Scaling Video Analytics Systems to Large Camera Deployments

Samvit Jain[†*], Ganesh Ananthanarayanan[†], Junchen Jiang[¶†], Yuanchao Shu[†], Joseph Gonzalez[*]
[†]Microsoft Research, [*]University of California at Berkeley, [¶]University of Chicago

ABSTRACT

Driven by advances in computer vision and the falling costs of camera hardware, organizations are deploying video cameras *en masse* for the spatial monitoring of their physical premises. Scaling video analytics to massive camera deployments, however, presents a new and mounting challenge, as compute cost grows proportionally to the number of camera feeds. This paper is driven by a simple question: can we scale video analytics in such a way that *cost grows sublinearly*, or even remains constant, as we deploy more cameras, while *inference accuracy remains stable*, or even improves. We believe the answer is yes. Our key observation is that video feeds from wide-area camera deployments demonstrate significant content correlations (e.g. to other geographically proximate feeds), both in space and over time. These *spatio-temporal correlations* can be harnessed to dramatically reduce the size of the inference search space, decreasing both workload and false positive rates in multi-camera video analytics. By discussing use-cases and technical challenges, we propose a roadmap for scaling video analytics to large camera networks, and outline a plan for its realization.

CCS CONCEPTS

• **Computer systems organization** → *Distributed architectures*; • **Networks** → *Network algorithms*; • **Information systems** → *Data analytics*; • **Computing methodologies** → *Computer vision tasks.*

KEYWORDS

video analytics; spatio-temporal correlations; neural networks; edge computing; streaming video

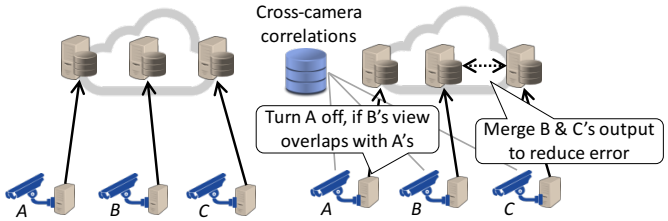
ACM Reference Format:
Samvit Jain, Ganesh Ananthanarayanan, Junchen Jiang, Yuanchao Shu, Joseph Gonzalez. 2019. Scaling Video Analytics Systems to Large Camera Deployments. In *The 20th International Workshop on Mobile Computing Systems and Applications (HotMobile '19), February 27–28, 2019, Santa Cruz, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3301293.3302366

HotMobile '19, February 27–28, 2019, Santa Cruz, CA, USA

ACM ISBN 978-1-4503-6273-3/19/02…$15.00
https://doi.org/10.1145/3301293.3302366

1 INTRODUCTION

Driven by plummeting camera prices and the recent successes of computer vision-based video inference, organizations are deploying cameras at scale for applications ranging from surveillance and flow control to retail planning and sports broadcasting. Processing video feeds from large deployments, however, requires a proportional investment in either compute hardware (i.e. expensive GPUs) or cloud resources (i.e. GPU machine time), costs from which easily exceed that of the camera hardware itself [1, 25]. A key reason for these large resource requirements is the fact that, today, video streams are analyzed *in isolation*. As a result, the compute required to process the video grows linearly with the number of cameras. We believe there is an opportunity to both stem this trend of linearly increasing costs, *and* improve accuracy, by viewing the cameras *collectively*.



(a) Current video analytics system (b) Example gains by leveraging cross-camera correlations

Figure 1: ***Contrasting (a) the traditional per-camera video analytics with (b) the proposed approach that leverages cross-camera correlations.***

This position paper is based on a simple observation—cameras deployed over any geographic area, whether a large campus, an outdoor park, or a subway station network, demonstrate significant content correlations—both spatial and temporal. For example, nearby cameras may perceive the same objects, though from different angles or at different points in time. We argue that these cross-camera correlations can be harnessed, so as to use *substantially fewer resources* and/or achieve *higher inference accuracy* than a system that runs complex inference on all video feeds independently. For example, if a query person is identified in one camera feed, we can then exclude the possibility of the individual appearing in a distant camera within a short time period. This eliminates extraneous processing and reduces the rate of false positive detections (Figure 1(a)). Similarly, one can improve accuracy by combining the inference results of multiple cameras that monitor the same objects from different angles (Figure 1(b)). Our initial evaluation on a real-world dataset with eight cameras shows that using cameras collectively can yield resource savings of *at least 74%*. More such opportunities are outlined in §3.

