Neural Network based Partial Tomography for In-Vehicle Network Monitoring

Amani Ibraheem*, Zhengguo Sheng*, George Parisis*, Daxin Tian†
*School of Engineering and Informatics, University of Sussex, UK
†School of Transportation Science and Engineering, Beihang University, China
Email: *{A.Ibraheem, Z.Sheng, G.Parisis}@sussex.ac.uk, †Dtian@buaa.edu.cn

***Abstract*—In-vehicle network monitoring is one of the important elements in vehicular network management and security. Most of the existing network monitoring approaches rely on measuring every part of the network. Such approaches overburden the network by transmitting active probes. In this work, we propose a new in-vehicle network monitoring approach that benefits from network tomography and the advances in deep learning to infer the network delay performance. Specifically, the available measurements can be used to estimate the performance of the remaining network where direct measurements cannot be applied. Performance evaluation has been conducted using in-vehicle network simulation with different TSN (Time-Sensitive Network) traffics and the proposed monitoring approach shows the delay estimation accuracy of up to 99%.**

I. INTRODUCTION

A. Background and Motivation

Network monitoring plays an indispensable role in network performance management. It can help achieve better management of network resources such as in load-balancing and bandwidth allocation. Moreover, with the new application of Ethernet-based vehicular architecture, vehicles will be more vulnerable to IP based attacks such as black holes and denial-of-service (DoS). Therefore, network monitoring is one of the key tasks in securing the network by which it can detect anomalies caused by such attacks. Current network monitoring approaches rely on direct auditing of full network including the internal network devices [1]. In in-vehicle network, direct measurement of network elements (e.g., ECU/links) is, however, not feasible due to traffic overhead, mission-critical requirements and difficulty in accessing a closed in-vehicle system.

Inspired by Vardi's work [2], an indirect approach called *Network Tomography* that is based on end-to-end network measurement can be used to infer the network's link-level metrics. Network tomography eliminates the need for internal measurement, and hence, it incurs less overhead in total. End-to-end measurements can be obtained using either active or passive monitoring. Active monitoring relies on actively sending designated probes for the purpose of measuring the end-to-end performance, while passive monitoring exploits the existing traffic. In this paper, we propose a novel approach for inferring link-level and path-segment as well as end-to-end path-level performance metrics in an accurate and timely fashion, by measuring performance along selected paths. To do so, we combine traditional network tomography with deep learning; the former is used to infer performance metrics in the network, partially, depending on the availability of passive, end-to-end measurements; the latter takes as input the results of this partial network tomography in order to train a deep neural network that is then used to estimate values of path-level metrics for the whole network. End-to-end path metrics can be trivially calculated by aggregating inferred (by partial network tomography) and estimated (by our trained model) metrics. The key motivation behind our work is the fact that link-level performance measurements may not always be available, feasible or economical to measure at all times in in-vehicle networks, especially using active measurements. At the same time, the performance characteristics of the network may be varying depending on the state of the vehicle and the drivers/passengers usage patterns of its various subsystems (e.g. for infotainment). In this paper, we focus on inferring and estimating delays but we believe that our approach is applicable to a wider range of performance metrics.

This paper has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 101006411.

B. Related Work

Most of existing works, [3], [4], employ active measurements to monitor network performance. Conversely, limited works have examined passive measurements [5]–[7]. Similar to these studies, we consider passive tomography, with end-to-end delay measurements. The desire to use passive tomography is due to multiple reasons: first, for a complicated in-vehicle network, inserting large number of probes might consequently affect the mission-critical network performance. Second, with passive tomography, the existing traffic can provide more realistic and accurate measurements than the ones with inserted probes which are different to the actual traffic. Third, the issues related to placing and minimising the number of monitors [8] are eliminated with passive tomography. Hence, for the aforementioned reasons, we employ passive measurements. However, the approach can be extended to active measurements.

On the other hand, for machine learning-based solutions, there are very limited number of studies, such as [9], [10], that consider employing machine learning with network tomography. Authors in [10] used neural network to infer link-

level parameters where they focus on optical networks and link attenuation. Neural network is utilised in [9] to estimate path-level performance based on measuring parts of the network. Our work is different to the approach presented in [9] in that it deals with varying performance characteristics in the network, and does not rely on active measurements. Moreover, in our proposal, end-to-end measurements are used to extract link-level performance metrics through partial network tomography, and as a result, training our neural network-based model is done with both end-to-end and link-level data.

C. Contribution

In this work, we propose a new monitoring approach for in-vehicle network that is based on network tomography and deep neural network. In particular, we assume that only part of end-to-end measurements are available. We use such available measurements to infer metrics of the internal network elements where direct monitoring is not possible. Further, we use the available measurements to estimate unmeasured parts of the network using neural networks. Our contribution is summarised below:

- We propose a partial network tomography based monitoring approach to infer link-level metrics without the need to access internal network elements using only end-to-end measurements of existing traffic.
- Using the available end-to-end measurements, we further estimate the network performance of unmeasured subset using deep neural networks. The proposed approach can estimate the performance with up to 99% accuracy.
- With OMNeT++ simulation and evaluation, we show that it is possible to infer overall in-vehicle network performance by only measuring a subset of the network.

