A Machine Learning Approach to Solving Large Bilevel and Stochastic Programs: Application to Cycling Network Design

Timothy C. Y. Chan, Bo Lin
Department of Mechanical & Industrial Engineering, University of Toronto, {tcychan, blin}@mie.utoronto.ca,

Shoshanna Saxe
Department of Civil & Mineral Engineering, University of Toronto, s.saxe@utoronto.ca,

We present a novel machine learning-based approach to solving bilevel programs that involve a large number of independent followers, which as a special case include two-stage stochastic programming. We propose an optimization model that explicitly considers a sampled subset of followers and exploits a machine learning model to estimate the objective values of unsampled followers. Unlike existing approaches, we embed machine learning model training into the optimization problem, which allows us to employ general follower features that can not be represented using leader decisions. We prove bounds on the optimality gap of the generated leader decision as measured by the original objective function that considers the full follower set. We then develop follower sampling algorithms to tighten the bounds and a representation learning approach to learn follower features, which can be used as inputs to the embedded machine learning model. Using synthetic instances of a cycling network design problem, we compare the computational performance of our approach versus baseline methods. Our approach provides more accurate predictions for follower objective values, and more importantly, generates leader decisions of higher quality. Finally, we perform a real-world case study on cycling infrastructure planning, where we apply our approach to solve a network design problem with over one million followers. Our approach presents favorable performance compared to the current cycling network expansion practices.

Key words: bilevel optimization; two-stage stochastic programming; machine learning; model reduction; cycling network design.

1. Introduction

This paper is concerned with solving bilevel optimization problems with a large number of followers and where the feasible region of the leader is independent of the followers. A wide range of decision problems can be modeled this way, including active transportation network design (Liu et al. 2019, 2021b, Lim et al. 2021), network pricing (Van Hoesel 2008, Alizadeh et al. 2013), energy pricing (Fampa et al. 2008, Zugno et al. 2013), and portfolio optimization (Carrión et al. 2009, Leal et al. 2020). This model also generalizes two-stage stochastic programming. Indeed, if the objectives of the leader and followers

are identical, then this model becomes a two-stage stochastic program where the set of followers represent scenarios. Thus, in this paper, the reader should think of "leader" in a bilevel program and "first-stage decision maker" in a stochastic program as synonymous, and similarly for "follower" and "second-stage decision maker". As we discuss the bilevel or stochastic programming literature below, we use the corresponding terminology.

The main challenge of solving a bilevel problem with a large set of followers $\mathcal{S}$ stems from having to solve a large number of follower problems to evaluate the quality of the leader's decision. For the bilevel problem we consider, given its relationship to stochastic programming, we can draw on approaches to deal with large $\mathcal{S}$ from both communities. Two predominant strategies are: i) solving the problem with a small sample of $\mathcal{S}$, and ii) approximating the followers' cost without explicitly solving the followers' problems.

Sampling a smaller follower set can be done via random sampling (Liu et al. 2021b, Lim et al. 2021) or clustering (Dupačová et al. 2003, Hewitt et al. 2021, Bertsimas and Mundru 2022). Given a sample $\mathcal{T} \subseteq \mathcal{S}$, we can obtain a feasible leader solution by solving the reduced problem, which improves computational tractability and solution interpretability due to the reduced problem size. Furthermore, under suitable regularity assumptions, an optimal solution to the reduced problem provides a bound on the optimal value and optimal leader solution to the original problem (Römisch and Schultz 1991, Römisch and Wets 2007). However, there is no theoretical guarantee on the performance of the optimal solution to the reduced problem as measured by the original problem's objective.

Regarding the second strategy, many different algorithms have been developed to approximate the followers' cost. Such approximations can be obtained by relaxing the constraint that the followers' decisions are optimal for their own objectives and progressively refining the relaxed problem until the generated follower solutions are indeed optimal. Algorithms along this direction include L-shaped methods (Birge and Louveaux 2011), vertex enumeration (Candler and Townsley 1982), the Kuhn-Tucker approach (Bard 2013), and penalty function methods (White and Anandalingam 1993). These algorithms are generally not able to deal with huge $\mathcal{S}$ because the relaxed problem size still increases drastically as $|\mathcal{S}|$ increases. To overcome this issue, machine learning (ML) methods have recently achieved encouraging performance on approximating the second-stage cost as a function of first-stage decisions (Liu et al. 2021a, Dumouchelle et al. 2022) and learning second-stage decision rules (Chen et al. 2008) in two-stage stochastic programming. However, the former requires

identifying features that can be compactly represented using the leader's decision, which is practically challenging, while the latter may produce infeasible decisions when nontrivial second-stage constraints are present. Moreover, neither method has optimality guarantees.

In this paper, we build on the ideas from both strategies. In particular, we consider sampling a subset $\mathcal{T} \subseteq \mathcal{S}$. However, we also augment the overall objective function with an estimate of the objective value of the unsampled followers, from $\mathcal{S} \backslash \mathcal{T}$, using an ML model. Unlike existing methods that use offline ML models to map leader decisions to objective values, our ML model intakes follower features that are independent of leader decisions. We embed the ML model training into the bilevel problem to link the ML model with leader decisions. When optimizing leader decisions, the ML model is trained on a dataset of the sampled followers on-the-fly. Simultaneous optimization and ML model training enables derivation of new theoretical guarantees for the generated leader decisions. Finally, to demonstrate our methodology, we apply it to a cycling infrastructure design problem, completed in collaboration with the City of Toronto using real data. The methods we develop in this paper were driven by the need to solve this large, real-world application.

1.1. Cycling Infrastructure Planning

Cycling has become an increasingly popular transportation mode due to its positive impact on urban mobility, public health, and the environment (Mueller et al. 2018, Kou et al. 2020) In fact, during the COVID-19 pandemic, cycling popularity increased significantly since it represented a low-cost and safe alternative to driving and public transit, facilitated outdoor activities, and improved access to essential services (Kraus and Koch 2021, Buehler and Pucher 2021). However, cycling safety and comfort concerns have been repeatedly identified as major factors that inhibit cycling uptake and overall mode choice (Dill and McNeil 2016, Li et al. 2017). Building high-quality cycling infrastructure is among the most effective ways to alleviate cycling stress and enhance cycling adoption (Buehler and Dill 2016). In this paper, we develop an optimization model to aid cycling infrastructure planning by maximizing "low-stress cycling accessibility" subject to a fixed budget. Low-stress cycling accessibility, defined as the total amount of "opportunities" (i.e., people, jobs, retail stores) reachable by individuals using streets that are perceived to be safe for cycling, has been widely adopted to assess the service provided by transportation infrastructure and the impact of cycling infrastructure (Sisson et al. 2006, Lowry et al. 2012, Furth et al. 2018, Kent and Karner 2019, Imani et al. 2019, Gehrke et al. 2020, Lin et al. 2021). In our

problem, a transportation planner designs a cycling network subject to a given budget, considering that cyclists will use the low-stress network to travel to opportunities via shortest paths. The planner's objective is to maximize the total number of opportunities accessible via low-stress routes. The resulting formulation for the City of Toronto includes over one million origin-destination pairs (i.e., followers) between small geographic units known as census dissemination areas, motivating the development of our methodology.

1.2. Contributions

1. We develop a novel ML-augmented optimization model for solving bilevel optimization problems with a large number of followers. Our objective function has an exact component for a sampled subset of followers and an approximate component derived from an ML model. Sampling improves computational tractability, while the ML model ensures that the objective function better captures the impact of the leader's decision on the unsampled followers. The training of the ML model is embedded into the optimization model to enable the usage of predictive features that cannot be compactly represented using leader decisions. Indeed, we develop a theoretical bound on the quality of the generated solution as measured on the original objective function with the full set of followers. Given that our model generalizes two-stage stochastic programming, this ML-augmented approach also represents a new way for solving stochastic programming problems.

2. Informed by our theoretical insights, we develop practical strategies to enhance the performance of the ML-augmented model, including i) follower sampling algorithms to tighten the theoretical bounds, ii) strategies for hyperparameter tuning, and iii) a novel representation learning framework that automatically learns follower features that are predictive of follower objective values. To the best of our knowledge, this is the first application of representation learning to bilevel optimization.

3. We demonstrate the effectiveness of our approach via computational studies on a synthetic cycling network design problem. We show that i) our learned features are more predictive of follower objective values compared to baseline features from the literature; ii) our follower sampling algorithms further improve the ML models' out-of-sample prediction accuracy by a large margin compared to baseline sampling methods; iii) our strong predictive performance translates into high-quality and stable leader decisions from the ML-augmented model. The performance gap between our approach and sampling-based models without the ML component is particularly large when the follower sample is small.

4. In collaboration with the City of Toronto, we perform a real-world case study on cycling infrastructure planning in Toronto, Canada. We solve a large-scale cycling network design problem, demonstrating that our approach can increase accessibility over a greedy method by 19.2%, averaging over different budgets. If we consider 100 km of cycling infrastructure to be designed using a greedy method, our method can achieve a similar level of accessibility using only 70 km, equivalent to $18M in potential cost savings, or a 11.2% increase in accessibility if all 100 km was designed using our approach.

Proofs are in the Electronic Companion.

2. Literature review

2.1. Integration of Machine Learning and Optimization

"Predict, then optimize" is a common modeling paradigm that uses machine learning models to estimate parameters in an optimization problem, which is then solved with those estimates to obtain decisions (Ferreira et al. 2016, Elmachtoub and Grigas 2022). Recent progress has been made in using ML models to prescribe decisions based on contextual features (Ban and Rudin 2019, Notz and Pibernik 2022, Babier et al. 2022, Bertsimas and Kallus 2020), to improve optimization algorithms (Khalil et al. 2016, Tang et al. 2020, Jia and Shen 2021, Morabit et al. 2021), and to build end-to-end optimization solvers (Vinyals et al. 2015, Bello et al. 2017, Khalil et al. 2017, Nazari et al. 2018, Kool et al. 2019).

Our idea of using an ML model to predict followers' costs is similar to a stream of literature that integrates offline ML models into the solution of optimization problems to map decision variables to uncertain objective values (Biggs et al. 2017, Mišić 2020, Liu et al. 2021a, Bergman et al. 2022, Dumouchelle et al. 2022). In contrast, we integrate the ML model training into the optimization problem and use predictive features that cannot be compactly represented using our decision variables.

2.2. Scenario Reduction in Stochastic Programming

Scenario reduction has been extensively studied in the stochastic programming literature. One stream of literature quantifies the similarity between individual scenarios and then applies clustering methods to select a subset. Common measures include the cost difference between single-scenario optimization problems (Keutchayan et al. 2021), the opportunity cost of switching between scenarios (Hewitt et al. 2021), and the distance between scenario feature vectors (Crainic et al. 2014, Wu et al. 2022). Another stream of literature selects

a scenario subset from a given set to minimize the discrepancy between the distributions described by the two sets. Probabilistic metrics, including Wasserstein distance (Römisch and Schultz 1991, Bertsimas and Mundru 2022) and Fortet-Mourier metrics (Dupačová et al. 2003), derived from the stability theory of stochastic programming and heuristic loss functions (Prochazka and Wallace 2020) have been proposed to measure such discrepancies.

Our proposal of learning follower representations is similar to the work of Wu et al. (2022). However, our approach does not rely on the graph structure of the optimization problem and considers the impact of the leader's solution on the followers' costs. Our theoretical results deviate from the second stream of literature in that instead of focusing on the stability of the optimal value and optimal leader's solution, we provide solution quality guarantees for the leader's solution obtained from solving our ML-augmented problem.

2.3. Active Learning

Active learning aims to reduce the cost of data labeling by selecting the most "informative" subset of data for machine learning model training. Such a subset is typically constructed through an iterative process where an ML model is trained on the labeled data at each iteration and then queries one or more unlabeled data points based on informativeness measures such as prediction uncertainty (Lewis and Gale 1994), and expected model change (Settles et al. 2007). See Settles (2009) for a comprehensive review of the available measures. Instead of iteratively selecting unlabeled data, a subset of data may be selected once to improve data labeling efficiency (Sener and Savarese 2018). Our approach is most similar to Sener and Savarese (2018) because including the exact objective for a sampled set of followers in the reduced problem is analogous to costly labeling of a small set of data points once. We extend the core-set method developed by Sener and Savarese (2018) to derive a theoretical bound for the quality of the leader decisions from our ML-augmented model.

2.4. Strategic Cycling Infrastructure Planning

Previous studies on strategic cycling infrastructure planning have considered a variety of approaches. Many papers greedily choose road segments to install new cycling infrastructure using expert-defined metrics (Lowry et al. 2012, Kent and Karner 2019, Olmos et al. 2020) or visualization techniques (Larsen et al. 2013, Putta and Furth 2019). Optimization-based approaches typically consider minimizing the travel cost (Sohn 2011, Mesbah et al. 2012, Duthie and Unnikrishnan 2014, Bagloee et al. 2016, Mauttone et al. 2017), maximizing total utility (Bao et al. 2017, Liu et al. 2019, 2021b, Lim et al. 2021, Ospina et al. 2022),

or maximizing total ridership (Liu et al. 2022) given a large number of origin-destination (OD) pairs. Due to the large problem size, such models are usually solved with heuristics. To the best of our knowledge, only Liu et al. (2021b), Lim et al. (2021), and Liu et al. (2022) solve the problems to optimality at a city scale by randomly sampling OD pairs or restricting the routes that can be used by each OD pair. Our work adds to the literature by developing a computationally tractable solution method that can solve larger problems and does not require restrictions such as limited routes for each OD pair.

3. Model Preliminaries

In this section, we present the general bilevel problem of interest, a reduced version based on sampling, and our proposed model. We will use the following notational conventions. Vectors and matrices are denoted in bold, and sets are denoted in calligraphic font. We let $\mathbb{1}(\cdot)$ denote the indicator function, $[x]^+ = \max\{0, x\}$ and $[m] = \{1, 2, \ldots, m\}$.

3.1. The Bilevel Model

The following is the bilevel optimization problem of interest:

$$\underset{\mathbf{x}, \mathbf{y}^1, \ldots, \mathbf{y}^m}{\text{minimize}} \quad f(\mathbf{x}) + \sum_{s \in \mathcal{S}} q^s g(\mathbf{x}, \mathbf{y}^s) \tag{1a}$$

$$\text{subject to} \quad \mathbf{y}^s \in \underset{\mathbf{y} \in \mathcal{Y}^s(\mathbf{x})}{\arg\min}\, h^s(\mathbf{x}, \mathbf{y}), \quad \forall s \in \mathcal{S} \tag{1b}$$

$$\mathbf{x} \in \mathcal{X}. \tag{1c}$$

Let $\mathbf{x} \in \mathbb{R}^{n_1}$ denote the leader's decision with feasible set $\mathcal{X} \subseteq \mathbb{R}^{n_1}$ and cost function $f\colon \mathbb{R}^{n_1} \to \mathbb{R}$. Let $\mathcal{S} = [m]$ be the set of m followers, $\mathbf{y}^s \in \mathbb{R}^{n_2}$ be the decision of follower $s \in \mathcal{S}$, $g\colon \mathbb{R}^{n_1+n_2} \to \mathbb{R}$ measure the cost of a follower's decision, and $q^s \in \mathbb{R}_+$ be a non-negative weight. To determine the optimal decision of follower s, we assume the follower is optimizing an objective function $h^s\colon \mathbb{R}^{n_1+n_2} \to \mathbb{R}$ subject to a non-empty feasible set $\mathcal{Y}^s(\mathbf{x}) \subseteq \mathbb{R}^{n_2}$ that depends on the leader's decision. Note that when g and h^s are identical for all $s \in \mathcal{S}$, this problem is equivalent to a two-stage stochastic program

$$\begin{aligned} \underset{\mathbf{x}, \mathbf{y}^1, \ldots, \mathbf{y}^m}{\text{minimize}} \quad & f(\mathbf{x}) + \sum_{s \in \mathcal{S}} q^s g(\mathbf{x}, \mathbf{y}^s) \\ \text{subject to} \quad & \mathbf{y}^s \in \mathcal{Y}^s(\mathbf{x}), \quad \forall s \in \mathcal{S} \\ & \mathbf{x} \in \mathcal{X}, \end{aligned}$$

where the decisions of the leader and followers correspond to the first-stage and second-stage decisions, respectively, S is the set of second-stage scenarios, and q^s (suitably normalized) is the probability of realizing scenario s.

To simplify the notation, we write the bilevel problem as

$$\min_{x\in\mathcal{X}} F(\mathbf{x}), \tag{2}$$

where

$$F(\mathbf{x}) := f(\mathbf{x}) + \sum_{s\in\mathcal{S}} q^s G^s(\mathbf{x}) \tag{3}$$

and

$$G^s(\mathbf{x}) := \min_{\mathbf{y}^s} \left\{ g(\mathbf{x},\mathbf{y}^s) \,\middle|\, \mathbf{y}^s \in \arg\min_{\mathbf{y}\in\mathcal{Y}^s(\mathbf{x})} h^s(\mathbf{y}) \right\}, \quad \forall s\in\mathcal{S}. \tag{4}$$

Let $\mathbf{x}^* \in \arg\min_{x\in\mathcal{X}} F(\mathbf{x})$ be an optimal solution to (2).

3.2. Reduced Model

Given a sampled follower set $\mathcal{T}\subseteq\mathcal{S}$, we consider the reduced problem

$$\min_{x\in\mathcal{X}} \bar{F}_{\mathcal{T}}(\mathbf{x}), \tag{5}$$

where

$$\bar{F}_{\mathcal{T}}(\mathbf{x}) := f(\mathbf{x}) + \sum_{t\in\mathcal{T}} r^t G^t(\mathbf{x}), \tag{6}$$

$G^t(\mathbf{x})$ is as defined in (4), and the weight assigned to scenario $t\in\mathcal{T}$, $r^t\in\mathbb{R}_+$, may be different from q^t, due to re-weighting. Let $\bar{\mathbf{x}}_{\mathcal{T}}$ be an optimal solution to (5).

For the special case of a two-stage stochastic program, stability results have been established for problem (5). Under certain regularity conditions, it is possible to bound $|F(\mathbf{x}^*) - \bar{F}_{\mathcal{T}}(\bar{\mathbf{x}}_{\mathcal{T}})|$ (Römisch and Schultz 1991, Römisch and Wets 2007, Bertsimas and Mundru 2022) and $\|\mathbf{x}^* - \bar{\mathbf{x}}_{\mathcal{T}}\|$ (Römisch and Schultz 1991). However, there is no bound in the literature on $|F(\mathbf{x}^*) - F(\bar{\mathbf{x}}_{\mathcal{T}})|$, which is something we develop using the model from the next section.

3.3. ML-Augmented Model

Given a sampled follower set $\mathcal{T}\subseteq\mathcal{S}$, we propose the following model

$$\underset{\mathbf{x}\in\mathcal{X},P\in\mathcal{P}}{\text{minimize}} \quad f(\mathbf{x}) + \sum_{t\in\mathcal{T}} q^t G^t(\mathbf{x}) + \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s P\left(\mathbf{f}^s\right) \tag{7a}$$

$$\text{subject to} \quad \sum_{t\in\mathcal{T}} m^t \left| P(\mathbf{f}^t) - G^t(\mathbf{x}) \right| \le \bar{L}. \tag{7b}$$

This formulation augments the reduced problem by integrating an ML model that predicts the costs of the unsampled followers in $\mathcal{S}\backslash\mathcal{T}$. The ML model is specified by $P\colon \mathbb{R}^\xi \to \mathbb{R}$, which predicts the cost of follower $s \in \mathcal{S}$ based on a feature vector $\mathbf{f}^s \in \mathbb{R}^\xi$. We use $\mathcal{P}$ to denote the function class of ML models, $\bar{L} \in \mathbb{R}_+$ to be a user-defined upper bound on the training loss, and m^t to be a weight assigned to follower t for calculating the training loss of P. The training of P on dataset $\{\mathbf{f}^t, G^t(\mathbf{x})\}_{t\in\mathcal{T}}$ is embedded into the problem via the training loss constraint (7b). We choose the L1 loss because it can be easily linearized.

Let $\mathcal{Z}(\mathcal{X}, \mathcal{P})$ denote the feasible region of problem (7). We can write problem (7) as

$$\min_{(\mathbf{x},P)\in\mathcal{Z}(\mathcal{X},\mathcal{P})} \hat{F}_\mathcal{T}(\mathbf{x}, P) \tag{8}$$

where

$$\hat{F}_\mathcal{T}(\mathbf{x}, P) := f(\mathbf{x}) + \sum_{t\in\mathcal{T}} q^t G^t(\mathbf{x}) + \sum_{s\in\mathcal{S}\backslash\mathcal{T}} q^s P\left(\mathbf{f}^s\right). \tag{9}$$

Problem (8) provides a general structure for our modeling approach. Its effectiveness on a given problem depends on multiple factors: i) function class $\mathcal{P}$, ii) weighting scheme m^t and upper bound $\bar{L}$, iii) sample $\mathcal{T}$, and iv) availability of predictive follower features $\mathbf{f}^s, s \in \mathcal{S}$. We address the first three items in Section 4 and the fourth in Section 5.

4. Integrating a Prediction Model

In Section 4.1, we introduce two classes of prediction models – one non-parametric and one parametric – that are compatible with our ML-augmented model. We provide theoretical bounds on performance in Section 4.2. Finally, we present algorithms and discuss practical implementation, based on insights from examining the bounds, in Section 4.3.

4.1. Function Classes

4.1.1. k-nearest neighbor regression. For any fixed $\mathbf{x}$, let

$$P(\mathbf{f}^s) = \frac{1}{k} \sum_{t\in\mathcal{T}_k(\mathbf{f}^s)} G^t(\mathbf{x})$$

where k denotes the size of neighborhood considered and $\mathcal{T}_k(\mathbf{f}^s) \subseteq \mathcal{T}$ represents the k-nearest neighbors in the sampled set of follower s. Since kNN regression is a non-parametric

method that does not require training, we do not require constraint (7b). Equivalently, we can simply set $\bar{L} = \infty$ or $m^t = 0$ for all $t \in \mathcal{T}$. Then, problem (8) becomes

$$\underset{\mathbf{x}\in\mathcal{X}}{\text{minimize}} \quad f(\mathbf{x}) + \sum_{t\in\mathcal{T}} q^t G^t(\mathbf{x}) + \sum_{s\in\mathcal{S}\setminus\mathcal{T}} \sum_{t\in\mathcal{T}_k(\mathbf{f}^s)} \frac{q^s}{k} G^t(\mathbf{x}). \tag{10}$$

Note that our ML-augmented model is also compatible with other non-parametric prediction models, such as locally weighted regression (Cleveland and Devlin 1988) and kernel regression (Parzen 1962), which assign non-uniform weights to nearest neighbors. However, for those models, our theoretical bound (Section 4.2.3) becomes a complicated function of the distances from each follower to its nearest neighbors, making it difficult to formulate practical follower selection algorithms to tighten the bound. We thus leave the integration of more sophisticated non-parametric models for future work.

