RoADTrain: Route-Assisted Decentralized Peer Model Training among Connected Vehicles

Han Zheng, Mengjing Liu, Fan Ye, Yuanyuan Yang
Department of Electrical and Computer Engineering, Stony Brook University
{han.zheng, mengjing.liu, fan.ye, yuanyuan.yang}@stonybrook.edu

***Abstract*—Fully decentralized model training for on-road vehicles can leverage crowdsourced data while not depending on central servers, infrastructure or Internet coverage. However, under unreliable wireless communication and short contact duration, model sharing among peer vehicles may suffer severe losses thus fail frequently. To address these challenges, we propose "RoADTrain", a *route-assisted* decentralized peer model training approach that carefully chooses vehicles with high chances of successful model sharing. It bounds the per round communication time yet retains model performance under vehicle mobility and unreliable communication. Based on shared route information, a connected cluster of vehicles can estimate and embed the link reliability and contact duration information into the communication topology. We decompose the topology into subgraphs supporting parallel communication, and identify a subset of them with the highest algebraic connectivity that can maximize the speed of the information flow in the cluster with high model sharing successes, thus accelerating model training in the cluster. We conduct extensive evaluation on driving decision making models using the popular CARLA simulator. RoADTrain achieves comparable driving success rates and $1.2-4.5\times$ faster convergence than representative decentralized learning methods that always succeed in model sharing (e.g., SGP), and significantly outperforms other benchmarks that consider losses by $17-27\%$ in the hardest driving conditions. These demonstrate that route sharing enables shrewd selection of vehicles for model sharing, thus better model performance and faster convergence against wireless losses and mobility.**

***Index Terms*—Vehicular learning; vehicular communication; decentralized learning**

I. Introduction

Autonomous/connected vehicles require models for various purposes, from object/lane detection to driving decisions [1]. Training such models requires huge amounts of data. Many companies collect data over multiple years with dedicated fleets, incurring enormous financial, operational, and human resources [2] [3]. Recently federated learning has enabled the possibility of crowdsourcing data from large numbers of common on-road vehicles [4]. Assuming suitable communication and incentives, this avoids the huge costs of dedicated fleets, and obtains data of far greater variety in possibly much shorter time. Vehicles collect, process data and train local models, then use cellular networks to share model updates with central servers to produce aggregated models.

Although recent work has shown the effectiveness of such approaches [5], cellular network coverage is often sporadic, and the data rates to the backend can be very limited and unstable [6]. Some work utilizes additional infrastructure (e.g., Road-Side Units) [7], which are not widely available either. Fully decentralized learning (e.g. gossip learning and peer-to-peer federated learning [8] [9]) where each vehicle trains local iteration(s), communicates the model with other nearby vehicles to produce aggregated models, is free from such constraints, and thus has become increasingly popular for vehicular learning.

However, existing work does not consider unreliable wireless communication or short contact durations. Wireless transmission losses among peer vehicles can be significant. Due to mobility, the contact duration in which two nearby vehicles are within the radio range of each other, may last only tens of seconds or even shorter. Thus contrary to assumed in existing work, model sharing can fail frequently.

Due to limited bandwidth of peer vehicular communication technologies, the time in sharing models between vehicles dominates local training time. For instance, using typical vehicular communication technologies (e.g., 802.11bd at up to 31Mbps [10]), a cluster of two dozen vehicles may take about 30 seconds to finish sharing compressed models among one hop neighbors. However, powerful TFLOPS-level onboard GPUs can cut down per round local training time to < 0.1 second [11]. Therefore, indiscriminately exchanging models with all neighbors may take time far exceeding the contact duration, and fails easily.

In this work, we explore a fully-decentralized, peer vehicular training approach under unreliable wireless communication and short contact duration. The key is to introduce *route sharing* among peer vehicles. Route indicates the waypoints of a vehicle to go in a short future (e.g., next 30 seconds), which can be obtained from navigation instructions in real time. Based on such waypoints from others, a vehicle can estimate the distances, thus rough data loss chances and contact durations with them. It can prioritize and schedule model sharing with neighbors of higher chances of completion, thus avoiding resource waste in time and bandwidth. Given the small size of route information (e.g., 184 bytes in our experiment settings), route sharing consumes little resource yet it enables shrewd selection of neighbors to greatly boost model sharing successes.

Specially, vehicles keep training local iteration(s), sharing routes and models, and updating models repeatedly. We call each cycle of training local iteration(s), the subsequent peer communication, and local model updates among vehicles

This work is supported in part by NSF grant 2007715.

within a connected cluster as *one round*. Considering the asymmetry in model sharing [12], we represent the cluster topology in each round as a directed graph called the *base graph*. Wireless loss and contact duration characteristics are estimated from route and then embedded into the base graph.

Motivated by theoretical results [13] [14], we identify a subset of edges forming a communication graph with the *maximum algebraic connectivity* by solving a constrained integer programming problem. The intuition is that such a communication graph enables vehicles to receive information from other non-directly connected vehicles with fewer intermediary relays. Thus vehicles acquire information from the entire cluster faster, accelerating training on the joint dataset.

We consider a synchronous scenario in this work, where vehicles in a connected cluster start each round at the same time, finish all the computation and communication, then start the next round. We note that there is work studying asynchronous decentralized learning [15]. However, asynchronicity may impede model convergence to the optima, and even higher throughput (number of tasks executed per time unit) does not necessarily guarantee faster convergence [16] [17]. So we focus on exploiting route sharing in the synchronous mode, and leave the more complex asynchronous mode in the future.

Our contribution is threefold:

- We propose a novel route-assisted, decentralized peer vehicular model training scheme to address practical challenges of unreliable wireless communication and short contact duration, which cause frequent model sharing failures among peer vehicles.
- We embed wireless loss and contact duration estimated from shared routes into the base topology of a connected vehicle cluster. Then we formulate an integer programming-based communication graph construction problem to maximize algebraic connectivity. We design base graph decomposition and topology-driven subgraph selection algorithms to choose neighbors with high chances of successful model sharing and enhance the information flow in the cluster, thus accelerating training.
- We evaluate RoADTrain on Bird-Eye-View (BEV)-based driving decision making task. We find that RoADTrain can converge to a loss approaching the lower bounds set by FedAvg [18] and SGP [12] which are not subject to model sharing failures, and $1.2-4.5\times$ faster convergence than SGP due to much shorter per round communication time ($2.7-11\times$). In online evaluation, RoADTrain achieves driving success rates competitive to FedAvg and SGP. It outperforms other decentralized benchmarks that are subject to model sharing failures by 17-27$\%$ higher driving success rates in the hardest driving condition.

To the best of our knowledge, RoADTrain is the first work addressing frequent model sharing failures caused by wireless loss and mobility in decentralized peer vehicular model training. The insight is that shared routes enable shrewd selection of neighbors with higher chances of model sharing success, thus better performance and faster convergence.

II. Preliminaries

A. Communication model

We consider a network of vehicles $\mathcal{V} = \{1, 2, ..., n\}$ and each vehicle drives independently. In each round, a vehicle updates its local model and exchanges it along with other assistant information (e.g. routes) with one hop neighbor(s). We assume that vehicles have access to assistant information such as routes, and do not consider cases where such information may not be available, such as driving without a specific destination or in areas where navigation services are not provided. We also assume that vehicles have similar hardware and communication capabilities. Similar to [19], communication between vehicles is carried out using via an orthogonal frequency division multiple-access (OFDMA) wireless channel [10] [20] consisting of R subcarriers, and each subcarrier can only be occupied by one sender in each round. We model data loss in vehicular wireless communication following [10].

Vehicle move in and out of each other's radio range constantly, thus the communication topology is time-varying. To address the synchronous scenario, we discretize the time-varying communication topology into a graph sequence in terms of rounds. We represent the communication topology in round k as a directed graph called *base graph* $\mathcal{G}^k$ with vertex set $\mathcal{V}$ and edge set $\mathcal{E}^k = \{(i,j)|i \neq j, i, j \in \mathcal{V}\}$. Vertex *i* denotes the vehicle *i*. Edge (i,j) in $\mathcal{G}^k$ denotes that vehicle i can send information to vehicle j. Note that in the base graph, two directed edges exist between vertex i and j if they are within the radio range, showing they can send to each other. However, in one round, RoadTrain may decide vehicle i sending model to vehicle j but not necessarily vice versa.

