2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

Towards "Smarter" Vehicles through Cloud-Backed Swarm Cognition

Augusto Vega, Alper Buyuktosunoglu, Pradip Bose

***Abstract*— We consider an exciting new computing paradigm called *cloud-backed swarm cognition*. It refers to an envisioned future where (mobile) edge devices work together collaboratively to provide intelligent services. The cloud provides a resilient back-up cover and supervision for such services. A specific example domain is that of smart, connected autonomous vehicles (e.g. cars, buses, trucks, or drones). In this computational paradigm, the cloud serves as the (relatively) stable repository of reference knowledge that is less frequently accessed than in non-swarm computation. The "vehicle swarm", on the other hand, serves as a (relatively) dynamic cache of knowledge at the "edge." Leveraging the collaborative swarm mode reduces real-time deadline pressures at the individual node level, while improving edge resilience through redundancy. Overall, this leads to smarter vehicles through: (a) improved edge inferential accuracy and (b) improved system-level energy efficiency. In this paper, we consider the system architecture represented by the cloud-backed swarm cognition apparatus. We provide a visionary perspective of the fundamental trade-offs that one must model and interpret in tuning the parameters of this new architecture in a scenario where a swarm of connected cars conducts real-time traffic sign recognition.**

I. INTRODUCTION

There is an incredible growth in commercial investment and public excitement around the topics of unmanned aerial vehicles, autonomous ground vehicles or even smart, connected cars with drivers. Smart ground vehicles, and especially those that are designed to be driver-less, are usually sensor-rich, with strict real-time performance constraints. Camera-equipped drones were primarily used before for collecting surveillance videos that were post-processed in server farms — but increasing new defense and civilian applications require real-time (image) data analytics that drive "instant" actions. So, the net picture that is emerging is that of an increasingly diverse range of computing platforms deployed in Connected and Automated Vehicles (CAVs), that all have one thing in common: namely, the back-end cloud servers that provide services and support in line with the particular application supported by the vehicles at the "edge" of the network.

In a paradigm where vehicle-to-vehicle communications (and collaborative swarm computing) are not utilized, the innovation domain is largely limited to the cloud. The key research issue there is energy-efficient scaling of cloud resources to handle the steady increase in vehicle volume. In the anticipated era of swarm computing at the edge, the cloud scalability pressure is lessened, and the innovation domain shifts to the vehicular swarm computing architecture. There is a promise of meeting real-time deadlines through cheaper (less sensor-rich, less power-hungry, lower performance) vehicle-embedded computing, and fewer power-consuming wireless accesses to the cloud, when the collaborative swarm computing aspect is leveraged.

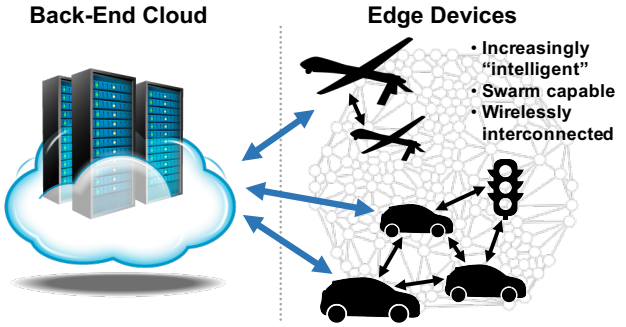


Fig. 1: New generation system architecture to support the escalating growth in cognitive services for Connected and Automated Vehicles, with variable wireless interconnects, real-time constraints and swarm computing capabilities.

In this article, we focus on the system-level architectural implications and innovation priorities in light of the above-mentioned, fast-changing CAV landscape. Given that the new generation system under consideration (see Figure 1) is not just a stand-alone vehicle, nor even a cloud data center — but a highly interconnected and dynamically malleable cloud-backed CAV infrastructure, we see new research opportunities. In particular, the increasing demand for *cognitive*[1] services in future generation CAVs intensifies the challenge of making the right trade-offs in capabilities at the edge versus the cloud, in a dynamic environment where wireless bandwidths are limited and variable. If we are entering a future where CAVs are able to engage in collaborative "swarm" computing to meet real-time deadlines for complex cognitive tasks, how should the system be architected to support such capabilities?

Swarm computing, an approach inspired by the collaborative and decentralized behavior of some systems in nature [1], gains particular importance in the context of the Internet of Things (IoT): with around 80 billion connected devices in the world by 2025 [2], the collective "cognitive" behavior will synergistically emerge from the "swarm" of edge devices and their collaboration. *Swarming* is expected to

This research was developed, in part, with funding from the Defense Advanced Research Projects Agency (DARPA). The views, opinions and/or findings expressed are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. This document is: Approved for Public Release, Distribution Unlimited.

All authors are with the IBM T. J. Watson Research Center, Yorktown Heights, NY, USA, 10598 {ajvega, alperb, pbose}@us.ibm.com

[1]The terms "cognitive computing" and "artificial intelligence" (AI) are used interchangeably throughout this paper.

find a breeding ground in the context of CAVs, since several key aspects of future transportation will not be possible by only amplifying the already-complex on-vehicle sensing and perception capabilities. Examples of this relatively new trend in automotive are the announcements made by some of the top car manufactures, including Honda's "Cooperative Mobility Ecosystem" and its "Safe Swarm" concept to negotiate complex driving situations [3], and Audi's "Swarm intelligence/Car-to-x" services [4]. In particular, *vehicle platooning* has drawn significant attention and research efforts in recent years due to the potential benefits that this approach can offer, including road capacity increase, energy consumption and exhaust emissions reduction, and driving safety improvement [5]. In general, the range of applications that can be potentially enabled through the cooperation of connected vehicles is vast and complex. It is not the focus of this work to delve into them, but detailed surveys related to this area have been presented in [5], [6], [7], [8].

