Bicycle Travel Time Estimation via Dual Graph-Based Neural Networks

Ting Gao, Winnie Daamen, Elvin Isufi, *Senior Member, IEEE*, and Serge P. Hoogendoorn

***Abstract*—In urban centers, cycling is increasingly popular as an eco-friendly transportation mode and a short-distance transport option, driving higher demand for accurate bicycle travel time estimation. Policymakers need to understand bicycle traffic for urban traffic management and sustainable transport promotion, while cyclists benefit from better route planning and improved network efficiency. However, urban bicycle travel time estimation has not received as much attention as car traffic estimation and presents several challenges: 1) Limited availability of structural cycling data, which can be inaccessible due to privacy concerns and/or severely biased by user demographics. 2) The diverse and complex behaviors of cyclists. 3) The lack of strict road constraints for cyclists and frequent rule violations, complicating the model definition of a comprehensive cycling infrastructure network. This paper presents the first study on urban bicycle travel time estimation using GPS tracking data. Leveraging graph-based deep learning's ability to learn from topological network information, we introduce the Dual Graph-based approach for bicycles (DG4b), which employs two parallel encode-process-decode pipelines: one for a shared undirected road network graph to capture intrinsic road characteristics, and another for a directed trip-specific graph reflecting unique trip features. The outputs are combined to estimate road segment speeds and overall trip travel time. When applied to a real-world dataset from Berlin, our method shows superior accuracy and reliability compared to baseline models, while maintaining low complexity. Our approach provides a novel perspective on integrating bicycling-specific characteristics and aims to inspire more future research in bicycle-related traffic estimation.**

***Index Terms*—Travel time estimation, bicycles, GCNNs, GPS tracking data.**

I. Introduction

URBAN cycling is gaining popularity as a response to the limitations of automobile-dependent transport, including congestion, parking problems, pollution, and resource depletion [1]. Governments and municipalities, particularly in cycling-friendly countries such as the Netherlands, Denmark, and Germany, are promoting this shift [2]. As cycling popularity grows, accurately estimating bicycle travel time has become increasingly important for congestion management, delay reduction, and safety. For cyclists, travel time estimation supports better route planning and departure timing [3]. At the municipal level, accurate bicycle travel time estimation provides critical input for transportation planning and decision-making. When bicycle travel times become less competitive relative to car or public transit options, bike commuting declines [4], ultimately undermining cities' sustainability and congestion reduction goals. Understanding these travel patterns helps designing more efficient urban networks that promote sustainable transportation [5]. Furthermore, bicycle travel time estimation informs the optimal placement of shared bicycle stations [6], traffic signal coordination to minimize cyclist-vehicle conflicts [7], and the identification of low-efficiency road segments that require targeted infrastructure improvements to enhance the cycling experience [8]. Despite its importance, bicycle travel time estimation remains underexplored compared to that of cars. This study addresses this gap by introducing an approach tailored for bicycles utilizing GPS tracking data.

A common confusion between travel time estimation and prediction is highlighted in [9]. Travel time prediction forecasts future trip durations using current and historical traffic data, while travel time estimation reconstructs travel times for completed trips to provide an overview of traffic conditions on specific road segments during a certain time frame [10]. In this study, we focus on travel time estimation.

Estimating urban bicycle travel time is however challenging due to data limitations and cyclist behavior variability. While GPS tracking data captures individual trip trajectories and travel times with good spatial coverage, it is prone to positioning errors, user biases, and low penetration rates. Additionally, bicycle GPS data is less readily available than car data due to insufficient structural collection efforts, though interest in gathering this data is increasing. For example, the Netherlands, which has the highest bicycle mode share, launched the "Talking Bike" campaign to collect cyclists' GPS data [11]. In Berlin, a project called SimRa [12] has been launched to collect bicycle travel data in order to identify when and where cyclists are most at risk, as well as to determine the main routes of bicycle traffic. However, even when available, such data often lack comprehensive trip details (e.g., traveler ID) due to privacy concerns [13]. In this study, we utilize anonymous data from the SimRa campaign to estimate bicycle travel time.

Received 18 November 2024; revised 10 May 2025 and 19 September 2025; accepted 28 October 2025. Date of publication 4 December 2025; date of current version 26 December 2025. This work was supported by the "EMERALDS" Project through the Horizon Europe Research and Innovation Programme under Grant 101093051. The Associate Editor for this article was L. Li. *(Corresponding author: Ting Gao.)*

Ting Gao, Winnie Daamen, and Serge P. Hoogendoorn are with the Department of Transport and Planning, Faculty of Civil Engineering and Geosciences, Delft University of Technology, 2628 CN Delft, The Netherlands (e-mail: t.gao-1@tudelft.com; w.daamen@tudelft.nl; s.p.hoogendoorn@tudelft.nl).

Elvin Isufi is with the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: e.isufi-1@tudelft.nl).

Digital Object Identifier 10.1109/TITS.2025.3633150

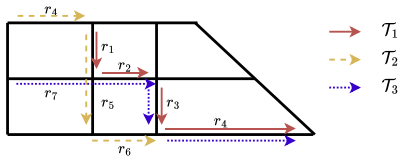


Fig. 1. Problem demonstration: $\mathcal{T}_1$ represents a trip with missing travel time, while $\mathcal{T}_2$ and $\mathcal{T}_3$ are observed trips with complete travel times and trajectory data. Road segment sequences for these observed trips are derived from GPS points via map-matching techniques [16]. Generally, only a part of the trajectories in $\mathcal{T}_2$ and $\mathcal{T}_3$ overlap with $\mathcal{T}_1$.

Cyclist behavior variability and unpredictability pose another major challenge in travel time estimation. Besides traffic lights, personal stops (e.g., shopping or deliveries) inflate travel time in GPS data, but these activity-related delays should be excluded as they do not reflect actual travel time. Cyclists also tend to violate traffic rules more frequently than car drivers, often using non-designated paths like footways [14] or traveling against traffic flow [15], making it impractical to rely solely on the designated cycleway network.

The travel time estimation problem is illustrated in Figure 1. In practice, some road segments planned by a trip may not be covered by any trajectories in the observed GPS dataset. To generalize travel time estimates from observed trips (e.g., $\mathcal{T}_2$, $\mathcal{T}_3$) to target trip trajectories (e.g., $\mathcal{T}_1$), extensive studies have been conducted for car traffic [9], [17]. However, no such studies have specifically addressed bicycles, which is non-trivial due to a series of challenges that we elaborate in the sequel. A straightforward approach is to derive travel time estimates from trips with overlapping trajectories or similar origin-destination information [18], [19], [20], [21], [22], [23], [24], [25]. However, this method can be inaccurate for trajectories involving rarely recorded roads and is significantly affected by outlier trips.

When approaching travel time estimation as a regression problem based on travel trajectory and start time in car traffic, deep learning methods offer a flexible solution. Some recent methods separate the trajectory from the road network and model it directly as a sequence learning problem [26], [27], [28], [29], [30], [31], while others incorporate the intrinsic relationships and interactions between road segments, leveraging the road network's topological structure for graph-based deep learning [32], [33].

Nevertheless, employing car traffic solutions directly to bicycles is not preferred for three main reasons: (1) Incompatible GPS data quality: Car traffic algorithms rely on GPS data with uniform sampling intervals or high sampling rates [26], [29], which is often not available for bicycles. (2) Volatile travel times: bicycle travel times can vary significantly between cyclists due to individual behavior, such as varying effort levels and propulsion differences—higher speeds demand greater physical exertion, in contrast to motor vehicles, which typically adhere to speed limits. Additionally, personal stops further skew the travel time distribution, posing challenges for certain algorithms [27]. (3) Lack of cycling-specific travel behavior integration: Car-focused algorithms assume well-defined car lanes, but cycling networks are more complex due to frequent rule violations [14], [15]. Additionally, these algorithms overlook unique cycling behaviors, such as limited speed ranges.

To address these issues, we propose DG4b (Dual Graph approach for Bicycle travel time estimation), a method specifically designed for bicycles. The dual graph refers to two interconnected yet distinct graph structures—one for the static road network and another for the dynamic trip-specific features—capturing both intrinsic road characteristics and individual cycling trajectories. Graph-based deep learning methods are selected for their advantages in representing and processing topological structures [34]. The main contributions of our work are:

1) **Pioneering Bicycle Travel Time Estimation:** This is the first work to address bicycle travel time estimation using urban GPS tracking data, filling the gap between the growing demand for bicycle travel time estimation and the lack of related studies. We integrate bicycle travel characteristics in our approach by accounting for variations in cycling behaviors and preferences, including preferred routes and typical speed limit.
2) **Minimal Input Requirements and Multi-Task Outputs:** Our approach requires only minimal input data, with no constraints on data sampling intervals or rates. It uses the road network, the road segments traveled, a peak/off-peak hour indicator, and the travel time. Besides estimating travel time, our method also outputs road segment speeds as an intermediate output.
3) **First Dual Graph Design for Bicycle Travel Time Estimation:** We introduce a novel deep neural network architecture featuring two parallel graph-based pipelines. The first pipeline operates on an undirected, static road network graph to capture consistent road characteristics across all trips. The second pipeline processes a directed graph of traveling road segments, handling the specific characteristics of each individual cycling trip. The combined outputs of both pipelines provide speed likelihood estimates for each road segment. The separation of the complex road network from the simpler trip-specific graph enables simultaneous processing of multiple trips.
4) **Real-World Case Study in Berlin**: We apply our method to a real-world dataset, SimRa, collected from the Berlin urban area. Given the lack of established benchmarks in bicycle travel time estimation, we consider five baseline models and include a common car-based model for comparison. Our method demonstrates the best overall performance. Additionally, we corroborate our model on the real-world Dutch Talking Bike dataset to demonstrate its generality under distribution shifts and biased trajectories, such as those from food delivery bike users. Our code is available on GitLab[1] in the spirit of open science.

The paper is organized as follows. Section II reviews the background literature. Section III provides the preliminary definitions and formulates the research problem. Section IV presents the DG4b method, including data features and model components. Section V applies our method to real-world data,

[1]https://gitlab.tudelft.nl/T.Gao-1/dg4b

comparing its performance with baseline models, and includes ablation study, model complexity evaluation, and generality validation. Section VI discusses the novelty of our method and the constraints on modeling choices imposed by the limitations of available bicycle data. Finally, Section VII concludes the paper.

