Unravelling System Optimums by trajectory data analysis and machine learning

Ruiwei Chen, Ludovic Leclercq, Mostafa Ameli

Unravelling System Optimums by trajectory data analysis and machine learning

Ruiwei Chen[a], Ludovic Leclercq[a,*], Mostafa Ameli[b]

[a] *Univ. Gustave Eiffel, Univ. Lyon, ENTPE, LICIT, F-69518, Lyon, France*
[b] *Univ. Gustave Eiffel, COSYS, GRETTIA, Paris, France*

Abstract

This work investigates network-related trajectory features to unravel trips that contribute most to system under-performance. When such trips are identified, feature analysis also permits determining the best alternatives in terms of routes to bring the system to its optimum. First, we define important trajectory features that unravel the trips contributing the most to the network under-performance. Second, based on supervised learning methods, we propose a two-step data-driven methodological framework to reroute a part of the users and make the system close to its optimum. The learning models are trained with trajectory features to identify which users should be selected, and which alternative routes should be assigned, thanks to the data and features obtained from two reference dynamic traffic assignment (DTA) simulations, under User-Equilibrium (UE) and System-Optimum (SO). We only focus on trajectory features that are accessible in real time, such as network features and regular travel time information, so that the methods proposed can be implemented without requiring cumbersome network monitoring and prediction. Finally, we evaluate the efficiency of the methods proposed through microscopic DTA simulations. The results show that by targeting 20 % of the users according to our selection model and moving them onto paths predicted as optimal alternative paths based on our rerouting model, the total travel time (TTT) of the system is reduced by 5.9 % in comparison to a UE DTA simulation. This represents 62.5 % of the potential TTT reduction from UE to SO, when all the users choose their path under the SO condition.

Keywords: Trajectory data analysis, Network-related trajectory features, System optimum (SO), Supervised learning, Polynomial regression, Naive Bayes classifier

1. Introduction

In urban areas, the conflict between the demand for increasing mobility and limited infrastructure capacities degrades the level of service of road networks. The consequences include: (i) economic loss resulting from wasted time and fuel in traffic jams, and (ii) environmental impacts due to air pollutants and noise emissions. Expanding infrastructures to improve network performance may not always be efficient in the long term (Braess et al., 2005). In this work, we focus on re-routing strategies that bring the system closer to the global optimum by moving some users out of their personal optimum. This solution occurs when all the users aim at minimizing their own travel cost, and it usually refers to user equilibrium (UE) (Wardrop, 1952; Smith, 1979). It is also known that the system optimum (SO) can be achieved if users cooperate to match a collective optimized function (i.e., minimizing total travel time) (Beckmann et al., 1956; Mahmassani and Peeta, 1993). Some studies show that SO can save up to 33 % of total travel costs in comparison to UE (Roughgarden and Tardos, 2002; Youn et al., 2008; van Essen et al., 2016). However, SO is achieved in return for higher individual travel costs experienced by some users. Therefore, instead of changing the paths of all users to achieve the perfect SO, it would be more efficient to focus on those who contribute the highest marginal gain in total travel cost to the whole system in return for a reasonable increase of their own travel costs.

Fortunately, researchers have found that only a small share of network users contribute most to system under-performance (Wang et al., 2012; Ameli et al., 2020b). The main challenges are how to identify (i) these users and

*Corresponding author. Tel. : +33 (0) 4 72 04 77 16.
Email address: ludovic.leclercq@univ-eiffel.fr (Ludovic Leclercq)

(ii) their alternative paths. To tackle these problems, we rely on trajectory data analysis and supervised learning models. In the past few decades, various trajectory data have become available. These data help engineers, decision makers and researchers to propose corresponding strategies for improving urban mobility (Gonzalez et al., 2008; Ma et al., 2015; Saeedmanesh and Geroliminis, 2016; Lopez et al., 2017). For example, with detailed Global Positioning System (GPS) data from mobile phones, Wang et al. (2012) showed that the congestion of a given network is mostly due to very few network users who travel on the most congested road segments. However, this conclusion was obtained by removing part of the traffic demand from certain origin-destination pairs (O-D pairs) without giving alternative routing solutions. Çolak et al. (2016) showed that if 10 % of drivers adjust their routing behavior under SO conditions, the whole system benefits from 40 % of the potential travel time saving that could be achieved if all users behaved unselfishly. Nevertheless, their conclusion was based on a traffic assignment model where the travel time on links depended only on its volume-over-capacity (VoC) ratio (BUREAU, 1964), i.e., the links between travel times are time-independent. Travel time delay is modeled by vertical queues, without considering spillbacks in congested situations. This is not representative enough of how congestion spreads in dense urban areas. In addition, the SO path distribution cannot be easily known in real life (Peeta and Mahmassani, 1995; Yildirimoglu and Kahraman, 2018) so that we are not able to ensure whether these users are traveling under SO conditions or not.

In this work, we present methods to both target the most contributive users, i.e., those with the highest marginal total travel time gain, and re-route them to alternative optimal paths. These methods are designed considering a dynamic framework. The main objective is to define a small, simple, and relevant set of trajectory features that determine the trips to be modified. Briefly speaking, trajectory features are the characteristics of a user's travel pattern, such as its length, travel time, average travel speed of the path, etc. In addition, the characteristic of the nodes and links on the pattern can also be taken into consideration, such as their topological features (degree, betweenness centrality, link lengths, etc), and traffic-related features (node/link capacity, traffic light cycles, etc). Since we aim to find the targeted users in practice based only on their trajectory features, these features should be kept easy to access in real-time. This is why we will consider only network features or regular travel time information without requiring expensive network monitoring and/or a prediction system. The second objective is to define alternative routes for the selected users by using the same features. A previous work presented in Chen and Leclercq (2019b) shows that that by rerouting a small part of users according to a single type of trajectory feature can improve the total system performance. However, this work will focus on selecting and rerouting users based on a combination of different trajectory features.

In this paper, we rely on results from dynamic UE and SO dynamic traffic assignment (DTA) simulations with the help of a microscopic simulator to investigate the differences in trajectory features. Note that with the same traffic demand on the same network, the assigned paths in UE and SO simulations can be compared pair-by-pair (Leclercq et al., 2016). This fact gives us access to the full picture of how UE / SO patterns differ. However, our purpose is to define a generic methodology that can rely on UE patterns based only on simple network-related features to perform user selection and re-routing. In addition, using a microscopic simulation framework allows us to easily assess network performance after selection and re-routing and provide a clear benchmark of our method. The contribution of this work is threefold:

- using comparative feature analysis of trajectories obtained in UE and SO simulations, we define the network-related trajectory features that unravel the trips contributing most to network under-performance;
- using the network-related trajectory features defined, we propose re-routing strategies for the selected users in order to improve total network performance (e.g., the total travel times of all vehicles);
- by applying machine learning techniques to vehicle trajectory data, we improve the total system performance of the network by changing the paths of a small share of network users without cumbersome DTA simulations.

One of the key contributions is that once the trajectory features are identified and the learning models are built, we are able to carry out user selection and re-routing strategies without knowing the actual network or trajectory features at SO state. The purposes of running a reference SO simulation are: (i) to provide the data set for training our machine learning models, and (ii) to serve as a criterion of the *optimal* network performance when we evaluate the efficiency of our method. However, during the method evaluation, it is still assumed that all the network users travel under UE perspectives and only their trajectory features at UE state are accessible. Under these assumptions, we can still

improve the whole network performance by targeting users and moving them to predicted alternative paths, based only on their trajectory features and/or regular travel time information under UE condition, thanks to our proposed learning models. This makes it possible to solve real-world congestion problems without cumbersome network monitoring and prediction.

The remainder of this paper is organized as follows. Firstly, we present the case study description, the overview of the method and the definitions of trajectory features in Section 2. Then Section 3 presents the case study results and discussions. Finally, Section 4 presents the main conclusions and future research perspectives of this work.

2. Methodology

This section presents the methodological framework, the definitions of trajectory features, and mathematical formulations of the supervised learning models used in this work. Table 1 shows the main notations.

2.1. Method overview

This subsection mainly focuses on the general methods without entering into detailed computation. Figure 1 presents the overall framework of our methodology. We carry out two reference DTA simulations under UE and SO conditions, with the same dynamic OD matrix and road network as inputs. These two scenarios are denoted as UE-ref and SO-ref, respectively. The main idea is to select α % of the users, give them rerouting guidance, and evaluate whether the system's performance is improved. Therefore, there are two important phases in the methodology: (i) the selection of targeted users and (ii) the determination of rerouting strategies. The total travel time (TTT) of all users in the whole network is considered as the criterion for evaluating the system's performance. The trajectories obtained in the SO simulation, named as SO paths/trajectories in the paper, are assumed to be the *optimal* paths for the system optimum. In fact, they are the paths that bring *the least marginal TTT* to the whole system when additional users travel in the network through these paths. The features of SO trajectories are used only for the construction of the data set. The main objective is to achieve a significant TTT reduction of the whole system by moving a small portion of the users onto their *predicted* SO trajectories, based on the application of data mining and supervised learning methods to trajectory feature data.

For each step, we first need to prepare the data set of trajectory features obtained from UE-ref and SO-ref, as shown in the blocks in the data preparation part of Figure 1. Based on the literature review, there are some key network features that contribute to system underperformance, such as the betweenness centrality (BC) of nodes and links (Freeman, 1977; Wang et al., 2008, 2012; Chen and Leclercq, 2019a,b; Bellocchi et al., 2020), corridor capacity (Laval and Castrillón, 2015), the distribution and cycle of traffic lights (Laval and Castrillón, 2015), etc. Section 2.3 will present the detailed definitions of the network-based trajectory features analyzed in this work. However, we first introduce the definition and computational methods for obtaining the path marginal cost (PMC).

In particular, we consider the differential of path marginal costs (diffPMC) between the UE and SO states of the network as the indicator to evaluate the marginal total travel gain that can be achieved by moving a user and then assess its potential contribution to improving the system's performance. Indeed, the marginal cost of a path represents the *increase* of travel cost to the whole system due to an additional vehicle on that path. In the SO state, users are on routes with equal and minimum PMC so that no users can shift to any other paths that have a lower marginal travel cost (Sheffi, 1985). Therefore, the re-assignment of the users with the largest PMC difference between UE and SO can shift the system close to the system optimum. However, this is not an option in practice as PMC can hardly be estimated in the reference situation (UE), and the differences in PMC are unavailable because the SO solution is unknown. This is the reason why we rely on analyzing simulated trajectory data. Here, we focus on network-related trajectory features that are easy to obtain in practice. Furthermore, to evaluate the quality of user selection, we are going to run new simulations considering that the selected users now have new predefined routes while the others keep traveling under the UE discipline. This permits us to account for the influence of the route guidance of the selected users on the other users in the system. In this work, we use a simulation-based method to compute PMC. Detailed definitions and computation methods are presented in Section 2.2.

