Machine Learning Network Tomography with partial topology knowledge and dynamic routing

Ippokratis Sartzetakis, Emmanouel (Manos) Varvarigos
National Technical University of Athens, Athens, Greece, Institute of Communication and Computer Systems, Athens, Greece
isartz@mail.ntua.gr, vmanos@mail.ntua.gr

***Abstract*—Networks are always progressing to support the evolving and diverse applications and the needs for improved capacity, latency and security. To this end, monitoring is key to ensuring the uninterrupted network operation and the QoS of the applications. Network Tomography uses a subset of monitoring information (corresponding to partial view of the network state) to estimate wide-sense network performance, including unmonitored parameters. In this paper, we present a novel Machine Learning (ML) formulation for Network Tomography. The proposed formulation accounts for realistic scenarios where: i) the existence of certain links of the network is not known (e.g., due to security reasons), ii) the routing is dynamic (non-deterministic), i.e., for the same origin-destination node pair, a different route may be selected depending on the state of certain links. Our simulations indicate that our proposal has better estimation accuracy compared to traditional algebraic or other ML approaches that cannot or do not take into account these two assumptions.**

Keywords*—*dynamic routing, machine learning, network tomography, partial topology knowledge

I. INTRODUCTION

Cloud and edge computing infrastructures play an essential part in today's economies. They offer a whole range of processing services, for example, online commerce, smartphone applications, video streaming, gaming, etc. At the same time their heavy use and rapid deployment makes them increasingly complex, heterogeneous and difficult to manage. The level of performance of the edge-cloud continuum determines the efficiency of the whole Information and Communication Technology (ICT) infrastructure, and the Quality of Service (QoS) perceived by the deployed applications. The heterogeneity of the networks coupled with the increased traffic volumes and rates, make their real time (dynamic) management both challenging and necessary. Multiple factors can affect network performance, such as: packet loss, abnormal delay, delay variance, bad load distribution and poor behavior of network operating systems or user applications. These factors can result in network soft (network performance degradation) or hard (network downtime) failures, causing significant costs. Therefore, it is critical to provide advanced network monitoring tools able to reflect changes in the network state and analyze them.

As is the case with any system, a network has to be observable in order to be manageable and stable. Network tomography (NT) [1] is an indispensable tool for this purpose. NT refers to large-scale network inference. It involves estimating network (path) performance parameters based on traffic measurements of a limited subset of network nodes and links. Indicative monitoring data set includes (but is not limited to): throughput, delay, jitter, packet delivery ratio, congestion, bandwidth for specific paths. Some of these metrics are additive per link (e.g., delay, jitter), or can be transformed to an additive form (e.g., use the log function in the case of the packet delivery ratio or reliability), or they are non-additive (e.g., the congestion level or the bandwidth, depends on the worst link of a path, involving a min operator).

NT reduces the monitoring needs for the whole network. Thus, it increases efficiency, reduces equipment and operating costs and can help verify service level agreements. The more accurate is the knowledge of the network state obtained through NT, the better positioned is the Orchestration layer in maximizing the resource utilization (including network, processing resources) in mixed cloud/edge environments. For example, in real time communication it is important to estimate in advance the quality of a connection before actually establishing it over a specific path [2]. NT can provide the means for this purpose, as an unestablished connection cannot really be monitored. Moreover, certain security events and anomalies can be promptly detected and dealt with, before they significantly affect the operation of the network.

Monitoring can be passive or active. In active monitoring, certain probes (that create extra traffic) are purposefully deployed to measure the required network segments. In passive monitoring, (part of) the existing traffic is monitored. In all cases, the need to keep the monitoring cost low implies that the number of deployed probes or of existing measured connections, is kept as small as possible. A major point of concern is the complexity of the monitoring solution. In large scale networks, vast amounts of monitoring data are generated. This makes the correlation of the data a very difficult task. Therefore, efficient algorithms need to be developed to infer the appropriate metrics. Another point of concern is that in complex networks a path may cross different and heterogeneous subnetworks. Under these circumstances, the complete network topology is usually not completely known by a single actor as it is not fully advertised. So, a (monitored) path may cross different networks and contain an unknown number of links. This makes difficult or even impossible the use of traditional NT algebraic methods. These methods assume complete knowledge of the network topology where the links are typically represented in a matrix. Moreover, in modern networks, the routing between an origin and a destination node may be dynamic (non-deterministic). It may change based on the network congestion. This should also be taken into account when designing a NT algorithm, to provide accurate estimations.