4.1.2. General parametric regression. Consider a general parametric regression model written as $P(\mathbf{f}^s; \mathbf{w})$, where $\mathbf{w}$ represents the parameters of model P and $\mathcal{W}$ is the feasible set of model parameters. Then, problem (8) becomes

$$\min_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}} \left\{ f(\mathbf{x}) + \sum_{t\in\mathcal{T}} G^t(\mathbf{y}) + \sum_{s\in\mathcal{S}\setminus\mathcal{T}} P(\mathbf{f}^s; \mathbf{w}) \,\middle|\, \sum_{t\in\mathcal{T}} m^t_{1,\mathcal{S}\setminus\mathcal{T}} \left| G^t(\mathbf{y}) - P(\mathbf{f}^t; \mathbf{w}) \right| \le \bar{L} \right\} \tag{11}$$

where $m^t_{1,\mathcal{S}\setminus\mathcal{T}}$ is the number of followers in $\mathcal{S}\setminus\mathcal{T}$ whose nearest neighbor in $\mathcal{T}$ is t.

For model (11) to be effective, one should choose a function class that can be compactly represented with $\mathbf{w}$ and $\mathbf{f}$. For example, a linear regression model $P(\mathbf{f}; \mathbf{w}) = \mathbf{w}^\intercal \mathbf{f}$ can be incorporated using only ξ additional continuous decision variables $\mathbf{w} \in \mathbb{R}^\xi$. An additional set of $|\mathcal{T}|$ continuous decision variables and $2|\mathcal{T}|+1$ linear constraints are required to linearize the L1 training loss. Such a representation is tractable when ξ and $|\mathcal{T}|$ are small. Though our theoretical results are applicable to general regression models, we focus on linear models in this paper for computational efficiency. This is not restrictive since nonlinearity can be incorporated through feature engineering.

4.2. Theoretical Properties

4.2.1. Prediction model setup. We start by formally defining the prediction problem that is embedded into our ML-augmented model. For any fixed leader decision $\mathbf{x}$, we are interested in a regression problem defined in a feature space $\mathcal{F} \subseteq \mathbb{R}^\xi$ and a target space $\mathcal{G}_\mathbf{x} \subseteq \mathbb{R}$. We denote by $\eta_\mathbf{x}(\cdot \,|\, \mathbf{f})$ the probability density function of the target variable given

a feature vector $\mathbf{f}$. We regard $G^s(\mathbf{x})$ as a random variable because the true mapping from features to this target may not be deterministic. For example, consider a network design problem where the follower's cost is the length of the shortest path from an origin to a destination using the network designed by the leader. If we use a one-dimensional binary feature that is 1 if both the origin and destination are in downtown and 0 otherwise, then all downtown OD pairs will share the same feature value but with shortest path lengths that could be drastically different.

4.2.2. Assumptions. Next, we introduce several assumptions that enable the derivation of our theoretical results in Sections 4.2.3 and 4.2.4.

ASSUMPTION 1 **(Independence)**. *For any followers $s, s' \in \mathcal{S}, s \neq s'$ and leader decisions $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$, the target (random) variables with distributions $\eta_{\mathbf{x}_1}(\cdot \,|\, \mathbf{f}^s)$ and $\eta_{\mathbf{x}_2}(\cdot \,|\, \mathbf{f}^{s'})$ are independent.*

ASSUMPTION 2 **(Predictivity)**. *There exists a $\bar{G} \in \mathbb{R}_+$ such that, for any fixed $\mathbf{x} \in \mathcal{X}$, $\mathbf{f} \in \mathcal{F}$, and g drawn from $\eta_{\mathbf{x}}(\cdot \,|\, \mathbf{f})$, there exists an interval $[\underline{g}, \bar{g}] \subseteq \mathbb{R}$ such that $\bar{g} - \underline{g} \leq \bar{G}$ and $g \in [\underline{g}, \bar{g}]$ almost surely.*

Assumption 1 holds for a wide range of applications where the original follower set $\mathcal{S}$ is independently sampled. For example, in transportation network design, OD pairs (followers) are usually independently sampled from survey data (Lim et al. 2021) or ridership data (Liu et al. 2021b, 2022). In two-stage stochastic programming, second-stage scenarios are usually from historical observations that can be regarded as independent samples (Shapiro et al. 2009, Birge and Louveaux 2011). Assumption 2 states that the chosen features should be predictive of the follower's cost. Given a fixed feature vector $\mathbf{f}$, if the associated follower cost could vary wildly, it would be difficult for any prediction model to achieve good performance. In contrast, if Assumption 2 holds and $\bar{G}$ is small, then achieving a small prediction error is theoretically possible if the ML model is properly chosen and trained. Note that for Theorems 1 and 2 to hold, Assumption 2 can be relaxed if the cost function g is bounded. However, using predictive features helps to narrow the interval, thus tightening the bounds. Assumptions 1 and 2 enable the application of concentration inequalities to bound the deviation of the total cost of out-of-sample followers from its expected value.

ASSUMPTION 3 **(Continuity of Follower Cost)**. *For any fixed $\mathbf{x} \in \mathcal{X}$, there exists a $\mu \in \mathbb{R}_+$ such that $\mathbb{E}_{G \sim \eta(\cdot \,|\, \mathbf{f})}[G \,|\, \mathbf{f}]$ is μ-Lipschitz continuous with respect to $\mathbf{f}$.*

ASSUMPTION 4 **(Continuity of the Prediction Model)**. *For any fixed* $\mathbf{w} \in \mathcal{W}$*, there exists a* $\lambda \in \mathbb{R}_+$ *such that* $P(\mathbf{f}; \mathbf{w})$ *is* λ*-Lipschitz continuous with respect to* $\mathbf{f}$*.*

Assumption 3 limits the change in the expected follower cost as a function of the change in feature space. Similar assumptions are commonly made to derive stability results for two-stage stochastic programming (Römisch and Schultz 1991, Römisch and Wets 2007, Bertsimas and Mundru 2022). Assumption 4 limits the expressive power of P. Given that P is trained on a small dataset associated with $\mathcal{T}$, limiting its expressive power is critical to avoid overfitting. Such a condition can be enforced by adding regularization constraints to $\mathcal{W}$. For example, for linear regression, we can set $\mathcal{W} = \{\mathbf{w} \in \mathbb{R}^\xi \mid \|w\|_1 \leq \lambda\}$. This assumption is needed only for parametric regression models.

Next, we present theoretical bounds for the quality of solutions from the kNN-augmented (Section 4.2.3) and parametric regression-augmented models (Section 4.2.4), and then follower selection and model tuning methods that tighten the bounds (Section 4.3).

4.2.3. Bound on kNN-augmented model solution.

THEOREM 1. *Given a follower sample* $\mathcal{T} \subseteq \mathcal{S}$ *and a neighborhood size* $k \in \{1, 2, \ldots, |\mathcal{T}|\}$*, let* $\mathbf{x}_{\mathcal{T}}^{kNN}$ *denote an optimal solution to problem* (10)*. If Assumptions 1–3 hold, with probability at least* $1-\gamma$*,*

$$F(\mathbf{x}_{\mathcal{T}}^{kNN}) - F(\mathbf{x}^*) \leq \sum_{s \in \mathcal{S}\setminus\mathcal{T}} \sum_{t \in \mathcal{T}_k(\mathbf{f}^s)} \frac{2\mu\bar{Q}}{k} d_{\mathcal{F}}(\mathbf{f}^s, \mathbf{f}^t) + \sqrt{2\bar{Q}^2\bar{G}^2 \left[|\mathcal{S}\setminus\mathcal{T}| + \sum_{t \in \mathcal{T}} (m_{k,\mathcal{S}\setminus\mathcal{T}}^t / k)^2 \right] \log(1/\gamma)}$$

where $d_{\mathcal{F}}$ *is a distance metric in* $\mathcal{F}$*,* $m_{k,\mathcal{S}\setminus\mathcal{T}}^t = \sum_{s \in \mathcal{S}\setminus\mathcal{T}} \mathbb{1}[t \in \mathcal{T}_k(\mathbf{f}^s)]$*, and* $\bar{Q} = \max_{s \in \mathcal{S}\setminus\mathcal{T}} q_s$*.*

Theorem 1 bounds the optimality gap of the solution generated by the kNN-augmented model on the original problem. The first term on the right-hand side corresponds to the prediction bias and the second term corresponds to the variance and Bayes error, both controlled by k and $\mathcal{T}$. The bias is proportional to the sum of the average distance from each $\mathbf{f}^s$ to its k-nearest neighbors and it increases as k increases. When the sample size $|\mathcal{T}|$ is fixed, the second term is controlled by $m_{k,\mathcal{S}\setminus\mathcal{T}}^t$. Note that $\sum_{t \in \mathcal{T}} m_{k,\mathcal{S}\setminus\mathcal{T}}^t = k|\mathcal{S}\setminus\mathcal{T}|$, so the second term is minimized when $m_{k,\mathcal{S}\setminus\mathcal{T}}^t$ are identical for all $t \in \mathcal{T}$, which follows from the Cauchy-Schwarz inequality. The intuition is that if the followers from $\mathcal{S}\setminus\mathcal{T}$ are "evenly" assigned to sample followers in $\mathcal{T}$, then the overall prediction performance on $\mathcal{S}\setminus\mathcal{T}$ is less affected by the random deviation of the individual cost of follower t, $G^t(\mathbf{x})$, from its expected value. In

the worst case, the second term is bounded by $\bar{Q}\bar{G}\sqrt{2\left(|\mathcal{S}\backslash\mathcal{T}|+|\mathcal{S}\backslash\mathcal{T}|^2/k\right)\log(1/\gamma)}$, which decreases as k increases, reflecting the bias-variance trade-off. Motivated by Theorem 1, we introduce how to select $\mathcal{T}$ and k to tighten the bound in Section 4.3.1.

4.2.4. Bound on parametric regression-augmented model solution.

THEOREM 2. *Given a follower sample $\mathcal{T}\subseteq\mathcal{S}$, let $\mathbf{x}_{\mathcal{T}}^{reg}$ denote the optimal solution to problem* (11) *and $\nu(s)$ denote the nearest neighbor of $\mathbf{f}^s$ in $\{\mathbf{f}^t\}_{t\in\mathcal{T}}$, if Assumptions 1–4 hold, with probability at least $1-\gamma$,*

$$F(\mathbf{x}_{\mathcal{T}}^{reg})-F(\mathbf{x}^*)\leq 2\bar{Q}\bar{L}+2\bar{Q}(\lambda+\mu)\sum_{s\in\mathcal{S}\backslash\mathcal{T}}d_{\mathcal{F}}(\mathbf{f}^s,\mathbf{f}^{\nu(s)})+\sqrt{2\bar{Q}^2\bar{G}^2\left[|\mathcal{S}\backslash\mathcal{T}|+\sum_{t\in\mathcal{T}}(m^t_{1,\mathcal{S}\backslash\mathcal{T}})^2\right]\log(1/\gamma)}$$

Theorem 2 bounds the optimality gap of the leader's decision generated by the parametric regression-augmented model on the original problem. The first term on the right-hand side is controlled by the training loss $\bar{L}$, whereas the second and third terms are controlled by $\mathcal{T}$. To reduce the second and third terms, $\mathcal{T}$ should be chosen such that followers $s\in\mathcal{S}\backslash\mathcal{T}$ are not too far away from its nearest neighbor in $\mathcal{T}$ (second term) and the assignment of followers in $\mathcal{S}\backslash\mathcal{T}$ to followers in $\mathcal{T}$ should be "even" (third term). Informed by Theorem 2, we discuss how to select $\bar{L}$ and $\mathcal{T}$ in Section 4.3.2.

4.3. Practical Implementation

4.3.1. ***k*NN-augmented model.** To tighten the bound in Theorem 1, we need to determine i) the follower sample $\mathcal{T}$, and ii) the value of k. Jointly optimizing $\mathcal{T}$ and k is challenging due to the complicated function form. We propose a practical solution method that enumerates different values of k and samples one follower set for each k by solving

$$\min_{\mathcal{T}\subseteq\mathcal{S}}\left\{\sum_{s\in\mathcal{S}\backslash\mathcal{T}}\sum_{t\in\mathcal{T}_k(\mathbf{f}^s)}d_{\mathcal{F}}(\mathbf{f}^s,\mathbf{f}^t)\,\middle|\,|\mathcal{T}|\leq p\right\}\tag{12}$$

where $p\ll|\mathcal{S}|$ is an upper bound on the sample size imposed by the available computational resources. We choose to focus on bias minimization because i) the bias-minimization problem corresponds to a vector-assignment p-median problem (Weaver and Church 1985), which is well-studied; ii) the variance term is obtained by applying the Hoeffding inequality (Hoeffding 1994) without considering the moments of $\eta_{\mathbf{x}}(\cdot\,|\,\mathbf{f})$, meaning that the bound might be loose; iii) variance-minimization can be implicitly considered by enforcing capacity constraints or penalizing "uneven" assignment in the objective function in problem (12).

In principle, one should enumerate all $k \in [p]$ and obtain a follower sample and a leader's decision by solving problems (12) and (10), respectively, for each k. The best leader's solution can then be selected based on the objective function of problem (2). However, since both problems (10) and (12) are challenging to solve, full enumeration of k introduces considerable computational burden. We thus narrow the search window for k based on the predictive performance of kNN regression on the follower costs under some sampled leader decisions. Specifically, we first randomly generate a set of feasible leader solutions and build a dataset of followers' costs for each one. We then select the best neighborhood size k^* based on the average out-of-sample predictive performance on these datasets. We then vary k in a small interval around k^*, and obtain the corresponding follower samples and leader solutions by solving Problems (12) and (10), respectively. The complete solution approach is presented as Algorithm 1.

Algorithm 1 A solution method using the kNN-augmented model

Input: Width of the search window ω; Number of leader decisions to sample n_d; Follower sample size p; Follower features $\{\mathbf{f}^s\}_{s\in\mathcal{S}}$.

Output: A leader solution $\hat{\mathbf{x}}^{k\text{NN}}$.

Randomly sample n_d feasible leader solutions $\{\mathbf{x}_i \in \mathcal{X}\}_{i=1}^{n_d}$.
for $i = 1$ **to** n_d **do**
    Generate a dataset $\mathcal{D}_i = \{\mathbf{f}^s, G^s(\mathbf{x}_i)\}_{s\in\mathcal{S}}$.
    Perform a random train-test split to obtain $\mathcal{D}_i^{\text{train}}$ and $\mathcal{D}_i^{\text{test}}$ such that $|\mathcal{D}_i^{\text{train}}| = p$.
    for $k = 1$ **to** p **do**
        Build kNN model $P_{i,k}$ using $\mathcal{D}_i^{\text{train}}$.
        Calculate out-of-sample loss $e_{i,k} = \frac{1}{|\mathcal{D}_i^{\text{test}}|} \sum_{(\mathbf{f}^s, G^s(\mathbf{x}_i))\in\mathcal{D}_i^{\text{test}}} |P_{i,k}(\mathbf{f}^s) - G^s(\mathbf{x}_i)|$.
Select the best $k^* \in \arg\min_{k\in[p]} \left\{ \frac{1}{n_d} \sum_{i=1}^{n_d} e_{i,k} \right\}$.
for $k \in \mathcal{K} := \{\max\{1, k^* - \omega\}, \ldots, \min\{p, k^* + \omega\}\}$ **do**
    Obtain leader's solution $\hat{\mathbf{x}}_k$ by solving problems (12) and (10) with k.
Select the best solution $\hat{\mathbf{x}}^{k\text{NN}} \in \arg\min_{\mathbf{x}} \{F(\mathbf{x}) \,|\, \mathbf{x} \in \{\hat{\mathbf{x}}_k\}_{k\in\mathcal{K}}\}$.

4.3.2. General parametric regression. To tighten the bound in Theorem 2, we need to determine i) the follower sample $\mathcal{T}$ and ii) the value of $\bar{L}$. To select $\mathcal{T}$, we focus on

minimizing the second term of the bound for similar reasons as discussed in the previous section. In this case, we solve the classical p-median problem

$$\min_{\mathcal{T}\subseteq\mathcal{S}} \left\{ \sum_{s\in\mathcal{S}\setminus\mathcal{T}} d_{\mathcal{F}}(\mathbf{f}^s, \mathbf{f}^{\tau(s)}) \,\middle|\, |\mathcal{T}| \leq p \right\}. \tag{13}$$

Regarding $\bar{L}$, choosing a small value will tighten the bound, but could lead to overfitting or even worse, render problem (11) infeasible. We propose a practical approach to iteratively search for an appropriate $\bar{L}$. The search starts from a given L_0, which is estimated using data associated with randomly generated leader decisions, and then gradually increases this value until the generated leader decision stops improving. The complete solution approach is presented as Algorithm 2.

Algorithm 2 A solution method using the parametric regression-augmented model

Input: Step size l_{step}; Number of leader decisions to sample n_d; Follower sample size p; Follower features $\{\mathbf{f}^s\}_{s\in\mathcal{S}}$.

Output: A leader solution $\hat{\mathbf{x}}^{\text{reg}}$.

Randomly sample n_d feasible leader solutions $\{\mathbf{x}_i \in \mathcal{X}\}_{i=1}^{n_d}$.
for $i = 1$ **to** n_d **do**
    Generate dataset $\mathcal{D}_i = \{\mathbf{f}^s, G^s(\mathbf{x}_i)\}_{s\in\mathcal{S}}$.
    Randomly select a training set $\mathcal{D}_i^{\text{train}} \subseteq \mathcal{D}_i$ such that $|\mathcal{D}_i^{\text{train}}| = p$.
    Train a prediction model $P_{i,k}\colon \mathbb{R}^\xi \to \mathbb{R}$ on $\mathcal{D}_i^{\text{train}}$.
    Calculate the training loss $e_i = \frac{1}{|\mathcal{D}_i^{\text{train}}|} \sum_{\mathbf{f}^s, G^s(\mathbf{x}_i)\in\mathcal{D}_i^{\text{train}}} |P_{i,k}(\mathbf{f}^s) - G^s(\mathbf{x}_i)|$.
Select a starting point $L_0 = \text{median}\{e_1, e_2, \ldots, e_{n_d}\}$.
Obtain $\mathcal{T}$ by solving Problem (13).
Obtain an initial solution $\mathbf{x}_0$ by solving Problem (11) with L_0 and $\mathcal{T}$.
Initialize step counter $s = 1$
repeat
    Update $L_s = L_{s-1} + l_{\text{step}}$.
    Obtain $\mathbf{x}_s$ by solving Problem (11) with L_s and $\mathcal{T}$.
until $F(\mathbf{x}_s) > F(\mathbf{x}_{s-1})$.
Select the best solution $\hat{\mathbf{x}}^{\text{reg}} = \mathbf{x}_{s-1}$.

5. Learning Follower Representations

In this section, we present a representation learning framework that maps a follower set $\mathcal{S}$ to a ξ-dimensional feature space. The learned follower features are used as inputs to the ML model embeded in problems (10) and (11), and to quantify similarity between followers to aid follower sampling in problems (12) and (13). This framework, as presented in Algorithm 3, consists of two steps. In the first step, we construct a relationship graph $\mathcal{R}$, where each node represents one follower and each edge is assigned a weight reflecting the similarity between the followers. In the second step, we adopt a graph embedding algorithm to map each node in $\mathcal{R}$ to a feature space $\mathcal{F} \subseteq \mathbb{R}^{\xi}$. We emphasize that this framework is compatible with any relationship graph and graph embedding algorithms, providing flexibility to tailor the framework to deal with different applications.

Algorithm 3 An Algorithm for Follower Embedding

Input: Number of leader decisions n_{sim}; Embedding size ξ; Walk per follower n_{walk}; Walk length l_{walk}; Window size w.

Output: Follower features $\{\mathbf{f}^s\}_{s\in\mathcal{S}}$.

Generate n_{sim} leader decisions $\{\mathbf{x}_i \in \mathcal{X}\}_{i=1}^{n_{\text{sim}}}$. ▷ Relationship graph construction
for $i = 1$ **to** n_{sim} **do**
    Calculate $G_i^s := G^s(\mathbf{x}_i)$ for all $s \in \mathcal{S}$.
Construct a graph $\mathcal{R} = (\mathcal{S}, \mathcal{A})$, where $\mathcal{A} = \{(s,t) \,|\, s,t \in \mathcal{S}, s \neq t\}$.
Calculate weight $\pi_{st} = \frac{1}{n_{\text{sim}}} \sum_{i=1}^{n_{\text{sim}}} \Phi(G_i^s, G_i^t)$ for all $(s,t) \in \mathcal{A}$.
Initialize walk container $\mathcal{C}$. ▷ Graph embedding
for $i = 1$ **to** n_{walk} **do**
    for $s \in \mathcal{S}$ **do**
        Initialize the current node $v_{\text{curr}} = s$ and the walk $\mathcal{W}_i^s = [v_{\text{curr}}]$.
        for $j = 1$ **to** l_{walk} **do**
            Sample the next node $v_{\text{next}} \sim Pr(v = t) = \pi_{v_{\text{curr}}t} / (\sum_{s\in\mathcal{S}} \pi_{v_{\text{curr}}s})$.
            Update $\mathcal{W}_i^s \leftarrow [\mathcal{W}_i^s; v_{\text{next}}]$ and $v_{\text{curr}} \leftarrow v_{\text{next}}$.
        Update $\mathcal{C} \leftarrow \mathcal{C} \cup \{\mathcal{W}_i^s\}$.
Learn follower features $\{\mathbf{f}^s\}_{s\in\mathcal{S}} \leftarrow Word2Vec(\mathcal{C}, w)$

5.1. Relationship Graph Construction

The core of the relationship graph construction is defining a weight for each edge that measures the similarity between followers. Many metrics have been proposed in the stochastic programming literature, with most focusing on the "opportunity cost" of switching between scenarios (i.e., followers). For example, Keutchayan et al. (2021) and Hewitt et al. (2021) define the opportunity cost of applying the first-stage decision that is optimal to scenario t in scenario s as $d(s,t) = G^s(\mathbf{x}^{t*}) - G^s(\mathbf{x}^{s*})$ where $\mathbf{x}^{s*}$ and $\mathbf{x}^{t*}$ denote the optimal first-stage solutions obtained by solving the single-scenario version of problem (2) with scenarios s and t, respectively. Building on a similar idea, Bertsimas and Mundru (2022) define a symmetric divergence measure for scenarios s and t as $(d(s,t) + d(t,s))/2$. While these opportunity cost-based measures have been shown to be effective and are compatible with the proposed framework, they require solving $|\mathcal{S}|$ single-scenario versions of problem (2) in advance, which is computationally expensive when $\mathcal{S}$ is huge.