Given the recent increase in interest in systems infrastructure for video analytics, we believe the important next step for the community is designing a software stack for *collective* camera analytics. Video processing systems today generally analyze video streams independently even while useful cross-camera correlations exist [19, 38]. On the computer vision side, recent work has tackled specific multi-camera applications (*e.g.* tracking [30, 35, 40]), but has generally neglected the growing cost of inference itself.

We argue that the key to scaling video analytics to large camera deployments lies in fully leveraging these latent cross-camera correlations. We identify several architectural aspects that are critical to improving resource efficiency and accuracy but are missing in current video analytics systems. First, we illustrate the need for a new system module that learns and maintains up-to-date *spatio-temporal correlations* across cameras. Second, we discuss online pipeline reconfiguration and composition, where video pipelines incorporate information from other correlated cameras (*e.g.* to eliminate redundant inference, or ensemble predictions) to save on cost or improve accuracy. Finally, to deal with any missed detections arising from our proposed optimizations, we note the need to process small segments of historical video at faster-than-real-time rates, alongside analytics on live video.

Our goal is not to provide a specific system implementation, but to motivate the design of an accurate, cost-efficient *multi-camera* video analytics system. Our hope is to inspire the practical realization of these ideas in the near future.

2 CAMERA TRENDS & APPLICATIONS

This section sets the context for using many cameras collaboratively by discussing (1) trends in camera deployments, and (2) the recent increased interest in cross-camera applications.

Explosive growth of smart camera installations: Organizations are deploying cameras *en masse* to cover physical areas. Enterprises are fitting cameras in office hallways, store aisles, and building entry/exit points; government agencies are deploying cameras outdoors for surveillance and planning. Two factors are contributing to this trend:

1. *Falling camera costs* enable more enterprises and business owners to install cameras, and at higher density. For instance, today, one can install an HDTV-quality camera with on-board SD card storage for $20, whereas three years ago the industry's first HDTV camera cost $1,500 [32]. Driven by the sharp drop in camera costs, camera installations have grown exponentially, with 566 PT of data generated by *new* video surveillance cameras worldwide *every day* in 2015, compared to 413 PT generated by newly installed cameras in 2013 [32].

   There has been a recent wave of interest in "AI cameras" – cameras with compute and storage on-board – that are designed for processing and storing the videos [4, 28]. These cameras are programmable and allow for running arbitrary deep learning models as well as classic computer vision algorithms. AI cameras are slated to be deployed at mass scales by enterprises.
2. *Advances in computer vision*, specifically in object detection and re-identification techniques [29, 40], have sparked renewed interest among organizations in camera-based data analytics. For example, transportation departments in the US are moving to use video analytics for traffic efficiency and planning [23]. A key advantage of using cameras is that they are relatively easy to deploy and can be purposed for multiple objectives.

Increased interest in cross-camera applications: We focus on applications that involve video analytics *across* cameras. While many cross-camera video applications were envisaged in prior research, the lack of one or both of the above trends made them either prohibitively expensive or insufficiently accurate for real-world use-cases.

We focus on a category of applications we refer to as *spotlight search*. Spotlight search refers to detecting a specific type of activity and object (*e.g.* shoplifting, a person), and then tracking the entity as it moves through the camera network. Both detecting activities/objects and tracking require compute-intensive techniques, *e.g.* face recognition and person re-identification [40]. Note that objects can be tracked both in the forward direction ("real-time tracking"), and in the backward direction ("investigative search") on recorded video. Spotlight search represents a broad template, or a core building block, for many cross-camera applications. Cameras in a retail store use spotlight search to monitor customers flagged for suspicious activity. Likewise, traffic cameras use spotlight search to track vehicles exhibiting erratic driving patterns. In this paper, we focus on spotlight search on live camera feeds as the canonical cross-camera application.

Metrics of interest: The two metrics of interest in video analytics applications are inference *accuracy* and *cost* of processing. Inference accuracy is a function of the model used for the analytics, the labeled data used for training, and video characteristics such as frame resolution and frame rate [18, 19, 38]. All of the above metrics also influence the *cost* of processing – larger models and higher quality videos enable higher accuracy, at the price of increased resource consumption or higher processing latency. When the video feeds are analyzed at an edge or cloud cluster, cost also includes the bandwidth cost of sending the videos over a wireless network, which increases with the number of video feeds.