In this paper we develop a novel ML formulation for NT. The features of the ML algorithm are designed to take into account the peculiarities of modern networks where the topology may or may not be completely known and the

This work was partially supported by the Horizon 2020 SERRANO project (grant agreement: 101017168)

978-1-6654-3540-6/22/$31.00 ©2022 IEEE

routing may be static or dynamic. Moreover, the features of the ML formulation can provide the required information to estimate both additive metrics, and non-linear metrics. We demonstrate through simulation experiments that the accuracy of our proposal is excellent in a variety of scenarios. In many cases it is far better than that of other ML or pure algebraic approaches. The improved accuracy can give the network operator a better view of the network conditions and enable better optimization decisions. Thus, the overall network efficiency and the application QoS can be improved.

The rest of the paper is organized as follows. Section II presents the related previous NT work. Section III describes the network scenario assumed. Then the proposed ML formulation is introduced. Section IV presents the simulation experiments the evaluation. Section V concludes the paper.

II. Related Work

The term Network Tomography was first mentioned in [1]. The initial problem statement was to estimate node-to-node network traffic intensity from link measurements. As the subsequent research is vast and ongoing, we will provide a short summary. More information can be found in the references of the mentioned work. Note that when we refer to delay in this paper, we assume that it comprises of propagation delay and queueing delay. The path delay is the sum of the delays of the contained links. In [3], the authors used unicast end-to-end traffic measurements and developed techniques to estimate link delay distribution. Passive network tomography was first introduced in [4]. The authors assumed the measurement of end-to-end performance metrics of existing traffic, and researched the problem of identifying lossy links. Reference [5] focused on characterizing end-to-end additive metrics. The authors find a minimal set of k linearly independent paths that can describe the metrics of all the other paths. The authors in [6] used multiple and simple one-way measurements among pairs of nodes. Then they estimated the one-way delay between network nodes. In doing so they used a global objective function that is affected by the network topology and not just by individual measurements. Reference [7] focused on the problem of identifying link level metrics from end-to-end metrics of selected paths. The authors developed a low-complexity algorithm to construct linearly independent, cycle-free paths between monitors without examining all candidate paths. The authors of [8] assumed a similar problem statement. They investigated the conditions under which the link level metrics could be acquired, depending on the network topology and the number and location of the monitors. In [9] various variations are surveyed, such as link delay inference through multicast end-to-end measurements, origin-destination matrix inference and topology identification. Survey [10] describes subsequent developments in NT. It also presents network coding and compressed sensing to improve estimation accuracy, computational complexity and probing and operational cost.

Another topic is that of failure detection using network tomography. In this scenario, network tomography pertains to identifying whether a network node has failed given binary end-to-end path metrics. In [11] the authors researched the conditions that need to be satisfied in order to identify a bounded number of node failures. They also quantified the maximum number of identifiable node failures and the largest node set within which failures can be localized for a given number of failures. In [12] the authors provided upper bounds on the maximum number of identifiable failed nodes, considering a certain number of monitored paths, constraints on the network topology, the routing scheme, and the maximum path length.

Recently, ML has been applied to the field of NT. The authors of [13] proposed a NT approach for non-deterministic routing where the measured flows for a given origin-destination node pair may cross different paths. In [14] a Neural Network is trained to infer additive metrics in an SDN/NFV environment. The authors in [15] also propose the use of neural network to infer metrics based only on the origin and destination node pair, and also to reconstruct the network topology. In [16], assuming in-vehicle network monitoring, neural networks are again applied to estimate the performance of an unmonitored part of a network. Finally, in [17] the same principles are applied to the domain of network slicing.

In this paper we consider that the network topology information may be incomplete as in [15], and at the same time, the routing may be dynamic (non-deterministic) as in [13]. We also assume that the specific paths that are monitored are a given, and that we want to make the most of the available information. We select a set of ML features that can be used to improve the accuracy of the performance metrics estimations under these assumptions. Our contributions are: 1) We present a network tomography approach based on ML, which accounts for incomplete knowledge of the underlying network topology and for dynamic routing, 2) Using end-to-end measurements we infer link-level (both additive and certain non-additive) metrics for known links. If a path crosses unknown segments of the topology, we infer metrics for the subset containing the unknown links 3) Using the above knowledge, we estimate performance metrics for unestablished or unmonitored paths, even with both incomplete topology knowledge (i.e., the exact routing is not known) and dynamic routing.

