A APPENDIX

A.1 Datasets

The statistics of the real-world datasets are shown in Table 6. For the initialization representation of nodes, if the source data already contains node features, we use them as the initialization node representations, otherwise we use binary encoding to convert the node identifier (e.g., node index) into a binary representation. For path performance metrics, we use the edge labels of the source data to generate the path labels. In addition, we generate random edge labels of other path performance metric types for some networks, and then generate the path labels. Path performance metrics of each dataset are shown in the Table 7.

For the synthetic dataset, we utilize the Erdos-Renyi algorithm to generate networks of different sizes. Then, we generate random edge labels for all the above path performance metrics. When generating random edge labels for all datasets, for the addition and min/max metrics, the random labels are in $[1, 100]$, and for the multiplication metric, the random labels are in $[0.9, 0.999]$. For the Boolean metrics, each edge is randomly assigned a state of 0 or 1, where path labels are controlled to be balanced (the number of positive and negative labels will not less than 30%).

A.2 Implementation

The training data for each dataset depends on the sampling rate, i.e., $\frac{|S|}{|T|}$ which indicates how many node pairs' end-to-end path performance metric values are used for training. Half of node pairs in S is used as training data, and the other half is used as the validation set. That is, the number of node pairs actually used as training data is $\frac{|S|}{2}$. The rest of node pairs in $T \setminus S$ are used as test data.

To train DeepNT, we use CrossEntropyLoss as the loss function for classification tasks (Boolean metric) and MSELoss as the loss function for regression tasks (additive, multiplicative, and min/max metrics). Adam optimizer is used to optimize the model. The learning rate is set to 1e-4 across all tasks and models. The training batch is set to 1024 and the test batch is 2048 for all datasets. We use GCN as the GNN backbone, and the number of layers of GCN is 2. We use the mean pooling as the READOUT function. An one-layer MLP is used to make predictions. All models are trained for a maximum of 500 epochs using an early stop scheme with the patience of 10. The hidden dimension is set to 256. The hyperparameters we tune include the number of sampled shortest paths N in 1, 2, 3, the sparsity parameter α in 10e-5, 10e-4, 10e-3, 10e-2, and the path performance bound parameter γ in 0.1, 0.25, 0.5, 1, 2, 4, 8, 16.

For the comparison methods, we follow the original settings provided by the authors. In particular, the ANMI threshold ratio is reported as 30%. AMPR requires multiple probe tests, and we set the number of probe tests to the number of placed monitors, since each monitor tests the end-to-end path performance with other nodes.

A.3 Ablation Study on information aggregation

We compare our path aggregation with aggregation operations from SEAL (subgraph aggregation) [67], LESSR (edge-order preserving aggregation) [10], PNA (principal neighborhood aggregation) [16], and OSAN (ordered subgraph aggregation) [46]. A brief description of these aggregation methods is provided in Table 9.

The results in Table 8 demonstrate that the path aggregation mechanism in DeepNT achieves the lowest MSE for predicting additive metrics across most datasets, outperforming alternative aggregation methods. This suggests that path-specific attention effectively captures the most probable paths with optimal performance, leading to finer-grained and more accurate network tomography. Edge-order preserving aggregation performs best in the transportation dataset, likely due to its ability to preserve sequential dependencies in structured paths. However, in other cases, subgraph and ordered subgraph aggregation methods exhibit higher errors, indicating that local subgraph representations alone may not be sufficient for capturing global path information required in network tomography.

A.4 Expressiveness Study

A path from a source node v to a target node u is denoted by $p_{uv} = [v_1, v_2, \dots, v_k]$, where $v_1 = v$, $v_k = u$, and $(v_i, v_{i+1}) \in E$ for $i \in \{1, \dots, k-1\}$. Paths contain distinct vertices, and the length of the path is given by $|p_{uv}| = k - 1$, defined as the number of edges it contains. In this work, we consider paths that adhere to these criteria.