Table 1: List of notations

Notations	Explanations
T	Travel time
$\mathcal{H}$	Planning horizon (the duration of $\mathcal{H}$ is 3600 minutes)
TTT	The total travel time of the network
$\Delta TTT_{\text{relative}}$	Relative TTT reduction with respect to UE-ref simulation
$\Delta TTT_{\text{relative w.r.t ref reduction}}$	Relative TTT reduction with respect to the *optimal* TTT reduction: $\text{TTT}_{\text{UE-ref}} - \text{TTT}_{\text{SO-ref}}$
LTT	Link travel time
PTT	Path travel time
LMC	Link marginal cost
PMC , $\widehat{\text{PMC}}$	Path marginal cost and dimensionless path marginal cost (normalized by free-flow travel time)
diffPMC	The difference between PMC of UE pattern and SO pattern
BC	Abbreviation of Betweenness Centrality
i	Index of user
j	Index of link
p, P	Index of trajectory feature, and the total number of analyzed trajectory features
α, α^*	Fraction and the *optimal* fraction of users that are selected to change trajectories
x, $\boldsymbol{x}$, $\boldsymbol{X}$	Scalar, vector, and matrix of attributes composed by trajectory feature(s)
y, $\boldsymbol{y}$	Target scalar and target vector to be predicted and they are obtained from the simulation data set
y', $\boldsymbol{y'}$	Scalar and vector *predicted* by supervised-learning model, with feature vectors as attributes
w, $\boldsymbol{w}$	Scalar and vector of regression coefficient
$f(\cdot)$	The learning model for targeted user selection, with trajectory features as input and diffPMC as output
$g(\cdot)$	The classifier for labeling whether a path is an SO path, with trajectory features as input and the path's class as output
C	Class of labels of a path
$\Delta\tau$	The snapshot time step of the microscopic simulator to collect simulation results (60s in this work)
$n(\tau)$	Spatially-averaged number of vehicles on a link during the time step $[\tau, \tau + \Delta\tau]$
$\phi(\cdot)$	Link performance function: the input is $n(\tau)$ and the output is link travel time
Δt	The time step for computing path travel time and path marginal cost (60s in this work)
$\mathcal{D}$	The total number of OD pairs in the network
$s_{i,d}$	The number of available alternative paths of the user i to the destination d, $d \in \mathbb{R}^D$
$\mathcal{G}$	The directed graph representing the network
$\mathcal{A}$	The set of links on the graph
$\mathcal{N}$	The set of nodes on the graph
$\mathcal{L}_i$	The trajectory of user i
L_i	Length of trajectory $\mathcal{L}_i$
J_i, K_i	Number of links and nodes on trajectory $\mathcal{L}_i$
μ_l	Mean distance between two consecutive traffic lights on a trajectory
σ_l	Standard deviation of distances between two consecutive traffic lights on a trajectory
$\overline{\mu}_l$	Dimensionless mean distance of two consecutive traffic lights on a trajectory
G_i^{BC}	Gini coefficient of node betweenness centrality on trajectory $\mathcal{L}_i$
$G_i^{\text{dynamicBC}}$	Gini coefficient of dynamic link betweenness centrality on trajectory $\mathcal{L}_i$
Q	One-lane link saturation capacity during green time
t_g, t_r	Green time and red time of traffic light cycle
μ_g, μ_r, σ_g, σ_r	Mean green / red time, and the corresponding standard deviations of traffic lights on a trajectory
δ	The coefficient of variance: $\frac{\sigma}{\mu}$
Cap_i	Mean MFD capacity of trajectory $\mathcal{L}_i$
η_i^{critical}	Normalized length of critical links on trajectory $\mathcal{L}_i$
$E_{i,j}$	Dynamic link betweenness centrality of link j on trajectory $\mathcal{L}_i$
$\overline{E}_i$, $\widehat{E}_i$	Mean and value of dynamic betweenness centralities of all the links on trajectory $\mathcal{L}_i$
BC_i, $\widehat{\text{BC}}_i$	Mean and median of betweenness centralities of all nodes on a trajectory $\mathcal{L}_i$

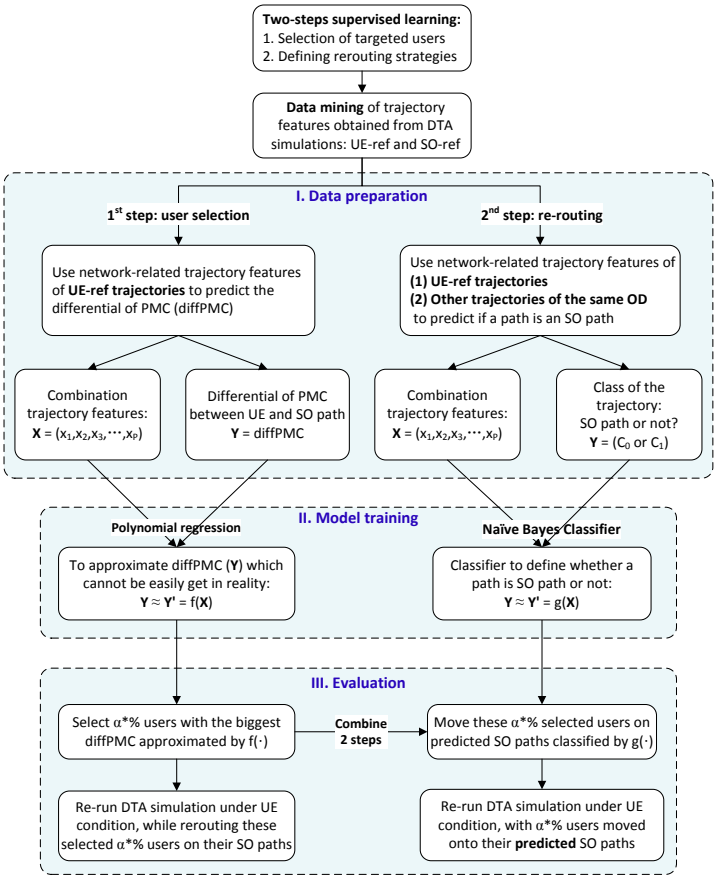


Figure 1: Methodology flow chart of this work

2.2. *Link marginal cost (LMC) and path marginal cost (PMC)*

The path marginal costs are computed based on time-dependent link marginal costs (LMC). LMC is defined as the change of link travel time caused by one additional unit of flow on the link. In the DTA simulation, LMC is time-dependent and can be computed by Equation (1) (Peeta and Mahmassani, 1995):

$$LMC_j(t,n) = \mathrm{T}_j(t,n) + n(t) \times \frac{\partial \mathrm{T}_j(t,n(t))}{\partial n(t)}, \tag{1}$$

where $\mathrm{T}_j(t)$ denotes the link travel time at t on link a_j, $n(t)$ denotes the spatially-averaged number of vehicles on link a_j. The challenge is to calculate $\frac{\partial \mathrm{T}_j(t,n(t))}{\partial n(t)}$. We assume that (i) the vehicle type on the network is single (i.e., mono class user), and (ii) the traffic light plan of the whole network is given. Therefore, the link marginal cost can be considered as an intrinsic feature of the link, and it depends only on n. It can be estimated via a simulation-based method. In this work, we use Symuvia[1] platform for dynamic traffic simulations. Symuvia is a dynamic microscopic trip-based simulator based on a Lagrangian discretization of the LWR model (Leclercq et al., 2007). Here, we present how we use Symuvia to compute simulation-based LMC. For the details of the traffic simulation, readers can refer to Ameli et al. (2020c).

The main idea of simulation-based methods is to approximate a link performance function, i.e., link travel time (LTT) vs. accumulation, for every link of the network. Then the LMC can be easily calculated by the deviation. To get

[1]Note that Symuvia is an open-source simulator which is available on GitHub: https://github.com/Ifsttar/Open-SymuVia

link performance functions, the main steps for computing LMC are as follows. First, we run a reference simulation with different traffic demand profiles. During each time step $[\tau, \tau + \Delta\tau]$, we compute the average travel time ($T_{\tau,j}$) and the spatially-averaged number of vehicles ($n_{\tau,j}$) on link a_j, and then save them as sample points $\{(n_{\tau,j}, T_{\tau,j})\}$. Second, for each link, we compute its maximum n by multiplying its length with the maximum density, and then split the values of n into $\mathcal{K}$ equal-width classes. Then for each class k, we gather all the points $(n_{\tau,j}, T_{\tau,j})$ of which $n_{\tau,j}$ belongs to the class k, and compute the corresponding average values of n and T during the whole simulation period, in order to get $(n_{k,j}, T_{k,j})$ for link a_j. At last, we fit a regression model representing a time-dependent link performance function $\phi_j : n \to \mathrm{T}$, based sample points of $(n_{k,j}, T_{k,j})$. In this work, we choose linear and/or quadratic regression models for fitting link performance functions. Therefore, during the time step $[\tau, \tau + \Delta\tau]$, we can compute $n(\tau)$ and $\frac{d\mathrm{T}}{dn(\tau)}$ for a given time τ. By replacing T_j in Equation (1) by ϕ_j, the LMC of a_j can be computed by Equation (2). It is worth mentioning that the estimated performance function ϕ_j is only used to compute LMC and then path marginal costs. The path travel times of users are still obtained from the results of microscopic simulator Symuvia, by computing the difference between the arrival time and enter time of users. These path travel times are used to compute User Equilibrium at each iteration.

$$LMC_j(\tau, n) = \phi_j(n(\tau)) + n(\tau) \times \frac{d\phi_j(n)}{dn}|_{n=n(\tau)}, \tag{2}$$

where $n(\tau)$ is estimated as $n(\tau) = \frac{TTT(\tau)}{\Delta\tau}$. $TTT(\tau)$ denotes the total travel time of all users travelling on link a_j during that time step $[\tau, \tau + \Delta\tau]$.

Therefore, the PMC on a trajectory $\mathcal{L}_i$ can then be obtained by summing up all the time-dependent LMC on the trajectory (Peeta and Mahmassani, 1995). The time step for computing PMC is $\Delta t = 60\,\mathrm{s}$ in our case study. The total number of time steps is denoted by $|\mathcal{H}|$. PMC_i is then computed as follows:

$$\mathrm{PMC}_i = \sum_{t}^{|\mathcal{H}|} \sum_{j} \mathrm{LMC}_{j,t} \delta_{j,t}, \tag{3}$$

where $\delta_{j,t}$ is the incidence indicator. If the entering time of user i on the link a_j belongs to the time interval $[t, t + \Delta t]$, then $\delta_{j,t} = 1$. Otherwise, $\delta_{j,t} = 0$. The computation of LMC and PMC is time-dependent. In SO-based DTA based on microscopic simulator, the PMCs are computed after network loading at each iteration. The links passed through by a user i are already known, and the average number of vehicles (denoted as $n(t)$) of all links at all time intervals $[t, t + \Delta t]$ are already computed, too. To compute the PMC of the user i, we check its entering time on each link that is on the current loaded path of the user i, and see the entering time belongs to which time interval. Then we use the corresponding $n(t)$ to compute LMC(t) based on Equation(2), and sum up all LMCs of all links on the path to get the PMC.

When we analyze trajectory features in order to determine which users should be re-routed and which alternative paths should be chosen, it is assumed that we don't have real-time arrival information of users on each link of their paths. Therefore, the computations of LMC, PMC and diffPMC are *departure-time centered*. In other words, we compute PMC and diffPMC of a user i by using Equation (3), where t corresponds to its entering time onto the network. In addition, it is worth noticing that the diffPMC is computed based only on features obtained in the UE simulation. Indeed, user i has a pair of paths resulting from the UE-ref and SO-ref simulations. As we assume that moving the user from the UE path to the SO path should bring the system close to its SO, users with a high diffPMC value should be selected first, as mentioned in Section 2.1. However, when we try to re-route a user who is traveling on the network under the UE condition, we only have access to traffic-state characteristic (LTTs and LMCs) in the UE state. Therefore, even though we can move it to its SO path, the diffPMC from UE to SO path should be computed based on the LMCs obtained in UE-ref. To be more precise, we denote $\mathrm{LMC}^{\mathrm{UE}}_{j,t}$ as the LMC of link a_j computed from the UE-ref simulation during the time step $[t, t + \Delta t]$. We denote $\mathrm{PMC}^{\mathrm{UE}}_i$ and $\mathrm{PMC}^{\mathrm{SO}}_i$ as the PMCs of the UE and the SO path for user i, respectively. These variables are computed by Equation (4):

$$\mathrm{PMC}^{\mathrm{UE}}_i = \sum_{t}^{|\mathcal{H}|} \sum_{j} \mathrm{LMC}^{\mathrm{UE}}_{j,t} \delta^{\mathrm{UE}}_{j,t}, \qquad \mathrm{PMC}^{\mathrm{SO}}_i = \sum_{t}^{|\mathcal{H}|} \sum_{j} \mathrm{LMC}^{\mathrm{UE}}_{j,t} \delta^{\mathrm{SO}}_{j,t}, \tag{4}$$

where $\delta^{\text{UE}}_{j,t}$ and $\delta^{\text{SO}}_{j,t}$ denote the incidence indicators defined in Equation (5) and Equation (6), respectively.

$$\delta^{\text{UE}}_{j,t} = \begin{cases} 1 & \text{if the user } i \text{ is on the link } a_j \text{ during } [t, t+\Delta t] \text{ in UE-ref simulation;} \\ 0 & \text{Otherwise.} \end{cases} \tag{5}$$

$$\delta^{\text{SO}}_{j,t} = \begin{cases} 1 & \text{if the user } i \text{ is on the link } a_j \text{ during } [t, t+\Delta t] \text{ in SO-ref simulation;} \\ 0 & \text{Otherwise.} \end{cases} \tag{6}$$

Therefore, the diffPMC of the user i in this work is calculated by the Equation (7).

$$\text{diffPMC}_i = \text{PMC}^{\text{UE}}_i - \text{PMC}^{\text{SO}}_i \tag{7}$$

It is worth mentioning that users may have the same paths under both UE and SO conditions. Therefore, they are already on their optimal paths under both UE and SO perspectives, and will not be chosen as users to be re-routed in this work. The definitions of PMC and diffPMC in Equation (4) to Equation (7) confirm that. In fact, if a user has the same path in both UE and SO simulation, its diffPMC is zero. Since we aim to re-route the users who have the largest value of diffPMC, these users with zero diffPMC would not be chosen. The value of diffPMC is the target value y that we aim to train in Section 2.5. Before presenting the learning process, we introduce detailed definitions and formulations of the network-related trajectory features that are used in this work in the following Section 2.3.