Our proposal is different to [15] in that we employ additional ML features to account for unknown parts of the network. The features are particularly useful in cases as in [13], where the routing is dynamic. This provides better accuracy for certain scenarios as we demonstrate in the simulations. To the best of our knowledge, there is no previous NT formulation that considers both dynamic (non-deterministic) routing and partial topology knowledge.

III. Network Tomography

In this section we will first describe the network scenario and the notation. Next, we will provide details on the algebraic methods that can be used to solve a basic NT problem. Afterwards, we will describe our proposed ML formulation.

A. Network Notation

We consider a network $N = (V, L)$ where V denotes the set of nodes and L the set of known directed links (thus, both (i, j) and (j, i) are present in L if nodes i and j are connected through a unidirectional link). We assume a set P of paths already established in the network. The routing matrix of the established paths is defined as the binary matrix $G_P \in \{0,1\}^{|P|\times|L|}$, where $G_P[p, l]$=1 when path p contains link l, and is 0, otherwise. Consider the end-to-end vector of parameters $\mathrm{y}_P \in \mathbb{R}^{|P|}$. Vector y_P can represent different performance parameters (e.g., delay, jitter, etc.). If the link-level vector parameters $\mathrm{x} \in \mathbb{R}^{|L|}$ are additive per link, vector y_P can be written as a linear combination of link-level vector parameters x, so that $\mathrm{y}_P = G_P\mathrm{x}$. We assume that we want to

estimate the end-to-end parameters of a set M of paths (either new or unmonitored ones), denoted by vector $y_M \in \mathbb{R}^M$, given that we know their routing $G_M \in \{0,1\}^{|M|\times|L|}$. Then:

$$y = \begin{bmatrix} y_P \\ y_M \end{bmatrix} = \begin{bmatrix} G_P \\ G_M \end{bmatrix} x \quad (1)$$

Consider, for example, the network of Fig. 1, where a set of $P = \{p_1, p_2\}$ paths are already established and correspond to submatrix G_P and the known end-to-end QoS values $y_P = \{y_1, y_2\}$. The values in vector y_P can be different for different applications and use cases. The path to be established is denoted by $M = \{m_3\}$ whose end-to-end value y_3 we want to estimate. We assume that all the (three) links the paths use are known. The routing can be described as:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad (2)$$

Since the new path m_3 contains links that are already in use by other paths, it is possible to estimate its end-to-end value. This can be achieved using the Moore-Penrose inverse $(\,.\,)^+$ of G_P (complexity $O(|P|^3)$), that is:

$$\hat{y_M} = G_M G_P^+ y_P \quad (3)$$

B. ML network tomography

The above algebraic formulation is normally used when the network topology is completely known. It is not expected to work well when certain links are not known. As we have already mentioned, this is the case in many of the modern networks. In this section we present a ML formulation that takes this into account. The features are chosen to summarize in a heuristic way the important characteristics of the paths with respect to the estimation problem. We organize the feature matrix Q so that each row corresponds to a path $p \in P$, while the columns represent (link or node, feature) pairs, for the links or nodes of the path and the values of the chosen features. In particular, we choose two kinds of features. The first set of features consists of all the known links of the topology (we define it as a set $|L|$). The second set of features consists of all the path nodes that can be origin or destination. More specifically, we define S as a $|P| \times |L|$ link-level feature matrix designed to take into account the *known* links that a path contains. Element S_{pl}, corresponds to path p and a known link l, and is set equal to 1 if path p contains link l, and equal to zero, otherwise. This feature is the equivalent of the routing matrix G_P. We also define the node-level feature matrix A to represent information regarding the origin and destination node of a path. In particular, A is a $|P| \times |V|$ node-level feature matrix. Element A_{pv} is equal to 1 if path p starts or ends at node v, and is set to zero, otherwise. Note that this formulation does not specifically take into account which node is origin and which one is destination. However, the information is indirectly captured by the ML algorithm as the set L contains both directions of a link (directed links). We also consider an additional feature to represent the ML bias term (denoted by BT). The bias term can account for monitoring errors or noise that cannot be reduced by any other means. We concatenate all the link-level feature matrices into one feature matrix defined as:

$$Q = [BT \quad S \quad A]_{|P|x(1+|L|+|V|)} \quad (4)$$

Similar to Eq. (1), Q_P corresponds to the set of features related to the end-to-end observed QoS values y_P. The features are designed to capture as much information as possible to evaluate the metric at hand. The absence of knowledge of certain links through which the path passes (*unknown* links) is counterbalanced by the knowledge of the origin and destination node of a path. Moreover, this formulation is better than assuming only the origin and destination node of a path as features. The knowledge of even a small subset of the links that a path contains can greatly increase the accuracy of the estimation. After the features have been defined, an appropriate ML algorithm can be used to solve the problem at hand. In this paper we evaluated various architectures of neural networks towards this end. Neural networks have the potential to represent complex functions and thus estimate both additive and certain non-additive metrics. The assessment of other algorithms was left a topic of future work.