In practice, we only consider paths up to a fixed length L. Let $\mathcal{P}^L_{uv}$ denote the set of the sampled top-N optimal paths from u to v, selected based on the best path performance, with lengths not exceeding L. Recall that S and T represent the set of node pairs with observed path performance and all possible node pairs, respectively. Define $\mathcal{SP} = \bigcup_{<u,v>\in S} \mathcal{P}^L_{uv}$ as the collection of all sampled paths, and let $\mathcal{AP}$ denote the collection of all paths between the node pair combinations in T. We have $\mathcal{P}^L_{uv} \subset \mathcal{SP} \subseteq \mathcal{AP}$, where $\mathcal{SP} \to \mathcal{AP}$ as $N \to \infty$ and $S \to T$. To strengthen the proof, we first introduce the concepts of WL-Tree and Path-Tree as defined by Michel et al..

Definition A.1 (WL-Tree rooted at v). Let $G = (V, E)$. A WL-Tree W^L_v is a tree rooted at node $v \in V$ encoding the structural information captured by the 1-WL algorithm up to L iterations. At each iteration, the children of a node u are its direct neighbors, $\mathcal{N}(u) = \{w \mid (u, w) \in E\}$.

Definition A.2 (Path-Tree rooted at v). Let $G = (V, E)$. A Path-Tree P^L_v rooted at a node $v \in V$ is a tree of height L, where the node set at level k is the multiset of nodes that appear at position k in the paths of $\mathcal{P}^L_v$, i.e., $\{u \mid p^L(k) = u \text{ for } p^L \in \mathcal{P}^L_v\}$. Nodes at level k and level $k + 1$ are connected if and only if they occur in adjacent positions k and $k + 1$ in any path $p^L \in \mathcal{P}^L_v$.

Theorem A.3 (DeepNT-$\mathcal{AP}$ Expressiveness Beyond 1-WL). *Let $G = (V, E)$, W^L_v and W^L_u denote the WL-Trees of height L rooted at nodes $v, u \in V$, respectively. Let $f_{DeepNT}(v)$ and $f_{DeepNT}(u)$ represent the embeddings produced by DeepNT when it has access to the complete set of paths $\mathcal{AP}$. Then the following holds:*

1. *If $W^L_v \neq W^L_u$, then $f_{DeepNT}(v) \neq f_{DeepNT}(u)$.*
2. *If $W^L_v = W^L_u$, it is still possible that $f_{DeepNT}(v) \neq f_{DeepNT}(u)$.*

[1] https://publicdata.caida.org/datasets/topology/ark
[2] https://snap.stanford.edu/data
[3] https://github.com/bstabler/TransportationNetworks

Table 6: Dataset Statistics: the number of networks, (average) nodes and edges. – indicates the dataset has a singe network.

Statistics	Internet		Social Network			Transportation				
	IPV4	IPV6	Epinions	Twitter	Facebook	Anaheim	Winnipeg	Terrassa	Barcelona	Gold Cost
Graphs	10	10	-	-	-	-	-	-	-	-
Nodes	2866.0	1895.7	75,879	81,306	4,039	416	1,057	1,609	1,020	4,807
Edges	3119.6	2221.7	508,837	1768,149	88,234	914	2,535	3,264	2,522	11,140

Table 7: Properties of datasets. ✓ of binary encoding indicates that the original data has no node features, and we use binary encoding to generate the initial node representation. ✗ means that binary encoding is not used, but the node features of the original data are used. For the path performance metrics, ✓ for one metric indicates that the network has the true edge (link) labels of that metric, while ✓ indicates that random edge labels are generated for that performance metric.

Properties	Internet[1]		Social Network[2]			Transportation [3]				
	IPV4	IPV6	Epinions	Twitter	Facebook	Anaheim	Winnipeg	Terrassa	Barcelona	Gold Cost
Binary Enc.	✓	✓	✗	✗	✗	✓	✓	✓	✓	✓
Additive Path Performance Metrics										
Delay			✓	✓	✓					
RTT	✓	✓								
Distance			✓	✓	✓					
Flow Time						✓	✓	✓	✓	✓
Multiplicative Path Performance Metrics										
Reliability	✓	✓								
Trust Decay			✓	✓	✓					
Min or Max Path Performance Metrics										
Bandwidth	✓	✓								
Capacity						✓	✓	✓	✓	✓
Boolean Path Performance Metrics										
Is Trustworthy			✓							
Is Secure						✓	✓	✓	✓	✓

Table 8: MSE results using different aggregation layers.

