Shooting Large-scale Traffic Engineering by Combining Deep Learning and Optimization Approach

CHENYI LIU, Tsinghua University, China and Zhongguancun Laboratory, China
HAOTIAN DENG, Tsinghua University, China
VANEET AGGARWAL, Purdue University, United States
YUAN YANG, Tsinghua University, China
MINGWEI XU, Tsinghua University, China

The rapid growth of global modern wide area networks has posed significant challenges to traffic engineering (TE). Existing TE methods often struggle to balance optimality with tractability, while recent machine learning based approaches fail to develop reliable strategies across diverse network scenarios. To address these issues, we introduce LO-TE, a novel TE solution integrating deep Learning and Optimization techniques. LO-TE operates in two phases: obtaining an initial solution and refining it to achieve a near-optimal TE solution. Our approach utilizes a scalable graph attention network for finding the necessary flows for refinement, paired with a refining algorithm based on linear programming. We demonstrate the application of LO-TE on three typical TE problems. We evaluate LO-TE on both real-world and self-generated large-scale topologies, demonstrating its strong generalizability across various TE problems and traffic models. The evaluation results indicate that LO-TE is 12x-188x faster than traditional TE optimization methods on large-scale network topologies, with an average performance gap of less than 6% compared to the optimal solution. Moreover, LO-TE outperforms state-of-the-art deep learning-based TE methods using limited training data, achieving only 1.8%-69% maximum link utilization under dynamic traffic conditions.

CCS Concepts: • **Networks** → **Traffic engineering algorithms**; • **Computing methodologies** → **Learning paradigms**.

Additional Key Words and Phrases: Traffic engineering, machine learning, network optimization

ACM Reference Format:
Chenyi Liu, Haotian Deng, Vaneet Aggarwal, Yuan Yang, and Mingwei Xu. 2025. Shooting Large-scale Traffic Engineering by Combining Deep Learning and Optimization Approach. *Proc. ACM Netw.* 3, CoNEXT1, Article 5 (March 2025), 21 pages. https://doi.org/10.1145/3709372

1 Introduction

Traffic engineering (TE) plays a crucial role in modern Wide Area Networks (WAN) to optimize network utilization and responsiveness, particularly amidst uncertain conditions [6, 14, 17]. Over the past decade, the evolution of WANs has been rapid, notably with the emergence of global Software-Defined WANs (SD-WAN) empowered by large-scale cloud providers. These SD-WANs now encompass hundreds of routing nodes and thousands of IP links, enabling the management of millions of site-to-site traffic demands with diverse Quality of Service (QoS) requirements [6].

Authors' Contact Information: Chenyi Liu, Tsinghua University, Beijing, China and Zhongguancun Laboratory, Beijing, China, chenyiliu9@gmail.com; Haotian Deng, Tsinghua University, Beijing, China, denght23@mails.tsinghua.edu.cn; Vaneet Aggarwal, Purdue University, West Lafayette, Indiana, United States, vaneet@purdue.edu; Yuan Yang, Tsinghua University, Beijing, China, yangyuan_thu@mail.tsinghua.edu.cn; Mingwei Xu, Tsinghua University, Beijing, China, xumw@tsinghua.edu.cn.

ACM 2834-5509/2025/3-ART5
https://doi.org/10.1145/3709372

However, the significant scale and dynamic nature of these networks present formidable challenges in designing highly efficient TE algorithms. The need for minute-level operational intervals further intensifies these challenges, particularly when dealing with diverse traffic demands [6]. Traditional approaches, often based on the multi-commodity flow (MCF) model, encounter scalability issues when applied to large-scale TE problems. Despite recent research efforts from academia and industry focusing on heuristic methods for accelerating large-scale TE problems, existing solutions struggle to strike a balance between optimality and tractability [22].

Machine learning has emerged as a promising solution for addressing TE challenges, with recent studies highlighting the potential of deep learning in expediting large-scale WAN traffic engineering [3, 24, 30]. However, current learning-based approaches are criticized for their poor reliability and limited generalizability across complicated and diverse network scenarios. In particular, deep learning models struggle to deliver precise solutions to complex large-scale TE problems that involve multiple objectives, thousands of constraints, and drastically changing traffic demands. In such cases, learning-based TE approaches may fail to balance different objectives while satisfying all constraints for unseen traffic demands, resulting in severe performance degradation and constraint violations.

To address these shortcomings, we propose LO-TE, an innovative solution that integrates deep learning and optimization techniques to tackle large-scale traffic engineering problems. LO-TE operates through a two-step process: first, generating an approximate initial TE solution, and second, refining this solution to achieve near-optimality.

The core innovation of LO-TE is its ability to leverage deep learning insights to refine a small portion of the initial approximate TE solution to the optimum, whether derived from heuristics or historical traffic matrices. This unified framework combines a scalable deep-learning feature extraction model with a linear programming (LP)-based solution-refining algorithm. In particular, the deep learning model predicts a small manageable subset of traffic demands for refinement, which the LP-based algorithm then uses to guide optimization directions. This approach significantly reduces the complexity and scale of the TE problem, making it tractable for large-scale scenarios while maintaining optimal or near-optimal performance.

One of the major challenges is designing a deep learning model to identify the minimal manageable subset of traffic demands for the TE solution refinement process. Identifying critical flows from a vast input state comprising millions of traffic flows is a complex task. In this work, we leverage the advantages of a graph attention network (GAT) [4], a state-of-the-art graph neural network, to extract key features from the large-scale input graph. Additionally, the input graph, which embeds network topology, traffic demand, and the initial TE solution, is designed to be scalable. This design allows for easy partitioning into batches to accommodate GPU memory limits and enable parallel processing. Notably, our input graph design incorporates the robustness of LO-TE. In particular, while the deep learning model may occasionally predict non-critical flows as important, this results in minimal additional computational overhead and does not lead to performance degradation.

To effectively train the GAT-based refining flow prediction model, we employ Binary Cross Entropy (BCE) loss, leveraging precalculated near-optimal refined solutions for supervised learning. Additionally, we introduce an unsupervised learning technique to further improve model performance and generalizability without ground-truth labeled data.

LO-TE is designed for a series of TE problems, encompassing a general TE problem formulation. We present a comprehensive formulation of the solution refining algorithm for this TE problem series, along with precise refining algorithms for three common TE problems: minimizing maximum link utilization (*MinMLU*), maximizing total throughput (*MaxThroughput*), and minimizing average path weight while maximizing total throughput (*MaxThroughput-MinWeight*). Moreover, we illustrate how to obtain the initial TE solution across various typical application scenarios, including different TE problems, real-world traffic models, and failure scenarios.

We evaluate the effectiveness of our approach on both synthetic and real-world large-scale network topologies. The results demonstrate the remarkable generalizability and scalability of LO-TE across various traffic engineering (TE) problems and traffic models. Specifically, LO-TE achieves speeds 12x to 188x faster than traditional optimization methods, with a performance gap of less than 6% from the optimal solution. Furthermore, LO-TE outperforms state-of-the-art deep learning-based TE methods, achieving a maximum link utilization ranging from 1.8% to 69% under dynamic traffic conditions. Additionally, LO-TE exhibits good performance under as link failures and drastic changes in network traffic. Through ablation studies and strategy visualizations, we delve deeper into the factors contributing to LO-TE's effectiveness.

We want to emphasize that this work introduces a groundbreaking approach to enhance large-scale traffic engineering problems by integrating machine learning with optimization techniques. While existing machine learning-based TE methods often struggle to directly achieve optimal solutions, our approach proposes using a graph attention network to identify critical components of the initial solution and refine it through an optimization framework. By leveraging an initial TE solution and an LP-based solution refining algorithm, the task for the GAT model is significantly simplified. This approach harnesses the complexities of traffic engineering problems and real-world traffic models, capitalizing on deep learning's ability to analyze complex, high-dimensional input data effectively. Moreover, it's worth noting that many network optimization problems, including network planning, share similarities with traffic engineering, suggesting broader applications for deep learning-based solution refining algorithms.

2 Background and Motivation

Typical TE approaches for WANs require a centralized controller to gather the network states, determine the routing paths and the corresponding traffic assignment for each flow demand, and then update the TE decision over the data plane accordingly. With the evolution of modern networks in recent years, many approaches for different objectives and accelerating large-scale TE problems have been proposed. In this section, we analyze the challenge of existing TE approaches facing large-scale network topology and show the key insights for this work.