The rest of the paper is organized as follows. Section II describes our network tomography model and problem statement. Our partial network tomography approach is introduced in Section III. Section IV describes our neural network based delay estimation approach. Simulation results and conclusion are discussed in Section V and Section VI respectively.

II. System Model and Problem Statement

A. Network Tomography Model

In this work, we assume that the in-vehicle network topology is known. We map the topology into undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{L})$ containing a set of nodes $\mathcal{V}$ interconnected through a set of links $\mathcal{L}$. A link $(u, v) \in \mathcal{L}$ connects two adjacent nodes $u, v \in \mathcal{V}$. Let $\gamma := |\mathcal{L}|$ denotes the total number of links in $\mathcal{G}$. For a network $\mathcal{G}$ shown in Fig. 1, $\mathcal{E} = \{0, 2, 4, 5\}$ and $\mathcal{R} = \{1, 3\}$, are edge nodes (e.g., radio interface connected to external world) and intermediate nodes (e.g., switches), respectively, where $\mathcal{E}, \mathcal{R} \subset \mathcal{V}$ and $\mathcal{E} \vee \mathcal{R} = \mathcal{V}$.

Definition 1: Given a network $\mathcal{G}$, the edge nodes in $\mathcal{E}$ are nodes with $d_i = 1$ and intermediate nodes in $\mathcal{R}$ are nodes with $d_i > 1$, where d_i is the degree of node $i \in \mathcal{V}$.

Let $\mathcal{P} = \{p_1, p_2, \ldots, p_z\}$ represents a set of all end-to-end paths (note that we use the terms "end-to-end path" and "path" interchangeably) and let $z := |\mathcal{P}|$ denotes the total number of paths in $\mathcal{G}$. A path $p_i = \{(u, a), \ldots, (b, v)\}$ represents a pair of edge nodes $(u, v) \in \mathcal{E}$ communicating with each other where $(u, a), (b, v) \in \mathcal{L}$ and $a, b \in \mathcal{R}$. We refer to (u, v) as path-segment if $u, v \in \mathcal{V}$ are not adjacent and $u \wedge v \notin \mathcal{E}$. An example of path-segment in Fig. 1 is $(0, 3)$.

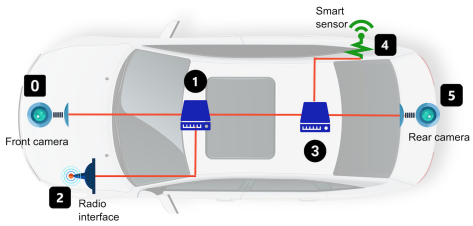


Fig. 1: In-vehicle network represented as graph with six nodes.

We formulate our network tomography problem into the following linear system:

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \tag{1}$$

where $\boldsymbol{y}$ is a vector of end-to-end measurements for paths in $\mathcal{P}$, $\boldsymbol{A}$ is a $k \times \gamma$ routing matrix, and $\boldsymbol{x}$ is a vector of link-level performance for links in $\mathcal{L}$ that we are interested to infer. We assume that the routing matrix is deterministic, therefore each node pair $(u, v) \in \mathcal{E}$ uses a single path to communicate and hence the routing matrix $\boldsymbol{A}$ includes entries $a_{ij} \in \{0, 1\}$, where $a_{ij} = 1$ if path i passes through link j, and $a_{ij} = 0$ otherwise. Let $\mathcal{M} \subset \mathcal{P}$ be a set of measured paths and let $k := |\mathcal{M}|$ and $\eta = z - k$ denote the total number of measured paths and unmeasured paths, respectively.

For the above network, there are five links $\mathcal{L} = \{(0, 1), (1, 2), (1, 3), (3, 4), (3, 5)\}$, and we assume throughout the paper that the set of paths is $\mathcal{P} = \{p_1, p_2, p_3, p_4, p_5\}$ with $p_1 = \{(0, 1), (1, 2)\}$, $p_2 = \{(0, 1), (1, 3), (3, 4)\}$, $p_3 = \{(0, 1), (1, 3), (3, 5)\}$, $p_4 = \{(2, 1), (1, 3), (3, 4)\}$, and $p_5 = \{(4, 3), (3, 5)\}$.