DeepNT	Internet	Social	Transportation	Synthetic
w/ subgraph	117.65	22.90	21.77	77.93
w/ edge-order	84.25	14.19	**17.11**	62.65
w/ principal	94.43	20.01	22.89	84.07
w/ ordered subg.	101.73	17.26	20.57	71.14
w/ path (ours)	**71.08**	**10.81**	18.66	**59.04**

Proof. To prove this statement, we refer to Theorem 3.3 from [43], which states that if W_v^L is structurally different from W_u^L (i.e., not isomorphic), then P_v^L is also structurally different from P_u^L. Moreover, P_v^L and P_u^L can still differ even if W_v^L and W_u^L are identical.

We first address the case where $W_v^L \neq W_u^L$. It follows that P_v^L and P_u^L will not be isomorphic. The path aggregation layer in DeepNT is straightforward to prove as injective, as it employs a permutation-invariant readout function. Consequently, DeepNT aggregates path-centric structural information from P_v^L and P_u^L to produce embeddings $f_{\text{DeepNT}}(v)$ and $f_{\text{DeepNT}}(u)$, which are guaranteed to be distinct.

Now consider the case where $W_v^L = W_u^L$. This implies that 1-WL cannot distinguish between v and u. However, P_v^L and P_u^L may still differ. The path aggregation layer of DeepNT captures path-centric structural information from P_v^L and P_u^L, resulting in distinct embeddings $f_{\text{DeepNT}}(v)$ and $f_{\text{DeepNT}}(u)$. Thus, DeepNT surpasses the expressiveness of 1-WL. This completes the proof. □

Finally, Theorem 4.1 can be readily proved by noting that the node pairs $\langle u, v\rangle$ and $\langle u', v'\rangle$ are represented by concatenated node embeddings. Since $\mathcal{P}_{uv}^L \neq \mathcal{P}_{u'v'}^L$, the distinctiveness of these representations is ensured by the expressive power of DeepNT, which surpasses that of 1-WL.

A.5 Proof of Theorem 4.2

Proof. For the weak connectivity of the entire graph, following Zhao et al., we show that for any nonzero vector $x \in \mathbb{R}^{|V|}$:

$$x^\top Z_G x = \sum_{i\neq j} A_{ij}(x_i - x_j)^2 + \frac{1}{|V|}\left(\sum_{i=1}^{|V|} x_i\right)^2 > 0.$$

The first term ensures that differences between connected nodes contribute positively, while the rank-1 term $\frac{1}{|V|}\left(\sum_{i=1}^{|V|} x_i\right)^2$ prevents the existence of isolated components when edge directions are ignored. This guarantees that the graph G is weakly connected.

For strong connectivity within each component $g \in \mathcal{G}$, we establish both necessity and sufficiency. For necessity, if g is strongly connected, then its adjacency submatrix A_g is irreducible by the

Table 9: Comparison of different graph aggregation models.