Motivated by the fact that evaluating followers' costs given a leader's solution is computationally cheaper, we propose a data-driven approach that quantifies follower similarity based on their costs under some sampled leader solutions. Specifically, we first randomly sample n_{sim} leader decisions $\{\mathbf{x}_i \in \mathcal{X}\}_{i=1}^{n_{\text{sim}}}$. For each sampled $\mathbf{x}_i$, we calculate the costs $G_i^s := G^s(\mathbf{x}_i)$ for all followers $s \in \mathcal{S}$ by solving the associated follower problems. We then define the weight of the edge between followers $s, t \in \mathcal{S}$ based on the distance between $(G_1^s, \ldots, G_{n_{\text{sim}}}^s)$ and $(G_1^t, \ldots, G_{n_{\text{sim}}}^t)$. Among the distance metrics we experimented with, the following definition of edge weight achieves the best performance in predicting followers' costs. Specifically, we define the weight of edge (s,t) in the relationship graph as

$$\pi_{st} = \frac{1}{n_{\text{sim}}} \sum_{i=1}^{n_{\text{sim}}} \Phi(G_i^s, G_i^t), \quad \forall s \neq t \in \mathcal{S},$$

where $\Phi : \mathbb{R} \times \mathbb{R} \to [0,1]$ is a user-selected function whose target space is restricted to $[0,1]$ to avoid π_{st} being dominated by a single sampled follower's solution. For instance, letting ϵ be a small positive constant, we use $\Phi(G_i^s, G_i^t) = \min\{G_i^s/(G_i^t + \epsilon),\, G_i^t/(G_i^s + \epsilon)\}$ in our cycling network design problem where the followers' costs are non-negative. For applications where the followers' costs can be negative, functions such as $\Phi(G_i^s, G_i^t) = \exp(-|G_i^s - G_i^t|)$ might be considered. The value of π_{st} can be interpreted as the average cost similarity between followers s and t under the sampled leader's decisions. Intuitively, considering more leader decisions provides more accurate estimates of such a relationship.

5.2. Follower Embedding

Once $\mathcal{R}$ is constructed, we adapt the DeepWalk algorithm proposed by Perozzi et al. (2014) to map nodes in the relationship graph to a ξ-dimensional feature space. The idea is to first generate a set of random walks in the relationship graph, and then apply the Word2Vec algorithm (Mikolov et al. 2013) to learn node features treating each node and each random walk as a word and a sentence, respectively. Unlike Perozzi et al. (2014), who generate random walks by uniformly sampling nodes connected to the current node, we generate random walks according to the weights assigned to edges incident to the current node. So, followers that yield similar results under the sampled leader decisions are likely to appear in a same walk, and thus will be positioned close to each other in the learned feature space.

6. Computational Study: Algorithm Performance on Synthetic Cycling Network Design Problem

In this section, we validate the effectiveness of our ML-augmented model with our representation learning framework for a cycling network design problem. We introduce the problem and its formulation in Section 6.1. We present two experiments to validate the predictive power of the learned follower features and the value of integrating an ML model in the optimization problem in Sections 6.2 and 6.3, respectively. We use a moderate-sized synthetic network (described in EC.4.1) that can be solved to optimality, which enables us to accurately measure the performance of our approach.

6.1. Maximum Accessibility Network Design Problem

The goal of the maximum accessibility network design problem (MaxANDP) is to design a cycling network subject to a fixed budget such that the total accessibility of a given set of OD pairs, denoted by $\mathcal{S}$, is maximized. Such a set may be defined based on geographical units (Imani et al. 2019, Lim et al. 2021) or ridership data (Liu et al. 2021b). Existing studies have proposed various metrics to measure accessibility, mostly focusing on first finding one or more routes between each OD pair using the designed network and then calculating the accessibility based on the selected routes. Such measures have been shown to be correlated with travel behavior data (Saghapour et al. 2017, Imani et al. 2019) and have been widely adopted to assess the performance of cycling networks (Vale et al. 2016).

Let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ be a directed graph where $\mathcal{E}$ is the set of edges and $\mathcal{N}$ is the set of nodes, corresponding to road segments and intersections, respectively. Each edge $(i, j) \in \mathcal{E}$

is assigned a travel time t_{ij}. We denote by $\mathcal{E}^+(i)$ and $\mathcal{E}^-(i)$ the sets of incoming and outgoing edges of node i, respectively. Edges and nodes are partitioned into high-stress and low-stress sets according to a cycling stress assessment based on road geometry, existing infrastructure, and vehicle traffic conditions (Landis et al. 1997, Harkey et al. 1998, Furth et al. 2016). We assume that cyclists prefer cycling on low-stress roads over high-stress roads. Sets with subscripts h and l indicate the high-stress and low-stress subsets of the original set, respectively. High-stress edges $(i,j) \in \mathcal{E}_h$ and nodes $i \in \mathcal{N}_h$ are assigned costs c_{ij} and b_i, respectively, corresponding to the costs of turning them into low-stress through building new infrastructure such as cycle tracks or traffic lights.

Let $\mathbf{x} \in \{0,1\}^{|\mathcal{E}_h|}$ and $\mathbf{z} \in \{0,1\}^{|\mathcal{N}_h|}$, respectively, denote the *edge selection* and *node selection* variables (referred to as *network design* decisions), whose components are 1 if that edge or node is chosen for the installation of infrastructure that makes it low stress. Edge and node selections are subject to budgets B_{edge} and B_{node}, respectively. Let $\mathbf{y}^{od} \in \{0,1\}^{|\mathcal{E}|}$ denote the *routing decision* associated with OD pair $(o,d) \in \mathcal{S}$. The routing problem on a network specified by $\mathbf{x}$ and $\mathbf{z}$ is characterized by an objective function $h^{od}(\mathbf{x},\mathbf{z},\cdot) : \{0,1\}^{|\mathcal{E}|} \to \mathbb{R}$ and a feasible set $\mathcal{Y}^{od}(\mathbf{x},\mathbf{z}) \subseteq \{0,1\}^{|\mathcal{E}|}$. A function $g(\mathbf{x},\mathbf{z},\cdot) : \{0,1\}^{|\mathcal{E}|} \to \mathbb{R}_+$ is used to calculate the accessibility of each OD pair based on the selected route(s). Each OD pair is weighted by a constant $q^{od} \in \mathbb{R}_+$ (e.g., population). The MaxANDP is formulated as

$$\underset{\mathbf{x},\mathbf{z},\mathbf{y}^{od}}{\text{maximize}} \quad \sum_{(o,d)\in\mathcal{S}} q^{od} g(\mathbf{x},\mathbf{z},\mathbf{y}^{od}) \tag{14a}$$

$$\text{subject to} \quad \mathbf{y}^{od} \in \underset{\mathbf{y}\in\mathcal{Y}^{od}(\mathbf{x},\mathbf{z})}{\arg\min}\ h^{od}(\mathbf{x},\mathbf{z},\mathbf{y}), \quad (o,d) \in \mathcal{S} \tag{14b}$$

$$\mathbf{c}^\intercal \mathbf{x} \le B_{\text{edge}} \tag{14c}$$

$$\mathbf{b}^\intercal \mathbf{z} \le B_{\text{node}} \tag{14d}$$

$$\mathbf{x} \in \{0,1\}^{|\mathcal{E}_h|}, \mathbf{z} \in \{0,1\}^{|\mathcal{N}_h|}, \tag{14e}$$

where $\mathbf{c}$ and $\mathbf{b}$ indicate cost vectors for high-stress edges and nodes, respectively. The objective function (14a) maximizes total cycling accessibility. Constraints (14b) ensure that the selected routes are optimal for the OD pairs' objective functions. Constraints (14c) and (14d) enforce budgets on the network design decisions. Constraints (14e) describe the domain of the decision variables.

To apply problem (14), the accessibility measure (specified by g) and the routing problem (specified by h^{od} and $\mathcal{Y}^{od}$) should be carefully chosen based on recent travel behavior data

in the studied area (Geurs and Van Wee 2004). To illustrate our methodology, we consider two problems: i) one that uses location-based accessibility measures and shortest-path routing problems, and ii) one proposed by Liu et al. (2021b) that employs a utility-based accessibility measure and discrete route choice models. We refer readers to Liu et al. (2021b) for the latter problem. We briefly describe the former problem next.

Location-based accessibility measures use a decreasing function of the travel time from origin to destination, namely an impedance function, to model the dampening effect of separation (Iacono et al. 2010). We consider a piecewise linear impedance function

$$g(\mathbf{y}^{od}) = \begin{cases} 1 - \beta_1 \mathbf{t}^\intercal \mathbf{y}^{od}, & \text{if } \mathbf{t}^\intercal \mathbf{y}^{od} \in [0, T_1) \\ 1 - \beta_1 T_1 - \beta_2 \left(\mathbf{t}^\intercal \mathbf{y}^{od} - T_1\right), & \text{if } \mathbf{t}^\intercal \mathbf{y}^{od} \in [T_1, T_2) \\ 0, & \text{if } \mathbf{t}^\intercal \mathbf{y}^{od} \geq T_2, \end{cases} \tag{15}$$

where $\mathbf{t}$ indicates a vector of edge travel times, $T_1, T_2 \in \mathbb{R}_+$ are breakpoints, and $\beta_1, \beta_2 \in \mathbb{R}_+$ are penalty factors for $[0, T_1)$ and $[T_1, T_2)$, respectively. This function can be used to approximate commonly used impedance functions, including negative exponential, rectangular, and linear functions (visualized in EC.4.2). While we consider two breakpoints for simplicity, the formulation can be easily generalized to account for more.

We use the level of traffic stress (LTS) metric (Furth et al. 2016) to formulate the routing problems. Let $\mathbf{A}$ be the node-edge matrix describing the flow-balance constraints on $\mathcal{G}$, and $\mathbf{e}^{od}$ be a vector whose o^{th} and d^{th} entries are 1 and -1, respectively, with all other entries 0. Given network design $(\mathbf{x}, \mathbf{z})$, the routing problem for $(o, d) \in \mathcal{S}$ is formulated as

$$\underset{\mathbf{y}^{od} \in \{0,1\}^{|\mathcal{E}|}}{\text{minimize}} \quad \mathbf{t}^\intercal \mathbf{y}^{od} \tag{16a}$$

$$\text{subject to} \quad \mathbf{A}\mathbf{y}^{od} = \mathbf{e}^{od} \tag{16b}$$

$$y_{ij}^{od} \leq x_{ij}, \quad \forall (i,j) \in \mathcal{E}_h \tag{16c}$$

$$y_{ij}^{od} \leq x_{wl} + z_i, \quad \forall i \in \mathcal{N}_h,\ (i,j) \in \mathcal{E}_h^-(i),\ (w,l) \in \mathcal{E}_h^-(i) \cup \mathcal{E}_h^+(i). \tag{16d}$$

Objective function (16a) minimizes the travel time. Constraints (16b) direct one unit of flow from o to d. Constraints (16c) ensure that a currently high-stress edge can be used only if it is selected. Constraints (16d) guarantee that a currently high-stress node can be crossed only if either the node is selected or all high-stress edges that are connected to this node are selected. Constraints (16d) are an exact representation of the intersection LTS

calculation scheme that assigns the low-stress label to a node if traffic signals are installed or all incident roads are low-stress (Imani et al. 2019). To ensure problem (16) is feasible, we add a dummy low-stress link from o to d and set its travel time to T_2. In doing so, the travel time is set to T_2 when the destination is unreachable using the low-stress network, corresponding to an accessibility of zero, as defined in equation (15). The full formulation is given in EC.2. We adapt the Benders approach from Magnanti et al. (1986) to solve synthetic instances to optimality. The algorithm and its acceleration strategies is in EC.3.

6.2. Experiment 1: Predicting OD-Pair Accessibility Using ML Models

In this experiment, we randomly generate network designs and calculate the accessibility for each OD pair under each design. The accessibility associated with each network design constitutes a dataset on which we perform train-test data splits, train ML models to predict OD-pair accessibility, and evaluate their out-of-sample prediction performance. Our goal is to i) compare the predictive power of our learned features and baseline features, ii) validate the effectiveness of our follower sampling method in improving the prediction accuracy, and iii) compare the prediction performance of online and offline ML methods.

Data generation. We consider three edge design budgets of 100, 300, and 500 distance units, corresponding to three levels of ambition, and generate 1,000 network designs for each. We set the node design budget to zero for computational tractability. We consider three location-based accessibility measures using piecewise linear impedance functions that mimic negative exponential (EXP), linear (LIN), and rectangular (REC) functions. We also consider the utility-based accessibility measure (UT) proposed by Liu et al. (2021b). The accessibility calculations are detailed in EC.4.2. In total, we generate 12,000 datasets (1,000 for each pair of design budget and accessibility measure).

Predictive power of the learned features. Since accessibility is a function of the travel time from origin to destination, which can be regarded as the optimal objective value of a two-stop traveling salesman problem (TSP), we employ the features proposed by Liu et al. (2021a) for predicting TSP objective value as our baseline features. For each dataset, we perform a random train-test split and then train two ML models using the same training sample, but with different OD-pair features. One uses the TSP features, while the other uses the features derived from our representation learning framework (REP). We consider ML models that are compatible with our ML-augmented model, including kNN,

linear regression, lasso, and ridge regression. We vary the training sample size between 1%–5% of all OD pairs because practical implementation of our approach would be based on sampling a small number of OD pairs only. For each pair of sample size and ML model, we report the median test MAE over ten random train-test splits. The details on the baseline features and ML hyper-parameters are given in EC.4.3 and EC.4.7, respectively.

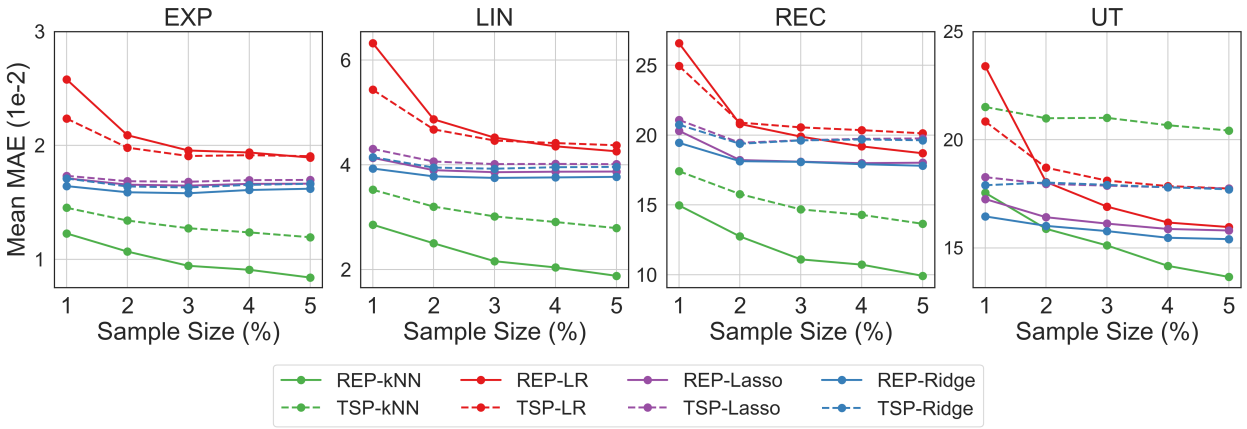

Figure 1 Mean out-of-sample MAE over 3,000 datasets of each accessibility measure when using different ML models and follower features. ML models are coded by colors and features are coded by line types.

We observe that ML models generally perform better with the REP features than with the TSP features. As presented in Figure 1, when using kNN, the REP features outperform the TSP features by a large margin (e.g. over 22.8%, 26.1%, 22.5%, and 25.7% for EXP, LIN, REC, and UT, respectively). However, linear regression performs better with the TSP features. One possible explanation is that linear regression models based on the REP features (16-dimensional) are more complicated than those based on the TSP features (9-dimensional), and thus are more likely to overfit the training data. When using lasso and ridge, the REP features are better than or competitive with the TSP features. Lasso and ridge also achieve lower prediction errors than linear regression, highlighting the importance of enforcing regularization constraints in the ML-augmented model (recall discussion following Assumption 4).

Effectiveness of the follower selection algorithm. Next, we use the REP features while applying different follower sampling methods, and compare the resulting test MAE achieved by the ML models. We consider i) p-median sampling as introduced in Section 4.3 (MED), 2) uniform sampling (UNI), which is commonly used in the transportation

literature, and 3) p-center sampling (CEN) from Sener and Savarese (2018). Since the p-center and p-median problems are $\mathcal{NP}$-hard, we adapt heuristics from Boutilier and Chan (2020) and Gonzalez (1985) to solve them (detailed in EC.4.4). As a result, all methods involve some randomness. To account for the randomness, we apply each sampling method 10 times with different random seeds and report the median test MAE on each dataset.

From Figure 2, we observe that MED typically achieves the lowest test MAE, regardless of the accessibility measures, ML models, and sample sizes. Especially when the sample size is extremely small, the gap between MED and UNI can be up to 27% of the MAE achieved by UNI. MED outperforms CEN by a large margin when using location-based accessibility measures and is competitive with CEN when using the utility-based measure.

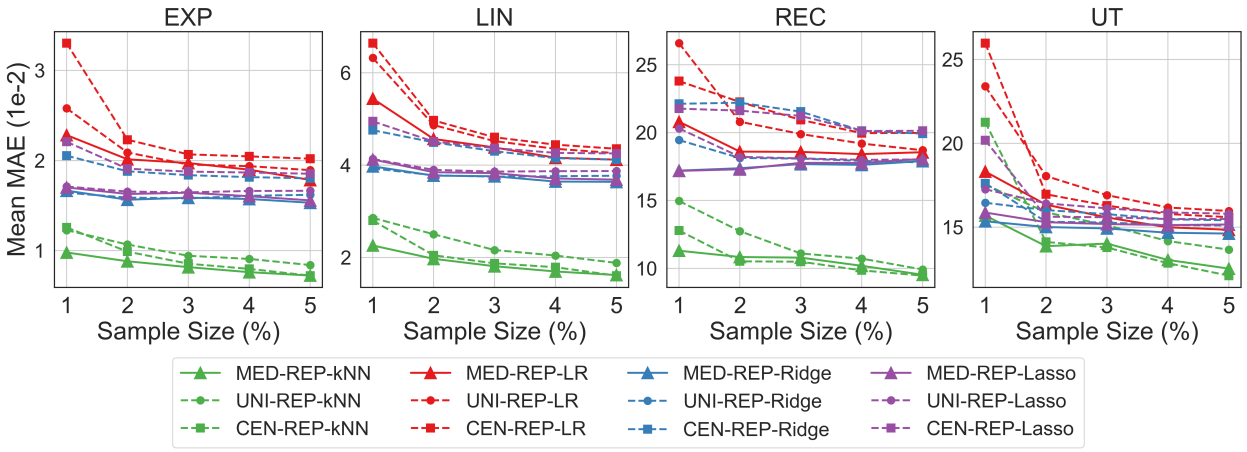

Figure 2 Mean test MAE over 3000 datasets associated with each accessibility measure when using different ML models and follow sampling methods. ML models are coded by colors and sampling methods are coded by markers. Our sampling method (MED) is further highlighted using solid lines, while baseline sampling methods are represented by dashed lines

Online ML model versus offline ML model. We also compare the approximations for the overall leader's objective achieved by the best online ML approach (train a kNN model using REP features and MED samples for each leader decision) versus an offline ReLU neural network (ReLU) (Dumouchelle et al. 2022) that predicts the total accessibility of all OD pairs directly based on leader decisions (detailed in EC.4.6). As presented in Figure 3, our online ML model provides better approximations using only 1–2% of all OD pairs as training data. These results provide an empirical justification for integrating ML model training into the optimization problem.

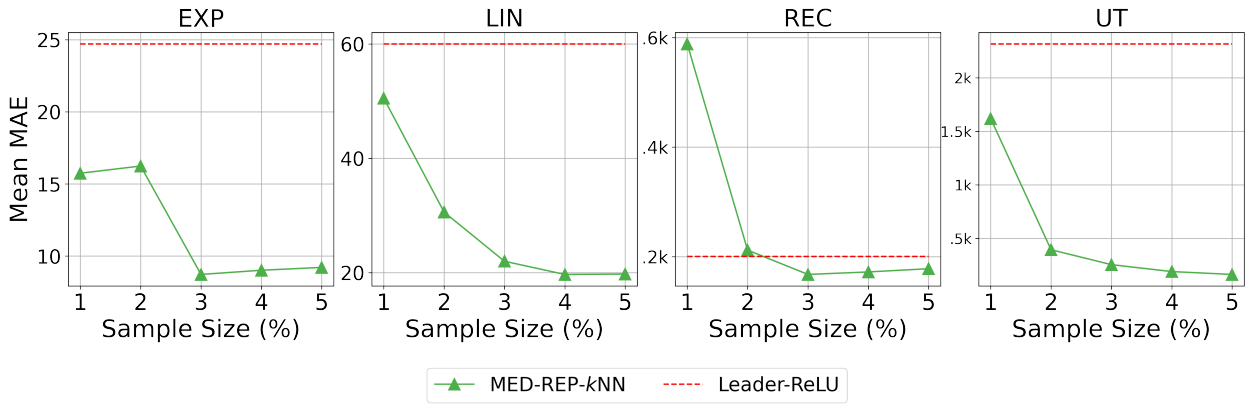

Figure 3 Mean MAE for predicting the leader's overall objective over 3000 datasets associated with each accessibility measure. "Leader-ReLU" indicates the performance of an offline ML model, "MED-REP-kNN" indicates the performance of the best online ML model.

