To operate intelligent vehicular applications such as automated driving, mechanisms including machine learning (ML), artificial intelligence (AI), and others are used to abstract knowledge from information. Knowledge is defined as a state of understanding obtained through experience and analysis of collected information, and it is promising for vehicular applications. However, to achieve its full potential, it requires a unified framework...
that is cooperatively created and shared. This article investigates the meaning and scope of knowledge as applied to vehicular networks and defines a structure for vehicular knowledge description, storage, and sharing. Through the example of passenger-comfort-based automated driving, we expose the potential benefits of such knowledge structuring for network load and delay.

Background
Over the last decade, we have witnessed the evolution of vehicular networking from vehicular ad hoc networks that enable spontaneous, direct communications among vehicles to connected vehicles generalizing information exchange among vehicles and infrastructure. Vehicular networking has developed as an enabler of innovative applications intended to improve traffic safety, reduce congestion, and even provide infotainment on board. Early applications were designed to provide information only to drivers, delegating any decision making to them.

However, in recent ambitious applications, such as automated driving or platooning, simple information treatment and forwarding mechanisms are no longer sufficient. Instead, decision making is based on models of the environment built from much more sizable sets of input information. Models are designed to learn from experience rather than react to static input signals. In this context, models have the potential to reduce the load and delay in vehicular networks, as key content is extracted from larger sets of input information. What is more, by favoring the distribution of models over static content, information privacy is improved. However, unlike with static content sharing, mechanisms to name, localize, and network/offload the knowledge creation capacities of models in vehicular networks are lacking.

The existing literature in the information science domain covers conceptual definitions of data, information, and knowledge [1]. In this article, for the sake of clarity, we make similar distinctions among these categories. As illustrated in Figure 1, the most fundamental element is data, which we define as an atomic value with a unit, e.g., 30 kph. Next, information is built by aggregating pieces of data that describe a situation [e.g., (17:00, 30 kph), indicating a vehicle’s speed at a given time]. On top of information lies knowledge, which describes general patterns and relationships obtained through the analysis of sets of information. For example, clustering or classification algorithms can be used to extract hidden relationships within a set. As an example, (17:00, 30 kph) ⇒ SCHOOL_RUSH_HOUR is knowledge associating a time and speed information with a context, e.g., the end of the school day.

Various techniques, such as AI, ML, or formal language, have been used to extract knowledge in vehicular contexts through the analysis of various sources of information. For example, Ruta et al. [2] used sensor information from multiple cars in a common geographical area to compose knowledge to recognize the high-level context of driving. Qi et al. [3] applied both learning algorithms and edge computing units offloading to provide optimal caching of high-level connected driving services to vehicles (such as image auto annotation or locally relevant recommendations). Khan et al. [4] applied deep learning to learn the transmission patterns of neighboring vehicles and paved the way for fewer packet collisions.

Regardless of the technique, extracting knowledge from information is a complex and expensive process, but the generated knowledge may be beneficial to other vehicles. So far, each vehicle remains autonomous for its knowledge building, requiring highly specialized algorithms and a large amount of input information, which could potentially be sourced from multiple different vehicles. This can be seen as a significant overhead, considering that knowledge can be shared but not individually recreated. As a reaction, research has recently focused on defining a knowledge-centric approach to networking, where information would no longer be the main focus. Instead, knowledge would be created by nodes in the network and directly stored and shared among them. Wu et al. [5] described the concept of a knowledge-centric networking (KCN) framework, separated into three building blocks: knowledge creation, composition, and distribution. A literature survey on the means of creating and distributing knowledge has been performed. However, the concept of knowledge remains abstract, and its implementation or format is left for future work.

The contributions of this article are as follows. Vehicular knowledge networking (VKN), a KCN framework applied to vehicular networks, is presented. It defines a common architecture for knowledge description, which is needed for the subsequent storage, composition, and exchange with other connected vehicles. As such, VKN is a framework that makes possible performance improvements in other applications. In a passenger-comfort-based rerouting application, we evaluate the load impact of VKN knowledge distribution in vehicular networks compared with information-centric networking (ICN)-based approaches and show an overhead reduction of 14 to 40%, depending on the network topology, through cooperative knowledge building and sharing.

Figure 1 The relationship of data, information, and knowledge.
Vehicular Information and Knowledge

In this section, we first describe the current forms of information in vehicular networks as well as various standards for information storing and sharing. Then, we build on this understanding to define a format for vehicular knowledge representation.