2.1 Existing Traffic Engineering Approaches

Traffic engineering problems involve optimizing network-wide objectives such as maximum link utilization, total throughput, and overall transmission cost/latency, while adhering to several QoS constraints. We represent the network topology as a directional graph G comprising a node set V and an edge set E. A commonly used method in practice based on path-based MCF model [19] optimizes the assignments x of a set of traffic demands D over a fixed, precalculated candidate path set P for topology G. We consider a general TE problem with n objective functions $\Omega_1, \Omega_2, \ldots, \Omega_n$ as a lexicographic optimization problem, where the different objectives are ranked in order of importance to the decision-maker. In particular, objective Ω_1 is the most important, objective Ω_2 is the next most important, and so on. We formulate the general TE problem as follows:

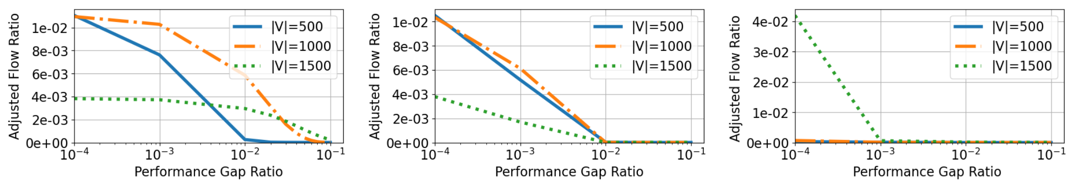
$$\begin{aligned} \operatorname{lex}\min_{x} \quad & \Omega_1(x, D, G, P), \Omega_2(x, D, G, P), \ldots, \Omega_n(x, D, G, P) && (1) \\ \text{s.t.} \quad & \text{DemandConstr}(x, D) && (2) \\ & \text{QoSConstr}(x, D, G, P) && (3) \end{aligned}$$

The linear constraints in Eq. (2) ensure that traffic assignment never exceeds or falls below traffic demands, while linear constraints in Eq. (3) ensure adherence to QoS requirements such as congestion-free traffic flow. Such a general TE problem could be formulated as a linear optimization problem and solved with off-the-shelf-optimizers [12]. However, as network scales rapidly grow, existing path-based MCF approaches encounter significant challenges.

Optmality vs. Tractability: Modern WANs, operated by global cloud providers, typically comprise hundreds of nodes and thousands of IP links, managing millions of traffic demands with diverse QoS requirements [6, 30]. The highly dynamic and unpredictable nature of traffic fluctuations in these networks necessitates minute-level TE decisions. However, traditional path-based MCF approaches [19], which rely on linear programming, struggle to adapt TE decisions to rapidly changing traffic while maintaining a sufficient number of candidate paths [6, 30]. Alternative methods, such as Demand Pinning (DP) and Partitioned Optimization Problems (POP), aim to optimize traffic by prioritizing large flows or dividing traffic demands into subproblems for parallel processing. Despite their promise, these methods often provide limited speed improvements and introduce significant performance gaps when managing large-scale TE problems [22]. With the exponential growth in network scale [33], there is an urgent need for scalable and effective TE solutions capable of addressing these challenges.

Deep Learning-based Approach: Machine learning has emerged as a promising approach for addressing large-scale TE problems. Recent studies [3, 24, 30] highlight the potential of deep learning in accelerating WAN traffic engineering. However, existing learning-based methods face significant challenges in training accurate and reliable deep learning models with limited data to directly address complex large-scale TE problems. In particular, it is quite difficult for deep learning models to simultaneously consider multiple optimization objectives (such as total throughput and QoS) while ensuring that the solutions satisfy millions of constraints. When the model's output solution deviates from the optimal solution, severe performance degradation may occur. In this work, we highlight that existing learning-based TE solutions are often criticized for their limited reliability and applicability, particularly in scenarios with scarce training data and rapidly fluctuating traffic demands.

2.2 Key Observation



(a) MinMLU. (b) MaxThroughput. (c) MaxThroughput-MinWeight.

Fig. 1. Correlation between the performance gap to the optimal solution and the minimum number of flows required for adjustment across three typical traffic engineering problems.

Solving large-scale TE problems is non-trivial, whether utilizing traditional off-the-shelf optimizers or deep learning methods. In this work, we propose a novel approach to address large-scale TE problems by integrating state-of-the-art deep learning techniques with traditional optimization methods. The core idea of our approach is based on the observation that attaining a near-optimal solution for a large-scale TE problem often requires only minor adjustments from an approximate initial TE solution. Thus, by leveraging an approximate initial TE solution, a promising deep learning model to determine the minimal necessary traffic flows for adjustments, and a linear optimization model for traffic reassignment, complex large-scale TE problems can be resolved efficiently and effectively.

We present the analysis of the key observation and the underlying principle as follows. To analyze the relationship between the performance gap to the optimal solution and the minimum traffic demands required for adjustment, we use a modeling approach. We formulate the problem of refining an TE solution towards its optimal value within a performance gap limit as a mixed 0-1 integer linear programming (MILP) problem:

$$\min_{x,q} \quad \sum_{d} q_d, \tag{4}$$

$$\text{s.t.} \quad |x^{d,p} - x^{d,p}_{\text{init}}| \le q_d, \quad \forall d \in D,\ p \in P_d, \tag{5}$$

$$q_d \in \{0, 1\}, \tag{6}$$

$$\Omega_i(x, D, G, P) \le \lambda \Omega_i(x_{\text{opt}}, D, G, P), 1 \le i \le n, \tag{7}$$

$$\text{DemandConstr}(x, D), \tag{8}$$

$$\text{QoSConstr}(x, D, G, P), \tag{9}$$

where $x^{d,p}$ represents the traffic assignments of flow demand d on path p and q_d indicates whether a traffic demand requires adjustment. Eq. (7) ensures that the solution performance does not exceed the allowed performance gap to the optimal solution by a positive factor λ[1].

The MILP problem described above poses challenges in large-scale networks due to its intractability. To address this, we relax q_d as a real variable constrained within the range $q_d \in [0, 1]$ to convert the MILP problem into an LP problem. This LP-relaxed formulation aids in determining the essential traffic demands requiring adjustment. Additionally, we introduce a rounding mechanism for the decision variables q_d, using a threshold of 0.01 in the LP-relaxed solution. Our experiments demonstrate that the rounded solution closely approximates the optimal MILP solution across all tractable test cases, typically involving topologies with approximately 100 nodes. Consequently, the following work utilizes the rounded LP-relaxed solution for data generation and analysis.

We analyze the correlation between the performance gap to the optimal solution and the minimum traffic demands required for adjustment over three typical TE problems, namely *MinMLU*, *MaxThroughput*, and *MaxThroughput-MinWeight*. The results, as depicted in Fig. 1, demonstrate a common observation across all three TE problems: only a small number of flows, particularly less than 1%, require traffic adjustment to achieve a nearly negligible gap (1%) to optimal network performance.

We note that the underlying principle behind this observation is that optimal solutions for different network topologies, traffic demands, and TE problems share many common patterns. Particularly, for TE problems focusing on load balancing (e.g., *MinMLU* and *MaxThroughput*), selecting shortest paths with minimum hops is often an effective strategy for many demands, as shortest paths occupy the least overall network link bandwidth. Additionally, the local network structure may encourage some flow demands to select non-shortest paths to avoid local congestion. Such load-balancing strategies apply to most traffic demands and only slight adjustments are required for removing local traffic overload. In particular, we find that a simple load-balancing solution, as outlined in § 3.2, only requires slight adjustment for most traffic demands, as shown in Fig. 1 (a) and (b).

Enhancing our key observation is the fact that traffic demands in practice often adhere to specific models like the hose model [7] or the gravity model [25]. These traffic models constrain traffic dynamics and bound the traffic volume in local topology structures. Consequently, the number of flows requiring adjustment could be quite small when traffic changes according to a specific model. This observation also helps us obtain the initial approximate TE solution for more complicated

[1]When the objective function is negative (e.g., for maximum total throughput, we have $\lambda < 1$; otherwise, $\lambda > 1$.

TE problems, where balancing different objectives (e.g., flow QoS and load balance) is necessary. In such cases, optimal TE solutions of historical traffic demands obeying specific models can be applied as approximate initial TE solutions for other traffic demands within the same traffic model, as shown in Fig. 1 (c).

Building on the key observations and principles, we explore training a deep learning model to predict minimal traffic demands for refinement. The model can identify flow features that may require reassignment based on different objectives and constraints. For instance, high-volume traffic passing through congested bottlenecks in the initial TE solution likely needs reassignment to avoid congestion and improve throughput. Predicting flows for refinement is simpler for the deep learning model than determining exact traffic assignments for millions of demands. Additionally, predicting which flows need refinement is more robust than predicting the entire TE solution, as incorrect predictions have less impact. Using an LP-based refinement algorithm, unnecessary traffic demands do not degrade performance, and missing necessary flows causes only minor performance loss. By predicting minimal demands for refinement, we significantly reduce the complexity of the LP-based TE optimization problem.