2.3. *Network-related trajectory features*

Network-related trajectory features are obtained by aggregating features of links and/or nodes. Here we use a modeled network to define different network-related trajectory features. A road network is modeled as a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, composed with K nodes and J links. $\mathcal{N} = \{n_1, n_2, \ldots, n_i, \ldots, n_K\}$ is the set of nodes and $\mathcal{A} = \{a_1, \ldots, a_j, \ldots, a_J\}$ is the set of links. A trajectory $\mathcal{L}_i$ with length L_i is composed by a set of links (a_j) and nodes (n_k):

$$\mathcal{L}_i = \{\{a_{i,1}, \ldots, a_{i,j}, \ldots, a_{i,J_i}\}, \{n_{i,1}, \ldots, n_{i,k}, \ldots, n_{i,K_i}\}\}, \tag{8}$$

where J_i and K_i denote the number of links and the number of nodes on $\mathcal{L}_i$, $a_{i,j} \in \mathcal{A}$, $n_{i,k} \in \mathcal{N}$. Therefore, network-related trajectory features of $\mathcal{L}_i$ result from aggregating network features of ($a_{i,j}$) and/or ($n_{i,k}$). There are two categories of network features to be aggregated: (i) graph-theory based and (ii) traffic-state based. We select the features based on a literature review, and keep those which can be calculate in real-time with limited information (such as monitoring, network traffic light system, and current travel time).

2.3.1. *Graph-theory based features: aggregation of centrality metrics*

The aggregate metrics of *critical* nodes on a trajectory can be considered as one of its network-related trajectory features. In graph theory, there are several metrics to define whether a node is critical to the whole network. For example, the *betweenness centrality (BC)* of a node n corresponds to the ratio of shortest paths crossing n over all possible shortest paths for all origin-destination pairs of the network (Freeman, 1977; Girvan and Newman, 2002). The BC of node n is calculated by Equation (9).

$$BC(n) = \sum_{i \neq j} \frac{\sigma_{ij}(n)}{\sigma_{ij}}, \tag{9}$$

where $\sigma_{ij}(n)$ denotes the number of shortest paths from any node n_i to any node n_j crossing node n, and σ_{ij} denotes the total number of shortest paths from n_i to n_j. In the case study in this work, the *shortest paths* for calculating BC

are measured by distance based on the topological parameters of the network, without taking into account the dynamic traffic condition. Therefore, for trajectory $\mathcal{L}_i$, the mean node BC of $\mathcal{L}_i$ is defined as follows:

$$\overline{BC}_i = \frac{1}{K_i} \sum_{n_{i,j}} BC(n_{i,j}). \tag{10}$$

where K_i denotes the number of nodes on $\mathcal{L}_i$. In addition to the mean value, we can also compute other statistical metrics of node BCs on $\mathcal{L}_i$, such as the median value of all node BCs, denoted as $\widehat{BC}_i$.

There may be some critical nodes on a trajectory that are the main contributions to the congestion. Therefore, we also consider the Gini coefficient of node BCs as a network-related trajectory feature to be analyzed. The Gini coefficient (G) is a measure of inequality, defined as the mean of absolute differences between all pairs of variables for a particular measure. The minimum value is 0 when all the measurements are equal. The theoretical maximum is 1 for an infinitely large set of observations where all the measurements but one have a value of 0, which is ultimate inequality. G is calculated by equation (11).

$$G = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} |x_i - x_j|}{2n^2 \bar{x}}, \tag{11}$$

where x denotes an observed value, n denotes the number of values observed, and $\bar{x}$ denotes the mean value of all x. In this work, for a given trajectory, the Gini coefficient of node BC can be considered as a measure of the existence of nodes with a large BC value on a trajectory. The Gini coefficient of node BC on a trajectory $\mathcal{L}_i$ can be calculated by:

$$G_i^{\mathrm{BC}} = \frac{\sum_{j=1}^{K_i} \sum_{h=1}^{K_i} |BC_{i,j} - BC_{i,h}|}{2K_i^2 \overline{BC}_i}, \tag{12}$$

where K_i denotes the number of nodes (intersections) on $\mathcal{L}_i$. $BC_{i,j}$ and $BC_{i,h}$ denote the BC of the j^{th} and h^{th} node on $\mathcal{L}_i$. $\overline{BC}_i$ denotes the mean value of all node BCs on a trajectory $\mathcal{L}_i$

2.3.2. *Graph-theory based features: mean distance between traffic lights*

In urban areas, the distribution of traffic lights on the network can affect the route choices of users, and contribute to the system's congestion. Here we present a dimensionless feature that can interpret the distribution of traffic lights on a path. For each trajectory $\mathcal{L}_i$, we compute the mean distance between two consecutive intersections with traffic lights (denoted as μ_{l_i}). Then we normalize it by dividing the length of $\mathcal{L}_i$:

$$\overline{\mu}_{l_i} = \frac{\mu_{l_i}}{L_i}. \tag{13}$$

In the two subsections above (Section 2.3.1 and Section 2.3.2), we present several important network-related trajectory features that depend only on the topological characteristics of the network. In the next subsection, we introduce several features that take the dynamic traffic states into account.

2.3.3. *Traffic-state features: mean MFD trajectory capacity*

An indicator of trajectory capacity is calculated by aggregating simple traffic-state features (e.g., fundamental diagram) and traffic light data of links on the trajectory. Let Q denote saturation capacity during the green time. For links with traffic lights, the outflow capacity of a link is calculated as $\frac{t_g}{t_g+t_r}Q$, where t_g and t_r represent the green time and red time of the traffic light cycle. With known link lengths and a given traffic light plan, we can compute the mean MFD capacity for trajectory $\mathcal{L}$, based on the conclusion of Laval and Castrillón (2015). A trajectory is considered as a one-lane corridor composed of several road segments separated by traffic lights. Let μ_l and σ_l denote the mean distance between two consecutive traffic lights and the corresponding standard deviation. μ_g, μ_r, σ_g and σ_r denote the mean green time, mean red time, and the corresponding standard deviation of traffic lights on $\mathcal{L}$. Let $\delta = \frac{\sigma}{\mu}$ denotes the

coefficient of variance. The mean MFD capacity is then determined by three dimensionless values: (i) mean red time over mean green time ratio: $\rho = \frac{\mu_r}{\mu_g}$, (ii) mean block length to mean green time ratio: $\lambda = \frac{\mu_l}{\mu_g}$, and (iii) the coefficient of variance of green light time, red-light time, and block length. It is assumed that the coefficient of variance is the same for these three variables, denoted as δ. With these parameters computed for a trajectory, an approximation for the mean corridor capacity is given in Laval and Castrillón (2015) and the formulation is presented in Equation (14). To make the features dimensionless for the analysis, the traffic flow is represented in the unit of Q, and the density is presented in jam density units.

$$\text{Cap} = min\{\frac{1}{1+\rho(0.58\delta\lambda + 1.64\lambda^2 - 5.3\lambda + 4.99)}; \frac{\mu_g}{\mu_g + \mu_r}\}. \tag{14}$$

The mean MFD capacity of a trajectory is considered as a network-related feature. All the parameters of Equation (14) can be obtained by aggregating a sequence of link lengths between each pair of constructive traffic lights, and the traffic cycle data of links on the trajectory.

Besides the MFD capacity, the percentage of critical links on a path can also indicate whether this trajectory contributes to the performance of the whole network. In this work, a link is considered as *critical* if it has a relatively small saturation capacity Q. In this work, we consider $a_{i,j}$ as *critical* link if $Q_j \leq 110\,\%\text{Cap}_i Q$, where $Q_j = \frac{t_g^j}{t_g^j + t_r^j} Q$. Therefore, the normalized length of critical links on $\mathcal{L}_i$ reads:

$$\eta_i^{\text{critical}} = \frac{\sum_{\text{critical link}} l}{L_i}. \tag{15}$$

η_i^{critical} is considered as metrics to quantify trajectory features related to critical links.

2.3.4. *Traffic-state features: dynamic betweenness centrality of links*

Bellocchi et al. (2020) define the concept of *reachability* of links by taking into account the dynamic traffic-state features of the network, i.e., the dynamic link betweenness centrality. When the average speed of each link $a_{i,j}$ is known, we can consider the following measure calculated as $E(a_{i,j}(t)) = \frac{e_i(t)+e_j(t)}{2}$, where

$$e_i(t) = \frac{1}{K-1} \sum_{h \in \mathcal{N} \backslash i} \frac{d_{kh}^{ff}}{\tau_{kh}(t)}, \tag{16}$$

where d_{kh}^{ff} denotes the travel time of the shortest time path in free-flow condition between nodes k and h (for all $h \in \mathcal{N}\backslash k$). $\tau_{kh}(t)$ denotes the actual path travel time (PTT) of the shortest-time path at time t, between the same pair of Origin-Destination nodes. Note that when we compute $e_i(t)$ here, t corresponds to the *departure time* of user i on the network. The PTT of user i is in fact computed by summing up all the *departure-time-centered* LTTs of links that are on the trajectory $\mathcal{L}_i$. The LTTs can be obtained by a simple monitoring system without any dynamic prediction. K denotes the total number of nodes in the network. $E(a(t))$ denotes a measure of the accessibility of link $a = (i, j)$ and it is evaluated with a number between 0 and 1. The closer the value of $E(a(t))$ is to 1.0, the less congested link a is. Then, for trajectory $\mathcal{L}_i$, we can compute the mean and median values of the dynamic link BCs on it: $\overline{E}_i$ and $\widehat{E}_i$.

Similarly, for a given trajectory $\mathcal{L}_i$, we can also compute the Gini coefficient of dynamic BCs of all the links on $\mathcal{L}_i$ by the following equation:

$$G_i^{\text{dynamicBC}} = \frac{\sum_{j=1}^{J_i} \sum_{h=1}^{J_i} |E_{i,j} - E_{i,h}|}{2J_i^2 \overline{E}_i}, \tag{17}$$

where J_i denotes the number of links on $\mathcal{L}_i$. $E_{i,j}$ denotes the dynamic link BC of link a_j on $\mathcal{L}_i$. $\overline{E}_i$ denotes the mean value of all the dynamic link BCs on the same trajectory $\mathcal{L}_i$ and it is computed by $\overline{E}_i = \frac{1}{J_i} \sum_{j=1}^{J_i} E_{i,j}$.