The base graph in round k can be abstracted as an adjacency matrix $A^k \in \mathbb{R}^{n\times n}$ with binary values. $A^k_{ij} = 1$ if edge $(i,j) \in \mathcal{E}^k$, otherwise $A^k_{ij} = 0$. For edge (i,j) in base graph $\mathcal{G}^k$, we use a real value $p^k_{(i,j)} \in [0,1]$ to indicate the probability that vehicle i can send model to vehicle j successfully. We denote the probabilities at round k as a matrix $P^k \in \mathbb{R}^{n\times n}$. Since the model size (even after compression) is much larger than the packet size S, one model requires N packets. We model p, model sharing success probability, using packet reception rate $\hat{p}$, where $\hat{p}$ can be roughly estimated.

B. Model training

In decentralized peer model training, each vehicle trains a model locally with identical model structure. Vehicles share models on the communication topology and aggregate received parameters to updates local models. Each vehicle *i* has its local dataset D_i. Our objective is to train models on the *n* vehicles with their local dataset, which can be formally defined by:

$$\begin{aligned} \min_{\mathrm{x_i}\in\mathbb{R},\mathrm{i}=1,\ldots,\mathrm{n}} \quad & \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{\xi_i\sim D_i}F_i(\boldsymbol{x}_i;\xi_i) \\ \text{subject to} \quad & \boldsymbol{x_i} = \boldsymbol{x_j}, \forall i,j = 1,...,n \end{aligned} \tag{1}$$

where $\boldsymbol{x_i}$ denotes the model parameters of vehicle *i*, $F_i(\boldsymbol{x}_i)$ denotes the loss function at vehicle *i*, ξ_i denotes a single data

sample or a mini-batch from local dataset D_i. Let $f_i(\boldsymbol{x}_i) = \mathbb{E}_{\xi_i \sim D_i} F_i(\boldsymbol{x}_i; \xi_i)$ denote the loss at vehicle *i*. The average loss is $f(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x})$.

Decentralized SGD (D-PSGD) [21] is a typical scheme to update models, where SGD can be replaced by other optimizer (e.g. Adam). In D-PSGD, the model aggregations among *n* nodes is represented by a mixing matrix $\Phi \in \mathbb{R}^{n\times n}$, where $\varphi_{ij} \in [0, 1]$ denoting the aggregation coefficient of the model from node *j* at node *i*. $\varphi_{ij} = 0$ means node *i* can not transmit information to node *j*. Each node runs computation iterations and communicates round by round in parallel. In the *k*th round, node *i* computes gradient $\nabla F_i(\boldsymbol{x}_i^k; \xi_i^k)$ on randomly sampled mini batch $\xi_i^k \subset D_i$, where $\boldsymbol{x}_i^k$ denotes the model parameter on node *i* at *k*th iteration. Then the node does SGD to update model by $\boldsymbol{x}_i^{k+\frac{1}{2}} \leftarrow \boldsymbol{x}_i^k - \gamma \nabla F_i(\boldsymbol{x}_i^k; \xi_i^k)$. γ is the step size. After the local update, node *i* exchanges models with one hop neighbors and averages models by $\boldsymbol{x}_i^{k+1} \leftarrow \sum_j^n \varphi_{ji} \boldsymbol{x}_i^{k+\frac{1}{2}}$.

The mixing matrix in D-PSGD is assumed fixed, symmetric and doubly stochastic, which are hard to satisfy in peer vehicular environment due to asymmetric communication. A more generalized variant is *Stochastic Gradient Push (SGP)* [12]. Given a time-varying, asymmetric, and column stochastic mixing matrix, one scalar parameter w_i is maintained and updated similarly in each round, as $w_i^{k+1} \leftarrow \sum_j^n \varphi_{ji}^k w_i^k$, and shown to guarantee convergence under the relaxed constraints towards mixing matrix. SGP is demonstrated to converge [12] under multiple assumptions which are commonly made in decentralized learning. We omit them due to space limitation.

We design our vehicular model training scheme based on SGP with focus on optimizing the communication topology, thus the mixing matrix for each round.

C. Problem formulation

In round k, we aim to identify a communication graph $\hat{\mathcal{G}}^k(\mathcal{V}, \hat{\mathcal{E}}^k)$ from the base graph and compute the corresponding mixing matrix Φ^k, where $\hat{\mathcal{E}}^k \subseteq \mathcal{E}^k$ for communication in each round under resource constraints. Specifically, given vertices i and j, let $b_{(i,j)}^k$ indicate whether edge (i, j) is selected for communication in round k, *i.e.*, $b_{(i,j)}^k = 1$ if vehicle i sends its model to vehicle j and $b_{(i,j)}^k = 0$, otherwise.

Because the communication topology keeps changing, We aim to bound the per round time so that vehicles can complete a full round (or even several) before the next change. The per round time includes local training time and communication time. Because of specialized on-board systems for ML-related tasks [22], the computation time for one or several local iterations becomes negligible (e.g. less than 0.01s). Thus we concentrate on optimizing the communication part and omit local training time. Like [19], we assume that vehicles can use subcarriers to communicate with different neighbors in parallel. We define sending/receiving a single model with one subcarrier as one *unit* time (e.g., about 2.6 seconds under our experiment settings), and define *per round communication time budget* B as the maximum units of time for sharing models among all one hop neighbors in one round.

We formulate the resource constraints in round k as:

$$\begin{cases} \sum_{(i,j)\in\hat{\mathcal{E}}^k} b_{(i,j)}^k \leqslant R, & \forall k \\ T^k \leqslant B, & \forall k \\ b_{(i,j)}^k \in \{0, 1\}, & \forall k \end{cases} \tag{2}$$

where T^k denotes the time consumption for all vehicles to finish the computation and communication in round k. The first set of inequalities guarantees the channel subcarriers constraints. Specifically, each edge will occupy one subcarrier. The second set of inequalities guarantees the per round time budget constraints.

III. Method

In this section, we present RoADTrain in detail (Fig. 1).

A. Base graph augmentation with route

In RoADTrain, vehicles share current locations, speeds and routes in a short future with neighbors in each round. Then vehicles can calculate the distances to others, thus estimating contact durations, and the probabilities of successful model sending among vehicles.

In round k, vehicle i prioritizes model sending to vehicle j if the respective contact duration is short. This allows vehicle i to share models with more neighbors, because vehicles of longer contact durations can afford to wait. However, if the contact duration is too short to finish model sending, such neighbors should be excluded to avoid resource waste. Let $z_{i,j}^k \in [0, 1]$ denote the priority that vehicle i sends its model to vehicle j in round k, which is formulated as follows:

$$z_{i,j}^k = \begin{cases} \frac{\hat{t}}{t_{i,j}^k}, & t_{i,j}^k \geqslant \hat{t}, \\ 0, & t_{i,j}^k < \hat{t}, \end{cases} \tag{3}$$

where $t_{i,j}^k$ indicates the contact duration of vehicles i and j in future from round k and $\hat{t}$ indicates the time required to send a model. We denote the priorities of sending models among vehicles as a matrix $Z^k \in \mathbb{R}^{n\times n}$.