II. Background

This section reviews road network modeling methods and deep learning techniques for sequential data. We extend both discussions to travel time prediction, as it is more studied in the literature and shares key similarities with travel time estimation in terms of network modeling and data processing.

A. Urban Road Network Modelling

Road network modeling extracts spatial information. Before the widespread adoption of graph-based methods, multivariate time series models [35] and Convolutional Neural Networks (CNNs) [36] were commonly used. Multivariate time series models capture temporal correlations across regions but struggle with nonlinear spatial dependencies, while CNNs, though effective for large datasets, are constrained by their Euclidean structure and are not well-suited for irregular road networks.

Graph-based methods have become essential in capturing the topological structure of road networks [37]. They directly model variations between different road types, providing a more nuanced understanding of road networks. Their flexibility and capability to handle non-Euclidean data structures made them popular in urban travel time analysis [35] and well-suited for modeling road networks in our research. Graph-based methods can encounter significant computational challenges when processing multiple large graphs in a single batch. To address this, our approach processes smaller trip-specific travel graphs in parallel with the complete large road graph, allowing us to share the complete road graph across all trips in a batch and reduce computational costs.

In graph-based methods, line graphs are frequently employed, where road segments are modeled as nodes and intersections as edges [38], [39], [40], [41]. Alternatively, some studies reverse this approach, modeling intersections as nodes and road segments as edges [42]. Other methods explore separate graphs for road segments and intersections [43], [44]. The choice of modeling method depends on the road structures that need to be captured and the resulting graph complexity. Line graphs are useful for differentiating intersections that connect roads in different directions, leading to a higher number of edges. Conversely, modeling intersections as nodes and road segments as edges can simplify the graph topology. When considering only the impacts from neighboring road segments, the adjacency matrix based on road segment connections captures direct spatial relationships, making line graphs a more effective option.

Trip sequences can also be represented as a directed graph of road segments [45], [46], which is chosen in our method because it clearly represents the trip path and direction, simplifying the integration of spatial and temporal information. By projecting trip trajectories onto the road network, we create a trip-specific directed graph. While directed graphs may struggle with long-range dependencies due to their focus on immediate connections, they facilitate information propagation from neighboring trip segments without relying on memory mechanisms.

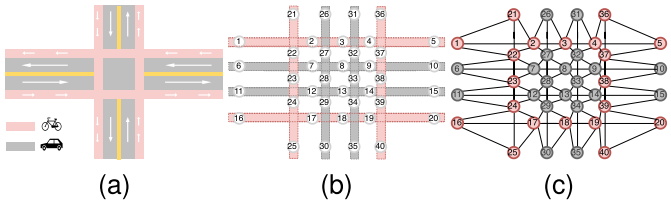


Fig. 2. Illustration of a line graph representation of an urban road intersection using OSM linestrings, allowing cyclists to freely choose their routes. (a) Intersection with bidirectional bicycle lanes (pink) and car lanes (grey). (b) Road segments are represented as linestrings in OpenStreetMap (OSM). (c) Road segments are represented as nodes in the line graph.

B. Deep Learning for Sequential Data

Travel time estimation and prediction both involve sequential data but differ in focus: estimation uses spatial sequences of ordered road segments, while prediction relies on temporal sequences of traffic conditions as time series data. Although both tasks utilize similar sequence data learning techniques, travel time prediction has been studied more extensively. Recurrent Neural Networks (RNNs) and their variants, such as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs), are employed in various studies [38], [47]. LSTMs and GRUs mitigate the vanishing gradient problem of traditional RNNs and improve the learning of long-term dependencies. However, these include complex gating mechanisms introducing computational overhead, as each gate needs to manage interactions between the input, previous hidden state, and cell state. With the rise of language models, attention mechanisms [39], [42] improve performance by focusing on important features within long sequences, although they also add to computational demands and more extensive hyperparameter tuning.

III. Problem Definition

In this section, we provide the necessary preliminaries to formulate the research problem.

Definition 1 (Road Line Network): We adopt a line graph approach as we focus on road segment interactions and it facilitates the road network representation. The road network is represented by an undirected line graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of links (edges). Each node $v_i \in \mathcal{V}$ represents a road segment, and each link $e_{ij} \in \mathcal{E}$ represents an intersection connecting two adjacent road segments from v_j to v_i.

Our extracted road network graph accounts for all roads in the network, recognizing that cyclists often disregard traffic rules by using roads designated for other traffic modes [14] and OpenStreetMap inaccuracies [16]. This results in a larger network compared to typical car traffic networks. In Figure 2, we present an example of constructing a line graph for an

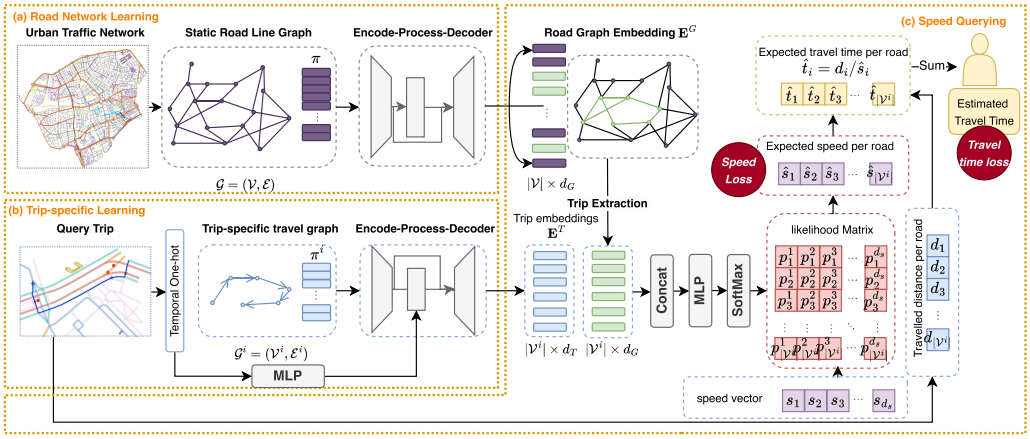


Fig. 3. DG4b model overview. The model consists of three main components: (a) the Road Network Learning component, which generates an embedding for the entire road network, (b) the Trip-Specific Learning component, which produces trip-specific embeddings for each road segment traveled during a trip, and (c) the Speed Querying component, which retrieves road segment embeddings from the comprehensive road graph embedding, combines them with trip-specific embeddings, and transforms them into a likelihood matrix corresponding to different travel speeds. The estimated speed for each road segment is derived from this matrix, and the final estimated travel time is obtained by summing the expected travel times for all road segments. The model's loss function comprises two parts: speed loss and travel time loss.

intersection. Each road segment is represented by a node, and we opt for an undirected graph structure to accommodate cyclists riding against the designated traffic flow [15].

Definition 2 (Trip): A trip $\mathcal{T}_i$ represents an itinerary from a departure location point to an endpoint in the network. The raw trip data consists of a sequence of GPS points and timestamps. After map-matching, the trip can be described by five attributes: $\mathcal{T}_i = \{\mathcal{P}^i, \mathcal{V}^i, \mathcal{R}^i, t^i, y^i\}$, where $\mathcal{P}^i = \{p_1^i, \cdots, p_m^i\}$ represents the sequence of raw GPS points with each point a two-dimensional vector containing longitude and latitude coordinates; $\mathcal{V}^i = \{v_1^i, \cdots, v_n^i\}$ denotes the sequence of road segments (nodes) traveled during the trip $\mathcal{T}_i$; and $\mathcal{R}^i = \{r_1^i, \cdots, r_n^i\} \in [0, 1]^n$ represents the traveled fraction of each road segment. Typically, each element falls within $[0, 1]$, as a trip could start or end in the any part of a road segment. Additionally, t^i is the starting time, and y^i is the total travel time.

Definition 3 (Bicycle Travel Time Estimation): Given a trip $\mathcal{T}_i$ whose travel time attribute is missing, we represent its route and departure information as $q_i = (\mathcal{P}^i, \mathcal{V}^i, \mathcal{R}^i, t^i)$. The training set of trips with complete attribute information is denoted as $\mathcal{D}$. The task aims to estimate the travel time y^i by learning a function f based on the road network $\mathcal{G}$ and the observations in the training set $\mathcal{D}$. This can be formulated as:

$$f(q_i|\mathcal{D}, \mathcal{G}) \rightarrow y^i. \tag{1}$$

IV. Methodology

In this section, we introduce the Dual Graph-based neural networks for Bicycle travel time estimation (DG4b). Bicycle travel time estimation considers two key factors: static road infrastructure and dynamic cyclist behavior. The road infrastructure remains consistent for all users, making it natural to model road segments as nodes and physical connections as edges in a network graph. This representation captures both segment-specific characteristics and spatial dependencies between neighboring segments.

Meanwhile, cyclist behavior is inherently dynamic, with individuals selecting different routes and departure times. This introduces trip-specific elements that a static road network alone cannot capture. To address this, we construct a directed trip-specific graph where each node represents a road segment along a particular trip. The state of each segment is influenced not only by its own attributes but also by previously traveled segments, capturing sequential dependencies such as accumulated fatigue affecting speed on subsequent segments.

To effectively model both global network structure and individual trip dynamics, we employ a Dual Graph Neural Network. As illustrated in Figure 3, our method processes two distinct but complementary graphs in parallel:

- The *road network learning component* processes the first graph—a universal road network graph encoding static infrastructure features shared across all trips.
- The *trip-specific learning component* processes the second graph—a directed trip-specific graph focusing on individual trip attributes.
- The *speed querying component* then integrates the learned representations from both graph networks to estimate travel speed for each road segment. The total travel time is calculated by summing the travel times across all segments.

The data scarcity in real-world scenarios has motivated our use of graph-based machine learning, which offers two key advantages: (1) Despite the low number of visits per road segment (node), we can extract common patterns across nodes with shared features, as all nodes share the same embedding transformation matrix. (2) The message propagation mechanism in graph-based machine learning naturally incorporates the graph structure, enriching the model with additional knowledge.