The two trajectory features presented in these two subsections, the mean MFD capacity and the dynamic BC of links, take into account the dynamics of the traffic such as the traffic light plan, the monitored LTTs, etc. However, they still remain simple to access and easy to calculate in practice.

2.4. Feature scaling and normalization

Since a trajectory has multiple features spanning varying degrees of magnitude, range, and units, we should carry out feature scaling before the application of machine learning techniques. According to their definitions, some of the trajectory features defined in Section 2.2 and Section 2.3 are already dimensionless: the median of node BC on $\mathcal{L}_i$ ($\widehat{BC}_i$), the Gini coefficient G_i^{BC} and $G_i^{\text{dynamicBC}}$, the MFD capacity (Cap_i), the normalized length of critical links on a trajectory (η_i^{critical}), and the normalized average distance between two consecutive intersections with traffic lights ($\overline{\mu}_{l_i}$).

Regarding the path marginal costs, we normalize the PMC by trajectory free-flow travel time (ffft), as shown in Equation (18). The trajectory free-flow travel time is computed by summing link ffft of the links that are passed through by the path. The link ffft can be approximated by dividing the distance of a_j by the maximum authorized speed on a_j, which depend only on the intrinsic features of a given network.

$$\widehat{\text{PMC}_i} = \frac{\text{PMC}_i}{\text{ffft}_i}. \tag{18}$$

Now we have clear definitions and formulations of all the trajectory features used in this work. In the next two subsections, Section 2.5 and Section 2.6, we present detailed formulations of the machine learning methods used in this work, following the two-step methodological framework introduced in Section 2.1 and Figure 1. In fact, apart from the features presented in Section 2.3, there are various kinds of other trajectory features such as path travel time, path travel distance, degree of nodes, degree of links, etc. According to some previous descriptive analysis and results presented in Chen and Leclercq (2019b), the betweenness centralities and mean MFD capacities are two potential features that can define SO paths. That is the reason why we mainly focus on the trajectory features presented in Section 2.3 in this work.

2.5. First phase: selection of targeted users

In this subsection, we present the methods for selecting targeted users based on the users' trajectory features, i.e., the left branch of 1^{st}- *step* shown in Figure 1. The main framework is consistent with the three parts introduced in Figure 1: data set preparation, model training, and evaluation.

First, we prepare the data set. For all the M users on the network, we compute their network-related features in the reference UE simulation and save them invector $\mathbf{x} \in \mathbb{R}^P$, where P denotes the number of trajectory features to be analyzed. We also compute diffPMC of each user, according to Equation (4) to Equation (7), and we save it in $y \in \mathbb{R}$.

Then, we carry out supervised learning with these M training points, in order to obtain $f : \mathbf{x} \rightarrow y$, where y is the target scalar to be approximated based on $\mathbf{x}$. Therefore, for any user i with P trajectory features under the UE condition, we can approximate its potential PMC decrease, i.e., diffPMC, by $y \approx y' = f(\mathbf{x}_i)$. The value of y' defines whether a user should be targeted to change trajectory. In this work, we mainly rely on linear regression (LR) and polynomial regression (PR) to train f (Seal, 1967; Cameron and Trivedi, 2005). The aim of the LR is to find a column vector $\mathbf{w} = (w_1, w_2, \ldots, w_p)^T$ so that $y = \varepsilon + \mathbf{x}^T \cdot \mathbf{w} = \varepsilon + \sum_{j=0}^{P} w_j \times x_j$. w_i denotes the weight associated to each factor in $\mathbf{x}$. $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ is the Gaussian error term. Based on M samples, we obtain the matrix expression of the LR: $\mathbf{y} = \mathbf{X} \cdot \mathbf{w} + \epsilon$, with $\epsilon \in \mathbb{R}^M$ as the vector of estimation error. Then we rely on the M sample set to estimate $\mathbf{w}$ by minimizing $\epsilon = \|\mathbf{y} - \mathbf{X} \cdot \mathbf{w}\|^2$ based on the least-squares estimation (Equation (19) and (20)).

$$\mathbf{w} = \arg\min_{\mathbf{w}} \sum_{i=1}^{M} \|y_i - \mathbf{x}^i \cdot \mathbf{w}\|^2 = \arg\min_{\mathbf{w}} \sum_{i=1}^{M} \|y_i - \sum_{j=0}^{P} w_j \times x_j^i\|^2 \tag{19}$$

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{20}$$

Moreover, there are often nonlinear relationships between the feature vector and the target. The polynomial regression (PR) model is also frequently used to address nonlinear relationships. The PR model is a polynomial transformation based on the LR model. For a model with only one variable, the k^{th} order polynomial model is given by $y = w_0 + w_1 x + w_2 x^2 + \ldots + w_k x^k + \varepsilon$. Here, the dimension of the feature vector is more than one and we should also take into

account the interaction terms between different features. For example, a polynomial regression model of the 2^{nd} order with two variables reads: $y = w_0 + w_1x_1 + w_2x_2 + w_{11}x_1^2 + w_{22}x_2^2 + w_{12}x_1x_2 + \varepsilon$. In this work, the dimension of the feature vector is $P = 7$. We save all the trajectory features of M users in the matrix of attributes $\boldsymbol{X} = [\boldsymbol{x}^i, \ldots, \boldsymbol{x}^P]$, and we save all the diffPMC of these M users in $\boldsymbol{y} \in \mathbb{R}^M$. Then, we train the LR and PR models based on the data set composed with $\boldsymbol{X}$ and $\boldsymbol{y}$. The detailed implementation and the numerical results are presented in Section 3.2.

Lastly, we evaluate the efficiency of our learning method with the steps shown in *part-III* of Figure 1. We re-run the DTA simulation by selecting a certain percentage of users according to their y', i.e., *predicted* diffPMC, and we move them to their *SO paths* simulated by the SO-ref scenario. Therefore, we evaluate whether rerouting these selected users can lead to significant network performance improvement or not, by comparing (i) the TTT obtained in this evaluation DTA scenario and (ii) the TTTs obtained in UE-ref and SO-ref simulations.

2.6. Second phase: re-routing strategy based on combining network-related features

Up to now, to evaluate the efficiency of the selection step, we have assumed that the *optimal* route guidance strategy for all the selected users is to assign them to their SO paths. Here, we relax this assumption. For a selected user, among all the paths in the feasible space, we aim to find the *best* path for bringing the system's performance closer to the system optimum, without knowing the SO solutions computed via cumbersome DTA simulations. To this end, we rely on supervised classification based on the network-related features of trajectories under the UE condition. By assuming that SO paths are the *optimal* paths for reaching the system optimum, we aim to classify whether a path is the SO solution or not based on network-related trajectory features. This step corresponds to the right branch of the 2^{nd} – step shown in Figure 1. We still need the results from the SO-ref simulation in order to prepare the data set and train our classification model. However, once the model is built and validated, we can label whether a path is an SO path or not, based only on the trajectory features.

2.6.1. Data preparation

We first prepare the data set by computing the features of trajectories obtained in the UE-ref and SO-ref simulations. More specifically, we assume that a user i entering the network has $s_{i,d}$ available paths, computed from both the UE-ref and SO-ref simulations, between its OD pair with d as the destination, $d \in \mathbb{R}^D$ where D denotes the total number of OD pairs in the network. Each of these paths has its own vector of features under: $\mathbf{x}_{s_{i,d}}$, which are composed of values of network-related features presented in Section 2.3 under the UE condition. For all these paths, we label them by two trajectory categories: (i) C_1 if the trajectory is the SO solution and (ii) C_0 otherwise. We save the labels of all the paths in the target vector y. In addition, we compute the trajectory features for all the $s_{i,d}$ paths, and we save the features of all the paths in the matrix of attributes $\boldsymbol{X} = \{\boldsymbol{x}_{s_{i,d}}\}_{i \in \mathbb{R}^M, d \in \mathbb{R}^D}$.

2.6.2. Model training

Using the prepared data set, we now train a model $g : \boldsymbol{X} \rightarrow y = (C_0 \text{ or } C_1)$, in order to predict whether a path is an SO path or not, based on the feature table $\boldsymbol{X}$.

In this work, we rely on the Bayes Theorem (Wu et al., 2008; Pearl, 2014; Taheri et al., 2014) to solve the classification problem. Classification learning is the process of predicting a discrete class label $C = \{C_1, \ldots, C_m\}$ for a test attribute $\mathcal{X} = \{X_1, \ldots, X_n\}$ (Wu et al., 2008; Pearl, 2014; Taheri et al., 2014). Here, learning algorithms are given the task of training a classifier that assigns a class label to a sample based on the attribute. In this work, we assume that there are only two classes: (i) C_0 contains the path that is not an SO path and (ii) C_1 contains the path is labeled as an SO path. Then we assign a trajectory $\mathcal{L}$ to C_0 or C_1, based on its feature vector $\mathbf{x} = (x_1, \ldots, x_P)^T$.

One effective classifier is the Bayes Classifier. It is constructed based on Bayes' theorem and the probability that a given sample belongs to a particular class. Let H be a hypothesis, such that the trajectory $\mathcal{L}$ belongs to class C. Our objective is to determine the probability that the hypothesis $H = Prob(H|\mathbf{x})$ holds, given that we already know $\mathbf{x}$. According to Bayes' theorem, $Prob(H|\mathbf{x})$ can be expressed as:

$$Prob(H|\mathbf{x}) = \frac{Prob(\mathbf{x}|H)Prob(H)}{Prob(\mathbf{x})}, \tag{21}$$

where $Prob(\mathbf{x}|H)$ denotes the posterior probability of $\mathbf{x}$ conditioned on H. $Prob(H)$ and $Prob(\mathbf{x})$ denote the prior probability of H and $\mathbf{x}$, respectively. When there are only two classes, a sample with attribute $\mathbf{x}$ is labeled as the class $C = C_0$ if and only if the following condition is met.

$$g_B(\mathbf{x}) = \frac{Prob(C = C_0|\mathbf{x})}{Prob(C = C_1|\mathbf{x})} \geq 1, \tag{22}$$

where $g_B(\mathbf{x})$ is called the Bayesian classifier.

The probabilities in Equation (21) can be estimated by the data set used for the learning process. However, the estimation of $Prob(\mathbf{x}|C)$ is non-trivial and learning an optimal Bayesian network is an NP-hard problem (Chickering, 1996; Chickering et al., 2004; Taheri et al., 2014). To simplify the problem, we assume that all the attributes are independent given the hypothesis, which leads to

$$Prob(\mathbf{x}|H) = Prob(x_1, \ldots, x_P|H) \approx \prod_{j=1}^{P} Prob(x_j|H), j \in \{1, 2, \ldots, P\}, \tag{23}$$

where P denotes the dimension of $\mathbf{x}$, i.e., the number of features to be analyzed for the trajectories in this work. Then the classifier in Equation (22) reads:

$$g^{\mathrm{NB}}(\mathbf{x}) \approx \frac{Prob(C = C_0)}{Prob(C = C_1)} \prod_{j=1}^{P} \frac{Prob(x_j|C = C_0)}{Prob(x_j|C = C_1)}. \tag{24}$$

$g^{\mathrm{NB}}(\mathbf{x})$ is called *Naive Bayes (NB)* classifier. The assumption that all the attributes are independent given that H is called *conditional independence*. The NB classifier is one of the most efficient and effective learning algorithms for discrete classification problems (Kotsiantis et al., 2007; Wu et al., 2008).