Information in Vehicular Networks
Nodes of the vehicular network may exchange diverse types of information, including but not limited to
- safety notifications, e.g., accidents or road conditions
- vehicle state information and sensor measurements
- navigation information, e.g., maps and road or parking data
- information on topics such as weather or traffic flow
- road-related information, e.g., gas station opening times
- multimedia content for user infotainment.

In European Telecommunications Standards Institute standards, the storage of information in connected vehicles is performed inside the local dynamic map (LDM) information base, which is divided into four layers:
1) permanent static data, i.e., map data
2) temporary static data, i.e., roadside infrastructure
3) temporary dynamic data, e.g., roadblocks and signal phases
4) highly dynamic data, e.g., vehicles and pedestrians.

The LDM provides a standard approach for storing information but not for naming it. Generally, any information may be stored in the LDM as long as it is labeled with a space–time area of relevance. This can lead to a lack of interoperability between the content generated by different providers. To tackle this issue, semantic standards have been developed to provide nodes with a common language and avoid redundancy. For example, the vehicle signal specification (VSS) and ontology [6] provide a standard way to address the state of vehicle components, e.g., steering wheels or window opening. Moreover, standard safety messages, such as the cooperative awareness message and decentralized environmental notification message in Europe or the basic safety message in the United States, have been defined to describe various types of information and events.

Finally, after it has been sensed and stored, information is spread within the vehicular network. In most vehicular applications, the information itself is more valuable than its source because it describes road events. Routing algorithms have thus been developed that focus on the information being shared rather than the host. ICN is a networking paradigm that may be suitable for some vehicular applications [7]. Rather than sending a request to a specific host, a request is disseminated to fetch specific information identified by a unique content name.

Knowledge Networking in Vehicular Networks
Compared with information, knowledge is condensed while maintaining reusability across different contexts. As such, we envision a shift away from the information-based architecture to the benefit of knowledge in future vehicular networks.

We summarize the related works on VKN. The concept of KCN was introduced in [5]. As with information, mechanisms must be developed to create, compose, store, and distribute knowledge. Applications have been studied that use the concepts of KCN. In [8], knowledge is created as an ML model predicting the video playback pattern of users. The knowledge is sent to edge servers to optimize video caching. In [9], statistical knowledge about network topology is used to optimize routing in unmanned aerial vehicle fleets.

While the concepts of KCN have been described in [5] and applied for specific applications in works such as [8] and [9], no formal structures for knowledge representation or protocols for knowledge distribution have been described. The lack of a generic framework to describe, store, and share knowledge prevents interoperable applications of KCN and hinders the potential of VKN.

In [10], knowledge is expressed and composed as deep learning models. In this article, we formalize a KCN framework and present structures to define and describe a generic form of knowledge that is not specialized in deep learning models. Based on the description of knowledge, we introduce protocols to cooperatively create, exchange, and localize knowledge following the dynamically evolving needs of vehicles.

Vehicular Knowledge Representation
We understand knowledge as an abstract content obtained from the analysis of larger sets of information [1]. Knowledge can be extracted from information using ML algorithms and divided into three classes.

1) Supervised learning applies to classification or regression. A model is trained based on a number of samples of the form (information, class) for classification or (information, value) for regression. Knowledge is extracted as the relationship between the information and its associated class, i.e., the function that takes information as an input and returns its estimated class.

2) Unsupervised learning extracts clusters of similar items in a set of information. It creates knowledge by exposing the relationships among information items and sorting them into different clusters.

3) Reinforcement learning, finally, can be used by an agent to learn the optimal behavior to adopt in a context of interaction with an environment to maximize a user-defined reward.

A trained ML model is a piece of knowledge able to return synthetic knowledge from input information. The knowledge extracted through learning techniques can be further leveraged through knowledge composition methods, where existing knowledge is further analyzed/collated to produce new knowledge. For instance, if a user wants to avoid traffic congestion, the system needs to first detect the
congested zones and decompose the necessary factors, including current location, destination, and estimated route/arrival time based on the current traffic. In this case, in addition to knowledge creation, the knowledge composition also collates some information and/or other knowledge, such as closed roads and/or construction zones, so as not to exacerbate the congestion.