This dual graph approach overcomes fundamental limitations of single-graph alternatives. A single-graph representation would either fail to adequately distinguish between infrastructure patterns and trip-specific behaviors or require

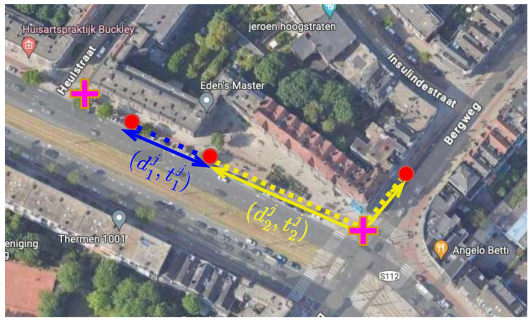


Fig. 4. Illustration of scenarios where consecutive GPS points are located on different road segments of trip $\mathcal{T}_j$. The red dots indicate mapped GPS points, while the road segment of interest is the one between the two crosses. The speed is assumed to be constant between consecutive GPS points.

an overly complex structure that becomes computationally unwieldy. As shown in Figure 2, cycling networks encompass various road types (e.g., pedestrian paths, car roads), leading to more complex networks than those for motorized traffic. By processing both graphs in parallel, our approach efficiently integrates spatial network information with trip-specific sequential dependencies, improving predictive accuracy while ensuring computational scalability.

In the following, we describe the data inputs of our model, provide a detailed explanation of each of the three components, and introduce the loss function that will be optimized.

A. Data Features

Our method requires three data inputs: a static road line graph, a trip-specific travel graph, and a trip temporal one-hot encoding. The latter two serve as inputs for the *trip-specific learning component*. In this subsection, we provide a detailed explanation of these inputs.

1) Static Road Line Graph: The road line graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represents road segments as nodes and intersections as edges. For node i, its node feature comprises the road length l_i, geolocations (longitude and latitude) of the starting and ending points, average observed passing speed s_i, road availability for bicycles (as a one-hot vector), direction information (as a one-hot vector), and the presence of traffic lights information (as a one-hot vector).

The average passing speed is calculated from observed trips with complete travel time information. Using map-matched GPS trajectories, the passing speed over a road segment for a trip can be derived through scaling and allocation [42], assuming that the traveling speed remains constant between two consecutive GPS points. As illustrated in Figure 4, if a road segment i is divided into m consecutive sub-segments, with each sub-segment having distance d_k^j and travel time t_k^j for $k = 1, 2, \ldots, m$, then the passing speed s_i^j of trip $\mathcal{T}_j$ over road segment i can be derived as:

$$s_i^j = \frac{\sum_{k=1}^{m} d_k^j}{\sum_{k=1}^{m} t_k^j}. \tag{2}$$

If no observed trips have traversed a road segment, we fill the missing data with the average speed of all segments. Additionally, we incorporate the road availability classification for bicycles as described in [16], which reflects different levels of comfort and safety perception for cyclists. Furthermore, we use one-hot encoding to indicate whether a road segment is unidirectional or bidirectional, as well as the presence of traffic lights on a road segment.

2) Trip-Specific Travel Graph: A directed trip-specific travel graph $\mathcal{G}^i = (\mathcal{V}^i, \mathcal{E}^i)$ is used to describe the trajectory of trip $\mathcal{T}_i$. In this graph, $\mathcal{V}^i$ represents the ordered set of traveled road segments (nodes), with the graph size $N_i = |\mathcal{V}^i|$. The edges $\mathcal{E}^i$ connect adjacent nodes to indicate the travel direction. To construct a trip-specific graph, road segments that are traveled multiple times are distinguished within the node set $\mathcal{V}^i$. For example, in the trip-specific travel graph $\mathcal{G}^i$ following the road sequence $r_1 \to r_2 \to r_3 \to r_1$, there are four ordered nodes: v_1^i, v_2^i, v_3^i, and v_4^i. Nodes v_1^i and v_4^i correspond to the same real-world road segment but are distinguished with different nodes in the trip-specific graph due to their distinct positions in the trip sequence. This distinction helps to capture the temporal dependencies and directionality of the trip, ensuring that road segments revisited later in the trip are treated differently from their initial traversal.

For each segment node $v_j^i \in \mathcal{V}^i$ related to trip $\mathcal{T}_i$, the node features are: the traveled road ratio r_j^i, the remaining distance ratio $d_j^i \in (0, 1]$, and the remaining segment ratio $n_j^i \in (0, 1]$. The remaining distance ratio d_j^i is the ratio of the remaining distance from the current node to the end of the trip, relative to the total travel distance, and the remaining segment ratio n_j^i is the completed proportion of the trip in terms of the number of road segments. They are defined as:

$$d_j^i = \frac{\sum_{k=j}^{N_i} l_k r_k^i}{\sum_{k=1}^{N_i} l_k r_k^i}, \tag{3}$$

$$n_j^i = 1 - \frac{j-1}{N_i}. \tag{4}$$

3) Temporal One-Hot Encoding: Temporal features, such as the time of day and day of the week, are widely used in research to capture temporal patterns related to varying traffic conditions (e.g., [27], [32]), with some studies also incorporating day of the year as a feature [26], [28]. These features are represented using one-hot encoding to learn distinct representations for different times. However, a more refined temporal categorization increases the data demand within each category, which can be problematic when the dataset is sparse. Given the high sparsity of our bicycle datasets, we use only a peak hour/off-peak hour index. Peak hours are defined based on automated traffic counts on German highways [48]. We apply the same time intervals for weekdays, specifically from 6:00 am to 8:00 am and from 3:00 pm to 6:00 pm. The one-hot encoding for peak hours reflects traffic congestion levels during these times.

B. Road Network Learning Component

The objective of the *road network learning component* is to generate a road graph embedding that covers all road segments and is consistent across all trips. This consistency captures the invariant nature of the road network, ensuring better

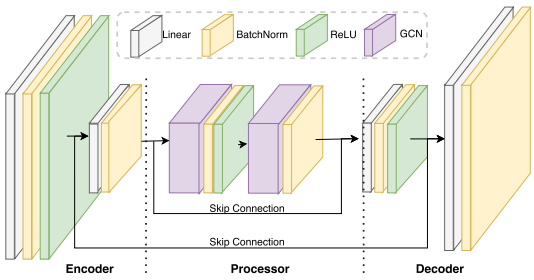


Fig. 5. Our implemented encode-process-decode structure. The encoder and decoder are both two-layer linear networks, while the processor is a Graph Convolutional Neural Network (GCNN). Skip connections are utilized between the encoder and decoder to enhance model expressiveness.

generalization across trips. Additionally, this approach allows the embeddings to be precomputed and reused, enhancing the model's transferability to different settings and scalability.

As illustrated in Figure 3, we implement an encode-process-decode structure, a widely adopted framework in deep learning approaches, offering flexibility to integrate various components. In our method, the encoder takes the static road line graph as input (detailed in Subsection IV-A) and generates node-wise embeddings capturing the intrinsic characteristics of the road network. The processor enhances these embeddings by integrating information from neighboring road segments. The decoder uses the processed embeddings to create a node-wise road graph embedding. We adopt the encode-process-decode configuration illustrated in Figure 5. Specifically, both the encoder and decoder implement two-layer linear dense networks, while the processor utilizes a Graph Convolutional Neural Network (GCNN).

1) Encoder: The encoder takes the graph feature $\mathbf{X}^e \in \mathbb{R}^{|\mathcal{V}|\times d_0}$ as input, where d_0 is the dimension of features detailed in Subsection IV-A. The first linear layer maps the raw features into a higher embedding dimension d_1 for each node:

$$\mathbf{X}_1^e = \text{ReLU}(\text{BatchNorm}(\mathbf{X}^e\mathbf{W}_1^e + \mathbf{b}_1^e)), \tag{5}$$

where $\mathbf{W}_1^e \in \mathbb{R}^{d_0\times d_1}$ and $\mathbf{b}_1^e \in \mathbb{R}^{d_1}$ are shared parameters across nodes. The second layer then compresses the feature dimension from d_1 to d_h with $\mathbf{W}_2^e \in \mathbb{R}^{d_1\times d_h}$ and $\mathbf{b}_2^e \in \mathbb{R}^{d_h}$:

$$\mathbf{X}_2^e = \text{BatchNorm}(\mathbf{X}_1^e\mathbf{W}_2^e + \mathbf{b}_2^e). \tag{6}$$

In the first layer, the ReLU activation function ($x' = \max\{0, x\}$) introduces non-linearity, while Batch Normalization is applied in both layers to improve training stability and accelerate convergence. The omission of activation functions in the second layer is a deliberate choice to focus on dimensionality reduction and linear transformation. By doing so, the encoder converts the input features into a more compact and informative representation, optimizing the feature space for subsequent processing in the processor stage.

2) Processor: We adopt a graph-based approach for its advantages discussed in Section II. We select GCNN [34] as our processor because of its simplicity and popularity, leveraging graph topology to propagate information between connected nodes. This is beneficial in traffic networks, where conditions at one road segment (node) often influence or correlate with those at neighboring segments (nodes). For cyclists, factors like road availability, bike lanes, and traffic signals at nearby segments also affect travel conditions.

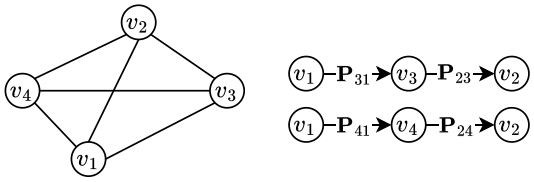


Fig. 6. An example of 2-step message propagation in a graph, where $\mathbf{P}_{ij}$ corresponds to the edge weight from node v_j to v_i. With the graph structure on the left, there are two paths possible from node v_1 and v_2 on the right.

Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with node feature matrix $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}|\times F}$, where x_{if} represents the f-th feature of node i, the GCNN operation maps the F-dimensional node features to F'. In our static road network learning stage, the feature dimension remains constant, i.e., $F = F' = d_h$. We use the normalized adjacency matrix $\mathbf{A}$ to construct the edge weight matrix $\mathbf{P} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$. Each entry of the adjacency matrix $\mathbf{A}$ is normalized by the in-degree matrix $\mathbf{D}$, where $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$. The edge weight matrix is $\mathbf{P} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ and each edge weight element $\mathbf{P}_{ji}$ contains edge propagation weights from node i to node j. This normalization ensures proper scaling of node influences based on their degrees, preventing high-degree nodes from dominating feature propagation.