Model	Formula	Description
DeepNT	$\hat{h}_v^{(n)} = h_v + \sigma\left(\sum_{x \in \mathcal{P}_{vu}^{(n)}} \alpha_{vx}^{(n)} h_x^{(n)}\right)$ $\alpha_{vx}^{(n)} = \text{softmax}(r^T [h_v, h_x^{(n)}])$ $h_v = \text{READOUT}(\hat{h}_v^{(n)} \cdot P_{vu}^{(n)} \subset P_{vu}^L)$	**Path aggregation**: Aggregates over sampled optimal paths.
SEAL	$h_v = \text{AGG}(h_v : v \in G_h^{x,y})$ $h_{vp} = \text{READOUT}(h_v : G_h^{x,y} \in \text{enclosing subgraphs})$	**Subgraph aggregation**: Aggregates node features within enclosing subgraphs extracted for target links.
LESSR	$h_v = \text{GRU}(h_v, h_z : z \in \mathcal{N}(v))$ $h_{vp} = \text{Attention}(h_v, h_x, \text{shortcut connections})$ $h_v = \text{READOUT}(\text{EOPA}(h_v), \text{SGAT}(h_v))$	**Edge-order preserving aggregation**: Combines GRU-based local aggregation with global attention from shortcut paths.
PNA	$h_{agg} = \text{AGG}(h_u : u \in \mathcal{N}(v))$ $h_{\text{scaled}} = \text{Scaler}(\text{deg}(v)) \cdot h_{\text{agg}}$ $h_v = \text{Combine}(h_{\text{scaled}})$	**Principal neighbourhood aggregation**: Combines mean, max, min, and std aggregators with degree-based scaling.
OSAN	$h_s = \text{AGG}(f_w(v) : v \in s)$ $h_v = \text{READOUT}(h_s : v \in s, s \in S)$	**Ordered subgraph aggregation**: Aggregates features from WL-labeled subgraphs containing the node.

Perron–Frobenius theorem, implying that $A_g + A_g^\top$ provides bidirectional coupling between any partition of nodes in g. Combined with the positive diagonal terms $\text{diag}(A_g \cdot \mathbf{1}^\top) + \text{diag}(A_g^\top \cdot \mathbf{1}^\top)$ and the rank-1 term $2\left(\frac{1}{|g|}\mathbf{1}\mathbf{1}^\top\right)$, this ensures $Z_g + Z_g^\top \succ 0$.

For sufficiency, we prove by contradiction. If g were not strongly connected, then A_g would be permutable to a block upper-triangular form as $A_g = \begin{pmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{pmatrix}$, where A_{11} and A_{22} correspond to disconnected subcomponents. We could construct a nonzero vector y conforming to these blocks such that $y^\top (Z_g + Z_g^\top) y = 0$, contradicting the positive definiteness of $Z_g + Z_g^\top$. Therefore, $Z_g + Z_g^\top \succ 0$ if and only if g is strongly connected. This completes the proof. □

A.6 Proof of Theorem 4.3

Proof Sketch. The coercivity of the objective function $g(\theta, \tilde{A}) + \alpha\|\tilde{A}\|_1$ ensures that the sequence $\{(\theta^k, \tilde{A}^k)\}$ is bounded. By the Bolzano–Weierstrass theorem, there exists at least one limit point $(\theta^*, \tilde{A}^*)$ [60]. Since the step size satisfies $\frac{1}{L} \leq \omega \leq \sqrt{\frac{L}{L+l}}$, the function sequence $\{\mathcal{F}(\theta^k, \tilde{A}^k)\}$ is non-increasing and converges to a finite value. The proximal gradient updates ensure $\|\nabla g(\theta^{k+1}, \tilde{A}^{k+1}) - \nabla g(\theta^k, \tilde{A}^k)\| \longrightarrow 0$, implying that the gradients converge. Taking the limit of the update steps, we obtain $\nabla_\theta g(\theta^*, \tilde{A}^*) = \mathbf{0}$, and the optimality condition for $\tilde{A}$ satisfies $\mathbf{0} \in \nabla_{\tilde{A}} g(\theta^*, \tilde{A}^*) + \alpha\partial\|\tilde{A}^*\|_1$. Since every limit point of $\{(\theta^k, \tilde{A}^k)\}$ satisfies these conditions, the sequence converges to a stationary point, completing the proof. □

Proof. With the chosen step size satisfying $\frac{1}{L} \leq \omega \leq \sqrt{\frac{L}{L+l}}$, the proximal gradient algorithm ensures:

$$\mathcal{F}(\theta^{k+1}, \tilde{A}^{k+1}) \leq \mathcal{F}(\theta^k, \tilde{A}^k).$$

This implies that the sequence $\{\mathcal{F}(\theta^k, \tilde{A}^k)\}$ is non-increasing. Since $\mathcal{F}(\theta, \tilde{A})$ is bounded below (due to the coercivity of the ℓ_1-regularization term $\alpha\|\tilde{A}\|_1$), the sequence $\{\mathcal{F}(\theta^k, \tilde{A}^k)\}$ converges to a finite value.

