CatETA: A Categorical Approximate Approach for Estimating Time of Arrival

Yongchao Ye, *Student Member, IEEE*, Yuanshao Zhu, *Member, IEEE*, Christos Markos, and James J. Q. Yu, *Senior Member, IEEE*

***Abstract*— Estimated time of arrival (ETA) is one of the critical services offered by navigation and hailing providers. The majority of existing solutions approach ETA as a regression problem and leverage GPS trajectories for estimation. However, the travel time fluctuates greatly between different trips, making simple regression methods skewed. Additionally, these methods are incapable of conducting estimation in practice because the trajectories of future trips are unknown. To jointly tackle these problems, we propose a novel Categorical approximate method to Estimate Time of Arrival (CatETA). Specifically, we formulate the ETA problem as a classification problem and label it with the average time of each category. To eliminate bias in categorical labeling, we approximate travel time using the weighted average of different classes in the testing stage. Then, we design a network structure that extracts the spatio-temporal features of link sequences and integrates a set of global information. Furthermore, we merge link sequences according to network topology and graph embedding to alleviate the computational burden associated with large-scale link networks. Comprehensive experiments on real-world datasets demonstrate that CatETA considerably improves the estimation performance and significantly reduces computational effort.**

***Index Terms*— Estimated time of arrival, traffic prediction, spatio-temporal data mining.**

I. Introduction

ESTIMATED time of arrival (ETA) or travel times estimation that estimates the travel time between a pair of origin and destination, is a critical component of modern intelligent transportation systems (ITS). It empowers online ride-hailing and navigation platforms to improve the quality of their services [1], [2], [3].

For instance, accurate ETA can assist businesses like Didi Chuxing, Uber, and Google Maps in determining optimal routes in a manner that not only satisfies operational objectives but also promotes customer satisfaction (see Fig. 1 for an illustration). Therefore, ETA is essential for these online transportation service providers to be successful [4], [5], [6].

Manuscript received 15 December 2021; revised 9 July 2022; accepted 13 September 2022. This work was supported in part by the Stable Support Plan Program of Shenzhen Natural Science Fund under Grant 20200925155105002 and in part by the Guangdong Provincial Key Laboratory under Grant 2020B121201001. The Associate Editor for this article was N. Bekiaris-Liberis. *(Yongchao Ye and Yuanshao Zhu contributed equally to this work.) (Corresponding author: James J. Q. Yu.)*

The authors are with the Guangdong Provincial Key Laboratory of Brain-Inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China (e-mail: 12032868@mail.sustech.edu.cn; zhuys2019@mail.sustech.edu.cn; 11760007@mail.sustech.edu.cn; yujq3@sustech.edu.cn).

Digital Object Identifier 10.1109/TITS.2022.3207894

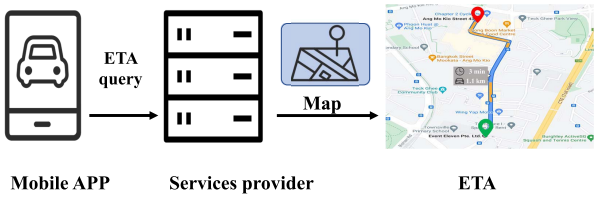


Fig. 1. Example of obtaining the estimated time of arrival (ETA) for a given origin-destination pair. The mobile app initiates an ETA query to the service provider, who then leverages the city map to determine the best route, before finally returning the trip details and ETA.

The problem of ETA has been extensively studied in recent years. A straightforward approach is to calculate the average travel time from origin to destination using historical data [7], [8], [9], [10], [11], [12]. However, these methods fall short when confronted with unknown or sparse routes (e.g., some routes do not belong to or are only rarely recorded in the historical database). Meanwhile, historical-based methods ignore external factors such as departure time or weather conditions, resulting in low ETA accuracy. In addition, history-based methods often fail to meet route planning targets, such as in cases where multiple routes connect the same origin-destination pair [5], thereby necessitating the use of advanced ETA solutions.

Given that global positioning system (GPS) trajectories contain rich temporal and spatial information, a body of research formulates the ETA problem as a regression problem subsequently modelled via deep learning algorithms [1], [2], [13], [14], [15], [16]. Some solutions divide the road network into links, i.e., road segments, which are defined as routes that connect two junctions and have no junctions in the middle. In this way, trips can be represented as a sequence of traversed links without requiring GPS-specific information [5], [6], [17], [18], [19]. By incorporating external features that have a significantly impact on the ETA problem (e.g., route state, distance, and departure time) they achieve highly competitive results [13], [20]. Taking Fig. 1 as an example, when users send an ETA query, the service provider only knows the start and end points of the user's trip. And the service provider therefore needs to determine the best path based on this query and return a sequence of links for this trip. Nonetheless, existing work has not yet addressed the following limitations:

- GPS trajectories exhibit diverse spatio-temporal correlations that are challenging to capture. In addition, GPS

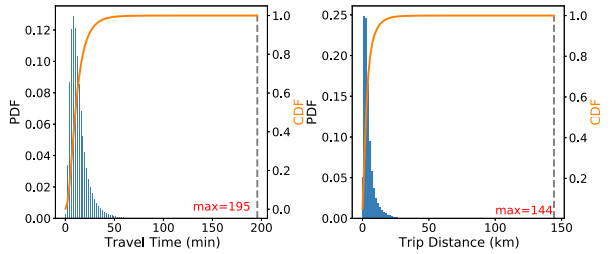


Fig. 2. Probability distribution function (PDF) and cumulative distribution function (CDF) of travel time and trip distance during a weekday. Most travel times are within 40 min, while the longest is 195 min; most trip distance are within 20 km, while the longest is 144 km. Each trip traverses an average of 87 links and a maximum of 757 links.

data are inextricably linked to privacy concerns, limiting large-scale data collection in real-world applications [21], [22]. Arguably most significantly, predicting future GPS coordinates for a given trip is unrealistic, rendering end-to-end inference impractical.

- Road traffic is dynamic, with complex spatial and temporal correlations among road network links [23]. Extracting and computing features for each link requires substantial computational resources that may not be available in time-critical use cases [18].
- Because the travel time distribution of trips is highly volatile, pure regression models may produce large errors in small-scale data. As depicted in Fig. 2, the majority of travel times during a weekday are within 40 min and most trip distance are within 20 km. This skewed distribution may cause a significant difference in model prediction among different travel time lengths.

To bridge these gaps, we propose a categorical approximate approach for estimating time of arrival termed CatETA. The primary contributions of this work are summarized as follows:

- We propose a travel time estimation method that takes as input a sequence of individual links. Unlike prior methods relying on GPS trajectories, the proposed method better suits mobile computing as it does not need to predict the GPS trajectories of travel trips on the client side.
- We build a deep neural network consisting of three main components designed to extract spatio-temporal features in addition to a set of global attributes. We further improve computational efficiency by merging the input link sequence based on spatial proximity.
- We propose an approximate categorical approach that groups travel time into classes and then estimates it using a weighted approximation of the various classes. Compared to regression methods, this approach can effectively reduce errors caused by skewed travel time distributions.
- We conduct comprehensive case studies on a large real-world dataset. The result shows that CatETA significantly outperforms other baselines in multiple scenarios. Finally, the hyperparameter settings are investigated and analyzed to provide insights into their selection.

The rest of this paper is organized as follows. Sec. II first reviews the literature on ETA, grouped into trajectory- and link-based methods. Then, Sec. III provides preliminary definitions and formalizes the problem to be addressed in this work. Sec. IV elaborates on the proposed CatETA framework and its constituting *spatio-temporal learning*, *attributes embedding*, and *prediction* components. Next, Sec. V conducts a series of experiments to assess the performance of CatETA against competitive baselines, in addition to a hyperparameter sensitivity test. Finally, Sec. VI concludes this paper.

II. Related Work

A large number of studies have been conducted on estimating ETA. Besides the naïve history-based approaches (e.g., [7], [8], [24], [25]), advanced ETA solutions can generally be classified into the following two broad categories from a data-driven perspective: GPS trajectories-based and link-based.

A. GPS Trajectories-Based ETA

With GPS sensors embedded in a multitude of modern smart devices, acquiring GPS trajectories for ETA studies is now easier than ever [4]. Wang *et al.* proposed an error feedback recurrent convolutional neural network (eRCNN) to estimate travel time and speed on sequential GPS points [26]. Although the proposed method considered the spatial-temporal correlation of trajectories and derived the estimated time based on the speed of each constituting segment, simply accumulating the travel time of each trajectory segment ultimately did not produce remarkably accurate results. Therefore, Dong *et al.* proposed schemes for directly estimating the arrival time of the entire trip in a later research [1]. Specifically, they used complex neural network models to capture fine-grained trajectories features and combine them with other attributes such as driver ID and departure time. Compared to the previous trajectory segment prediction and accumulation approach, this framework achieves higher accuracy.

To further improve the accuracy of ETA, Zhang *et al.* mapped GPS trajectories to a grid and combined the spatio-temporal information embedding with auxiliary features like driving and traffic statuses (both short- and long-term) in the grid to design an end-to-end ETA model [13]. Shen *et al.* which adopts tensor decomposition and graph embedding [14]. This scheme employed a non-negative tensor decomposition method to recover the travel speed distribution and then integrated a neural network model for representation learning. In addition, it used graph embedding to generate representations of road network structures, achieving a high level of estimation accuracy. Considering the impact of different transportation modes on ETA, Xu *et al.* proposed a multi-task learning model for ETA. This method is able to capture the interaction between transportation modes and travel time, which can recommend the appropriate transportation mode for the user and then estimate the associated ETA [27].