Thus, the word **knowledge** can refer to both 1) algorithms able to synthesize sets of information into pieces of knowledge, which we refer to as **knowledge models**, and 2) abstracted information obtained by applying models, which we refer to as **knowledge samples**. Both aspects should be considered as we describe a formalization of knowledge definition, storage, and distribution in vehicular networks. Figure 2 illustrates when and how knowledge is handled in vehicular networks for safety and driving-related applications. On the left of the figure, information is meant to be stored in the LDM and has a time and area of validity. For the sake of interoperability with other vehicles, we consider it to be named and structured following well-known constraints, e.g., the VSS specification.

A knowledge model, typically implemented as a trained ML algorithm, takes information as the input and produces output knowledge. In Figure 2, we distinguish two aspects of a knowledge model: semantic description and bytecode. The semantic description is used to describe the unique name, version, necessary input, produced output, and potential preconditions necessary to apply the model. We define the bytecode of a model as the executable file that produces an output from a well-formed set of input information. We identify three possible outputs to a model. The model may output another knowledge model, parameterized by the inputs of the original model. Alternatively, a knowledge model may output an actuation signal or a knowledge sample that, like the information used to produce it, has a time and area of validity. While it is abstract and obtained through analysis, a knowledge sample is structured similarly to information. As a consequence, it can be fed as the input into another knowledge model, generating new composed knowledge.

Figure 3 displays an application of this definition of knowledge related to the estimation of passenger comfort on board a highly automated vehicle (HAV). The top part of the figure describes a knowledge model, **model.env_comfortable**, to infer a value of passenger comfort from three road-related inputs: traffic conditions, visibility, and two-wheeler concentration. Then, in the bottom part, **model.fetch_driving_behavior** takes generic contextual information as the input, namely, the obtained comfort level and the given town of application of the model. Based on this input, it tailors the parameters of a personalized output knowledge model to provide real-time driving assistance to the ego vehicle, optimized depending on the requested comfort level and town. We return to the illustrated models in the “Application: Passenger Comfort-Based Driving” section, where we describe an application of VKN.

**Aspects of VKN**

Technologies such as vehicular clouds, fog computing, and edge computing provide support for knowledge storage and distribution within vehicular networks [11]. Nevertheless, a challenge for achieving knowledge networking is related to the identification, naming, and localization of knowledge. For example, upon reaching a new city, e.g., in a foreign country where human drivers behave differently, an HAV might need knowledge about how to drive there. This is a complex task due to the following:

- The requested knowledge should be identified and named. For example, should the requested knowledge involve urban driving, intersections only, and/or handling of two-wheelers?

---

**Figure 2** The vehicular knowledge ecosystem. LDM: local dynamic map.
Then, the requested knowledge should be located, although it is not straightforward to determine who owns the knowledge. It could be located within a vehicular cloud or an edge unit. Due to the dynamic aspect of vehicular networks as well as optimizations in knowledge caching [12], a knowledge discovery or subscription mechanism is required.

Vehicular clouds and edge computing units are supporting technologies that enable efficient knowledge sharing and storage. VKN as a framework supports knowledge identification, cooperative creation, and localization mechanisms. As such, it can be implemented on top of such architectures to enable efficient knowledge networking in vehicular networks.

**Knowledge Description and Storage**

The creation of knowledge in vehicular networks takes place at two levels:

- Using ML algorithms, automakers and organizations train and provide models capable of generating knowledge from a set of inputs (information or knowledge samples).
- The models are applied to real inputs, returning new knowledge samples or models.

In some cases, automakers keep models proprietary intentionally to protect their competitiveness in the market. However, in other cases, as considered in [10], a lack of cooperation in knowledge model building may lead to the inefficient use of resources. Similar knowledge models are likely to be independently trained by competing entities, leading to redundant computations. Nevertheless, no common format to describe the input, output, and preconditions of a model is provided by training entities, preventing the cooperative use of knowledge models by a larger number of nodes.

To tackle these issues, we separate the semantic description of a model from its actual execution place, as shown in Figure 2. A knowledge model description formally states 1) a unique name and version code for the model and 2) the input, output, and preconditions to its application. This is lightweight content that is shareable with multiple nodes. The knowledge model’s bytecode is the executable file performing the creation of knowledge samples. Even if a vehicle is not in possession of a model’s bytecode, it may request knowledge creation from another node, following the constraints detailed in the model description. In addition to using unique names and version codes for each model, knowledge synchronization mechanisms should be defined to ensure that two remote nodes have the same understanding of a piece of knowledge. This aspect is further detailed in the “Knowledge Routing and Synchronization” section.