However, most of these forms of cooperative mobility remain in purely information passing or communication relay systems, where vehicles in the swarm share (usually real-time) information among themselves and with the surrounding traffic infrastructure to proactively guide driving decisions. We believe that swarm computing can play a key role in augmenting the "cognitive" capabilities of future CAVs through sharing information inferred from their on-board artificial intelligence/machine learning (AI/ML) models and using it *not only* to guide the driving decisions but also to improve the robustness of those same AI/ML models (in a form of *lifelong learning*). Sudden changes in in-field data patterns that deviate significantly from datasets used during in-factory training or other environmental fluctuations can impact the accuracy of the AI/ML models. Our ultimate goal is to leverage swarm computing principles to develop reliable inference engines for CAVs that do not break in the field, but can keep providing accurate-enough results even in highly-dynamic and changing scenarios.

In the context of the cloud-backed swarm cognition infrastructure depicted in Figure 1, we dwell on the fundamentals of the supporting system architecture and make the following new contributions:

- We formulate a first-order analytical model (CREATE) to understand the fundamental performance trade-offs at the system architectural paradigm depicted in Figure 1 with swarm computing (collaborative) capabilities. Using this model, we provide parametric analysis that brings out the key bottlenecks when it comes to real-time execution.
- We provide an in-depth analysis summary based on the detailed understanding gleaned from the CREATE model. In particular, we provide a visionary position statement about what the priorities are in formulating a scalable, robust cloud-backed swarm cognition system architecture for the future CAV landscape.
- We present a study of the cloud-backed swarm cognition apparatus in the context of real-time traffic sign recognition. We show that, when orchestrated in a proper manner, the swarming operation and back-end cloud elements can improve recognition accuracy and robustness over a case where vehicles recognize road signs in stand-alone mode.

The rest of the paper is organized as follows: Section II breaks down the high-level view expressed in Figure 1 into a more detailed description that is then captured through a first-order analytical model named CREATE with *swarm computing* (collaborative) capabilities. Section III evaluates CREATE in the context of traffic sign recognition for prospective CAVs and presents detailed simulation results. Section IV describes a hypothetical implementation of the on-board swarm cognition architecture. Finally, the future work and our conclusions are presented in Sections V and VI, respectively.

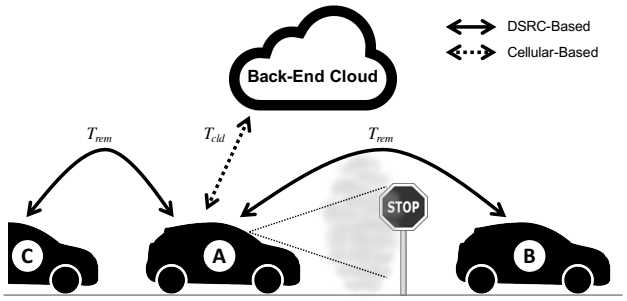


Fig. 2: Illustrative autonomous driving scenario where automobiles depend on accurate and fast road sign recognition.

II. CLOUD-BACKED SWARM COGNITION

This section presents the Computing with Resilient and Efficient Architectural TEchniques (CREATE), a first-order analytical model to understand the fundamental performance and power efficiency trade-offs in scenarios with *swarm computing* (collaborative) characteristics. We analyze a cloud-backed swarm cognition case study around the advanced navigation technology required by prospective Connected and Automated Vehicles (CAVs), with focus on the key challenge of how to derive meaning in real or near real time from the surrounding environment, including road signs and markings, pedestrians, bicycles and other vehicles nearby. This challenge gets exacerbated under particular conditions (like rainy or foggy weather) that prevent easy detection or recognition by the sole use of on-board computation and sensing capabilities [9]. Figure 2 depicts an illustrative scenario where CAVs need to detect and recognize a *stop* sign post in the presence of foggy weather. The vehicles can interact to each other as well as to a back-end cloud infrastructure through *vehicle–to–everything* (V2X) communication protocols [10]. More specifically, a vehicle (for example, car (A)) can send requests to and receive responses from other vehicles through dedicated short-range communication (DSRC) [11], and to/from the cloud through cellular communication, like 5G connectivity [12], [13]. These requests and responses have associated latencies: T_{rem} and T_{cld} in the figure. The ultimate goal for car (A) is to

detect and recognize the approaching *stop* sign within a given worst-case execution time (WCET) regardless of the weather conditions, either by making use of the on-board computing capabilities and/or with the support of other nearby vehicles and the cloud. If car (A)'s on-board AI/ML model cannot recognize the *stop* sign with acceptable confidence, then other nearby cars or the cloud can be contacted to obtain a more accurate answer. Of course, such a request may also fail if either other vehicles cannot provide a more accurate answer, or they are not reachable, or the request/response latencies violate the WCET (i.e. the real-time reaction is not guaranteed).

It is evident from Figure 2 that mobile cognition systems may have to deal with disparate (or variable confidence) inference results from different AI/ML models when presented with the same input data. For example, car (A) in Figure 2 could recognize the *stop* sign with 86% confidence while car (B) could generate the same inference result with a higher 93% confidence[2]. In some cases, 86% confidence can be acceptable for some of the less critical self driving decisions (like taking a road exit which is predicted to be less congested); in cases like timely and accurate recognition of road signs, it is not. In order to study trade-offs like these ones, we propose the CREATE model. Figure 3 presents the application of CREATE to the specific traffic sign recognition case study under discussion.