Information propagation within K-hop neighbors with adaptive feature transformation matrices [49] is represented as:

$$y_{jf'} = \sigma\left(\sum_{k=0}^{K}\sum_{f=1}^{F}\sum_{i\in\mathcal{V}}\mathbf{P}_{ji}^k x_{if} w_{ff'}^k\right), \tag{7}$$

where σ is a non-linear activation function, and $y_{jf'}$ represents the f'-th feature of node j after propagation. This process aggregates information from K-hop neighbors. At each step k, node j is influenced by its k-hop neighbors i, weighted by $\mathbf{P}_{ji}^k$ and transformed by $w_{ff'}^k$. In vector form, this can be expressed as:

$$\mathbf{Y} = \sigma\left(\sum_{k=0}^{K}\mathbf{P}^k\mathbf{X}\mathbf{W}_k\right), \tag{8}$$

where $\mathbf{Y} \in \mathbb{R}^{|\mathcal{V}|\times F'}$ is the output feature matrix, and $\mathbf{W}_k \in \mathbb{R}^{F\times F'}$ is the learnable feature transformation matrix. Notably here the learnable parameters are independent of the graph structure, favoring the scalability.

In this study, we choose $K = 3$. A small K restricts information propagation to nearby nodes, whereas a large K can incorporate irrelevant distant nodes. The physical meaning of the (i, j)-th element of $\mathbf{P}^k$ is the sum of all path weights from node j to node i within k steps, where each path weight is the product of the edge weights along that path. For example, in Figure 6, when $k = 2$, there are two paths from v_1 to v_2, with the weights of each path respectively being $\mathbf{P}_{31}\times\mathbf{P}_{23}$ and $\mathbf{P}_{41}\times\mathbf{P}_{24}$. The entry $\mathbf{P}_{21}^2$ is the sum of these two path weights, representing the total influence or connectivity strength from node v_1 to node v_2 across all possible 2-step paths and is applied to all feature dimensions.

3) Decoder: Similar to the encoder, the decoder is composed of two linear layers. Skip connections are used to connect the encoder and decoder, allowing for the direct information passing across the network and helping to preserve important details that might otherwise be lost through multiple layers of processing. Denoting the processed graph feature as $\mathbf{X}^d \in \mathbb{R}^{|\mathcal{V}| \times d_h}$, the decoder is expressed as:

$$\mathbf{X}_1 = \text{ReLU}(\text{BatchNorm}((\mathbf{X}^d + \mathbf{X}_2^e)\mathbf{W}_1^d + \mathbf{b}_1^d)), \quad (9)$$

$$\mathbf{E}^G = \text{BatchNorm}((\mathbf{X}_1 + \mathbf{X}_1^e)\mathbf{W}_2^d + \mathbf{b}_2^d), \quad (10)$$

where $\mathbf{W}_1^d \in \mathbb{R}^{d_h \times d_1}$, $\mathbf{b}_1^d \in \mathbb{R}^{d_1}$, $\mathbf{W}_2^d \in \mathbb{R}^{d_1 \times d_G}$, and $\mathbf{b}_2^d \in \mathbb{R}^{d_G}$. The decoder amplifies the feature dimensions for each node, allowing for more complex transformations and capturing higher-level representations for the final road graph embedding.

C. Trip-Specific Learning Component

A trip is a series of road segments, thus requiring the extraction of sequential information for the travel time estimation problem. Our approach represents a trip as a graph. This representation naturally models travel trajectories over the road network, and we have carefully designed the node features for the trip-specific graph to intuitively and efficiently capture travel dynamics, as detailed in Subsection IV-A.

As shown in Figure 3, the *trip-specific learning component* pipeline resembles that of the *road network learning component*, with the main difference being that the input to the processor is the summation of the processed directed travel graph features from the encoder and the temporal embeddings obtained from a 2-layer perceptron (MLP). Moreover, the encoder processes the trip-specific graph, while the decoder generates the trip-specific embeddings for each road segment. Denote the input trip graph as $\mathbf{X}_0 \in \mathbb{R}^{|\mathcal{V}_i| \times d_0}$ and the temporal input as $\mathbf{X}_1 \in \mathbb{R}^{|\mathcal{V}_i| \times d_1}$ (a stack of temporal features for all road segments). The *trip-specific learning component* can be summarized as:

$$\mathbf{X}_0' = \text{Encoder}(\mathbf{X}_0), \quad (11)$$

$$\mathbf{X}_1' = \text{MLP}(\mathbf{X}_1), \quad (12)$$

$$\mathbf{X} = \text{Processor}(\mathbf{X}_0' + \mathbf{X}_1'), \quad (13)$$

$$\mathbf{E}^T = \text{Decoder}(\mathbf{X}). \quad (14)$$

Here, $\mathbf{E}^T$ represents the trip-specific embedding. The adjacency matrix in the processor is directed and has the following format:

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{pmatrix}. \quad (15)$$

When the edge weight matrix is set equal to the adjacency matrix, i.e., $\mathbf{P} = \mathbf{A}$, the processor in the *trip-specific learning component* resembles the sequential convolution [50], but it is adapted to road segments in our case.

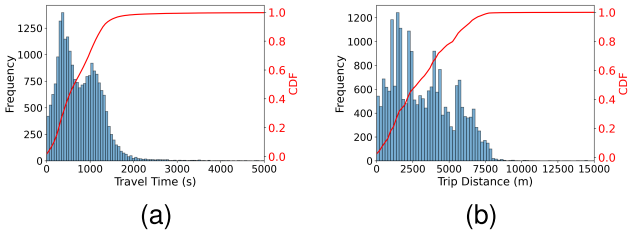


Fig. 7. Frequency and cumulative distribution function (CDF) of (a) travel time and (b) trip distance from the SimRa dataset. For better visualization, travel time is capped at 5000 seconds and distance at 15000 meters.

D. Speed Querying Component

The objective of the *speed querying component* is to utilize the universal road graph embedding along with trip-specific representations to estimate trip travel time, with road-segment-wise speed as an intermediate output.

As shown in Figure 7, the bicycle dataset is not only severely skewed but also presents challenges typical of real-world datasets. The volatile travel time distribution can bias model training and limit generalization to unseen data. To address this challenge, [32] proposed a categorical approximation solution. This method groups travel time data based on the cumulative distribution function (CDF), ensuring an equal distribution of trips across these groups and transforming the estimation task into predicting group labels. However, the group labeling used in this method is risky for bicycle travel time estimation, as it heavily relies on an unbiased data distribution and can lead to overly detailed group divisions for short trips and overly coarse groups for longer trips.

In contrast, we avoid grouping travel times and propose a speed-based approach. Our speed selection is independent of the CDF and unaffected by data distribution. Assuming a maximum cycling speed of 10 m/s [51], slightly above the 30 km/h limit in urban areas per Dutch traffic laws[2] to account for rule violations, this speed also encompasses the speed range observed in the berlin SimRa dataset [52]. We discretize speeds from 0 to 10 m/s into a speed vector (e.g., $[0.25, 0.75, \ldots, 9.75]$ for granularity $g = 0.5$ m/s). The trip travel time is estimated by determining the likelihood of each speed option at each road segment. By doing so, we transform the estimation problem from a long-tailed distribution over a large time interval to a limited speed range.

As shown in Figure 3, the speed vector is defined as $[s_1, s_2, \ldots, s_{d_s}]$ (e.g., $[0.25, 0.75, \ldots, 9.75]$). For a trip with $|\mathcal{V}^i|$ road segments, the corresponding segments are extracted from the road graph embedding $\mathbf{E}^G \in \mathbb{R}^{|\mathcal{V}| \times d_G}$ (output of the *road network learning component*), forming $\mathbf{E}_T^G \in \mathbb{R}^{|\mathcal{V}^i| \times d_G}$. The embeddings $\mathbf{E}_T^G$ and trip-specific embeddings $\mathbf{E}^T \in \mathbb{R}^{|\mathcal{V}^i| \times d_T}$ (output of the *trip-specific learning component*) respectively represent the intrinsic road network and trip-specific aspects. These two representations are combined by concatenation and mapped to a d_s-dimensional speed latent space:

$$\mathbf{E}^S = \text{MLP}(\text{Concat}([\mathbf{E}_T^G, \mathbf{E}^T])). \quad (16)$$

[2]https://www.government.nl/topics/bicycles/safe-cycling

Each row $\mathbf{e}_i^S \in \mathbb{R}^{d_s}$ is the representation of road segment i in the speed space. After applying SoftMax for each row $\mathbf{e}_i^S$, the likelihood of different traveling speeds on segment i can be derived and the segment-wise estimated travel time is given by:

$$p_k = \frac{\exp(\mathbf{e}_i^S[k])}{\sum_{j=1}^{d_s} \exp(\mathbf{e}_i^S[j])}, \tag{17}$$

$$\hat{t}_i = \frac{d_i}{\hat{s}_i} = \frac{d_i}{\sum_{k=1}^{d_s} s_k p_k}, \tag{18}$$

where p_k is the likelihood of speed s_k, $\hat{s}_i$ is the estimated speed for road segment i, and d_i is the actual traveled distance over it. The overall estimated travel time $\hat{y}$ is then calculated as the sum of the travel times for N travelled road segment:

$$\hat{y} = \sum_{i=1}^{N} \hat{t}_i. \tag{19}$$

E. Loss Function

During our end-to-end training phase, we have road segment speeds as intermediate outputs and the trip travel time as the final output, which enables us to train the model by combining both the road segment speed perspective and the travel time perspective. We use the normalized Mean Squared Error (nMSE) for travel time loss $\mathcal{L}_t$, as it normalizes errors by trip length and addresses the imbalance in error distribution across different trip durations, while Mean Absolute Error (MAE) is applied to speed loss $\mathcal{L}_s$ due to its robustness against speed outliers. The final loss $\mathcal{L}$ is then:

$$\mathcal{L} = (1-\alpha)\mathcal{L}_t + \alpha\mathcal{L}_s, \tag{20}$$

where $\alpha \in [0, 1]$ is a factor scaling the contribution of the speed loss. The speed loss guides the training from a road segment perspective, and the travel time loss is used to reduce the overall trip-wise error.

V. Experiments

In this section, we first introduce the real-world dataset used in this study: the German SimRa dataset. Next, we evaluate the proposed DG4b framework and compare its performance with various baseline models. We then conduct an ablation study to assess the contribution of different model components and analyze the computational complexity relative to other deep-learning baselines. Finally, we corroborate our model on the Talking Bike dataset to evaluate its generality.