The boundedness of $\mathcal{F}(\theta^k, \tilde{A}^k)$ ensures that the sequence $\{(\theta^k, \tilde{A}^k)\}$ is bounded, which means $\{(\theta^k, \tilde{A}^k)\}$ has at least one limit point $(\theta^*, \tilde{A}^*)$.

Since $\|(\theta^{k+1}, \tilde{A}^{k+1}) - (\theta^k, \tilde{A}^k)\| \longrightarrow 0$, and ∇g is Lipschitz continuous, it follows that:

$$\|\nabla g(\theta^{k+1}, \tilde{A}^{k+1}) - \nabla g(\theta^k, \tilde{A}^k)\| \longrightarrow 0.$$

Therefore, the gradients $\nabla g(\theta^k, \tilde{A}^k)$ converge to $\nabla g(\theta^*, \tilde{A}^*)$ as $k \longrightarrow \infty$.

The update for θ in Algorithm 1 is:

$$\theta^{k+1} - \overline{\theta}^k = -\omega\nabla_\theta g(\overline{\theta}^k, \overline{A}^k).$$

As $k \longrightarrow \infty, \overline{\theta}^k \longrightarrow \theta^*, \theta^{k+1} - \overline{\theta}^k \longrightarrow 0$ and $\nabla_\theta g(\overline{\theta}^k, \overline{A}^k) \longrightarrow \nabla_\theta g(\theta^*, \tilde{A}^*)$. Then, it follows that:

$$\nabla_\theta g(\theta^*, \tilde{A}^*) = \mathbf{0}.$$

The update for $\tilde{A}$ in Algorithm 1 involves solving the proximal operator:

$$\tilde{A}^{k+1} = \arg\min_{\tilde{A}} \left(\frac{1}{2}\left\|\tilde{A} - \left(\overline{A}^k - \omega\nabla_{\tilde{A}} g(\overline{\theta}^k, \overline{A}^k)\right)\right\|_F^2 + \omega\alpha\|\tilde{A}\|_1\right).$$

This optimization is equivalent to applying the proximal mapping:

$$\tilde{A}^{k+1} = \text{prox}_{\omega\alpha\|\cdot\|_1}\left(\overline{A}^k - \omega\nabla_{\tilde{A}} g(\overline{\theta}^k, \overline{A}^k)\right),$$

where $\text{prox}_{\lambda\|\cdot\|_1}(f) = S_\lambda(f)$ is the soft-thresholding operator. The proximal mapping satisfies the optimality condition:

$$\mathbf{0} \in \tilde{A}^{k+1} - \left(\overline{A}^k - \omega\nabla_{\tilde{A}} g(\overline{\theta}^k, \overline{A}^k)\right) + \omega\alpha\partial\|\tilde{A}^{k+1}\|_1.$$

Rearranging this condition gives:

$$\mathbf{0} \in \nabla_{\tilde{A}} g(\overline{\theta}^k, \overline{A}^k) + \frac{1}{\omega}(\tilde{A}^{k+1} - \overline{A}^k) + \alpha\partial\|\tilde{A}^{k+1}\|_1.$$

As $k \longrightarrow \infty$, the extrapolated sequence $\overline{A}^k \longrightarrow \tilde{A}^*$ and the proximal updates $\tilde{A}^{k+1} \longrightarrow \tilde{A}^*$. Consequently, the term $(\tilde{A}^{k+1} - \overline{A}^k)/\omega \longrightarrow \mathbf{0}$. Thus, the limit point $\tilde{A}^*$ satisfies:

$$\mathbf{0} \in \nabla_{\tilde{A}} g(\theta^*, \tilde{A}^*) + \alpha\partial\|\tilde{A}^*\|_1.$$

We conclude that $(\theta^*, \tilde{A}^*)$ is a stationary point of the optimization problem since both optimality conditions are satisfied:

$$\mathbf{0} \in \nabla_\theta g(\theta^*, \tilde{A}^*), \quad \mathbf{0} \in \nabla_{\tilde{A}} g(\theta^*, \tilde{A}^*) + \alpha\partial\|\tilde{A}^*\|_1.$$

The algorithm may adjust $\tilde{A}^{k+1}$ to ensure connectivity. This adjustment does not violate convergence guarantees because it is a bounded perturbation that preserves the descent property.