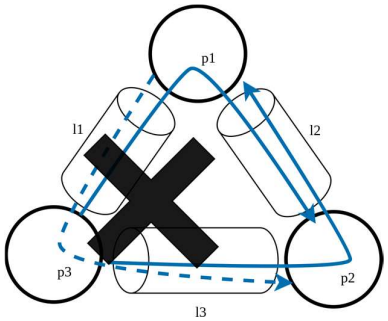

Fig. 1. A network with 2 monitored paths and one candidate (or unmonitored), sharing one known link and one origin node. Parts of the topology (crossed) may be unknown.

C. Neural Network Architectures

A neural network (NN) is a set of connected functions that interpret a set of inputs into a desired kind of output. A NN can be trained to learn a function f through which a set of features explain a respective set of observations. In our case:

$$y_P = f(Q_P) \quad (5)$$

A fully connected NN consists of an input layer, one (or more) hidden layer(s), and an output layer. Each node of one layer is fully connected to all the nodes of the next layer. This enables data propagation from one layer to another. Each layer multiplies the input by a weight matrix and adds a bias term. The hidden layers can be wider (have more nodes) than the input and output. In these cases, the neural network may be able to learn more complex relationships between the features and thus have better accuracy. A trial-and-error approach is usually required to find the most suitable architecture. In any case the specific problem at hand is not very complicated. Therefore, a relatively shallow NN architecture is expected to have good estimation accuracy. Note that the training of a neural network can be time consuming. The model is expected to perform well without the need for retraining, unless the network state changes significantly. The inference is much faster as we will show in the simulation section. In the next section we present the evaluation of different architectures and compare our proposed tomography framework to alternatives.

IV. Results

To evaluate our proposed formulation, we performed a number of simulation experiments. We assumed the Deutsche Telecom (DT) topology of Fig. 2 with 12 nodes and 20 unidirectional links. In the figure, the length of each link in km is also depicted. We considered that the required estimations metrics are: i) the delay, which is additive per link, and ii) the bandwidth, which is non-additive and depends on the worst link of the path (min operation over the links of the path). We

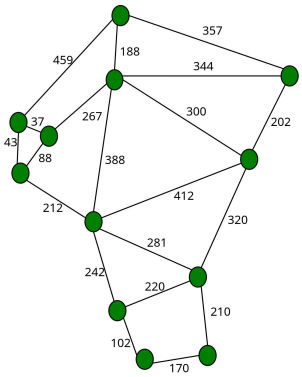

Fig. 2. The DT topology with the link lengths in Km

set the delay of each link to be numerically equal to its length. So, the delay of a path is equal to the sum of the delays of its links. Without loss of generality, we set the bandwidth of each link to be numerically equal to its length. The bandwidth of a path equals to the minimum bandwidth of the links that it contains. We assumed various different loads of 100, 200, 300, 400, 500 connections with uniformly chosen source-destination nodes. For each source-destination pair we used a k-shortest path algorithm with $k = 1,2,3$ to decide the routing. When $k > 1$ the specific route for each source-destination pair was chosen uniformly over these k paths. We also assumed an increasing number of unknown links, where, for a given link that is considered unknown, we removed its related value from the routing matrix G, and from the feature matrix S. Simulations were performed in MATLAB with a quad core GPU@3GHZ. The training of the NN was used 2000 epochs, chosen by testing the performance of other options. The ReLu activation function and the Mean Square Error (MSE) loss function were employed. We used 90% of the established paths for training, and 10% for testing. We exclude from the testing set paths containing a link that is not used in the training set. We run 200 independent iterations and averaged the results.