A. Dataset

1) SimRa Dataset: The SimRa dataset is a crowdsourced German bicycle dataset collected via a mobile app [53]. The project was initiated in June 2019, and the dataset is publicly accessible through GitHub.[3] We selected Berlin as our study area, as it is the main data collection city, and used data up to the latest available (January 2025). The crowdsourced nature of data collection ensures a diverse user base, making the dataset more representative.

TABLE I
Number and Proportion of Trips Categorized by Time of Day (off-Peak Vs. Peak Hours) of SimRa Dataset

	#Trips	Proportion
Off-peak hour	13,336	59.75%
peak hour	8,984	40.25%

We only consider trips with more than three non-static GPS records, as trips with fewer records may not accurately capture travel behavior. Such trips could represent incomplete data and provide limited information about travel times or speeds, making them too aggregated for analysis.

The obtained trip statistics are presented in Table I. The dataset is split into training, validation, and test sets with a 70%/10%/20% ratio, ensuring a close travel time distribution across each subset.

Regarding the road network, we construct the line graph as described in Subsection IV-A based on OpenStreetMap data.[4] The complete road graph comprises 47,896 nodes and 150,172 edges for the SimRa dataset. Given the time span and size of the road network, our dataset has a rather sparse distribution.

B. Evaluation Metrics

We use four common metrics for evaluation: Root Mean Square Error (RMSE), Mean Average Error (MAE), Mean Absolute Percentage Error (MAPE), and Satisfaction Rate (SR). These metrics are commonly used in literature on travel time estimation [29], [32], [47]. Their mathematical formulas are as follows:

$$\text{RMSE} = \sqrt{\frac{1}{|\mathcal{T}_i|}\sum_{i=1}^{|\mathcal{T}_i|}(y^i - \hat{y}^i)^2}, \tag{21}$$

$$\text{MAE} = \frac{1}{|\mathcal{T}_i|}\sum_{i=1}^{|\mathcal{T}_i|}|y^i - \hat{y}^i|, \tag{22}$$

$$\text{MAPE} = \frac{1}{|\mathcal{T}_i|}\sum_{i=1}^{|\mathcal{T}_i|}\left|\frac{y^i - \hat{y}^i}{y^i}\right| \times 100\%, \tag{23}$$

$$\text{SR} = \frac{1}{|\mathcal{T}_i|}\sum_{i=1}^{|\mathcal{T}_i|}\left(\left|\frac{y^i - \hat{y}^i}{y^i}\right| \le 20\%\right) \times 100\%, \tag{24}$$

where $\hat{y}^i$ is the estimated travel time of trip $\mathcal{T}_i$ and $|\mathcal{T}_i|$ is the number of trips evaluated. By definition, MAPE and SR are highly sensitive to short trips, where even small absolute errors can result in disproportionately high relative errors, making them less suitable for evaluating short trips.

C. Experiment Set-up

We use a low-dimensional setting to maintain low model complexity. Specifically, we set the hidden feature dimension d_h to 16 for both the *road network learning component* and the *trip-specific learning component*. The hidden dimension in the MLP for processing temporal features is set to 16. By default, we set the granularity of the speed tensor $g = 0.5$, representing

[3] https://github.com/simra-project/

[4] https://download.bbbike.org/osm/bbbike/

a speed tensor of 20 speeds. The graph propagation step is set to $K = 3$ and the speed loss scaling factor $\alpha = 0.4$.

All experiments were conducted on an Apple M2 Pro chip. The model was implemented using PyTorch and trained with the Adam optimizer. The learning rate is set to 0.001, and the batch size is 1024.

D. Performance Comparison

In this subsection, we first evaluate the performance of our method against baseline models on the SimRa dataset, followed by an analysis of model performance during peak and off-peak hours.

1) Models: As shown in the Introduction section, no models have yet been developed for bicycle travel time estimation. Consequently, we select six baseline models based on the following criteria: (1) their application to car travel time estimation; (2) their adaptability to our bicycle dataset; and (3) their representation of a diverse range of methods, including statistical approaches, simple machine learning techniques, and deep learning methods. The selected models include three statistical approaches—Historical Average (HA) [22], [25], [29], Temporally Weighted Neighbors (TEMP) [22], [28], [31], and Linear Regression (LR) [22], with HA and LR being basic statistical methods and TEMP specifically developed for travel time estimation; two machine learning models—Light Gradient Boosting Machine (LightGBM) [40], [54] and Multi-Layer Perceptron Neural Network (mlpNN) [29]; and one deep learning model commonly used for car travel time estimation—Deep learning framework for Travel Time Estimation (DeepTTE) [26], [27], [28], [29], [32]. For models where randomness plays a role, the performance is evaluated by averaging the results across five random seeds. For baseline models, we use grid search to identify optimal hyperparameters.

(1) **HA:**Historic Average estimates the travel time for each road segment using the average speeds from peak or off-peak hours. The total trip travel time is the sum of its composing road segment travel times. For road segments with no recorded average speeds or those with only speeds below 0.01 m/s, the mean speed of all road segments is used as a substitute.

(2) **TEMP:**Temporally Weighted Neighbors estimates travel time by averaging the travel times of trips with neighboring origins and destinations. If no neighboring trips are available, we use the mean travel time of the entire training dataset.

(3) **LR:**Linear Regression uses a linear model to estimate travel time, incorporating features such as estimated travel speed (from HA), the number of road segments traversed, travel distance, origin/destination coordinates, and a peak/off-peak hour indicator.

(4) **LightGBM:**The Light Gradient Boosting Machine outperforms other models within the GBDT family with faster training speed, better accuracy, and lower memory consumption [40]. We utilize the same features as in the LR model.

(5) **mlpNN:**The Multi-Layer Perceptron Neural Network utilizes fully connected layers, batch normalizations, and activation functions to approximate the travel time. Same features as in the LR model are used, with peak/off-peak hours represented through one-hot encoding.

(6) **DeepTTE:** The Deep learning framework for Travel Time Estimation is a well-established baseline for car travel time estimation. It uses LSTM to capture spatio-temporal dependencies after transforming the raw GPS trajectory into a series of features. We use the features described in [29], including date, week, time, travel distance, geolocations, temporal gaps, and distance gaps. We exclude driver ID and taxi state, as these features are not applicable to our dataset.

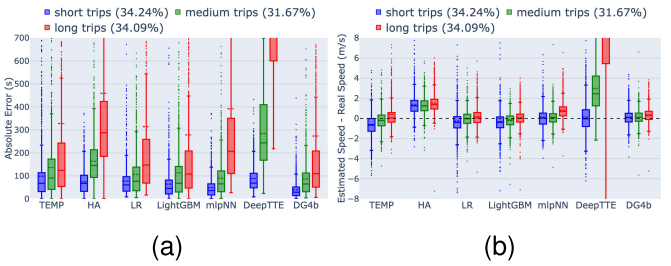


Fig. 8. Box plot of errors for different methods across each trip category on test dataset. The mean error is represented by a dotted line, and the median by a solid line. (a) Absolute error box plots for various methods using five random seeds, with absolute errors capped at 700 seconds for improved visualization. (b) Box plots of speed errors (estimated minus actual speeds), restricted to a range of -8 m/s to 8 m/s for easier visualization.

2) Performance: Each model is trained on the full training dataset, with separate evaluations for short, medium, and long trips. Table II presents the average performance of each method over five random seeds. Figure 8 displays the absolute travel time estimation error and speed error for each method. The speed error is defined as the difference between the estimated travel speed, derived from travel distance and estimated travel time, and the ground truth.

According to Table II, our method outperforms baseline models across all trip lengths. The improvement is most significant for short trips, achieving over a 13% gain in each evaluation metric. Short-trip estimation is inherently more challenging as these trips exhibit greater variability, they are more easily affected by factors such as traffic lights, whereas such influences tend to average out over longer trips.

For medium and long trips, our model still achieves the best performance, though the improvement is less pronounced compared to short trips. As shown in Figure 8, we observe reduced error variation compared to the baseline models, highlighting that our method is able to generate more accurate and stable travel time and speed estimates.

Among the baseline models, mlpNN performs best for short and medium trips, while LightGBM excels for long trips, indicating that mlpNN is better suited for capturing complex patterns in shorter trips. Conversely, simpler models such as LR and LightGBM outperform more complex machine learning frameworks like mlpNN and DeepTTE for long trips. This highlights the importance of selecting appropriate baseline models based on trip length while also demonstrating that our model consistently performs well across all categories.

TABLE II

PERFORMANCE COMPARISON BETWEEN BASELINE MODELS AND OUR METHOD. RESULTS REPRESENT AVERAGE PERFORMANCE ACROSS EXPERIMENTS WITH FIVE RANDOM SEEDS. THE BEST-PERFORMING BASELINE MODEL IS INDICATED IN BOLD, WHILE THE BEST AMONG ALL METHODS IN BOLD RED

Method	Total				Short (≤8 min)				Medium (8-16 min)				Long (>16 min)			
	RMSE	MAE	MAPE	SR	RMSE	MAE	MAPE	SR	RMSE	MAE	MAPE	SR	RMSE	MAE	MAPE	SR
HA	623.41	233.8	25.94	39.42	91.63	74.16	27.38	38.06	197.56	163.89	23.57	43.14	1046.63	459.11	26.7	37.32
TEMP	641.23	188.69	36.82	62.28	150.76	98.65	73.33	45.26	201.12	136.42	19.35	67.54	1070.44	327.71	16.37	74.51
LR	586.5	167.28	35.91	64.69	105.33	77.33	74.55	48.23	148.65	106.46	15.11	74.67	988.62	314.13	16.42	71.97
LightGBM	**553.29**	**152.78**	23.46	**70.88**	107.63	64.45	39.21	57.95	200.73	113.13	16.2	74.47	**921.41**	**278.34**	**14.39**	**80.55**
mlpNN	623.95	178.5	**18.0**	67.25	**66.87**	**48.63**	**20.06**	**64.2**	**125.09**	**89.21**	**12.71**	**80.92**	1059.75	391.91	20.85	57.61
DeepTTE	922.32	507.15	57.18	19.86	117.24	88.68	54.34	41.67	324.17	284.22	38.91	16.85	1543.78	1134.66	77.01	0.75
DG4b	**527.81**	**134.2**	**14.17**	**78.39**	**56.28**	**39.9**	**16.4**	**72.62**	**120.82**	**86.5**	**12.32**	**82.42**	**894.71**	**273.26**	**13.64**	80.43
Improvement	**4.61%**	**12.16%**	**21.28%**	**10.6%**	**15.84%**	**17.95%**	**18.25%**	**13.12%**	**3.41%**	**3.04%**	**3.07%**	**1.85%**	**2.9%**	**1.83%**	**5.21%**	**-0.15%**

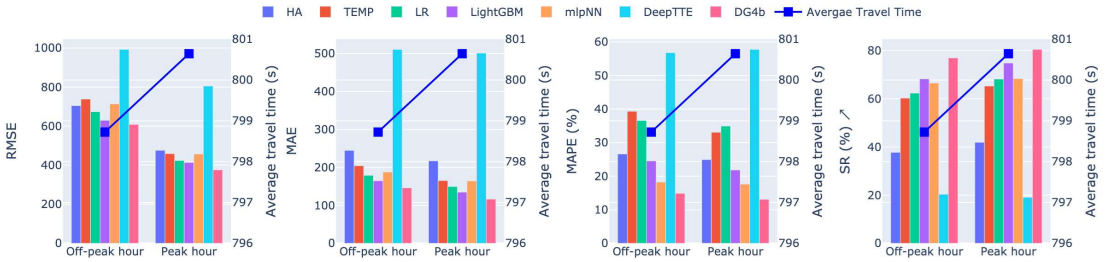


Fig. 9. Peak/Off-peak hour performance comparison (RMSE, MAE, MAPE, and SR) for different methods. The line graph indicates that trips sampled during peak hours have similar average travel times than off-peak hours.

