4-2019

Dependable machine intelligence at the tactical edge

Archan MISRA
Singapore Management University, archanm@smu.edu.sg

Kasthuri JAYARAJAH
Singapore Management University, kasthurij@smu.edu.sg

Dulanga Kaveesha Weerakoon WEERAKOON MUDIYANSELAGE
Singapore Management University, dulangaw@smu.edu.sg

Randy Tandriansyah DARATAN
Singapore Management University, rtdaratan@smu.edu.sg

Shuochao YAO

See next page for additional authors

Dependable Machine Intelligence at the Tactical Edge

Archan Misra[a], Kasthuri Jayarajah[a], Dulanga Weerakoon[a], Randy Tandriansyah[a], Shuochao Yao[b], Tarek Abdelzaher[b]

[a]Singapore Management University, Singapore;
[b]University of Illinois, Urbana-Champagne, USA

ABSTRACT

The paper describes a vision for dependable application of machine learning-based inferencing on resource-constrained edge devices. The high computational overhead of sophisticated deep learning learning techniques imposes a prohibitive overhead, both in terms of energy consumption and sustainable processing throughput, on such resource-constrained edge devices (e.g., audio or video sensors). To overcome these limitations, we propose a "cognitive edge" paradigm, whereby (a) an edge device first autonomously uses statistical analysis to identify potential collaborative IoT nodes, and (b) the IoT nodes then perform real-time sharing of various intermediate state to improve their individual execution of machine intelligence tasks. We provide an example of such collaborative inferencing for an exemplar network of video sensors, showing how such collaboration can significantly improve accuracy, reduce latency and decrease communication bandwidth compared to non-collaborative baselines. We also identify various challenges in realizing such a cognitive edge, including the need to ensure that the inferencing tasks do not suffer catastrophically in the presence of malfunctioning peer devices. We then introduce the soon-to-be deployed Cognitive IoT testbed at SMU, explaining the various features that enable empirical testing of various novel edge-based ML algorithms.

Keywords: Machine Learning; Tactical Edge; Collaborative Sensing; Video Analytics; Cognitive Edge;

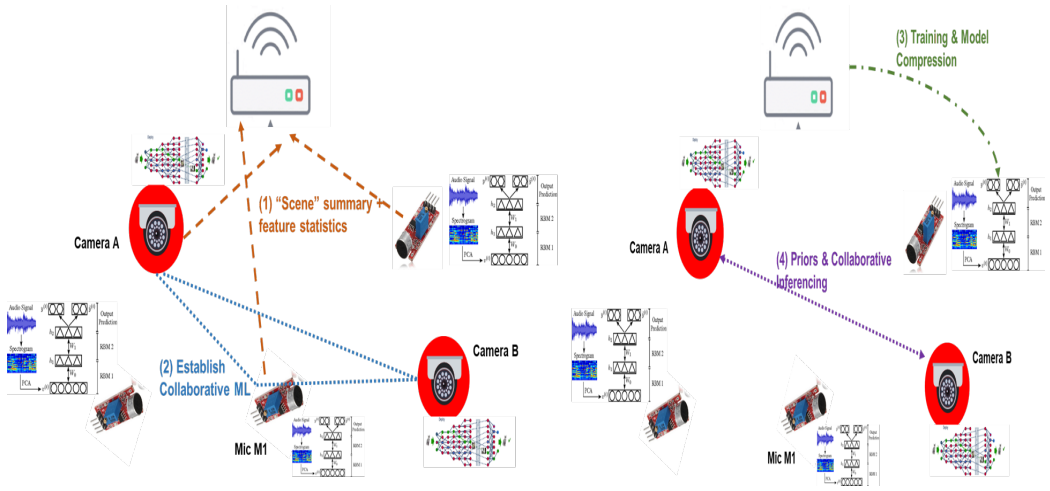
1. INTRODUCTION

There is a growing trend in creating "smart spaces", which are instrumented with a variety of resource-constrained sensors and IoT devices. For instance, a recent report forecasts that the smart home market spanning smart speakers, smart lighting, thermostats, etc. is expected to see double digit growth for the next 5 years *. In addition, such sensors (e.g., a collection of networked cameras) are expected to be widely deployed in a variety of outdoor spaces–e.g., in smart cities or in tactical military battlefields. With recent advances in machine intelligence technologies (especially, the dramatic progress in the use of Deep Neural Networks (DNNs) for perceptual tasks), such sensors enable the use of automated, real-time *sense-making* for a variety of applications and services.

Achieving this vision will, however, require us to tackle a key *resource bottleneck.* Techniques such as DNNs achieve dramatically higher accuracy in perceptual intelligence tasks, but also impose dramatically higher computational and energy loads—e.g., the state-of-the-art DNNs for vision-based object detection require specialized, *energy-hungry* GPUs for low-latency operation. As a result, these devices typically offload their computation (e.g., object recognition) to nearby, resource-rich computational entities such as cloudlets (over WLAN) or *edge computing devices* [1]. Compared to the alternative of offloading to cloud servers (over WAN), such edge-based processing offers the benefits of lower propagation delays (thereby allowing real-time operation) and greater privacy [1, 2, 3, 4, 5]. While such edge augmentation of computation has clear benefits, present edge computing models focus only on such computational offloading in an *isolated* fashion: each sensor node effectively utilizes its individual allocated computing slice on an edge device, and thus cannot take advantage of the observations and intelligence that other nearby sensors possess.

To overcome these resource limitations, we advocate the vision of a *Cognitive Edge* architecture (illustrated in Figure 1), which involves two fundamental new paradigms of embedded machine intelligence in pervasive, resource-constrained devices:

*https://www.idc.com/getdoc.jsp?containerId=prUS44971219



(a) Edge Brokering Collaboration, (b) Collaborative IoT Intelligence

Figure 1: **Overview of Cognitive Edge**: (a) Edge device performs statistical analysis on statistical summaries to identify collaboration opportunity between (Camera A, Microphone M1 and Camera B). (b) The Edge device can then assist devices with functions such as Model Compression. Individual IoT nodes (e.g., Camera A & Camera B) then collaborate to optimize their pipelines.