To estimate $\hat{t}$, we first look up packet reception ratio $\hat{p}$ by referencing the distance between two vehicles [10] and calculate the expected number of transmissions per packet with up to three retransmissions [23] as:

$$\mathbb{E}(\#\ of\ TX) = \hat{p} + 2(1-\hat{p})\hat{p} + 3(1-\hat{p})^2\hat{p} + 4(1-\hat{p})^3 \tag{4}$$

Given data transmission rate r, we estimate $\hat{t}$ as the expected transmission time for all packets in the model:

$$\hat{t} = \mathbb{E}(\#\ of\ TX)NS/r \tag{5}$$

Recall that we use P^k to denote the matrix of probabilities of successful model sending among vehicles at round k. With up to three retransmissions per packet, the probability p of successful sending of all N packets is calculated as:

$$p = (1 - (1 - \hat{p})^4)^N \tag{6}$$

We augment the base graph by assigning edge weights. The adjacency matrix $\tilde{A}^k$ of the augmented base graph $\tilde{\mathcal{G}}^k$ is:

$$\tilde{A}^k = A^k \odot Z^k \odot P^k \tag{7}$$

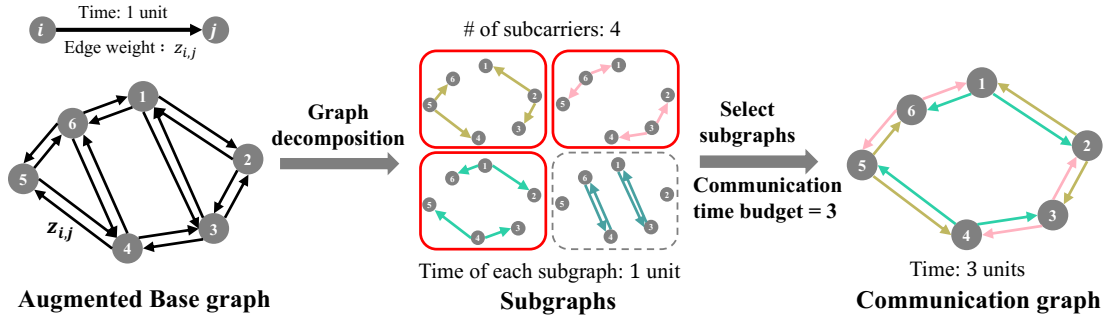


Fig. 1. Illustration of RoADTrain. There are three main steps in RoADTrain. In each round, RoADTrain first augments the base graph with route information sharing among vehicles, which embeds wireless loss and contact duration estimated from route to the designed edge weights. The amount of time of sending/receiving a single model over an directed edge is treated as one unit time. Then the base graph is then decomposed into a series of subgraphs with the guide of edge weights under the subcarrier constraint to support parallel communication. Given a per round communication time budget, RoADTrain takes a union with the highest algebraic connectivity of some subgraphs as a communication graph by solving a constrained optimization problem. Vehicles communicate over the communication graph in this round.

where $\odot$ indicates Hadamard product and the element is denoted by $\tilde{a}^k$. With such augmentation, we embed route information into the base graph, which serves subsequent base graph decomposition and communication graph construction.

B. Base graph decomposition

We omit negligible times in training local iterations and sharing assistant information, the per round communication time of one vehicle increases monotonically with the number of neighbors. Given R channel subcarriers, communicating models on a subgraph including no more than R edges takes one unit time, since communications on those edges can happen in parallel. Moreover, in the synchronous scenario, communicating over a union of n such subgraphs without common edges takes no more than n unit time, by simply communicating on one subgraph at a time. This inspires us: we can bound per round communication time within budget B by letting vehicles communicating models on a union of B such subgraphs each with no more than R edges.

Thus we decompose the augmented base graph $\tilde{\mathcal{G}}^k$ into disjoint subgraphs $\{\tilde{\mathcal{G}}_m^k(\mathcal{V}, \tilde{\mathcal{E}}_m^k)|m = 1, ..., M\}$, where $\tilde{\mathcal{G}}^k(\mathcal{V}, \tilde{\mathcal{E}}^k) = \bigcup_{i=1}^{M} \tilde{\mathcal{G}}_m^k(\mathcal{V}, \tilde{\mathcal{E}}_m^k)$ and $\tilde{\mathcal{E}}_i^k \bigcap \tilde{\mathcal{E}}_j^k = \emptyset$ if $i \neq j$. We propose a heuristic graph decomposition algorithm that samples edges to create a series of subgraphs (Algorithm 1). To cover more vertices and avoid an isolated vertex (i.e., a vehicle with no neighbors in a subgraph), the edge sampling starts from the vertex with the least number of neighbors. Given a vertex with multiple edges, the edge with largest edge weight $\hat{a}^k$ is chosen to add to the current subgraph under construction, because the neighboring vertex has high priority and probability of successful model sharing. The above repeats until we construct a subgraph with R edges. Then we construct the next subgraph, and so on, until there are no more edges left in the base graph.

Algorithm 1 consists of two parts. One is to sort all vertices in $\mathcal{V}$ according to their in-degrees, which is designed to cover vertices weakly connected to others and thus easily overlooked. The time complexity is $\mathcal{O}(n \log n)$ with common sorting algorithms (e.g. quick sort). The other is to visit each edge once during the graph decomposition, at time complexity of $\mathcal{O}(|\mathcal{E}|)$. In the worst case, it degrades to $\mathcal{O}(n^2)$ when the graph is fully-connected. Therefore, the overall time complexity of Algorithm 1 in the worst case is $\mathcal{O}(n^2)$.

Algorithm 1 Route assisted graph decomposition

Input: Augmented base graph $\tilde{\mathcal{G}}(\mathcal{V}, \tilde{\mathcal{E}})$; Number of channel subcarriers R;
Output: Disjoint subgraphs $\mathcal{S} = \{\tilde{\mathcal{G}}_m(\mathcal{V}, \tilde{\mathcal{E}}_m)|m = 1, ..., M\}$;

$\mathcal{S} = \{\}$
$V_s \leftarrow$ Sorted vertices in the graph $\tilde{\mathcal{G}}(\mathcal{V}, \tilde{\mathcal{E}})$ in ascending order according to number of neighbors
$\tilde{\mathcal{E}}_c \leftarrow \tilde{\mathcal{E}}$ (Make a copy of edge set $\tilde{\mathcal{E}}$)
while $\tilde{\mathcal{E}}_c \neq \emptyset$ **do**
  Initialize a empty temporary edge set $\mathcal{E}_{tmp} = \{\}$
  $\mathcal{V}_c \leftarrow \mathcal{V}$ (Make a copy of vertex set)
  for u in V_s **do**
    if $\mathcal{V}_c = \emptyset$ or $|\mathcal{E}_{tmp}| = R$ **then**
      Break
    end if
    if $v \notin \mathcal{V}_c$ **then**
      Continue
    end if
    for edges in $\{(u, v)|v \in \mathcal{V}_c\}$ **do**
      if $(u, v') \in \tilde{\mathcal{E}}_c$ and $\tilde{a}_{(u,v')}$ is largest in $\tilde{a}_{(u,v)}$ **then**
        $\tilde{\mathcal{E}}_c \leftarrow \tilde{\mathcal{E}}_c - (u, v')$
        $\mathcal{E}_{tmp} \leftarrow \mathcal{E}_{tmp} \bigcup \{(u, v')\}$
        Break
      end if
    end for
    $\mathcal{V}_c \leftarrow \mathcal{V}_c - u$
    $\mathcal{V}_c \leftarrow \mathcal{V}_c - \hat{v}$
  end for
  $\mathcal{S} \leftarrow \mathcal{S} \bigcup \{(\mathcal{V}, \mathcal{E}_{tmp})\}$
end while

In a typical connected cluster of two dozen vehicles, graph decomposition takes merely tens of milliseconds, or even less with powerful onboard hardware [11], negligible compared to sending models.

C. Communication graph construction

The decomposition may create $M > B$ subgraphs each taking one unit time but having varying chances of model sharing successes with neighbors. We aim to select a subset of B subgraphs that can maximize the speed of information flow among vehicles with high sharing success chances to accelerate model convergence and improve performance. Following the insight that higher algebraic connectivity leads to faster information flow, we design an algorithm to select at most B subgraphs to form a communication graph of the maximum algebraic connectivity.