As a requirement for model description, the inputs and outputs of a model should be named according to standard semantics specifications, such as VSS [6]. By consulting the specification for the name associated with each input or output, nodes are able to deduce the format of information and knowledge samples required to apply the model. Moreover, preconditions to the model application may be set, e.g., limited to a given town.

![Figure 3](image-url)

*Figure 3* Knowledge models for passenger comfort handling.
A candidate model description language matching these requirements is OWL-S [13], which was originally developed for automatic web services discovery, composition, and invocation. The process model standard of OWL-S provides a means of description for the set of input, output, and preconditions of a model. Figure 4 provides an example of an OWL-S description of the model.env_comfort model introduced in Figure 3.

As part of VKN, we separate the storage of models’ descriptions and bytecodes. Figure 5 illustrates the onboard unit of a connected vehicle. The facilities layer contains the LDM, which is able to store information as well as knowledge samples obtained from abstraction. As part of VKN, we add a knowledge layer as an interface between the applications and information storage.

In a knowledge base (KB), a list of known knowledge model descriptions is stored.

A local storage in the knowledge layer may store knowledge model bytecodes. The stored bytecodes are independent of the model descriptions stored in the KB.

As depicted in Figure 5, the KB is connected both with the ego vehicle’s onboard local model storage and remotely with the KBs of other vehicles. As such, it is responsible for the orchestration of knowledge creation in vehicular networks. To create knowledge, access to both a model’s bytecode and input is needed. If the relevant input is stored locally in the LDM, the KB can obtain the model’s bytecode with the right version code through local storage, if available, or by requesting a remote KB.

Another option to perform knowledge creation, especially if no relevant input is locally available, is to forward a request for remote knowledge creation to another vehicle that possesses the relevant input. The remote vehicle can then, in turn, issue a request for the required model if it is not locally stored. The modalities of such knowledge sharing requests are described in the “Knowledge Distribution” section. This allows for a flexible framework that is able to expand or limit the creation and distribution of knowledge within a certain group of vehicles, as needed. It is also responsible for knowledge synchronization between nodes.

**Knowledge Distribution**

As we separate knowledge models’ bytecodes and their descriptions, a

---

Figure 4: The comfort model description structure in OWL-S.

```
1 <AtomicProcess ID="model.env_comfort:1.1">
2  <hasInput resource="#traffic" />
3  <hasInput resource="#visibility" />
4  <hasInput resource="#twoWheelers" />
5  </AtomicProcess>
6  <Input ID="traffic">
7   <parameterType resource="#Road.Traffic" />
8  </Input>
9  <Input ID="visibility">
10  <parameterType resource="#Road.Visibility" />
11  </Input>
12  <Input ID="twoWheelers">
13   <parameterType resource="#TwoWheelers.Concentration" />
14  </Input>
15  <Output ID="comfort">
16   <parameterType resource="#Road.ComfortLevel" />
17  </Output>
```

---

Figure 5: The onboard storage of knowledge and information.
structural need appears for the distribution of knowledge. Nodes of the vehicular network are interconnected, and knowledge creation may be the product of cooperation among multiple nodes.

We consider two contexts for knowledge distribution within vehicular networks:

1) the distribution of knowledge models, able to produce new knowledge samples
2) the distribution of knowledge creation capacities as a means of outsourcing knowledge sample creation to remote vehicles.

As an example, we consider a vehicle $v$ about to drive in an unknown environment, e.g., to cross a new city. The VKN should allow two types of knowledge sharing for $v$. It may 1) request knowledge directly as a model advising a driving behavior based on the local context or 2) request the generation of a knowledge sample by a remote vehicle, e.g., the generation of the estimated driving comfort based on the local context.

Knowledge Model Request
We define protocols to request and retrieve knowledge models in vehicular networks. Models may be requested based on 1) their names, if known by the requester, or 2) their input and output parameters to discover unknown knowledge.