We use the data from UE and SO reference simulations to train the NB classifier in this work. All the paths used are identified in both simulations for each OD pair. These paths are added to the set of alternative paths. Then, we compute the feature vectors $\mathbf{x}$ for all paths. For a trajectory $\mathcal{L}$ between a given OD pair, we define the class variable y as:

$$y = \begin{cases} 0 & \text{if } C_{\mathcal{L}} = C_0; \\ 1 & \text{if } C_{\mathcal{L}} = C_1. \end{cases} \tag{25}$$

Therefore, the NB classifier to be trained is defined as follows:

$$\widehat{y} = g^{\mathrm{NB}}(\mathbf{x}) = \arg\max_{y} Prob(y) \prod_{j=1}^{P} P(x_j|y). \tag{26}$$

$Prob(y)$ can be estimated by the relative frequency of $y = 1$ and $y = 0$ in $\mathbf{Y}$. For $P(x_j|y)$, we apply the Gaussian Naive Bayes algorithm to solve the classification problem, where the posterior probability of features $Prob(x_j|C)$ is assumed to be Gaussian:

$$P(x_j|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp(-\frac{(x_j - \mu_y)^2}{2\sigma_y^2}), \tag{27}$$

where σ_y and μ_y are estimated by the data set. By substituting Equation (27) in Equation (26), the Gaussian NB classifier reads:

$$\widehat{y} = g^{\mathrm{NB}}(\mathbf{x}) = \arg\max_{y} Prob(y) \prod_{j=1}^{P} \frac{1}{\sqrt{2\pi\sigma_y^2}} exp(-\frac{(x_j - \mu_y)^2}{2\sigma_y^2}). \tag{28}$$

The detailed implementation of training an NB classifier based on the data set in this work and the case study results are presented in Section 3.4.2.

2.6.3. Method evaluation

Lastly, we evaluate the performance of the combination of the two-step methods: the user selection and re-routing method. We first select α^* % from all the M users as targets, according to their *predicted* diffPMC with the help of the f trained in the Section 2.5. We assume that the OD pairs and departure time of these users are known, and then we compute the trajectory features under the UE condition for all the alternative paths of the targeted users in order to obtain the table of feature attributes: $\mathbf{X}_{\text{target}}$. Therefore, we can predict the labels of all the alternative paths for the targeted users, via $Y' \approx g(\mathbf{X}_{\text{target}})$. Finally, we re-run a UE simulation with the same demand profile and network, after pre-defining the paths of selected users as the paths labeled by $Y' = C_1$. By comparing the TTT of (i) the whole system in this test scenario with (ii) the TTT obtained in UE-ref and SO-ref simulations, we evaluate the efficiency of the two-step methods proposed.

3. Numerical experiment and results

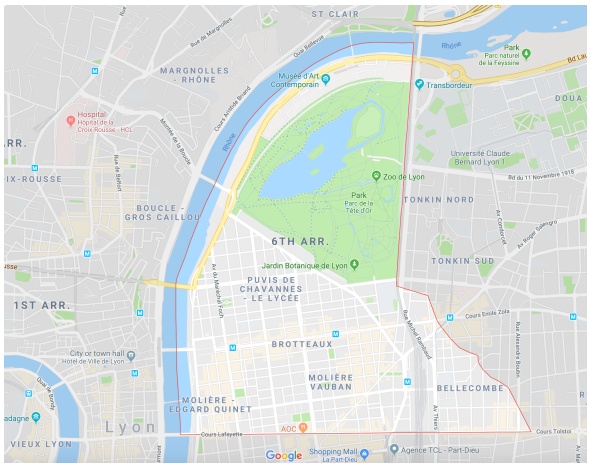
In the previous section, the general methodological framework is presented as well as the detailed formulations of the LR, PR and NB models. Note that, in the methods proposed, we fix the fraction of selected users in priority to assure a fair comparison of TTT reduction and evaluate the efficiency of the method. In practice, we can also select users and define alternative paths according to a certain threshold of the computed diffPMC instead of fixing a certain number of targeted users. In addition, the proposed methodology can be easily implemented, and we are free to choose DTA models and/or simulators. As long as we are able to obtain full information on network-related trajectory features, we can apply the methods proposed to any other DTA simulators/platforms. In this work, we apply this framework to a numerical case study with the microscopic simulator Symuvia. In this section, we present the case study scenarios and results.

In this study, we run two reference simulations via Symuvia based on the UE and SO conditions on the same road network and with a given demand profile. The calculation process for the UE and SO solutions are detailed in (Ameli et al., 2020a,b). We can compute their PMCs and network-related trajectory features listed in Section 2.2 and Section 2.3. The data set of the trajectory features will then be used in the supervised learning process. Recall that we use only link features under the UE condition when computing the traffic-state related features of both the UE trajectories and SO trajectories. More precisely, we use link travel time, link marginal cost, and link average speed under the UE condition to compute PTT, PMC, diffPMC, and dynamic BC of links. Therefore, the trajectory feature differences for a pair of trajectories result only from whether the trajectory is chosen under a UE or under an SO perspective. This helps us to unravel the characteristics of the SO structure under UE-based traffic conditions.

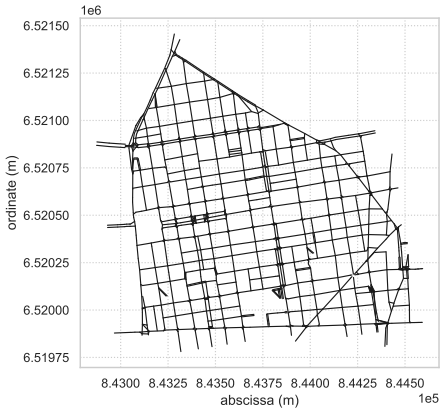
3.1. Case study description

In this study, we consider the road network of the 6^{th} district of Lyon (*Lyon6*), France. The area of Lyon6 is presented in Figure 2(a). Figure 2(b) shows the link-level representation of the main road network. There are a total of 786 links, 457 nodes, and 710 OD pairs in the network. Figure 2(c) presents the time-dependent traffic demand in the network. There are a total 13475 users (M = 13475) traveling on the network during a simulation period of 4 hours (from 7:00 to 11:00 a.m.). Figure 2(c) shows that the peak hour is from 8:00 to 9:30 am.

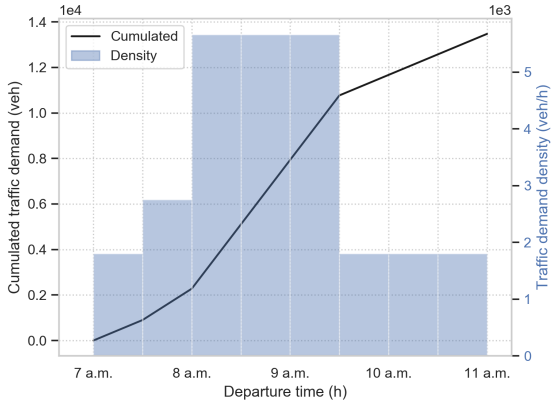
Figure 2(d) presents the macroscopic fundamental diagram (MFD) of the whole Lyon6 network in the UE and SO simulations. It represents the spatial average of network traffic conditions (total travel distance vs. total experienced travel time) through time. The MFD confirms that the traffic conditions in SO are better than in UE, as the total travel time is shorter in SO than in UE for a given total travel distance.



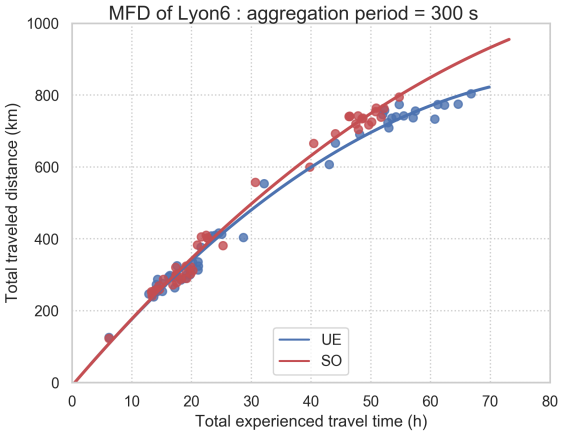
(a) Area of the 6^{th} district of Lyon, France, © Google Maps 2021



(b) The modeled network of Lyon6 (excluding the park).



(c) Traffic demand



(d) MFDs in UE and SO reference simulations

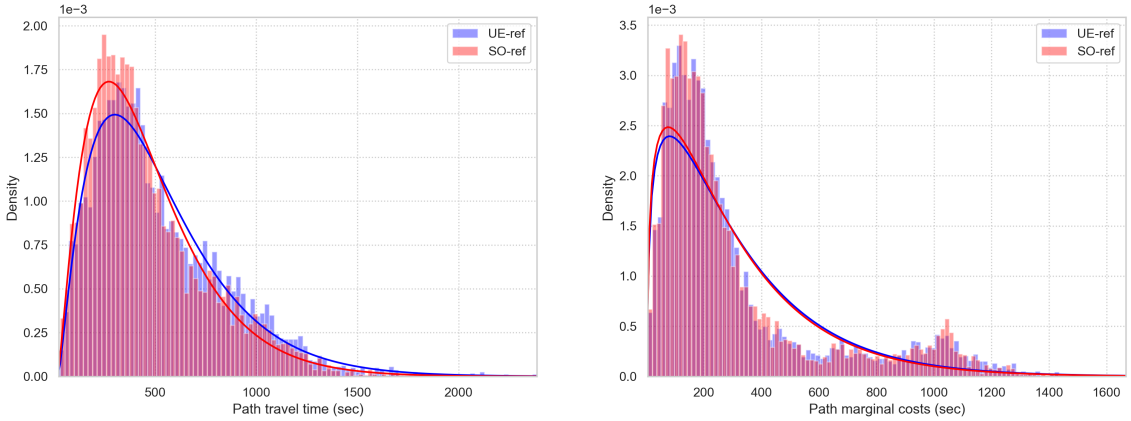
Figure 2: Lyon6 network: test case characteristics in UE-ref and SO-ref simulations with Symuvia

3.2. Results from reference scenarios

Figure 3(a) presents the distribution of PTT on the network of both the reference UE simulation (UE-ref) and the SO simulation (SO-ref). We can see that the PTT decreases from UE-ref to SO-ref, and the network is less congested in the SO than in the UE-ref simulation. Figure 3(b) shows the distribution of PMC. We also find the PMCs of the SO trajectories are smaller than the UE trajectories. This observation is consistent with the assumption of the SO condition where users choose paths with the minimum PMC between its OD pair.

We assume that the TTT of the whole system reaches its minimum under the SO condition. Therefore, the TTT reduction from UE-ref to SO-ref denotes the improvement of the *optimal* performance that can be achieved when all users choose their paths under the SO perspective. The reference simulations show that from the UE to SO, the TTT of all users falls by $6.56 \cdot 10^5$ seconds, representing 9.44 % of the TTT in the UE-ref simulation, as presented in Table 2. This reduction of TTT from UE-ref to SO-ref is then considered as the reference TTT reduction when evaluating the efficiency of targeting and re-routing strategies. More precisely, we use two criteria to evaluate the improvement of network performance: (i) the relative TTT reduction with respect to the UE-ref simulation (Equation (29)) and (ii) the relative TTT reduction with respect to the *optimal* TTT reduction from the reference scenarios (Equation (30)).

$$\Delta TTT_{\text{relative}} = 100\,\% \times \frac{(TTT_{\text{UE-ref}} - TTT_{\text{test scenario}})}{TTT_{\text{UE-ref}}}. \tag{29}$$



(a) The distribution of PTT.

(b) The distribution of PMC

Figure 3: The distributions of path travel time (PTT) and path marginal cost (PMC)

$$\Delta TTT_{\text{relative w.r.t ref reduction}} = 100\,\% \times \frac{(TTT_{\text{UE-ref}} - TTT_{\text{test scenario}})}{TTT_{\text{UE-ref}} - TTT_{\text{SO-ref}}}. \tag{30}$$

Before the application of the supervised learning methods mentioned in Section 2 to our case study, we first determine the *optimal* fraction of users to be targeted, i.e., α^* %, based on a sensitivity analysis presented in Section 3.2. Thereafter, when evaluating the proposed learning methods, we take α^* % directly as the fraction of user selection. The sensitivity analysis and results are presented in the following Section 3.2. Moreover, the reference test case presented in Section 3.2 can serve as the criterion for the method evaluation. The results from this reference case will confirm the fact that we can achieve a significant TTT reduction by selecting a small share of the users who have the largest diffPMC values and moving them to their SO paths. Therefore, when we evaluate the proposed learning models, we compare the TTT reduction of the test cases with the TTT reduction obtained in the reference cases.