3 NEW OPPORTUNITIES IN CAMERA DEPLOYMENTS

Next, we explain the key benefits – in efficiency and accuracy – of cross-camera video analytics. The key insight is that scaling video analytics to many cameras does not necessarily stipulate a linear increase in cost; instead, one can significantly improve cost-efficiency as well as accuracy by leveraging the spatio-temporal correlations across cameras.

3.1 Key enabler: Cross-camera correlations

A fundamental building block in enabling cross-camera collaboration are *spatio-temporal correlation* profiles across cameras. At a high level, these spatio-temporal correlations capture the relationship between the content of camera A and the content of camera B over a time delta Δt.[1] This correlation manifests itself in at least three different forms. Firstly, the same object can appear in multiple cameras, *i.e. content* correlation, at the same time (*e.g.* cameras in

[1] The correlation reduces to "spatial-only", when $\Delta t \rightarrow 0$.

the same room) or at different points in time (*e.g.* cameras placed at two ends of a hallway); Secondly, multiple cameras may share similar characteristics, *i.e. property* correlation, *e.g.* the types, velocities, and sizes of contained objects. Thirdly, one camera may have a different viewpoint on objects than another, resulting in a *position* correlation, *e.g.* some cameras see larger/clearer faces since they are deployed closer to eye level.

As we will show next, the prevalence of these cross-camera correlations in dense camera networks enables key opportunities to use the compute (CPU, GPU) and storage (RAM, SSD) resources on these cameras *collaboratively*, by leveraging their network connectivity.

3.2 Better cost efficiency

Leveraging cross-camera correlations improves the *cost efficiency* of multi-camera video analytics, without adversely impacting accuracy. Here are two examples.

C1: Eliminating redundant inference

In cross-camera applications like spotlight search, there are often far fewer objects of interest than cameras. Hence, ideally, query resource consumption over multiple cameras should not grow proportionally to the number of cameras. We envision two potential ways of doing this by leveraging content-level correlations across cameras (§3.1).

- When two spatially correlated cameras have overlapping views (*e.g.* cameras covering the same room or hallway), the overlapped region need only be analyzed once.
- When an object leaves a camera, only a small set of relevant cameras (*e.g.* cameras likely to see the object in the next few seconds), identified via their spatio-temporal correlation profiles, need search for the object.

In spotlight search, for example, once a suspicious activity or individual is detected, we can selectively trigger object detection or person re-identification models only on the cameras that the individual is likely to traverse. In other words, we can use spatio-temporal correlations to narrow the search space by *forecasting* the trajectory of objects.

We analyze the popular "DukeMTMC" video dataset [31], which contains footage from eight cameras on the Duke University campus. Figure 2 shows a map of the different cameras, along with the percentage of traffic leaving a particular camera i that next appears in another camera j. Figures are calculated based on manually annotated human identity labels. As an example observation, within a time window of 90 minutes, 89% of all traffic leaving Camera 1 first appears at Camera 2. At Camera 3, an equal percentage of traffic, about 45%, leaves for Cameras 2 and 4. Gains achieved by leveraging these spatial traffic patterns are discussed in §3.4.

C2: Resource pooling across cameras

Since objects/activities of interest are usually sparse, most cameras do not need to run analytics models all the time. This creates a substantial heterogeneity in workloads across different cameras. For instance, one camera may monitor a central hallway and detect many candidate persons, while another camera detects no people in the same time window.

Such workload heterogeneity provides an opportunity for dynamic offloading, in which more heavily utilized cameras transfer part of their analytics load to less-utilized cameras. For instance, a camera that runs complex per-frame inference can offload queries on some frames to other "idle" cameras whose video feeds are static. Figure 3 shows the evident imbalance in the number of people detected on two cameras across a 500 second interval. By *pooling* available resources and balancing the workload across multiple cameras, one can greatly reduce resource provisioning on each camera (*e.g.* deploy smaller, cheaper GPUs), from an allocation that would support peak workloads. Such a scheme could also reduce the need to stream compute tasks to the cloud, a capability constrained by available bandwidth and privacy concerns. *Content* correlations (§3.1) directly facilitate this offloading as they foretell query trajectories, and by extension, workload.