3 Methodology

3.1 Overview

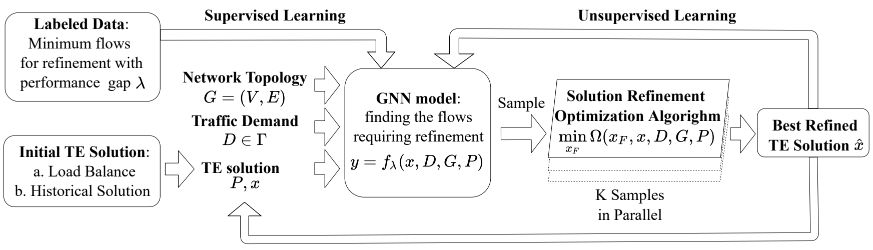

Supervised Learning
Unsupervised Learning
Labeled Data: Minimum flows for refinement with performance gap λ
Network Topology $G=(V,E)$
Traffic Demand $D \in \Gamma$
TE solution P, x
Initial TE Solution: a. Load Balance b. Historical Solution
GNN model: finding the flows requiring refinement $y=f_\lambda(x,D,G,P)$
Sample
Solution Refinement Optimization Algorighm $\min_{x_F} \Omega(x_F, x, D, G, P)$
K Samples in Parallel
Best Refined TE Solution $\hat{x}$

Fig. 2. Overall framework of LO-TE.

In this section, we introduce LO-TE, a novel approach designed to address large-scale TE problems by integrating deep learning and optimization techniques. LO-TE operates in two main steps: first, obtaining an initial approximate TE solution, and then refining it using a unified scheme, as illustrated in Fig. 2.

The core idea behind our approach is to identify the minimal traffic demands that require refinement from the vast array of traffic assignments in the original TE solution. To achieve this, we employ a scalable GAT to extract features from the extensive input state and distinguish the traffic demands likely in need of reassignment.

The refinement process could be conducted in parallel. A series of selected traffic demand subsets, determined by the GAT model's output distribution, serve as input to produce updated traffic assignments. We present a general formulation for the LP-based solution refinement problem and provide detailed designs for three typical TE problems: *MinMLU*, *MaxThroughput*, and *MaxThroughput-MinWeight*, in § 4.

To train the GAT model, we use labeled data that records the initial TE solution and the ground-truth TE solution with minimal flows for refinement regarding a small performance gap. Additionally, we propose an unsupervised learning approach to fine-tune the GAT model without a ground-truth TE solution. This enhances LO-TE's adaptability to network evolution.

3.2 Initial TE solution

Based on observations from data analysis and experiments on different TE problems, we discuss four approaches for obtaining approximate initial TE solutions in this section.

Minimum Hop: The path with the minimum hop count saves overall link bandwidth. Selecting the min-hop paths for all traffic demands can provide good approximate initial TE solutions for load balance objectives. However, network congestion may still occur on bottleneck links when all traffic demands take min-hop paths.

Load Balance: We define a load balance initial TE solution, achieved with a unit traffic volume for all traffic demands. The load balance solution consistently demonstrates strong performance, requiring minimal refinements to approach near-optimal TE solutions for arbitrary traffic demands. This is because the load balance solution provides good load-balancing traffic assignments, considering the network structure, such as bottleneck links.

Greedy QoS: In scenarios prioritizing QoS-aware TE like *MaxThroughput-MinWeight*, we introduce the Greedy QoS heuristic for TE solution generation. Specifically, we opt for the shortest paths with the smallest sum of weights for each traffic demand. Greedy QoS performs well in light-load scenarios, where only a small portion of traffic demands need reassignment to avoid congestion. However, it fails to balance load-balancing and QoS objectives under medium and heavy load scenarios.

Historical Solution: Traffic demands typically follow specific rules or constraints, which bound network dynamics and load levels at each local network structure. We found that optimal TE solutions under the same traffic models usually share common patterns. Thus, we can use the optimal solution to historical traffic demands as an efficient approximate initial solution for complicated TE problems with multiple objectives.

Coping with Failure: Robustness to failure scenarios is a critical challenge for large-scale TE problems. A commonly used approach [20] to address failures quickly is to proportionally redistribute the traffic traversing the failed node or link to the remaining paths based on the current traffic assignments. LO-TE adopts this processed solution as the initial TE solution and refines it to further enhance performance under failure scenarios.

3.3 Predicting Refinement Flows with GNN

In this work, We formulate the task of predicting the minimum flows required for refinement to achieve a near-optimal solution within a certain performance gap as follows

$$y = f_\lambda(x, D, G, P), D \in \Gamma \tag{10}$$

where x represent the traffic assignments over candidate path set P, and traffic demand D obeys the traffic model Γ. λ is the factor bounding the performance satisfying the target performance gap.

A crucial aspect of predicting fine-tuned flows for large-scale TE problems involves extracting essential information from complex network topologies with thousands of nodes, millions of traffic demands, and numerous candidate paths. This also includes considering the current TE solution's traffic assignment over these paths. Existing research, such as [30], establishes bidirectional relations between IP links and candidate paths in an input graph, using graph neural networks to exchange information among nodes representing candidate paths and IP links. However, these input graph models face scalability challenges, as GPU memory usage during the message-passing process can surpass hardware limits with increasing topology scale and the number of candidate paths.

We introduce a three-layer single-directional input graph for the feature aggregation process of graph neural network, as shown in Fig. 3. The IP link layer represents the link state, initialized with link capacity and usage under the current TE solution. The candidate path layer reflects the path state, initialized with traffic assignments from the current TE solution over each path. The flow

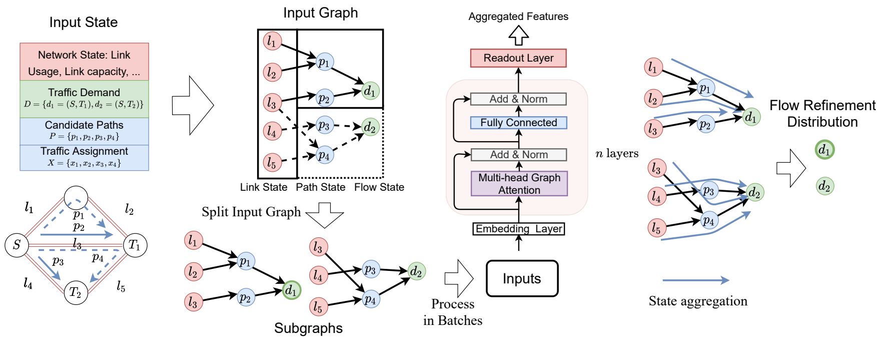


Fig. 3. GAT-based refinement flow prediction model.

layer represents traffic demand states, initialized with demand volume. Messages pass from the IP link layer to corresponding candidate paths and terminate at the flow state layer. The final flow states serve as input features for the fine-tuned flow prediction. This three-layer single-directional message-passing process can be easily split into batches and processed in parallel, circumventing the limitation of single GPU memory.

In our approach, we employ the graph attention network [4] for feature extraction and state aggregation. GAT is a type of graph neural network that utilizes attention mechanisms to process graph data. Attention coefficients are computed based on the attention mechanism, allowing target nodes to aggregate state information from nodes in the previous layer.

In summary, we transform the input state into the input graph, which could be subsequently divided into several subgraph batches based on hardware resources and settings. This input graph or its subgraphs are then fed into a GAT-based feature extraction model. The feature extraction model includes a state embedding layer, two state aggregation layers, and a readout layer to predict the probability of whether the traffic assignment of a flow requires refinement. Additionally, we implement residual and normalization mechanisms in each state aggregation layer to support a stack of more state aggregation layers.

3.4 LP-based Solution Refinement

In LO-TE, we select $\alpha * |V|$ flows with the highest predicted probability by the GAT model as the manageable traffic demands for solution refinement. Notably, with the number of flows for refinement bounded by $O(|V|)$, the number of decision variables reduces to $O(|V|)$, and the number of constraints in the refinement problem reduces to $O(|V| + |E|)$, whereas the original TE optimization problem scales as $O(|V|^2)$ decision variables and $O(|V|^2 + |E|)$ constraints.

We propose a linear optimization model to optimize the performance of the TE solution based on predicted flows for refinement. The general formulation of the linear optimization problem for refining the traffic demand subset D_F is as follows:

$$\begin{aligned} \operatorname{lex}\min_{x_F} \quad & \text{QoSPenalty}(x_F, x_{init}, D, G, P), \Omega_1(x_F, x_{init}, D, G, P), \Omega_2, \ldots, \Omega_n \\ \text{s.t.} \quad & \text{DemandConstr}(x_F, D) \end{aligned} \tag{11}$$

where QoSPenalty(x_F, x_{init}, D, G, P) represents the linear penalties indicating the violation of QoS constraints (e.g., traffic overload on the link) in the original problem. We set the QoSPenalty to the highest priority ensuring that the refinement algorithm prioritizes satisfying QoS constraints.

