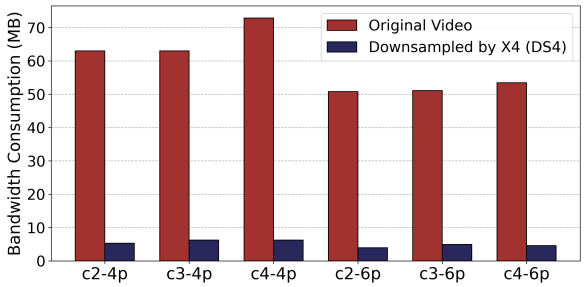


Fig. 10: Bandwidth consumption (the cx-yp besides x-axis represents video streams from the y people scenario in camera x from the EPFL dataset)

Instance-specific Scaling Factor. Adapting to varying video contents is important and helpful to achieving better video analytics performance, as we have seen in lots of prior studies. For example, Palleon [7] investigated adaptive model selection to cope with dynamic class skew; RECL [16] shared across edge devices a model zoo that comprises expert models previously trained for all edge devices, enabling history model reuse across video sessions. Inspired by them, we are going to incorporating instance-specific scaling factors into Scrava in future work.

Scrava in practice. Most of the modules in Scrava run on the server-side, except the bandwidth prediction module. Note that, the bandwidth prediction module can be implemented as an overlay on the default bandwidth interface. This approach offers several advantages. Firstly, Scrava requires little changes on the client side, making it readily deployable. Secondly, Scrava allows client-defined bandwidth prediction methods, in case that a client has prior knowledge of the network conditions.

VII. Conclusion

In camera networks, the simultaneous transmission of video streams from a large number of cameras to an edge server imposes stringent bandwidth requirements. To mitigate the impact of network fluctuations on camera video resolution and to improve the accuracy of cross-camera video analytic applications, this paper proposes and implements a cross-camera real-time video stream analytic system, Scrava. Scrava exploits super resolution to optimize the cross-camera video analytic pipeline and verifies the enhancement effect of super resolution in cross-camera video analytic applications through simulation experiments. This paper concludes that super resolution can mitigate the negative impact of resolution degradation under poor network conditions, thus enhancing the performance of cross-camera video stream analytics.

Acknowledgments

We thank the editor and anonymous reviewers. This work was supported in part by NSFC (62202233), Double Innovation Plan of Jiangsu Province (JSSCBS20220409), Grant from State Key

Laboratory for Novel Software Technology, Nanjing University (KFKT2024B18), Nanjing Key S&T Special Projects (202309006), and Collaborative Innovation Center of Novel Software Technology and Industrialization.

REFERENCES

[1] N. Ahn, B. Kang, and K.-A. Sohn, "Fast, accurate, and lightweight super-resolution with cascading residual network," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 252–268.

[2] H. Bert, G. Jacco, and S. Simon. (2023, Mar.) The wonder shaper 1.4.1. [Online]. Available: https://github.com/magnific0/wondershaper

[3] G. Bradski, "The opencv library." *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, vol. 25, no. 11, pp. 120–123, 2000.

[4] M. Broström, "Real-time multi-object tracking and segmentation using yolov8 with strongsort and osnet." [Online]. Available: https://github.com/mikel-brostrom/yolov8_tracking

[5] T. Y.-H. Chen, L. Ravindranath, S. Deng, P. Bahl, and H. Balakrishnan, "Glimpse: Continuous, real-time object recognition on mobile devices," in *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, 2015, pp. 155–168.

[6] K. Du, A. Pervaiz, X. Yuan, A. Chowdhery, and J. Jiang, "Server-driven video streaming for deep learning inference," in *SIGCOMM '20: Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, 2020.

[7] B. Feng, Y. Wang, G. Li, Y. Xie, and Y. Ding, "Palleon: A runtime system for efficient video processing toward dynamic class skew," in *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, 2021, pp. 427–441.

[8] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua, "Multicamera people tracking with a probabilistic occupancy map," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 267–282, 2007.

[9] D. Gray, S. Brennan, and H. Tao, "Evaluating appearance models for recognition, reacquisition, and tracking," in *Proc. IEEE international workshop on performance evaluation for tracking and surveillance (PETS)*, vol. 3, 2007, pp. 1–7.

[10] M. Grinberg, *Flask web development: developing web applications with python*. O'Reilly Media, Inc., 2018.

[11] H. Guo, S. Yao, Z. Yang, Q. Zhou, and K. Nahrstedt, "Crossroi: cross-camera region of interest optimization for efficient real time video analytics at scale," in *Proceedings of the 12th ACM Multimedia Systems Conference*, 2021, pp. 186–199.

[12] M. Hanyao, Y. Jin, Z. Qian, S. Zhang, and S. Lu, "Edge-assisted online on-device object detection for real-time video analytics," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[14] S. Jain, X. Zhang, Y. Zhou, G. Ananthanarayanan, J. Jiang, Y. Shu, P. Bahl, and J. Gonzalez, "Spatula: Efficient cross-camera video analytics on large camera networks," in *2020 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 2020, pp. 110–124.

[15] J. Jiang, G. Ananthanarayanan, P. Bodik, S. Sen, and I. Stoica, "Chameleon: scalable adaptation of video analytics," in *SIGCOMM '18: Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, 2018, pp. 253–266.

[16] M. Khani, G. Ananthanarayanan, K. Hsieh, J. Jiang, R. Netravali, Y. Shu, M. Alizadeh, and V. Bahl, "RECL: Responsive resource-efficient continuous learning for video analytics," in *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, 2023, pp. 917–932.