While GPS trajectories-based approaches have enabled significant improvements in ETA, these solutions face serious challenges: 1) These schemes utilize data consisting of real GPS points, but it is very difficult to predict future GPS points in real scenarios; 2) These schemes rely heavily on the sampling frequency and accuracy of the GPS points, which

is far from perfect for low-frequency sampling and long-term prediction.

B. Link-Based ETA

To address the aforementioned challenges, a few link-based methods were proposed lately. These methods view trips as a series of traversed links and consider their interactions, which resolves the dependence on GPS points and enables trip planning for ETA prediction. Wang *et al.* combined the advantages of the wide linear model, the deep neural network, and the recurrent neural network to propose a wide depth recurrent (WDR) learning model for estimating the arrival time [6]. This scheme agglomerates multiple trip features, such as traffic, personalized, and global information to accurately predict the travel time for a given planned trip. Based on this concept, researchers introduced an auxiliary task to learn personalized driving information (driving preferences, driving speed, etc.) in a multi-task learning framework [28]. In addition, Sun *et al.* proposed a road network metric learning framework for learning representations of elements (such as links) in road networks. Besides performing ETA prediction, they also designed auxiliary learning tasks to improve the quality of embedding vectors and proposed a triangle loss function to improve the efficiency of metric learning [29]. Considering that previous work fail to capture the movement behavior of the embedding in the trip well, Fu and Lee proposed Deep Image-based Spatio-Temporal network (DeepIST) [19]. This framework represents the trip as a sequence of "generalized images" containing sub-paths and auxiliary information, and designed a novel two-dimensional convolutional (pathCNN) for spatio-temporal modeling along the trip.

Since graph neural networks (GNN) have achieved remarkable success in modeling unstructured data, researchers have also attempted to model the spatial and temporal dependence of road traffic through spatio-temporal graph convolution [5], [30], [31]. Jin *et al.* proposed a spatio-temporal graph neural network framework for the ETA problem. Specifically, their approach adopts a multi-scale deep spatio-temporal graph convolutional network to capture structured spatio-temporal traffic conditions along with a transformer layer to extract long-term time dependencies. Finally, the two components are integrated into a real-time traffic condition representation to estimate arrival times by a gated fusion module [32]. Hong *et al.* utilized heterogeneous information graphs to transform the roadmap into a multi-relational network in the ETA task, and introduced a network based on vehicle trajectories to jointly consider traffic behavior patterns to achieve accurate ETA predictions [5]. To further exploit the spatial and temporal correlation in the traffic road network, i.e., the contextual information and connectivity of links. Fang *et al.* employed a graph attention mechanism to capture the relationship of spatio-temporal information. Then, convolution layers are applied to capture the contextual information in a local window [33]. Based on this network framework, they further proposed a self-supervised meta-learning scheme for en-route ETA, which can fast adapt to the user's driving preferences and improve the time estimation for the remaining links. Specifically, this scheme considers the observed behaviors in the traveled links as training examples and the future behaviors in the remaining links as test examples, and then utilizes self-supervised learning techniques (equivalent to generating a large number of synthetic learning tasks) to further improve performance [34]. Moreover, Yu proposed a Bayesian and geometric deep learning based approach to estimate the travel time distribution of road links in a citywide traffic network. The evaluation shows that the probabilistic distribution describe the ETA better than the deterministic model due to the uncertainty caused by the unstable traffic [35].

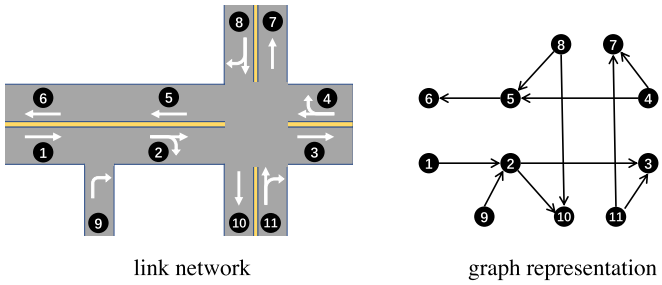


Fig. 3. Illustration of graph representation of link network. Each link in the road network is a node in the graph.

However, link-based approaches require feature extraction using complex neural networks for each link, which inevitably leads to a heavy computational burden [18]. On the other hand, general regression models cannot accurately predict arrival times due to the large variation in data distribution across travel times and the number of links. Inspired by the previous work, we propose a novel categorical approximate method to estimate the time of arrival without relying on GPS predictions, and use graph embedding to obtain the correlation of road link, which can significantly reduce the computation burden.