B. Problem Statement

In order to uniquely solve $\boldsymbol{x}$ in (1), the routing matrix $\boldsymbol{A}$ used for the measurement should be invertible, and for $\boldsymbol{A}$ to be invertible, it has to be full-rank square matrix. Specifically, for our network tomography problem, in order to identify all link-level metrics, two conditions should be satisfied: (i) the number of end-to-end measurements should be equal to the number of links in the network ($k = \gamma$), and (ii) the available end-to-end measurements should be linearly independent. The first condition ensures that $\boldsymbol{A}$ will be square matrix, while the second condition ensures that the square matrix $\boldsymbol{A}$ is full-rank matrix ($rank(\boldsymbol{A}) = \gamma$). Because we focus on passive tomography that is based on measuring the existing traffic, the available measurements can form a routing matrix that can be either:

1) full-rank matrix with $rank(\boldsymbol{A}) = \gamma$ and $k = \gamma$.
2) rank-deficient matrix with $rank(\boldsymbol{A}) < \gamma$ and $k = \gamma$.
3) rank-deficient matrix with $rank(\boldsymbol{A}) < \gamma$ and $k < \gamma$.

The first case satisfies both conditions (i) and (ii), and therefore $\boldsymbol{x}$ can be uniquely identified using (1). The second case only

satisfies condition (i), while the third case does not satisfy either conditions. Thus, the last two cases cannot use (1) to infer all link-level metrics in the network. In this paper we assume that the in-vehicle network falls under either one of the last two cases where the available measurements cannot form a full rank-matrix. Consider the network shown in Fig. 1, there are five links, $\gamma = 5$, therefore, to uniquely solve $\boldsymbol{x}$ in (1), five independent measurements should be available, i.e., $k = 5$, and hence $\boldsymbol{A}$ should be full-rank with $rank(\boldsymbol{A}) = 5$. However, the matrix in in-vehicle network is unlikely to form a full-rank matrix due to dependent or inadequate end-to-end measurements. For instance, suppose that the measured paths for network $\mathcal{G}$ in Fig. 1 are $\mathcal{M} = \{p_1, p_2, p_5\}$, hence (1) can be written as

$$\begin{pmatrix} y_1 \\ y_2 \\ y_5 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} \tag{2}$$

In (2), only limited number of measurements are available (y_1, y_2, y_5) for paths in $\mathcal{M}$, while others are unknown (y_3, y_4), for paths in $\mathcal{P} \backslash \mathcal{M}$. Besides, $k < \gamma$ which makes the routing matrix rank-deficient with $rank(\boldsymbol{A}) = 3$.

In the following sections, we show that (1) can be used to infer part of link-level and/or path-segment metrics using partial network tomography. In addition, for unmeasured paths, the available measurements (e.g., y_1, y_2, and y_5 in (2)) can be used to estimate the unmeasured ones (e.g., y_3 and y_4). Moreover, when a path involves more than one traffic type and not all of them can be measured, we show that by measuring one traffic type, we can estimate the other's performance.

III. Partial Network Tomography

In this section, we define our partial network tomography approach where some link-level (or path-segement) metrics are inferred using the available end-to-end measurements. For the network shown in Fig. 1, consider the case where the routing matrix $\boldsymbol{A}$ is rank-deficient as in (2). In such case, it is not possible to uniquely identify all link-level metrics $\boldsymbol{x}$ in $\mathcal{G}$. However, subset of link-level and/or path-segment metrics can be identified using partial network tomography of a partial network $\mathcal{S}$.

Theorem 1. *To perform partial network tomography on a given network $\mathcal{G}$, the selected partial network $\mathcal{S}$ should include at least three edge nodes, $|\mathcal{E}|^{\mathcal{S}} \geq 3$.*

Proof. From ***Definition 1***, we know that each edge node i has at most one connected link ($d_i = 1$). And if $|\mathcal{E}|^{\mathcal{S}} < 3$, e.g., $|\mathcal{E}|^{\mathcal{S}} = 2$ there are at least two links in $\mathcal{S}$, i.e., $\gamma^{\mathcal{S}} \geq 2$. And because $\mathcal{G}$ is undirected network and the routing between any pair is deterministic, the maximum number of end-to-end measurements of partial network $\mathcal{S}$ with $|\mathcal{E}|^{\mathcal{S}} = 2$ is $\frac{|\mathcal{E}|^{\mathcal{S}} \times (|\mathcal{E}|^{\mathcal{S}} - 1)}{2} = 1$, ($k^{\mathcal{S}} = 1$). Therefore, $k^{\mathcal{S}} < \gamma^{\mathcal{S}}$ and hence the routing matrix $\boldsymbol{A}^{\mathcal{S}}$ is not full-rank with $rank(\boldsymbol{A}^{\mathcal{S}}) = 1$, so $\boldsymbol{x}^{\mathcal{S}}$ cannot have unique solution. □

Theorem 2. *To perform partial network tomography on a partial network $\mathcal{S} \subset \mathcal{G}$ with $|\mathcal{E}|^{\mathcal{S}} \geq 3$, the number of available end-to-end measurements should be greater than two, i.e., $k^{\mathcal{S}} > 2$.*