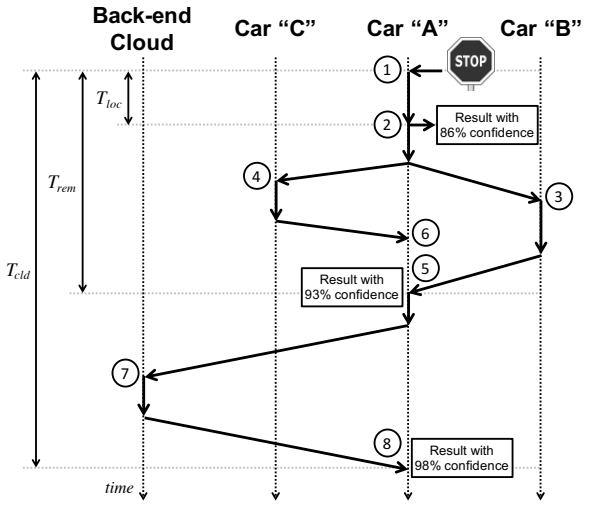


Fig. 3: Cloud-backed swarm cognition (CREATE).

First, the sign recognition AI/ML model in car (A) receives a video frame from the on-board camera(s) with the information of the *stop* sign post (point labeled (1) in Figure 3). The on-board AI/ML model processes the frame and classifies it with, for example, a confidence of 86% (point (2)) which is lower than expected (due to the presence of fog as it is depicted in Figure 2). An 86% confidence is not high-enough for sign recognition and, therefore, car (A) broadcasts a request for a more accurate classification of the *stop* sign. Nearby cars (B) and (C) receive the request (points (3) and (4), respectively) and send back responses that are received by car (A) in points (5) and (6). In this illustrative example, only car (B) provides an answer (for example, with a 93% confidence); car (C) may not have enough on-board computing capabilities to process the request. If this confidence is not high-enough either, then car (A) can "decide" to fall back on the cloud where more precise road sign information is available either from the use of more accurate AI/ML classification models or by accessing cloud-based high-definition digital maps [14], [15]. Car (A) receives the response from the cloud in point (8). The key question here is if the latencies associated with local processing of the road sign (T_{loc}) or obtaining a response from a remote car (T_{rem}) or from the cloud (T_{cld}) violate the WCET or not. In this scenario, this real-time deadline is primarily determined by the distance between car (A) and the road sign, the velocity at which car (A) is approaching the sign, and the coefficient of friction between the tires and the road surface.

To summarize, this illustrative example of the CREATE model exhibits three alternatives:

1) A 86%-confidence result generated in a time T_{loc}.
2) A 93%-confidence result generated in a time T_{rem}.
3) A 98%-confidence result generated in a time T_{cld}.

In this particular example, $T_{loc} < T_{rem} < T_{cld}$. If T_{cld} is smaller than the WCET, then waiting for a highly-accurate response from the cloud is an option. Otherwise, the real-time recognition of the road sign is jeopardized and car (A) would have to fall back to the response from car (B) or its locally-generated inference, even if the associated confidences are lower. In this case, the car will resort to a built-in exception handling mechanism if an approaching road sign cannot be classified with enough confidence, like slowing the car down to a safer velocity or even fully stopping it [16]. It is important to mention that the scenario presented in Figure 3 allows some optimizations, like speculatively broadcasting the request to all the nearby cars and the cloud at the same time or sending the request to specific vehicles (e.g. only to car (B)) to reduce the pressure on the wireless channels. In general, nearby cars in the swarm act as fast repositories while the back-end cloud behaves like a slower albeit more stable one.

We illustrate the use of CREATE to model the latency at which car (A) classifies road signs (with eventual support from nearby cars or the cloud) under different traffic conditions. More specifically, we model the scenario presented in Figure 3 where the goal is to estimate T_{loc}, T_{rem} and T_{cld} to satisfy the following objective function:

$$\begin{aligned} T_{avg} &= Pr_{loc} \times T_{loc} + Pr_{rem} \times T_{rem} + Pr_{cld} \times T_{cld} \\ T_{avg} &< WCET \end{aligned} \quad (1)$$

In Equation 1, Pr_{loc}, Pr_{rem} and Pr_{cld} are the probabilities that the road sign is classified locally, on a nearby car or

[2]Note that these confidence values are for illustration purposes only.

in the cloud, respectively. These probabilities highly depend on the accuracy of the AI/ML models and the accessibility to nearby cars and to the cloud. For example, if car (A) can accurately classify road signs using the on-board model most of the time, then Pr_{loc} dominates the equation over Pr_{rem} and Pr_{cld} in Equation 1. On the other hand, if the on-board model or nearby cars are unable to generate accurate classifications, then Pr_{cld} governs the equation. The other components of the equation are the latencies T_{loc}, T_{rem} and T_{cld}. T_{loc} is basically the time it takes the on-board model to classify an input camera frame. T_{rem} and T_{cld} are the times that are taken by the models on other cars and in the cloud, respectively, to classify an input frame including the associated transfer latencies and queuing overheads[3]. For the sake of illustration, in this study the vehicles and the back-end cloud make use of the same AI/ML model for road sign classification. More specifically, we adopt a convolutional neural network (CNN) similar to the one proposed by Chilamkurthy in [17] to classify road signs from the German Traffic Sign Recognition Benchmark (GTSRB) dataset [18]. Road sign images in the GTSRB dataset have different resolutions and sizes; we conservatively take the largest found image (181 Kbytes) to compute car-to-car and car-to-cloud transfer latencies. We assume dedicated short-range communication (DSRC) between cars and cellular communication to access the back-end cloud with transfer rates of 27 Mbps and 50 Mbps, respectively [19]. We run and characterize the road sign classification CNN model on an Intel Core i7-3740QM CPU running at 3.5 GHz to emulate the back-end cloud and a quad-core 64-bit ARM A57 CPU running at 1.73 GHz to emulate the edge mobile device (car). On average, the CNN model classifies one road sign image in 5.92 ms when running on the cloud and in 135.12 ms when running on the vehicle. Equation 1 can then be rewritten in the following way:

$$\begin{aligned} T_{avg} = {} & Pr_{loc} \times CPU_{loc} \\ & + Pr_{rem} \times \left(\frac{S}{BW_{rem}} + WAIT_{rem} + CPU_{rem}\right) \\ & + Pr_{cld} \times \left(\frac{S}{BW_{cld}} + WAIT_{cld} + CPU_{cld}\right) \end{aligned} \quad (2)$$

CPU_{loc}, CPU_{rem} and CPU_{cld} represent the CPU times of the on-board computer, other cars' on-board computers, and the cloud, respectively. For the sake of illustration, we assume that the on-board computers in the vehicle swarm present similar computation capabilities (i.e. $CPU_{loc} = CPU_{rem}$). Parameter S represents the size of the traffic sign image, while the available wireless bandwidth to other (nearby) cars and to the cloud are captured by BW_{rem} and BW_{cld}, respectively. Parameters $WAIT_{rem}$ and $WAIT_{cld}$ capture the dynamics of the swarm since the number of vehicles and the rate at which they send requests to other vehicles or to the cloud determine the contention experienced by the requests. Due to the complexity associated with the modeling of $WAIT_{dev}$ and $WAIT_{cld}$, we build a general-purpose swarm computing simulator called *SwarmSim*. SwarmSim supports the simulation of arbitrary cloud-backed mobile cognition scenarios with *swarming* aspects, capturing the fundamentals of CREATE. More specifically, the user can create swarms of *agents* that can be either mobile or stationary and deploy them in a user-defined *environment*. The agents and the environment are fully parametrizable; in this study, we define agents that mimic cars moving across a two-dimensional grid-like environment. Table I describes the parameters of the CREATE model along with corresponding values used during the SwarmSim simulations.

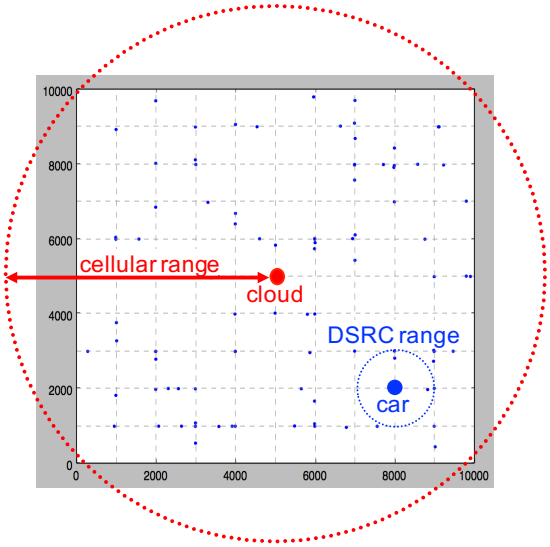


Fig. 4: Screenshot of the two-dimensional grid-like environment modeled using SwarmSim. Dimensions are expressed in meters. Horizontal and vertical lines represent "roads" and points represent cars. Annotations indicate the adopted ranges for DSRC and cellular communication.

We use SwarmSim to simulate scenarios where different numbers of cars ($N = 10, 50, 100$) move across a two-dimensional 10×10-km environment. In other words, we model different car densities to mimic *rural*- and *urban*-like scenarios. The two-dimensional environment is a grid of "roads" (as shown in Figure 4) through which cars move with random velocities (up to 110 km/h, in this particular case study). At road intersections, cars can randomly change the road they are moving through. Cars can process up to two requests in parallel ($PAR_{loc} = PAR_{rem} = 2$), either locally-generated or from other cars (local requests have higher priority than remote ones). The back-end cloud can process a significantly larger amount of simultaneous requests — we model two cases: $PAR_{cld} = 20$ and 40. Traffic sign classification requests take 135.12 ms on a car's CPU and 5.92 ms on the cloud, and confidences associated to classifications performed by cars are randomly generated from a Gaussian distribution with mean $\mu = 0.98$ and standard deviation $\sigma = 0.03$ (running the convolutional neural network every time would throw away the benefits of using SwarmSim for large-scale simulations). In other words, confidences are randomly generated around 98%, since $C_{min} = 98\%$ is

[3]As in any computing system, the on-board vehicle computers and the cloud can process a finite number of request at the same time.

TABLE I: CREATE model parameters.