- *Collaborative IoT Intelligence:* Under this paradigm [6, 7], the inferencing pipelines of multiple such devices do not operate in isolation, but adapt their operation to take advantage, in real time, of insights and inferences provided by other peer nodes. This paradigm rests on the belief that the resource requirements on individual devices (e.g., in terms of energy and computational power) can be dramatically reduced, if the nodes effectively share this burden of sense-making. This approach is motivated by the observation that IoT devices and sensors are often deployed in dense configurations, resulting in varying degrees of redundant or correlated coverage. By sharing appropriate information among such peer nodes (e.g., features such as color histograms or object bounding boxes, as shown in [6] for the case of a video sensing application), this approach can reduce the need for redundant computations and help improve the overall inferencing accuracy.
- *Edge-Coordinated Collaboration:* While collaborative execution of inferencing pipelines on IoT devices can improve metrics such as accuracy and latency, the devices must first resolve the question "Who do I collaborate with?". We envision a future operating environment, where the individual IoT nodes are deployed with very little centralized control (e.g., dropped opportunistically on a specific area by one or more drones) and thus have no prior knowledge of the overall distribution of other sensing nodes. In such a scenario, the edge platform morphs from being just a platform for computational offloading to a *coordinator* that brokers such collaborative execution of inferencing tasks. Very specifically, the edge device autonomously discovers the spatiotemporal correlations among different such IoT devices and then helps the appropriate subsets of devices establish the aforementioned collaborative processing pipeline. Note that the IoT deployments will often involve a diverse set of devices, such as cheap RFID tags, moderately-priced microphone sensors and expensive infra-red cameras.The edge device must not only accurately capture the accuracy–resource needs for specific device types (e.g., [8] for video analytics), but also discover and foster such collaboration opportunities across different sensing modes.

Key Contributions: We believe that these trends enable two key, previously untapped, opportunities for machine intelligence at the edge. In this paper, we make the following key contributions:

- *Identify Key Research Challenges:* Besides introducing our vision for the *Cognitive Edge*, we describe, often with specific examples, the challenges that must be overcome for us to realize the benefits of the proposed collaborative machine intelligence paradigm.
- *Early Examples of Dependable Collaborative IoT Intelligence:* Using a set of networked video sensors as an exemplar, we provide initial evidence showing how collaborative learning and inferencing, over individual frames of

video data, can improve the accuracy for a typical "people counting" task. We also explain how reputation-based mechanisms may be used to tackle the challenge of performance loss when operating in the presence of adversarial collaborators.

- *Introduction of SMU's Cognitive IoT Testbed:* The vision proposed here calls for a new paradigm of real-time collaborative operation among different sensor/IoT nodes. While such collaboration can lead to better collective accuracy and/or reduced compute cycles, there are fundamental trade-offs that need to be studied. A key one relates to the benefits of collaboration vs. the additional communication overhead, especially as collaboration will require the communication of 'computational state' among multiple potential IoT devices, which may often be operating under significant bandwidth limitations (e.g., a tactical military network). To develop practically useful primitives for such collaboration, we require diverse real-world testbeds that enable study of the developed algorithms under diverse real-world conditions, such as poor ambient illumination and variable crowd density. We shall describe the key design features and early implementation of SMU's upcoming testbed, which will enable experimental studies on collaborative campus-scale sensing.

2. CHALLENGES FOR THE COGNITIVE EDGE

We envision the *Cognitive Edge* to be composed of sensor nodes that are diverse in terms of their sensing and sense-making capabilities (e.g., audio for speaker detection, RFID tags for occupancy counting, etc.). The real-time collaboration among nodes aims to improve operational efficiency (1) by selective/altered execution of the sense-making pipelines, and/or (2) by selective activation of the sensors. In the former, we expect that the sensors themselves are smart enough to learn and adapt their execution at run time based on information from peer nodes. In the latter case, we assume that the devices themselves are *dumb*, but rely on a smart, matchmaker that brokers between them.

To elaborate further, we take the illustrative example of an environment where multiple, networked cameras are working together for vision tasks such as object detection, recognition and re-identification (see Figure 2). We assume that some of these cameras (or, *peer nodes*) have on-device compute power that allows them to run full or partial deep learning pipelines on-board whilst some are only capable of sensing who then offload the sensed, raw data to a more powerful device nearby for running the inferences.

2.1 An Illustrative IoT Environment

Figure 2 shows a system of five networked cameras (A, B, C, D, and E). Cameras A, B and C, and Cameras C, D, and E share overlapping views, respectively. Due to the presence of such overlap, the cameras observe significant spatiotemporal-content correlations in the scenes they process. Cameras A, B, and C (referred to as *smart cameras*) are equipped with GPUs that allow them to process their respective video feeds, at near-real time, on-device. However, D and E (referred to as *dumb cameras*) are merely sensors that forward their raw feeds to a nearby edge node (or to a peer camera C), depending on the dynamic state of available bandwidth and computation cycles on the other nodes.

In the baseline, *non-collaborative* setup, each of these smart cameras will execute DNNs for people detection *ndependently*. Additionally, both D and E, despite their significant overlap resulting from spatial proximity, are always *on* and upload their raw video feeds incurring high bandwidth costs. In the *Collaborative model* that we propose, the peer cameras A, B and C share their respective inferences and intermediate features/metadata with each other, on a per-frame basis. By combining the inferences of the peers (which effectively serve as 'priors') with its own input (each image frame), each camera runs a modified, *collaborative* inferencing pipeline that is likely to offer improved accuracy. For example, the detection accuracy may be enhanced when one or more objects are occluded in one camera's view (e.g., Camera A is occluded by the presence of the tree and fails to observe the "green" person) but visible in the field of view (*FoV*) of another camera (e.g., Camera B observes the "green person", thereby helping camera A overcome its limitation).

To illustrate a significant potential drawback of such collaborative inferencing, the figure also illustrates an adversarial camera, C, which shares misleading or incorrect features/metadata–e.g., it informs cameras A, B, D and E that it can only observe the "purple" individual, deliberately omitting the other ("green") person. To remain robust under such conditions, the proposed Cognitive Edge should posses additional built-in *resilience mechanisms*– e.g., Camera A, who shares/receives inferences from both its neighbors B and C, will *learn* from past observations that Camera C is *adversarial* and that B is *trust-worthy*, and makes the correct inference that there are in fact 2 persons in the current frame (although its own view is occluded) by combining its own and B's inferences and *disregarding* C.