Summary. i) our learned features are predictive of accessibility and are generally more predictive than baseline features. ii) By applying follower sampling methods derived from solving a p-median problem, we can substantially improve the out-of-sample prediction accuracy of the ML models with our learned features. iii) the best online ML model yields a better approximation for the original objective than an offline ML model.

6.3. Experiment 2: Generating Leader Decisions using ML-augmented Models

Next, we investigate the extent to which our learned features and our follower samples can assist the ML-augmented model in generating high-quality leader (i.e., transportation planner) decisions. We consider the reduced model, kNN-augmented model, and linear regression-augmented model using MED and UNI samples, totaling six methods for generating leader decisions. Results for CEN samples are in EC.4.9, as they are similar to UNI samples. We create 12 problem instances on the synthetic network (one for each pair of design budget and accessibility measure). We vary the sample size from 1% to 5%. We apply each model 10 times using 10 samples generated with different random seeds and report the average optimality gap of the leader decisions on the original problem.

From Figure 4, our first observation is that using MED samples enhances the performance of both the ML-augmented models and the reduced model. Significant performance gaps are observed for the two ML-augmented models in all problem instances considered. Using the MED samples on average reduces the optimality gap by 5.7% and 5.2% for the kNN-augmented and linear regression-augmented models, respectively. Even for the

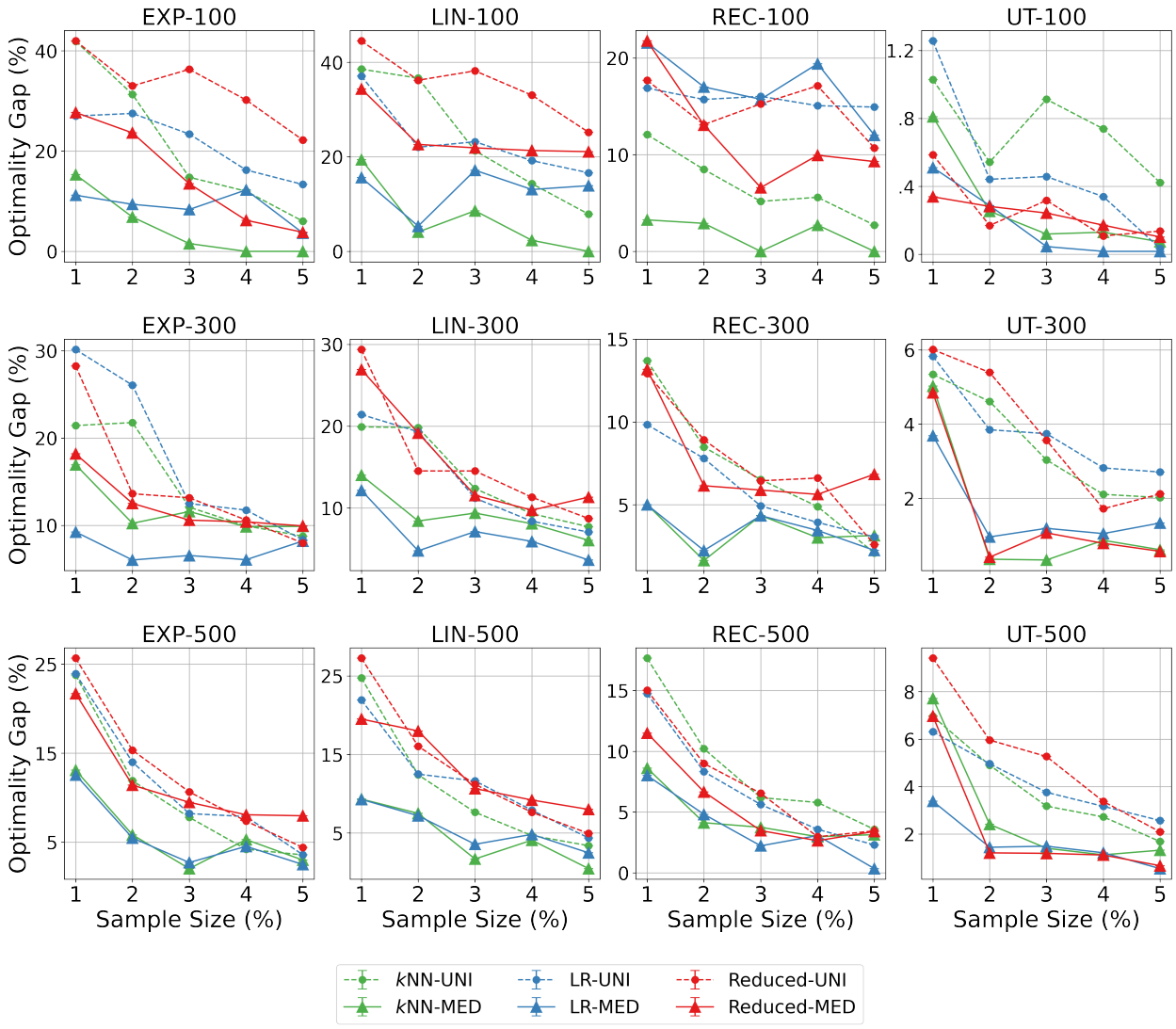

Figure 4 Optimality gap of leader decisions from the three sampling-based models on the 12 problem instances. Problem instances follow the naming convention of "accessibility measure"-"budget" and the solution methods follow the naming convention of "model"-"sampling method".

reduced model, our sampling strategy is competitive with or better than uniform sampling, with a difference in optimality gap of up to over 20% (e.g., EXP-100, 3% training sample). These results highlight the importance of sample selection for both models.

Our second observation is that when using the MED samples, the ML-augmented models generally achieve better and more stable performance compared to the reduced model. As shown in Figure 4, the ML-augmented model (kNN-MED or REG-MED) generally outperforms Reduced-MED by a large margin, especially when using location-based accessibility measures and when the sample size is extremely small (1%). Figure 5 compares

the stability of Reduced-MED and kNN-MED. kNN-MED generally achieves a lower standard deviation of optimality gaps as Reduced-MED, even though they use exactly the same samples. The integrated ML component helps to better capture the impact of leader decisions on unsampled followers, leading to solutions of higher quality and stability.

Finally, we observe no dominance of a single ML-augmented model. REG-MED and kNN-MED are comparable on mid- and large-budget instances (300 and 500). kNN-MED performs better when using location-based accessibility measures and a small budget (100). One possible reason is that when the budget is limited, few OD pairs have positive accessibility while all others have zero accessibility. The feature-accessibility relationship is highly non-linear, making kNN a better fit for the accessibility prediction than linear regressions.

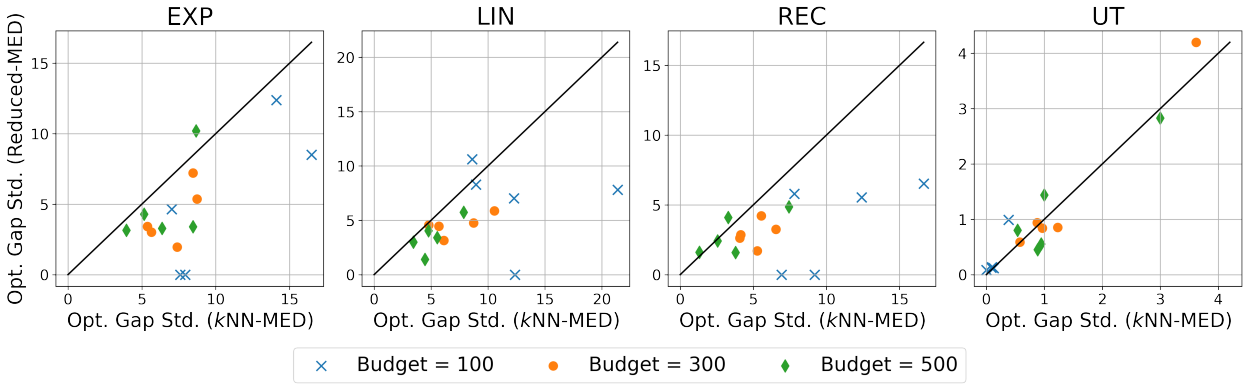

Figure 5 The standard deviation of the optimality gaps achieved by Reduced-MED versus kNN-MED. Results from instances with different design budgets are coded by colors and markers. The black diagonal lines indicate equal standard deviations.

7. Case Study: Cycling Infrastructure Planning in the City of Toronto

In this section, we present a case study applying our methodology to the City of Toronto, Canada. Toronto has built over 65 km of new cycling infrastructure from 2019–2021, partially in response to the increased cycling demand amid the COVID-19 pandemic. It plans to expand the network by 100 km from 2022–2024. We started a collaboration with the City's Transportation Services Team in September 2020, focusing on developing quantitative tools to support cycling infrastructure planning in Toronto. As an evaluation metric, low-stress cycling accessibility has been used by the City of Toronto to support project prioritization (City of Toronto 2021a,b). We introduce Toronto's cycling network in Section

7.1 and use our methodology to examine actual and future potential decisions regarding network expansion in Section 7.2.

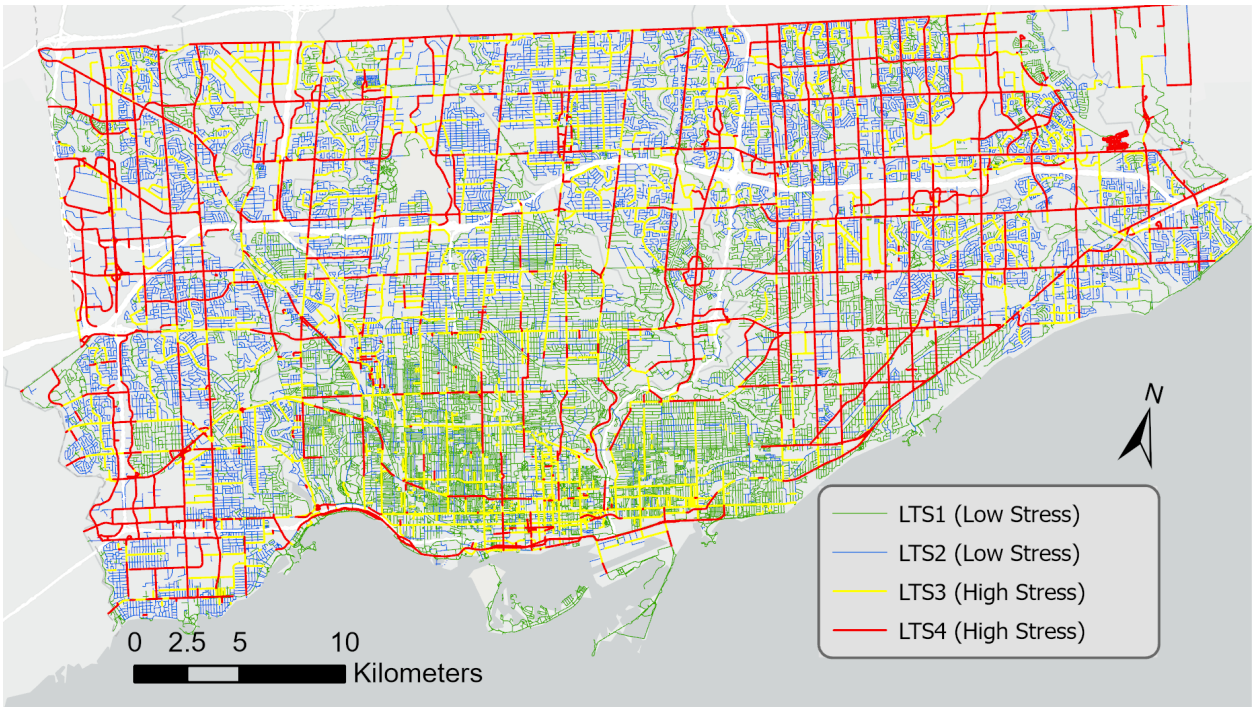

Figure 6 Level of traffic stress of Toronto's road network (July 2021).

7.1. Cycling Network in Toronto

We construct Toronto's cycling network based on the centerline network retrieved from the Toronto Open Data Portal (City of Toronto 2020). We pre-process the network by removing roads where cycling is legally prohibited, deleting redundant nodes and edges, and grouping arterial roads into candidate cycling infrastructure projects (detailed in EC.5.1). The final cycling network has 10,448 nodes, 35,686 edges, and 1,296 candidate projects totaling 1,913 km. We use the methods and data sources summarized in Lin et al. (2021) to calculate the LTS of each link in the cycling network. LTS1 and LTS2 links are classified as low-stress, while LTS3 and LTS4 links are high-stress since LTS2 corresponds to the cycling stress tolerance for the majority of the adult population (Furth et al. 2016). Although most local roads are low-stress, high-stress arterials create many disconnected low-stress "islands", limiting low-stress cycling accessibility in many parts of Toronto (Figure 6).

The city is divided into 3,702 geographical units called dissemination areas (DAs). We define each DA centroid as an origin with every other DA centroid that is reachable within 30 minutes on the overall network being a potential destination, totalling 1,154,663 OD pairs. The low-stress cycling accessibility of a given DA is defined as the sum of the destination opportunities that are reachable within 30 minutes using the low-stress network. We use DA job counts retrieved from the 2016 Canadian census (Statistics Canada 2016) to represent opportunities. The resulting accessibility measure has been shown to be highly correlated with cycling mode choice in Toronto (Imani et al. 2019). We assume a constant cycling speed of 15 km/h for travel time calculation.

7.2. Expanding Toronto's Cycling Network

As a part of our collaboration, in January 2021 we were asked to evaluate the accessibility impact of three project alternatives for building bike lanes (see Figure 7) to meet the direction of Toronto's City Council within the City's adopted Cycling Network Plan, intended to provide a cycling connection between midtown and the downtown core (City of Toronto 2021b). These projects were proposed in 2019 but their evaluation and implementation were accelerated because of increased cycling demand during COVID. We determined that alternative 2 had the largest accessibility impact. It was ultimately implemented due to its accessibility impact and other performance indicators (City of Toronto 2021b).

This decision-making process exemplifies the current practice of cycling infrastructure planning in Toronto: i) manually compile a list of candidate projects, ii) rank the candidate projects based on certain metrics, and iii) design project delivery plans (City of Toronto 2021c). From a computational perspective, steps i) and ii) serve as a heuristic for solving MaxANDP. This heuristic approach was necessary for several reasons, including political buy-in for the three alternatives, and the computational intractability of solving MaxANDP at the city scale prior to developing the methodology in this paper. Now, we can use our ML-augmented model to search for project combinations without pre-specifying candidates.

To this end, we first apply the ML-augmented model with a budget of 6 km (similar to alternative 2). The optimal projects (see Figure 7) improve Toronto's total low-stress cycling accessibility by approximately 9.5% over alternative 2. Instead of constructing only one corridor as in alternative 2, the ML-augmented model selects six disconnected road segments. Some of them serve as connections between existing cycling infrastructure, others bridge currently disconnected low-stress sub-networks consisting of low-stress local roads.

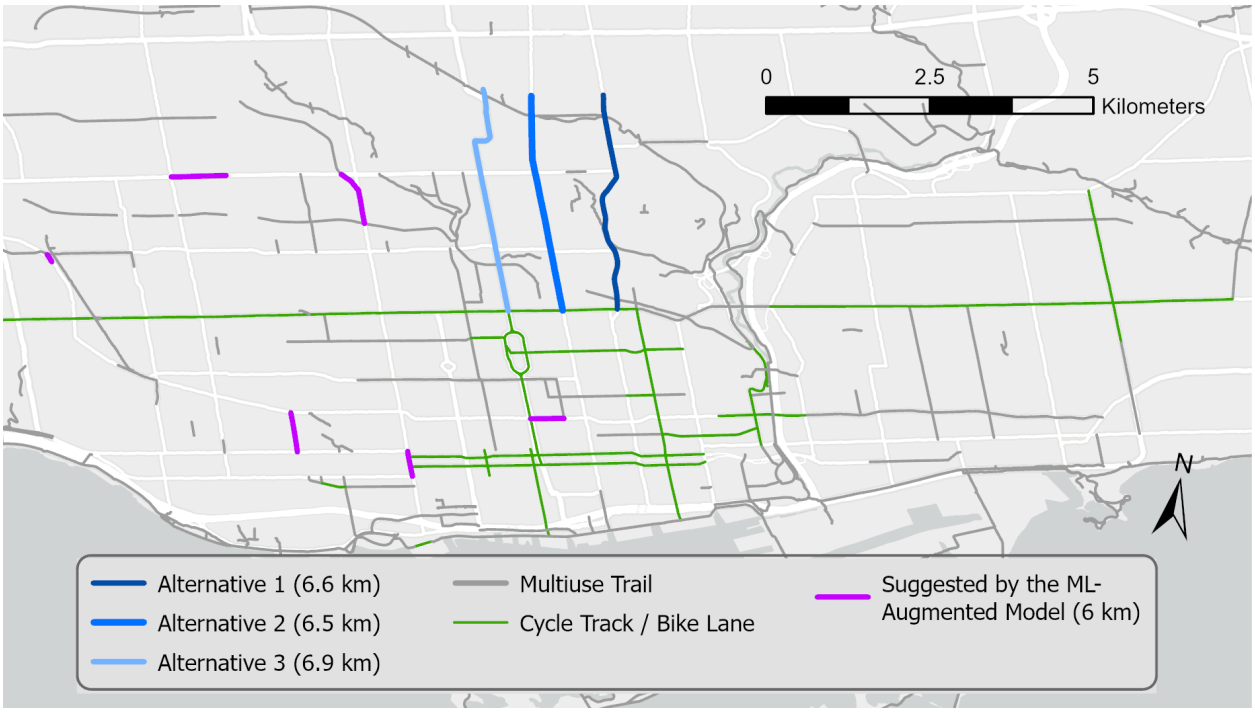

Figure 7 **Project alternatives and the existing cycling infrastructure in the City of Toronto (January 2021).**

Next, we increase the road design budget from 10 to 100 km in increments of 10 km. The 100 km budget aligns with Toronto's cycling network expansion plan for 2022–2024 (City of Toronto 2021a). We compare our approach to a greedy heuristic that iteratively selects the candidate project that leads to the maximum increase in total accessibility until the budget is depleted. The greedy heuristic took over 3 days to expand the network by 100 km as each iteration involves solving millions of shortest path problems. Our approach took around 4 hours to find a leader decision using a sample of 2,000 OD pairs (1.7% of all OD pairs). Given this speedup, we can solve our model multiple times with different samples and report the best solution as measured by the total accessibility of all OD pairs. The computational setups of the greedy heuristic and our approach are detailed in EC.5.3.

As shown in Figure 8, when holding both methods to the same computational time (meaning that we solve our ML-augmented model with 21 different sets of OD pair samples and taking the best solution), our approach increases accessibility by 19.2% on average across different budgets. For example, with a budget of 70 km, we can improve the total accessibility by a similar margin as achieved by the greedy heuristic using a 100 km budget, corresponding to a savings of 18 million Canadian dollars estimated based on the City's

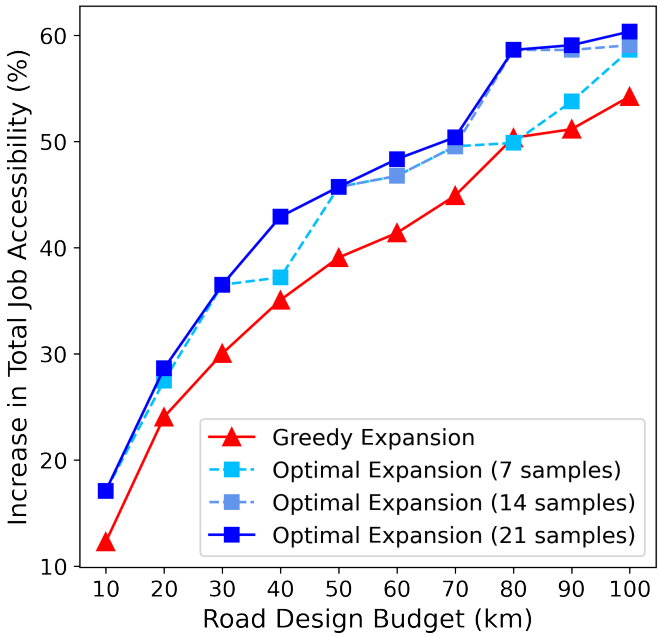

Figure 8 **The performance profiles of the greedy and optimal expansions. Note that using 21 different sets of OD pairs samples results in the same solution time as the greedy expansion. Hence, 7 and 14 samples correspond to $1/3$ and $2/3$ of the solution time of greedy.**

proposed budget (City of Toronto 2021a). If instead we used the full 100 km budget, we would achieve 11.3% greater accessibility. The improvements mainly come from identifying projects that have little accessibility impact when constructed alone, yet significantly improve the accessibility of their surrounding DAs when combined (visualized in EC.5.4). These projects are typically not directly connected to existing cycling infrastructure, and thus are difficult to identify through manual analysis. Finally, we note that solution quality was similar between 14 and 21 samples, meaning that with we can achieve the above gains while simultaneously reducing solution time by approximately 33%.

In summary, our approach is a valuable tool for transportation planners to search for optimal project combinations that maximize the low-stress cycling accessibility. Although this is not the only goal of cycling network design, we believe it can be useful in at least three contexts: i) in the long term, our model can be used to generate a base plan that can later be tuned by transportation planners; ii) in the near term, our approach can efficiently search for project combinations from a large pool that would be very difficult to analyze manually; iii) Given a fixed budget, our model provides a strong benchmark against which to validate the goodness of human-proposed solutions.