Therefore, the sequence $\{(\theta^k, \tilde{A}^k)\}$ converges to the stationary point $(\theta^*, \tilde{A}^*)$: $\lim_{k\to\infty}(\theta^k, \tilde{A}^k) = (\theta^*, \tilde{A}^*)$. This establishes the convergence of the algorithm and completes the proof.

□

A.7 Proof of Theorem 4.4

Proof. As $N \to \infty$, the sampled paths $\mathcal{SP}$ converge to the complete set of paths for the observed node pairs S. As $S \to T$, the set of observed node pairs expands to include all possible node pairs in $T = V \times V$. Therefore, $\mathcal{SP} \to \mathcal{AP}$ as $N \to \infty$ and $S \to T$, which means DeepNT-$\mathcal{SP}$ converges to DeepNT-$\mathcal{AP}$.

By Theorem A.3 (DeepNT-$\mathcal{AP}$ Expressiveness Beyond 1-WL) and Theorem 4.1 (DeepNT-$\mathcal{AP}$ Distinguishes Node Pairs Beyond 1-WL), DeepNT-$\mathcal{AP}$ can uniquely identify and distinguish all node pairs based on differences in their path sets. Moreover, by Theorem 4.3 (Convergence of DeepNT's Optimization), the optimization of DeepNT converges to a stationary point $(\theta^*, \tilde{A}^*)$. This ensures that the empirical loss over the observed pairs S is minimized:

$$\mathcal{L}(\theta^*, \tilde{A}^*) = \sum_{\langle u,v\rangle \in S} l(f_{\text{DeepNT}}(u, v; \theta^*, \tilde{A}^*), y_{uv}),$$

where $l(\cdot,\cdot)$ measures the error between the predicted metrics $\hat{y}_{uv}$ and the true metrics y_{uv}. Consequently, as $S \to T$, the training error diminishes:

$$\mathbb{E}_{\langle u,v\rangle\sim S}[|\hat{y}_{uv} - y_{uv}|] \to 0.$$

As $S \to T$, the observed set S becomes dense, covering all node pairs in $T = V \times V$. Therefore, the unobserved set $T \setminus S$ becomes empty, i.e., $T \setminus S \to \emptyset$. Combined with the expressiveness of DeepNT and the completeness of path sampling, the model generalizes well to unobserved pairs $\langle u, v\rangle \in T \setminus S$. This ensures that the generalization error also diminishes:

$$\mathbb{E}_{\langle u,v\rangle\sim T\setminus S}[|\hat{y}_{uv} - y_{uv}|] \to 0.$$

Combining these results, the total error for all node pairs in T, which is the sum of the training error and the generalization error, converges to zero:

$$\epsilon_{\text{total}} = \mathbb{E}_{\langle u,v\rangle\sim S}[|\hat{y}_{uv} - y_{uv}|] + \mathbb{E}_{\langle u,v\rangle\sim T\setminus S}[|\hat{y}_{uv} - y_{uv}|] \to 0.$$

Finally, as $N \to \infty$ and $S \to T$, DeepNT-$\mathcal{SP}$ converges to DeepNT-$\mathcal{AP}$, and the predicted metrics $\hat{y}_{uv}$ converge in expectation to the true metrics y_{uv}:

$$\lim_{S\to T}\lim_{N\to\infty} \mathbb{E}_{\langle u,v\rangle\sim T}[|\hat{y}_{uv} - y_{uv}|] = 0.$$

This completes the proof. □

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009