TABLE III

ABLATION STUDY FOR DIFFERENT MODEL COMPONENTS. VARIANT 1: THE ROAD NETWORK LEARNING COMPONENT IS REMOVED; VARIANT 2: THE TRIP-SPECIFIC LEARNING COMPONENT IS REMOVED

Parameter	Short (≤8 min)				Medium (8-16 min)				Long (>16 min)			
	RMSE	MAE	MAPE	SR	RMSE	MAE	MAPE	SR	RMSE	MAE	MAPE	SR
Default	56.28	39.9	16.4	72.62	120.82	86.5	12.32	82.42	894.71	273.26	13.64	80.43
Variant-1	62.28	45.72	18.52	66.89	136.66	100.76	14.34	75.26	926.82	324.36	17.4	69.12
Variant-2	60.78	43.79	17.6	69.06	131.4	94.26	13.36	78.05	951.95	289.89	14.38	78.71

Moreover, DeepTTE performs worse than mlpNN, likely due to its direct processing of raw GPS data sequences. At higher sampling frequencies, the increased number of data points may overwhelm the model, making it difficult to extract meaningful contextual information. This emphasizes the importance of properly modeling trip representations. Additionally, our implementation involved scaling down the embedding dimensions of the DeepTTE model to accommodate computational constraints and the limited dataset size. This dimensional reduction may have affected the model's representational capacity, suggesting that alternative architectures better suited for smaller datasets might be needed, but this is not in the scope of this study.

We also note that statistical methods such as TEMP and HA, which rely on average travel times of trips with similar origins and destinations or recorded average speeds, are ineffective for all trip durations. This suggests that trip duration is influenced not only by average speed and neighboring trips, but also by the specific road network infrastructure and surrounding environment of the selected path.

3) Peak versus Off-Peak Hours: Considering that traffic volumes vary between peak and off-peak hours, we compare model performances during these times in Figure 9. We observe similar average travel times during peak and off-peak hours in the SimRa dataset, with all models generally performing better during peak hours. This could be because cyclists tend to follow more consistent routes and patterns during peak hours in Berlin, when traffic is denser. In contrast, off-peak hours can feature more sporadic traffic, influenced by individual cyclists' preferences. Overall, our method demonstrates good performance across both peak and off-peak hours, highlighting its robustness in varying conditions.

E. Ablation Study

In this subsection, we conduct ablation studies to assess the contribution of different model components to the overall performance. To isolate the roles of the intrinsic road network and trip-specific perspectives, we propose two variants: Variant-1, which removes the *road network learning component*, and Variant-2, which removes the *trip-specific learning component*. The results in Table III indicate that Variant-1 consistently performs worse than Variant-2 across nearly all metrics, suggesting that road infrastructure and environmental factors have a stronger impact on model performance than trip-specific path choices. Notably, both variants underperform relative to the complete default model across all trip categories. This performance gap highlights that integrating both road network and trip-specific perspectives significantly enhances travel time estimation accuracy.

TABLE IV
THE NUMBER OF TRAINABLE PARAMETERS AND TRAINING TIME PER EPOCH FOR DEEP-LEARNING BASED METHODS

Method	#Parameters	Training time(s)/epoch
mlpNN	847	∼1s
DeepTTE	17,669	∼410s
DG4b	18,664	∼21s

F. Model Complexity

Table IV presents the number of trainable parameters and the training time per epoch for deep-learning methods. To ensure fair comparison and feasible computation time, we proportionally scaled down DeepTTE's vector dimensions based on our dataset size. Despite having comparable parameters to our DG4b model, DeepTTE's training time is approximately 20 times longer than ours. This significant efficiency gap highlights the drawbacks of DeepTTE's overly complex architecture with LSTMs and attention mechanisms: complexity fails to translate into performance, as it achieved the worst results among all deep learning models tested.

The mlpNN model, while extremely efficient with minimal parameters, demonstrates clear limitations in generalizing to long trips in Table II. This illustrates the weakness of simplistic uninterpretable approaches for spatiotemporal problems. In contrast, our graph-based approach functions as a grey-box model, strategically incorporating domain knowledge into the architecture while maintaining an optimal balance between interpretability and computational efficiency.

Our experimental results conclusively demonstrate that parameter count alone is an insufficient metric for model evaluation. Despite having slightly more parameters than alternatives, our model achieves superior performance across all metrics while requiring minimal hyperparameter tuning and reasonable training time.

G. Generality of DG4b

Our previous case study was based on the German SimRa dataset. In this subsection, we apply our model to the Dutch Talking Bike dataset to evaluate its generality. The Talking Bike dataset was collected in the Netherlands between October 2020 and October 2022. For this dataset, we focus on Rotterdam, the second-largest Dutch city. Over 95% of the data was collected via trackers installed on bicycles. A large portion of the data originates from food delivery riders. Due to privacy constraints, we cannot identify specific delivery trips or exact timestamps, leading to long stops during deliveries that introduce noise into travel time estimation. After applying a specialized preprocessing step to distinguish delays caused by cyclist behavior (e.g., stopping for food delivery) from those caused by traffic signals, and removing trips with outlier speeds, the resulting dataset contains 175,397 trips, over 85% of which are shorter than 5 minutes (hereafter referred to as short trips).

When examining the inherent map-matching errors in the Talking Bike dataset, we observe that short trips exhibit significantly higher errors compared to longer trips. Nevertheless, given the scarcity and value of real-world bicycle trajectory data, we include them in our analysis to evaluate the generality of our method. Since the data quality for trips longer than 5 minutes is relatively good, we conducted a data sensitivity analysis to assess the impact of short-trip data during the training phase. Specifically, we randomly sampled 80%, 50%, and 30% of the original short-trip data for training, while keeping the validation and test sets unchanged.

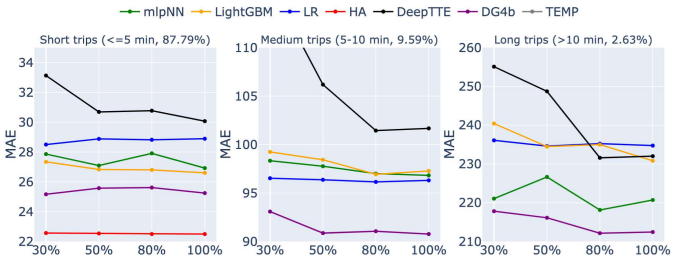


Fig. 10. Evolution of MAE for short, medium, and long trips across different methods. The x-axis represents different percentages of short trips sampled during the training phase. For better visualization, the MAE values for short, medium and long trips are capped at 35, 110 and 260, respectively.

When implementing DG4b, we adopt peak and off-peak hour definitions consistent with those of the Dutch public transport fare system [55], [56]. Given the high proportion of short trips, we use Root Mean Square Error (RMSE) as the travel time loss function instead of nMSE, as the latter disproportionately biases the model toward very short trips. The universal road line graph used in our analysis comprises 13,118 nodes and 53,304 edges.

In Figure 10, we observe a general improvement in model performance as more short trips are sampled in the training dataset. This indicates that, although the dataset is biased toward short trips, which exhibit significant variability in data quality, removing these trips would be unwise, as it could exacerbate the issue of data sparsity. Short trips provide information on factors like traffic congestion during peak hours and specific road segments. This information can be valuable for improving the estimation of longer trips. Notably, the improvement in DeepTTE is the most significant as the training size increases, likely due to its larger model size requiring more data.

Our results indicate that our model performs well in handling variations in data size and achieves better results for trips longer than 5 minutes. We would also like to highlight that both the Talking Bike and SimRa datasets exhibit similar challenges related to the sparsity of real-world bicycle trajectory data. Despite differences in geographic regions and data collection methods, our model consistently performs well across both datasets. This indicates that DG4b generalizes effectively to real-world bicycle mobility data collected under diverse conditions.

Interestingly, during the grid search for baseline models, the number of model parameters varies significantly across datasets. The mlpNN model ranged from 847 parameters for the SimRa dataset to 69,647 for the Talking Bike dataset, while DeepTTE ranged from 17,669 to 314,091 parameters. Even when applying the same hyperparameter settings optimized for the SimRa dataset to the Talking Bike dataset, our model still achieved the best performance for medium and long trips. This

finding further demonstrates the transferability of our model across different datasets with minimal adjustments.

VI. Discussion

The bicycle travel time estimation framework presented in this study offers municipalities actionable insights for sustainable urban transportation planning. By enabling more accurate estimation of cycling accessibility and efficiency, this approach supports evidence-based decisions for infrastructure investment and policy interventions that can promote cycling as a viable alternative to motorized transport.

To achieve this practical utility, we propose a novel dual graph-based deep learning solution for bicycle travel time estimation. Our method uses the term "dual graph" to refer to two distinct physical underlying structures: the universal road network graph and the trip-specific graph, which together capture both static infrastructure characteristics and dynamic cyclist behaviors while remaining computationally efficient even with limited trajectory data.