8. Conclusion

In this paper, we present a novel ML-based approach to solving bilevel (stochastic) programs with a large number of independent followers (scenarios). We build on two existing strategies—sampling and approximation—to tackle the computational challenges imposed by a large follower set. The model considers a sampled subset of followers while integrating an ML model to estimate the impact of leader decisions on unsampled followers. Unlike existing approaches for integrating optimization and ML models, we embed the ML model training into the optimization model, which allows us to employ general follower features that may not be compactly represented by leader decisions. Under certain assumptions, the generated leader decisions enjoy solution quality guarantees as measured by the original objective function considering the full follower set. We also introduce practical strategies, including follower sampling algorithms and a representation learning framework, to enhance the model performance. Using both synthetic and real-world instances of a cycling network design problem, we demonstrate the strong computational performance of our approach in generating high-quality leader decisions. The performance gap between our approach and baseline approaches are particularly large when the sample size is small.

Acknowledgments

The authors are grateful to Sheng Liu, Merve Bodur, Elias Khalil, and Rafid Mahmood for helpful comments and discussions. This research is supported by funding from the City of Toronto and NSERC Alliance Grant 561212-20. Resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute.

References

Alizadeh S, Marcotte P, Savard G (2013) Two-stage stochastic bilevel programming over a transportation network. *Transportation Research Part B: Methodological* 58:92–105.

Babier A, Chan TCY, Diamant A, Mahmood R (2022) Learning to optimize contextually constrained problems for real-time decision-generation. *arXiv preprint arXiv:1805.09293* .

Bagloee SA, Sarvi M, Wallace M (2016) Bicycle lane priority: Promoting bicycle as a green mode even in congested urban area. *Transportation Research Part A: Policy and Practice* 87:102–121.

Ban GY, Rudin C (2019) The big data newsvendor: Practical insights from machine learning. *Operations Research* 67(1):90–108.

Bao J, He T, Ruan S, Li Y, Zheng Y (2017) Planning bike lanes based on sharing-bikes' trajectories. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1377–1386.

Bard JF (2013) *Practical bilevel optimization: algorithms and applications*, volume 30 (Springer Science & Business Media).

Bello I, Pham H, Le QV, Norouzi M, Bengio S (2017) Neural combinatorial optimization with reinforcement learning. *International Conference on Learning Representations Workshop*.

Bergman D, Huang T, Brooks P, Lodi A, Raghunathan AU (2022) Janos: an integrated predictive and prescriptive modeling framework. *INFORMS Journal on Computing* 34(2):807–816.

Bertsimas D, Kallus N (2020) From predictive to prescriptive analytics. *Management Science* 66(3):1025–1044.

Bertsimas D, Mundru N (2022) Optimization-based scenario reduction for data-driven two-stage stochastic optimization. *Operations Research* 0(0).

Biggs M, Hariss R, Perakis G (2017) Optimizing objective functions determined from random forests, available at SSRN: https://ssrn.com/abstract=2986630.

Birge JR, Louveaux F (2011) *Introduction to stochastic programming* (Springer Science & Business Media).

Bodur M, Luedtke JR (2017) Mixed-integer rounding enhanced benders decomposition for multiclass service-system staffing and scheduling with arrival rate uncertainty. *Management Science* 63(7):2073–2091.

Boutilier JJ, Chan TCY (2020) Ambulance emergency response optimization in developing countries. *Operations Research* 68(5):1315–1334.

Buehler R, Dill J (2016) Bikeway networks: A review of effects on cycling. *Transport Reviews* 36(1):9–27.

Buehler R, Pucher J (2021) COVID-19 impacts on cycling, 2019–2020. *Transport Reviews* 41(4):393–400.

Candler W, Townsley R (1982) A linear two-level programming problem. *Computers & Operations Research* 9(1):59–76.

Carrión M, Arroyo JM, Conejo AJ (2009) A bilevel stochastic programming approach for retailer futures market trading. *IEEE Transactions on Power Systems* 24(3):1446–1456.

Chen X, Sim M, Sun P, Zhang J (2008) A linear decision-based approximation approach to stochastic programming. *Operations Research* 56(2):344–357.

City of Toronto (2020) City of Toronto open data. https://www.toronto.ca/city-government/data-research-maps/open-data/, accessed: 2020-09-15.

City of Toronto (2021a) 2021 cycling network plan update. Accessed via https://www.toronto.ca/legdocs/mmis/2021/ie/bgrd/backgroundfile-173663.pdf on July 8, 2022.

City of Toronto (2021b) ActiveTO: Lessons learned from 2020 and next steps for 2021. Accessed via https://www.toronto.ca/legdocs/mmis/2021/ie/bgrd/backgroundfile-164864.pdf on July 21, 2022.

City of Toronto (2021c) Cycling network plan update — external stakeholders briefing summary. Accessed via https://www.toronto.ca/wp-content/uploads/2021/06/8ea2-External-Briefing-Meeting-Summary-June-7-2021.pdf on July 21, 2022.

Cleveland WS, Devlin SJ (1988) Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American Statistical Association* 83(403):596–610.

Crainic TG, Hewitt M, Rei W (2014) Scenario grouping in a progressive hedging-based meta-heuristic for stochastic network design. *Computers & Operations Research* 43:90–99.

Dill J, McNeil N (2016) Revisiting the four types of cyclists: Findings from a national survey. *Transportation Research Record* 2587(1):90–99.

Dumouchelle J, Patel R, Khalil E, Bodur M (2022) Neur2sp: Neural two-stage stochastic programming. *arXiv preprint arXiv.2205.12006* .

Dupačová J, Gröwe-Kuska N, Römisch W (2003) Scenario reduction in stochastic programming. *Mathematical Programming* 95(3):493–511.

Duthie J, Unnikrishnan A (2014) Optimization framework for bicycle network design. *Journal of Transportation Engineering* 140(7):04014028.

Elmachtoub AN, Grigas P (2022) Smart "predict, then optimize". *Management Science* 68(1):9–26.

Fampa M, Barroso L, Candal D, Simonetti L (2008) Bilevel optimization applied to strategic pricing in competitive electricity markets. *Computational Optimization and Applications* 39(2):121–142.

Ferreira KJ, Lee BHA, Simchi-Levi D (2016) Analytics for an online retailer: Demand forecasting and price optimization. *Manufacturing & Service Operations Management* 18(1):69–88.

Fischetti M, Ljubić I, Sinnl M (2017) Redesigning benders decomposition for large-scale facility location. *Management Science* 63(7):2146–2162.

Furth PG, Mekuria MC, Nixon H (2016) Network connectivity for low-stress bicycling. *Transportation Research Record* 2587(1):41–49.

Furth PG, Putta TV, Moser P (2018) Measuring low-stress connectivity in terms of bike-accessible jobs and potential bike-to-work trips. *Journal of Transport and Land Use* 11(1):815–831.

Gehrke SR, Akhavan A, Furth PG, Wang Q, Reardon TG (2020) A cycling-focused accessibility tool to support regional bike network connectivity. *Transportation Research Part D: Transport and Environment* 85:102388.

Geurs KT, Van Wee B (2004) Accessibility evaluation of land-use and transport strategies: review and research directions. *Journal of Transport Geography* 12(2):127–140.

Gonzalez TF (1985) Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science* 38:293–306.

Gurobi Optimization, LLC (2022) Gurobi Optimizer Reference Manual. URL https://www.gurobi.com.

Harkey DL, Reinfurt DW, Knuiman M (1998) Development of the bicycle compatibility index. *Transportation Research Record* 1636(1):13–20.

Hewitt M, Ortmann J, Rei W (2021) Decision-based scenario clustering for decision-making under uncertainty. *Annals of Operations Research* 1–25.

Hochbaum DS, Shmoys DB (1985) A best possible heuristic for the k-center problem. *Mathematics of Operations Research* 10(2):180–184.

Hoeffding W (1994) Probability inequalities for sums of bounded random variables. *The Collected Works of Wassily Hoeffding*, 409–426 (Springer).

Iacono M, Krizek KJ, El-Geneidy A (2010) Measuring non-motorized accessibility: issues, alternatives, and execution. *Journal of Transport Geography* 18(1):133–140.

Imani AF, Miller EJ, Saxe S (2019) Cycle accessibility and level of traffic stress: A case study of Toronto. *Journal of Transport Geography* 80:102496.

Jia H, Shen S (2021) Benders cut classification via support vector machines for solving two-stage stochastic programs. *INFORMS Journal on Optimization* 3(3):278–297.

Kent M, Karner A (2019) Prioritizing low-stress and equitable bicycle networks using neighborhood-based accessibility measures. *International Journal of Sustainable Transportation* 13(2):100–110.

Keutchayan J, Ortmann J, Rei W (2021) Problem-driven scenario clustering in stochastic optimization. *arXiv preprint arXiv:2106.11717* .

Khalil E, Dai H, Zhang Y, Dilkina B, Song L (2017) Learning combinatorial optimization algorithms over graphs. *Advances in Neural Information Processing Systems*, 6348–6358.

Khalil E, Le Bodic P, Song L, Nemhauser G, Dilkina B (2016) Learning to branch in mixed integer programming. *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Kool W, van Hoof H, Welling M (2019) Attention, learn to solve routing problems! *7th International Conference on Learning Representations, ICLR*, URL https://openreview.net/forum?id=ByxBFsRqYm.

Kou Z, Wang X, Chiu SFA, Cai H (2020) Quantifying greenhouse gas emissions reduction from bike share systems: a model considering real-world trips and transportation mode choice patterns. *Resources, Conservation and Recycling* 153:104534.

Kraus S, Koch N (2021) Provisional COVID-19 infrastructure induces large, rapid increases in cycling. *Proceedings of the National Academy of Sciences* 118(15).

Landis BW, Vattikuti VR, Brannick MT (1997) Real-time human perceptions: toward a bicycle level of service. *Transportation Research Record* 1578(1):119–126.

Larsen J, Patterson Z, El-Geneidy A (2013) Build it. but where? the use of geographic information systems in identifying locations for new cycling infrastructure. *International Journal of Sustainable Transportation* 7(4):299–317.

Leal M, Ponce D, Puerto J (2020) Portfolio problems with two levels decision-makers: Optimal portfolio selection with pricing decisions on transaction costs. *European Journal of Operational Research* 284(2):712–727.

Lewis DD, Gale WA (1994) A sequential algorithm for training text classifiers. *SIGIR'94*, 3–12.

Li S, Muresan M, Fu L (2017) Cycling in Toronto, Ontario, Canada: Route choice behavior and implications for infrastructure planning. *Transportation Research Record* 2662(1):41–49.

Lim J, Dalmeijer K, Guhathakurta S, Van Hentenryck P (2021) The bicycle network improvement problem: Optimization algorithms and a case study in Atlanta. *arXiv preprint arXiv:2107.04451* .

Lin B, Chan TCY, Saxe S (2021) The impact of COVID-19 cycling infrastructure on low-stress cycling accessibility: A case study in the City of Toronto. *Findings* 19069.

Liu H, Szeto W, Long J (2019) Bike network design problem with a path-size logit-based equilibrium constraint: Formulation, global optimization, and matheuristic. *Transportation Research Part E: Logistics and Transportation Review* 127:284–307.

Liu S, He L, Shen ZJM (2021a) On-time last-mile delivery: Order assignment with travel-time predictors. *Management Science* 67(7):4095–4119.

Liu S, Shen ZJM, Ji X (2021b) Urban bike lane planning with bike trajectories: Models, algorithms, and a real-world case study. *Manufacturing & Service Operations Management* 0(0).

Liu S, Siddiq A, Zhang J (2022) Planning bike lanes with data: Ridership, congestion, and path selection, available at SSRN: https://ssrn.com/abstract=4055703.

Lowry MB, Callister D, Gresham M, Moore B (2012) Assessment of communitywide bikeability with bicycle level of service. *Transportation Research Record* 2314(1):41–48.

Magnanti TL, Mireault P, Wong RT (1986) Tailoring benders decomposition for uncapacitated network design. *Netflow at Pisa*, 112–154 (Springer).

Mauttone A, Mercadante G, Rabaza M, Toledo F (2017) Bicycle network design: model and solution algorithm. *Transportation Research Procedia* 27:969–976.

Mesbah M, Thompson R, Moridpour S (2012) Bilevel optimization approach to design of network of bike lanes. *Transportation Research Record* 2284(1):21–28.

Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, volume 26.

Mišić VV (2020) Optimization of tree ensembles. *Operations Research* 68(5):1605–1624.

Morabit M, Desaulniers G, Lodi A (2021) Machine-learning–based column selection for column generation. *Transportation Science* 55(4):815–831.

Mueller N, Rojas-Rueda D, Salmon M, Martinez D, Ambros A, Brand C, De Nazelle A, Dons E, Gaupp-Berghausen M, Gerike R, et al. (2018) Health impact assessment of cycling network expansions in european cities. *Preventive Medicine* 109:62–70.

Naoum-Sawaya J, Elhedhli S (2013) An interior-point benders based branch-and-cut algorithm for mixed integer programs. *Annals of Operations Research* 210(1):33–55.

Nazari M, Oroojlooy A, Snyder L, Takác M (2018) Reinforcement learning for solving the vehicle routing problem. *Advances in Neural Information Processing Systems*, 9839–9849.

Notz PM, Pibernik R (2022) Prescriptive analytics for flexible capacity management. *Management Science* 68(3):1756–1775.

Olmos LE, Tadeo MS, Vlachogiannis D, Alhasoun F, Alegre XE, Ochoa C, Targa F, González MC (2020) A data science framework for planning the growth of bicycle infrastructures. *Transportation Research Part C: Emerging Technologies* 115:102640.

Ospina JP, Duque JC, Botero-Fernández V, Montoya A (2022) The maximal covering bicycle network design problem. *Transportation Research part A: Policy and Practice* 159:222–236.

Papadakos N (2008) Practical enhancements to the Magnanti-Wong method. *Operations Research Letters* 36(4):444–449.

Parzen E (1962) On estimation of a probability density function and mode. *The Annals of Mathematical Statistics* 33(3):1065–1076.

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12:2825–2830.

Perozzi B, Al-Rfou R, Skiena S (2014) Deepwalk: Online learning of social representations. *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 701–710.

Prochazka V, Wallace SW (2020) Scenario tree construction driven by heuristic solutions of the optimization problem. *Computational Management Science* 17(2):277–307.

Putta T, Furth PG (2019) Method to identify and visualize barriers in a low-stress bike network. *Transportation Research Record* 2673(9):452–460.

Řehůřek R, Sojka P (2010) Software Framework for Topic Modelling with Large Corpora. *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, 45–50.

Römisch W, Schultz R (1991) Stability analysis for stochastic programs. *Annals of Operations Research* 30(1):241–266.

Römisch W, Wets RB (2007) Stability of ε-approximate solutions to convex stochastic programs. *SIAM Journal on Optimization* 18(3):961–979.

Saghapour T, Moridpour S, Thompson RG (2017) Measuring cycling accessibility in metropolitan areas. *International Journal of Sustainable Transportation* 11(5):381–394.

Sener O, Savarese S (2018) Active learning for convolutional neural networks: A core-set approach. *International Conference on Learning Representations*.

Settles B (2009) Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

Settles B, Craven M, Ray S (2007) Multiple-instance active learning. *Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems*, 1289–1296.

Shapiro A, Dentcheva D, Ruszczynski A (2009) *Lectures on stochastic programming: modeling and theory* (SIAM).

Sisson SB, Lee SM, Burns EK, Tudor-Locke C (2006) Suitability of commuting by bicycle to Arizona elementary schools. *American Journal of Health Promotion* 20(3):210–213.

Sohn K (2011) Multi-objective optimization of a road diet network design. *Transportation Research Part A: Policy and Practice* 45(6):499–511.

Statistics Canada (2016) Population and dwelling count, 2016 census. https://www12.statcan.gc.ca/census-recensement/2016/dp-pd/hlt-fst/pd-pl/Table.cfm?Lang=Eng&T=1902&PR=35&S=3&O=D&RPP=50, accessed: 2020-11-15.

Tang Y, Agrawal S, Faenza Y (2020) Reinforcement learning for integer programming: Learning to cut. *International Conference on Machine Learning*, 9367–9376 (PMLR).

Vale DS, Saraiva M, Pereira M (2016) Active accessibility: A review of operational measures of walking and cycling accessibility. *Journal of Transport and Land Use* 9(1):209–235.

Van Hoesel S (2008) An overview of stackelberg pricing in networks. *European Journal of Operational Research* 189(3):1393–1402.

Vinyals O, Fortunato M, Jaitly N (2015) Pointer networks. *Advances in Neural Information Processing Systems*, 2692–2700.

Weaver JR, Church RL (1985) A median location model with nonclosest facility service. *Transportation Science* 19(1):58–74.

White DJ, Anandalingam G (1993) A penalty function approach for solving bi-level linear programs. *Journal of Global Optimization* 3(4):397–419.

Wu Y, Song W, Cao Z, Zhang J (2022) Learning scenario representation for solving two-stage stochastic integer programs. *International Conference on Learning Representations*.

Zetina CA, Contreras I, Cordeau JF (2019) Exact algorithms based on benders decomposition for multicommodity uncapacitated fixed-charge network design. *Computers & Operations Research* 111:311–324.

Zugno M, Morales JM, Pinson P, Madsen H (2013) A bilevel model for electricity retailers' participation in a demand response market environment. *Energy Economics* 36:182–197.

Electronic Companion

EC.1. Omitted Proof of Statements

Proof of Theorem 1 We start by decomposing the optimality gap as follows.

$$
\begin{aligned}
&F(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - F(\mathbf{x}^*) \\
=&\left[F(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - F_{\mathcal{T}}^{k\mathrm{NN}}(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - F(\mathbf{x}^*) + F_{\mathcal{T}}^{k\mathrm{NN}}(\mathbf{x}^*)\right] + \left[F_{\mathcal{T}}^{k\mathrm{NN}}(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - F_{\mathcal{T}}^{k\mathrm{NN}}(\mathbf{x}^*)\right] \\
\le& F(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - F_{\mathcal{T}}^{k\mathrm{NN}}(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - F(\mathbf{x}^*) + F_{\mathcal{T}}^{k\mathrm{NN}}(\mathbf{x}^*) \\
=& \underbrace{\sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s \mathbb{E}[G^s(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}})] - \sum_{s\in\mathcal{S}\setminus\mathcal{T}}\sum_{t\in\mathcal{T}_k(\mathbf{f}^s)} \frac{q^s}{k}\mathbb{E}[G^t(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}})] - \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s \mathbb{E}[G^s(\mathbf{x}^*)] + \sum_{s\in\mathcal{S}\setminus\mathcal{T}}\sum_{t\in\mathcal{T}_k(\mathbf{f}^s)} \frac{q^s}{k}\mathbb{E}[G^t(\mathbf{x}^*)]}_{(1)} \\
&\underbrace{\sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s G^s(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - \sum_{s\in\mathcal{S}\setminus\mathcal{T}}\sum_{t\in\mathcal{T}_k(\mathbf{f}^s)} \frac{q^s}{k}G^t(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s \mathbb{E}[G^s(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}})] + \sum_{s\in\mathcal{S}\setminus\mathcal{T}}\sum_{t\in\mathcal{T}_k(\mathbf{f}^s)} \frac{q^s}{k}\mathbb{E}[G^t(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}})]}_{(2)} \\
&\underbrace{- \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s G^s(\mathbf{x}^*) + \sum_{s\in\mathcal{S}\setminus\mathcal{T}}\sum_{t\in\mathcal{T}_k(\mathbf{f}^s)} \frac{q^s}{k}G^t(\mathbf{x}^*) + \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s \mathbb{E}[G^s(\mathbf{x}^*)] - \sum_{s\in\mathcal{S}\setminus\mathcal{T}}\sum_{t\in\mathcal{T}_k(\mathbf{f}^s)} \frac{q^s}{k}\mathbb{E}[G^t(\mathbf{x}^*)]}_{(3)}
\end{aligned}
$$

The third line holds because $\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}$ is the optimal solution to problem (10).