Proof. We know from **Theorem 1** that partial network tomography should be performed on partial network $\mathcal{S}$ with at least $|\mathcal{E}|^{\mathcal{S}} = 3$, hence there are at least $\gamma^{\mathcal{S}} \geq 3$ (see ***Definition 1***). Now suppose that $k^{\mathcal{S}} = 2$, then the routing matrix $\boldsymbol{A}^{\mathcal{S}}$ is $k^{\mathcal{S}} \times \gamma^{\mathcal{S}}$ with $k^{\mathcal{S}} < \gamma^{\mathcal{S}}$. Thus, $\boldsymbol{A}^{\mathcal{S}}$ is not full-rank and cannot be reduced to a full-rank matrix, consequently $\boldsymbol{x}^{\mathcal{S}}$ cannot be uniquely solved. □

Therefore, partial network $\mathcal{S}$ should include at least three edge nodes, i.e., $|\mathcal{E}|^{\mathcal{S}} \geq 3$ (see **Theorem 1**) and number of end-to-end measurements of more than two, $k^{\mathcal{S}} > 2$ (see **Theorem 2**). Equation (1) then can be rewritten as

$$\boldsymbol{y}^{\mathcal{S}} = \boldsymbol{A}^{\mathcal{S}} \boldsymbol{x}^{\mathcal{S}} \tag{3}$$

where $\boldsymbol{y}^{\mathcal{S}}$, $\boldsymbol{A}^{\mathcal{S}}$, and $\boldsymbol{x}^{\mathcal{S}}$ are end-to-end measurements vector, routing matrix, and link-level (or path-segment) measurements vector, respectively, for partial network $\mathcal{S}$.

The process of the partial network tomography is illustrated in Algorithm 1. The algorithm starts by finding any partial network $\mathcal{S}$ with $|\mathcal{E}|^{\mathcal{S}} \geq 3$ and $k^{\mathcal{S}} > 2$ (line 5). For such partial network $\mathcal{S}$, it checks for the rank of its routing matrix $rank(\boldsymbol{A}^{\mathcal{S}})$. If it is full-rank, it directly solves $\boldsymbol{x}^{\mathcal{S}}$ using (3) (line 15). Otherwise, if $\boldsymbol{A}^{\mathcal{S}}$ is dependent, it will be reduced into $\tilde{\boldsymbol{A}}^{\mathcal{S}}$ by removing redundant columns (line 7-8). It then solves $\boldsymbol{x}^{\mathcal{S}}$ using $\tilde{\boldsymbol{A}}^{\mathcal{S}}$ in (3). Otherwise, if the system (3) is inconsistent, it should find another partial network (line 11-13). To further illustrate Algorithm 1, consider a partial network $\mathcal{S}$ from the network shown in Fig. 1 with $\mathcal{E}^{\mathcal{S}} = \{0, 2, 4\}$. In such partial network, we have $\gamma^{\mathcal{S}} = 4$, therefore, in order to identify all link-level metrics, we need $k^{\mathcal{S}} = 4$ independent measurements. However, with $|\mathcal{E}|^{\mathcal{S}} = 3$, the maximum number of measurements is $k^{\mathcal{S}} = 3$ (see proof of **Theorem 1**). As a result not all link-level metrics can be uniquely identified, instead path-segments can be identified. The matrix for such partial network with $\mathcal{M} = \{p_1, p_2, p_4\}$ is

$$\boldsymbol{A}^{\mathcal{S}} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} \tag{4}$$

It is clear that (4) is dependent with one redundant column. The matrix then can be reduced to $\tilde{\boldsymbol{A}}^{\mathcal{S}}$ by removing the redundant column and assign the two links $(1,3), (3,4) \in \mathcal{L}$ to a path-segment $(1,4)$. Thus, with $\tilde{\boldsymbol{A}}^{\mathcal{S}}$ we can solve $\boldsymbol{x}^{\mathcal{S}}$ in (3).

IV. Delay Estimation with Neural Networks

To further complement the partial network tomography and infer full path-level performance of in-vehicle network, we introduce in this section our deep neural network (DNN) approach to estimate the performance of unmeasured paths given the available measurements.