III. Problem Definition

Definition 1 (Link): A link represents a road segment, which is defined as a tuple $v = (v_{\text{id}}, v_{\text{ratio}}, v_{\text{status}})$ with three components, where v_{id} is the id number of the link. v_{ratio} indicates the ratio of this link traversed in a trip (Since a trip may start or end at the middle of a link, for the first and last links in a trip, $v_{\text{ratio}} \in (0, 1]$, otherwise $v_{\text{ratio}} = 1$). v_{status} denoted as the status of the link at the trip departure time. v_{status} including five types: 0—unknown, 1—clear, 2—slow, 3—congestion, and 4—gridlock.

Definition 2 (Link Network): The basic link network is represented as a directed graph of links $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V}$ is the set of graph nodes, that is, links in the traffic network. Fig. 3 shows an example of graph representation where the 11 links in the link network are represented by 11 nodes in the graph, and connectivity are represented by edges in the graph. The connectivity between links is represented by $\mathcal{A}$, where $\mathcal{A}_{ij} = 1$ denotes that link v_j is the following one of link v_i, i.e., vehicles can travel from v_i to v_j.

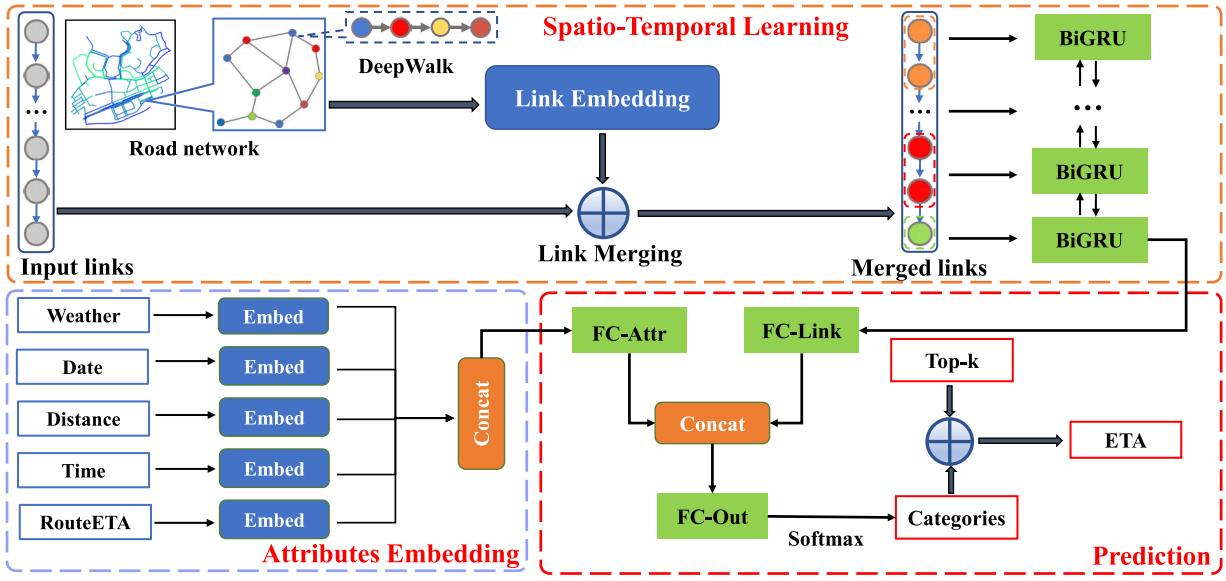


Fig. 4. Proposed CatETA consists of three core components: spatio-temporal learning, attribute embedding, and prediction. The prediction component combines information from both the spatio-temporal learning and attribute embedding components and predicts the category to which the trip ETA belongs.

Definition 3 (Trip): A trip x^i is defined as $x^i = \{s^i, o^i, d^i, V^i\}$, where s^i is the departure time, o^i is the origin location, d^i is the destination, $V^i = \{v_1, v_2, \ldots, v_n\}$ represents a sequence of links that the trip traverses on.

Definition 4 (Estimated Travel Time): For a given query $q^i = [o^i, d^i, s^i]$, the goal of ETA is to predict the travel time $\hat{y}^i$ using the formed trip and external attributes (such as week date, distance, weather conditions) of this query. We assume that V^i in trip x^i are designated by the user or generated by a routing algorithm of service providers, and this assumption was also adopted in previous literature and in practical navigation systems [1], [6], [13].