According to Assumption 3, we have

$$
\begin{aligned}
(1) &\le \sum_{s\in\mathcal{S}\setminus\mathcal{T}}\sum_{t\in\mathcal{T}_k(\mathbf{f}^s)} \frac{q^s}{k}\left\{\left|\mathbb{E}[G^s(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}})] - \mathbb{E}[G^t(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}})]\right| + \left|\mathbb{E}[G^s(\mathbf{x}^*)] - \mathbb{E}[G^t(\mathbf{x}^*)]\right|\right\} \\
&\le \sum_{s\in\mathcal{S}\setminus\mathcal{T}}\sum_{t\in\mathcal{T}_k(\mathbf{f}^s)} \frac{2\mu\bar{Q}}{k} d_{\mathcal{F}}(\mathbf{f}^s, \mathbf{f}^t)
\end{aligned}
$$

Let $\bar{Q} = \max_{s\in\mathcal{S}} q_s$ and $m_{k,\mathcal{S}\setminus\mathcal{T}}^t$ denote the number of followers in $\mathcal{S}\setminus\mathcal{T}$ whose k-nearest neighbors in $\mathcal{T}$ include follower t, we have

$$
\begin{aligned}
(2)+(3) = \bar{Q}\Bigg[&\sum_{s\in\mathcal{S}\setminus\mathcal{T}}\left[G^s(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - G^s(\mathbf{x}^*)\right] - \sum_{t\in\mathcal{T}}\frac{m_{k,\mathcal{S}\setminus\mathcal{T}}^t}{k}\left[G^t(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - G^t(\mathbf{x}^*)\right] \\
&- \sum_{s\in\mathcal{S}\setminus\mathcal{T}}\mathbb{E}[G^s(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - G^s(\mathbf{x}^*)] + \sum_{t\in\mathcal{T}}\frac{m_{k,\mathcal{S}\setminus\mathcal{T}}^t}{k}\mathbb{E}[G^t(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - G^t(\mathbf{x}^*)]\Bigg]
\end{aligned}
$$

According to the Assumptions 1 and 2, we can regard $\left[G^s(\mathbf{x}_{\mathcal{T}}^{k\mathrm{NN}}) - G^s(\mathbf{x}^*)\right]$ as independent random variables that are bounded in an interval of length $2\bar{G}$ almost surely. Similarly,

independent random variables $-m^t_{k,\mathcal{S}\setminus\mathcal{T}}\left[G^t(\mathbf{x}^{k\text{NN}}_{\mathcal{T}}) - G^t(\mathbf{x}^*)\right]/k$ are bounded in an interval of length $2m^t_{k,\mathcal{S}\setminus\mathcal{T}}\bar{G}/k$ almost surely. By applying Hoeffding inequality (Hoeffding 1994), we have, with probability at least $1-\gamma$,

$$(2)+(3) \leq \sqrt{2\bar{Q}^2\bar{G}^2\left[|\mathcal{S}\setminus\mathcal{T}| + \sum_{t\in\mathcal{T}}(m^t_{k,\mathcal{S}\setminus\mathcal{T}}/k)^2\right]\log(1/\gamma)}$$

We therefore conclude that with probability at least $1-\gamma$,

$$F(\mathbf{x}^{k\text{NN}}_{\mathcal{T}}) - F(\mathbf{x}^*) \leq \sum_{s\in\mathcal{S}\setminus\mathcal{T}}\sum_{t\in\mathcal{T}_k(\mathbf{f}^s)}\frac{2\mu\bar{Q}}{k}d_{\mathcal{F}}(\mathbf{f}^s,\mathbf{f}^t) + \sqrt{2\bar{Q}^2\bar{G}^2\left[|\mathcal{S}\setminus\mathcal{T}| + \sum_{t\in\mathcal{T}}(m^t_{k,\mathcal{S}\setminus\mathcal{T}}/k)^2\right]\log(1/\gamma)}$$

□

Proof of Theorem 2 We start by decomposing the optimality gap as follows.

$$\begin{aligned}
F(\mathbf{x}^{reg}_{\mathcal{T}}) - F(\mathbf{x}^*) =& [F(\mathbf{x}^{reg}_{\mathcal{T}}) - F^{reg}_{\mathcal{T}}(\mathbf{x}^{reg}_{\mathcal{T}}, \mathbf{w}^{reg}_{\mathcal{T}}) - F(\mathbf{x}^*) + F^{reg}_{\mathcal{T}}(\mathbf{x}^*, \mathbf{w}^*)] + [F^{reg}_{\mathcal{T}}(\mathbf{x}^{reg}_{\mathcal{T}}, \mathbf{w}^{reg}_{\mathcal{T}}) - F^{reg}_{\mathcal{T}}(\mathbf{x}^*, \mathbf{w}^*)] \\
\leq& F(\mathbf{x}^{reg}_{\mathcal{T}}) - F^{reg}_{\mathcal{T}}(\mathbf{x}^{reg}_{\mathcal{T}}, \mathbf{w}^{reg}_{\mathcal{T}}) - F(\mathbf{x}^*) + F^{reg}_{\mathcal{T}}(\mathbf{x}^*, \mathbf{w}^*) \\
\leq& \underbrace{\sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s\mathbb{E}[G^s(\mathbf{x}^{reg}_{\mathcal{T}})] - \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s\mathbb{E}[G^{\nu(s)}(\mathbf{x}^{reg}_{\mathcal{T}})] - \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s\mathbb{E}[G^s(\mathbf{x}^*)] + \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s\mathbb{E}[G^{\nu(s)}(\mathbf{x}^*)]}_{(1)} \\
&\underbrace{+ \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s P(\mathbf{f}^{\nu(s)}; \mathbf{w}^{reg}_{\mathcal{T}}) - \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s P(\mathbf{f}^s; \mathbf{w}^{reg}_{\mathcal{T}}) - \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s P(\mathbf{f}^{\nu(s)}; \mathbf{w}^*) + \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s P(\mathbf{f}^s; \mathbf{w}^*)}_{(2)} \\
&\underbrace{+ \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s G^{\nu(s)}(\mathbf{x}^{reg}_{\mathcal{T}}) - \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s P(\mathbf{f}^{\nu(s)}; \mathbf{w}^{reg}_{\mathcal{T}}) - \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s G^{\nu(s)}(\mathbf{x}^*) + \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s P(\mathbf{f}^{\nu(s)}; \mathbf{w}^*)}_{(3)} \\
&\underbrace{+ \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s G^s(\mathbf{x}^{reg}_{\mathcal{T}}) - \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s\mathbb{E}[G^s(\mathbf{x}^{reg}_{\mathcal{T}})] + \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s\mathbb{E}[G^{\nu(s)}(\mathbf{x}^{reg}_{\mathcal{T}})] - \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s G^{\nu(s)}(\mathbf{x}^{reg}_{\mathcal{T}})}_{(4)} \\
&\underbrace{- \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s G^s(\mathbf{x}^*) + \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s\mathbb{E}[G^s(\mathbf{x}^*)] - \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s\mathbb{E}[G^{\nu(s)}(\mathbf{x}^*)] + \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s G^{\nu(s)}(\mathbf{x}^*)}_{(5)}
\end{aligned}$$

According to Assumption 3, we have

$$(1) \leq \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s\left\{\left|\mathbb{E}[G^s(\mathbf{x}^{reg}_{\mathcal{T}})] - \mathbb{E}[G^{\nu(s)}(\mathbf{x}^{reg}_{\mathcal{T}})]\right| + \left|\mathbb{E}[G^s(\mathbf{x}^*)] - \mathbb{E}[G^{\nu(s)}(\mathbf{x}^*)]\right|\right\} \leq 2\mu\bar{Q}\sum_{s\in\mathcal{S}\setminus\mathcal{T}} d_{\mathcal{F}}(\mathbf{f}^s, \mathbf{f}^{\nu(s)})$$

According to the Lipschitz Property of P, we have

$$(2) \leq \sum_{s\in\mathcal{S}\setminus\mathcal{T}} q^s \left\{ \left| P(\mathbf{f}^{\nu(s)}; \mathbf{w}_{\mathcal{T}}^{reg}) - P(\mathbf{f}^s; \mathbf{w}_{\mathcal{T}}^{reg}) \right| + \left| P(\mathbf{f}^{\nu(s)}; \mathbf{w}^*) - P(\mathbf{f}^s; \mathbf{w}^*) \right| \right\} \leq 2\lambda\bar{Q} \sum_{s\in\mathcal{S}\setminus\mathcal{T}} d_{\mathcal{F}}(\mathbf{f}^s, \mathbf{f}^{\nu(s)})$$

Since the training loss of P is bounded by $\bar{L}$, we have

$$(3) \leq \bar{Q} \left\{ \sum_{t\in\mathcal{T}} m^t_{1,\mathcal{S}\setminus\mathcal{T}} \left| P(\mathbf{f}^t; \mathbf{w}_{\mathcal{T}}^{reg}) - G^t(\mathbf{x}_{\mathcal{T}}^{reg}) \right| + \sum_{t\in\mathcal{T}} m^t_{1,\mathcal{S}\setminus\mathcal{T}} \left| P(\mathbf{f}^t; \mathbf{w}^*) - G^t(\mathbf{x}^*) \right| \right\} \leq 2\bar{Q}\bar{L}$$

We have

$$\begin{aligned}(4)+(5) = \bar{Q} \Bigg\{ & \sum_{s\in\mathcal{S}} \left[G^s(\mathbf{x}_{\mathcal{T}}^{reg}) - G^s(\mathbf{x}^*) \right] - \sum_{t\in\mathcal{T}} m^t_{1,\mathcal{S}\setminus\mathcal{T}} \left[G^t(\mathbf{x}_{\mathcal{T}}^{reg}) - G^t(\mathbf{x}^*) \right] \\ & - \sum_{s\in\mathcal{S}} \mathbb{E}\left[G^s(\mathbf{x}_{\mathcal{T}}^{reg}) - G^s(\mathbf{x}^*) \right] + \sum_{t\in\mathcal{T}} m^t_{1,\mathcal{S}\setminus\mathcal{T}} \mathbb{E}\left[G^t(\mathbf{x}_{\mathcal{T}}^{reg}) - G^t(\mathbf{x}^*) \right] \Bigg\}\end{aligned}$$

According to the Assumptions 1 and 2, we can regard $\left[G^s(\mathbf{x}_{\mathcal{T}}^{knn}) - G(\mathbf{x}^*) \right]$ as independent random variables that are bounded in an interval of length $2\bar{G}$ almost surely. Similarly, independent random variables $-m^t_{1,\mathcal{S}\setminus\mathcal{T}} \left[G^t(\mathbf{x}_{\mathcal{T}}^{knn}) - G^t(\mathbf{x}^*) \right]$ are bounded in an interval of length $2m^t_{1,\mathcal{S}\setminus\mathcal{T}}\bar{G}$ almost surely. By applying Hoeffding inequality (Hoeffding 1994), we have, with probability at least $1-\gamma$,

$$(4)+(5) \leq \sqrt{2\bar{Q}^2\bar{G}^2 \left[|\mathcal{S}\setminus\mathcal{T}| + \sum_{t\in\mathcal{T}} (m^t_{1,\mathcal{S}\setminus\mathcal{T}})^2 \right] \log(1/\gamma)}$$

We therefore conclude that with probability at least $1-\gamma$,

$$F(\mathbf{x}_{\mathcal{T}}^{reg}) - F(\mathbf{x}^*) \leq 2\bar{Q}\bar{L} + 2\bar{Q}(\lambda+\mu) \sum_{s\in\mathcal{S}\setminus\mathcal{T}} d_{\mathcal{F}}(\mathbf{f}^s, \mathbf{f}^{\nu(s)}) + \sqrt{2\bar{Q}^2\bar{G}^2 \left[|\mathcal{S}\setminus\mathcal{T}| + \sum_{t\in\mathcal{T}} (m^t_{1,\mathcal{S}\setminus\mathcal{T}})^2 \right] \log(1/\gamma)}$$

□

EC.2. Formulation of MaxANDP using Location-Based Accessibility Measures

In this section, we present the full formulation of MaxANDP, as presented in (14), using a piecewise linear impedance function g and shortest-path routing problems as introduced in Section 6.1.

Since g is a decreasing function of travel time and the objectives of the followers are to minimize travel time, the objectives of the leader and followers are aligned. Therefore,

the optimality conditions (14b) can be replaced with the routing constraints (16b)–(16d), resulting in a single-level formulation:

$$\underset{\mathbf{x},\mathbf{y},\mathbf{z}}{\text{maximize}} \quad \sum_{(o,d)\in\mathcal{S}} q^{od} g(\mathbf{y}^{od}) \tag{EC.1a}$$

$$\text{subject to} \quad \mathbf{c}^\intercal \mathbf{x} \le B_{\text{edge}} \tag{EC.1b}$$

$$\mathbf{b}^\intercal \mathbf{z} \le B_{\text{node}} \tag{EC.1c}$$

$$\mathbf{A}\mathbf{y}^{od} = \mathbf{e}^{od}, \quad \forall (o,d) \in \mathcal{S} \tag{EC.1d}$$

$$y_{ij}^{od} \le x_{ij}, \quad \forall (i,j) \in \mathcal{E}_h,\ (o,d) \in \mathcal{S} \tag{EC.1e}$$

$$y_{ij}^{od} \le x_{wl} + z_i, \quad \forall i \in \mathcal{N}_h,\ (i,j) \in \mathcal{E}_h^-(i),\ (w,l) \in \mathcal{E}_h(i),\ (o,d) \in \mathcal{S} \tag{EC.1f}$$

$$0 \le y_{ij}^{od} \le 1, \quad \forall (i,j) \in \mathcal{E}\ (o,d) \in \mathcal{S} \tag{EC.1g}$$

$$\mathbf{x} \in \{0,1\}^{|\mathcal{E}_h|} \tag{EC.1h}$$

$$\mathbf{z} \in \{0,1\}^{|\mathcal{N}_h|}. \tag{EC.1i}$$

It is known that the parameter matrix of the flow balance constraints (EC.1d) is totally unimodular. Moreover, for any fixed feasible $\mathbf{x}$ and $\mathbf{z}$, each row in the parameter matrix of constraints (EC.1e)–(EC.1g) has one "1" with all other entries being zero. Therefore, for any fixed $\mathbf{x}$ and $\mathbf{z}$, the parameter matrix of constraints (EC.1d)–(EC.1g) is totally unimodular. We thus can discard the integrality constraints on $\mathbf{y}^{od}$ and treat them as continuous decision variables bounded in $[0,1]$. The only thing left is to linearize function g, which depends on the values of β_1 and β_2.

Concave impedance function. When $\beta_1 \le \beta_2$, g can be treated as a concave function of travel time $\mathbf{t}^\intercal \mathbf{y}^{od}$ on $[0, T_2]$. We introduce continuous decision variables $v^{od} \in \mathbb{R}_+$, for any OD pairs $(o,d) \in \mathcal{S}$, representing the accessibility of OD pairs (o,d). Problem (EC.1) can then be written as

$$\underset{\mathbf{v},\mathbf{x},\mathbf{y},\mathbf{z}}{\text{maximize}} \quad \sum_{(o,d)\in\mathcal{S}} q^{od} v^{od} \tag{EC.2a}$$

$$\text{subject to} \quad \text{(EC.1b)–(EC.1f)}$$

$$v^{od} \le \alpha_1 - \beta_1 \mathbf{t}^\intercal \mathbf{y}^{od}, \quad \forall (o,d) \in \mathcal{S} \tag{EC.2b}$$

$$v^{od} \le \alpha_2 - \beta_2 \mathbf{t}^\intercal \mathbf{y}^{od}, \quad \forall (o,d) \in \mathcal{S} \tag{EC.2c}$$

$$v^{od} \ge 0, \quad \forall (o,d) \in \mathcal{S} \tag{EC.2d}$$

$$\text{(EC.1g)–(EC.1i)}$$

where $\alpha_1 = 1$ and $\alpha_2 = 1 + (\beta_2 - \beta_1)T_1$ are the intercepts of the linear functions in $[0, T_1)$ and $[T_1, T_2)$ respectively.

Convex impedance function. When $\beta_1 > \beta_2$, g can be treated as a strictly convex function of travel time $\mathbf{t}^\intercal \mathbf{y}^{od}$ on $[0, T_2]$. We introduce binary decision variables r^{od}, for any OD pairs $(o,d) \in \mathcal{S}$, representing if the travel time of pair (o,d) is in $[0, T_1]$ $(=1)$ or not $(=0)$. We introduce continuous decision variabel $u_t^{od} \in \mathbb{R}_+$, for any OD pairs $(o,d) \in \mathcal{S}$ and any of the two travel time intervals $t \in \{1,2\}$. Problem (EC.1) can then be written as

$$\underset{\mathbf{r},\mathbf{u},\mathbf{x},\mathbf{y},\mathbf{z}}{\text{maximize}} \quad \sum_{(o,d)\in\mathcal{S}} \sum_{t=1}^{2} q^{od}(\alpha_t - \beta_t u_t^{od}) \tag{EC.3a}$$

$$\text{subject to} \quad \text{(EC.1b)–(EC.1f)}$$

$$u_1^{od} \le T_1 r^{od}, \quad \forall (o,d) \in \mathcal{S} \tag{EC.3b}$$

$$T_1(1 - r^{od}) \le u_2^{od} \le T(1 - r^{od}), \quad \forall (o,d) \in \mathcal{S} \tag{EC.3c}$$

$$u_1^{od} + u_2^{od} = \mathbf{t}^\intercal \mathbf{y}^{od}, \quad \forall (o,d) \in \mathcal{S} \tag{EC.3d}$$

$$u_t^{od} \ge 0, \quad \forall (o,d) \in \mathcal{S},\ t \in \{1,2\} \tag{EC.3e}$$

$$r^{od} \in \{0,1\}, \quad \forall (o,d) \in \mathcal{S} \tag{EC.3f}$$

$$\text{(EC.1g)–(EC.1i)}. \tag{EC.3g}$$

EC.3. Solution Method for MaxANDP using Location-Based Accessibility Measures

In this section, we present a Benders decomposition algorithm that takes advantage of the block structure of problem (EC.1) (i.e. the routing problems of the OD pairs are independent of each other) along with its acceleration strategies. This algorithm is applicable to both convex and concave impedance functions, and can be easily extended to solving the ML-augmented model of MaxANDP. For ease of presentation, we treat g as a function for travel time $\mathbf{t}^\intercal \mathbf{y}^{od}$ in this section.

EC.3.1. Benders Decomposition

We start by introducing the Benders reformulation of problem (EC.1). We introduce continuous decision variables τ^{od}, for any OD pair $(o,d) \in \mathcal{S}$, indicating the travel time from o to d using the low-stress network. We then re-written problem (EC.1) as

$$\underset{\mathbf{x},\mathbf{y},\mathbf{z},\boldsymbol{\tau}}{\text{maximize}} \quad \sum_{(o,d)\in\mathcal{S}} q^d g(\tau^{od}) \tag{EC.4a}$$

$$\text{subject to} \quad (\text{EC.1b}), (\text{EC.1c}), (\text{EC.1h}), (\text{EC.1i})$$
$$\tau^{od} \geq \min_{\mathbf{y}^{od}} \left\{ \mathbf{t}^\intercal \mathbf{y}^{od} \,\middle|\, (\text{EC.1d}) - (\text{EC.1g}) \right\}, \quad \forall (o,d) \in \mathcal{S}. \tag{EC.4b}$$

For each $(o,d) \in \mathcal{S}$, We associate unbounded dual variables $\boldsymbol{\lambda}^{od}$ with constraints (EC.1d) and non-negative dual variables $\boldsymbol{\theta}^{od}$, $\boldsymbol{\delta}^{od}$, and $\boldsymbol{\pi}^{od}$ with constraints (EC.1e)–(EC.1g), respectively. Given any network design $(\mathbf{x}, \mathbf{z})$, we formulate the dual of the routing problem as

$$\underset{\boldsymbol{\theta},\boldsymbol{\delta},\boldsymbol{\pi} \geq 0,\ \boldsymbol{\lambda}}{\text{maximize}} \quad -\lambda_d^{od} + \lambda_o^{od} - \sum_{(i,j)\in\mathcal{E}_h} x_{ij}\theta_{ij}^{od} - \sum_{i\in\mathcal{N}_h} \sum_{(i,j)\in\mathcal{E}_h^-(i)} \sum_{(w,l)\in\mathcal{E}_h(i)} (x_{wl}+z_i)\delta_{ijwl}^{od} - \sum_{(i,j)\in\mathcal{E}} \pi_{ij}^{od} \tag{EC.5a}$$

$$\text{subject to} \quad -\lambda_j^{od} + \lambda_i^{od} - \mathbb{1}\left[(i,j)\in\mathcal{E}_h\right]\theta_{ij}^{od} - \mathbb{1}(i\in\mathcal{N}_h) \sum_{(w,l)\in\mathcal{E}_h(i)} \delta_{ijwl}^{od} - \pi_{ij}^{od} \leq t_{ij}, \quad \forall (i,j)\in\mathcal{E}. \tag{EC.5b}$$

Since the routing problem associated with each OD pair is always feasible and bounded, its dual (EC.5) is also feasible and bounded. Let Π^{od} denote the set of extreme points of problem (EC.5). According to the duality theory, constraints (EC.4b) can be replace by

$$\tau^{od} \geq -\lambda_d^{od} + \lambda_o^{od} - \sum_{(i,j)\in\mathcal{E}_h} x_{ij}\theta_{ij}^{od} - \sum_{i\in\mathcal{N}_h} \sum_{(i,j)\in\mathcal{E}_h^-(i)} \sum_{(w,l)\in\mathcal{E}_h(i)} (x_{wl}+z_i)\delta_{ijwl}^{od} - \sum_{(i,j)\in\mathcal{E}} \pi_{ij}^{od}$$
$$\forall \boldsymbol{\lambda}^{od}, \boldsymbol{\theta}^{od}, \boldsymbol{\delta}^{od}, \boldsymbol{\pi}^{od} \in \Pi^{od},\ (o,d)\in\mathcal{S}. \tag{EC.6}$$

This set of constraints is of exponential size, but can be solved with a cutting-plane method that iterates between problems (EC.4) and (EC.5). More specifically, we initialize problem (EC.4) without any of the constraints (EC.6). We solve problem (EC.4) to obtain feasible $\mathbf{x}$ and $\mathbf{z}$ and trial values of τ^{od}. For each $(o,d)\in\mathcal{S}$, we then solve a problem (EC.5) with the $\mathbf{x}$ and $\mathbf{s}$ and check if the trail value of τ^{od} and the optimal solution of problem (EC.5) satisfy constraint (EC.6) or not. If not, the violated cut is added to problem (EC.4). This process repeats until no new cut is added to problem (EC.4).

Although this Benders decomposition algorithm largely reduces the problem size and improves computational efficiency, it is insufficient to deal with our synthetic instances due to the widely acknowledged primal degeneracy issue for network flow problems (Magnanti et al. 1986, Naoum-Sawaya and Elhedhli 2013) . We thus adopt a cut enhancement method and some acceleration strategies, which we describe in the next two sections, respectively.