The refinement flow prediction and solution refinement process could be repeated for several steps. After the final solution refinement, we use a simple policy to ensure the output solution by

LO-TE is congestion-free. In particular, we normalize the traffic assignment over a path with the maximum link utilization on the path.

3.5 Training and LO-TE

In this section, we introduce two approaches for training the GAT-based model for predicting manageable traffic demand sets for solution refinement. Initially, we employ supervised learning to train the GAT-based model using ground-truth labels generated by the optimization approach. Subsequently, we also propose an unsupervised learning approach to train the GAT-based model without labeled data to improve the generalizability of LO-TE.

Supervised Learning: We generate ground-truth TE solutions using the LP-relaxed optimization model shown in Eq. (4) - (9). These ground-truth solutions help the model identify the minimal number of flows from the initial TE solution that need adjustment to achieve a specified performance gap. By comparing the initial TE solution with the ground-truth solution, we obtain labels indicating whether a flow's traffic assignment needs refinement. Notably, only a small subset of flows requires refinement. Directly supervising the model with these labels would lead to a conservative approach, resulting in many positive samples that cannot be accurately predicted. Therefore, we use a weighted Binary Cross Entropy loss function for supervised learning:

$$\mathcal{L}_{SL}(y, \hat{y}) = \sum_{d \in D} \frac{1}{n_{\hat{y}_d}} (\hat{y}_d \log y_d + (1 - \hat{y}_d) \log(1 - y_d)) \tag{12}$$

where y_d and $\hat{y}_d$ indicate whether demand d is predicted to be refined or labeled for refinement, respectively. The weight for traffic demand d is determined by the number of demands that share the same label (i.e., whether they are labeled for refinement).

The performance of the model trained with supervised learning is significantly influenced by the characteristics (e.g., traffic model) of the training data, which may gradually change when applied in practice. Additionally, generating ground-truth labels for every solution in the refinement process is impractical due to high computational overhead, especially for large-scale TE problems. Reinforcement learning, commonly used in recent research, struggles with poor convergence when faced with the enormous action space composed of millions of traffic demands. We discuss the reinforcement learning techniques we explored in appendix A. To address these challenges, we propose an unsupervised learning technique to further enhance the model's performance.

Unsupervised Learning: During the unsupervised learning process, we independently select K traffic demand subsets based on the output probabilities of the GAT model. For the k-th sample case, we select $\alpha \cdot |V|$ flows with the highest predicted probabilities, as well as an additional $\alpha \cdot |V|$ flows with the highest probabilities that were not selected in the $(k-1)$ sample cases before. An additional objective with the lowest priority is introduced to minimize the adjustments made to the selected traffic demands, effectively filtering out the most unnecessary demands. The calculated necessary flows from all sample cases are then combined, and the refinement algorithm is run again to determine the final set of necessary traffic flows. These final flows are used as the ground-truth labels, and the loss function in Eq. (12) is applied to update the model.

We provide additional details on training and implementation in appendix B, as constrained by the page limit.

4 Use cases

In this work, we apply LO-TE to three typical TE problems: *MinMLU*, *MaxThroughput*, *MaxThroughput-MinWeight*. For each TE problem, we can tailor a specific linear optimization problem based on the general formulation presented in Eq. (11). We provide the explicit formulation of the solution refinement problem for three typical TE problems: *MinMLU*, *MaxThroughput*, and *MinWeight*, and elaborate on the insights into these solution refinement problems.

4.1 MinMLU

MinMLU aims to minimize the maximum link utilization to ensure load balance and prevent congestion. The solution refinement problem for *MinMLU* focuses on refining the traffic assignment for flows associated with the most utilized links. The fine-tuning problem for *MinMLU* is formulated as follows:

$$\begin{aligned} \min_{x_F} \quad & U \\ \text{s.t.} \quad & \sum_{p \in P_d} x_F^{d,p} = 1, \forall d \in D_F \\ & \sum_{d \in D} \sum_{p \in P_d(e)} f_d \cdot x_F^{d,p} \leq U \cdot C(e), \forall e \in E \\ & x_F^{p,d} \geq 0, \forall d \in D_F, p \in P_d \end{aligned} \tag{13}$$

where f_d denotes the traffic volume of flow demand d, $P_d(e)$ denotes the candidate paths of demand d traversing link e, and $C(e)$ denotes the bandwidth capacity of link e. Since *MinMLU* does not have QoS constraints, the fine-tuning problem focuses solely on minimizing the MLU. We find that the general load balancing solution in § 3.2 is good enough to be applied as an approximate initial TE solution in LO-TE.

4.2 MaxThroughput

MaxThroughput aims to maximize total throughput considering network topology, available link bandwidth, and traffic demands. The refinement optimization problem for *MaxThroughput* is more complex than *MinMLU* as it needs to consider all the traffic demands rather than focus on the most overloaded link:

$$\begin{aligned} \operatorname{lex}\min_{x_F} \quad & \sum_{e} Z(e), -\sum_{d \in D_F} f_d \sum_{p \in P_d} x_F^{d,p} \\ \text{s.t.} \quad & \sum_{p \in P_d} x_F^{d,p} \leq 1, \forall d \in D_F \\ & Z(e) \geq \sum_{d \in D_F} \sum_{p \in P_d(e)} f_d x_F^{d,p} + \cdot U_{-F}(e) - C(e), \forall e \in E \\ & x_F^{p,d} \geq 0, \forall d \in D_F, p \in P_d \\ & Z(e) \geq 0, \forall e \in E \end{aligned} \tag{14}$$

where $U_{-F}(e)$ denotes the background link usage on link e from non-refined flow demands. In the solution refinement optimization problem, congestion-free constraints must be initially ensured. Although *MaxThroughput* is a more difficult TE problem than *MinMLU*, the general load balancing solution in § 3.2 is still good enough to be applied as an approximate initial TE solution in LO-TE.

4.3 MaxThroughput-MinWeight

Certain services necessitate minimizing the average transmission weight (e.g., cost, latency) while concurrently preventing congestion [15, 18]. In such cases, the TE solution must consider both total throughput and the overall weight while adhering to congestion-free constraints. The refinement optimization problem is formulated as follows:

$$\begin{aligned}
\operatorname{lex}\min_{x_F} \quad & \sum_{e} Z(e), -\sum_{d\in D_F} f_d \sum_{p\in P_d} x^{d,p}, \sum_{d\in D_F} f_d \sum_{p\in P_d} w_p x_F^{d,p} \\
\text{s.t.} \quad & \sum_{p\in P_d} x_F^{d,p} \le 1, \forall d \in D_F \\
& Z(e) \ge \sum_{d\in D} \sum_{p\in P_d(e)} f_d x_F^{d,p} + \cdot U_{-F}(e) - C(e), \forall e \in E \\
& x_F^{p,d} \ge 0, \forall d \in D_F, p \in P_d \\
& Z(e) \ge 0, \forall e \in E
\end{aligned} \tag{15}$$

where w_p denotes total weight of path p. We note that *MaxThroughput-MinWeight* is the most difficult problem in this work and we select optimal solution to historical traffic demands belong the same traffic model as initial TE solution.

5 Evaluation

In this section, we first describe the methodology for evaluating LO-TE in § 5.1. We then compare the optimality and tractability of LO-TE with state-of-the-art TE methods in § 5.2, showing that LO-TE achieves near-optimal network performance and significant acceleration on large-scale network topologies. Next, we assess LO-TE's robustness in the face of drastically changing traffic in § 5.3 and link failures in § 5.4. In § 5.5, we evaluate LO-TE's generalizability to different TE problems. We further evaluate its scalability under limited GPU memory in § 5.6 and examine the contribution of key components in LO-TE through an ablation study in § 5.7. Finally, we visualize the strategy behind LO-TE in appendix C.1.

5.1 Methodology

We implement LO-TE using Pytorch and PyG [8][2]. We evaluate our approach on a Linux server with two 2 GHz Xeon(R) Gold 6330 CPU, 512 GB Memory, and 2 RTX 6000 GPU. We list the experiment settings below.

Topology: We train and evaluate our approach using both real-world large-scale network topologies and generated topologies. For LO-TE's pretraining, we select 10 medium-sized WAN topologies from the Internet Topology Zoo [16]. We then evaluate LO-TE on the pan-European research network GEANT [26], and two large-scale WAN topologies, Cogentco and Kdl, from the Topology Zoo. Additionally, we assess LO-TE on an AS-level Internet topology, ASN [5], along with three generated AS-level topologies using the BRITE topology generator with the Waxman model [21]. A summary of these network topologies is provided in Table 1. For clarity, we remove nodes with a degree of one from the topologies.