Formally, given a directed graph $\mathcal{G}$ with adjacency matrix A, the algebraic connectivity of the graph [24] is defined as :

$$\begin{aligned} a(\mathcal{G}) &= \min_{x\in\mathbb{R}^V, x\neq 0, x\perp \mathbf{1}} \frac{x^T L x}{x^T x} \\ &= \lambda_{min}(\frac{1}{2}Q^T(L+L^T)Q) \end{aligned} \tag{8}$$

where L is the Laplacian matrix of $\mathcal{G}$ defined as $L = D - A$ with the diagonal matrix of vertex out-degrees D, Q is an n by $n-1$ matrix whose columns form an orthonormal basis of the orthogonal complement of $\mathbf{1}$, and λ_{min} is the smallest eigenvalue of L. Algebraic connectivity quantifies how fast the information can flow in a network: the higher the algebraic connectivity, the denser the graph, and the easier the information flows from one vertex to another. E.g., in a cyclic graph with n vertices, information from any vertex can reach others after at most n hops. The algebraic connectivity of a cyclic graph monotonically decreases as n increases, since more hops are needed to flow through all nodes.

Theoretical results show that the number of training round in decentralized optimization is minimized upon the highest algebraic connectivity of the communication topology [13], [14]. Since model sharing between two vertices can fail due to unreliable wireless communication and short contact duration, we emphasize that the algebraic connectivity in RoADTrain is not only about the number of hops, but also greatly affected by the weight of each edge (Eqn. (7)).

When choosing B subgraphs to form the current communication graph, we also consider the communication graphs in the last H rounds. We prioritize vehicles not well synchronized in the past, so as to accelerate overall model training. We select subgraphs from $\{\tilde{\mathcal{G}}_m(\mathcal{V}, \tilde{\mathcal{E}}_m) | m = 1, ..., M\}$ to form the actual communication graph $\hat{\mathcal{G}}$ by maximizing the algebraic connectivity of the union of at most B subgraphs in the current round plus past H historical communication graphs. H is carefully selected to be large enough to meet the mixing connectivity assumption in SGP [12] mentioned in Section II.

We formulate the optimization problem as follows:

$$\begin{aligned} \max_{c_1^k,...,c_M^k} \quad & \lambda_{min}(\frac{1}{2}Q^T(\sum_{m=1}^{M} c_m^k(\tilde{L}_m^k + (\tilde{L}_m^k)^T) \\ & + \sum_{j=k-H}^{k} (\hat{L}^j + (\hat{L}^j)^T))Q) \\ \text{subject to} \quad & \sum_{m=1}^{M} c_m^k \leqslant B, \\ & c_m^k \in \{0,1\}, \forall m \in \{1,...,M\}, \end{aligned} \tag{9}$$

where $\tilde{L}_m^k$ is the Laplacian matrix of subgraph $\tilde{\mathcal{G}}_m^k$, $\hat{L}^j$ is the Laplacian matrix of communication graph $\hat{\mathcal{G}}^j, (k-H \leq j < k)$, and c_m^k is a binary indicator of $\tilde{\mathcal{G}}_m^k$ representing whether $\tilde{\mathcal{G}}_m^k$ is selected. We aim to maximize the algebraic connectivity by choosing proper values for c_m^ks.

Since the communication graph construction decision c_m^k is binary from $\{0,1\}$, this is a typically integer programming problem. In general, finding the optimal solution is NP-hard [19]. Thus, we relax c_m^k to real instead of binary, so that the optimization problem is convex and can be solved efficiently. After solving the convex optimization problem, we select subgraphs with B-largest indicators. By communicating models with neighbors on selected subgraphs in sequence, vehicles can finish model exchanging within the per round communication time budget. We define the communication graph $\hat{\mathcal{G}}^k$ for round k as the union of these selected subgraphs:

$$\hat{\mathcal{G}}^k = (\mathcal{V}, \hat{\mathcal{E}}^k = \bigcup_{j=1}^{B} \tilde{\mathcal{E}}_j^k) \tag{10}$$

where the adjacency matrix of $\hat{\mathcal{G}}^k$ is represented by $\hat{A}^k$.

D. Decentralized peer vehicular learning

After exchanging models with one hop neighbors, vehicles calculate the mixing matrix for model aggregation. In RoADTrain, the mixing matrix Φ^k is column-stochastic and asymmetric instead of doubly-stochastic and symmetric, since vehicle i may send model to vehicle j, but not necessarily vice versa. The mixing matrix Φ^k is designed as the column-normalized $\hat{A}^k$, whose element is computed as:

$$\varphi_{i,j}^k = \hat{a}_{i,j}^k / \sum_{i=1}^{n} \hat{a}_{i,j}^k, \ i = 1, ..., n, j = 1, ..., n \tag{11}$$

When the mixing matrices Φ^k are asymmetric, following SGP [12], one additional scalar parameter w_i^k is maintained at each vehicle to help models on vehicles converge to the same place. The parameter is initialized to be the same at all vehicles, and updated with local models.

Now we describe how to implement the procedure of RoADTrain in a decentralized way in practice. In each round, vehicles first try to construct consistent cluster topology by exchanging local topologies among themselves multiple times. The main idea is to gradually expand the discovered local

topology over repeated exchange. At the beginning, vehicle i only knows a local graph centered at itself within 1 hop on the base graph $\mathcal{G}^k$, which corresponds to the ith row and the ith column of adjacency matrix A^k. After exchanging local graphs with its 1-hop neighbors once, it knows its local graph within two hops. Assume the diameter of the base graph is d. After at most d times of exchange, vehicle i gets the complete base graph $\mathcal{G}^k$, thus adjacency matrix A^k.

Communicating an adjacency matrix (1KB without compression) with one subcarrier takes negligible time (e.g., 7.7 milliseconds, assuming 1Mbps per subcarrier, following 31Mbps radios [10] and 30 subcarriers [19]) under typical settings (say a couple dozen vehicles). When the cluster is large, using basic flooding for the above discovery may suffer high overheads of broadcast storms. We note that there is existing work (e.g. clustering based schemes [25] proven to propagate messages effectively with low overhead. For instance, topology discovery protocols [25], [26] among a large cluster of 100 vehicles take 2000 packets, which can finish within three seconds under our experiment setting (e.g., 802.11bd, 31 Mbps with packet size of 1500 bytes). Message losses during such cluster topology discovery is already compensated by repeated exchange, and can be further alleviated by existing techniques (e.g. packet re-transmissions [23] and hybrid automatic repeat request (HARQ) [27]). Thus obtained adjacency matrices differ at most slightly among vehicles. We show that RoADTrain is robust under such losses and differences in Section IV-F.

Once the cluster topology is obtained, vehicles share route, location and speed information with one hop neighbors. Since the information size is very small (e.g., 184 bytes in our experiments), even with potential re-transmissions, we can ignore the time in sharing route information.

After forming the complete base graph $\mathcal{G}^k$ and route information sharing, vehicles augment the base graph as in Eqn. (7) and decompose the base graph into subgraphs by using Algorithm 1. Then vehicles solve the optimization problem in Eqn. (9), construct the actual communication graph $\hat{\mathcal{G}}^k$ with Eqn. (10) and compute the mixing graph as in Eqn. (11). All computations are conducted by vehicles in parallel. We summarize our proposed algorithm in Algorithm 2.

Algorithm 2 Decentralized peer vehicular learning

Input: Initialize $\hat{\boldsymbol{x}}_i^0 = \boldsymbol{x}_i^0$ and $w_i^0 = 1$ for all vertices (vehicles) $i \in \{1, ..., n\}$, learning rate γ, and number of training round K;