Parameter	Description	Values
N	Number of vehicles in the swarm.	10, 50, 100
VEL	Velocity at which vehicles move in the environment.	0..110 km/h (random)
DIM_{env}	Environment dimensions.	10 × 10 km
CPU_{loc}	Computation time to classify an input image on the on-board CPU.	135.12 ms
CPU_{rem}	Computation time to classify an input image on a remote vehicles's CPU.	135.12 ms
CPU_{cld}	Computation time to classify an input image on the cloud CPU.	5.92 ms
PAR_{loc}	Number of requests that can be processed simultaneously on the on-board CPU	2
PAR_{rem}	Number of requests that can be processed simultaneously on a remote vehicles's CPU	2
PAR_{cld}	Number of requests that can be processed simultaneously on the cloud CPU	20, 40
C_{min}	Minimum confidence required to accept the AI/ML model output as *valid*.	98%
$WCET$	Worst-case execution time; i.e. maximum amount of time to obtain a valid AI/ML model output for a given input without violating the real-time aspect of the operation.	1 to 10 sec
S	Size of the image (e.g. road sign) being classified.	181 KB [18]
BW_{rem}	Wireless connection bandwidth between nearby vehicles (DSRC-based).	27 Mbps [19]
RG_{rem}	Wireless connection range between nearby vehicles (DSRC-based).	1,000 meters
BW_{cld}	Wireless connection bandwidth between a vehicles and the cloud (cellular-based).	50 Mbps [19]
RG_{cld}	Wireless connection range between a vehicles and the cloud (cellular-based).	10,000 meters
$ITER$	Number of simulated iterations.	100,000

the minimum confidence required to accept a response as *valid* in this illustrative case study. On the other hand, we assume that responses from the cloud are highly accurate (i.e. 100% confidence). During simulation, each car determines its reachable neighbors based on proximity and the coverage of the dedicated short-range communication (DSRC) protocol in use (1,000 meters, in this illustrative case study). Reachable cars at one point in time may become unreachable at the next one due to their displacement. In other words, not all the requests sent to nearby cars will have responses. On the other hand, access to the cloud is guaranteed all the time, as shown in Figure 4. However, due to queuing effects, requests sent to the cloud may also fail to complete. The ultimate goal of this study is to determine and understand the fundamental trade-offs of the cloud-backed swarm cognition approach when applied to traffic sign recognition in latest generation CAVs. And specifically, to determine when and under which circumstances the WCET (i.e. the real-time deadline) is satisfied and when it is not. In practice, the WCET is the maximum time that the car has to *perceive* the traffic sign and *react*, including the interaction with other nearby cars and with the back-end cloud, in order to be able to maneuver (e.g. stop) in a timely manner. Realistic perception-reaction times in the case of a human driver can be in the order of a few seconds[4]. Therefore, we decide to explore WCETs in a similar order of magnitude in our simulations, ranging from 1 to 10 seconds. Finally, we simulate 100,000 iterations for each configuration. In SwarmSim, the granularity of iterations is user-configurable; for the sake of simplicity, we adopt a granularity of one second per iteration.

III. EVALUATION

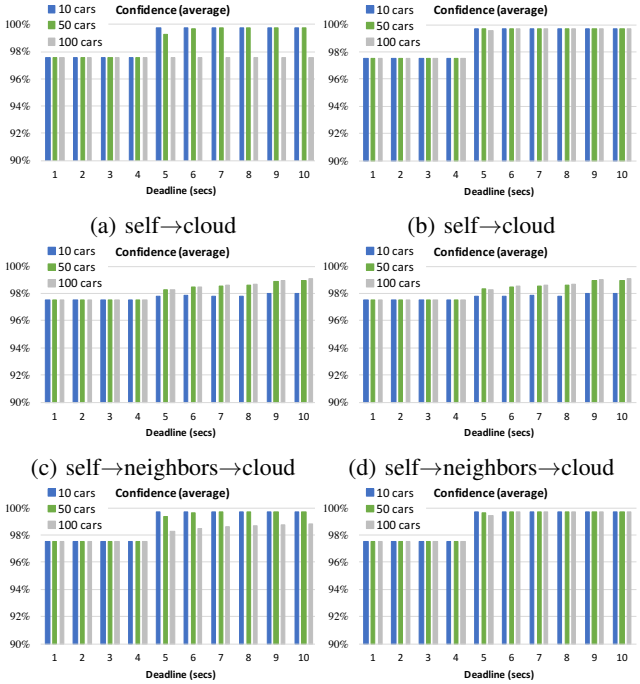
This section presents the results obtained from SwarmSim for the traffic sign recognition case study. We model the following three scenarios:

- **self→cloud** Traffic sign classification is performed on board and, if not successful, sent to the cloud.
- **self→neighbors→cloud** Traffic sign classification is performed on board and, if not successful, sent to nearby cars. If a valid response cannot be obtained from the neighbors either, then it is sent to the cloud.
- **self→neighbors+cloud** Traffic sign classification is performed on board and, if not successful, sent *simultaneously* to nearby cars and the cloud.

The figure of merit is the confidence of the traffic sign recognition process. Figure 5 presents the average confidence as a function of the deadline (WCET), with cloud CPU parallelism (PAR_{cld}) of 20 and 40 in the left and right columns, respectively. At first glance, we observe that the minimum 98% confidence required to consider a classification as *valid* is not satisfied for stringent deadline values below 5 seconds. In this particular study, requests that cannot be satisfied locally (i.e. on board) take *at least* 5 seconds to complete when they are sent to either other cars or to the cloud, as a result of the 1-sec/iteration granularity adopted in SwarmSim[5]. If the deadline is set below 5 seconds, then the requests have to be served locally, with normally distributed confidences around 98%. If the deadline is relaxed, then requests can eventually reach out to other cars and/or the cloud to obtain higher-confidence responses. Figure 5 shows that the density of cars can also affect confidence due to an increase in the number of requests generated. For example, the average confidence does not reach the minimum 98% when 100 cars are modeled in the *self→cloud* scenario and a maximum cloud parallelism of 20 (Figure 5a). In this case, the requests that cannot be satisfied locally are sent

[4]A perception-reaction time of 1.5 seconds is considered standard [20].

[5]The adopted simulation parameters are for illustration purposes only. Therefore, latencies are expected to be smaller in a real scenario.