Algorithm 1: Partial Network Tomography

Inputs : Network $\mathcal{G}$, end-to-end measurement vector $\boldsymbol{y}$ with y_i for each path $p_i \in \mathcal{M}$

Output: A set of inferred link-level and/or path-segment delays $\boldsymbol{x}^{\mathcal{S}}$

Infer Delay $(\boldsymbol{x}^{\mathcal{S}})$
- $\mathcal{M} \leftarrow$ all measured paths $p_i \in \mathcal{P}$;
- $\boldsymbol{x} \leftarrow \phi$;
- **while** $rank(\boldsymbol{A}) < \gamma$
  - find sub-network $\mathcal{S}$ s.t. $|\mathcal{E}|^{\mathcal{S}} \geq 3$ and $k^{\mathcal{S}} > 2$;
  - **if** $rank(\boldsymbol{A}^{\mathcal{S}}) < \gamma^{\mathcal{S}}$ **then**
    - **if** *system is dependent* **then**
      - reduce $\boldsymbol{A}^{\mathcal{S}}$ to $\tilde{\boldsymbol{A}}^{\mathcal{S}}$ by removing redundant column/s;
      - solve $\boldsymbol{x}^{\mathcal{S}}$ in (3) using $\tilde{\boldsymbol{A}}^{\mathcal{S}}$;
      - **return** $\boldsymbol{x}^{\mathcal{S}}$;
    - **else**
      - system is inconsistent;
      - **go to** *5*;
  - **else**
    - solve $\boldsymbol{x}^{\mathcal{S}}$ in (3);
    - **return** $\boldsymbol{x}^{\mathcal{S}}$;
  - **for each** $x_i^{\mathcal{S}} \in \boldsymbol{x}^{\mathcal{S}}$ **do**
    - **if** $x_i^{\mathcal{S}}$ *is a link-level metric for* $(u, v) \in \mathcal{L}$ **then**
      - $x_i^{\mathcal{S}} \cup \boldsymbol{x}$

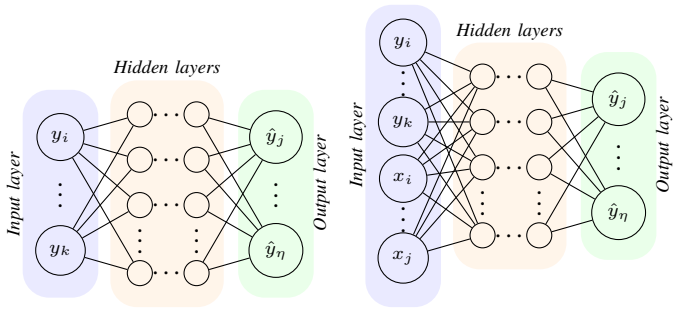
A. Neural Network Delay Estimation (NNDE)

As a benchmark solution, when there is only end-to-end available measurements, the DNN takes these measurements as input to estimate the performance of umeasured paths. The DNN structure in this case is shown in Fig. 2a. Particularly, we feed the neural network with the available end-to-end delay measurements $\boldsymbol{y}$ for paths in $\mathcal{M}$ to estimate the performance of unmeasured paths in $\mathcal{P}\backslash\mathcal{M}$. For example, for the network in Fig. 1, suppose that $\mathcal{M} = \{p_1, p_2, p_5\}$ hence, the input to the neural network will be a vector $\boldsymbol{y} = (y_1, y_2, y_5)^{\mathsf{T}}$. The neural network then outputs estimations for unmeasured paths in $\mathcal{P}\backslash\mathcal{M}$, in this case the output will be $\hat{y}_3$ and $\hat{y}_4$.

B. Neural Network Delay Tomography (NNDT)

As we know from Section III, network tomography can help infer proportional of link-level performance. Such inferred metrics can be further combined with NNDE to improve the estimation accuracy of unmeasured paths. Therefore, the input layer of DNN in this case includes $\boldsymbol{y}$ for paths in $\mathcal{M}$ and $\boldsymbol{x}^{\mathcal{S}}$ as shown in Fig. 2b. As in NNDE, the network then estimates the values for unmeasured paths in $\mathcal{P}\backslash\mathcal{M}$.

DNN operates in two basic processes: feed-forward and backpropagation. In feed-forward, the network starts by taking the input, multiplying it by a matrix of random weights and the result is then passed to the first hidden layer in which an activation function is applied. The output of the first hidden layer is then passed to the next layer where the same operations are repeated until the output layer that produces estimated values $\hat{\boldsymbol{y}}$ for unmeasured paths in $\mathcal{P}\backslash\mathcal{M}$. The estimated values



(a) NNDE (b) NNDT

Fig. 2: Deep neural network structure for NNDE and NNDT.

are then compared against the actual values using a cost function. In this work, we use Mean-Squared Error (MSE) as our cost function. It is defined by

$$C = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{5}$$

where N is the total number of datapoints in a batch, y_i and $\hat{y}_i$ are the actual and estimated end-to-end delays, respectively, for paths in $\mathcal{P}\backslash\mathcal{M}$. The goal is to minimise (5) so that the estimated value is close to the actual value. This is achieved with backpropagation where the error is backpropagated throughout the hidden layers using gradient descent and adjust the weights accordingly to minimise (5) and hence, improve the estimation accuracy.

V. Performance Evaluation

In this section, we first evaluate the accuracy of our NNDE approach in estimating the delay of unmeasured paths in $\mathcal{P}\backslash\mathcal{M}$. Then, we show the results when using the partial network tomography with neural network in NNDT. We evaluate our approaches in an in-vehicle network with different traffic types using TSN [11] standards as well as non-TSN traffic.