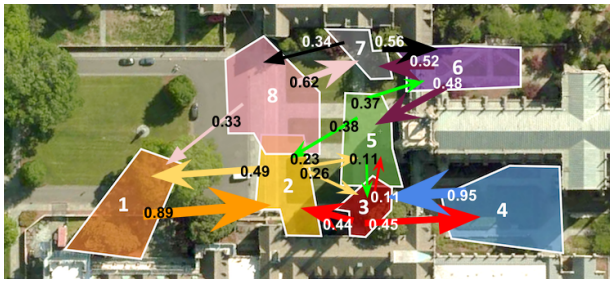


Figure 2: *Camera topology and traffic flow in the DukeMTMC dataset [31].*

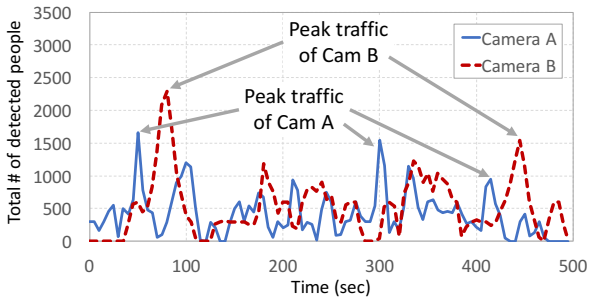


Figure 3: *Number of person detections on two cameras in [31] in each 5-second window over a span of 500 seconds. Note the heterogeneity in traffic patterns.*

3.3 Higher inference accuracy

We also observe opportunities to improve inference accuracy, without increasing resource usage.

A1: Collaborative inference

Using an ensemble of identical models to render a prediction is an established method for boosting inference accuracy [14]. The technique also applies to model ensembles consisting of multiple, correlated video pipelines (*e.g.* with different perspectives on an object). Inference can also benefit from hosting dissimilar models on different cameras. For instance, camera A with limited resources uses a specialized, low cost model for flagging cars, while camera B uses a general model for detection. Then camera A can offload its video to camera B to cross-validate its results when B is idle.

Table 1: *Spotlight search results for various levels of spatio-temporal correlation filtering. A filtering level of k% signifies that a camera must receive k% of the traffic from a particular source camera to be searched. Larger k (e.g. $k = 10$) corresponds to more aggressive filtering, while $k = 0$ corresponds to the baseline, which searches all of the cameras. Results aggregated over 100 tracking queries on the 8 camera DukeMTMC dataset.*

Filtering level (%)	Detections processed	Savings (vs. baseline)	Recall (%)	Precis. (%)
0%	76,510	0.0	**57.4**	60.6
1%	29,940	60.9	55.0	81.4
3%	22,490	70.6	55.1	**81.9**
10%	19,639	**74.3**	55.1	**81.9**

Cameras can also be correlated in a *mutually exclusive* manner. In spotlight search, for example, if a person p is identified in camera A, we can preclude a detection of the same p in another camera whose view does not overlap with A. Knowing where an object is likely *not* to show up can significantly improve tracking precision over a naïve baseline that searches all of the cameras. In particular, removing unlikely candidates from the space of potential matches reduces false positive matches, which tend to dislodge subsequent tracking and bring down precision (see §3.4).

A2: Cross-camera model refinement

One source of video analytics error stems from the fact that objects look differently in real-world settings than in training data. For example, some surveillance cameras are installed on ceilings, which reduces facial recognition accuracy, due to the oblique viewing angle [26]. These errors can be alleviated by retraining the analytics model, using the output of another camera that has an eye-level view as the "ground truth". As another example, traffic cameras under direct sunlight or strong shadows tend to render poorly exposed images, resulting in lower detection and classification accuracy [12], whereas cameras without lighting interference yield better inference performance. Since lighting conditions change over time, two such cameras can complement each other, via collaborative model training. Opportunities for such cross-camera model refinement are a direct implication of position correlations (§3.1) across cameras.

3.4 Preliminary results

Table 1 contains a preliminary evaluation of our spotlight search scheme on the Duke dataset [31], which consists of 8 cameras. We quantify resource savings by computing the ratio of 1) the number of person detections processed by the baseline (*i.e.* 76,510) to 2) the number of person detections processed by a particular filtering scheme (*e.g.* 22,490). Observe that applying spatio-temporal filtering results in significant resource savings and much higher precision, compared to the baseline, at the price of slightly lower recall.