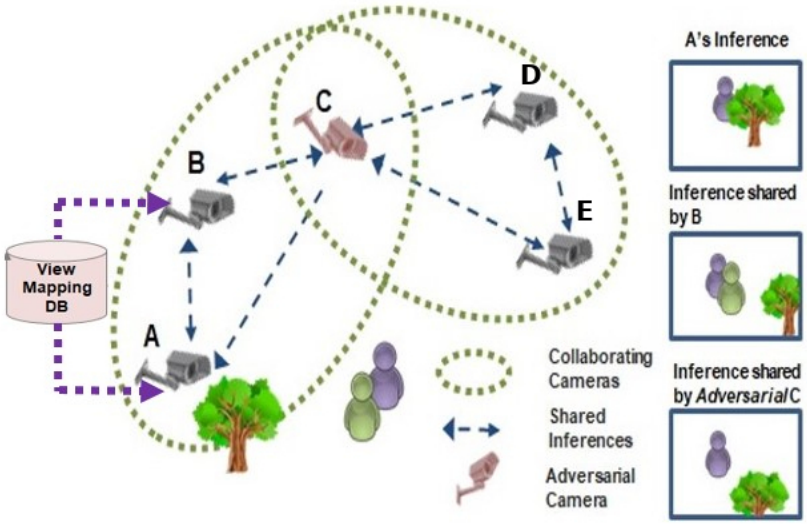


Figure 2: Illustrative IoT environment with collaborative cameras.

Additionally, in this scenario, the edge node will act as broker between cameras D and E (which do not possess on-board compute power). Over time, the edge will learn the spatial correlation between D and E that will allow it to selectively trigger/activate D and E to turn ON (and OFF) depending on the nature of the anticipated scenes to limit the amount of raw video that they generate and communicate over the network.

2.2 Key Challenges

Our overall collaborative sensing approach promises to significantly improve the latency and accuracy of sense-making on resource-constrained devices, with the active support of an edge platform providing the necessary coordination functionalities. However, there are various challenges that our proposed vision will have to tackle and solve, before the framework can be realized, especially in resource-constrained tactical military environments.

Discover and Exploit Correlations: Our concept of collaborative sensing envisages the use of an edge device as a coordinator–one that autonomously discovers *spatiotemporal* correlations among nearby devices and thereby enables collaboration among them. To discover such correlations, the individual IoT nodes will certainly have to share appropriate features from their underlying sensor data streams. However, determining the appropriate set of such features, especially in a multi-modal sensing environment, is an open and challenging research problem. There are several factors to consider. First, the features must be *discriminative*–i.e., they must have enough informative power for the edge node to identify *real* correlations, while minimizing the likelihood of *false matches* (spurious correlations). Once such correlations have been identified, the individual IoT nodes can then establish the specific *transformations* that exploit such correlation. For example, in Figure 2, the cameras may learnt the homographic transformation [9] between their views: the "View Mapping DB" component holds such mappings, and allows each camera to translate the observations of its peer cameras to its own internal coordinate system. Second, given the practical need to operate in bandwidth-constrained scenarios, the features themselves must be *parsimonious* in usage of communication resources, as otherwise the communication bandwidth and energy overheads with negate any benefits of collaboration. Third, in many situations, especially in military scenarios, the IoT nodes may not necessarily trust the coordinating edge node and thus may not be willing to share sensitive features (for security and privacy reasons). Thus, it'll be necessary to identify features that are both discriminatory and avoid possible leaks of the underlying sensor information streams. In Section 3.2, we shall share preliminary evidence from our recent work on exemplars of such features, and the associated accuracy improvement and bandwidth overheads.

Handle Disparate Sensing Modalities: In this paper we introduce the idea of cognitive edge and to validate our claim of the benefit of collaboration, we use video sensors. Down the road we may want to extend our proposed architecture to other types of sensor modalities. Although we introduce the frameworks and key concepts by taking video data as a

proof of concept, extending this to other types of sensor modalities (audio, motion, magnetic) remains as an open challenge. Whether and how cross-modal correlations can be learned and applied for operational improvements is an important question to be explored.

Achieve Resiliency under Adversarial Behavior: As illustrated in Figure 2, the presence of an adversarial IoT node (eg., Camera C) can affect the performance and accuracy of the whole proposed system. Since such adversarial nodes will share their inference results with the neighboring nodes, we need to design appropriate mechanisms to detect and isolate such adversarial nodes. In Section 3.3), we shall describe our early efforts in enabling resilient-collaboration, via the use of reputation scores. This mechanism is, however, based on the assumption that a malicious camera node *continuously* and *randomly* modifies its shared information features (the bounding boxes of detected people objects). Truly malicious nodes may, however, exhibit more complicated and sinister behavior–e.g., failing to report accurate bounding boxes only for selected individuals of high interest. Moreover, the proposed histogram-based approach is likely to be applicable only for nodes with concurrent spatial overlap. We will clearly need to build more sophisticated strategies to handle a wider variety of adversarial behavior (e.g., multiple colluding nodes), especially when the nodes are correlated *temporally* rather than spatially.

Handle Deployments with Variable Node Density & Administrative Control: Different environments may show wide variability in the density of deployed nodes. In scenarios where nodes are densely deployed, we will need to adaptively optimize the *degree of collaboration* so as to address the likely diminishing return from sharing of information with multiple peer sensors. To obtain an appropriate tradeoff between performance improvement gains and additional communication overhead, we will thus need to develop appropriate *adaptive collaboration mechanisms.* Such adaptive mechanisms are also important in situations where all the IoT devices do not belong to a single administrative entity–e.g., a military scenario involving machine intelligence utilizing nodes belonging to different members of a multinational coalition force. In such cases, there may be constraints on the duration for which one can access and utilize data from a coalition partner's node, implying the need to develop mechanisms that maximize the performance gain while adhering to such constraints.

Handle Dynamic Workloads & Contexts: When IoT devices are deployed statically, one can initially assume that the spatiotemporal correlations are relatively invariant, implying a long-term stability in the set of collaborating devices. However, there are various situations where the desired set of collaborating nodes varies dynamically. For example, correlations among a set of campus-deployed video sensors may be driven by the typical spatiotemporal patterns of human movement (e.g., people entering through the main entrance typically exit the elevator on levels 2 or 3). However, such movement patterns may be both *time* (e.g., different patterns during classes and after hours) and *context* dependent (e.g., different patterns may emerge during campus events). Similarly, our collaborative model may also be employed by a set of mobile IoT nodes (e.g., a fleet of UAVs tasked with monitoring a geographic region). In this case, as the nodes move around, the set of proximate IoT devices and their likely future movement patterns may change dynamically, requiring continual updates to the collaborative inferencing strategies.

Support Flexible Experimentation on IoT Testbeds: As the vision proposed here calls for a new model of real-time collaborative operation among co-located, networked sensors, an experimentation platform to test out new algorithms under realistic experimental conditions is warranted. Such testbeds should allow for researchers and system designers to be able to seamlessly deploy their versions of algorithms (e.g., modified object detectors to exploit collaboration) without the overhead of deploying custom IoT networks from scratch. Such an experimentation platform should also allow for repeatable testing under *dynamic working conditions* such as sensor density & environment parameters (e.g., lighting, crowd density). In Section 4, we describe our early efforts in building and deploying such a testbed in our SMU campus in Singapore.