A. Additive metrics evaluation

First, we evaluated several different NN architectures that employed the features described in section III.B. In Fig. 3 we present the estimation accuracy of three architectures that achieved the best results for the estimation of the delay. We plot the accuracy in terms of Mean Absolute Error (MAE) as a function of the number of established paths in the network. We employed a trial-and-error approach to find the most suitable architectures that exhibit good accuracy. We omit the depiction of detailed training performance indicators due to space limitations. The proposed NN had three fully connected layers. The number of outputs of the first, second and third layer was four, two and one time(s) the number of features (32), respectively. The alternative NN1 also had three fully connected layers. The number of outputs of the first and second layer was equal to the number of features. The outputs of the third layer were equal to half the number of features. The alternative NN2 had two fully connected layers. The number of outputs of both layers was equal to the number of features. In Fig. 3a, $k = 1$, so that each source-destination pair can only be served by one path (static routing). The accuracy of the three architectures is very similar and increases as the number of paths increases, since the NNs are better trained, with more data. As k increases (Figs. 3b and 3c), a source-destination pair may be routed over different paths (non-deterministic/dynamic routing). This means that the neural network should correlate a larger amount of information to provide an accurate output (i.e., the origin and destination pair do not contain the required information for accurate estimation). The proposed NN has marginally better accuracy. It seems that the additional outputs of the layers allow the neural network to learn better the relationships between the features, at least for smaller number of established paths. We also compared the three architectures for the case of the bandwidth estimation and the results were similar.

We then compared the accuracy of four algorithms: i) our proposed NN formulation, ii) a NN formulation where the features are only the links of the paths, iii) a NN formulation where the features are only the origin and destination nodes of the paths, and iv) a traditional algebraic matrix inversion solution. Fig. 4 presents the MAE of the four examined algorithms for different number of measured paths. In Fig. 4a, where $k = 1$, we notice that a simple matrix inversion can achieve good accuracy even with a relatively small number of measured paths. The algorithms that are based on NN require a larger number of established paths to be trained so as to have good accuracy. The NN that uses only the links as features, achieves slightly better accuracy earlier than the proposed NN (in this scenario). The reason is that the proposed NN has more features thus requiring a larger training set. The NN that only uses the origin and destination node requires the largest training set. The reason is that the NN needs to be trained with all possible combinations of origin and destination nodes in order to be able to provide an accurate estimate. For 500 established paths, all algorithms have MAE less than 10^{-4}, the maximum error is less than 10^{-3} and the 95% confidence interval is approximately $\pm 10^{-5}$. In Figs. 4b and 4c, where we have $k > 1$, the simple matrix inversion method still achieves the best accuracy in all cases (the MAE and the maximum error are similar to the case where $k = 1$). The NN that relies only on the origin and destination node does not have enough

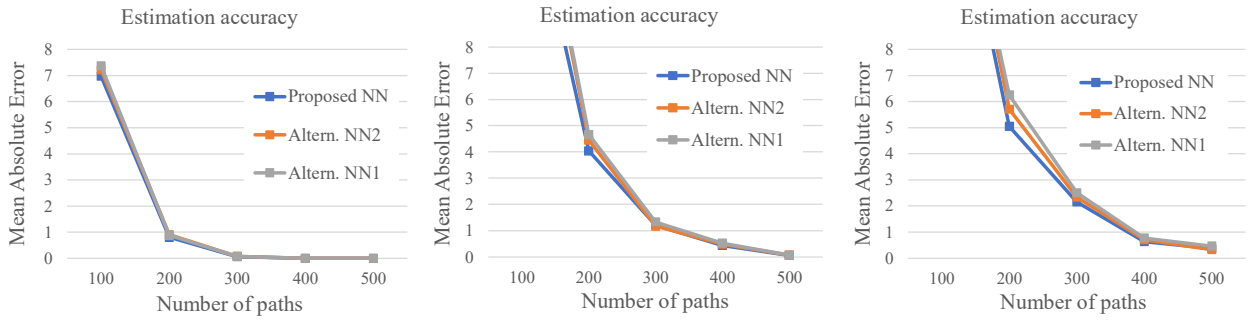

Fig. 3. Accuracy (delay metric) of three NN architectures for different number of paths and a) $k = 1$, b) $k = 2$, c) $k = 3$