The issued request takes the following pseudocode form:

1) REQUEST MODEL name: [model_name]  
   CONTEXT version ≥ 1.0  
   last_update: [time]
2) REQUEST MODEL output: Road.ComfortLevel  
   CONTEXT last_update: [time]  
   spatial_relevance: [location]

The request is distributed in the vehicular network following a process that we describe next. A response to the issued request by a remote vehicle could take the form:

1) REQUESTED MODEL name: [model_name]  
   CONTEXT version ≥ 1.0  
   last_update: [time]
   RESPONSE bytecode: [bytecode]
2) REQUESTED MODEL  
   output: Road.ComfortLevel  
   CONTEXT last_update: [time]  
   spatial_relevance: [location]
   RESPONSE name: [model.env_comfort:1.1]  
   model_description: [input, outputs ...]  
   bytecode: [bytecode]

Knowledge Application Request
Then, vehicles may request the remote application of a knowledge model within a given context to retrieve knowledge samples without performing local input data collection and computation. We give an example request and response for the remote creation of knowledge about the driving comfort level in a distant area:

- APPLY model.env_comfort:1.1  
  IN [location]
- COMPUTED model.end_comfort:1.1  
  IN [location] BY [node_address]  
  RESULT Road.ComfortLevel: FAIR

Knowledge Routing and Synchronization
To route knowledge requests in vehicular networks, we extend existing content-centric networking (CCN) interests-based routing mechanisms. CCN is an implementation of the ICN paradigm.

When a vehicle $v$ issues a knowledge creation request, the following steps occur:

1) The request is transmitted to an initial selection of remote nodes.
2) When receiving a knowledge creation request, a remote node checks i) whether it owns a model matching the request and ii) whether the context faced by the remote node matches the request.
3) If the conditions are matched, the remote node computes and returns the requested knowledge to $v$. Otherwise, the request is further transmitted to another neighboring node.

Finally, content synchronization mechanisms implemented as part of ICN, as surveyed in [14], can be adapted for knowledge. Models could be divided between 1) ready-trained models, whose synchronization should follow described mechanisms, and 2) models being trained, where no fixed version of the model is distributed. In that case, synchronization could take the age of the last contribution to the model into account. Similarly, transmission delay constraints are challenging and should be addressed through caching mechanisms, e.g., using vehicular clouds [11].

Application: Passenger-Comfort-Based Driving
As an application of VKN and to show the potential benefits for vehicular networks, we investigate the use case of passenger-comfort-based automated driving. We describe a simple model using rule-based semantics as an example. However, the VKN framework supports a greater complexity on knowledge definition and its composition as, for example, ML algorithms. Then, we evaluate the overhead associated with comfort knowledge distribution for both ICN- and VKN-based approaches.

Comfort Knowledge Models
As an example, we define a simplified knowledge model, model.env_comfort, to determine the level of passenger comfort in a given area, as introduced in Figure 3. The model reads a set of area-related inputs with semantically defined names:

1) the current traffic conditions,  
   $tr \in \text{Road.Traffic} \equiv [\text{FLUID, CONGESTED}]$
2) the visibility in the area, $v \in \text{Road.Visibility} \equiv [\text{CLEAR, OBSTRUCTED}]$
3) the concentration of two-wheelers in the surroundings, $c_{tw} \in \text{TwoWheelers.Concentration} \equiv \{\text{HIGH, MEDIUM, LOW}\}$.

The model outputs a discrete qualification of the level of comfort associated with driving in the area as $cft \in \text{Road.ComfortLevel} \equiv \{\text{GOOD, FAIR, POOR}\}$. While Algorithm 1 provides a simple pseudocode implementation of the model as an example, the framework also supports complex models, e.g., ML for realistic applications.

Moreover, as detailed in Figure 3, the model.fetch_driving_behavior model returns an ML-trained model on how to adapt driving behavior in the input town and comfort level.

**Evaluation: Comfort Knowledge Distribution**

To describe and evaluate the potential performance gains of VKN over traditional information-centric schemes, we study the bandwidth impact of knowledge distribution in vehicular networks. While we use comfort knowledge as a case study, the evaluation scheme is independent of the considered models and shows an initial comparison of the knowledge distribution performances of VKN and ICN.

We investigate the following scenario: a set of vehicles wish to obtain comfort knowledge about various areas as the result of a model.env_comfort application to personalize their driving behavior. The scenario, as illustrated by Figure 6 is as follows:

- A set of vehicles is simulated in a 1-km² area divided into a square grid, with each cell representing a distinct area where input information can be sensed.
- Vehicles’ movements are simulated following the random waypoint mobility model.
- A timeline of $R = 10,000$ knowledge requests is generated. Following a Poisson process, each vehicle periodically requests comfort knowledge from an area, i.e., a grid cell.