3. DEPENDENDABLE COLLABORATIVE INTELLIGENCE AT THE EDGE

In this section, we describe some of our recent efforts (borrowing on results presented in greater detail in [6, 7]), in exploring (1) features and methods for collaboration for improved accuracy, and (2) mechanisms for resilient operation under adversarial settings. All our experimental results presented here are based on the PETS 2009 benchmark dataset for detection and tracking of people across multiple cameras. We describe this dataset and the performance metrics next.

3.1 Evaluation Setup

We use video feeds from the eight synchronous cameras present in the PETS 2009 dataset [10]. The cameras are placed outdoors and capture pedestrian traffic under varying density settings. We limit our experiments to four of the eight views (views 5-8 with view 5 being considered as the *reference* camera with respect to which we report all our findings) for which manually annotated ground truth was provided [11]. The videos were captured at a frame rate of 7 FPS and a resolution of 720×576 resulting in a total of 795 frames per camera (3180 frames across the 5 cameras). We build on the implementation of the Single Shot Detector (SSD)[†], originally proposed by Liu et al.[12]. The SSD is a DNN for object detection, trained on the PASCAL VOC dataset [13]. We limit our experiments to "person" detection alone. To extract the spatial correlations between pairs of camera views, we use a simple annotation tool to mark corresponding points between pairs of views, and then generate the homography matrices using such matched points. In the evaluations, we consider two accuracy metrics: (i) *Multiple Objects Detection Accuracy* (MODA) as defined by Bernadin et al. [14] and (b) the $F-$score (in its usual meaning) for the person detection task.

3.2 Improving Operational Efficiency through Collaboration

As mentioned previously, we hypothesize that operating in a collaborative mode can benefit in terms of improved accuracy, or reduced processing latency/compute cycles, or both.

Improving Accuracy: In our recent works, we have demonstrated the use of two alternate designs for collaboration: (1) Collaborative Non-Maximum Suppression (CNMS [6] [7]), and (2) Collaborative–SSD (CSSD [7]).

In CNMS, we alter the "Non-maximum Suppression" step which is a basic post-processing step executed in most deep-learning based object detectors. At this step, the bounding boxes output by the deep learner are checked for redundancy (by quantifying the overlap with other bounding boxes measured using Intersection-Over-Union or *IoU* values) and suppressed. In the collaborative version of this suppression step, each camera takes as input (i) the bounding boxes from running the SSD on its own view, and (ii) the coordinate-transformed bounding boxes shared by the peer cameras from running the SSD pipeline on their own respective views, and then performs a Bayesian-based non-maximum suppression. The Bayesian probabilities (weights) for fusion are derived from the confidence values associated with the bounding boxes.

On the other hand, in CSSD, we encode the collaborative information on to SSD DNN pipeline, itself. In addition to the input image from the current camera, CSSD takes a binary mask from each of the peer cameras as an auxiliary input. These binary input masks are generated from the mapped bounding box coordinates from the peer nodes, with pixels inside the bounding box encoded as '1' and pixels outside encoded as '0'. In effect, one can image that the conventional RGB imaged (with a *depth*=3) has been augmented to have a higher depth of $N + 3$ (where N is the number of collaborating peer cameras). Note that, unlike CNMS, CSSD requires the modification of the architecture of the SSD pipeline, and thus also requires the re-training of the underlying DNN.

In Table 1, we tabulate the accuracy (measured as $F-$score) and the average time taken for processing (i.e., processing latency) a single frame for the baseline SSD and the two collaborative SSD alternatives. We make the following observations:

- Both collaborative models outperform the non-collaborative baseline SSD (improvement of 4.5% for CNMS and 11.2% with CSSD).
- The collaborative models, however, have a slight increase in in inference latency due to the additional processing involved in combining the inferences from multiple peers.

In addition, We have demonstrated in our earlier work [6] that the marginal improvement in accuracy diminishes with an increasing number of collaborators: in particular, we found that collaborating with 2–3 peers offers the best trade-off between accuracy improvement and processing overhead.

Reducing Communication Bandwidth: We now compare the communication bandwidth overhead of our collaborative models (which require sharing of the bounding box coordinates with peer cameras) with an alternative centralized approach, where each camera merely uploads its raw video feed to a centralized server (which can then fuse the inputs and apply a corresponding DNN pipeline). Table 2 provides a comparison of the network latency in terms of the total bandwidth

[†]Available from `https://github.com/weiliu89/caffe/tree/ssd`

Table 1: Accuracy and Latency comparison of Collaborative Models

Model	**Inference Time**	**Accuracy**
SSD Baseline	80 msec	71%
CNMS	100 msec	75.5%
CSSD	85 msec	82.2%

required for executing people detection algorithms in one central server (i.e., requiring raw images to be uploaded over WAN/LAN) vs. sharing bounding box coordinates among neighboring nodes (i.e., via CNMS/CSSD). Evidently, the proposed collaborative architecture requires 10-times lower bandwidth than the alternative. We note that *smart* cameras operating in the non-collaborative mode do not incur network latency as they process on-board but are disadvantaged in terms of the potential accuracy possible.

Table 2: Bandwidth comparison of processing at the server vs processing at the cognitive edge

	Total Bandwidth per Node
Uploading raw images to WAN/LAN (Baseline Architecture)	1.5Mbps
Sharing bounding box coordinates (Collaborative Architecture)	150Kbps

Reducing Latency: We next provide some preliminary results on possible approaches for using collaborative inferencing to reduce the processing latency. Table 3 provides the latency numbers from trials using a Raspberry Pi 3 Model B+ while running inferences using the SSD pipeline, with and without a Vision Processing Unit (VPU, e.g., Intel Movidius Stick) [‡]. We see that the default model (where the image inputs are re-sized to 300 $\times$ 300), running on the device's native CPU takes an average of 1000 msec to process a single image. This limits the maximum frames per second (*fps*) of execution ($= 1$) whilst the camera's *fps* is much greater ($= 30$). Of course, if the device is equipped with the VPU, the time taken drops significantly (down to 86 msec)–note, however, that even this reduced latency is insufficient to support the full throughput rate (30 fps $\rightarrow\sim$ 30 msecs).