Traffic Demand: For the GEANT topology, we use a publicly available 2-week traffic trace dataset, consisting of traffic matrices aggregated every 15 minutes [26]. For other large-scale network topologies, we evaluate our approach using two popular real-world traffic models, detailed as follows:

- *Traffic Burst* model simulates temporally varying traffic volumes. The base traffic demand is generated using the gravity model [25] and normalized to ensure the optimal maximum link utilization is 1. For each traffic matrix (TM) in this model, we randomly select 5% of the traffic demands and double their volumes to simulate bursty traffic.

[2]The code implementation is available at https://github.com/netlab-lcy/LO-TE.

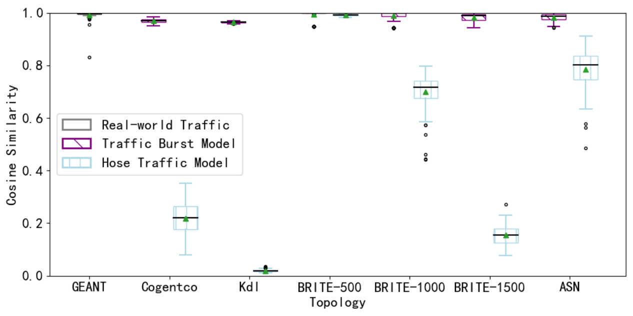


Fig. 4. Cosine similarity analysis for different traffic models using a window of 5 historical traffic matrices.

- *Hose* model is commonly used for traffic engineering and network planning by global service and cloud providers [2, 7]. In contrast to the Traffic Burst model, the Hose model allows for more dynamic traffic behavior, such as spatial drift in traffic demands, by only constraining the total ingress and egress traffic volumes at each node. We generate traffic demands that satisfy the Hose model as described in [2], and normalize the TMs of each topology with a consistent factor to ensure the overall optimal maximum link utilization is approximately 1.

We analyze the stability and predictability of real-world traffic and two synthetic traffic models, as shown in Fig. 4, by assessing the similarity between the current traffic matrix (TM) and historical TMs [20]. Specifically, for each observed TM, we consider a window of historical TMs, identify those most similar to the current TM, and compute the cosine similarity between them. Fig. 4 presents the distribution of these similarities, where each candlestick represents the distribution of cosine similarities for different traffic types. The box range extends from the 25th to the 75th percentile. A cosine similarity closer to 1 indicates that the traffic is more consistent with past traffic, suggesting higher stability and predictability, while a similarity closer to 0 indicates greater variability in the traffic pattern.

Our observations show that the traffic distribution in the Traffic Burst model is relatively stable, displaying similar patterns to real-world traffic but with slightly higher dynamics. Conversely, traffic demands generated by the Hose model exhibit significantly more dynamics across most network topologies. For evaluation, we use the Traffic Burst model for primary experiments and assess robustness to drastic traffic changes using the Hose model in § 5.3.

Baselines: We compare our approach with several heuristics applied to real-world global WANs [6], optimization-based algorithms, and state-of-the-art machine learning-based TE algorithms:

- *Path-MCF*: Path-MCF solves the original path-based MCF problem using Gurobi [12], an off-the-shelf optimizer. We adopt Path-MCF as an optimal baseline, using the same precalculated edge-disjoint shortest paths as LO-TE for candidate paths.
- *CSPF*: The Constrained Shortest Path First (CSPF) algorithm provisions each flow with the required traffic demand, considering the physical capacity of the network. CSPF selects the shortest path with the minimum weights among all paths that can accommodate the traffic demand. We implement the version described in [6] as a baseline.
- *TEAL*: TEAL [30] is a state-of-the-art TE algorithm that combines graph neural networks (GNN) and reinforcement learning (RL) to process traffic demands and output optimized network configurations. Leveraging GPU parallelism, TEAL accelerates TE control.
- *DOTE*: DOTE [24] is a TE algorithm that directly trains a deep neural network to predict the traffic matrix for the next time step and optimize traffic assignments based on historical traffic demand data.

- *FIGRET*: FIGRET [20] introduces a deep-learning based Fine-Grained Robustness-Enhanced TE scheme. This approach provides a novel solution by offering varying levels of robustness enhancements tailored to the unique traffic characteristics of different source-destination pairs.

Metrics: We consider the following performance metrics in evaluation.

- *Computation Time*: We measure the total time of each approach to calculate the TE decisions.
- *Maximum Link Utilization*: We apply the maximum link utilization over the whole network to measure the overall congestion level.
- *Demand Satisfaction*: We normalize the total throughput with the traffic demands to measure the capability of optimizing network throughput and satisfying flow demands. Note that in *MaxThroughput* and *MaxThroughput-MinWeight*, both LO-TE and baselines ensures to be congestion-free.
- *Average Transmission Weight*: We sum the traffic on each path according to the total weight of the links along the path, and normalize it by the total network throughput to obtain the average transmission weight of the unit traffic.

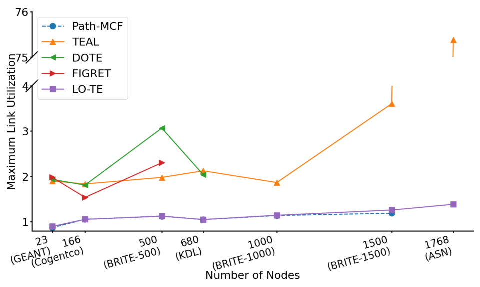
Training and Testing: We start by pretraining a set of general model parameters for a TE problem using 10 medium-sized topologies from the Topology Zoo, with generated labeled data and supervised learning, as detailed in Table 1. The labeled data used for pretraining LO-TE is generated with an allowed performance gap of 1%. For each application scenario (i.e., topology and traffic model), we further refine the pretrained model using unsupervised learning. For the GEANT topology, we utilize the first week's traffic demands for unsupervised learning and the second week's traffic demands for testing. For other large-scale topologies and traffic models, we generate 100 traffic matrices for unsupervised learning and an additional 100 traffic matrices for evaluation. We evaluate LO-TE on the three typical TE problems described in § 4, using *MinMLU* as the default TE problem for evaluation. The *Historical Solution* from § 3.2 is selected as the default initial TE solution. We apply precalculated edge-disjoint paths between each source-destination node pair as candidate paths for LO-TE, following a similar approach to TEAL.

5.2 Optimality vs. Tractability

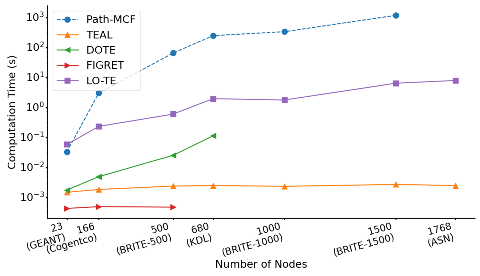
We present the average maximum link utilization (MLU) results and computation times of LO-TE and baseline algorithms in Fig.5, along with the cumulative distribution function (CDF) results for two representative topologies in Fig.12. Note that CSPF is excluded as it is unsuitable for the MinMLU problem, which requires congestion-free solutions. LO-TE achieves near-optimal performance in terms of MLU while significantly reducing computation time for large-scale network topologies. Specifically, LO-TE delivers a 12x-188x speed-up in large-scale topology cases compared to the optimization-based approach, with an average MLU gap of less than 6% relative to the optimal solution. We summarize the key findings below:

Learning-based TE: Current state-of-the-art learning-based TE algorithms demonstrate sub-optimal performance in the evaluation. The two DNN-based schemes, DOTE and FIGRET, are impractical for large-scale topologies with over 1000 nodes due to excessive GPU memory requirements. Furthermore, all three learning-based baselines fail to achieve optimal MLU, resulting in 46%-5300% higher MLU compared to LO-TE. These results highlight the challenges faced by existing learning-based TE algorithms in reliably and optimally addressing large-scale, complex TE problems, particularly when training data is limited.

Optimization-based TE: The computation time for optimization-based TE (Path-MCF) increases rapidly with topology size, becoming impractical (exceeding 1 hour) for ASN. An interesting observation is that for large-scale topologies, computation times for different traffic matrices



(a) Average maximum link utilization.



(b) Average computation time.

Fig. 5. Comparison between the performance and computation time of LO-TE with Path-MCF, TEAL, DOTE, and FIGRET across different network topologies for the MinMLU problem.

can vary significantly. Moreover, some cases are incorrectly marked as infeasible by the Gurobi optimizer due to numerical precision issues.