Then, after presenting the reference test cases, we apply the two-step learning framework to our case study in Section 3.2 and Section 3.4.2. To prepare the data set, we compute trajectory features for all the users and prepare the data set, including: (i) the training feature vectors composed of network-related trajectory features, (ii) the diffPMC for all users as the target vector, and (iii) the labels of simulated paths as the target vector. To select the users, we use the data set to train the LR and PR regression models. For re-routing, we train the NB classifier to predict whether a given path is an SO path or not. Lastly, for the evaluation phase of both learning processes, we run DTA simulations with the selected users moving onto their simulation/predicted SO paths in order to evaluate the TTT reduction of the whole network.

3.3. The reference case: selection of targeted users according to actual path marginal cost

In this subsection, we aim to determine the *optimal* percentage α^* of users to be selected. With the α^* determined, we then re-run a test scenario by selecting α^* % users with the largest values of diffPMC, and move them to the SO paths obtained in the SO-ref simulation. Then the TTT reduction of this test scenario in comparison to the TTT in the UE-ref simulation is the criterion of the potential TTT reduction that can be achieved by rerouting α^* % users.

The main steps for the sensitivity analysis to determine α^* are the following. First, we compute the diffPMC for all the M users based on the simulation results obtained in UE-ref and SO-ref, according to Equation (4) to Equation (7). Then we rank all users according to their values of diffPMC, and select α % of them with the largest values of diffPMC as targeted users. Afterwards, we give these selected users predefined paths obtained in SO-ref and re-run a DTA simulation where other users still travel under the UE condition. Finally, we evaluate the TTT reduction of the system in the test scenario in comparison to UE-ref simulation.

Table 2: Simulation results of the UE-reference and SO- reference scenarios of the Lyon6 network. The bold numbers represent the results by re-routing an *optimal* fraction of users to SO paths.

	UE	SO	Results after rerouting different fraction of selected users (α %) with large diffPMC value									
Different scenarios	Ref	Ref	1 %	2 %	3 %	4 %	5 %	10 %	15 %	**20 %**	25 %	30 to 70 %
TTT (10^6 s)	6.94	6.29	6.93	6.73	6.72	6.62	6.68	6.66	6.47	**6.42**	6.52	6.39
ΔTTT (10^5 s)	–	6.56	0.18	2.17	2.21	3.30	2.73	2.87	4.81	**5.28**	4.31	5.58
ΔTTT per user (s)	–	48.7	1.3	16.1	16.4	24.5	20.3	21.3	35.7	**39.2**	32.0	41.4
ΔTTT_{rel} (%)	–	9.44	0.26	3.13	3.18	4.74	3.93	4.14	6.92	**7.60**	6.20	8.02
$\Delta TTT_{\text{rel. to ref}}$ (%)	–	100	2.76	33.14	33.69	50.24	41.62	43.81	73.33	**80.56**	65.71	85.01

(TTT is the total travel time. ΔTTT is the difference of TTT between the UE-ref simulation and the SO-ref simulation. ΔTTT/user denotes the TTT reduction per user. The relative TTT reduction (ΔTTT_{rel}) is computed by Equation (29). The relative TTT with respect to the reference ($\Delta TTT_{rel.\ to\ ref}$) is computed by Equation (30).)

To analyze the sensitivity of system performance improvement to the fraction of selected users, we select $\alpha \in \{1, 2, 3, 4, 10, 15, 20, 25, 30, 35, 40, 45, 50, 60, 70\}$ and run the corresponding test scenarios mentioned in the above paragraph. Then we compute the relative TTT reductions according to the criteria mentioned in Section 3.2, in order to find the *optimal* fraction α^* %. The results of these simulation scenarios are presented in Table 2. The results show that we can achieve a significant improvement in network performance by moving only a small share of the users to the paths considered to be *optimal* for the whole system, i.e., the paths assigned under the SO principle. Table 2 shows that re-routing 20 % the users to their SO-trajectories can lead to more than 80 % reduction in TTT concerning the reference case. In average, the travel time saving is 39 s per user. In other words, if we can efficiently target the 1/5 of the users that have the highest values of diffPMC, we can achieve more than 3/4 of the TTT reduction of the reference case, where all the users travel on SO trajectories instead of traveling under the UE principle.

We also carry out benchmark (BM) scenarios by randomly selecting α % of the users, with $\alpha \in \{1, 2, 3, 4, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 60, 70\}$. For each fraction α %, we take 8 random samples of the users independently, and give them predefined paths as SO trajectories. Then we run DTA simulations under the UE condition to evaluate the TTT reduction in comparison to the UE-ref simulation. The results in Table 2 and the results of the BM scenarios are plotted in Figure 4. The results of the BM scenarios show that randomly moving a fraction of the users to their SO paths cannot ensure a reduction of system TTT until we choose a sufficient number of users (30 %). In addition, the random selection is not efficient at all. In fact, by randomly selecting 70 % of the users and re-routing them to SO paths, we can achieve on average less than 5 % of relative TTT reduction with respect to the TTT in the UE-ref simulation. However, by selecting only 20 % of the users who have the largest potential decrease of PMC, i.e., diffPMC, we can already obtain more than 7.5 % of TTT reduction from the UE-ref simulation.

Figure 4 shows that the diffPMC is a good criterion for selecting targeted users. By selecting users according to their PMCs reduction value from UE-ref to SO-ref and moving them onto SO paths, the system TTT reduction is significantly higher than in the cases where the users are selected randomly.

Based on these subsection results, we consider that $\alpha^* = 20$ is the reference fraction for selecting targeted users. There are a total 13475 users in the network. The reference number of users that we select is then denoted as $M_{\text{target}} = 2696$. In addition, the TTT reduction obtained after moving 20 % of the users to their SO path, 7.60 %, is considered as a criterion of potential TTT reduction when evaluating the efficiency of our learning methods for user selection (Section 3.2) and user re-routing (Section 3.4.2).

3.4. First step results: selection of targeted users by predicting diffPMC based on other trajectory features

The results in the reference case in Section 3.2 show that we can obtain more than 7.6 % of TTT reduction w.r.t UE case, by only selecting 20 % of users who have the largest values of diffPMC. Their diffPMC are computed based actual UE and SO paths obtained in the reference scenarios. However, we aim to re-route users without knowing their SO paths ahead, and the actual diffPMCs are not accessible. TIn this subsection, we aim to approximate users' diffPMC based on network-related trajectory features via a supervised learning process, and then show that we can also reduce the system TTT significantly, by re-routing 20 % of the users who have the largest values of *learned* diffPMC. More precisely, we aim to select M_{target} users according to the diffPMC *learned* by a regression model, and

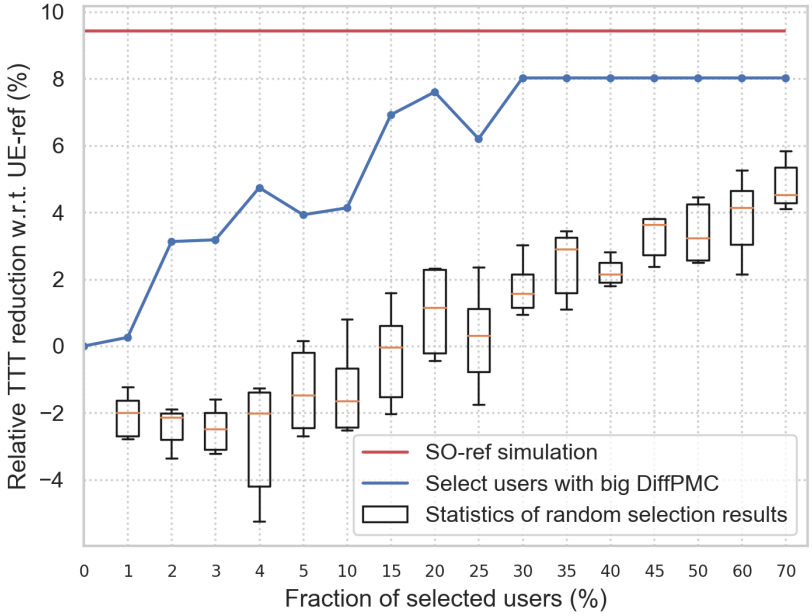


Figure 4: Relative TTT reduction with respect to TTT in UE-ref after selecting different fractions of users and re-routing them to SO paths. The reference relative TTT reduction from the UE-ref to the SO-ref simulation is 9.44 %. The black-line boxplots represent the statistics of TTT reduction after randomly moving users to their SO trajectories: (i) the orange lines represent median values, (ii) the boxes represent the interquartile ranges, and (iii) the upper and lower whiskers represent the 95^{th} and 5^{th} percentile.

move them on paths obtained from the SO-ref simulation in order to achieve a reduction of system TTT as close to that obtained in Section 3.2 as possible. This step corresponds to the 1^{st} step of our methodological framework, presented as the left branch of the method flow chart (Figure 1) in Section 2.

3.4.1. Data preparation

First, we prepare the data set for training the regression model. Note that we still use the results from the UE-ref and SO-ref simulations as the data set. Among the 13,475 users on the network, we select $M = 13127$ users' trajectory feature data as the sample set. The eliminated users have not finished their trips in our simulation horizon. Each user i has a pair of trajectories, $\mathcal{L}_i^{\text{UE}}$ and $\mathcal{L}_i^{\text{SO}}$ obtained from the two simulations, respectively. We compute the normalized PMC reduction of i by $y_i = \text{diffPMC}_i = \widehat{\text{PMC}}_i^{\text{UE}} - \widehat{\text{PMC}}_i^{\text{SO}}$ (ref. Equation (4) to Equation (7)). Then we save the computed diffPMCs in the target vectors to be approximated denoted by $\mathbf{y} \in \mathbb{R}^M$. For the same users, we also compute other dimensionless trajectory features based on the results of the UE-ref simulation: $\widehat{BC}_i$, G_i^{BC}, $\widehat{E}_i$, $G_i^{\text{dynamicBC}}$, Cap_i, η_i, μ_{l_i}. The values of these features define the feature vectors for user i, $\mathbf{x}^i = (x_1, \ldots, x_P)^T$ where $P = 7$. Then we define an $M \times P$ data matrix $\mathbf{X} = [\mathbf{x}^i \ldots \mathbf{x}^M]^T$, where $M = 13127$, which is the number of the sample size. Therefore, the objective is to use the known data set $\mathbf{X}$ and $\mathbf{y}$ in order to train a model:

$$f : \mathbb{R}^p \to \mathbb{R}$$
$$\mathbf{x} \to y = f(\mathbf{x}).$$

3.4.2. Training and evaluation of the regression model

Here, we train two regression models (LR and PR), f_{lin} and f_{poly}, based on the simulated data set $\mathbf{y}$ and $\mathbf{X}$. Then we compute the approximate $\mathbf{y}'_{\text{lin}} = f_{\text{lin}}(\mathbf{X})$ and $\mathbf{y}'_{\text{poly}} = f_{\text{poly}}(\mathbf{X})$. Therefore, we can sort the users by y' and select 20 % of the users with the biggest predicted differential of PMC. Afterward, we give these users SO-ref trajectories and re-run the DTA simulation, in order to evaluate the TTT reduction of the system.

Table 3: Scores of the LR and PR models for predicting the PMC reduction from UE to SO, based on all the seven network-related features. The correlation and RMSE are computed between (i) y (the actual PMC reduction obtained in UE-ref and SO-ref scenarios) and (ii) y' (the learned PMC reduction with trajectory features as attributes).