To further reduce the latency, we considered two possible modifications to the DNN pipeline, both of which collaboratively utilize information from peer cameras. Our key insight is that, when such an infrastructure is deployed in places such as a university or office campus, the environment is likely to have varying levels of crowd density throughout the day. We then assume that each camera can use the bounding box information from its peer cameras (specifically, the *count* of people objects) to predict the likely people density in the upcoming frames. If this predicted crowd density is low, then the camera may switch to a lighter version of the SSD DNN pipeline, either by (i) reducing the image resolution to 100 $\times$ 100, or by (ii) downsampling the anchor boxes (see Liu et al.[12] for details). We see that both these alternatives result in a significant reduction in the inferencing latency (25 msec and 75 msec respectively). More importantly, we observed that these simpler models did not suffer from any drop in accuracy when the crowd density was low. The results shows that such collaboration-driven adaptation (dynamically switching between low and high complexity DNN models) can enable a camera to process images at rates up to 40 fps, without any loss of accuracy.

3.3 Establishing Resiliency in Collaborative Settings

As illustrated in Figure 2, sensors may need to operate in adversarial settings. In such situations, collaboration may actually degrade performance. Accordingly, frameworks for collaborative IoT intelligence also need to develop resiliency mechanisms to detect and isolate such adversarial nodes.

We summarize here two key mechanisms for building in resilience that we have explored recently:

[‡] `https://www.movidius.com/`

Table 3: Comparison of processing latency (Raspberry Pi 3 Model B+[§] under different collaborative models

	Latency (msec)
Inference on RPI CPU (Without Movidius)	1000msec
Inference on Movidius (300x300)	86msec
Inference on Movidius (300x300) (Reduced Anchor Boxes)	75msec
Inference on Movidius (100x100)	25msec

1. **At run time:** Individual networked-cameras compute and maintain a *reputation score* of their respective peer camera nodes, and selectively choose *whom* to collaborate with, on a per-frame basis.
2. **At train time:** We incorporate resilience by introducing random perturbations in the peer information (e.g., binary masks) during the training phase.

The first approach of maintaining a reputation score was introduced in [6] together with the CNMS collaborative model. The reputation score is updated real-time and is quantified using the mutual correspondence among the inference results of the two cameras. More specifically, the detected objects obtained from both cameras (the peer camera and the reference one) are first *matched* using the Hungarian algorithm based on the location of the boxes. Subsequently, for matched objects, their content similarity is computed by measuring the correlation between the color histograms of the corresponding matched boxes. The reputation score of the peer camera is then updated, on a per-frame basis, as follows: $R_{new} = R_{old} + I \times C$, where $I = 1$ if a match is found, and is 0 otherwise, $C \in [-1, 1]$ represents the correlation value and R is the reputation score.

In the second approach, we explore the possibility of incorporating robustness with CSSD. During the training of the CSSD model with the PASCAL VOC dataset, as a step of data augmentation, we randomly hide and translate the ground truth bounding boxes. We note here that as the PASCAL VOC dataset was not originally meant for multi-camera, multi-view settings, it does not consist of "peer" nodes – hence, we rely on perturbed versions of ground-truth boxes as being generated by trust-worthy as well as malicious peers.

In order to evaluate the effectiveness of these mechanisms, we conduct a number of experiments. We first perturb a camera's view with Gaussian noise of increasing intensity to understand the impact of noise on inference accuracy in

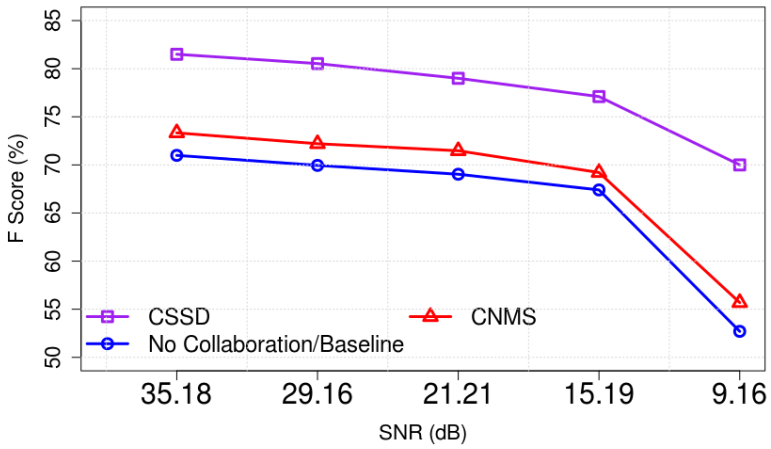
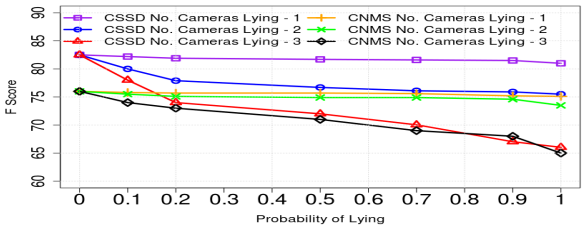
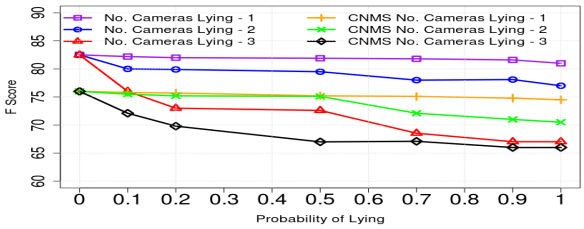


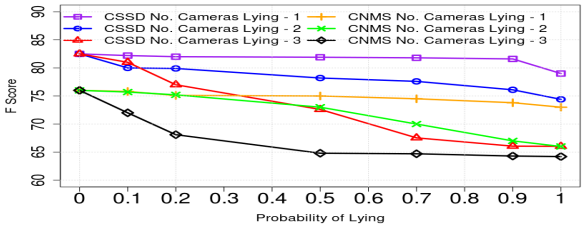
Figure 3: Noise vs. Inferencing Accuracy [7]



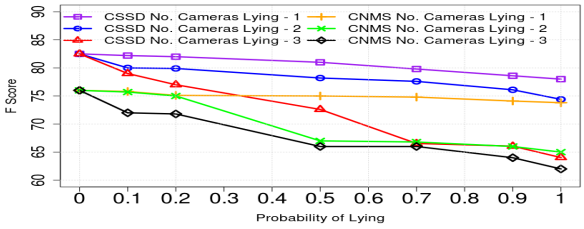
(a) Original detections



(b) Low Noise



(c) Medium Noise



(d) High Noise, Baseline $F-$score=52%

Figure 4: Performance of (*CNMS* and *CSSD*) under Adversarial Conditions.

collaborative and non-collaborative modes. Figure 3 (reproduced from [7]) shows the signal-to-noise ratio (SNR) of the perturbations (in dB) on the $x-$ axis and the resulting variation in F-Score of people detection on the $y-$ axis. We observe that (1) both CNMS and CSSD are more resilient to noise than the baseline, and (2) that CSSD achieves a significantly higher accuracy at all SNR levels (11% compared to baseline where the camera is not collaborating).