We implement both a VKN- and ICN-based approach to compare their overhead impact on comfort knowledge distribution. Figure 7 describes the process of knowledge creation performed when a vehicle $v$ requests comfort knowledge implemented for both VKN and ICN with dashed

---

**Algorithm 1** A simplified algorithm to compute comfort from environmental parameters.

**Input**

$c_{tw} \in \{\text{HIGH, MEDIUM, LOW}\}$

$v \in \{\text{CLEAR, OBSTRUCTED}\}$

$tr \in \{\text{FLUID, CONGESTED}\}$

**Output**

$c_{ft} \in \{\text{GOOD, FAIR, POOR}\}$

1: if $c_{tw} = \text{LOW}$ and $v = \text{CLEAR}$ and $tr = \text{FLUID}$ then
2: $c_{ft} \rightarrow \text{GOOD}$
3: else if $c_{tw} = \text{HIGH}$ then
4: $c_{ft} \rightarrow \text{POOR}$
5: else
6: $c_{ft} \rightarrow \text{FAIR}$
7: end if

---

**Figure 6** The knowledge distribution overhead evaluation setup.
lines. Through VKN, vehicles use knowledge application requests to obtain remotely created knowledge instead of direct model and input information requests in the ICN-based approach. Figure 8 summarizes the VKN remote application of model.env_comfort. A vehicle ego_node on the left directly transmits a knowledge creation request to remote_node in the target area in possession of the required model’s version 1.1 bytecode. In turn, remote_node computes the comfort knowledge using locally sensed input.

We ran 100 simulations for two distinct scenarios, the results of which are summarized in Table 1. The first scenario involved a density of 1,000 vehicles/km² in a grid divided into 50-m² cells to imitate urban conditions. There, the number of model transfers was reduced by 15 ± 0.2% using VKN over the ICN approach. The second scenario simulated rural conditions, with a density of 200 vehicles/km² in a grid of 200-m² cells. There, the number of model transfers was reduced by 44 ± 0.4%.

The bytecode size of ML models depends on their nature and complexity. It ranges from megabytes to hundreds of megabytes for deep neural networks. Depending on the model, VKN thus allows a moderate to strong reduction of overhead. While an extra 0.3 to 0.6 model discovery messages/requests were transmitted using the VKN approach for the urban and rural scenarios, respectively, their size was negligible in front of the bandwidth saved through reduced model transfers. Considering an equivalent size of model input and output, the VKN approach becomes beneficial in terms of overhead from a model size of 100 KB. From 1 MB, it reaches a stable 14% or 40% overhead.

The remote input content is fetched through an ICN named request. The knowledge application request is forwarded to vehicle v. The target model is downloaded to v from a centralized data center. The model is applied locally with local input and returned to the original knowledge-requesting vehicle, if any.

**Figure 7** A comparison of the VKN- and ICN-based knowledge creation processes.
Ego Car Current Position

Vehicular Knowledge Query:
APPLY model.env_comfort:1.1 IN AreaA
RETURN TO @ego_car

Vehicular Knowledge Query Response:
COMPUTED model.env_comfort:1.1 IN AreaA
RETURN TO @remote_car
RESULT Road.ComfortLevel: FAIR

Area A

model.env_comfort:1.1 Description
Input
Road.Traffic   enum [FLUID, CONGESTED]
Road.Visibility enum [CLEAR, OBSTRUCTED]
TwoWheelers.    enum [HIGH, MEDIUM, LOW]
Concentration   enum [GOOD, FAIR, POOR]
Output
Road.ComfortLevel enum [GOOD, FAIR, POOR]

Table 1. The number of communications/10,000 requests in the urban and rural scenarios.

| Transmissions/10,000 Requests | Urban Scenario (1,000 Vehicles) | Rural Scenario (200 Vehicles) |
|------------------------------|--------------------------------|-------------------------------|
|                              | VKN   | ICN  | VKN   | ICN  |
| Model bytecode               | 851 ± 2 | 1,000 | 111 ± 2 | 200 |
| Model output                 | 9,013 ± 11 | 9,822 ± 3 | 0 |
| Model input                  | 0      | 9,006 ± 11 | 0 | 9,592 ± 4 |
| Model discovery              | 2,864 ± 12 | 0 | 5,703 ± 30 | 0 |

| VKN Overhead Reduction | Urban Scenario (1,000 Vehicles) | Rural Scenario (200 Vehicles) |
|------------------------|--------------------------------|-------------------------------|
|                        | VKN   | ICN  | VKN   | ICN  |
| 1 MB/bytecode          | 0.85 GB | 1 GB  | 0.11 GB | 0.2 GB |
| 1 KB/discovery         | 2.9 MB  | 0     | 5.7 MB  | 0     |
| Total overhead         | 0.86 GB | 1 GB  | 0.12 GB | 0.2 GB |
| Difference             | 14%    | 40%   |       |       |

Figure 8. Comfort level retrieval in a remote area using VKN.