LO-TE: LO-TE demonstrates competitive performance, achieving up to two orders of magnitude acceleration compared to Path-MCF for all large-scale network topologies. Notably, LO-TE significantly outperforms all state-of-the-art learning-based TE algorithms, highlighting its strong generalizability and capability to handle complex TE problems.

ASN: We use ASN as a challenging test case with diverse link capacities. While LO-TE shows a slight increase in MLU on ASN, it still delivers acceptable performance. In contrast, TEAL exhibits poor performance, as it is specifically designed for the MaxThroughput problem, and its ADMM component cannot be directly applied to the MinMLU problem. However, this further underscores the challenges of developing a reliable strategy for large-scale, complex TE optimization problems using deep reinforcement learning.

5.3 Performance facing traffic change

We compare the performance of LO-TE on traffic matrices generated by the Hose model with baseline TE schemes to evaluate LO-TE's capability in handling drastically changing network traffic. The results are shown in Fig. 6. Across all network topologies, LO-TE achieves comparable MLU to the optimal solution (Path-MCF), demonstrating its strong capability to handle traffic variations. In contrast, state-of-the-art learning-based TE schemes show a significant gap in MLU performance compared to the optimal value. Additionally, as the topology scale increases, the MLU also rises.

The underlying reason may be the highly dynamic nature of the traffic's spatial distribution. Traffic demands may randomly migrate between endpoints in each test case, resulting in a wider range of flow changes. This makes traffic demands less predictable, posing challenges for TE schemes that heavily rely on historical TMs to predict future TMs. The greater diversity in traffic demand distribution also increases the difficulty of learning a accurate and reliable neural network model for predicting TE solutions.

5.4 Robustness to link failure

LO-TE addresses link failures by first reassigning the affected traffic flows using simple heuristics, and then using the processed TE solution as the initial TE solution. This initial traffic reassignment requires minimal computation time, and the entire process operates at the second-level scale, comparable to a standard inference step.

We compare the performance of LO-TE under link failure scenarios with Path-MCF, DOTE, and FIGRET. Note that the implementation of TEAL does not support link failure scenarios for

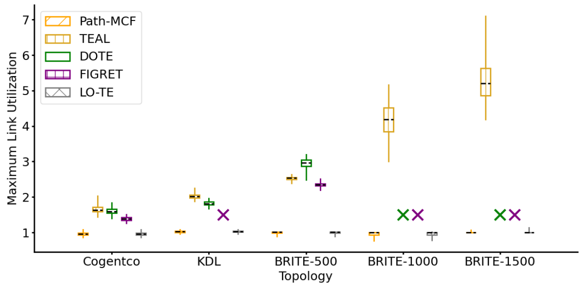


Fig. 6. Maximum link utilization of LO-TE and baseline TE schemes under highly dynamic traffic conditions using the Hose traffic model. Cross mark indicates that the corresponding TE scheme is infeasible for the topology.

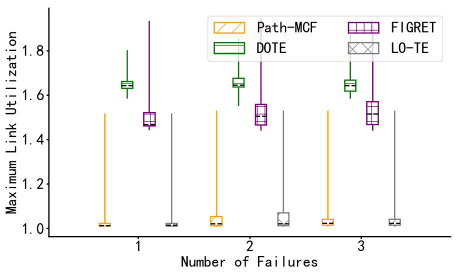
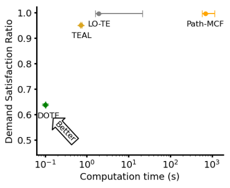
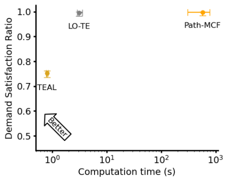


Fig. 7. Maximum link utilization of LO-TE and baseline TE schemes under failure scenarios on the Cogentco topology.

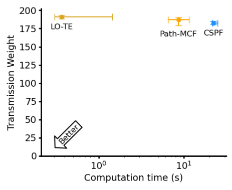


(a) Kdl.

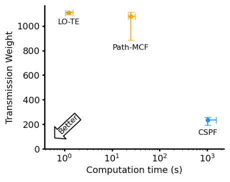


(b) BRITE-1500.

Fig. 8. Performance of LO-TE and baseline TE schemes for the MaxThroughput problem.



(a) Cogentco.



(b) BRITE-500.

Fig. 9. Performance of LO-TE and baseline TE schemes for the MaxThroughput-MinWeight problem.

the MinMLU TE problem. The results are presented in Fig. 7. LO-TE demonstrates comparable performance recovery to the optimal solution (Path-MCF), indicating a good robustness towards failure scenarios. In contrast, DOTE and FIGRET exhibit inferior performance, partly due to their suboptimal original TE solutions.

5.5 Generalizability for different TE problems

LO-TE naturally supports a wide range of TE problems with different objectives and constraints. We evaluate the performance of LO-TE for MaxThroughput and MaxThroughput-MinWeight objectives. Note that TEAL and DOTE do not support MaxThroughput-MinWeight, and FIGRET only supports MinMLU.

MaxThroughput: The evaluation results for MaxThroughput are shown in Fig. 8. On the KDL topology, LO-TE satisfies 99.6% of traffic demands, outperforming DOTE (63.8%) and TEAL (95.0%). For BRITE-1500, TEAL experiences a 25% performance degradation compared to LO-TE, highlighting its lower stability relative to LO-TE.

MaxThroughput-MinWeight: LO-TE consistently achieves a performance gap of less than 1% compared to the optimal solution in terms of throughput, the highest priority objective. In Fig. 9, we present the average transmission weight and computation time for LO-TE and baseline algorithms. LO-TE achieves comparable transmission weight to the optimal Path-MCF while offering significant acceleration in computation time. An interesting observation is that in the BRITE-500 topology, Path-MCF also exhibits a gap compared to CSPF. This discrepancy is attributed to the use of fixed candidate paths. However, CSPF requires much more computational time for searching paths iteratively.

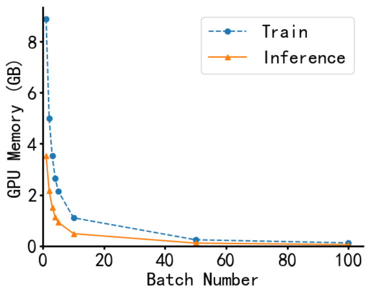


Fig. 10. GPU memory occupy of LO-TE's scalable GAT model processing input graph during training and inference.

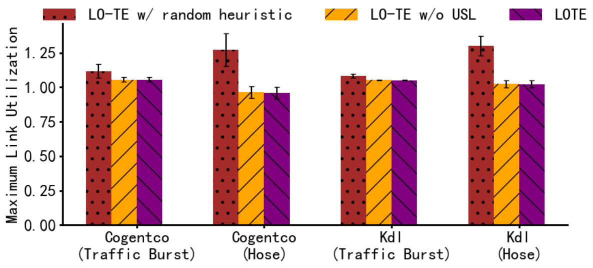


Fig. 11. Ablation study on the key components of LO-TE, including the GAT-based refinement flow prediction model and the unsupervised learning approach.

5.6 Scalability of LO-TE

LO-TE utilizes a splittable input graph design, allowing a large input graph to be divided into subgraphs and processed in batches. We evaluate the GPU memory usage of the GAT model under different numbers of batches for splitting the original input graph. The results are shown in Fig. 10. As the number of batches increases, the GPU memory required for processing the input graph decreases significantly. This scalable input graph design enhances LO-TE's ability to handle super-large-scale network topologies by enabling the processing of subgraph batches separately on single for multiple GPUs.

5.7 Ablation Study

We evaluate the performance of LO-TE's key components, including the GAT-based refinement flow prediction model and the unsupervised learning approach. The results are presented in Fig. 11.

GAT-based refinement flow prediction model: To assess the importance of the GAT-based model, we replace it with a random selection heuristic. The results show that the random heuristic leads to a 32% increase in MLU compared to using the trained GAT-based model to select the same number of flows for refinement.

Unsupervised learning: The pretrained general model, even without prior exposure to the target topologies, demonstrates surprisingly strong performance. This highlights the remarkable generalizability of LO-TE.

6 Conclusion

We introduced LO-TE, a TE solution that integrates deep learning and optimization to tackle large-scale TE problems. LO-TE operates in two phases: generating an initial solution and refining it using a scalable graph attention network to identify critical flows, paired with an LP-based refinement algorithm. Evaluations across various TE problems, dynamic traffic conditions, link failures, and large-scale topologies demonstrate LO-TE's generalizability and efficiency. It significantly accelerates computation while maintaining near-optimal performance and outperforms existing ML-based TE methods in dynamic traffic with limited training data. Overall, LO-TE provides an efficient, scalable solution for complex network optimization, highlighting the potential of combining deep learning and optimization for large-scale TE.