Next, we investigate the robustness of the collaborative models whle operating under adversarial conditions. As we describe in Weerakoon et al.[7], we mimic such adversarial behaviour by configuring each peer camera to *lie* with a certain probability – *lying* refers to the camera either completely suppressing a detection or merely perturbing the location coordinates of the bounding box of detected object. In Figure 4a, we plot the variation of the F-Score (on the $y-$ axis) against increasing probability of lying (on the $x-$ axis). The resulting accuracy is reported for differing number of peer cameras generating false bounding box coordinates (1, 2 and 3) where the reference camera collaborates with 3 peers at most. Further, we extend the study by varying the intensity of perturbations (low, medium and high) and plot the resulting accuracy in Figures 4d, 4c and 4b.

The figures show that *CSSD* is significantly more robust to such adversarial behavior than *CNMS*, with its accuracy dropping marginally even for moderate noise and large values of p, *as long as the number of adversarial cameras is 1 or 2*. *CNMS*, on the other hand, experiences a sharper ($\sim$15%) drop in accuracy even under modest adversarial behavior ($p = 0.3$). The results show the promise of *CSSD's* collaborative learning approach, but also illustrate the need to build additional *explicit* mechanisms to protect such frameworks against adversarial behavior.

4. THE "COGNITIVE IOT" TESTBED

We now describe the preliminary design and implementation of the "Cognitive IoT" Testbed that we have developed and deployed in SMU campus network. The testbed is intended to enable real-world study of different collaborative sensing techniques, under dynamic workloads and various environmental context. Figure 5 shows the top level architecture of the proposed Cognitive IoT testbed. As illustrated in the figure, the system comprise of a server and client service. The server is responsible for handling the top level configuration of the system, providing a user interface through which researchers can control their experiments and configuring the entire collaborative system. The Client service runs on multiple IoT nodes, each of which can run its custom collaborative machine intelligence protocols. Researchers can customize the collaborative logic as well as the layout of the IoT nodes to suit their needs.

Based on the above figure, each IoT node supports the following key functional components:

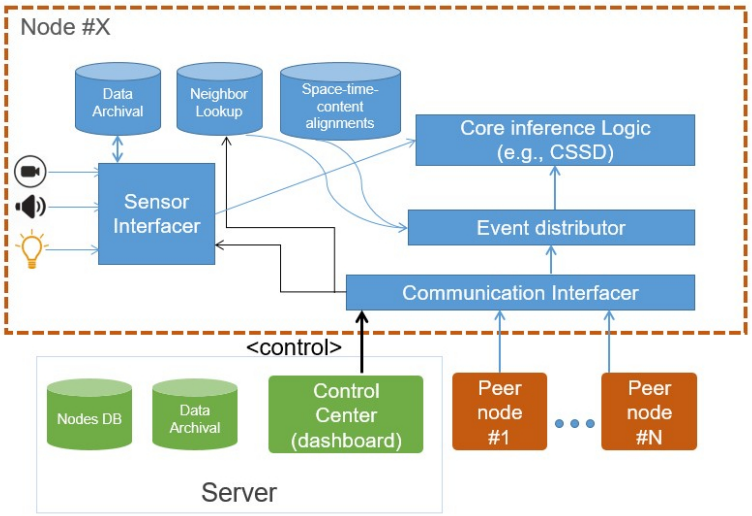


Figure 5: SMU Cognitive IoT Testbed.

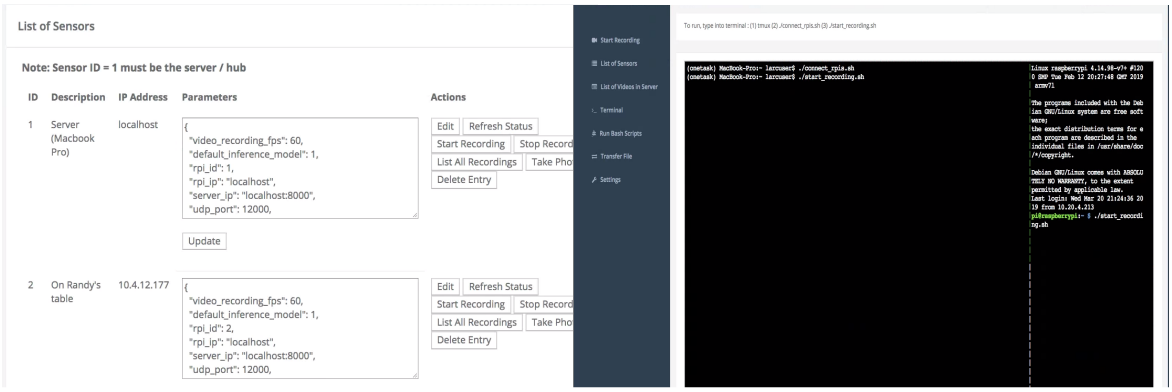
1. **Communication interface:** This module consists of processes to allow a camera to communicate with the control centre (to receive appropriate configuration parameters and commands) and also to all the other cameras. Each camera, as well as the control center, hosts a Web server, and uses HTTP messages to communicate among themselves. In addition, each camera has a direct UDP-based communication interface with each individual peer camera to transfer runtime data (such as the bounding box coordinates) efficiently.

2. **Event distributor:** This module interprets and forwards the instructions/data received from the communication interface to destination modules by either RPC or publish-subscribe mechanism. The instructions can range from starting/stopping processes (recording/inference), changing properties of the other running processes during runtime (e.g which inference model to use, which cameras to send/receive message from during inference stage), etc. By allowing hot swapping of an individual node's inferencing module and collaboration messages, this module enables us to additionally simulate adversarial behavior by other nods (e.g., by instructing a camera to incorrectly perform random discard of bounding box information).

3. **Sensor interfacer:** This is implemented as a multi-process module that connects to the underlying sensor (video camera in our initial setup), retrieves the corresponding data stream, performs appropriate buffering (e.g., discarding intermediate frames if the inference engine's throughput is lower than the frame rate) and sends it to both the inference engine (for real-time machine intelligence) and the data archival component (for future trace-based playback and analysis). To help emulate various controlled experiments, the module also includes "virtual sensors" that effectively bind to, and replay, video files stored on the server.