Output: Models on vehicles $\{\boldsymbol{x}_i | i \in \{1, ..., n\}\}$

for $k = 1, ..., K$, at vertex i **do**
  Sample a new mini-batch $\xi_i^k \sim D_i$ from local dataset
  Compute mini-batch gradient at $\boldsymbol{x}_i^k : \nabla F_i(\boldsymbol{x}_i^k; \xi_i^k)$
  $\hat{\boldsymbol{x}}_i^{k+\frac{1}{2}} \leftarrow \hat{\boldsymbol{x}}_i^k - \gamma \nabla F_i(\boldsymbol{x}_i^k; \xi_i^k)$
  Propagate local topology to neighbors for multiple times and form the complete base graph $\mathcal{G}^k$
  Share route, location and speed information with one hop neighbors
  Generate the augmented base graph $\tilde{\mathcal{G}}^k$
  Decompose the augmented base graph $\tilde{\mathcal{G}}^k$ into subgraphs $\{\tilde{\mathcal{G}}_m^k(\mathcal{V}, \tilde{\mathcal{E}}_m^k) | m = 1, ..., M\}$ using Algorithm 1
  Select B subgraphs $\{\tilde{\mathcal{G}}_m^k(\mathcal{V}, \tilde{\mathcal{E}}_m^k) | m = 1, ..., B\}$ by solving the optimization problem in Eqn. (9)
  Form the communication graph $\hat{\mathcal{G}}^k$ with the adjacency matrix $\hat{A}^k$ by using Eqn. (10)
  Compute mixing matrix Φ^k by using Eqn. (11)
  for $m = 1, ..., B$ **do**
    Send $(\varphi_{j,i}^k \hat{\boldsymbol{x}}_i^{k+\frac{1}{2}}, \varphi_{j,i}^k w_i^k)$ to vertex j if $(i, j) \in \tilde{\mathcal{E}}_m^k$, $\forall j \in \mathcal{V}$;
    receive $(\varphi_{i,j}^k \hat{\boldsymbol{x}}_j^{k+\frac{1}{2}}, \varphi_{i,j}^k w_j^k)$ from vertex j if $(j, i) \in \tilde{\mathcal{E}}_m^k$, $\forall j \in \mathcal{V}$
  end for
  $\hat{\boldsymbol{x}}_i^{k+1} \sum_j \varphi_{i,j}^k \hat{\boldsymbol{x}}_j^{k+\frac{1}{2}}$
  $w_i^{k+1} \leftarrow \sum_j \varphi_{i,j}^k w_j^k$
  $\boldsymbol{x}_i^{k+1} \leftarrow \hat{\boldsymbol{x}}_i^{k+1} / w_i^{k+1}$
end for

IV. Experiments

A. Experimental setup

We consider a driving decision making task. We aim to train imitation learning models with the same structure as the privileged agent in [3] using ResNet-18 as the backbone. The model takes a Bird-Eye-View (BEV) map (a tensor depicting the front view of a vehicle in a top-down view) and assistant information (e.g. speed and high-level command) as model inputs, and outputs the next few waypoints to go.

We collect training data in the popular CARLA simulator [28]. Following the same setting as [12], we run 32 built-in expert autopilot vehicles in a simulated world including both urban and rural environments. The simulated world is about 1km×1km, the largest built-in map size supporting multiple expert autopilots in CARLA. Expert autopilots can perform safe and professional driving using the built-in model and privileged information in CARLA. Their start points and destinations are randomly assigned, and the routes provided by CARLA based on road topology. After reaching the destination, an autopilot selects the next destination on the map randomly. We add 50 cars and 250 pedestrians in the same simulated world, initialized at random locations but just keep roaming (following the current road and randomly selecting a direction at intersections without specific destination) as the background traffic. In the following, unless otherwise stated, we use "vehicle" to refer to expert autopilot for simplicity.

Vehicles collect data at 2 frames per second of their surroundings in the simulated world Each frame contains a BEV map produced by CARLA, the location of the vehicle and assistant information needed for training local iterations. We firstly run vehicles for 1 hour in the simulated world, so each vehicle collects data of 7200 frames used for training local iterations. To simulate decentralized peer model training with on-road vehicles, we needs locations of vehicles to construct time-varying communication topology (i.e., base graphs). We run vehicles in the same simulated world for another 120 hours

to collect sufficient locations at 2 fps.

We train one model for each vehicle simultaneously, simulate communication among vehicles and perform model aggregation. We use the same loss function as [3] in training models[1]. The hyperparameters of all models are the same. The learning rate and batch size are set to $1e^{-4}$ and 32. No weight decay is used in training. We set the window size in Eqn. (9) $H = 15$. We simulate vehicular communication using parameters similar to IEEE 802.11bd [29] [23]: data rate with packet size of 1500 bytes is about 31Mbps, communication range is 500m, with up to three retransmissions per packet. We set the number of available subcarriers $R = 30$ and subdivide data rates among subcarriers as [19]. We also build a look-up table of packet reception rate versus distance based on the results in [10]. We generate the base graphs of vehicles based on the collected vehicle locations.

Regarding route sharing, a vehicle shares its speed, current location, and route in the next 30 seconds (60 waypoints) in each round. The total size of route information in one round is about 184 bytes. The size of a 32×32 base graph adjacency matrix is 1KB. The size of our driving decision making model is about 52MB. We utilize a compression approach [30] that reduces the model size to 0.2MB, amounting to about 133 packets. Thus sending/receiving one model on one subcarrier with retransmissions takes about 2.6 seconds on average, which is treated as one unit time in our experiments. We set the per round communication time budget $B = 2$, which corresponds to 5.2 seconds.

We conduct experiments using an NVIDIA RTX2060 GPU, which takes 10 milliseconds to finish Algorithm 1 and 0.8 seconds to solve Eqn. (3) on average. With TFLOPS-level onboard GPUs [31] these will become negligible. We focus on communication time and ignore such local computation time. In the experiments, we consider two cases of 32 and 16 vehicles, respectively, representing different vehicle densities in training scenarios. In the case of 16 vehicles, they are randomly chosen from all 32 ones before training.

B. Benchmarks

We compare RoADTrain with the following benchmarks.

- FedAvg [18] is the classic federated learning scheme. Workers train local iterations and update models to a central server for aggregation. For FedAvg, we do not consider data loss, number of subcarriers or per round communication time constraints. We regard FedAvg as an upper bound for model training performance.
- SGP [12] is a typical scheme for decentralized learning on time-varying directed graphs. Similarly, we do not consider the same three constraints thus it serves as another upper bound for fully decentralized training. We assume vehicles share models with all one hop neighbors at each round without failures.
- C-SGP is a constrained version of SGP that uses the same local model updating scheme but is constrained by the number of subcarriers and per round communication time. At each round, C-SGP randomly selects $R*B$ edges on the base graph for model sharing to ensure that the communication will be finished within the communication time budget. We assume losses from wireless and mobility can lead to unexpected sharing failures.
- C-MATCHA is a constrained version of MATCHA [32]. MATCHA is representative in topology construction based decentralized learning work, which decomposes the base graph into node pairs and selects some node pairs to construct a sparse subgraph at each epoch for model sharing under the communication time budget. C-MATCHA considers the number of subcarriers and per round communication time constraint. We implement C-MATCHA similar to [19]. We also assume that losses from wireless and mobility lead to sharing failures.

[1]Note that in actual training, due to wireless loss, models on vehicles may not be exactly the same as described in (1). We consider training converged when the differences among models are small enough.

C. Topology analysis

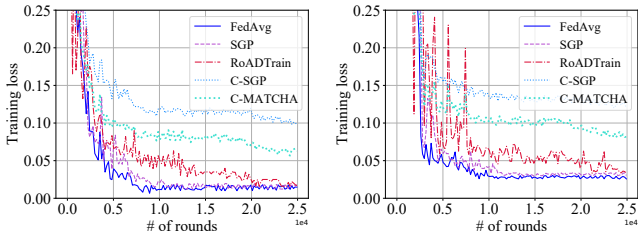
Before evaluating model performance, we analyze the characteristics of the base graphs generated from the collected locations of vehicles. With 32 and 16 vehicles, we observe that there are on average 633, 155 edges on the base graph, respectively. The connected vehicle cluster takes about 33 seconds and 8 seconds, respectively, to finish sharing compressed models among all one hop neighbors. These may significantly exceed the contact duration lengths.

Under RoADTrain, only selected neighbors are chosen to share model. Thus the per round communication time can be bounded into a predefined budget B. When $B = 2$ (5.2 seconds per round), vehicles can finish one round with a relatively stable topology. We also observe that in the case of 16 vehicles, the cluster may split into disconnected components in the middle of the whole training process, where vehicles within each component continue the sharing and training. Due to the closed map, vehicles will keep moving and become connected again. Thus we observe that training still converges. We will discuss open maps in Section V.