IV. Methodology

In this section, we give a detailed description of the proposed CatETA approach. As depicted in Fig. 4, the proposed model consists of three components: *spatio-temporal learning* component, *attributes embedding* component, and *prediction* component. The spatio-temporal learning component comprises two steps for extracting spatio-temporal features from link sequences. We begin by merging spatially neighboring links according to the graph embedding of the link network. A bi-directional gated recurrent unit (BiGRU) is further applied to learn representations of fine-grained temporal features. The attributes embedding component converts travel times to categorical values and embeds external attributes such as departure time, date, distance, and weather conditions. Finally, the prediction component takes the outputs of the preceding components as inputs and uses categorical approximation to estimate the travel time. Following elaborates on the details of these components.

A. Spatio-Temporal Learning Component

1) Link Merging: As mentioned previously, a trip comprises a link sequence and a set of external attributes. Existing research extracts temporal dependencies from link sequences via recurrent neural networks (RNN), such as long short-term memory (LSTM) [1], [18]. In practice, however, a trip is divided into short links of varying lengths, which results in the link sequence length fluctuating between different trips, with the shortest being less than 10 and the longest greater than 500. Directly feeding such link sequences into an RNN may incur massive computational costs [18]. To tackle this drawback, we merge spatially adjacent links before extracting the temporal features using RNN. This process reduces fluctuations in link sequence length while also reducing subsequent computational burden (See spatio-temporal learning component in Fig. 4, where the input links are merged after link embedding and merging, and the merged links with similar properties have the same color).

Given that we have a link network $\mathcal{G}$ denoting adjacent relationships. Graph embedding, which maps a graph to an appropriate low-dimensional vector space, has been shown to be a highly effective approach to characterizing spatial relationships [14], [36], [37]. A general expression of graph embedding is to learn a mapping function $f : \mathbb{R}^{N\times N} \rightarrow \mathbb{R}^{N\times h}$ that embeds a sparse network $\mathbb{R}^{N\times N}$ into low-dimensional representations $\mathbb{R}^{N\times h}$, where $h \ll N$. In this work, we utilize DeepWalk to learn the representation of links as it exhibits significant performance in various network embedding tasks [37]. Recall that we have a link network $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, DeepWalk first generates walk sequences starting from a specific link v in $\mathcal{V}$ by random walks. The transition probability between link v_i and v_j is defined as:

$$\Pr(v_i|v_j) = \frac{d_{ij}}{\sum_{v_i\in\mathcal{N}(v_j)} d_{ij}}, \quad v_i \in \mathcal{N}(v_j), \tag{1}$$

where $\mathcal{N}(v_j)$ is the set of adjacent links of j, and d_{ij} is the distance from link v_i to v_j. After generating walk sequences by random walks, the objective is to estimate the probability that the next link of the walking sequence $\{v_0, v_1, \ldots, v_{i-1}\}$

is v_i, which can be formulated as $\Pr(v_i|\{v_0, v_1, \ldots, v_{i-1}\})$. As graph embedding methods are designed to learn a dense representation of the sparse network, a mapping function $\theta : v \in \mathcal{V} \rightarrow \mathbb{R}^{|\mathcal{V}\times d|}$ is introduced to embed links in the graph into latent vectors. In this work, θ is an empirical two-layer neural network whose input is the one-hot encoding of link v_i, and the output $\theta(v_i) \in \mathbb{R}^d$ is the embedding vector. Thus, the problem can be formulated as,

$$\Pr\left(v_i|\{\theta(v_1), \theta(v_1), \ldots, \theta(v_{i-1})\}\right). \tag{2}$$

Instead of calculating Eq. 2, DeepWalk applies Skip-Gram [38] for relaxation, which ignores the order constraint in the sequence and calculate the probability $\Pr(v_k|\theta(v_i))$, where v_k is the link in the walking sequence of v_i. The objective can be formulated as,

$$\arg\min_{\theta} -\log \Pr\left(\{v_{i-w}, \ldots, v_{i+w}\}|\theta(v_i)\right), \tag{3}$$

where v_i is a link in walking sequence $\{v_{i-w}, \ldots, v_{i+w}\}$, w is the window size, respectively. The learning process aims to maximize the probability that adjacent links of v_i are in the walking sequence of a window size w; in this work, we set the walk length to 30 and the window size to 10, which is the combination that performed best in preliminary experiments. Thus, links with common neighbors are expected to have similar embeddings.