4 ARCHITECTING FOR CROSS-CAMERA ANALYTICS

We have seen that exploiting spatio-temporal correlations across cameras can improve cost efficiency and inference accuracy in multi-camera settings. Realizing these benefits in practice, however, requires re-architecting the underlying video analytics stack. This section articulates the key missing pieces in current video analytics systems, and outlines the core technical challenges that must be addressed in order to realize the benefits of collaborative analytics.

4.1 What's missing in today's architecture?

The proposals in §3.2 and §3.3 require four basic capabilities.

#1: Cross-camera correlation database: First, a new system module must be introduced to learn and maintain an up-to-date view of the spatio-temporal correlations between any pair of cameras (§3.1). Physically, this module can be a centralized service, or a decentralized system (with each camera maintaining a local copy). Different correlations can be represented in various ways. For example, content correlations can be modeled as the *conditional probability* of detecting a specific object in camera B at time t, given its appearance at time $t - \Delta t$ in camera A, and stored as a discrete, 3-D matrix in a database. This database of cross-camera correlations must be dynamically updated, because the correlations between cameras can vary over time: video patterns can evolve, cameras can enter or leave the system, and camera positions and viewpoints can change. We discuss the intricacy of discovering these correlations, and the implementation of this new module, in §4.2.

#2: Peer-triggered inference: Today, the execution of a video analytics pipeline (what resources to use and which video to analyze) is largely pre-configured. To take advantage of cross-camera correlations, an analytics pipeline must be aware of the inference results of other relevant video streams, and support *peer-triggered inference* at runtime. Depending on the content of other related video streams, an analytics task can be assigned to the compute resources of *any* relevant camera to process *any* video stream at *any* time. This effectively *separates* the logical analytics pipeline from its execution. To eliminate redundant inference (C1 of §3.2), for instance, one video stream pipeline may need to dynamically trigger (or switch off) another video pipeline (Figure 4.c). Similarly, to pool resources across cameras (C2 of §3.2), a video stream may need to dynamically offload computation to another camera, depending on correlation-based workload projections (Figure 4.d). To trigger such inference, the current inference results need to be shared in real-time *between* pipelines. While prior work explores task offloading across cameras and between the edge and the cloud [9, 20], the trigger is usually workload changes on a single camera. We argue that such dynamic triggering must also consider events on the video streams of other, related cameras.

#3: Video pipeline composition: Analyzing each video stream in isolation also precludes learning from the content of other camera feeds. As we noted in §3.3, by combining the inference results of multiple correlated cameras, *i.e.* composing multiple video pipelines, one can significantly improve inference accuracy. Figure 4 shows two examples. Firstly, by sharing inference results across pipelines in real-time (Figure 4.e), one can correct the inference error of another less well-positioned camera (A1 in §3.3). Secondly, the inference model for one pipeline can be refined/retrained (Figure 4.f) based on the inference results of another better positioned camera (A2 in §3.3). Unlike the aforementioned reconfiguration of video

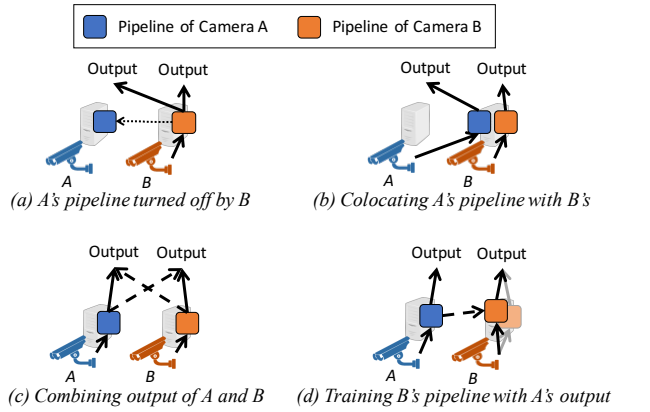


Figure 4: *Illustrative examples of peer-triggered inference (a, b) and pipeline composition (c, d).*

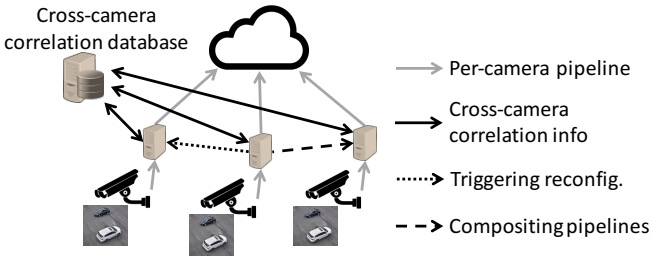


Figure 5: *Envisioned cross-camera architecture.*

pipelines, *merging* pipelines in this way actually impacts inference output.