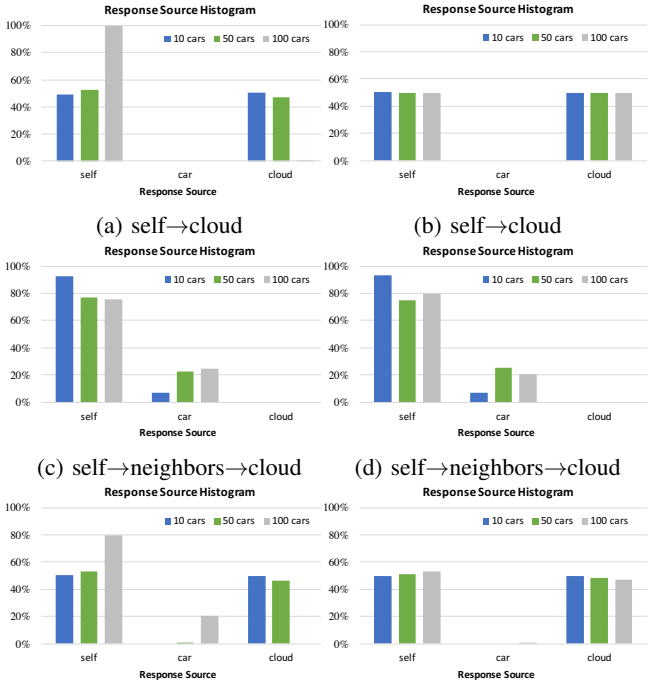


(a) self→cloud (b) self→cloud

(c) self→neighbors→cloud (d) self→neighbors→cloud

(e) self→neighbors+cloud (f) self→neighbors+cloud

Fig. 5: Average traffic sign recognition confidence for different numbers of cars in three different situations: *self→cloud*, *self→neighbors→cloud* and *self→neighbors+cloud*. Left column: cloud parallelism of 20. Right column: cloud parallelism of 40.



(a) self→cloud (b) self→cloud

(c) self→neighbors→cloud (d) self→neighbors→cloud

(e) self→neighbors+cloud (f) self→neighbors+cloud

Fig. 6: Histograms of sources of responses for different numbers of cars in three different situations: *self→cloud*, *self→neighbors→cloud* and *self→neighbors+cloud*. Deadline: 5 seconds. Left column: cloud parallelism of 20. Right column: cloud parallelism of 40.

directly to the cloud (nearby cars are not leveraged). The cloud quickly becomes saturated with requests, and most of them end up expiring. On the other hand, if nearby cars can be leveraged (Figures 5c and 5e), then better responses are obtained from them, even if the cloud is saturated. These preliminary results indicate that future purely-cloud-based solutions for automotive may face scalability problems as the number of vehicles grows. We argue that hybrid approaches that combine local vehicle-to-vehicle swarming and back-end cloud support can result in more robust solutions for scenarios like the one studied here.

The sources of responses are presented in Figure 6 for a deadline of 5 seconds. We observe that the cloud never has the chance to provide responses due to saturation when its CPU parallelism is 20 and the number of cars is 100 (Figures 6a and 6e). The cloud never provides responses in the *self→neighbors→cloud* scenario either, regardless of its CPU parallelism (20 or 40), because of the latency added due to going to neighboring cars first (Figures 6c and 6d). The overall confidence shown in Figure 5 along with the sources of responses presented in Figure 6 provide a better picture of some of the possible bottlenecks and associated trade-offs in cloud-backed swarm cognition scenarios. AI/ML systems that form the core foundation of today's AI systems are quite fragile when it comes to sudden changes in in-field data patterns that deviate significantly from datasets used during in-factory training. How to create reliable inference engines (classifiers) that do not break in the field, but can keep providing accurate-enough results even in highly-dynamic and changing scenarios is a key problem in AI today. We claim that solutions will have to combine local swarming as well as cloud support aspects (and do it with a high degree of adaptation) in order to cope with environmental fluctuations (like congested wireless links or limited computational capacity) to guarantee a certain level of robustness — measured as *confidence* in this particular case study.

IV. HIGH-LEVEL ON-BOARD ARCHITECTURE

Previous sections discuss a swarm cognition framework based on single-modal data, where traffic sign recognition is resolved using only image frames provided by an on-board video camera. This "simplification" allows us to keep the focus on the swarming aspect of the problem without having to delve into the complexity of sensor fusion. In practice, CAVs rely on multi-modal sensory data that is combined to generate a richer "picture" of the surrounding environment. The proposed cloud-backed swarm cognition approach can also work in the case of multi-modal sensor fusion. Figure 7 presents a hypothetical high-level view of the proposed approach. Multiple sources of sensory data go through a sensor fusion stage before feeding the on-board inference engine (box labeled (1)). The inference engine can use a multiplicity of deep learning models to classify the provided input and generate an inference result with

an associated confidence. The inference engine can also broadcast a request for a more accurate classification to other nearby vehicles and/or to the back-end cloud through V2V/V2I communication. This request can be broadcast either in parallel to the execution of the on-board inference engine or after the confidence of the on-board inference result is evaluated, as it is discussed in previous sections. The remote responses received (if any) also include inference results and associated confidences. All of them along with the locally-generated one are compared and the best result is selected (box labeled (2)). The *best* inference result is not necessarily the one with the highest confidence. A more robust approach consists in taking the most frequent one provided they satisfy a minimum confidence level — for example, 8 out of 10 cars respond "saying" that the sign is a *stop* sign, and all these 8 responses satisfy a minimum confidence. Finally, the selected inference result is injected into the *action engine* to control the vehicle, and it is also used to update the on-board inference engine (box labeled (3)). If the inference engine is composed of deep learning models, then the selected inference result along with the associated input sensory data are used to update the models parameters (*weights*) using, for example, stochastic gradient descent. This is a key aspect of the proposed approach that enables continuous improvement of the vehicle's "cognitive" capabilities and, consequently, adaptive operation.