D. Results of model training

We first study the model training speed in terms of rounds. Fig. 2(a) and Fig. 2(b) summarize the results for 32 and 16 vehicles, correspondingly. For 32 vehicles, RoADTrain converges to the same training loss as FedAvg and SGP, after $2.6\times$ and $2.4\times$ more rounds, respectively. For 16 vehicles, it converges to similar training losses, using $2.4\times$ and $2.3\times$ additional rounds, respectively. While C-SGP and C-MATCHA still have much larger loss even after 2.5e4 rounds, possibly due to insufficient training.

We validate the consideration by studying the per round model sharing failure rate of decentralized benchmarks and RoADTrain, which is defined by the ratio between the edges with sharing failure and all selected edges for sharing in each round. The results are summarized in Table I. We observe that both C-SGP and C-MATCHA have much (at most 40%) higher



(a) n=32 (b) n=16

Fig. 2. Results of training loss vs. # of rounds

per round sharing failure rates on average than RoADTrain in the two cases of 32 and 16 vehicles. This is because without the help of route information, C-SGP and C-MATCHA can easily select edges with high wireless loss or too short contact durations, causing frequent sharing failures. While the average per round sharing failure rate in RoADTrain is only about 12% and 14% in two cases, respectively, since RoADTrain takes advantage of route information to be aware of and deprioritize edges with high probabilities of sharing failure. The differences in such edges can partly explain that why RoADTrain can converge to a much smaller training loss than the two benchmarks.

TABLE I
PER ROUND MODEL SHARING FAILURE RATE ON AVERAGE (%)

Number of vehicles (n)	C-SGP	C-MATCHA	RoADTrain
32	51	49	12
16	54	52	14

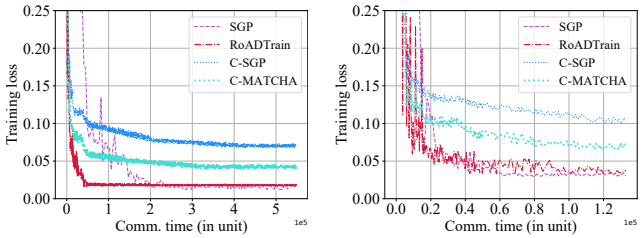
We note that vehicles may split into disconnected components in the middle of training in the case of 16 vehicles due to the lower density of vehicles. The results in Fig. 2(b) demonstrate that RoADTrain can still converge at marginally 0.016 higher loss and 15% more rounds. This is because these vehicles move in a closed map, thus they will meet and become connected again. RoADTrain maximizes the algebraic connectivity of not just the current round, but a few consecutive rounds (see Eqn. (9)). It tends to select edges connecting different components for sharing models, thus compensating insufficient information exchange in cluster splitting rounds. So such temporary cluster splitting does not harm RoADTrain much. Of course, such reconnection may not happen in an open map, which we will discuss in Section V.

We also observe that given limited data on each vehicle, more vehicles joining the training bring more data, thus achieving lower training loss with less fluctuation. E.g., loss in the case of 32 vehicles is 44% lower than that of 16 vehicles.

Next we study the training loss in terms of communication time (in unit). Recall that one unit time corresponds to the time of sending/receiving one compressed model over one subcarrier on one edge. Fig. 3(a) and Fig. 3(b) summarize the results for 32 and 16 vehicles, respectively. In the case of 32 vehicles, RoADTrain converges to the same training loss as SGP and with 4.5× total communication time reduction. RoADTrain cuts down per round communication time by 11×. This is due to much less but highest quality neighbors selected for model sharing, whereas in SGP all neighbors are selected. Even with 2.4× more rounds to train, RoADTrain still converges 4.5× faster.

For 16 vehicles, RoADTrain converges to the same training loss as SGP at 1.2× faster convergence. It cuts down per round communication time by 2.7×, because in SGP there are simply much less neighbors with 16 vehicles. Thus 2.3× more rounds leads to 1.2× faster convergence.

We also observe that C-SGP and C-MATCHA do not converge to the same training loss as SGP or RoADTrain, similar to the results in Fig. 2. This is because they both choose neighbors indiscriminately, thus losses from wireless and mobility cause frequent sharing failures. The two approaches still suffer from the insufficient training.



(a) n=32 (b) n=16

Fig. 3. Results of training loss vs. Communication time (in unit)

The results demonstrate that RoADTrain can converge to similar training losses as federated learning and communication-unconstrained decentralized training, under comparable or even less (4.5×) time by optimized neighbor selection in model sharing, assuming negligible computing time with modern hardware.

E. Online evaluation

Next we conduct an online testing using driving success rate in the CARLA benchmark as the metric [28]. After training, the model is deployed on a testing autopilot provided with next few waypoints to go in an urban driving setting. The original CARLA benchmark consists of four different driving conditions including driving straight (Straight), driving with one turn (One Turn), full navigation with multiple turns (Navi. (Empty)), and the same full navigation routes but with traffic (Navi. (Normal)), each of which has 25 predefined navigation routes. Besides these four conditions, we test models with a more challenging condition called Navi. (Dense), where the total number of roaming cars and pedestrians is 1.2× that Navi. (Normal). A trial on a given route is considered successful if the testing autopilot arrives at the destination within a given time without hitting other vehicles or pedestrians.

Table II and Table III show the driving success rate with 32 and 16 vehicles. Compared to FedAvg or SGP, we observe that RoADTrain can achieve competitive driving success rate in all driving conditions, with at most $5-6\%$ less success in both cases of 32 vehicles and 16 vehicles. This echos training loss

comparison results that RoADTrain can achieve performance competitive to unconstrained upper bound approaches, even in presence of unreliable wireless communication and highly dynamic, ever-changing topology of vehicles.

In addition, RoADTrain outperforms C-SGP and C-MATCHA significantly in both two cases. In the case with 32 vehicles, RoADTrain achieves 23% and 17% driving success rate improvements in Navi. (Dense) compared to C-SGP and C-MATCHA, respectively. In the case with 16 vehicles, RoADTrain achieves 27% and 18% driving success rate improvements in Navi. (Dense), respectively. This further demonstrate that, with the help of route information, RoADTrain can successfully enable peer-wise vehicular model training while driving in a fully decentralized way with unreliable wireless communication and high mobility of vehicles.

TABLE II
DRIVING SUCCESS RATE (%) (N=32)

Task	FedAvg	SGP	C-SGP	C-MATCHA	RoADTrain
Straight	100	100	96	98	100
One Turn	100	100	93	96	100
Navi. (Empty)	98	98	76	82	95
Navi. (Normal)	94	93	64	72	89
Navi. (Dense)	83	82	54	60	77

TABLE III
DRIVING SUCCESS RATE (%) (N=16)

Task	FedAvg	SGP	C-SGP	C-MATCHA	RoADTrain
Straight	100	100	93	96	100
One Turn	100	100	87	90	99
Navi. (Empty)	91	91	70	76	90
Navi. (Normal)	85	83	55	64	80
Navi. (Dense)	75	74	42	51	69

F. Non-identical base graphs

As mentioned in Section III-D, base graphs constructed on different vehicles at each round may have small differences due to wireless losses in discovery. We simulate the situation by adding disturbances to adjacency matrices of base graphs. Specifically, '1' in a adjacency matrix (indicating one vehicle can communicate with another) have a probability to become '0', if the transmitted short discovery message is lost. Table. IV shows the driving success rates of models trained with non-identical base graphs in the case of 32 and 16 vehicles, respectively. We observe that when the probability equals 0.01, compared to the RoADTrain model trained with identical base graphs, the driving success rate reductions are almost negligible (at most 2% in Navi. (Normal) with 16 vehicles). When the probability equals 0.1, the driving success rate reductions are at most 5%. In reality, assuming a lower transmission success of 0.6 on average, the probability of '1' changing to '0' is about 0.026 given three retransmissions in 802.11bd [23], lower than 0.1 we set. The probability can be even lower when adding HARQ technique. The results demonstrate that RoADTrain is robust under different base graphs during discovery.