A. Experiment Setup

1) Network Simulation

To evaluate the performance of our approach, we conducted simulations using OMNeT++ and CoRE4INET [12] in an in-vehicle network topology with Ethernet backbone as shown in Fig.3. We ran the simulation for 5000 times. Switches processing delays are uniformly distributed between the range of 10 μs and 80 μs. The traffic we used are both Ethernet and

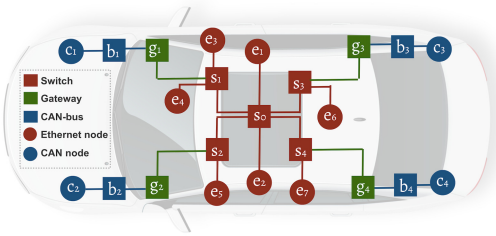


Fig. 3: Ethernet-backbone in-vehicle network topology.

CAN (Controller Area Network) traffics. For the Ethernet, we

used different traffics based on TSN standards [11] including AVB, IEEE802.1Q, as well as best effort traffic. For non-TSN, we used best effort traffic for all paths. In our network shown in Fig. 3, we have $|\mathcal{E}| = 11$ edge nodes, $\gamma = 23$ links, and $z = 10$ paths with $\mathcal{P} = \{p_1, p_2, \ldots, p_{10}\}$ as illustrated in TABLE I, where node pairs (c_1, c_3), (c_2, c_4), (e_3, e_2), (e_5, e_2), (e_2, e_1), (e_4, e_7), (e_6, e_7), (e_3, e_5), (e_3, e_6) and (e_5, e_6) communicate respectively via paths $p_1, \ldots, p_{10}$. For TSN, the type and payload of traffic used in each path is shown in TABLE I. Note that path p_7 has two different traffics, to distinguish between the two types we denote the path with IEEE802.1 traffic by ${}^1p_7$ while the path with best effort traffic by ${}^2p_7$. For each traffic, we recorded its end-to-end delay to construct the dataset we use for training our model.

Path	**Traffic type**	**Payload (B)**
$p_1 = \{(c_1, b_1), (b_1, g_1), (g_1, s_1), (s_0, s_1), (s_0, s_3), (s_3, g_3), (g_3, b_3), (b_3, c_3)\}$	CAN/ Best effort	8
$p_2 = \{(c_2, b_2), (b_2, g_2), (g_2, s_2), (s_0, s_2), (s_0, s_4), (s_4, g_4), (g_4, b_4), (b_4, c_4)\}$	CAN/ Best effort	8
$p_3 = \{(e_3, s_1), (s_0, s_1), (s_0, e_2)\}$	AVB-A	393
$p_4 = \{(e_5, s_2), (s_0, s_2), (s_0, e_2)\}$	AVB-A	393
$p_5 = \{(e_2, s_0), (s_0, e_1)\}$	AVB-B	786
$p_6 = \{(e_4, s_1), (s_0, s_1), (s_0, s_4), (s_4, e_7)\}$	IEEE802.1Q	100
$p_7 = \{(e_6, s_3), (s_0, s_3), (s_0, s_4), (s_4, e_7)\}$	IEEE802.1Q Best effort	100 46
$p_8 = \{(e_3, s_1), (s_0, s_1), (s_0, s_2), (s_2, e_5)\}$	Best effort	46
$p_9 = \{(e_3, s_1), (s_0, s_1), (s_0, s_3), (s_3, e_6)\}$	Best effort	46
$p_{10} = \{(e_5, s_2), (s_0, s_2), (s_0, s_3), (s_3, e_6)\}$	Best effort	46

TABLE I: Paths and traffic types used in simulation with TSN traffics.

2) Neural Network Model and Data Processing

We used DNN as shown in Fig. 2. The structure of our DNN consists of two hidden layers. The number of hidden neurons in each layer is $2k - 1$ in case of NNDE and $2(k + |\boldsymbol{x}|^{\mathcal{S}}) - 1$ in NNDT. The model is trained over 1000 epochs with early stopping [13] to avoid overfitting. We used the dataset of 5000 samples generated by our simulation, then split it into 60%, 25% and 15% as training, validation and testing sets, respectively. Moreover, MinMaxScalar is applied to scale all the data values between 0 and 1. We used the training data to train the model and the validation data with early stopping for cross-validation. The test set is used to test the model performance on new data. Moreover, we used ReLU (Rectified Linear Unit) activation function for all hidden layers and linear function for the output layer. For backpropagation, we used *Adam* optimisation function [14] to minimise the cost function (5). Our model is trained using mini batches of size 40.