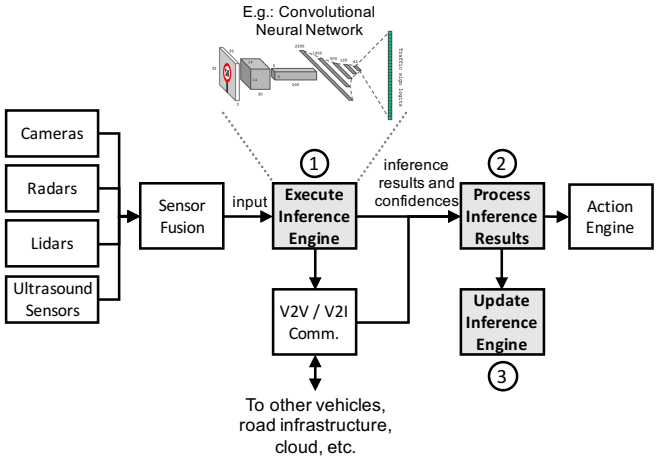


Fig. 7: High-level view of a possible on-board swarm cognition architecture.

It is important to note that this section provides a high-level hypothetical implementation of the swarm cognition architecture, without considering aspects like security and privacy during the interaction with other vehicles and with the back-end cloud. Also, Figure 7 assumes that the on-board inference engine can be *easily* updated once the best inference result is known. In practice, the update of the on-board inference engine is not a straightforward process since it may result in a degradation of the factory-trained deep learning models (e.g. due to overfitting). These topics are out of the scope of this work and they will be discussed in future publications.

V. ONGOING AND FUTURE WORK

Cloud-backed swarm cognition is a vast research area with several aspects that go beyond *just* performance or accuracy. This section elaborates on some of the key issues in which we focus our current research efforts. Unfortunately due to space limitations we cannot discuss them here, but we plan to include them in an extended version of this paper.

We will extend CREATE to explore the fault-tolerance (or system resilience) limits of cloud-backed swarm cognition systems by injecting statistical, parametric uncertainties into our model. Based on the analysis of these results, we plan to devise specific resilience features into our model.

System resilience is particularly relevant when it comes to ensure adaptive operation of the swarm cognition system. For example, Connected and Automated Vehicles (CAVs) that operate in swarm mode should be able to cope with other defective or malicious vehicles and be able to react to circumstances that are outside their in-factory training. Some elements of adaptation have been recently explored in the context of distributed consensus for automotive [21], and the imperative need for systems with *lifelong learning* capabilities [22] is reflected (for example) in a currently ongoing DARPA program [23]. We aim at integrating system-level resilience and adaptability into our CREATE model.

We plan to formulate and develop the mathematical framework required to *formally* understand the system architecture modeled by CREATE. We envision it to be founded on the mathematical framework of *distributed consensus* algorithms, which plays a key role in cloud-backed swarm cognition applications. The distributed consensus theory does not contemplate *at the same time* the real-time and resilient operation of these algorithms in highly-dynamic scenarios (like those faced by CAVs), except for a few partially related works [24], [25], [26], [27]. We will provide the existing distributed consensus theory with the elements to be used in cloud-backed (real-time) swarm cognition environments.

Finally, we are working on ultra low voltage hardware accelerators and hardware support for deep learning applications as part of the DARPA PERFECT program [28], [29]. In this context, we already studied the fragility of deep learning models when it comes to errors in the models parameters (*weights*) [30]. Along these lines, cloud-backed swarm cognition constitutes a complementary approach for reliable and energy efficient operation of the deep learning models.

VI. CONCLUSIONS

The new generation CAV landscape is rapidly becoming a highly interconnected and dynamically malleable cloud-backed vehicle infrastructure. This new generation system architecture is heterogeneous and wirelessly interconnected with variable reliability and bandwidth characteristics, and stringent real-time constraints. In this context, we envision a future where CAVs are able to engage in collaborative

"swarm" computing to meet real-time deadlines for complex cognitive tasks. In this visionary paper, we study a cloud-backed mobile cognition paradigm with swarm computing aspects. We believe that cognitive solutions for CAVs that do not consider this novel architectural vision will fall short of the level of *adaptation* that future vehicles will require to operate properly in highly-dynamic environments.

ACKNOWLEDGMENTS

We are grateful to all our colleagues within the IBM-led project titled "Efficient Resilience in Embedded Computing," sponsored by DARPA under its PERFECT program. This includes researchers from Stanford University, Harvard University and University of Virginia – in addition, of course, to many other IBM research personnel, including Dr. Jeff Burns, who has been an inspirational supporter of the cloud-backed swarm cognition project.

REFERENCES

[1] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. New York, NY, USA: Oxford University Press, Inc., 1999.

[2] M. Kanellos, "152,000 smart devices every minute in 2025: IDC outlines the future of smart things." http://www.forbes.com/sites/michaelkanellos/2016/03/03/152000-smart-devices-every-minute-in-2025-idc-outlines-the-future-of-smart-things/#1d4cd6b069a7, March 2016.

[3] Honda Motor Co., Ltd., "Honda introduces "cooperative mobility ecosystem" at CES 2017." http://world.honda.com/news/2017/c170106eng.html.

[4] Audi AG, "Swarm intelligence/"car-to-x"." https://www.audi-mediacenter.com/en/connectivity-techday-6597/swarm-intelligencecar-to-x-6602.