EC.3.2. Pareto-Optimal Benders Cut

We adapt the cut enhancement method proposed by Magnanti et al. (1986) to generate pareto-optimal Benders cut by solving an auxiliary problem after a cut is identified by solving problem (EC.5). Let $(\bar{\mathbf{x}}, \bar{\mathbf{z}})$ denote a relative inner point of the feasible region specified by constraints (EC.1b)–(EC.1c), $\eta^{od}(\mathbf{x}, \mathbf{z})$ denote the optimal value of problem (EC.5) given $\mathbf{x}$ and $\mathbf{z}$. The Auxiliary problem associated with $(o,d) \in \mathcal{S}$ is

$$\underset{\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\delta},\boldsymbol{\pi}\geq 0}{\text{maximize}} \quad -\lambda_d^{od} + \lambda_o^{od} - \sum_{(i,j)\in\mathcal{E}_h} \bar{x}_{ij}\theta_{ij}^{od} - \sum_{i\in\mathcal{N}_h} \sum_{(i,j)\in\mathcal{E}_h^-(i)} \sum_{(w,l)\in\mathcal{E}_h(i)} (\bar{x}_{wl} + \bar{z}_i)\delta_{ijwl}^{od} - \sum_{(i,j)\in\mathcal{E}} \pi_{ij}^{od} \tag{EC.7a}$$

$$\text{subject to} \quad -\lambda_d^{od} + \lambda_o^{od} - \sum_{(i,j)\in\mathcal{E}_h} x_{ij}\theta_{ij}^{od} - \sum_{i\in\mathcal{N}_h} \sum_{(i,j)\in\mathcal{E}_h^-(i)} \sum_{(w,l)\in\mathcal{E}_h(i)} (x_{wl} + z_i)\delta_{ijwl}^{od} - \sum_{(i,j)\in\mathcal{E}} \pi_{ij}^{od} = \eta^{od}(\mathbf{x}, \mathbf{z}) \tag{EC.7b}$$

$$-\lambda_j^{od} + \lambda_i^{od} - \mathbb{1}\left[(i,j)\in\mathcal{E}_h\right]\theta_{ij}^{od} - \mathbb{1}(i\in\mathcal{N}_h) \sum_{(w,l)\in\mathcal{E}_h(i)} \delta_{ijwl}^{od} - \pi_{ij}^{od} \leq t_{ij}, \quad \forall (i,j)\in\mathcal{E}. \tag{EC.7c}$$

Problem (EC.7) is different from problem (EC.5) in that it has an additional constraint (EC.7b) ensuring that the solution generated by problem (EC.7) is optimal to problem (EC.5). We follow Lim et al. (2021) to initialize the relative inner points as

$$\bar{x}_{ij} = \min\{1,\ \frac{B_{\text{edge}}}{2|\mathcal{E}_h|c_{ij}}\}, \quad \forall (i,j)\in\mathcal{E}_h,$$
$$\bar{z}_i = \min\{1,\ \frac{B_{\text{node}}}{2|\mathcal{N}_h|b_i}\}, \quad \forall i\in\mathcal{N}_h.$$

Through the solution process, every time an integral network design decision $(\mathbf{x}', \mathbf{z}')$ is found, we follow Fischetti et al. (2017) and Papadakos (2008) to update the relative inner point as

$$\bar{x}_{ij} \leftarrow \frac{1}{2}(\bar{x}_{ij} + x'_{ij}), \quad \forall (i,j)\in\mathcal{E}_h,$$
$$\bar{z}_i \leftarrow \frac{1}{2}(\bar{z}_{ij} + z'_{ij}), \quad \forall i\in\mathcal{N}_h.$$

This cut enhancement strategy requires longer time to generate a single cut as an auxiliary problem has to be solved. However, according to our computational experiments, it significantly reduces the number of cuts needed for solving the problem, and thus achieves shorter overall computation time compared to naive implementation of the Benders decomposition algorithm.

EC.3.3. Other Acceleration Strategies

We adopt the following strategies to further accelerate the benders decomposition algorithm.

- *Initial Cut Generation.* Before solving problem (EC.4), we apply the Benders decomposition algorithm to solve its linear-programming (LP) relaxation. Following Fischetti et al. (2017), Bodur and Luedtke (2017) and Zetina et al. (2019), we then add the cuts that are binding at the optimal solution of the LP relaxation to problem (EC.4). These cuts help to obtain the LP-relaxation bound at the root node of the branch-and-bound tree.

- *Flow Variable Reduction.* In Problem (EC.1), routing variables are created for all $(o,d) \in \mathcal{S}$ and all $(i,j) \in \mathcal{E}$. However, given that a dummy low-stress link whose travel time is $\mathcal{T}_2$ is added to connect each OD pair, edges that are far away from the origin and destination will not be used. Therefore, for each OD pair $(o,d) \in \mathcal{S}$, we generate routing variables x_{ij}^{od} only if the sum of travel time from o to node i, travel time along edge (i,j), and the travel time from node j to d is less than T. This pre-processing strategy significantly reduces the problem size.

- *Design Variable Reduction.* In Problem (EC.4), road design decisions for different edges are made separately. However, a more practical way of cycling infrastructure planning is to build continuous cycling infrastructure on road segments, each consisting of multiple edges. Such road segments can be identified through communication with transportation planners. We incorporate this consideration by replacing the road design variables y_{ij} with y_p where p indicates the road segment that edge (i,j) belongs to. In our computational and case studies, we group all edges between two adjacent intersections of arterial roads into one project, resulting in 84 and 1,296 projects in the synthetic and real networks, respectively. This preprocessing strategy reduces the number of binary decision variables.

EC.4. Computational Study Details

EC.4.1. The Synthetic Grid Network

As presented in Figure EC.1, we create a synthetic network comprising a set of arterial roads, which are assumed to be high-stress, and local roads, which are assumed to be low-stress. The arterial roads constitute a 6x6 grid. Intersections of arterial roads are assumed to be signalized. On each arterial road segment, we place three nodes, each representing the intersection of the arterial road and a local road. Each of these intersections is assigned

a traffic signal with a probability of 0.3. We generate 72 population centroids randomly distributed within the 36 major grid cells. Each centroid represents one origin and is a destination for all other centroids. We create low-stress edges that connect each centroid to 70% of the intersections around the major grid cell in which that centroid is located. All edges are bidirectional. Each direction is assigned a travel time randomly distributed between 1 and 5. We use a constant travel speed of 1 to convert the generated travel time to distance. We consider all the OD pairs between which the shortest travel time on the overall network is less than 60, each assigned a weight uniformly distributed between 1 and 10. The network consists of 1,824 edges, 373 nodes, and 3,526 OD pairs. Arterial edges are grouped into 84 candidate projects. Setting the road design budget to 100, 300, and 500 roughly corresponds to selecting 5, 10, and 15 projects, respectively.

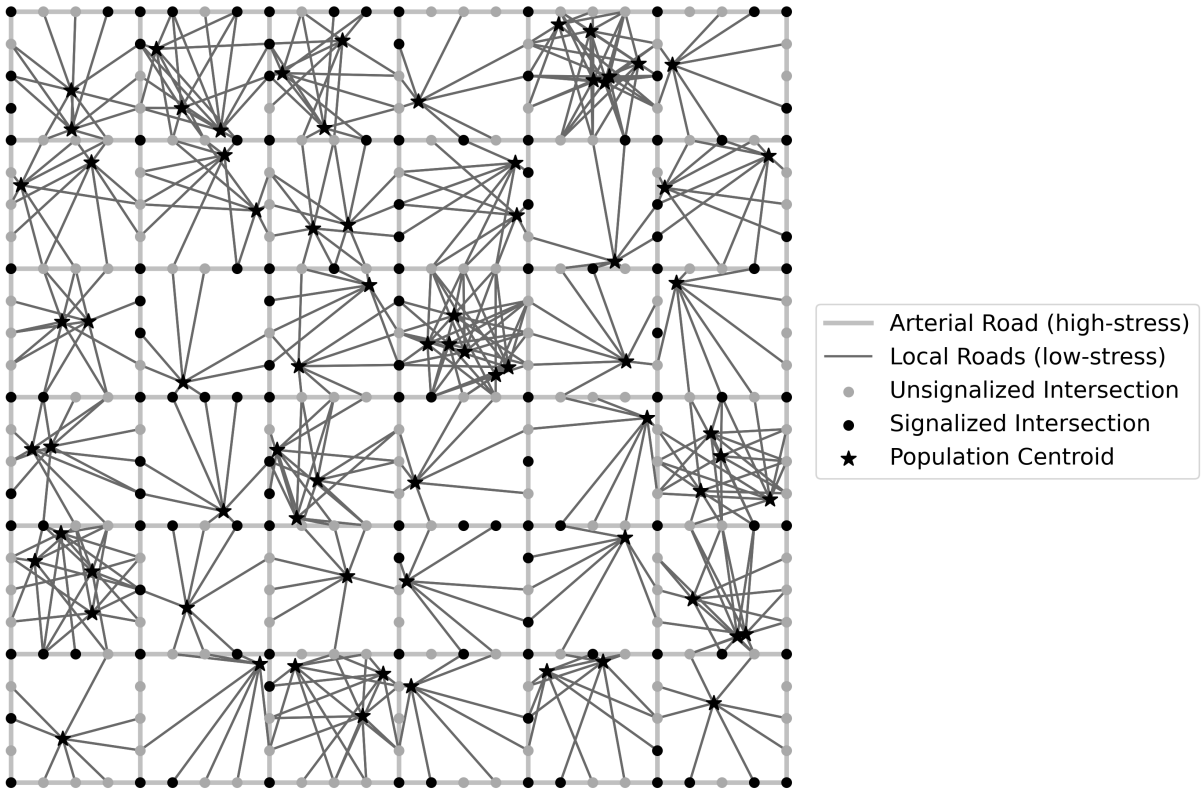

Figure EC.1 A synthetic grid network. Network components that are highlighted in black or dark grey constitute a low-stress network while others are high-stress.

EC.4.2. Accessibility Calculations

EC.4.2.1. Location-based accessibility measures. We vary the parameters of the piecewise linear function g, mimicing three commonly used impedance function for location-based accessibility (Figure EC.2):

1. *Negative exponential function*: $\beta_1 = 0.0375$, $\beta_2 = 0.00625$, $T_1 = 20$, $T_2 = 60$.
2. *Linear function*: $\beta_1 = 1/60$, $\beta_2 = 0$, $T_1 = 60$, $T_2 = 60$.
3. *Rectangular function*: $\beta_1 = 0.001$, $\beta_2 = 0.471$, $T_1 = 58$, $T_2 = 60$.

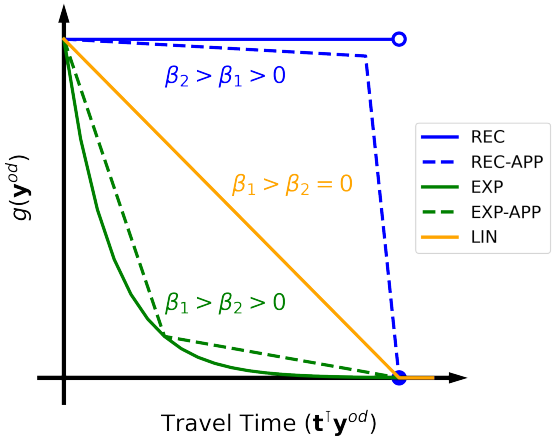

Figure EC.2 Negative exponential (EXP), rectangular (REC), and linear (LIN) impedance functions, and their piecewise linear approximations (APP).

EC.4.2.2. Utility-based accessibility measure. We adopt the utility-based measure proposed by Liu et al. (2021b). It requires as input a set of candidate routes for each OD pair and a constant α that reflects cyclists' preference between bike-lane continuity and bike-lane coverage along the routes. Following Liu et al. (2021b), we set the value of α to 1.05. We generate three candidate routes for each OD pair by solving three shortest path problems on the overall network using different edge travel cost. Specifically, we consider i) randomly generated travel time (Section EC.1), ii) Euclidean distance between the two ends, and iii) a uniform cost of 1 for all arterial roads and a uniform cost of 10 for other roads. The first and second definitions correspond to the goals of time minimization and distance minimization, both are commonly used by map software to generate route recommendations. The third definition reflects the preference for biking on major roads.

EC.4.3. Predictive Features

EC.4.3.1. Learned features. We apply the representation learning technique introduced in Section 5 to learn OD-pair features for MaxANDP. Implementation details are illustrated as follows.

- *Relationship Graph Construction.* We sample n_{sim} leader decisions for constructing the relationship graph. Ideally, the leader decisions should be sampled considering a specific design budget. However, the MaxANDP will be solved with various budgets in the computational studies (and in real-world transportation planning settings). Learning follower features multiple times may incur considerable computational burdens. Therefore, we consider learning features that are applicable to MaxANDP instances with different budgets by tweaking the leader decision sampling procedure. The high-level idea is to sample leader decisions with budgets randomly generated from a "wide" range that covers most budgets that may be used during the planning phase. As a result, the learned features contain information about follower similarity under various design budgets. Specifically, our leader-decision sampling procedure is parameterized by $\bar{P}$ and $\bar{Q}$ indicating, respectively, the maximum number of projects and the maximum number of nodes that can be selected in each sampled leader decision. Before generating a leader decision, we first randomly generate the number of projects and the number of nodes to be selected from intervals $[1, \bar{P}]$ and $[1, \bar{Q}]$, respectively. We then randomly select projects and nodes to form a leader decision. The procedure is summarized in Algorithm 4. In our computational studies, we set $\bar{P} = 25$, $\bar{Q} = 10$, and $n_{\text{sim}} = 5{,}000$. We present computational results about the impact of n_{sim} on feature quality in Section EC.4.8.
- *Follower Embedding.* In our computational study, we set $n_{\text{walk}} = 50$, $n_{\text{length}} = 20$, $\omega = 5$, and $\xi = 16$. We investigate the impact of ξ on feature quality in Section EC.4.8.

EC.4.3.2. TSP features. We adapt the features proposed by Liu et al. (2021a) for predicting TSP objective values. Specifically, each OD pair is assigned a nine-dimensional feature vector whose entries are

- The coordinates of the origin.
- The coordinates of the destination.
- The Euclidean distance between the origin and the center of the grid.
- The Euclidean distance between the destination and the center of the grid.
- The Euclidean distance between the origin and the destination.

Algorithm 4 Sampling leader's decision for our cycling network design problem

Input: Number of decisions n_{sim}; Set of cycling infrastructure project $\mathcal{P}$; Set of high-stress nodes $\mathcal{N}_h$, Maximum number of infrastructure project selected $\bar{P}$; Maximum number of nodes selected $\bar{Q}$.

Output: A set of leader's decisions $\bar{\mathcal{X}}$.

Initialize $\bar{\mathcal{X}} = \{\}$.
for $i = 1$ **to** n_{sim} **do**
    Generate $p \sim \text{Uniform}(1, \bar{P})$.
    Generate $q \sim \text{Uniform}(1, \bar{Q})$.
    Uniformly sample $\mathcal{P}_i \subseteq \mathcal{P}$ such that $|\mathcal{P}_i| = p$.
    Uniformly sample $\mathcal{Q}_i \subseteq \mathcal{N}_h$ such that $|\mathcal{Q}_i| = q$.
    Update $\bar{\mathcal{X}} \leftarrow \bar{\mathcal{X}} \cup \{(\mathcal{P}_i, \mathcal{Q}_i)\}$

- The area of the smallest rectangular that covers both the origin and the destination.
- The travel time of the shortest path from the origin to the destination on the overall network.

We perform a min-max normalization to scale each entry to $[0, 1]$.

EC.4.4. Follower Sampling Methods

We consider three follower sampling methods: i) vector assignment p-median sampling, ii) uniform sampling, and iii) p-center sampling. Uniform sampling selects each follower with a uniform probability, while the other two methods require solving an optimization problem. We next introduce the algorithms that we adopt to solve the optimization problems.

EC.4.4.1. Vector assignment p-median sampling. We adapt the meta-heuristic proposed by Boutilier and Chan (2020) for solving the classical p-median problem to solve the vector assignment p-median problem introduced in Section 4.3.

Starting from a randomly generated initial solution $\mathcal{T}$, the algorithm iterates between a "swap" phase and an "alternation" phase for $n_{\text{iteration}}$ iterations. In the swap phase, we create a solution in the neighborhood of $\mathcal{T}$ by randomly removing one follower from $\mathcal{T}$ and adding one follower from $\mathcal{S} \backslash \mathcal{T}$ to $\mathcal{T}$. We update $\mathcal{T}$ if the neighbor solution achieves a lower objective value for the vector assignment p-median problem. We perform at most n_{swap} such swaps in each iteration. In the alternation phase, for each follower $t \in \mathcal{T}$, we first find a follower set from $\mathcal{S}$ whose k-nearest neighbor in $\mathcal{T}$ includes t, and then solve an

1-median problem on this set. We solve in total $|\mathcal{T}|$ 1-median problems and their optimal solutions form a new solution to the vector assignment p-median problem. We update $\mathcal{T}$ if this solution achieves a lower objective value. The algorithm is summarized in Algorithm 5

Algorithm 5 A Meta-Heuristic for Solving the Vector-Assignment p-Median Problem

Input: A set of followers $\mathcal{S}$; Follower features $\{\mathbf{f}^s\}_{s\in\mathcal{S}}$; A distance metric in the feature space $d_{\mathcal{F}}$; Number of followers to select p; Number of medians each node is assigned to k; Maximum number of iterations $n_{\text{iteration}}$; Number of swaps in each iteration n_{swap}; An oracle that calculates the objective value of the vector assignment p-median problem $O_{p-\text{median}}$ for any given solution; An oracle that finds the k-nearest neighbor of a follower in a given set $O_{k\text{NN}}$.

Output: A set of selected followers $\mathcal{T}$.

Randomly sample $\mathcal{T}\subseteq\mathcal{S}$ such that $|\mathcal{T}|=p$.
Initialize iteration counter $m_{\text{interation}}=1$
while $m_{\text{iteration}}\leq n_{\text{iteration}}$ **do**
    for $i=1$ **to** n_{swap} **do** ▷ Swap
        Randomly sample a follower $s\in\mathcal{S}\backslash\mathcal{T}$ and a follower $t\in\mathcal{T}$.
        Create a follower set $\mathcal{T}'=\mathcal{T}\backslash\{t\}\cup\{s\}$
        if $O_{p-\text{median}}(\mathcal{T}')<O_{p-\text{median}}(\mathcal{T})$ **then** update $\mathcal{T}\leftarrow\mathcal{T}'$
    Initialize a follower set $\mathcal{T}''=\{\}$ ▷ Alternation
    for $t\in\mathcal{T}$ **do**
        Create a follower set $\mathcal{S}^t=\{s\in\mathcal{S}\backslash\mathcal{T}\,|\,t\in O_{k\text{NN}}(s,\mathcal{T})\}$.
        Solve an 1-median problem on $\mathcal{S}^t$: $s''\leftarrow\arg\min_{s\in\mathcal{S}^t}\sum_{l\in\mathcal{S}^t}d_{\mathcal{F}}(\mathbf{f}^s,\mathbf{f}^l)$.
        Update $\mathcal{T}''\leftarrow\mathcal{T}''\cup\{s''\}$
    if $O_{p-\text{median}}(\mathcal{T}'')<O_{p-\text{median}}(\mathcal{T})$ **then** update $\mathcal{T}\leftarrow\mathcal{T}''$
    Update $m_{\text{iteration}}\leftarrow m_{\text{iteration}}+1$

We note that the implementation of this meta-heuristic requires calculating the distance between every pair of followers in the feature space. For $\mathcal{S}$ that is relatively small, we pre-calculate and store the distance matrix of the followers. However, storing the distance matrix (over 200 GB) in RAM is practically prohibitive for a large $\mathcal{S}$ (e.g. the MaxANDP

of Toronto's road network). To tackle the challenge, we calculate the distances on the fly when needed during the searching process. More specifically, we only calculate the distances from the current "medians" to other followers, resulting in a much smaller distance matrix that can be stored in RAM. A GPU with 24 GB of RAM (NVIDIA RTX6000) is employed to accelerate the distance-matrix calculation.

EC.4.4.2. p-center sampling the p-center sampling select followers by solving the follower problem

$$\min_{\mathcal{T}\subseteq\mathcal{S}} \left\{ \max_{s\in\mathcal{S}\setminus\mathcal{T}} \min_{t\in\mathcal{T}} d_{\mathcal{F}}(\mathbf{f}^s, \mathbf{f}^t) \,\middle|\, |\mathcal{T}| = p \right\} \tag{EC.10}$$

We adapt a greedy heuristic to solve this problem. Specifically, we initialize the follower sample with a randomly selected follower $s \in \mathcal{S}$. Next, we iteratively select one unselected follower and add it to the follower sample until p followers have been selected. In each iteration, we first select calculate the shortest distance from each unselected follower to the follower sample, and then select the follower with the largest distance. This greedy heuristic generates 2-optimal solution for problem (EC.10). This is the best possible approximation that a heuristic algorithm can provide for the p-center problem because, for any $\delta < 2$, the existence of δ-approximation implies $\mathcal{P} = \mathcal{NP}$ (Gonzalez 1985, Hochbaum and Shmoys 1985). To empirically improve the solution quality, we apply this algorithm for n_{repeat} times and select the one that achieves the lowest objective value for problem (EC.10). For the computational experiments presented in Section 6, we set the value of n_{repeat} to 200. The heuristic is summarized in Algorithm (6).

EC.4.5. Computational Setups

All the algorithms were implemented using Python 3.8.3 on an Intel i7-8700k processor at 3.70 GHz and with 16GB of RAM. Optimization algorithms were implemented with Gurobi 9.1.2 (Gurobi Optimization, LLC 2022). The DeepWalk algorithm was implemented with Gensim 4.1.2 (Řehůřek and Sojka 2010). All the ML models for cycling accessibility prediction were implemented with Scikit Learn 1.0.2 (Pedregosa et al. 2011).

EC.4.6. Online ML Model Details

We follow Dumouchelle et al. (2022) to train a ReLU neural network to predict the leader's overall objective (the total accessibility of all OD pairs in $\mathcal{S}$) based on the leader's decision (network design). We choose to use the ReLU neural network because a trained ReLU neural network can be represented as a mixed integer program that can be used to search

Algorithm 6 A Greedy Heuristic for Solving the p-Center Problem

Input: A set of followers $\mathcal{S}$; Follower features $\{\mathbf{f}^s\}_{s\in\mathcal{S}}$; A distance metric in the feature space $d_{\mathcal{F}}$; Number of followers to select p; An oracle that calculates the objective value of the p-center problem $O_{p\text{-center}}$; Number of times the process is repeated n_{repeat}.

Output: A set of selected followers $\mathcal{T}$.

for $i = 1$ **to** n_{repeat} **do**
    Randomly sample a follower $s \in \mathcal{S}$.
    Initialize follower sample $\mathcal{T}_i = \{s\}$.
    while $|\mathcal{T}_i| \leq p$ **do**
        Calculate the distances to the selected set $d^s = \min_{t\in\mathcal{T}_i} d_{\mathcal{F}}(\mathbf{f}^s, \mathbf{f}^t)$ for all $s \in \mathcal{S}\backslash\mathcal{T}_i$.
        Select a follower $s' = \arg\max_{s\in\mathcal{S}\backslash\mathcal{T}_i} d^s$.
        Update $\mathcal{T}_i = \mathcal{T}_i \cup \{s'\}$.
Select $\mathcal{T} = \arg\min_{i=1\in[n_{\text{repeat}}]} O_{p\text{-center}}(\mathcal{T}_i)$

for the best leader's decision Dumouchelle et al. (2022). We use the sampling method described in EC.4.3 for learning follower features to generate 5,000 leader decisions and calculate the total accessibility of all OD pairs under each of the sampled leader decisions, constituting a data set of 5,000 data points. Increasing the size of the training dataset has marginal benefits according to our experiment.