TABLE IV
DRIVING SUCCESS RATE WITH NON-IDENTICAL BASE GRAPHS (%)

Task	P.=0.01(n=32)	P.=0.1(n=32)	P.=0.01(n=16)	P.=0.1(n=16)
Straight	100	100	100	100
One Turn	100	98	99	97
Navi. (Empty)	95	92	89	86
Navi. (Normal)	88	86	78	75
Navi. (Dense)	76	73	68	64

G. Window size and ablation study

We further evaluate the effects of window size H in Eqn. (9). Recall that we set $H = 15$ by default. We consider two additional different values (5 and 30) for H. The corresponding driving success rates are shown in Table. V. We observe that too small or too large window size can hurt model performance with at most 7% (in Navi. (Dense) with 16 vehicles) and 5% (in Navi. (Normal / Dense) with 16 vehicles) driving success rate reduction. This is because too small window size may not contain enough historical information, while too large window size contains obsolete historical information, both detrimental to subgraph selection. We plan to study an adaptive window size selection strategy in the future.

TABLE V
DRIVING SUCCESS RATE WITH DIFFERENT WINDOW SIZE H (%)

Task	H=5(n=32)	H=30(n=32)	H=5(n=16)	H=30(n=16)
Straight	100	100	100	100
One Turn	100	100	98	99
Navi. (Empty)	92	92	87	88
Navi. (Normal)	86	85	74	75
Navi. (Dense)	73	74	62	64

We also evaluate the effects of algebraic connectivity optimization with an ablation study. Instead of generating communication graph by solving Eqn. (9), we randomly select B number of subgraphs and test the trained model performance using the same settings as online evaluation. Table VI shows that the driving success rates by conducting random subgraph selection. We observe that driving success rate reduces at most 7% (in Navi. (Dense)) with 32 vehicles and 9% (in Navi. (Dense)) with 16 vehicles. The results demonstrates that optimizing the algebraic connectivity of the communication graph indeed effectively benefit model training.

TABLE VI
DRIVING SUCCESS RATE (%) WITH RANDOM SUBGRAPH SELECTION

Task	n=32	n=16
Straight	100	100
One Turn	100	98
Navi. (Empty)	90	84
Navi. (Normal)	83	72
Navi. (Dense)	70	60

V. DISCUSSION

Model training under open maps or large clusters. Since our interests mainly lie in addressing the data loss and short

contact duration challenges, we focus on a closed map with a fixed set of vehicles in simulation where vehicles do not move in/out of the map (like in [19]). The vehicle cluster may split into unconnected components, which can reconnect as vehicles keep moving. In a more realistic open map where vehicles can freely move out and in, further adaptations may be needed to achieve effective training, e.g., flexible device participation [33]. Another issue is possibly very large clusters when many vehicles are together (e.g. in a traffic jam), which may cause higher computation load. We can divide such a large vehicle cluster into smaller clusters to avoid large computation latency [25]. We are also interested in characterizing different traffic flow dynamics (e.g., regular vs. rush hour) to gain further insights into vehicular model training.

Improvements on embedding and subcarrier reuse. For simplicity, we use a quite straightforward method (Eqn. (3), (7)) to embed the route information into edge weights, and it shows good performance. More sophisticated methods (e.g. [34]) can be explored. In communication, two senders may use the same subcarrier if they do not cause the hidden terminal problem. This may further increase the degree of parallelism and accelerate model sharing. We leave these possible improvements in the future.

Accuracy of simulation results. Conducting real-world experiments using real vehicles, particularly on a medium or large scale requires substantial hardware, manual, and financial resources. As an alternative, we use CARLA, a widely-used simulator in autonomous driving research. CARLA can simulate various types of maps and driving environments with a high degree of realism, making it a reliable means for experimental studies. Previous research, such as [35], has shown that when appropriately configured, results generated through the CARLA simulator are nearly identical to those obtained from real-world experiments. Therefore, we believe our results are reasonably close to real-world experiments.

Heterogeneity in vehicles. Recall that we assume all vehicles are equipped with similar hardware and employ identical communication technology, with communication interference being evaluated in a subcarrier-specific manner. In the real world, vehicles may possess diverse hardware and communication technologies, which could potentially result in greater losses, which can still be accommodated within existing work using a larger loss probability. We will study the impact of such heterogeneity on performance and adapt it to a more generalized scenario in the future.

Other radios suitable for vehicles. We use radio parameters similar to those of 802.11bd. New Radio V2X (NR-V2X) [36] is a promising vehicular communication technology, which has no backward compatibility constraints and shows improvements over predecessors. Recent data-centric radios have also shown high-rate, low-loss multicast capability [37], ideal for a vehicle to share a model with multiple neighbors, greatly reducing the latency.

Incentives. As vehicular crowd sensing and collaborative learning emerge as a promising diagram, some work discusses the incentives or markets to stimulate cooperation among vehicles/drivers [38], [39]. They aim to stimulate more participants while striking a balance between the conflicting interests of the crowd sourcing platform and drivers. These are orthogonal to our work and we simply assume sufficient incentives exist.

VI. Related Work

We contextualize our work within prior results on (i) vehicular collaboration and (ii) decentralized deep model learning.

Recent works show the effectiveness of federated learning in collaborative vehicle learning from different aspects [40]. While with constrained communication bandwidth, it is possible central server becomes the communication bottleneck of the infrastructure [5]. Some other works inspect collaborative, peer vehicle perceptions with Vehicle-to-Vehicle communication under limited bandwidth. The central intuition is to fuse observations from multiple vehicles to get a complete, non-occlusive view. [41] propose a sensor data fusion scheme to enhance perceptive ability with LiDAR 3D point clouds collected from different positions and angles of connected vehicles. [42] proposes a data-sharing policy that minimizes the amount of information disclosed for the cooperative perception of autonomous vehicles. [43] proposes a scheme for one-hop multicasting of high-volume sensor data to improve the communication completion ratio. Unlike RoADTrain, which optimizes the training process under unreliable wireless communication and short contact duration, these works focus mainly on collaborative environment perception or task scheduling.

Recent research has introduced various schemes for decentralized deep model learning in the vehicular scenario. For instance, [44] proposes a reinforcement learning based approach for multi-agent cooperation in a lane change scenario. Some works explore the trade-off between communication efficiency and convergence accuracy in distributed deep model training. [45] presents a divide-and-shuffle synchronization to realize communication efficiency without sacrificing convergence accuracy. These works are mainly circumstanced in data centers where model and data are partitioned among multiple GPUs to accelerate deep model training, where the communication topology between workers is fixed, and the communication among workers is reliable. RoADTrain focuses on peer model training in a different vehicular scenario with these works.

VII. Conclusion

We explore an fully-decentralized model training paradigm for on-road vehicles without assistance from central servers and infrastructures. To address the model sharing failure among vehicles due to unreliable wireless communication and short contact duration, we propose a route assisted topology driven peer model training approach named "RoADTrain". The main ideas include selecting a subset of high-quality edges from the communication topology for model sharing to facilitate the information flow among vehicles. Empirical results on the driving decision making task demonstrate that RoADTrain achieves comparable driving success rates and faster convergence than representative unconstrained decentralized learning

methods, and significantly outperforms other benchmarks that consider losses in the hardest driving conditions.

REFERENCES

[1] Y. Ma, Z. Wang, H. Yang, and L. Yang, "Artificial intelligence applications in the development of autonomous vehicles: a survey," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 2, pp. 315–329, 2020.

[2] F. Codevilla, E. Santana, A. M. López, and A. Gaidon, "Exploring the limitations of behavior cloning for autonomous driving," in *ICCV 2019*, 2019, pp. 9329–9338.

[3] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, "Learning by cheating," in *ICRA 2020*, 2020.

[4] Y. Zhao, J. Zhao, M. Yang, T. Wang, N. Wang, L. Lyu, D. Niyato, and K.-Y. Lam, "Local differential privacy-based federated learning for internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8836–8853, 2020.

[5] Z. Zhang, S. Wang, Y. Hong, L. Zhou, and Q. Hao, "Distributed dynamic map fusion via federated learning for intelligent networked vehicles," in *ICRA 2021*. IEEE, 2021, pp. 953–959.

[6] J. Xu and H. Wang, "Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective," *IEEE Transactions on Wireless Communications*, vol. 20, no. 2, pp. 1188–1200, 2020.