Acknowledgment

This work of Chenyi Liu, Haotian Deng, Yuan Yang, and Mingwi Xu are supported by the National Natural Science Foundation of China under Grant (62221003 and 62132004). Mingwei Xu and Yuan Yang are the corresponding authors. We thank the reviewers for their valuable comments.

References

[1] Firas Abuzaid, Srikanth Kandula, Behnaz Arzani, Ishai Menache, Matei Zaharia, and Peter Bailis. 2021. Contracting wide-area network topologies to solve flow problems quickly. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. 175–200.

[2] Satyajeet Singh Ahuja, Varun Gupta, Vinayak Dangui, Soshant Bali, Abishek Gopalan, Hao Zhong, Petr Lapukhov, Yiting Xia, and Ying Zhang. 2021. Capacity-efficient and uncertainty-resilient backbone network planning with hose. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*. 547–559.

[3] Guillermo Bernárdez, José Suárez-Varela, Albert López, Bo Wu, Shihan Xiao, Xiangle Cheng, Pere Barlet-Ros, and Albert Cabellos-Aparicio. 2021. Is machine learning ready for traffic engineering optimization?. In *2021 IEEE 29th International Conference on Network Protocols (ICNP)*. IEEE, 1–11.

[4] Shaked Brody, Uri Alon, and Eran Yahav. 2021. How Attentive are Graph Attention Networks? *arXiv* (2021).

[5] CAIDA. 2022. The CAIDA AS Relationships Dataset.

[6] Marek Denis, Yuanjun Yao, Ashley Hatch, Qin Zhang, Chiun Lin Lim, Shuqiang Zhang, Kyle Sugrue, Henry Kwok, Mikel Jimenez Fernandez, Petr Lapukhov, et al. 2023. EBB: Reliable and Evolvable Express Backbone Network in Meta. In *Proceedings of the ACM SIGCOMM 2023 Conference*. 346–359.

[7] John P Eason, Xueqi He, Richard Cziva, Max Noormohammadpour, Srivatsan Balasubramanian, Satyajeet Singh Ahuja, and Biao Lu. 2023. Hose-based cross-layer backbone network design with Benders decomposition. In *Proceedings of the ACM SIGCOMM 2023 Conference*. 333–345.

[8] Matthias Fey and Jan E. Lenssen. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

[9] Bernard Fortz, Jennifer Rexford, and Mikkel Thorup. 2002. Traffic engineering with traditional IP routing protocols. *IEEE communications Magazine* 40, 10 (2002), 118–124.

[10] Nan Geng, Tian Lan, Vaneet Aggarwal, Yuan Yang, and Mingwei Xu. 2020. A multi-agent reinforcement learning perspective on distributed traffic engineering. In *2020 IEEE 28th International Conference on Network Protocols (ICNP)*. IEEE, 1–11.

[11] Nan Geng, Mingwei Xu, Yuan Yang, Chenyi Liu, Jiahai Yang, Qi Li, and Shize Zhang. 2021. Distributed and adaptive traffic engineering with deep reinforcement learning. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*. IEEE, 1–10.

[12] Gurobi Optimization, LLC. 2022. Gurobi Optimizer Reference Manual. https://www.gurobi.com

[13] Chi-Yao Hong, Srikanth Kandula, Ratul Mahajan, Ming Zhang, Vijay Gill, Mohan Nanduri, and Roger Wattenhofer. 2013. Achieving high utilization with software-driven WAN. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*. 15–26.

[14] Chi-Yao Hong, Subhasree Mandal, Mohammad Al-Fares, Min Zhu, Richard Alimi, Chandan Bhagat, Sourabh Jain, Jay Kaimal, Shiyu Liang, Kirill Mendelev, et al. 2018. B4 and after: managing hierarchy, partitioning, and asymmetry for availability and scale in google's software-defined WAN. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. 74–87.

[15] Paras Jain, Sam Kumar, Sarah Wooders, Shishir G Patil, Joseph E Gonzalez, and Ion Stoica. 2023. Skyplane: Optimizing transfer cost and throughput using {Cloud-Aware} overlays. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. 1375–1389.

[16] Simon Knight, Hung X Nguyen, Nickolas Falkner, Rhys Bowden, and Matthew Roughan. 2011. The internet topology zoo. *IEEE Journal on Selected Areas in Communications* 29, 9 (2011), 1765–1775.

[17] Umesh Krishnaswamy, Rachee Singh, Nikolaj Bjørner, and Himanshu Raj. 2022. Decentralized cloud wide-area network traffic engineering with {BLASTSHIELD}. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. 325–338.

[18] Umesh Krishnaswamy, Rachee Singh, Paul Mattes, Paul-Andre C Bissonnette, Nikolaj Bjørner, Zahira Nasrin, Sonal Kothari, Prabhakar Reddy, John Abeln, Srikanth Kandula, et al. 2023. {OneWAN} is better than two: Unifying a split {WAN} architecture. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. 515–529.

[19] Praveen Kumar, Yang Yuan, Chris Yu, Nate Foster, Robert Kleinberg, Petr Lapukhov, Chiun Lin Lim, and Robert Soulé. 2018. {Semi-Oblivious} Traffic Engineering: The Road Not Taken. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. 157–170.

[20] Ximeng Liu, Shizhen Zhao, Yong Cui, and Xinbing Wang. 2024. FIGRET: Fine-Grained Robustness-Enhanced Traffic Engineering. In *Proceedings of the ACM SIGCOMM 2024 Conference*. 117–135.

[21] Alberto Medina, Anukool Lakhina, Ibrahim Matta, and John Byers. 2001. BRITE: An approach to universal topology generation. In *MASCOTS 2001, Proceedings Ninth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*. IEEE, 346–353.

[22] Pooria Namyar, Behnaz Arzani, Ryan Beckett, Santiago Segarra, Himanshu Raj, and Srikanth Kandula. 2022. Minding the gap between fast heuristics and their optimal counterparts. In *Proceedings of the 21st ACM Workshop on Hot Topics*

in Networks. 138–144.

[23] Deepak Narayanan, Fiodar Kazhamiaka, Firas Abuzaid, Peter Kraft, Akshay Agrawal, Srikanth Kandula, Stephen Boyd, and Matei Zaharia. 2021. Solving large-scale granular resource allocation problems efficiently with pop. In *Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles*. 521–537.

[24] Yarin Perry, Felipe Vieira Frujeri, Chaim Hoch, Srikanth Kandula, Ishai Menache, Michael Schapira, and Aviv Tamar. 2023. {DOTE}: Rethinking (Predictive){WAN} Traffic Engineering. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. 1557–1581.

[25] Matthew Roughan. 2005. Simplifying the synthesis of Internet traffic matrices. *ACM SIGCOMM Computer Communication Review* 35, 5 (2005), 93–96.

[26] Steve Uhlig, Bruno Quoitin, Jean Lepropre, and Simon Balon. 2006. Providing public intradomain traffic matrices to the research community. *ACM SIGCOMM Computer Communication Review* 36, 1 (2006), 83–86.

[27] Asaf Valadarsky, Michael Schapira, Dafna Shahaf, and Aviv Tamar. 2017. Learning to route. In *Proceedings of the 16th ACM workshop on hot topics in networks*. 185–191.

[28] Xipeng Xiao, Alan Hannan, Brook Bailey, and Lionel M Ni. 2000. Traffic Engineering with MPLS in the Internet. *IEEE network* 14, 2 (2000), 28–33.

[29] Zhiyuan Xu, Jian Tang, Jingsong Meng, Weiyi Zhang, Yanzhi Wang, Chi Harold Liu, and Dejun Yang. 2018. Experience-driven networking: A deep reinforcement learning based approach. In *IEEE INFOCOM 2018-IEEE conference on computer communications*. IEEE, 1871–1879.

[30] Zhiying Xu, Francis Y Yan, Rachee Singh, Justin T Chiu, Alexander M Rush, and Minlan Yu. 2023. Teal: Learning-Accelerated Optimization of WAN Traffic Engineering. In *Proceedings of the ACM SIGCOMM 2023 Conference*. 378–393.

[31] Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. 2023. Large language models as optimizers. *arXiv preprint arXiv:2309.03409* (2023).

[32] Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Basar. 2018. Fully decentralized multi-agent reinforcement learning with networked agents. In *International Conference on Machine Learning*. PMLR, 5872–5881.

[33] Hang Zhu, Varun Gupta, Satyajeet Singh Ahuja, Yuandong Tian, Ying Zhang, and Xin Jin. 2021. Network planning with deep reinforcement learning. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*. 258–271.

A Refinforcement Learning for LO-TE

In this section, we show the details of the reinforcement learning techniques explored in this work, which has been proved to be no feasible facing large-scale TE problems. Particular, we propose a general reward function, derive the policy gradient, and find reinforcement learning never work for such large-scale action spaces.