#4: Fast analytics on stored video: Recall from §2 that spotlight search can involve tracking an object *backward* for short periods of time to its first appearance in the camera network. This requires a new feature, lacking in most video stream analytics systems: fast analysis of stored video data, *in parallel* with analytics on live video. Stored video must be processed with very low latency (*e.g.* several seconds), as subsequent tracking decisions depend on the results of the search. In particular, this introduces a new requirement: processing many seconds or minutes of stored video at *faster-than-real-time* rates.

Putting it all together: Figure 5 depicts a new video analytics system that incorporates these proposed changes, along with two new required interfaces. Firstly, the correlation database must expose an interface to the analytics pipelines that reveals the spatio-temporal correlation between any two cameras. Secondly, pipelines must support an interface for real-time communication, to (1) trigger inference (C1 in §3.2) and (2) share inference results (A1 and A2 in §3.3). This channel can be extended to support the sharing of resource availability (C2) and optimal configurations (C3).

4.2 Technical challenges

In this section, we highlight the technical challenges that must be resolved to fully leverage cross-camera correlations.

1) Learning cross-camera correlations: To enable multi-camera optimizations, cross-camera correlations need to be extracted in the first place. We envision two basic approaches. One is to rely on domain experts, *e.g.* system administrators or developers who deploy cameras and models. They can, for example, calibrate cameras to determine the overlapped field of view, based on camera locations and the floor plan. A data-driven approach is to *learn* the correlations from the inference results; *e.g.* if two cameras identify the same person in a short time interval, they exhibit a *content correlation*.

The two approaches represent a tradeoff— the data-driven approach can better adapt to dynamism in the network, but is more computationally expensive (*e.g.* it requires running an offline, multi-person tracker [30] on all video feeds to learn the correlations). A hybrid approach is also possible: let domain experts establish the initial correlation database, and dynamically update it by periodically running the tracker. This by itself is an interesting problem to pursue.

2) Resource management in camera clusters: Akin to clusters in the cloud, a set of cameras deployed by an enterprise also represents a "cluster" with compute capacity and network connectivity. Video analytics work must be assigned to the different cameras in proportion to their available resources, while also ensuring high utilization and overall performance. While cluster management frameworks [13] perform resource management, two differences stand out in our setting. Firstly, video analytics focuses on analyzing video *streams*, as opposed to the batch jobs [37] dominant in big data clusters. Secondly, our spatio-temporal correlations enable us to *predict* person trajectories, and by extension, forecast future resource availability, which adds a new, temporal dimension to resource management.

Networking is another important dimension. Cameras often need to share data in real-time (*e.g.* A1, A2 in §3.3, #3 in §4.1). Given that the links connecting these cameras could be constrained wireless links, the network must also be appropriately scheduled jointly with the compute capacities.

Finally, given the long-term duration of video analytics jobs, it will often be necessary to *migrate* computation across cameras (*e.g.* C2 in §3.2, #2 in §4.1). Doing so will require considering both the overheads involved in transferring state, and in loading models onto the new camera's GPUs.

3) Rewind processing of videos: Rewind processing (#4 in §4.1)—analyzing recently recorded videos—in parallel with live video requires careful system design. A naïve solution is to ship the video to a cloud cluster, but this is too bandwidth-intensive to finish in near-realtime. Another approach is to process the video where it is stored, but a camera is unlikely to have the capacity to do this at faster-than-real-time rates, while also processing live video.

Instead, we envision a MapReduce-like solution, which utilizes the resources of many cameras by (1) partitioning the video data and (2) calling on multiple cameras (and cloud servers) to perform rewind processing in parallel. Care is required to orchestrate computation across different cameras, in light of their available resources (compute and network). Statistically, we expect rewind processing to involve only a small fraction of the cameras at any point in time, thus ensuring the requisite compute capacity.