[7] X. Zhang, A. Zhang, J. Sun, X. Zhu, Y. E. Guo, F. Qian, and Z. M. Mao, "Emp: Edge-assisted multi-vehicle perception," in *MobiCom 2021*, 2021, pp. 545–558.

[8] M. A. Dinani, A. Holzer, H. Nguyen, M. A. Marsan, and G. Rizzo, "Gossip learning of personalized models for vehicle trajectory prediction," in *WCNCW 2021*. IEEE, 2021, pp. 1–7.

[9] G. Di Giacomo, J. Härri, and C. F. Chiasserini, "Edge-assisted gossiping learning: Leveraging v2v communications between connected vehicles," in *ITSC 2022*. IEEE, 2022, pp. 3920–3927.

[10] W. Anwar, N. Franchi, and G. Fettweis, "Physical layer evaluation of v2x communications technologies: 5g nr-v2x, lte-v2x, ieee 802.11 bd, and ieee 802.11 p," in *VTC2019-Fall*. IEEE, 2019, pp. 1–7.

[11] "Nvidia drive thor strikes ai performance balance, uniting av and cockpit on a single computer." [Online]. Available: https://https://blogs.nvidia.com/blog/2022/09/20/drive-thor/

[12] A. Nedić and A. Olshevsky, "Stochastic gradient-push for strongly convex functions on time-varying directed graphs," *IEEE Transactions on Automatic Control*, vol. 61, no. 12, pp. 3936–3947, 2016.

[13] A. Nedić, A. Olshevsky, and M. G. Rabbat, "Network topology and communication-computation tradeoffs in decentralized optimization," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 953–976, 2018.

[14] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic control*, vol. 57, no. 3, pp. 592–606, 2011.

[15] B. Recht, C. Re, S. Wright, and F. Niu, "Hogwild!: A lock-free approach to parallelizing stochastic gradient descent," *Advances in neural information processing systems*, vol. 24, 2011.

[16] G. Neglia, G. Calbi, D. Towsley, and G. Vardoyan, "The role of network topology for distributed machine learning," in *INFOCOM 2019*. IEEE, 2019, pp. 2350–2358.

[17] A. Kadav and E. Kruus, "Asap: asynchronous approximate data-parallel computation," *arXiv preprint arXiv:1612.08608*, 2016.

[18] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *AISTATS 2017*. PMLR, 2017, pp. 1273–1282.

[19] Z. Meng, H. Xu, M. Chen, Y. Xu, Y. Zhao, and C. Qiao, "Learning-driven decentralized machine learning in resource-constrained wireless edge computing," in *INFOCOM 2021*. IEEE, 2021, pp. 1–10.

[20] W. Ahn and R. Y. Kim, "Distributed triggered access for bsm dissemination in 802.11 bd v2v networks," *Applied Sciences*, vol. 10, no. 1, p. 311, 2019.

[21] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," *NeuralPS 2017*, vol. 30, 2017.

[22] A. Du, Y. Shen, and L. Tseng, "Carml: distributed machine learning in vehicular clouds," in *MobiCom 2020*, 2020, pp. 1–3.

[23] S. Zeadally, M. A. Javed, and E. B. Hamida, "Vehicular communications for its: Standardization and challenges," *IEEE Communications Standards Magazine*, vol. 4, no. 1, pp. 11–17, 2020.

[24] C. W. Wu, "Algebraic connectivity of directed graphs," *Linear and multilinear algebra*, vol. 53, no. 3, pp. 203–223, 2005.

[25] L. Zhang and H. El-Sayed, "A novel cluster-based protocol for topology discovery in vehicular ad hoc network," *Procedia Computer Science*, vol. 10, pp. 525–534, 2012.

[26] L. Zhang, H. Elsayed, and E. Barka, "A novel location service protocol in multi-hop clustering vehicular ad hoc networks," in *2011 International Conference on Innovations in Information Technology*. IEEE, 2011, pp. 386–391.

[27] A. Ahmed, A. Al-Dweik, Y. Iraqi, H. Mukhtar, M. Naeem, and E. Hossain, "Hybrid automatic repeat request (harq) in wireless communications systems and standards: A contemporary survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2711–2752, 2021.

[28] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *CoRL 2017*. PMLR, 2017, pp. 1–16.

[29] "Ieee p802.11 - next generation v2x study group." [Online]. Available: https://www.ieee802.org/11/Reports/tgbd_update.htm

[30] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," *arXiv preprint arXiv:1712.01887*, 2017.

[31] "Tesla's new hw3 self-driving computer — it's a beast (cleantechnica deep dive)." [Online]. Available: https://cleantechnica.com/2019/06/15/teslas-new-hw3-self-driving-computer-its-a-beast-cleantechnica-deep-dive

[32] J. Wang, A. K. Sahu, Z. Yang, G. Joshi, and S. Kar, "Matcha: Speeding up decentralized sgd via matching decomposition sampling," in *2019 Sixth Indian Control Conference*. IEEE, 2019, pp. 299–300.

[33] Y. Ruan, X. Zhang, S.-C. Liang, and C. Joe-Wong, "Towards flexible device participation in federated learning," in *AISTATS 2021*. PMLR, 2021, pp. 3403–3411.

[34] G. Sun, R. Liang, H. Qu, and Y. Wu, "Embedding spatio-temporal information into maps by route-zooming," *IEEE transactions on visualization and computer graphics*, vol. 23, no. 5, pp. 1506–1519, 2016.

[35] S. Shi, J. Cui, Z. Jiang, Z. Yan, G. Xing, J. Niu, and Z. Ouyang, "Vips: real-time perception fusion for infrastructure-assisted autonomous driving," in *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, 2022, pp. 133–146.

[36] "3gpp: Etsi work programme report." [Online]. Available: http://www.3gpp.org/release-16

[37] M. Elbadry, F. Ye, P. Milder, and Y. Yang, "Pub/sub in the air: A novel data-centric radio supporting robust multicast in edge environments," in *2020 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 2020, pp. 257–270.

[38] R. Ding, Z. Yang, Y. Wei, H. Jin, and X. Wang, "Multi-agent reinforcement learning for urban crowd sensing with for-hire vehicles," in *INFOCOM 2021*. IEEE, 2021, pp. 1–10.

[39] C. Xiang, Y. Li, Y. Zhou, S. He, Y. Qu, Z. Li, L. Gong, and C. Chen, "A comparative approach to resurrecting the market of mod vehicular crowdsensing," in *Proc. IEEE Conf. Comput. Commun*, 2022, pp. 1–10.

[40] W. Ni, S. Zhu, M. M. Karim, A. Asheralieva, J. Kang, Z. Xiong, and C. Maple, "Lagrange coded federated learning (l-cofl) model for internet of vehicles," in *ICDCS 2022*. IEEE, 2022, pp. 864–872.

[41] Q. Chen, S. Tang, Q. Yang, and S. Fu, "Cooper: Cooperative perception for connected autonomous vehicles based on 3d point clouds," in *ICDCS 2019*. IEEE, 2019, pp. 514–524.

[42] C. Qiu, S. Yadav, A. Squicciarini, Q. Yang, S. Fu, J. Zhao, and C. Xu, "Distributed data-sharing consensus in cooperative perception of autonomous vehicles," in *ICDCS 2022*. IEEE, 2022, pp. 1212–1222.

[43] J. Shen, H. Zhu, Y. Cai, B. Zhai, X. Wang, S. Chang, H. Cai, and M. Guo, "mmv2v: Combating one-hop multicasting in millimeter-wave vehicular networks," in *ICDCS 2022*. IEEE, 2022, pp. 735–742.

[44] Z. Liang, J. Cao, S. Jiang, D. Saxena, and H. Xu, "Hierarchical reinforcement learning with opponent modeling for distributed multi-agent cooperation," in *ICDCS 2022*. IEEE, 2022, pp. 884–894.

[45] W. Wang, C. Zhang, L. Yang, K. Chen, and K. Tan, "Addressing network bottlenecks with divide-and-shuffle synchronization for distributed dnn training," in *INFOCOM 2022*. IEEE, 2022, pp. 320–329.