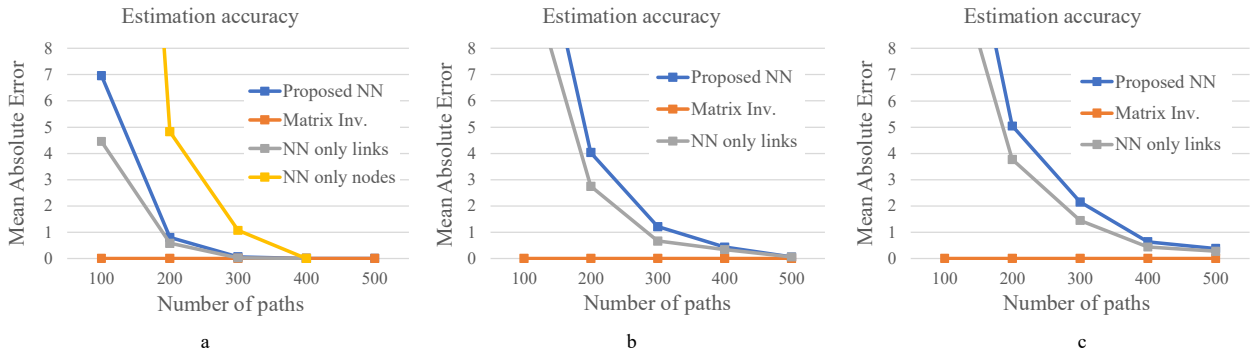

Fig. 4. Accuracy (delay metric) of the algorithms for different number of paths and a) $k = 1$, b) $k = 2$, c) $k = 3$

information to provide an accurate estimate. Its accuracy is not depicted in these figures, but is examined in Fig. 5. The other NNs require larger number of established paths to achieve comparable accuracy, since the non-deterministic routing requires more training data in order for the NN to understand it. In the case of 500 paths, the MAE of both algorithms are approximately the same, and it is in the order of $5*10^{-2}$ for k=2 and $3*10^{-1}$ for $k = 3$. The maximum error is approximately 150 and 220 delay units and the 95% confidence interval is approximately ± 0.5 and ± 0.8 for $k = 2$ and $k = 3$ respectively. For 500 paths the NN required approximately 0.3 seconds for training. The matrix inversion required $4*10^{-4}$ seconds, and the NN inference was slightly faster but in the same order of magnitude.

In Fig. 5 we assumed 400 established paths and we examine the MAE of the algorithms for an increasing number of unknown links. In Fig. 5a, k is equal to 1. Note that the blue line of the proposed NN is under the yellow one of the NN that employs as features only nodes. The matrix inversion and the neural network that relies only on the links contained on a path, exhibit the worst accuracy as the number of unknown links increases. The reason is that these algorithms do not have the appropriate information to compensate for the missing topology knowledge. The proposed NN has a bit better MAE than the NN that uses only the origin and destination node as features. The MAE of our proposal is approximately 10^{-2} for most cases (until the number of unknown links is 16), and is approximately 0.2 for the other cases. The maximum error ranges from 10^{-3} for small number of unknown links, to approximately 100 and 700 when the unknown links are more. The 95% confidence interval ranges respectively from ± 0.2 to ± 2.5. As we can see these algorithms are able to compensate for the unknown network links by using the origin and destination node to estimate the delay of each path. In Figs. 5b and 5c we have $k > 1$. We notice that the proposed NN still achieves good accuracy even with 12 unknown links. After that, its accuracy deteriorates rapidly, and with 20 unknown links (all the links of the network), its accuracy is (as expected) equal to the NN whose features are only the origin and destination node. For $k = 2$ and $k = 3$, the MAE is approximately 0.2 and 2 respectively until the unknown links are 12. The maximum error is above 200 in almost all cases and the confidence interval is ± 1 and ± 2. As we can see, the combination of the features of the proposed NN perform well in this scenario. The two kind of features can work together and provide estimates even for a large number of unknown links, and for dynamic routing ($k > 1$). This is particularly useful in modern network scenarios where the topology of the network may not be completely known due to security reasons and also connections with the same origin and destination nodes may be routed differently over the physical topology.

B. Non-additive metrics evaluation

In this subsection we compare the accuracy of the three different ML algorithms for the case of the bandwidth estimation, which is a non-additive metric. The matrix inversion obviously cannot be evaluated in this case. In Fig. 6, we compare the accuracy of the algorithms again in terms of MAE and for different number of measured paths. In Fig. 6a, where $k = 1$, we can see that all algorithms achieve excellent accuracy with a relatively low number of measured paths. The MAE for 500 paths is in the order of 10^{-5} for all the algorithms, the maximum error is approximately 10^{-3} and the 95% confidence interval is approximately $\pm 10^{-5}$. In Figs. 6b, 6c where $k > 1$, the accuracy of the NN that uses only the origin and destination nodes is again not good, since it still lacks the necessary information to provide an accurate estimate. The other two algorithms have similar performance. Their MAE is in the order of 10^{-2} and 10^{-1}, the maximum error is 40 and 100 and the 95% confidence interval ± 0.16 and ± 0.33 for $k = 2$ and $k = 3$ respectively. These numbers are slightly worse than the case of the delay estimation. This indicates that the