5 RELATED WORK

Finally, we put this paper into perspective by briefly surveying topics that are related to multi-camera video analytics.

Video analytics pipelines: Many systems today exploit a combination of camera, smartphone, edge cluster, and cloud resources to analyze video streams [18, 19, 22, 33, 38]. Low cost model design [11, 19, 34], partitioned processing [6, 16, 39], efficient offline profiling [18, 33], and compute/memory sharing [17, 21, 24] have been extensively explored. Focus [15] implements low-latency search, but targets historical video, and importantly does not leverage any cross-camera associations. Chameleon [18] exploits content similarity across cameras to amortize query profiling costs, but still *executes* video pipelines in isolation. Our specific goal is to meet the joint objectives of high accuracy and cost efficiency in a *multi-camera* setting. In general, techniques for optimizing individual video pipelines are orthogonal to techniques for cross-camera analytics, and could be co-deployed.

Camera networks: Multi-camera networks (*e.g.* [2, 3]) and applications (*e.g.* [10]) have been explored as a means to enable cross-camera communication (*e.g.* over WiFi), and allow power-constrained cameras to work collaboratively.

Our work is built on these communication capabilities, but focuses on building a custom data analytics stack that spans a cluster of cameras. While some camera networks do perform analytics on video feeds (*e.g.* [5, 7, 8, 39]), they have specific objectives (*e.g.* feature augmentation [7], camera network topology inference [8], minimizing bandwidth utilization [39]), and fail to address the growing resource cost of video analytics, or provide a common interface to support various vision tasks. Our objective is to provide system-level support for these capabilities.

Geo-distributed data analytics: Analyzing data stored in geo-distributed services (*e.g.* data centers) is a related and well-studied topic (*e.g.* [27, 36]). The key difference in our setting is that camera data exhibits spatio-temporal correlations, which as we have seen, can be used to achieve major resource savings and improve analytics accuracy.

6 CONCLUSIONS

The increasing prevalence of enterprise camera deployments presents a critical opportunity to improve the efficiency and accuracy of video analytics via spatio-temporal correlations. The challenges posed by cross-camera applications call for a major redesign of the video analytics stack. We hope that the ideas in this paper both motivate this architectural shift, and highlight potential technical directions for its realization.

ACKNOWLEDGMENTS

We would like to acknowledge the helpful discussions and feedback from Victor Bahl and Alec Wolman. In addition to NSF CISE Expeditions Award CCF-1730628, this research is supported by gifts from Alibaba, Amazon Web Services, Ant Financial, Arm, CapitalOne, Ericsson, Facebook, Google, Huawei, Intel, Microsoft, Scotiabank, Splunk and VMware.