[17] J. Li, L. Liu, H. Xu, S. Wu, and C. J. Xue, "Cross-camera inference on the constrained edge," in *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*. IEEE, 2023, pp. 1–10.

[18] J. Li, F. Fang, K. Mei, and G. Zhang, "Multi-scale residual network for image super-resolution," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 517–532.

[19] P. Li, G. Li, Z. Yan, Y. Li, M. Lu, P. Xu, Y. Gu, B. Bai, Y. Zhang, and D. Chuxing, "Spatio-temporal consistency and hierarchical matching for multi-target multi-camera vehicle tracking." in *CVPR Workshops*, 2019, pp. 222–230.

[20] W. Li, R. Zhao, T. Xiao, and X. Wang, "Deepreid: Deep filter pairing neural network for person re-identification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 152–159.

[21] Y. Li, A. Padmanabhan, P. Zhao, Y. Wang, and R. Netravali, "Reducto: On-camera filtering for resource-efficient real-time video analytics," in *SIGCOMM '20: Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, 2020.

[22] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, "Enhanced deep residual networks for single image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 136–144.

[23] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 2016, pp. 21–37.

[24] X. Liu, P. Ghosh, O. Ulutan, B. Manjunath, K. Chan, and R. Govindan, "Caesar: cross-camera complex activity recognition," in *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*, 2019, pp. 232–244.

[25] H. B. Pasandi and T. Nadeem, "Convince: Collaborative cross-camera video analytics at the edge," in *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 2020, pp. 1–5.

[26] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[28] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[29] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking," in *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part II*. Springer, 2016, pp. 17–35.

[30] X. Wang, "Intelligent multi-camera video surveillance: A review," *Pattern recognition letters*, vol. 34, no. 1, pp. 3–19, 2013.

[31] M. Ye, J. Shen, G. Lin, T. Xiang, L. Shao, and S. C. Hoi, "Deep learning for person re-identification: A survey and outlook," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 6, pp. 2872–2893, 2021.

[32] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Deep metric learning for person re-identification," in *2014 22nd international conference on pattern recognition*. IEEE, 2014, pp. 34–39.

[33] S. Zhang, C. Wang, Y. Jin, J. Wu, Z. Qian, M. Xiao, and S. Lu, "Adaptive configuration selection and bandwidth allocation for edge-based video analytics," *IEEE/ACM Transactions on Networking*, vol. 30, no. 1, pp. 285–298, 2022.

[34] T. Zhang, A. Chowdhery, P. Bahl, K. Jamieson, and S. Banerjee, "The design and implementation of a wireless video surveillance system," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, 2015, pp. 426–438.

[35] W. Zhang, Z. He, L. Liu, Z. Jia, and Y. Zhang, "Elf: Accelerate high-resolution mobile deep vision with content-aware parallel offloading," in *The 27th Annual International Conference On Mobile Computing And Networking (MobiCom)*, 2021.

[36] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "Bytetrack: Multi-object tracking by associating every detection box," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXII*. Springer, 2022, pp. 1–21.

[37] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, "Scalable person re-identification: A benchmark," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1116–1124.

[38] L. Zheng, Y. Yang, and A. G. Hauptmann, "Person re-identification: Past, present and future," *arXiv preprint arXiv:1610.02984*, 2016.

Yu Liang is a lecturer at Nanjing Normal University. She received the MS and PhD degrees from Nanjing University in 2011 and 2021, respectively. She was a senior software engineer in Trend Micro China Development Center between 2011 and 2017. Her research interests include edge intelligence and edge computing. Her publications include those appeared in TMC, TPDS, TON, Computer Networks, Computer Communications, IEEE ICDCS, IEEE MSN, and IEEE Globecom.

Sheng Zhang is an associate professor with Nanjing University, and a member of the State Key Lab. for Novel Software Technology. He received the BS and PhD degrees from Nanjing University. His current research interests include edge computing and edge intelligence. He regularly publishes in scholarly journals and conference proceedings, such as *JSAC*, *TMC*, *TON*, *TPDS*, *TC*, *MobiHoc*, *ICDCS*, *ICDE*, and *INFOCOM*. He is the recipient of CCFSys Best Paper Award (2023), IEEE ICPADS Outstanding Paper Runner-Up Award (2021), IEEE ICCCN Best Paper Award (2020), IEEE MASS Best Paper Runner-Up Award (2012). He is a senior member of IEEE.

Jie Wu (F'09) is the Director of the Center for Networked Computing and Laura H. Carnell professor at Temple University. He also serves as the Director of International Affairs at College of Science and Technology. He served as Chair of Department of Computer and Information Sciences from the summer of 2009 to the summer of 2016 and Associate Vice Provost for International Affairs from the fall of 2015 to the summer of 2017. Prior to joining Temple University, he was a program director at the National Science Foundation and was a distinguished professor at Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. Dr. Wu regularly publishes in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including IEEE Transactions on Mobile Computing, IEEE Transactions on Service Computing, Journal of Parallel and Distributed Computing, and Journal of Computer Science and Technology. Dr. Wu was general co-chair for IEEE MASS 2006, IEEE IPDPS 2008, IEEE ICDCS 2013, ACM MobiHoc 2014, ICPP 2016, and IEEE CNS 2016, as well as program co-chair for IEEE INFOCOM 2011 and CCF CNCC 2013. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). Dr. Wu is a CCF Distinguished Speaker and a Fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.