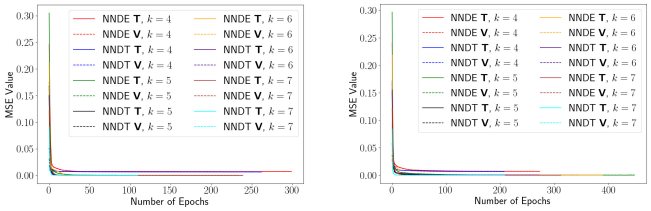
B. Partial Network Tomography

To perform partial network tomography for the network shown in Fig. 3, we used partial network $\mathcal{S}$ with $\mathcal{E}^{\mathcal{S}} = \{e_3, e_5, e_6, e_7\}$ and $\mathcal{M}^{\mathcal{S}} = \{{}^2p_7, p_8, p_9, p_{10}\}$ which satisfies the conditions in Algorithm 1, line 5. Using (3), the path-segments' metrics $\boldsymbol{x}^{\mathcal{S}}$ for $(e_3, s_0), (e_5, s_0), (e_6, s_0), (e_7, s_0)$ are inferred. The inferred metrics are then added to the input of neural network in NNDT case as shown in Fig. 2b.

C. Results

1) Training

We first evaluate our model's learning performance to ensure that it is not overfitting, this is shown in Fig. 4 where we used cross-validation. The results show that in both NNDE and NNDT, training and validation are almost aligned and the model stops training before it starts to overfit with maximum number of epochs equal to 448 in case of NNDE for non-TSN when $k = 5$ (Fig. 4b) and 300 in case of NNDE for TSN when $k = 4$ (Fig. 4a). We can see that when the number of measured paths i.e., k increases, the model with TSN learns faster as shown in Fig. 4a. In addition, in all different values of k, the number of epochs in NNDE is larger than in NNDT. This means that the more information added from the partial network tomography in NNDT allowed the model to learn quicker than when the only available information is the path-level measurements as in NNDE.



(a) TSN (b) non-TSN

Fig. 4: Training performance with cross-validation. **T**: training, **V**: validation. The y-axis represents the MSE (used as cost function for training).

2) Testing

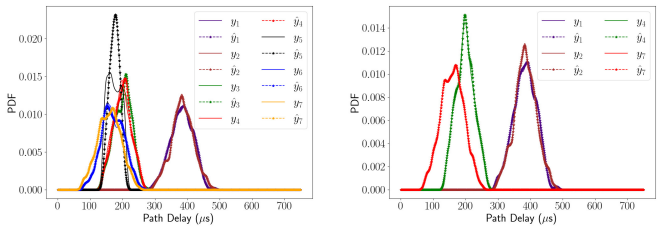
After training our model, we evaluate its performance on new datapoints that the model has never seen before using the test set. In TABLE II, we show the MAPE (Mean Absolute Percentage Error) values for model estimations on the test set. In our scenario, partial network tomography is performed when 40% or more of the network is measured with $\mathcal{M} = \{{}^2p_7, p_8, p_9, p_{10}\}$. As seen, for both TSN and non-TSN, the error rates of both models, NNDE and NNDT, decrease when the number of available measurements increase reaching up to 99% accuracy when 50% or more is measured. Moreover, both NNDE and NNDT almost have similar performance in estimating the unmeasured paths, sometimes with slight improvement when using NNDT.

Furthermore, in Fig. 5, we show the distribution of the estimated paths' delays in NNDT when 40% of the network is measured (Fig. 5a) and when 70% is measured (Fig. 5b) with TSN. We can see in Fig. 5a that the estimated and actual delays are very closed to each other except for y_5 and $\hat{y}_5$ where there is marginal difference. This is because the available measurements are for paths in $\mathcal{M}$ that do not share any common link with path $p_5 \in \mathcal{P} \backslash \mathcal{M}$, as compared

%	Measured paths ($\mathcal{M}$)	with TSN		non-TSN	
		NNDE	NNDT	NNDE	NNDT
10	${}^1p_7$	10.009	N/A	10.244	N/A
20	$p_8, {}^2p_7$	8.225	N/A	8.881	N/A
30	${}^1p_7, p_8, p_{10}$	4.854	N/A	5.292	N/A
40	${}^2p_7, p_8, p_9, p_{10}$	2.522	2.395	2.806	1.886
50	$p_5, {}^2p_7, p_8, p_9, p_{10}$	1.176	1.162	0.934	0.492
60	$p_5, p_6, {}^2p_7, p_8, p_9, p_{10}$	0.780	0.819	0.389	0.562
70	$p_3, p_5, p_6, {}^2p_7, p_8, p_9, p_{10}$	0.665	0.606	0.143	0.155

TABLE II: MAPE values on test set. The first column indicates the measured subset in percentage.

with other paths where they share one or more links with at least one path in $\mathcal{M}$. On the other hand, when more paths are measured, as shown in Fig. 5b the distribution of actual and estimated values are nearly overlapped. Note that in both cases, $\hat{y}_7$ is accurately estimated for ${}^1p_7$ due to the available measurement of the same path but different traffic type (i.e., ${}^2p_7$ for best effort traffic).