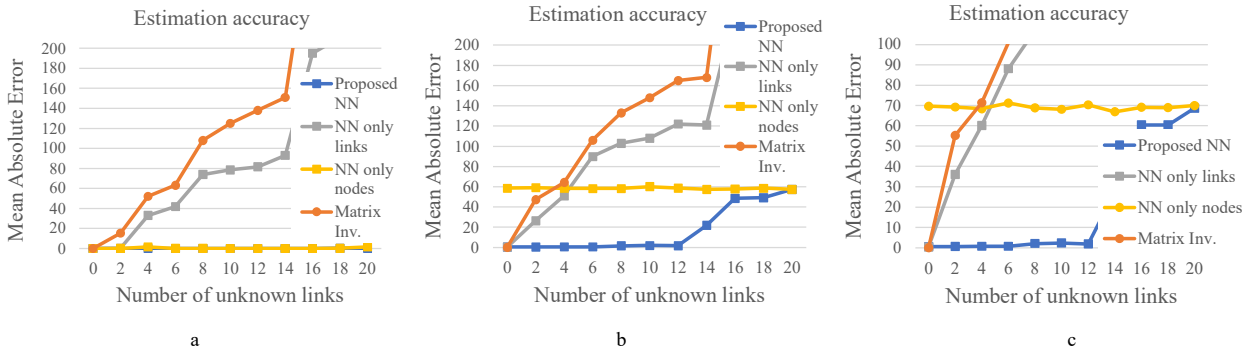

Fig. 5. Accuracy (delay metric) of the algorithms for different number of unknown links and a) $k = 1$, b) $k = 2$, c) $k = 3$

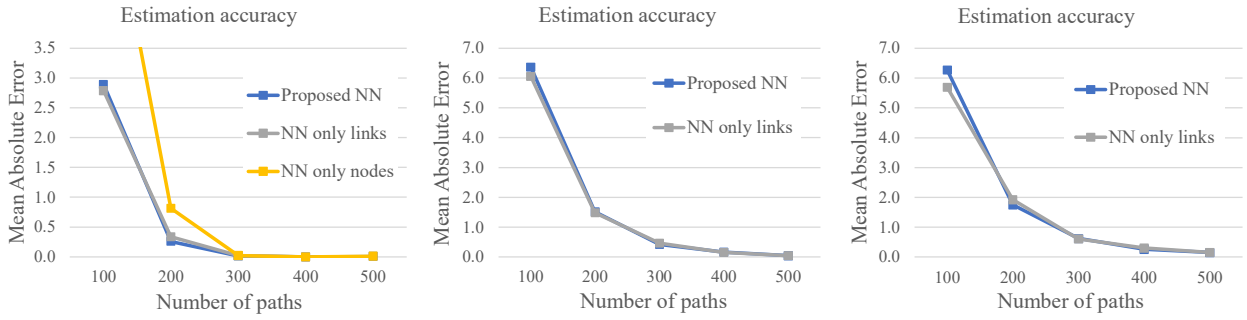

Fig. 6. Accuracy (bandwidth metric) of the algorithms for different number of paths and a) $k = 1$, b) $k = 2$, c) $k = 3$

examined NNs may not be the best fit for this kind of non-linear estimation. Finally, in Fig. 7 we considered 400 established paths and we again examined the MAE of the three algorithms for an increasing number of unknown links. For all the algorithms we notice similar behavior to the case of Fig. 5 (and again slightly worse MAE), indicating that our proposed NN can achieve good estimation accuracy for both additive and non-additive metrics examined.

V. Conclusions

In this paper we presented a novel ML formulation for Network Tomography under the assumptions of incomplete topology knowledge and dynamic routing. We designed suitable features that work well under these assumptions that are present in modern networks. The results indicate significant improvement in estimation accuracy compared to other approaches especially when the number of unknown network links is high and there are many different routing decisions for a given source destination pair (the difference in accuracy can be in an order of magnitude). Future work includes the evaluation of other ML algorithms such as gaussian process regression, and the localization of failures.

References

[1] Y. Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," J. Amer. Statist. Assoc. 91 365–377, 1996.

[2] C. A. Rødbro, P. A. Chou, Ü. Dogan, "Prediction of Bandwidth and Additive Metrics for Large Scale Network Tomography," Microsoft Research Technical Report MSR-TR-2016-57.