When applied to the real-world SimRa dataset, our method demonstrates significant improvements over baseline models despite dataset scarcity constraints. To further showcase the generality of our approach across diverse settings, we also evaluate it on the Talking Bike dataset. The results highlight that our method generalizes effectively across different cities and countries, without relying on specific data sampling rates. Moreover, our method consistently extracts useful trip information even from datasets with varying GPS quality.

Whereas existing dual graph designs often focus on node-wise and edge-wise representations that suit vehicular networks [43], [44], [57], [58], our approach accounts for the complexity of bicycle traffic, where intersections may involve diverse lane types—such as pedestrian paths, dedicated cycle lanes, and shared car lanes. By recognizing and integrating these distinct patterns, our model offers a more context-aware representation of bicycle movement through urban environments.

We acknowledge that bicycle travel times can vary significantly between individuals due to factors such as age, gender, fitness level, cycling experience, and bicycle type [59], [60], [61]. However, due to strict privacy regulations, accessing personal cyclist data is highly challenging. An alternative approach would be to frame the bicycle travel time estimation problem as a time-series prediction task, where travel times from preceding timestamps are used to predict the next timestamp. This would shift the application focus from trip planning and route optimization to real-time traffic monitoring and short-term forecasting. However, this approach would require real-time information about bicycle travel times, which presents a significant challenge. Given the current difficulties in obtaining representative historical datasets, acquiring the necessary real-time data is currently infeasible. While this direction is beyond the scope of this study, it presents a promising avenue for future research.

Although our model can handle sparse datasets, its performance could still improve with a larger, more comprehensive dataset. Such datasets may become available in the future as data collection techniques evolve and data protection measures become more robust. When a larger dataset is available, finer temporal modeling can be applied. This would involve considering not only the peak vs. off-peak index, but also features such as the day of the week, month, and season. These additional temporal features would allow the model to capture more complex periodic patterns in cycling behavior that vary across different time scales. When possible, identifying stops caused by intersection red lights and separating trips that experience intersection stops from those that do not can improve model performance.

VII. Conclusion

In an era where sustainable transportation and data-driven methods are increasingly valued, and bicycles play a crucial role in green and healthy mobility, our study addresses a critical gap, namely urban travel time estimation for cyclists. We introduce a pioneering dual graph-based method that not only achieves overall superior performance on a real-world dataset, but also gives a first demonstration how bicycle-specific trip characteristics can be effectively integrated with graph-based deep learning. Our approach involves two separate graph pipelines: a static road graph applicable to all trips and a trip-specific graph carefully designed to reflect travel progress. By processing these graphs in parallel, our model can easily handle multiple trips simultaneously. We further innovate by mapping the graph embeddings to a speed vector and estimating road segment speeds using a likelihood matrix, which not only provides meaningful intermediate outputs but also aids in model training. This method marks a significant advancement in bicycle travel time estimation, combining practical performance with novel graph-based techniques.

However, we acknowledge that several questions remain, opening avenues for future research. Our method currently does not address the spatial bias in the real-world dataset. Employing data augmentation techniques could help create a more uniformly distributed dataset while preserving individual cycling travel patterns. Additionally, integrating external features such as temperature and precipitation could potentially enhance the model, though their relevance and the added complexity warrant careful consideration. Exploring model explainability through various explanation tools could also improve our understanding of the model's decision-making process and enhance its transparency. Despite these unresolved questions, this research provides a foundation for future data-driven studies in bicycle-related traffic state estimation and aims to contribute to the advancement of this field.

References

[1] R. Hickman and D. Banister, *Transport, Climate Change and the City*. Evanston, IL, USA: Routledge, 2014.

[2] J. Pucher and R. Buehler, "Making cycling irresistible: Lessons from The Netherlands, Denmark and Germany," *Transp. Rev.*, vol. 28, no. 4, pp. 495–528, Jul. 2008.

[3] G. de Moraes Ramos, T. Mai, W. Daamen, E. Frejinger, and S. P. Hoogendoorn, "Route choice behaviour and travel information in a congested network: Static and dynamic recursive models," *Transp. Res. C, Emerg. Technol.*, vol. 114, pp. 681–693, May 2020.

[4] R. Cervero, S. Denman, and Y. Jin, "Network design, built and natural environments, and bicycle commuting: Evidence from British cities and towns," *Transp. Policy*, vol. 74, pp. 153–164, Feb. 2019.

[5] M. Jenks and C. Jones, Eds., *Dimensions of the Sustainable City* (Future City), vol. 2. Dordrecht, The Netherlands: Springer, 2010.

[6] Y.-H. Seo, D.-K. Kim, S. Kang, Y.-J. Byon, and S.-Y. Kho, "Rebalancing docked bicycle sharing system with approximate dynamic programming and reinforcement learning," *J. Adv. Transp.*, vol. 2022, May 2022, Art. no. 2780711.

[7] Q. He, K. L. Head, and J. Ding, "Multi-modal traffic signal control with priority, signal actuation and coordination," *Transp. Res. C, Emerg. Technol.*, vol. 46, pp. 65–82, Sep. 2014.

[8] H. Mahfouz, R. Lovelace, and E. Arcaute, "A road segment prioritization approach for cycling infrastructure," *J. Transp. Geography*, vol. 113, Dec. 2023, Art. no. 103715.

[9] U. Mori, A. Mendiburu, M. Álvarez, and J. A. Lozano, "A review of travel time estimation and forecasting for advanced traveller information systems," *Transportmetrica A, Transp. Sci.*, vol. 11, no. 2, pp. 119–157, Feb. 2015.

[10] P. H. L. Bovy and R. Thijs, Eds., *Estimators of Travel Time for Road Networks: New Developments, Evaluation Results, and Applications*. Delft, The Netherlands: Delft Univ. Press, 2000.

[11] Mobiliteitsplatform.*Siemens En Ringring Verzamelen Fietsdata Voor Talking Bikes*. Accessed: Sep. 4, 2024. [Online]. Available: https://www.mobiliteitsplatform.nl/artikel/siemens-en-ringring-verzamelen-fietsdata-voor-talking-bikes

[12] Digital-Future Berlin. (2024). *SimRa—Safety in Bicycle Traffic*. Accessed: Mar. 18, 2025. [Online]. Available: https://www.digital-future.berlin/en/research/projects/simra/

[13] K. Lee and I. N. Sener, "Emerging data for pedestrian and bicycle monitoring: Sources and applications," *Transp. Res. Interdiscipl. Perspect.*, vol. 4, Mar. 2020, Art. no. 100095.

[14] S. O'Hern, A. N. Stephens, K. L. Young, and S. Koppel, "Personality traits as predictors of cyclist behaviour," *Accident Anal. Prevention*, vol. 145, Sep. 2020, Art. no. 105704.

[15] A. K. Huemer, "Motivating and deterring factors for two common traffic-rule violations of cyclists in Germany," *Transp. Res. F, Traffic Psychol. Behav.*, vol. 54, pp. 223–235, Apr. 2018.

[16] T. Gao, W. Daamen, P. Krishnakumari, and S. Hoogendoorn, "Map-matching for cycling travel data in urban area," *IET Intell. Transp. Syst.*, vol. 18, no. 11, pp. 2178–2203, Nov. 2024.

[17] Z. Zheng, Y. Ye, Y. Zhu, S. Zhang, and J. J. Q. Yu, "Data-driven methods for travel time estimation: A survey," in *Proc. IEEE 26th Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2023, pp. 1292–1299.

[18] D. Bertsimas, A. Delarue, P. Jaillet, and S. Martin, "Travel time estimation in the age of big data," *Oper. Res.*, vol. 67, no. 2, pp. 498–515, Mar. 2019.

[19] J. J. Q. Yu, "Citywide estimation of travel time distributions with Bayesian deep graph learning," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 3, pp. 2366–2378, Mar. 2023.

[20] C. B. Gardner, S. D. Nielsen, M. Eltved, T. K. Rasmussen, O. A. Nielsen, and B. F. Nielsen, "Calculating conditional passenger travel time distributions in mixed schedule- and frequency-based public transport networks using Markov chains," *Transp. Res. B, Methodol.*, vol. 152, pp. 1–17, Oct. 2021.

[21] A. Prokhorchuk, J. Dauwels, and P. Jaillet, "Estimating travel time distributions by Bayesian network inference," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 5, pp. 1867–1876, May 2020.

[22] H. Wang, X. Tang, Y.-H. Kuo, D. Kifer, and Z. Li, "A simple baseline for travel time estimation using large-scale trip data," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–22, Mar. 2019.

[23] M. Rahmani, E. Jenelius, and H. N. Koutsopoulos, "Route travel time estimation using low-frequency floating car data," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2013, pp. 2292–2297.

[24] W. Luo, H. Tan, L. Chen, and L. M. Ni, "Finding time period-based most frequent path in big trajectory data," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Jun. 2013, pp. 713–724.

[25] R. Li, G. Rose, and M. Sarvi, "Evaluation of speed-based travel time estimation models," *J. Transp. Eng.*, vol. 132, no. 7, pp. 540–547, Jul. 2006.

[26] T. Liao, L. Han, Y. Xu, T. Zhu, L. Sun, and B. Du, "Multi-faceted route representation learning for travel time estimation," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 9, pp. 11782–11793, Sep. 2024.

[27] C. Wang, F. Zhao, H. Zhang, H. Luo, Y. Qin, and Y. Fang, "Fine-grained trajectory-based travel time estimation for multi-city scenarios based on deep meta-learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 15716–15728, Sep. 2022.

[28] L. Han, B. Du, J. Lin, L. Sun, X. Li, and Y. Peng, "Multi-semantic path representation learning for travel time estimation," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 13108–13117, Aug. 2022.

[29] D. Wang, J. Zhang, W. Cao, J. Li, and Y. Zheng, "When will you arrive? Estimating travel time based on deep neural networks," in *Proc. AAAI Conf. Artif. Intell.*, Apr. 2018, vol. 32, no. 1, pp. 2500–2507.

[30] Y. Zhu, Y. Ye, Y. Liu, and J. J. Q. Yu, "Cross-area travel time uncertainty estimation from trajectory data: A federated learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 24966–24978, Dec. 2022.

[31] Z. Wang, K. Fu, and J. Ye, "Learning to estimate the travel time," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 858–866.

[32] Y. Ye, Y. Zhu, C. Markos, and J. J. Q. Yu, "CatETA: A categorical approximate approach for estimating time of arrival," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 24389–24400, Dec. 2022.