Research Applicability

We described an application of cooperative knowledge creation. By remotely applying models in the area where their input is sourced, unnecessary transfers of information are avoided to the benefit of knowledge. Similarly, mechanisms have been defined in the literature to train knowledge models themselves while avoiding the transmission of training information due to privacy and efficiency concerns. Federated learning is an open research topic in which multiple nodes cooperatively train a shared model without directly exchanging training information. Rather, model updates are separately trained by each node with local input and subsequently aggregated.

However, it is not a trivial matter to ensure that all of the local nodes are interested in training and using the same model. Before being able to start the training, the nodes should be able to determine who among their neighbors is in possession of what type of model and has access to what type of information. VKN can be used to orchestrate the client selection process of federated learning algorithms, delegating the model training to remote vehicular nodes to select the most pertinent training nodes depending on their available input and knowledge [15].

Conclusions

Vehicular networks have been extensively studied in the past years. Several standards have been developed to store and share information. However, challenges remain to transition from an ICN model to a model where common standards for knowledge characterization, description,
storage, and sharing allow nodes in vehicular networks to take full advantage of data-driven AI techniques.

In this article, using a common definition of knowledge, we determined under what forms it exists in vehicular networks, allowing us to concretely propose a structure for knowledge description, storage, and sharing. Through a passenger comfort-based rerouting application, we exemplified the concept and showed a significant overhead reduction. Finally, we noted the potential benefits of VKN for the open topic of federated learning. Future work will focus on implementing, simulating, and measuring the benefits of using VKN through packet-level simulations.

Author Information

**Duncan Deveaux** (deveaux@eurecom.fr) is a Ph.D. student in the Communication Systems Department, EURECOM, Campus Sophia Tech, Sophia Antipolis, 06904, France. He received his M.Sc. degree in engineering, specializing in computer science and applied mathematics, in 2018. His research interests involve knowledge networking for next-generation vehicular networks. As part of his Ph.D. studies, he has been working to define and evaluate a framework enabling mutual understanding and exchange of knowledge in vehicular networks.

**Takamasa Higuchi** (takamasa.higuchi@toyota.com) is a principal researcher at InfoTech Labs, Toyota Motor North America R&D, Mountain View, California, 94043, USA. He received his Ph.D. degree from Osaka University in 2014. He started his professional career as an assistant professor at Osaka University and contributed to various research projects related to mobile computing, crowdsensing, and ad hoc networks. He joined Toyota in 2016 and works on vehicular cloud computing, cooperative perception, and heterogeneous vehicular networking.

**Seyhan Uçar** (seyhan.ucar@toyota.com) is currently working as a principal researcher in intelligent mobility systems at InfoTech Labs, Toyota Motor North America R&D, Mountain View, California, 94043, USA. He received his M.Sc. and Ph.D. degrees in computer science and engineering from Koç University in 2013 and 2017, respectively. He is now working on intelligent transportation systems and applications and analyzing the impact of connected vehicles on transportation safety and management.

**Jérôme Härri** (haerr@eurecom.fr) is a professor with the Communication Systems Department, EURECOM, Campus Sophia Tech, Sophia Antipolis, 06904, France. His research interests are related to wireless vehicular communication and networking, traffic flow modeling, the Internet of Things, control system optimization, and their mutual interactions in future automated vehicles. He has authored or coauthored more than 80 international journal and conference papers and is involved in various national and European research projects related to connected vehicles.

**Onur Altintas** (onur.altintas@toyota.com) is the InfoTech Labs Fellow at InfoTech Labs, Toyota North America R&D, Mountain View, California, 94043, USA. He has been with the Toyota Group since 1999. He is the co-founder and has been general cochair of the IEEE Vehicular Networking Conference since 2009. He serves as an associate editor for IEEE Intelligent Transportation Systems Magazine, IEEE Transactions on Intelligent Vehicles, and IEEE Vehicular Technology Magazine. He is an IEEE Distinguished Lecturer.