[3] N. G. Duffield, J. Horowitz, F. L. Presti, D. Towsley, "Network delay tomography from end-to-end unicast measurements," in Springer Thyrrhenian Int. Workshop on Digital Communications, Sept 2001.

[4] V. N. Padmanabhan, L. Qiu, H. J. Wang, "Passive network tomography using bayesian inference," in Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement, Nov. 2002.

[5] Y. Chen, D. Bindel, R. H. Katz, "Tomography-based overlay network monitoring," in Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement, Oct. 2003.

[6] O. Gurewitz, I. Cidon, M. Sidi, "One-way delay estimation using network-wide measurements," IEEE Transactions on Information Theory, 52(6), 2710-2724, 2006.

[7] L. Ma, T. He, K. K. Leung, D. Towsley, A. Swami, "Efficient identification of additive link metrics via network tomography," in IEEE 33rd International Conference on Distributed Computing Systems, July 2013.

[8] L. Ma, T. He, K. K. Leung, A. Swami, D. Towsley, "Inferring link metrics from end-to-end path measurements: Identifiability and monitor placement," IEEE/ACM Transactions on Networking, 22(4), 1351-1368, 2014.

[9] R. Castro, M. Coates, G. Liang, R. Nowak, B. Yu., "Network Tomography: Recent Developments," Statistical Science, 19(3), 499-517, 2004.

[10] G. Kakkavas, D. Gkatzioura, V. Karyotis, S. Papavassiliou, "A review of advanced algebraic approaches enabling network tomography for future network infrastructures," Future Internet, 12(2), 20, 2020.

[11] L. Ma, T. He, A. Swami, D. Towsley, K. K. Leung, "Network Capability in Localizing Node Failures via End-to-End Path Measurements," in IEEE/ACM Transactions on Networking, 25(1), 434-450, Feb. 2017.

[12] N. Bartolini, T. He, H. Khamfroush, "Fundamental limits of failure identifiability by boolean network tomography," in IEEE INFOCOM May 2017.

[13] R. Tagyo, D. Ikegami, R. Kawahara, "Network Tomography using Routing Probability for Undeterministic Routing," IEICE Transactions on Communications, 2020EBP3149, 2021.

[14] M. Rahali, J. Sanner, G. Rubino, "TOM: a self-trained Tomography solution for Overlay networks Monitoring," in IEEE Consumer Communications & Networking Conference (CCNC), Jan. 2020.

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

[16] A. Ibraheem, Z. Sheng, G. Parisis, D. Tian, "Neural Network based Partial Tomography for In-Vehicle Network Monitoring," in IEEE International Conference on Communications Workshops, June 2021.

[17] A. Rkhami, Y. Hadjadj-Aoul, G. Rubino, A. Outtagarts, "On the use of machine learning and network tomography for network slices monitoring," in IEEE 22nd International Conference on High Performance Switching and Routing (HPSR), Jun. 2021.

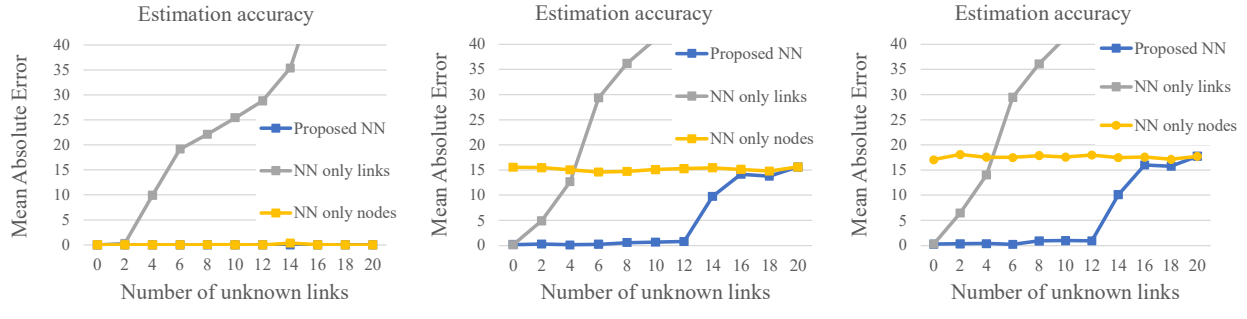

Fig. 7. Accuracy (bandwidth metric) of the algorithms for different number of unknown links and a) k=1, b) k=2, c) k=3