[33] Y. Li, K. Fu, Z. Wang, C. Shahabi, J. Ye, and Y. Liu, "Multi-task representation learning for travel time estimation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, London, U.K., Jul. 2018, pp. 1695–1704.

[34] F. Gama, E. Isufi, G. Leus, and A. Ribeiro, "Graphs, convolutions, and neural networks: From graph filters to graph neural networks," *IEEE Signal Process. Mag.*, vol. 37, no. 6, pp. 128–138, Nov. 2020.

[35] W. Jiang and J. Luo, "Graph neural network for traffic forecasting: A survey," *Expert Syst. Appl.*, vol. 207, Nov. 2022, Art. no. 117921.

[36] H. Zhou, Y. Zhao, J. Fang, X. Chen, and K. Zeng, "Hybrid route recommendation with taxi and shared bicycles," *Distrib. Parallel Databases*, vol. 38, no. 3, pp. 563–583, Sep. 2020.

[37] J. Ye, J. Zhao, K. Ye, and C. Xu, "How to build a graph-based deep learning architecture in traffic domain: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 5, pp. 3904–3924, May 2022.

[38] Q. Wang, C. Xu, W. Zhang, and J. Li, "GraphTTE: Travel time estimation based on attention-spatiotemporal graphs," *IEEE Signal Process. Lett.*, vol. 28, pp. 239–243, 2021.

[39] X. Fang, J. Huang, F. Wang, L. Zeng, H. Liang, and H. Wang, "ConSTGAT: Contextual spatial–temporal graph attention network for travel time estimation at Baidu maps," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 2697–2705.

[40] G. Jin, M. Wang, J. Zhang, H. Sha, and J. Huang, "STGNN-TTE: Travel time estimation via spatial–temporal graph neural network," *Future Gener. Comput. Syst.*, vol. 126, pp. 70–81, Jan. 2022.

[41] A. Derrow-Pinion et al., "ETA prediction with graph neural networks in Google maps," 2021, *arXiv:2108.11482*.

[42] M. Xu, J. Fang, and Y. Tong, "An intelligent adaptive spatiotemporal graph approach for GPS-data-based travel-time estimation," *IEEE Intell. Transp. Syst. Mag.*, vol. 14, no. 5, pp. 222–237, Sep. 2022.

[43] H. Wang et al., "Multi-task weakly supervised learning for origin-destination travel time estimation," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 11, pp. 11628–11641, Nov. 2023.

[44] G. Jin, H. Yan, F. Li, J. Huang, and Y. Li, "Spatio-temporal dual graph neural networks for travel time estimation," *ACM Trans. Spatial Algorithms Syst.*, vol. 10, no. 3, pp. 1–22, Sep. 2024.

[45] H. Yu, X. Guo, X. Luo, W. Bian, and T. Zhang, "Construct trip graphs by using taxi trajectory data," *Data Sci. Eng.*, vol. 8, no. 1, pp. 1–22, Mar. 2023.

[46] A. Tafreshian and N. Masoud, "Trip-based graph partitioning in dynamic ridesharing," *Transp. Res. C, Emerg. Technol.*, vol. 114, pp. 532–553, May 2020.

[47] Y. Shen, C. Jin, J. Hua, and D. Huang, "TTPNet: A neural network for travel time prediction based on tensor decomposition and graph embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 9, pp. 4514–4526, Sep. 2022.

[48] K. Lemke, "Estimation of the peak-hour demand in the German highway capacity manual," *Proc. Social Behav. Sci.*, vol. 16, pp. 762–770, Jun. 2011.

[49] J. Du, S. Zhang, G. Wu, J. M. F. Moura, and S. Kar, "Topology adaptive graph convolutional networks," 2017, *arXiv:1710.10370*.

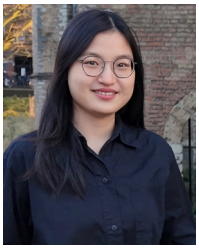
[50] M. Sabbaqi and E. Isufi, "Graph-time convolutional neural networks: Architecture and theoretical analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 12, pp. 14625–14638, Dec. 2023.

[51] Z. Huang, X. Guo, C. Zhang, and H. Zhang, "Modeling the effects of bus stops on bicycle traffic flow by cellular automata," *J. Adv. Transp.*, vol. 2018, Sep. 2018, Art. no. 5876104.

[52] A.-S. Karakaya, I.-A. Stef, K. Köhler, J. Heinovski, F. Dressler, and D. Bermbach, "Achieving realistic cyclist behavior in SUMO using the SimRa dataset," *Comput. Commun.*, vol. 205, pp. 97–107, May 2023.

[53] A.-S. Karakaya, J. Hasenburg, and D. Bermbach, "SimRa: Using crowdsourcing to identify near miss hotspots in bicycle traffic," *Pervas. Mobile Comput.*, vol. 67, Sep. 2020, Art. no. 101197.

[54] G. Ke et al., "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2017, pp. 3146–3154.
[55] NS.(2024). *Wanneer Reist U Met Korting?*. Accessed: Sep. 16, 2024. [Online]. Available: https://www.ns.nl/uitgelicht/wanneer-reizen-met-voordeel/wanneer-reist-u-met-korting.html
[56] *Highway Capacity Manual*, H. C. Manual, Washington, DC, USA, 2000, p. 1.
[57] G. Jin, H. Yan, F. Li, Y. Li, and J. Huang, "Dual graph convolution architecture search for travel time estimation," *ACM Trans. Intell. Syst. Technol.*, vol. 14, no. 4, pp. 1–23, Aug. 2023.
[58] H. Yan, G. Jin, D. Wang, Y. Liu, and Y. Li, "Jointly modeling intersections and road segments for travel time estimation via dual graph convolutional networks," in *Proc. Int. Conf. Spatial Data Intell.* Cham, Switzerland: Springer, 2022, pp. 19–34.
[59] H. Yan, K. Maat, and B. van Wee, "Cycling speed variation: A multilevel model of characteristics of cyclists, trips and route tracking points," *Transportation*, vol. 52, no. 5, pp. 1–30, Oct. 2025.
[60] N. Wheeler, R. Conrad, and M. A. Figliozzi, "A statistical analysis of bicycle rider performance: The impact of gender on riders' performance at signalized intersections," in *Proc. 89th Annu. Meeting Transp. Res. Board January*, Jul. 2010, vol. 10, no. 14, pp. 1–21.
[61] K. Schleinitz, T. Petzoldt, L. Franke-Bartholdt, J. Krems, and T. Gehlert, "The German naturalistic cycling study-comparing cycling speed of riders of different e-bikes and conventional bicycles," *Saf. Sci.*, vol. 92, pp. 290–297, Feb. 2017.



Ting Gao received the bachelor's degree in information and computing science and the master's degree in electronic information from Beihang University, Beijing, China, in 2020 and 2022, respectively, and the Diplôme d'Ingénieur degree from CentraleSupélec, Paris-Saclay University, France, in 2022. She is currently pursuing the Ph.D. degree with Delft University of Technology, funded by European Horizon Project EMERALDS. Her research interests include developing AI solutions for urban traffic state estimation and prediction. She also has a general interest in machine learning, including relational data, generative models, and reinforcement learning. She is a Co-Organizer of the Graph&Data@TU Delft seminar.

Winnie Daamen received the Ph.D. degree in transport and planning from Delft University of Technology (TU Delft), Delft, The Netherlands, in 2004. She is currently an Associate Professor with the Department of Transport and Planning, TU Delft. Her research interests include data collection (offline and real-time, for all modes) and analyses, theory and model formulation, model implementation, and management for pedestrians and cyclists. She is a member of the Pedestrian and Evacuation Dynamics (PED) and the Traffic and Granular Flow (TGF) Steering Committee. She is also an Associate Editor of IEEE OPEN JOURNAL OF THE INTELLIGENT TRANSPORTATION SYSTEMS SOCIETY and a member of the Editorial Advisory Board of *Transportation Research Part C: Emerging Technologies* and *Physica A: Statistical Mechanics and Its Applications*.

Elvin Isufi (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in electronic and telecommunication engineering from the University of Perugia, Italy, in 2012 and 2014, respectively, and the Ph.D. degree in electrical engineering from Delft University of Technology (TU Delft), The Netherlands, in 2019. In 2019, he was a Post-Doctoral Researcher with the Department of Electrical and Systems Engineering, University of Pennsylvania. He is currently an Associate Professor with the Faculty of Electrical Engineering, Mathematics and Computer Science, TU Delft, where he also co-directs the AIdroLab–the TU Delft AI Laboratory on graph-based learning for water networks. His research interests include the structure and dynamics in relational data, such as graphs and topologies with applications in critical infrastructure networks. His main research domain is with the IEEE Signal Processing Society (SPS), where he also received the 2022 IEEE SPS Best Ph.D. Dissertation Award, the IEEE ICASSP 2025 Best Conference Paper Award, and paper recognitions at IEEE CAMSAP in 2017, DSLW in 2021 and 2022, ICASSP in 2023, and Asilomar in 2025. He was a NWO VENI Fellow in 2024 and a Horizon MSCA Co-Fund Fellow in 2019. He is a member of the IEEE SPS Technical Committee on Signal Processing for Communication and Networking and the IEEE SPS Youth Professional Committee. He serves as an Associate Editor for IEEE TRANSACTIONS ON SIGNAL PROCESSING, position that he also had held for *Signal Processing* (Elsevier) (2024–2026).

Serge P. Hoogendoorn was appointed as an Antonie van Leeuwenhoek Professor of traffic operations and management in 2006. He has been the Chair of the Department of Transport and Planning since 2018. He is currently (one of the four) a Distinguished Professor of smart urban mobility with Delft University of Technology. He has a part-time appointment with Monash University. He is an Honorary Professor with the School of Transportation, Southeast University, China. He is the PI Mobility with the Institute of Advanced Metropolitan Solutions (www.ams-amsterdam.com). His current research interests include smart urban mobility, with focal areas, such as theory, modeling, and simulation of traffic and transportation networks, including cars, pedestrians, cyclists, and novel public transport services; development of methods for integrated management of these networks; impact of uncertainty of travel behavior and network operations; and impact of ICT on network flow operations, robustness, and resilience, and urban data and its applications. In all these topics, his work considers both recurrent and non-recurrent situations.