REFERENCES

[1] 2018. WyzeCam. https://www.wyzecam.com/.
[2] Kevin Abas et al. 2014. Wireless smart camera networks for the surveillance of public spaces. *IEEE Computer* (2014).
[3] Hamid Aghajan and Andrea Cavallaro. 2009. *Multi-camera networks: principles and applications*. Academic press.
[4] Amazon. 2017. AWS DeepLens. https://aws.amazon.com/deeplens/.
[5] Shayan M. Assari et al. 2016. Human Re-identification in Crowd Videos Using Personal, Social and Environmental Constraints. In *ECCV*.
[6] Tiffany Chen et al. 2015. Glimpse: Continuous, Real-Time Object Recognition on Mobile Devices. In *ACM SenSys*.
[7] Y. Chen et al. 2018. Person Re-Identification by Camera Correlation Aware Feature Augmentation. *IEEE TPAMI* 40, 2 (Feb 2018), 392–408.
[8] Yeong-Jun Cho et al. 2017. Unified Framework for Automated Person Re-identification and Camera Network Topology Inference in Camera Networks. In *ICCV*.
[9] Eduardo Cuervo et al. 2010. MAUI: Making Smartphones Last Longer with Code Offload. In *ACM MobiSys*.
[10] Bernhard Dieber et al. 2011. Resource-aware coverage and task assignment in visual sensor networks. *Transactions on Circuits for Video Technology* (2011).
[11] Biyi Fang et al. 2018. NestDNN: Resource-Aware Multi-Tenant On-Device Deep Learning for Continuous Mobile Vision. In *ACM MobiCom*.
[12] Ting Fu et al. 2017. Automatic Traffic Data Collection under Varying Lighting and Temperature Conditions in Multimodal Environments: Thermal versus Visible Spectrum Video-Based Systems. *Journal of Advanced Transportation* (2017).
[13] Benjamin Hindman et al. 2011. Mesos: A platform for fine-grained resource sharing in the data center. In *NSDI*.
[14] Geoffrey Hinton et al. 2015. Distilling the Knowledge in a Neural Network. In *NIPS Deep Learning and Representation Learning Workshop*.
[15] Kevin Hsieh et al. 2018. Focus: Querying Large Video Datasets with Low Latency and Low Cost. *USENIX OSDI*.
[16] Chien-chun Hung et al. 2018. VideoEdge: Processing Camera Streams using Hierarchical Clusters. In *ACM SEC*.
[17] Angela H Jiang et al. 2018. Mainstream: Dynamic Stem-Sharing for Multi-Tenant Video Processing. In *USENIX ATC*.
[18] Junchen Jiang et al. 2018. Chameleon: Video Analytics at Scale via Adaptive Configurations and Cross-Camera Correlations. In *ACM SIGCOMM*.
[19] Daniel Kang et al. 2017. NoScope: Optimizing Neural Network Queries over Video at Scale. In *VLDB*.
[20] Robert LiKamWa et al. 2013. Energy Characterization and Optimization of Image Sensing Toward Continuous Mobile Vision. In *ACM MobiSys*.
[21] Robert LiKamWa et al. 2015. Starfish: Efficient Concurrency Support for Computer Vision Applications. In *ACM MobiSys*.
[22] Sicong Liu et al. 2018. On-Demand Deep Model Compression for Mobile Devices: A Usage-Driven Model Selection Framework. In *ACM MobiSys*.
[23] Franz Loewenherz et al. 2017. Video Analytics Towards Vision Zero. In *ITE Journal*.
[24] Akhil Mathur et al. 2017. DeepEye: Resource Efficient Local Execution of Multiple Deep Vision Models Using Wearable Commodity Hardware. In *ACM MobiSys*.
[25] Microway. 2018. NVIDIA Tesla P100 Price Analysis. https://www.microway.com/.
[26] NPR. [n. d.]. It Ain't Me, Babe: Researchers Find Flaws In Police Facial Recognition Technology. https://www.npr.org/sections/alltechconsidered/2016/10/25/499176469.
[27] Qifan Pu et al. 2015. Low latency geo-distributed data analytics. In *ACM SIGCOMM*.
[28] Qualcomm. 2018. Vision Intelligence Platform. https://www.qualcomm.com/news/.
[29] J. Redmon et al. 2016. You Only Look Once: Unified, Real-Time Object Detection. In *IEEE CVPR*.
[30] Ergys Ristani and Carlo Tomasi. 2018. Features for Multi-Target Multi-Camera Tracking and Re-Identification. In *IEEE CVPR*.
[31] Ergys Ristani et al. 2016. Performance Measures and a Data Set for Multi-Target, Multi-Camera Tracking. In *ECCV Workshop*.
[32] SecurityInfoWatch. 2016. Data generated by new surveillance cameras to increase exponentially in the coming years. http://www.securityinfowatch.com/.
[33] Haichen Shen et al. 2016. MCDNN: An Approximation-Based Execution Framework for Deep Stream Processing Under Resource Constraints. In *ACM MobiSys*.
[34] Haichen Shen et al. 2018. Fast Video Classification via Adaptive Cascading of Deep Models. In *IEEE CVPR*.
[35] Bi Song et al. 2010. Tracking and activity recognition through consensus in distributed camera networks. *Transactions on Image Processing* (2010).
[36] Ashish Vulimiri et al. 2015. Wanalytics: Geo-distributed analytics for a data intensive world. In *ACM SIGMOD*.
[37] M. Zaharia et al. 2010. Spark: cluster computing with working sets. In *USENIX HotCloud*.
[38] Haoyu Zhang et al. 2017. Live Video Analytics at Scale with Approximation and Delay-Tolerance. In *USENIX NSDI*.
[39] Tan Zhang et al. 2015. The Design and Implementation of a Wireless Video Surveillance System. In *ACM MobiCom*.
[40] Liang Zheng et al. 2017. Person Re-Identification in the Wild. In *IEEE CVPR*.