Next, we use K-Means clustering to group the links into M classes based on their embeddings. Since spatially neighboring links shall be assigned to the same class after clustering, the original link sequences can be merged according to the clustering labels. This is illustrated in Fig 4, where links are colored according to the clustering result: links with the same color are merged. The merged link is denoted by $V' = \{v'_1, v'_2, \ldots, v'_d\}$, whereas the original one is $V = \{v_1, v_2, \ldots, v_n\}$, where $d < n$. This procedure can be expressed as $\{v_i, v_{i+1}, \ldots, v_j\} \rightarrow v'$, and the number of original links merged in v' is denoted as $|v'|$. Note that the number of original links contained in each merged link may vary, as does the length of the merged link sequence. The link features are aggregated by the following rules:

$$v'_{\text{ratio}} = \frac{1}{|v'|}\sum v_{\text{ratio}}, \tag{4a}$$

$$v'_{\text{status}} = \frac{1}{|v'|}\sum v_{\text{status}}, \tag{4b}$$

$$v'_{\text{len}} = \frac{|v'|}{n}. \tag{4c}$$

The link ratio and status are averaged. v'_{len} is designed to represent the weight of v' in the whole link sequence V. l_{status} are categorical attributes including the five different type of link status in the definition. We encode v_{status} to $\mathbb{B}^5$ by one-hot. Finally, we concatenate $\{v'_{\text{ratio}}, v'_{\text{status}}, v'_{\text{len}}\}$ as link feature and obtain a merged link feature sequence L' of shape $\mathbb{R}^{d\times 7}$.

2) Temporal Feature Representation: With link sequences being merged according to their spatial neighborhood, we can obtain preliminary spatio-temporal information. However, empirical results indicate that direct and exclusive use of these features produced poor results for ETA. Owing to the GRU design, which is based on temporal state propagation that can handle sequences with variable length [39], we employ this structure to further extract high-dimensional spatio-temporal correlations.

GRU is a practical variant of RNN with a straightforward structure. By employing reset and update gates, GRU overcomes the gradient vanishing problem inherent in conventional RNN when processing long sequences and has a more straightforward structure compared to LSTM [40], [41]. Reset and update gates are gated units that determine whether to save or remove previous step information in the next state. Given a sequence $\boldsymbol{x}$ as input, the hidden state at time step t can be calculated by

$$r_t = \sigma\left(W_r \cdot \left[h_{t-1}, x_t\right]\right), \tag{5a}$$

$$z_t = \sigma\left(W_z \cdot \left[h_{t-1}, x_t\right]\right), \tag{5b}$$

$$\tilde{h}_t = \tanh\left(W \cdot \left[r_t * h_{t-1}, x_t\right]\right), \tag{5c}$$

$$h_t = z_t * \tilde{h}_t + (1 - z_t) * h_{t-1}, \tag{5d}$$

where r_t and z_t are the reset and update gates, h_{t-1} is the previous hidden state, $\tilde{h}_t$ is the current candidate state, W are trainable weights that can be updated via backpropagation, σ is the sigmoid activation function, and $*$ denotes the Hadamard product. As the traffic status on adjacent links are mutually influencing, the correlation between links are bidirectional. In this study, we follow the design of previous ETA studies [13], [42] that utilize bidirectional recurrent layers as the feature extraction component and stack two GRU layers to capture both forward and backward data propagation in time [43]. In the proposed model, we feed the merged link sequence $v' = \{v'_1, v'_2, \ldots, v'_d\}$ to a BiGRU module. The forward and backward hidden states at time t are denoted by $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$, respectively. States $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ are concatenated as the output hidden state $h_t = \overrightarrow{h_t}||\overleftarrow{h_t}$. The hidden size of the BiGRU cells is empirically set to 256.

B. Attributes Embedding Component

In aforementioned spatio-temporal learning component, fine-grained spatio-temporal features are learned from link sequences. Next, we focus on analyzing and mining other trip attributes.

As shown in Fig. 2, travel time follows a skewed distribution; that is, while the majority of trips take less than 40 min, there are some particularly long ones. A common approach in existing work is to normalize travel time and then train a regression model for prediction. Neither z-score nor min-max normalization, however, are capable of adequately normalizing such a skewed distribution. Thus, we propose to convert travel time to categorical values and solve the ETA task in a classification manner. Besides the above features, we utilize a statistical value, which is denoted as *Route-ETA*. *Route-ETA* is a simple accumulation of the average travel time for each link in a trip, calculated by the distance of the link divided by the historical speed at the trip departure time. Although *Route-ETA* ignores the dynamic changes in traffic and can cause an accumulation of errors, it can still be used as an additional information source. If we divide travel time

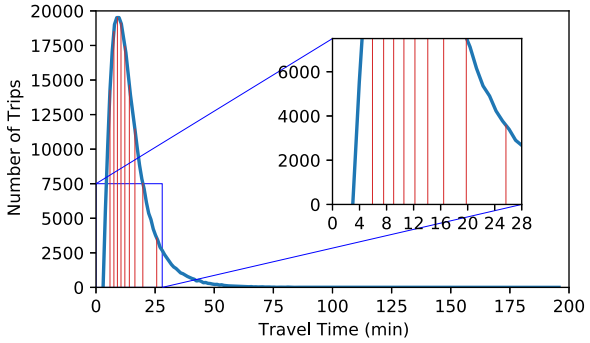


Fig. 5. Example of dividing travel time into 10 classes, where the blue curve is the distribution of travel time during a weekday and red vertical lines are the class thresholds. Since the number of trips in each class is the same, the thresholds are not equally spaced.