4. **Core inference logic:** This module (which implements the baseline and SSD pipelines described earlier) contains the inference engine and processes to load specific models (e.g. different DNN models), execute the models and archive the inference results. Our node device consist of a Raspberry Pi equipped with a camera and Intel Movidius Neural Compute Stick. Intel Movidius is equipped with a Vision Processing Unit (VPU), specially designed for parallel execution of algorithms. In our system we use the VPU to run our SSD DNN at the edge while the CPU will be fetching videos from the cameras. As a point of comparison, a CPU-only execution of the SSD DNN takes ~ 1 sec per frame, whereas a similar execution in the VPU takes 85 msec.

5. **Space-time correlator:** This module stores the spatiotemporal correlations among different cameras. In particular, if two cameras have overlapping views, the alignment will include the coordinate transformation matrix; on the other hand, if the two cameras are temporally correlated, this module will store the statistics (e.g., mean, variance) of the underlying temporal shift. Such shift is used by the Inferencing Engine, for example, to align the frames from different cameras for collaborative processing.



(a) Camera Configuration Settings (b) Command line interface

Figure 6: Illustrative images for the Testbed Web Dashboard

6. **Data archival:** This module stores and tags output data (video, bounding boxes) from other processes. Note that each output data may have its own custom format meta-data–e.g., a video file may include parameters such as resolution, location and recording duration. The server stores and indexes such metadata, allowing the dashboard interface to perform metadata search queries (e.g., 'find HD videos shot by camera 1 last Monday'). .

4.1 Web Dashboard

At its core, the web dashboard (illustrated in Figure 6) provides the following capabilities:

1. **Adding new cameras and to view current state of the system**

   Each camera is an entity/record in the system with some static properties (location, IP address) and dynamic properties that may be modified during experiment (whether a camera is currently recording, camera resolutions, inference model being used, etc). The dashboard provides CRUD functionality (create, read, update, delete) to view and update the state of the system via web interface. During experiment execution, any changes to a camera parameter triggers the dashboard backend to connect to the camera and immediately update its properties.

2. **Automating parts of setting up and running experiments**

   To run an experiment, the web dashboard provides a screen to select which cameras to participate, along with the video files to use for each camera. Once the experiment is set up, the dashboard will prepare a configuration file for each camera, containing the experiment parameters e.g live recording vs replay, recording resolution, duration, fps, etc. The dashboard will also show a Web page with multiple command line interfaces. Each of the command line connects via ssh to the camera and allows the user to see the actual experiment running.

3. **Managing and transferring files across the cameras and the server**

   Other than changing the runtime behaviour of the cameras, the dashboard also provides features to manage static assets such as files (codes, models), videos, etc. The current implementation supports both a typical GUI-based selection method for individual files, as well as an *rsync* capability to support a transfer of multiple files between different cameras or to server. To manage videos, the dashboard provides a screen to transfer videos between cameras and server. It also provides an API to upload video along with its metadata. Once uploaded, the metadata will be indexed, for subsequently user-driven, query-based selection and retrieval.

4.2 Extension to other Sensor Modalities

While the the Cognitive IoT testbed presently consists of video sensors, our generic vision is to build a unified platform consisting of heterogeneous sets of sensors–e.g., cameras, microphones and motion sensors. The overall architecture is deliberately defined to be easily extensible to such multi-modal sensors. As shown in the figure 5, the sensor interfacer module will handle the data retrieval from the sensor. Modifications and enhancements to the collaboration logic are also possible through appropriate changes in the *Core Inference Logic module*.

5. RELATED WORK

Edge computing technologies are at the forefront of enabling situation awareness via real-time analytics. Many early examples of traditional edge computing have been in the domain of video-based applications and services due to the high resource demands required in processing images using state-of-the-art deep learners [1, 3, 5]. In recent times, multi-device cooperation, at the edge, has received considerable attention. For example, recent literature in multi-camera systems [15, 16, 17], and unmanned aerial vehicle swarms [18, 19] have explored early ideas of cooperation among peer devices owing to their natural advantage in terms of improving accuracy as well as in reducing the bandwidth and latency overhead involved in communicating with a central server.

Early efforts in describing the need for enabling collaborative intelligence among heterogeneous IoT devices, complementary to our vision, have been advocated recently [15, 16, 17, 20]. Qiu et al.[15] describe a scenario where cameras of differing capabilities co-exist in the same network (as we illustrate in Section 2.1): fixed surveillance cameras and resource-constrained mobile devices with cameras. The authors demonstrate that moving vehicles can be tracked seamlessly across this heterogeneous camera network through selective actuation of devices without overly draining the mobile devices. In essence, the resource-intensive video analytics pipeline is performed on the cloud and the mobile cameras are consulted intermittently, only to resolve ambiguities. Further, Lee et al. [16] demonstrate significant savings in bandwidth needs (of *dumb* cameras that offload raw footage to a central cloud) – they show that by establishing space-time relationships, apriori, between co-existing cameras, that they can be selectively turned ON (and OFF) leading to as much as 238 times savings in bandwidth at a miss rate of only 15% for a vehicle detection task. Similarly, Jain et al. [17] also show that significant correlations exist between co-located cameras, and discuss configurations of video analytics pipelines that can be triggered by peer cameras leading to both cost efficiency and superior inference accuracy. Most recently, we describe our vision [20] for providing machine intelligence as a service at the edge for resource-constrained devices – in addition to outlining core capabilities required for enabling such a service (e.g., scheduling, caching, resource profiling), we also describe opportunities for the convergence of the idea of collaboration between devices and deep intelligence as a service.

One of our main goals in enabling collaboration is in reducing the processing latency of running DNNs on resource-constrained devices. While we advocate for information sharing between co-located devices to run altered pipelines, at run time, to save time, separately, there have been efforts in recent times to cut down on execution time for device operating in isolation – i.e., without collaboration. For example, Yao et al.[21, 22] demonstrate that compressing DNNs offline through selective pruning of the network can lead to significant savings in run time without compromise in accuracy. Exploiting the fact that scenes do not change so drastically between consecutive frames in streaming videos, Huynh at al. [23] show that intelligent caching (and, hence, avoiding redundant) computations at intermediate layers of the deep networks can help in saving time. While these frameworks are prescribed for individual devices, we believe that our proposed vision is complementary to this line of investigations, and that convergence would lead to greater savings.