References

[1] C. Zin, “Conceptual approaches for defining data, information, and knowledge,” J. Amer. Soc. Inform. Sci. Technol., vol. 38, no. 4, pp. 479–493, Feb. 2007. doi: 10.1002/asi.25068.

[2] M. Ruta, F. Scioscia, F. Grameneglia, S. Ieva, E. D. Sciscio, and R. P. De Vera, “A knowledge fusion approach for context awareness in vehicular networks,” IEEE Internet Things J., vol. 5, no. 4, pp. 2407–2419, Aug. 2018. doi: 10.1109/JIOT.2018.2815009.

[3] Q. Qi et al., “Knowledge-driven service offloading decision for vehicular edge computing: A deep reinforcement learning approach,” IEEE Trans. Veh. Technol., vol. 68, no. 5, p. 1, 2019. doi: 10.1109/TVT.2019.2894437.

[4] M. I. Khan, F.-X. Aubet, M.-O. Pahl, and J. Härri, “Deep learning–aided resource orchestration for vehicular safety communication,” in Wireless Days 2019, IEEE/IFIP Days 2019, 11th ed., Manchester, U.K., Apr. 2019.

[5] D. Wu, Z. Li, J. Wang, Y. Zheng, M. Li, and Q. Huang, “Vision and challenges for knowledge centric networking,” IEEE Wireless Commun., vol. 26, no. 4, pp. 117–123, Aug. 2019. doi: 10.1109/MWC.2019.1800323.

[6] B. Klotz, R. Troncy, B. Wilms, and C. Bonnet, “VSo—a vehicle signal and attribute ontology,” in Proc. 9th Int. Semantic Sensor Netw. Workshop (SSY 2018), Monterey, CA, Oct. 2018.

[7] H. Yao, M. Li, J. Du, P. Zhang, C. Jiang, and Z. Han, “Artificial intelligence for information-centric networks,” IEEE Commun. Mag., vol. 57, no. 6, pp. 47–53, June 2019. doi: 10.1109/MCOM.2019.1800734.

[8] H. Hao, C. Xu, M. Wang, H. Xie, Y. Liu, and D. O. Wu, “Knowledge-centric proactive edge caching over mobile content distribution network,” in Proc. IEEE Int. Conf. Comput. Commun. Workshops (INFOCOM WKSHPS). IEEE INFOCOM 2018, pp. 450–455. doi: 10.1109/INFOCOM.2018.8409505.

[9] X. Zhang, H. Wang, and H. Zhao, “An SDN framework for UAV backbone network towards knowledge centric networking,” in Proc. IEEE Int. Conf. Comput. Commun. Workshops (INFOCOM WKSHPS), 2018, pp. 456–461.

[10] D. Sapra and A. D. Pimentel, “Deep learning model reuse and composition in knowledge centric networking,” in Proc. 29th Int. Conf. Comput. Commun. Netw. (ICCCN), 2020, pp. 1–11. doi: 10.1109/ICCCN49398 .2020.9266684.

[11] T. Higuchi, J. Joy, F. Dressler, M. Gerla, and O. Altintas. “On the feasibility of vehicular micro clouds,” in Proc. IEEE Veh. Netw. Conf. (VNC), Nov. 2017.

[12] P. Mach and Z. Becvar, “Mobile edge computing: A survey on architecture and computation offloading,” IEEE Commun. Surveys Tuts., vol. 19, no. 3, pp. 1628–1656, 2017. doi: 10.1109/COMST.2017.2682318.

[13] D. Martin et al., “Bringing semantics to web services: The OWLS approach,” J. Cardoso and A. Sheth, Semantic Web Services and Web Process Composition, 2005, pp. 26–42.

[14] N. Lal, S. Kumar, and V. K. Chaurasiya, “An efficient update strategy for content synchronization in Content-Centric Networking (CCN)” China Commun., vol. 16, no. 1, pp. 108–118, 2019.

[15] D. Deveaux, T. Higuchi, S. Uçar, C.-H. Wang, J. Härri, and O. Altintas, “On the orchestration of federated learning through vehicular knowledge networking,” in Proc. IEEE Veh. Netw. Conf. (VNC), 2020, pp. 1–8. doi: 10.1011/VNC51378.2020.9318386.