The best hyper-parameters used for training this neural network are summarized as follows. We follow Dumouchelle et al. (2022) to use only one hidden layer with 64 nodes. We set the learning rate to 0.01 and batch size to 128. We use an ADAM optimizer and the mean-squared-error loss function to train the neural network for 10,000 epochs (after the training loss function becomes stable). The model is implemented using PyTorch 1.10.2 on a 14-inch Macbook Pro running on an M1 Pro CPU with 16 GB of RAM.

EC.4.7. Offline ML Model Details

For the four ML models we consider in Section 6, we select the hyper-parameters (if any) based on the mean of median out-of-sample prediction performance over 1000 datasets. We note that we do not create a validation set because the goal is to achieve as good performance as possible on the out-of-sample follower set $\mathcal{S}\backslash\mathcal{T}$. The generalization of the ML models outside $\mathcal{S}$ is not of interest in our study. The Linear regression does not involve

any hyper-parameter. The neighborhood sizes of kNN, the regularization factors of the lasso regression and ridge regression are summarized in Tables EC.1–EC.3, respectively.

Table EC.1 Neighborhood sizes of k-nearest neighbor regression.

		Accessibility Measure			
Feature	Budget	EXP	LIN	REC	UT
Learning	100	1	1	1	1
	300	1	1	1	1
	500	1	1	1	1
TSP	100	1	1	1	1
	300	1	1	1	1
	500	1	1	1	1

Table EC.2 Regularization parameters of lasso regressions.

		Accessibility Measure			
Feature	Budget	EXP	LIN	REC	UT
Learning	100	0.004	0.008	0.020	0.020
	300	0.004	0.008	0.010	0.020
	500	0.004	0.006	0.010	0.020
TSP	100	0.006	0.010	0.040	0.020
	300	0.008	0.010	0.030	0.030
	500	0.007	0.010	0.020	0.040

Table EC.3 Regularization parameters of ridge regressions.

		Accessibility Measure			
Feature	Budget	EXP	LIN	REC	UT
Learning	100	50	60	40	30
	300	50	40	10	60
	500	30	20	10	100
TSP	100	260	240	140	150
	300	210	170	170	150
	500	150	150	150	100

EC.4.8. Additional Results on the Impact of Hyperparameters on Feature Quality

We focus on two hyperparameters that may affect the quality of the REP features: i) feature dimensionality ξ, and ii) the number of leader decisions to sample n_{sim}. The idea is to first use a relatively large n_{sim}, which makes sure the embedding algorithm is well-informed about the relationship between followers, to find the smallest ξ that supports ML models to achieve "good" prediction performance. We want ξ to be small because it helps to reduce the size of the ML-augmented model. Once a ξ is chosen, we then search for a

small n_{sim} that makes the learned features perform well. We want n_{sim} to be small because it reduces the computational efforts in constructing the follower relationship graph. We focus on the prediction performance of kNN using UNI samples because kNN constantly achieves the best prediction performance among all ML models considered.

EC.4.8.1. The impact of ξ. We follow the convention to set the value of ξ to the powers of two. We vary ξ in $\{2, 4, 8, 16\}$ because the synthetic network has 3,526 followers and the smallest training sample considered is 1% of them, corresponding to 35 followers. Training linear regression models using 35 data points and features of 32 dimensions or more may lead to serious overfitting issues. We set n_{sim} to 5,000 when choosing ξ.

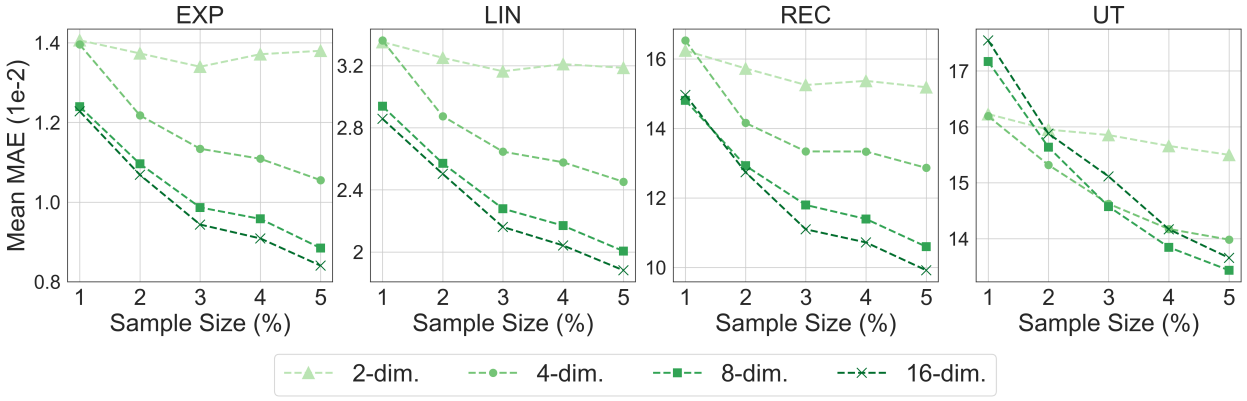

Figure EC.3 Out-of-sample prediction performance of kNN using REP features of two, four, eight, and 16 dimensions.

Figure EC.3 summarizes the predictivity of REP features of different dimensions. For location-based accessibility measures, increasing the number of dimensions leads to lower out-of-sample prediction error. For utility-based accessibility measures, REP features of four dimensions achieve the lowest error when the sample size is 1% or 2% of the original follower set, while 8- and 16-dimensional REPs become more predictive with larger training samples. For convenience, we choose to set $\xi = 16$ for all accessibility measures. However, based on the results presented in Figure EC.3, carefully tuning ξ for different problems may improve ML models' out-of-sample prediction accuracy.

EC.4.8.2. The impact of n_{sim}. We then fix $\xi = 16$ and vary n_{sim} in {10, 100, 1,000, 5,000}. Figure EC.4 summarizes the predictivity of REP features learned with different n_{sim}. In general, considering more leader decisions lead to better prediction performance of

the REP features, but the marginal improvement decreases as n_{sim} increases. For location-based accessibility measures, the improvement from $n_{\text{sim}} = 100$ to n_{sim} =5,000 is negligible compared to the improvement from $n_{\text{sim}} = 10$ to $n_{\text{sim}} = 100$. For the utility-based measure, REP features learned with n_{sim}=1,000 is as predictive as REP features learned with n_{sim} = 5,000.

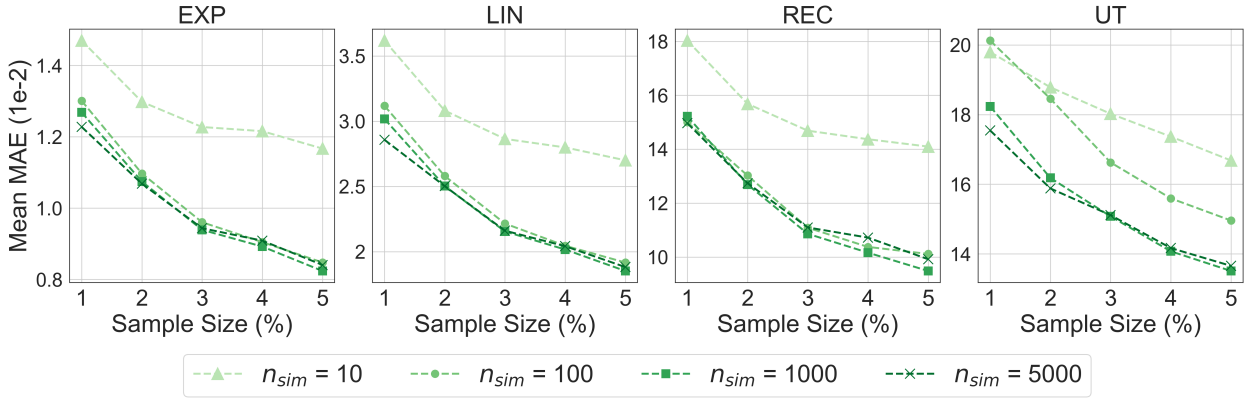

Figure EC.4 Out-of-sample prediction performance of kNN using REP features learned using 10, 100, 1000, and 5000 sampled leader decisions.

EC.4.9. Additional Results of Experiment 2

The full computation results from experiment 2 are visualized in Figure EC.5, including kNN-CEN, LR-CEN, and Reduced-CEN which are omitted in Section 6.3 for brevity. Overall, the performance of CEN samples is quite similar to that of UNI samples, with an exception for LIN instances where CEN samples are competitive with MED samples for both the reduced model and the ML-augmented models.

EC.5. Case Study Details

EC.5.1. Network Construction and Pre-Processing

We retrieve Toronto centerline network from Toronto Open Data Portal (City of Toronto 2020). We follow the following steps to process the network data.

1. We remove roads where cycling is physically impossible or legally prohibited, including "highway", "highway ramp", "railway", "river", "hydro line", "major shoreline", "major shoreline (land locked)", "geostatistical line", "creek/tributary", "ferry lane" per the City's definition.

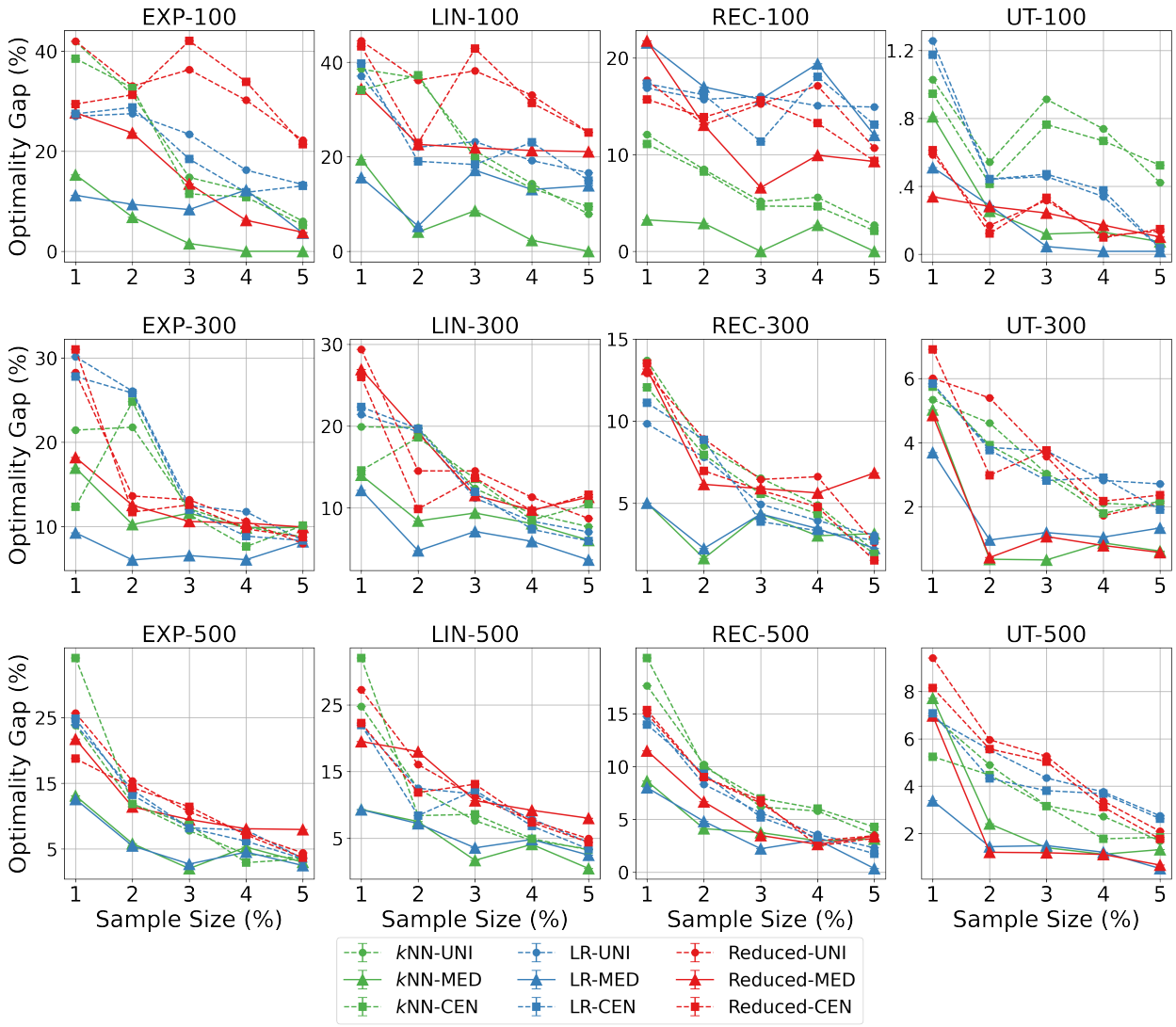

Figure EC.5 Optimality gap of leader decisions from the three sampling-based models on the 12 problem instances using UNI, MED, and CEN samples.

2. We remove all redundant nodes. A node is considered redundant if it is not an end of a road or an intersection of three or more edges. These nodes are included in the original network to depict the road shape, but are unnecessary from a modeling perspective.

3. We replace local roads with low-stress edges that connect DA centroids to intersections along arterial roads. We solve a shortest-path problem from each DA centroid to each intersection located on its surrounding arterial roads using the low-stress network. If a low-stress path is found, we add a bi-directional low-stress edge that connects the DA centroid and the intersection and set its travel time to the travel time along the path. All

local roads are then removed because we do not consider building new cycling infrastructure on local roads, and the role of local roads in our problem is to connect DA centroids to arterial roads, which can be served by the added artificial edges. The node and edge removal procedures are illustrated in Figure EC.6.

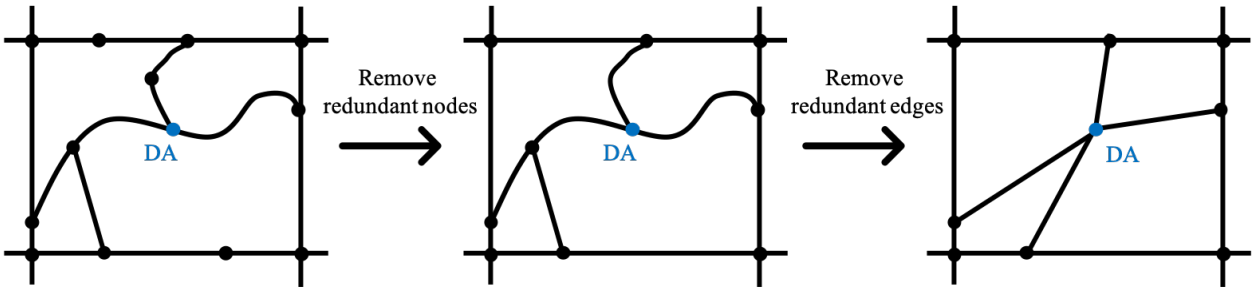

Figure EC.6 The procedures of removing redundant nodes and edges.

4. We group arterial edges to form candidate projects. A candidate project is defined as a continuous road segment that connects two adjacent intersections of arterial roads. Such a road segment may be represented as multiple arterial edges in the network due to the presence of arterial-local intersections. Grouping these edges together allows us to create fewer road design variables. The average length of the projects is 1.48 km.

5. Each DA is represented by its geometric centroid and is manually connected to its nearest point in the cycling network using a low-stress edge with zero travel time.

6. We discard all the OD pairs that are currently connected via a low-stress path because building new cycling infrastructure will not affect the accessibility of these OD pairs.

EC.5.2. Follower Embedding Details

We apply similar procedures as introduced in Section EC.4.3 to learn follower features. We highlight the key difference in each step as follows.

- *Relationship Graph Construction.* We sample n_{sim} =10,000 leader decisions with $\bar{P} =$ 300 and $\bar{Q} = 100$. We note that 149,496 (11.3%) of the 1,327,849 OD pairs have zero accessibility under all sampled leader decisions. As a result, their similarities to other OD pairs are all zero according to the adopted similarity measure. These OD pairs are mostly outside downtown Toronto, where the road networks are highly stressful. Connecting these OD pairs with low-stress routes requires constructing a large amount of new cycling infrastructure, which is beyond the considered budget (100 km). We choose to exclude these OD

pairs from OD pair embedding and thus exclude them from OD pair selection and ML-augmented model to avoid the computational burdens of sampling more leader decisions. However, we do take these OD pairs into consideration when evaluating leader decisions.

- *Follower Embedding.* We set $n_{\text{walk}} = 50$, $n_{\text{length}} = 50$, $\omega = 5$, and $\xi = 32$.

EC.5.3. Computational Setups

Optimization algorithms were implemented with Python 3.8.3 using Gurobi 9.1.2 (Gurobi Optimization, LLC 2022) on an Intel i7-8700k processor at 3.70 GHz and with 16GB of RAM. The heuristic for solving vector-assignment p-median problem is accelerated with h an NVIDIA P100 GPU. The DeepWalk algorithm was implemented with Gensim 4.1.2 (Řehůřek and Sojka 2010).

For optimal network expansions, we use follower samples of 2,000 followers (OD pairs), and solve the kNN-augmented model with them. We use the kNN-augmented model because the network design budget ($\leq$ 100 km) falls into the small budget regime, where the kNN-augmented model generally outperforms the linear regression-augmented as presented in Section 6.3. We set the solution time limit to 3 hours for all optimization models. The greedy network expansion is implemented using the same machine as used by the optimization models and is parallelized using eight threads.

EC.5.4. Comparison between Greedy and Optimal Expansions

Figure EC.7 presents the cycling infrastructure projects selected by the greedy heuristic and our approach given a road design budget of 70 km, as an example. The optimal expansion is 11.2% better than the heuristic expansion as measured by the improvement in Toronto's total low-stress cycling accessibility. Both algorithms choose many cycling infrastructure projects in the downtown core area, where a well-connected low-stress cycling network has already been constructed, and where job opportunities are densely distributed. These projects connect many DAs to the existing cycling network and thus grant them access to job opportunities via the existing network. However, unlike the greedy heuristic that spends almost all the road design budgets to expand the existing network, our approach identifies four groups of projects that are not directly connected to the existing network (as highlighted by the black frames in Figure EC.7). The greedy heuristic does not select these projects because they have little impact on the total cycling accessibility if

constructed alone. However, when combined, these projects significantly improve the accessibility of their surrounding DAs by breaking the high-stress barriers between low-stress cycling islands.

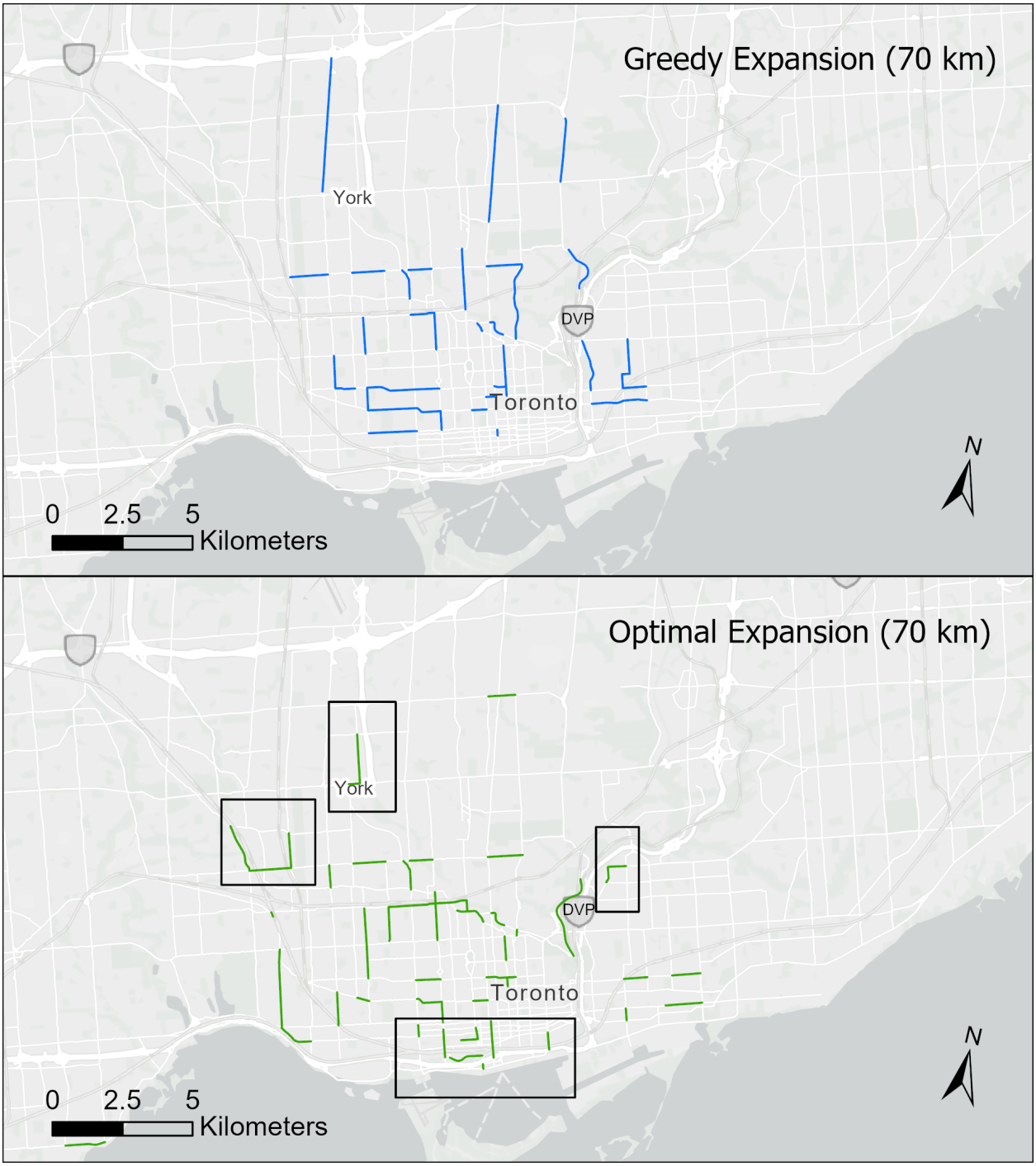

Figure EC.7 Greedy and optimal expansions given a road design budget of 70 km.