[5] D. Jia, K. Lu, J. Wang, X. Zhang, and X. Shen, "A survey on platoon-based vehicular cyber-physical systems," *IEEE Communications Surveys Tutorials*, vol. 18, pp. 263–284, First Quarter 2016.

[6] J. E. Siegel, D. C. Erb, and S. E. Sarma, "A survey of the connected vehicle landscape–architectures, enabling technologies, applications, and development areas," *IEEE Transactions on Intelligent Transportation Systems*, vol. PP, pp. 1–16, October 2017.

[7] T. L. Willke, P. Tientrakool, and N. F. Maxemchuk, "A survey of inter-vehicle communication protocols and their applications," *IEEE Communications Surveys Tutorials*, vol. 11, pp. 3–20, June 2009.

[8] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil, "Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions," *IEEE Communications Surveys Tutorials*, vol. 13, pp. 584–616, July 2011.

[9] N. Boudette, "5 things that give self-driving cars headaches." https://nyti.ms/2jVRRLn, 2016.

[10] K. Abboud, H. A. Omar, and W. Zhuang, "Interworking of DSRC and cellular network technologies for V2X communications: A survey," *IEEE Transactions on Vehicular Technology*, vol. 65, pp. 9457–9470, December 2016.

[11] Wikipedia, "IEEE 802.11p — Wikipedia, the free encyclopedia." https://en.wikipedia.org/w/index.php?title=IEEE_802.11p&oldid=774355529, 2017.

[12] Y. Gao and A. Wachtel, "Connecting cars on the road to 5G." http://www.huawei.com/en/xlabs/insights-whitepapers/huawei-whitepaper-connected-car-on-the-road-to-5g, 2016.

[13] Mobile Europe, "5G connected car trial tops 3.6GBps using beamforming tech." http://www.mobileeurope.co.uk/press-wire/5g-connected-car-trial-tops-3-6gbps-using-beamforming-tech, 2017.

[14] "HERE." https://here.com.

[15] B. Berman, "Whoever owns the maps owns the future of self-driving cars." http://www.popularmechanics.com/cars/a21609/here-maps-future-of-self-driving-cars/, 2016.

[16] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "On a formal model of safe and scalable self-driving cars," *CoRR*, vol. abs/1708.06374, 2017.

[17] S. Chilamkurthy, "Keras tutorial – traffic sign recognition." https://chsasank.github.io/keras-tutorial.html, 2017.

[18] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, vol. 32, pp. 323–332, 2012.

[19] Z. Hameed Mir and F. Filali, "LTE and IEEE 802.11p for vehicular networking: a performance evaluation," *EURASIP Journal on Wireless Communications and Networking*, vol. 2014, no. 1, p. 89, 2014.

[20] U.S. Department of Transportation – National Highway Traffic Safety Administration (NHTSA), "Why your reaction time matters at speed." https://one.nhtsa.gov/nhtsa/Safety1nNum3ers/august2015/S1N_Speeding-August2015_812008.pdf, 2015.

[21] K. Saulnier, D. Saldaña, A. Prorok, G. J. Pappas, and V. Kumar, "Resilient flocking for mobile robot teams," *IEEE Robotics and Automation Letters*, vol. 2, pp. 1039–1046, April 2017.

[22] S. Thrun, *Lifelong Learning Algorithms*, pp. 181–209. Springer US, 1998.

[23] DARPA, "Toward machines that improve with experience." https://www.darpa.mil/news-events/2017-03-16, 2017.

[24] J. F. Hermant and G. L. Lann, "Fast asynchronous uniform consensus in real-time distributed systems," *IEEE Transactions on Computers*, vol. 51, pp. 931–944, August 2002.

[25] H. Moser and U. Schmid, "Reconciling fault-tolerant distributed algorithms and real-time computing," *Distributed Computing*, vol. 27, pp. 203–230, June 2014.

[26] G. L. Lann, "Asynchrony and real-time dependable computing," in *Proceedings of the 8th International Workshop on Object-Oriented Real-Time Dependable Systems*, WORDS 2003, pp. 18–25, January 2003.

[27] F. Muñoz, E. S. Espinoza Quesada, H. La, S. Salazar, S. Commuri, and L. Garcia Carrillo, "Adaptive consensus algorithms for real-time operation of multi-agent systems affected by switching network events," *International Journal of Robust and Nonlinear Control*, vol. 27, no. 9, pp. 1566–1588, 2017.

[28] "Power efficiency revolution for embedded computing technologies (PERFECT)." https://www.darpa.mil/program/power-efficiency-revolution-for-embedded-computing-technologies.

[29] R. Bertran, P. Bose, D. Brooks, J. Burns, A. Buyuktosunoglu, N. Chandramoorthy, E. Cheng, M. Cochet, S. Eldridge, D. Friedman, H. Jacobson, R. Joshi, S. Mitra, R. Montoye, A. Paidimarri, P. Parida, K. Skadron, M. Stan, K. Swaminathan, A. Vega, S. Venkataramani, C. Vezyrtzis, G. Y. Wei, J. D. Wellman, and M. Ziegler, "Very low voltage (VLV) design," in *Proceedings of the 35th IEEE International Conference on Computer Design*, ICCD 2017, pp. 601–604, November 2017.

[30] A. Vega, S. Eldridge, A. Buyuktosunoglu, and P. Bose, "Mobile cognition: Enabling deep learning computing in the internet of things era," in *Proceedings of the Government Microcircuit Applications and Critical Technology Conference*, GOMACTech 2017, March 2017.