into C classes, we get the mathematical expression for class thresholds:

$$\int_{\beta_i}^{\beta_{i+1}} f(t)\mathrm{d}t = \frac{1}{C}, \quad \forall i \in [1, C], \tag{6}$$

where $f(t)$ is the probability density function of travel time in the train set and β_i is the threshold of class i, respectively. As illustrated in Fig. 5, with the number of classes C set to 10, trips are divided into ten classes by multiple thresholds (red vertical lines). Subject to Eq. 6, the trip counts in all classes are equal. Therefore, the thresholds are more tight between 8 min and 16 min as individual trip time more likely falls into this range, and is more sparsely spaced otherwise where there are less long and short trips. The label c_i of class i is defined as the average travel time for all trips within the class.

Since all trips in a class are represented by a single categorical label, there is a bias between the assigned label and the ground truth, which in turn may compromise prediction accuracy. We assess the bias in categorical labeling by using all trips during a weekday and converting their respective travel time to categorical values. Table I shows the average Mean Absolute Percentage Error (MAPE) and Mean Absolute Error (MAE) in an ideal situation, i.e., each class is correctly classified. It is evident that dividing travel time into more categories can reduce the bias of the prediction. However, as the number of categories C increases, it becomes harder to accurately classify each category in practice. Nevertheless, the result also indicates that the accuracy is acceptable when a sufficient number of bins is used though categorizing by travel time inevitably renders information loss. There is a clear trade-off between class number and prediction accuracy, and detailed results of this categorical formulation will be shown in Sec. V-F.

In addition, external attributes have a positive effect on travel time estimation accuracy [1], [6], [44]. We consider travel distance, weather condition, departure time, and day of week in this work. The travel distance is normalized by z-score, and the remaining three categorical attributes are embedded into low-dimensional vectors using linear linear layers [1], [6], [44]. The linear layer is formulated as $y = Ax^T + b$, where x is the one-hot encoded vector and

TABLE I
BIAS IN CATEGORICAL LABELING OVER NUMBER OF CLASSES. IN AN IDEAL SITUATION WHERE EACH CATEGORY IS CLASSIFIED CORRECTLY

# Class C	10	50	100
MAPE (%)	8.06	1.91	0.82
MAE	77.43	21.07	11.33

$A \in \mathbb{R}^{D\times V}$ is the weight matrix, V is the vocabulary size, and D is the embedding dimension. In this study, the departure time in a day is divided into 288 slices (5 min time window), and the weather condition is recorded as five types, namely rainstorm, heavy rain, moderate rain, showers, and cloudy. We embed departure time to $\mathbb{R}^8$, day of week to $\mathbb{R}^3$, and weather to $\mathbb{R}^3$.

Besides the above features, we utilize a statistical value, which is denoted as *Route-ETA*. *Route-ETA* is a simple accumulation of the average travel time for each link in a trip, calculated by the distance of the link divided by the historical speed at the trip departure time. Although *Route-ETA* ignores the dynamic changes in traffic and can cause an accumulation of errors, it can still be used as an additional information source. Similar to the travel time, we convert *Route-ETA* into categorical values using the same threshold β. The embedding of each attribute is concatenated as the output of the attributes embedding component.

C. Prediction Component

After learning spatio-temporal features and embedding external attributes, the fully connected (FC) layer is applied to extract features further and make predictions, which is a common practice in previous ETA models [1], [14]. As demonstrated in Fig. 4, the prediction component is constructed by three groups of consecutive FC layers, namely, `FC-link`, `FC-attr` and `FC-out`. `FC-link` accepts the hidden vector of last BiGRU cell as input, and `FC-attr` accepts as input the concatenated attributes embedding. `FC-link` and `FC-attr` both have the same hidden and output layer sizes, which are empirically set to 1024 and 128, respectively. The outputs of `FC-link` and `FC-attr` are first concatenated and then fed to `FC-out`. The hidden size of `FC-out` is empirically set to 1024, while the output size depends on the number of classes C. Except for the output layer in `FC-out`, we apply the Rectified Linear Unit (ReLU) activation function to the others. At training time, we apply dropout [45] with probability 20% to all FC layers to prevent overfitting.