		Polynomial				Ref selection
	Linear	$d = 3$	$d = 4$	$d = 5$	**d = 6**	20 % bigDiffPMC
Correlation	0.1489	0.4022	0.4812	0.5516	**0.5701**	-
RMSE	0.1775	0.1522	0.1395	0.1263	**0.1226**	-
ΔTTT_{rel} w.r.t UE-ref (in %)	-8.6	4.13	2.38	5.55	**6.50**	7.60
ΔTTT_{rel} w.r.t SO-ref reduction (in %)	- 91.40	43.79	25.24	58.84	**68.42**	80.56

(*The relative TTT reduction is computed by Equation (29). The relative TTT with respect to reference is computed by Equation (30). The column titled by 20 % bigDiffPMC" in the table presents the case where we select 20 % of the users with the largest value of diffPMC selected and move to their SO paths.)*

Table 3 presents the results of this learning phase. The correlation and the root-mean-square error (RMSE) are computed between (i) the actual differential of PMC from UE-ref to SO-ref, and (ii) the predicted differential of PMC learned from $\mathbf{X}$, based on the LR model and the PR model with different degrees. The results show that the 6^{th} order PR model is able to efficiently predict the PMC reduction. By selecting 20 % of users with a large value of *predicted* diffPMC, we can achieve a 6.5 % reduction of TTT in comparison to the UE-ref simulation. According to Equation (30), this TTT reduction obtained in the test case reaches 68% of the reference reduction, which is the case where all the users travel under the SO condition.

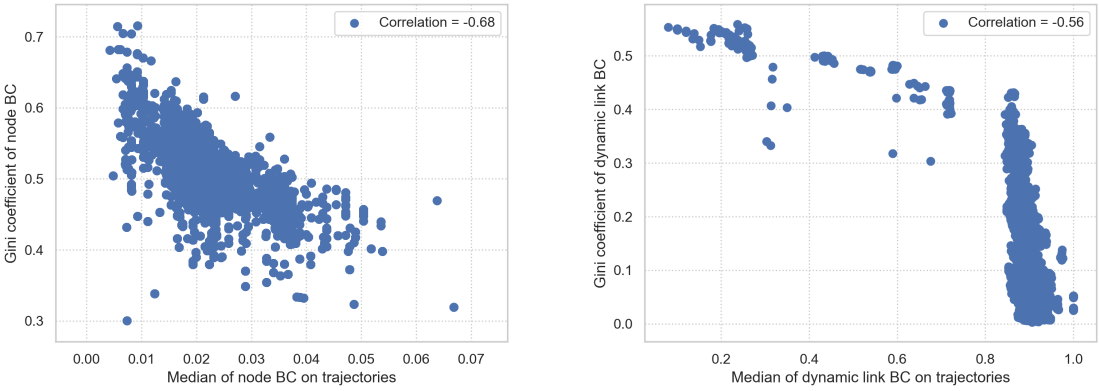
3.5. Second step results: re-routing strategies by predicting SO paths based on trajectory features

The results from Section 3.2 are obtained after re-routing the selected users to their *exact* SO paths resulting from the SO-ref simulation. The purpose of this section is to predict whether a path is an SO path or not, based on the known network-related trajectory features. The problem can be easily transformed into a classification task: to predict whether a path belongs to the class of SO paths or not, for all the possible paths between a given OD pair. The implementation and results presented in this subsection correspond to the 2^{nd} step of our methodological framework, presented as the right branch of the method flow chart (Figure 1) in Section 2.

3.5.1. Data preparation

Here we prepare the target vectors $\mathbf{y}$ and matrix of attributes $\mathbf{X}$, following the steps presented in Section 2.6.1. Let M_{all} denote the feasible paths for all users between all OD pairs. The matrix of the target vectors for training the NB model is composed of the actual classes of the paths: $\mathbf{Y}_{\text{all}} = [y_1, \ldots, y_i, \ldots, y_{M_{\text{all}}}]$ where $y_i = 0$ if $C(\mathcal{L}_i) = C_0$ and $y_i = 1$ if $C(\mathcal{L}_i) = C_1$. We also compute the network-related features of all these paths and construct the attribute matrix $\mathbf{X}_{\text{all}} = [\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_{M_{\text{all}}}]$, where $\mathbf{x}_i \in \mathbb{R}^5$ which means that we consider only five features as attributes, since we have 2 pairs of correlated feature. In fact, in order to satisfy the conditional independence assumption of NB classifier, we first carried out a correlation analysis with $\mathbf{X}$. The correlations in Figure 5 show that there are two pairs of trajectory features that are not independent: (i) $\widehat{BC}$ and G^{BC} (Figure 5(a)), (ii) $\widehat{E}_i$ and $G^{\text{dynamicBC}}$ (Figure 5(b)). If the correlation between two features is estimated as high, then this couple of features are dependent on each other and cannot both be taken into account when constructing the matrix of attributes in the NB training process. For example, Figure 5(a) shows that $\widehat{BC}$ and G^{BC} are highly correlated. Figure 5(b) shows that $\widehat{E}_i$ and $G^{\text{dynamicBC}}$ are highly correlated. Therefore, we do not take the two Gini coefficients into account when constructing the attribute vector for training g^{NB} in this work.

In our case study, there are a total of $M_{\text{all}} = 107385$ possible paths for all the users of all OD pairs in the Lyon6 network. To ensure that we have sufficient data to train $g^{\text{NB}}(\mathbf{x})$, we exclude the paths between OD pairs where fewer than 30 vehicles are traveling between them. The final number of remaining paths used in the learning process is $M_{\text{dataset}} = 45848$. Then, we separate the data set into two parts: the training set and testing set. Each of them consists of 50 % of the trajectories randomly selected among all the M_{dataset} paths. The corresponding class variables and feature vectors of these trajectories are denoted by $\mathbf{Y}_{\text{train}} = [y_1, \ldots, y_{M_{\text{train}}}]$, $\mathbf{X}_{\text{train}} = [\mathbf{x}_1, \ldots, \mathbf{x}_{M_{\text{train}}}]$, $\mathbf{Y}_{\text{test}} = [y_1, \ldots, y_{M_{\text{test}}}]$, and $\mathbf{X}_{\text{test}} = [\mathbf{x}_1, \ldots, \mathbf{x}_{M_{\text{test}}}]$.



(a) (b)

Figure 5: (a): The estimated correlation between the median node BC ($\widehat{BC}$) and the Gini coefficient of BC (G^{BC}); (b): The estimated correlation between median dynamic link BC ($\widehat{E_i}$) and the Gini coefficient of dynamic link BC (G^{BC}). (If the correlation between two features is estimated as high, then this couple of features are dependent on each other, and cannot both be taken into account when constructing the matrix of attributes in the NB training process.)

Table 4: Correction of class prediction by using the trained g^{NB}.

	$\widehat{\mathbf{Y}}_{\mathrm{test}}$ v.s. $\mathbf{Y}_{\mathrm{test}}$	$\widehat{\mathbf{Y}}_{\mathrm{all}}$ v.s. $\mathbf{Y}_{\mathrm{all}}$
Number of samples	22924	107385
% of correctly predicted classes	88.13 %	87.42 %

3.5.2. *Training and evaluation of the Naive Bayes classifier*

Here we use $\mathbf{Y}_{\mathrm{train}}$ estimate $Prob(y)$, σ_y and μ_y in Equation (28). With a known $\mathbf{X}_{\mathrm{train}}$, we can train $g^{\mathrm{NB}}(\mathbf{x})$. Note that we assume that the distribution of Y is the same for all the paths, so that $Prob(y)$, σ_y and μ_y remain the same for $\mathbf{Y}_{\mathrm{test}}$ and $\mathbf{Y}_{\mathrm{all}}$. We can therefore estimate $\widehat{\mathbf{Y}}_{\mathrm{test}} \approx g^{\mathrm{NB}}(\mathbf{X}_{\mathrm{test}})$ and $\widehat{\mathbf{Y}}_{\mathrm{all}} \approx g^{\mathrm{NB}}(\mathbf{X}_{\mathrm{all}})$. Then we compare the predicted class variables and the data set in order to evaluate the performance of the classifier. Table 4 presents the correction rate of the well-predicted class variables of $\mathbf{Y}_{\mathrm{test}}$ and $\mathbf{Y}_{\mathrm{all}}$. If a simulated UE path is correctly labeled as C_0 or a simulated SO path is correctly labeled as C_1 by the trained NB classifier, then the conclusion about the prediction for this path is *correct*. The percentage of correctly predicted classes in Table 4 is computed by dividing the number of paths whose classes are correctly predicted by the total number of paths of the data set.

3.6. *Results of combining two-step learning processes: selection and re-routing*

Recall that in the reference case presented in Section 3.2, the TTT of the system is reduced 7.6 % after re-routing 20 % of the users with the largest value of diffPMC to their exact SO paths. In this subsection, we investigate whether system performance can still be significantly improved after selecting a small part of the users according to their *estimated* diffPMC, and rerouting them to their *predicted* SO paths. More precisely, we evaluate the efficiency of the two – step learning methodology in terms of TTT reduction in comparison to the UE-ref scenario, as shown in *part-III* of Figure 1 in Section 2.1. We select 20 % of the users according to the diffPMC predicted by the PR model $f_{\mathrm{poly}}(\cdot)$ in Section 3.2. Instead of moving them into simulated SO paths, as presented in Section 3.2, we now reroute them on paths that are predicted to be SO paths according the NB classifier $g^{\mathrm{NB}}(\cdot)$ result in Section 3.5.2. Then we evaluate the TTT of this test scenario and compare it to the TTT in the UE-ref simulation.

First, for a selected user entering the network at time t, we identify their OD pair. Second, we search all the possible paths between this OD pair and compute the feature vector $\mathbf{x}$ for each alternative path. Then, we rely on

Table 5: Rerouting strategies for targeted users

No.	Case	Action
1	None of the alternative paths is labeld as SO path	Do not move the user
2	One and only one path is predicted as SO path	Move the user onto this path
3	More than one alternative paths are labeld as SO paths	Randomly select **one path** from the predicted SO paths, and move the user on it

$g^{\mathrm{NB}}(\mathbf{x})$ to predict whether a path is an SO-path or not. Since there may be more than one feasible path for a user, there are three cases for the rerouting strategy, shown in Table 5.

Among all the 2696 selected users, 641 users have at least one predicted SO path among their alternative paths of the same OD pair. These 641 users are finally targeted and moved to the predicted SO paths. There are some users whose trajectories are not changed according to the NB classifier. This may be due to two reasons: (i) the predicted SO paths are the same as the UE paths, and (ii) all the candidate paths of these users are predicted as non-SO paths. Therefore, we pre-define the paths for these 641 users and run DTA simulations while the other users continue traveling under the UE condition. Besides, in order to take into account the eventual estimation error caused by the random selection of predicted SO paths in the case of the 3^{rd} situation mentioned in Table 5, we carry out 12 independent DTA simulations. In each of these simulations, we randomly select only one predicted SO path for the selected users who have at least two *predicted* SO paths.

The final simulation results are presented in Figure 6. The results show that by selecting 20 % of the users with the largest *predicted* diffPMC, and moving them onto the *predicted* SO paths, we can achieve on average 5.9 % of total travel time in comparison to the reference UE simulation. This performance is significantly better than the BM scenarios, where we randomly select 20 % of the users and move them on their SO paths obtained from SO-ref. We also observe a small variation of TTT reduction in the 12 independent simulation scenarios. This infers that there might be more than one alternative path that has close characteristics compared to the SO paths.