6. CONCLUSION

This paper introduces our vision of the *Cognitive Edge*, a framework where an edge device is used not just for conventional computation offloading but to enable broader collaborative execution of machine intelligence tasks on resource-constrained IoT nodes. We view such collaboration as central to our vision of executing complex DNN-based machine intelligence pipelines on such IoT devices, but with significantly lower energy overheads and latency and (hopefully) higher accuracy. We have outlined the key open challenges for such a framework, including the need to develop effective tradeoff mechanisms between communication and computing overhead, under dynamically changing conditions such as the density of the sensor nodes or the environmental context being monitored. We provided early examples of the promise of this approach, using as an exemplar a distributed outdoor video sensing network and a people counting task. We have also outlined the design of SMU's Cognitive IoT testbed, which is intended to facilitate flexible experimentation with various DNN-based pipelines and multiple sensing modalities. We hope that this testbed can be integrated into a broader, global testbed for "IoT Machine Intelligence" research.

7. ACKNOWLEDGEMENTS

This material is supported partially by by the Air Force Research Laboratory, under agreement number FA5209-17-C-0006, and partially by the National Research Foundation, Prime Ministers Office, Singapore under its International Research Centres in Singapore Funding Initiative. The view and conclusions contained herein are those of the authors and should

not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the US Government.

References

[1] Satyanarayanan, M., Chen, Z., Ha, K., Hu, W., Richter, W., and Pillai, P., "Cloudlets: at the leading edge of mobile-cloud convergence," in [*2014 6th International Conference on Mobile Computing, Applications and Services (MobiCASE)*], 1–9, IEEE (2014).

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

[3] Satyanarayanan, M., "Edge computing for situational awareness," in [*Local and Metropolitan Area Networks (LANMAN), 2017 IEEE International Symposium on*], 1–6, IEEE (2017).

[4] Satyanarayanan, M., Simoens, P., Xiao, Y., Pillai, P., Chen, Z., Ha, K., Hu, W., and Amos, B., "Edge analytics in the internet of things," *IEEE Pervasive Computing* (2), 24–31 (2015).

[5] Simoens, P., Xiao, Y., Pillai, P., Chen, Z., Ha, K., and Satyanarayanan, M., "Scalable crowd-sourcing of video from mobile devices," in [*Proceeding of the 11th annual international conference on Mobile systems, applications, and services*], 139–152, ACM (2013).

[6] Misra, A., Weerakoon, D., and Jayarajah, K., "The challenge of collaborative iot-based inferencing in adversarial settings," in [*The First International Workshop on Internet of Things for Adversarial Environments, INFOCOM*], IEEE (2019).

[7] Weerakoon, D., Jayarajah, K., Tandriansyah, R., and Misra, A., "Resilient collaborative intelligence for adversarialiot environments," in [*(Under Review) FUSION*], IFIF (2019).

[8] Jiang, J., Ananthanarayanan, G., Bodik, P., Sen, S., and Stoica, I., "Chameleon: Scalable adaptation of video analytics," in [*Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication (SIGCOMM'18)*],

[9] Hartley, R. and Zisserman, A., [*Multiple view geometry in computer vision*], Cambridge university press (2003).

[10] Ferryman, J. and Shahrokni, A., "Pets2009: Dataset and challenge," in [*2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*], IEEE (2009).

[11] Xu, Y., Liu, X., Qin, L., and Zhu, S.-C., "Cross-view people tracking by scene-centered spatio-temporal parsing.," in [*AAAI*], 4299–4305 (2017).

[12] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C., "SSD: Single shot multibox detector," in [*ECCV*], (2016).

[13] Everingham, M., Eslami, S. M., Gool, L., Williams, C. K., Winn, J., and Zisserman, A., "The pascal visual object classes challenge: A retrospective," *Int. J. Comput. Vision* **111**(1) (2015).

[14] Bernardin, K., Elbs, A., and Stiefelhagen, R., "Multiple object tracking performance metrics and evaluation in a smart room environment," in [*Sixth IEEE International Workshop on Visual Surveillance, in conjunction with ECCV*], **90**, 91, Citeseer (2006).

[15] Qiu, H., Liu, X., Rallapalli, S., Bency, A. J., Chan, K., Urgaonkar, R., Manjunath, B., and Govindan, R., "Kestrel: Video analytics for augmented multi-camera vehicle tracking," in [*Internet-of-Things Design and Implementation (IoTDI), 2018 IEEE/ACM Third International Conference on*], 48–59, IEEE (2018).

[16] Lee, J., Abdelzaher, T., Qiu, H., Govindan, R., Marcus, K., Hobbs, R., Suri, N., and Dron, W., "On tracking realistic targets in a megacity with contested air and spectrum access," *MILCOM* (2018).

[17] Jain, S., Ananthanarayanan, G., Jiang, J., Shu, Y., and Gonzalez, J., "Scaling video analytics systems to large camera deployments," in [*In Proc. of HotMobile*],

[18] Chen, X., Purohit, A., Dominguez, C. R., Carpin, S., and Zhang, P., "Drunkwalk: Collaborative and adaptive planning for navigation of micro-aerial sensor swarms," in [*Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*], *SenSys '15* (2015).

[19] Bürkle, A., "Collaborating miniature drones for surveillance and reconnaissance," in [*Unmanned/Unattended Sensors and Sensor Networks VI*], **7480**, 74800H, International Society for Optics and Photonics (2009).

[20] Abdelzaher, T., Yao, S., Hao, Y., Zhao, Y., Piao, A., Shao, H., Liu, D., Liu, S., Hu, S., Weerakoon, D., Jayarajah, K., and Misra, A., "Eugene: Towards deep intelligence as a service," in [*The 39th IEEE International Conference on Distributed Computing Systems (ICDCS 2019), Dallas, Texas USA. July 710, 2019*],

[21] Yao, S., Zhao, Y., Zhang, A., Su, L., and Abdelzaher, T., "Deepiot: Compressing deep neural network structures for sensing systems with a compressor-critic framework," in [*The 15th ACM Conference on Embedded Networked Sensor Systems (ACM SenSys), 2017*],

[22] Yao, S., Zhao, Y., Shao, H., Liu, S., Liu, D., Su, L., and Abdelzaher, T., "Fastdeepiot: Towards understanding and optimizing neural network execution time on mobile and embedded devices," in [*The 16th ACM Conference on Embedded Networked Sensor Systems (ACM SenSys), 2018*],

[23] Huynh, L. N., Lee, Y., and Balan, R. K., "Deepmon: Mobile gpu-based deep learning framework for continuous vision applications," in [*Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*], *MobiSys '17* (2017).