We propose maximizing the expected refinement gain at each iterative step while controlling the complexity of the LP-based solution refinement problem. To achieve this, we constrain the number of selected flow demands for refinement to $O(|V|)$, the number of nodes in the network topology. Specifically, we aim to keep the refinement set at $O(|V|)$ by penalizing excessive flow selections in the refinement performance gain loss function:

$$\mathcal{L}_{RL} = \hat{\Omega}(y) - \beta \,\mathrm{ReLU}\left(\sum_{d \in D} y_d - \alpha |V|\right), \tag{16}$$

where $\hat{\Omega}(y)$ represents the overall objective function in the solution refinement problem, encompassing TE objectives and QoS penalties, and β is a sufficiently large penalty factor.

Following the theoretical results in [32], the gradient of the expected loss with respect to the output probability distribution of flow demands is given by:

$$\nabla \mathbf{E}[\mathcal{L}_{RL}] = \sum_{d \in D} (\mathcal{L}_{RL} - F(y_{-d})) \nabla \log p(y_d), \tag{17}$$

where $F(y_{-d})$ is a function independent of the sampled result y_d.

To approximate the expected gradient in Eq. (17), we sample K finetuning cases in parallel at each iteration. The baseline function $F(y^{-d})$ is set based on the average performance gain among samples from the output distribution, significantly enhancing the convergence of the unsupervised learning process.

However, in large-scale network scenarios, the reinforcement learning-based approach proved infeasible due to non-convergence.

B Details of Training and Implementation

Training Process: We leverage the good generalizability of the GAT-based model by initially training it on a set of medium-scale network topologies with supervised learning. Subsequently, we utilize the pre-trained model as the starting point and fine-tune it specifically for large-scale topologies using unsupervised learning.

Implementation Implementing an efficient solution framework for large-scale TE problems is not an easy task. In LO-TE, we prioritize tensor calculations over *for* loops to optimize performance. We leverage sparse tensors and GPU acceleration supported by PyTorch to speed up the process. Additionally, by taking advantage of the TE solution refinement characteristics, we only need to send or change a small number of variables during the refinement process, and the refinement model building-up time could be reduced to be small accordingly.

C Additional Experiment Results

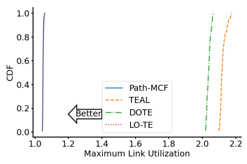
C.1 Visualization of LO-TE strategy

In this section, we analyze what LO-TE exactly learned by visualizing the output probability predicted by LO-TE and the ground-truth label of the minimum flows in need of refinement. We take the Cogentco topology and Traffic Burst Model as an example and show the results in Fig. 13.

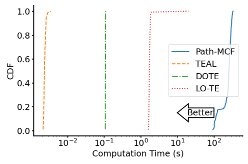
Notably, LO-TE learns to figure out almost all the necessary traffic demands for solution refinement labeled as ground-truth results in both training and testing sets. It is also reasonable for

Table 1. Network topologies used in evaluation.

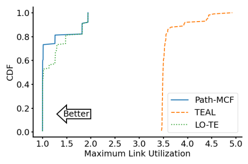
	Topology Number	Node Number	Edge Number	Type	Link Capacity
Pretrain	10	80 - 136	184 - 360	Real-WAN	Homogeneous
GEANT	1	23	74	Real-WAN	Homogeneous
Cogentco	1	166	424	Real-WAN	Homogeneous
KDL	1	680	1642	Real-WAN	Homogeneous
BRITE-500	1	500	2000	Generate-AS	Homogeneous
BRITE-1000	1	1000	4000	Generate-AS	Homogeneous
BRITE-1500	1	1500	6000	Generate-AS	Homogeneous
ASN	1	1739	8558	Real-AS	Heterogeneous



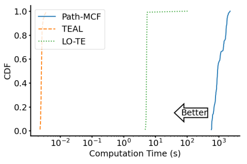
(a) MLU on KDL.



(b) Computation time on Kdl.

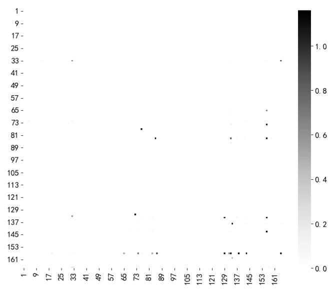


(c) MLU on BRITE-1500.



(d) Computation time on BRITE-1500.

Fig. 12. CDFs for the MLU and computation time of LO-TE and baseline algorithms on Kdl and BRITE-1500.


(a) Optimum.

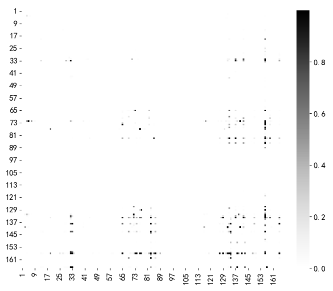

(b) LO-TE.

Fig. 13. Comparison between the distribution of traffic demands in need of refinement predicted by LO-TE and the ground-truth optimal solution. The results are based on the Cogentco topology and Traffic Burst traffic model. We average the ground-truth results and LO-TE outputs among the training and testing datasets to obtain the heatmap.

LO-TE to select some suspicious but unnecessary traffic demands to avoid missing necessary traffic demands. We also note that for a specific traffic model, the ground-truth distributions of minimal refinement traffic demands are consistent for both training and testing sets. Moreover, we find that only a small set of flows is likely to require refinement for a specific network topology and traffic model. Such characteristics of the refinement traffic demands make it easy for deep learning to make accurate predictions.

D Related work

Traffic Engineering has been a fundamental part of global service and cloud provider networks. Network operators have long relied on TE to maximize network utilization, ensure fairness among flows, and ensure the Quality of Service for the service flows [6, 14, 17]. As cloud services expand and global connectivity increases, TE has become even more critical in recent years. However, the rapid growth in network scale has introduced new challenges. Modern wide area networks managed by global cloud providers often have hundreds of nodes and thousands of IP links, catering to millions of traffic demands with diverse QoS requirements [6, 30]. This exponential growth necessitates more scalable and efficient TE solutions to handle the increased complexity and volume of network traffic [33]. We discuss traditional TE approaches, which are mainly based on optimization or heuristics, and recent machine learning-based TE approaches as follows.

Traditional Traffic Engineering Approaches Historically, Internet Service Providers (ISPs) used switch-native protocols such as MPLS and OSPF for traffic engineering in their networks [9, 28]. Over the last decade, large cloud providers have developed centralized software-defined TE systems in their planet-scale WANs to explicitly optimize for network characteristics like low latency, high throughput, and resilience to failures [13, 14]. Existing TE approaches usually frame the problem of traffic allocation as an optimization problem, which they solve periodically to compute flow allocations [19]. However, as WANs and traffic matrices have scaled up, the time required to solve these optimization problems has become a significant bottleneck in the TE control loop. Researchers have proposed techniques to reduce the TE optimization solving time, either by breaking the original TE problems into smaller subsets and then combining these solutions to compute traffic allocations for the entire network [1, 23], or by using simple heuristics to determine the traffic assignment for less important traffic demands to reduce the optimization problem scale [22]. Despite these advancements, traditional approaches like path-based Multi-Commodity Flow methods struggle to quickly adjust to traffic fluctuations while maintaining an adequate number of candidate paths for minute-level scheduling intervals [6]. Such techniques for enhancing large-scale TE problems often provide limited speed-up ratios and may introduce substantial performance degradation when facing increasingly large and complex network topologies [22, 30].

Machine Learning-Based Traffic Engineering Approaches Recently, machine learning has emerged as a promising approach to address the challenges of large-scale TE. Researchers have explored solving TE problems for optimal traffic allocation using unsupervised learning [24], deep reinforcement learning [27, 29], and multi-agent deep reinforcement learning [10, 11, 30]. However, existing approaches suffer from poor generalizability towards complex TE problems for larger network topologies, drastically changed traffic demands, and more complex objectives and constraints. The primary challenge lies in the difficulty of training deep learning models to solve complex optimization problems, such as large-scale traffic engineering, accurately and reliably [31]. Our work, on the contrary, provides a novel and general perspective by integrating traditional optimization approaches with state-of-the-art deep learning methods, to solve large-scale TE problems efficiently with negligible performance degradation.

E Discussion

We discuss some major concerns about LO-TE here.

What if the initial TE solution is suboptimal? In reality, selecting multiple initial TE solutions and refining them in parallel can reduce the impact of suboptimal starting points. For instance, we could select several historical TMs with different traffic patterns. Additionally, we could also select different types of initial TE solutions (e.g., Load Balance and Historical Solution).

Received June 2024; revised November 2024; accepted December 2024