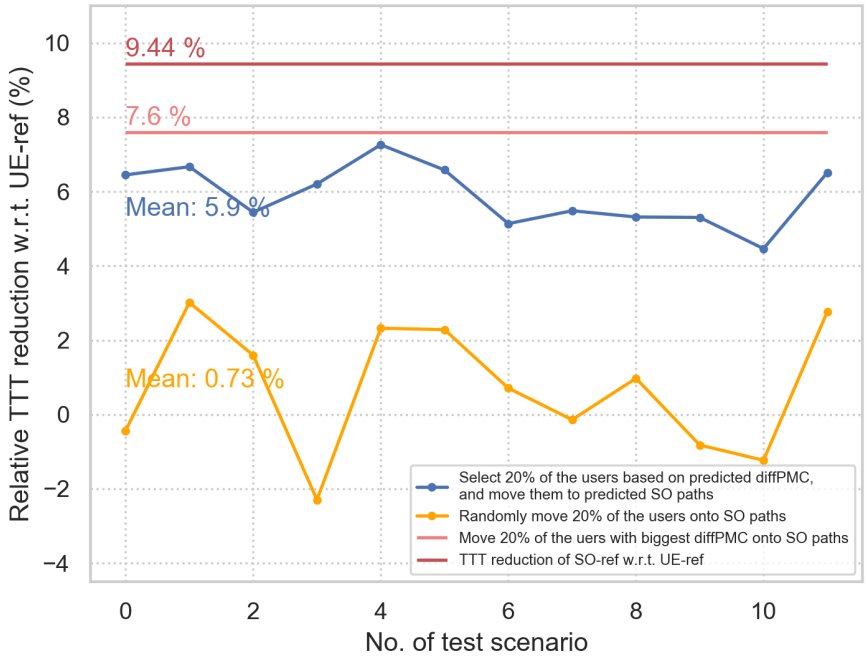


Figure 6: The relative TTT reduction with respect to UE-ref simulation for different test scenarios.

Furthermore, the final number of targeted users is 641, representing less than 5 % of the total users in the network. Nevertheless, by moving them onto the *predicted* SO paths, we achieve on average a TTT reduction of more than 62 % in comparison to the reference case, where all the users choose their paths under the SO condition. In addition, among all the users, there are about 68 % of them have decreased their individual travel time in the re-routing scenario, with respect to the UE-ref scenario. The average travel time saving is about 146 s. For the rest 32 % of users who have increased their travel time after rerouting, the average increased travel time per user is about 200 s.[0]RC→LL: For the users that have been re-routed, there are 3.6 % of them who have increased their individual travel time after being re-routed, but the average increased travel time is 40 s. This is an acceptable increase of individual travel time and therefore there would be no difficulties for user compliance if we aim to apply the method into practical cases. Moreover, the results show that thanks to the trained PR and NB models, we are able to select and re-route a small share of network users to make a significant improvement to the performance of the whole network based only on their network-related trajectories.

4. Conclusions and discussions

This work investigates network-related trajectory features to unravel the trips that contribute most to the system's under-performance. When such trips are identified, feature analysis also permits identifying the best alternatives in terms of routes to bring the system to its optimum.

Four features: node BC, dynamic link BC, the mean MFD capacity, and the mean distance between traffic lights, are identified as the key network-related features. The combination of these features defines the potential trips that contribute most to the system's under-performance. With these features as attributes, we build two learning models in order to (i) select the targeted trips that are critical for the system under-performance and (ii) re-route the users onto alternative paths in order to improve the whole system performance. We carry out a case study based on two reference DTA simulations for a middle-size network, from the perspectives of UE and SO. The microscopic traffic simulator (Symuvia) is used to obtain user trajectories and their corresponding network-related features. Note that the methodology proposed can be adapted to other DTA models and/or simulators because it uses the simulation results independently. In other words, as long as we are able to get full information on network-related trajectory features, we can apply the methods proposed to any other DTA simulators/platforms.

The application of our methodology to the real test case (Lyon6 network) shows that by selecting only 20 % of the trips and moving them onto *predicted optimal* paths; we can achieve an improvement of more than 62 % in network performance in comparison to the case in which all the users move from UE to SO paths. These results show that we can achieve a significant improvement in network performance by targeting a small share of users and moving them to *predicted* alternative paths, based only on their trajectory features and/or regular travel time information under the UE condition thanks to the learning models we propose.

This work opens various perspectives. The analyzed feature attributes can be obtained based on intrinsic features of the network (such as graphical characteristics, the spatial distribution of traffic lights), and/or a simple traffic monitoring system. Therefore, it would be possible to apply the proposed methodological framework to real-life practice if we have GPS trajectory data and network characteristics. Then we could re-route a small share of network users in the real-world via a centralized route planning system by providing users route guidance to reduce traffic congestion throughout the whole network. In particular, this real-world application would be promising if connected autonomous vehicles (CAVs) represent a large share of network users in the future since it would be easier and more efficient to re-route CAVs, which are already monitored by a centralized system.

In addition, we can consider more than one criterion to evaluate the improvement of system performance. In this work, we only consider minimizing the system's total travel time. In future works, we could also consider adding environmental criteria, such as minimizing the total air pollutant emissions of the whole system. Moreover, we can still improve the performance of model training, concerning the learning model of the user selection step, the 6^{th} order of the polynomial regression model might be heavy if we deal with a data set with high-dimensional data. Other supervised learning models can also be investigated, such as Neural Network or Gaussian process regression with different kernels. For the classifier training, one direction can be to investigate better classification models to unravel SO path characteristics in order to avoid the cases (i) where there are than 1 predicted alternative paths or (ii) there is no predicted path at all, for a selected user. One possibility is to use an imbalance classification process because we have an imbalanced data set of feasible paths, where there are much more non-SO paths than SO paths.

5. Acknowledgements

This project has received funding from the European Research Council (ERC) in the framework of the European Union's Horizon 2020 research and innovation program (Grant agreement No. 646592 – MAGnUM project).

References

Ameli, M., Lebacque, J.-P., Leclercq, L., 2020a. Cross-comparison of convergence algorithms to solve trip-based dynamic traffic assignment problems. Computer-Aided Civil and Infrastructure Engineering 35 (3), 219–240.

Ameli, M., Lebacque, J.-P., Leclercq, L., 2020b. Improving traffic network performance with road banning strategy: A simulation approach comparing user equilibrium and system optimum. Simulation Modelling Practice and Theory 99, 101995.

Ameli, M., Lebacque, J.-P., Leclercq, L., 2020c. Simulation-based dynamic traffic assignment: Meta-heuristic solution methods with parallel computing. Computer-Aided Civil and Infrastructure Engineering 35 (10), 1047–1062.

Beckmann, M., McGuire, C. B., Winsten, C. B., 1956. Studies in the economics of transportation. Tech. rep.

Bellocchi, L., Bellagha, M. H., Latora, V., Geroliminis, N., 2020. A measure of dynamic efficiency for multimodal interconnected urban systems. In: Transportation Research Board 99th Annual Meeting (TRB2020). TRB.

Braess, D., Nagurney, A., Wakolbinger, T., 2005. On a paradox of traffic planning. Transportation science 39 (4), 446–450.

BUREAU, O. P. R., 1964. Traffic assignment manual. US Department of Commerce.

Cameron, A. C., Trivedi, P. K., 2005. Linear Models. Cambridge University Press, p. 65âĂŞ115.

Chen, R., Leclercq, L., 2019a. Characterizing system optimum by trajectory data analysis. In: 22nd EURO Working Group on Transportation Meeting (EWGT 2019). EWGT.

Chen, R., Leclercq, L., Sep. 2019b. Unravelling System Optimum Structure by trajectory data analysis. In: hEART 2019, 8th Symposium of the European Association for Research in Transportation. Budapest, Hungary, p. 9p, hEART 2019, 8th Symposium of the European Association for Research in Transportation, Budapest, HONGRIE, 04-/09/2019 - 06/09/2019.
URL https://hal.archives-ouvertes.fr/hal-02436879

Chickering, D. M., 1996. Learning bayesian networks is np-complete. In: Learning from data. Springer, pp. 121–130.

Chickering, D. M., Heckerman, D., Meek, C., 2004. Large-sample learning of bayesian networks is np-hard. Journal of Machine Learning Research 5 (Oct), 1287–1330.

Çolak, S., Lima, A., González, M. C., 2016. Understanding congested travel in urban areas. Nature communications 7, 10793.

Freeman, L. C., 1977. A set of measures of centrality based on betweenness. Sociometry, 35–41.

Girvan, M., Newman, M. E., 2002. Community structure in social and biological networks. Proceedings of the national academy of sciences 99 (12), 7821–7826.

Gonzalez, M. C., Hidalgo, C. A., Barabasi, A.-L., 2008. Understanding individual human mobility patterns. nature 453 (7196), 779.

Kotsiantis, S. B., Zaharakis, I., Pintelas, P., 2007. Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer engineering 160 (1), 3–24.

Laval, J. A., Castrillón, F., 2015. Stochastic approximations for the macroscopic fundamental diagram of urban networks. Transportation Research Procedia 7, 615–630.

Leclercq, L., Laval, J. A., Chevallier, E., 2007. The lagrangian coordinates and what it means for first order traffic flow models. In: Transportation and Traffic Theory 2007. Papers Selected for Presentation at ISTTT17Engineering and Physical Sciences Research Council (Great Britain) Rees Jeffreys Road FundTransport Research FoundationTMS ConsultancyOve Arup and Partners, Hong KongTransportation Planning (International) PTV AG.

Leclercq, L., Verchier, A., Krug, J., Menendez, M., 2016. Investigating the performances of the method of successive averages for determining dynamic user equilibrium and system optimum in manhattan networks. In: DTA2016, 6th International Symposium on Dynamic Traffic Assignment. pp. 1–p.

Lopez, C., Leclercq, L., Krishnakumari, P., Chiabaut, N., Lint, H., 2017. Revealing the day-to-day regularity of urban congestion patterns with 3d speed maps. Scientific Reports 7 (1), 14029.

Ma, X., Yu, H., Wang, Y., Wang, Y., 2015. Large-scale transportation network congestion evolution prediction using deep learning theory. PloS one 10 (3), e0119044.

Mahmassani, H. S., Peeta, S., 1993. Network performance under system optimal and user equilibrium dynamic assignments: implications for ATIS. Transportation Research Board.

Pearl, J., 2014. Probabilistic reasoning in intelligent systems: networks of plausible inference. Elsevier.

Peeta, S., Mahmassani, H. S., 1995. System optimal and user equilibrium time-dependent traffic assignment in congested networks. Annals of Operations Research 60 (1), 81–113.

Roughgarden, T., Tardos, É., 2002. How bad is selfish routing? Journal of the ACM (JACM) 49 (2), 236–259.

Saeedmanesh, M., Geroliminis, N., 2016. Clustering of heterogeneous networks with directional flows based on âĂIJsnakeâĂİ similarities. Transportation Research Part B: Methodological 91, 250–269.

Seal, H. L., 1967. Studies in the history of probability and statistics. xv the historical development of the gauss linear model. Biometrika 54 (1-2), 1–24.

Sheffi, Y., 1985. Urban transportation networks.

Smith, M. J., 1979. The existence, uniqueness and stability of traffic equilibria. Transportation Research Part B: Methodological 13 (4), 295–304.

Taheri, S., Yearwood, J., Mammadov, M., Seifollahi, S., 2014. Attribute weighted naive bayes classifier using a local optimization. Neural Computing and Applications 24 (5), 995–1002.

van Essen, M., Thomas, T., van Berkum, E., Chorus, C., 2016. From user equilibrium to system optimum: a literature review on the role of travel information, bounded rationality and non-selfish behaviour at the network and individual levels. Transport reviews 36 (4), 527–548.

Wang, H., Hernandez, J. M., Van Mieghem, P., 2008. Betweenness centrality in a weighted network. Physical Review E 77 (4), 046105.
Wang, P., Hunter, T., Bayen, A. M., Schechtner, K., González, M. C., 2012. Understanding road usage patterns in urban areas. Scientific reports 2, 1001.
Wardrop, J. G., 1952. Road paper. some theoretical aspects of road traffic research. Proceedings of the institution of civil engineers 1 (3), 325–362.
Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Philip, S. Y., et al., 2008. Top 10 algorithms in data mining. Knowledge and information systems 14 (1), 1–37.
Yildirimoglu, M., Kahraman, O., 2018. Searching for empirical evidence on traffic equilibrium. PloS one 13 (5), e0196997.
Youn, H., Gastner, M. T., Jeong, H., 2008. Price of anarchy in transportation networks: efficiency and optimality control. Physical review letters 101 (12), 128701.