(a) NNDT in case of 40% (b) NNDT in case of 70%

Fig. 5: Probability density function (PDF) for the distribution of actual and estimated paths' delays for in-vehicle network with TSN.

To further evaluate the contribution of each path's measurement in estimating the overall network delay performance, we show in Fig 6 the MAPE value for each measured path in $\mathcal{M}$ when only 10% of the network is measured, i.e., $k = 1$. As shown, when $\mathcal{M} = \{p_7\}$, the MAPE is at its lowest value for both TSN and non-TSN. In contrast, the highest MAPE value for TSN and non-TSN achieved when $\mathcal{M} = \{p_5\}$ and $\mathcal{M} = \{p_4\}$, respectively. This is because, e.g, p_5 has only one link that is shared with two paths only in the network (i.e., link (s_0, e_2) is shared with p_3 and p_4). While p_7 has two path-segments ((s_0, e_6) and (s_0, e_7)) shared with p_6, p_9 and p_{10}, in addition to two more links ((s_0, s_3) and (s_0, s_4)) that are shared with p_1 and p_2.

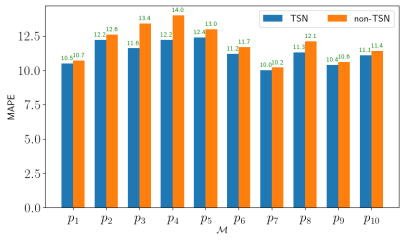


Fig. 6: MAPE values on test set for NNDE when only 10% of the network is measured (i.e., $k = 1$).

VI. Conclusion

In this paper, we have proposed a partial network tomography approach to infer link-level and/or path-segment metrics of partial network using only the available end-to-end measurements. Moreover, we have proposed a deep neural network delay estimation approach to estimate the unmeasured end-to-end delay of in-vehicle network. In our approach, we have used a subset of measured paths as input to train the model and estimate the delay of remaining paths in the network. Moreover, we have used the inferred metrics from the partial tomography to estimate the performance of unmeasured paths. The results shows that with only measuring end-to-end subset of the network, the overall performance can be accurately estimated with up to 99% accuracy. We believe that our approach is suitable for networks with variable traffics such as in-vehicle networks where direct monitoring of full network is not desirable as such monitoring traffic may overwhelm the network. In our future work, we will further extend our approach to detect anomalies in such deterministic Ethernet-based in-vehicle networks.

References

[1] W. Wu, R. Li, G. Xie, J. An, Y. Bai, J. Zhou, and K. Li, "A survey of intrusion detection for in-vehicle networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 919–933, 2019.

[2] Y. Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," *Journal of the American statistical association*, vol. 91, no. 433, pp. 365–377, 1996.

[3] R. Cáceres, N. G. Duffield, J. Horowitz, and D. F. Towsley, "Multicast-based inference of network-internal loss characteristics," *IEEE Transactions on Information theory*, vol. 45, no. 7, pp. 2462–2480, 1999.

[4] N. G. Duffield and F. L. Presti, "Multicast inference of packet delay variance at interior network links," in *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No. 00CH37064)*, vol. 3, pp. 1351–1360, IEEE, 2000.

[5] Y. Tsang, A. Tsang, M. Coates, and R. Nowak, "Passive unicast network tomography based on tcp monitoring," in *Rice University, ECE Department*, Citeseer, 2000.

[6] Y. Tsang, M. Coates, and R. Nowak, "Passive network tomography using em algorithms," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221)*, vol. 3, pp. 1469–1472, IEEE, 2001.

[7] V. N. Padmanabhan, L. Qiu, and H. J. Wang, "Passive network tomography using bayesian inference," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pp. 93–94, 2002.

[8] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Inferring link metrics from end-to-end path measurements: Identifiability and monitor placement," *IEEE/ACM Transactions on Networking*, vol. 22, no. 4, pp. 1351–1368, 2014.

[9] L. Ma, Z. Zhang, and M. Srivatsa, "Neural network tomography," *arXiv preprint arXiv:2001.02942*, 2020.

[10] Z. Yana, R. Gu, T. Dong, J. Yin, S. Li, Z. Liu, T. Zhang, and Y. Ji, "An artificial neural network based attenuation tomography in free space optical network," in *2018 International Conference on Networking and Network Applications (NaNA)*, pp. 52–57, IEEE, 2018.

[11] "IEEE 802.1 Time-Sensitive Networking Task Group." https://1.ieee802.org/tsn/. [Online].

[12] T. Steinbach, P. Meyer, S. Buschmann, and F. Korf, "Extending omnet++ towards a platform for the design of future in-vehicle network architectures," *arXiv preprint arXiv:1609.05179*, 2016.

[13] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.