Following the three groups of FC layers, the output of `FC-out` is processed by a softmax function which generates probabilities over different classes:

$$p_i = \mathrm{softmax}(z_i) = \frac{e^{z_i}}{\sum_{i=1}^{C} e^{z_i}}, \tag{7}$$

where C denotes total classes and p_i is the probability that the input belongs to class i. The loss function $\mathcal{L}_s$ is defined

by the categorical cross entropy loss:

$$\mathcal{L}_s = -\sum_i y_i \log(p_i), \tag{8}$$

where y_i is the binary ground-truth value that the input belongs to class i. $\mathcal{L}_s$ can be minimized through gradient descent.

D. Categorical Approximating Estimation Approach

In previous subsections, we introduce the structure of three components that learns spatio-temporal features from the link sequence and embeds external factors. In addition, we divide travel times into groups and train a classifier to solve the ETA task. Nevertheless, as travel times are divided into groups, using categorical labels directly as the final prediction may undermine the model performance, particularly for trips with travel times close to their assigned group's thresholds. Thus, we design a categorical approximating approach to reduce bias in the model inference stage. Specifically, we take into account the model confidence in classifying the input to each class. we select those categories with relatively high confidence and estimate travel time by the weighted average of the corresponding categorical labels. This is based on the hypothesis that if a trip has similar confidence in several classes, its travel time is close to the threshold of these categories. In this work, we use the output probabilities of the softmax function as the classification confidence and choose the classes with top-k probabilities as $\mathcal{K}$. Hyperparameter k defines the degree of approximation by indicating the number of categories for approximation. The weighted approximating estimation can be formulated as:

$$\hat{y}_i = \sum_{j \in \mathcal{K}} \frac{p_j c_j}{\sum_{j \in \mathcal{K}} p_j}, \tag{9}$$

where $\hat{y}_i$ is the estimated travel time of trip i, p_j is the probabilistic output of the softmax activation function, c_j is the class label, and $\mathcal{K}$ is the set of classes with relatively high p_j.

V. Case Studies

In this section, we first evaluate the proposed CatETA on a large-scale, real-world trajectory dataset and compare our approach with established ETA baselines. We then investigate the sensitivity of CatETA to hyperparameter variations. Finally, we analyze the generality of CatETA by constructing a link-based dataset from GPS trajectories.

A. Dataset

We perform our experiments on a real-world ride-hailing dataset provided by the Didi Chuxing GAIA Project.[1] The dataset contains approximately nine million trip records from August 1st to August 31st, 2020. During pre-processing stage, we remove abnormal records with very short travel time (less than 3 min) and distance (less than 1 km). There is a sparse adjacency table that records the adjacent relationships between links, which contains 882, 718 links. Each link has an average of 2.6 neighbors. The travel times of each link in a trip are unavailable in this dataset. Since the dataset is collected in urban areas, the trained model may only be applicable for prediction in urban scenarios. It is important to note that, while there are several datasets available for ETA, they are based on GPS trajectories alone and are relatively less practical for use as per analyses in Sec. II-A. The selected dataset from the GAIA Project is the *only* publicly accessible link-based dataset for ETA at the current stage.

[1]This dataset can be downloaded (following approval of access request) at https://outreach.didichuxing.com/research/opendata

B. Experimental Settings

The following hyperparameters are used in the experiments:

- Link cluster number M: The default setting is $M = 80,000$, which is about 10% of links in the network.
- Travel time class number C: We set $C = 50$ by default, which means the travel time is divided into 50 categories.
- Categorical approximating k: The travel time is estimated by weighting the class with the top-k softmax probabilities. The default setting is $k = 5$.

For cross-validation, we select the last week in the dataset as the test set. The rest of the dataset is sequentially divided into training set and validation set with a ratio of 3 : 1. The model is trained using the Adam optimizer [46] with an initial learning rate of 0.001, while the batch size is set to 1024. All models are implemented using PyTorch [47], and all case studies are conducted on a server with an NVIDIA GeForce RTX 2080Ti GPU.

C. Evaluation Metrics

We select the Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), and Satisfaction Rate (SR) as the measures of estimation accuracy; these metrics are commonly used in ETA [6], [14]. They are defined as follows:

$$\text{MAPE}(y, \hat{y}) = \frac{1}{N}\sum_i^N \left|\frac{y^{(i)} - \hat{y}^{(i)}}{y^{(i)}}\right| \times 100\%, \tag{10}$$

$$\text{MAE}(y, \hat{y}) = \frac{1}{N}\sum_i^N \left|y^{(i)} - \hat{y}^{(i)}\right|, \tag{11}$$

$$\text{SR}(y, \hat{y}) = \frac{1}{N}\sum_i^N \left(\left|\frac{y^{(i)} - \hat{y}^{(i)}}{y^{(i)}}\right| \leq 15\%\right) \times 100\%, \tag{12}$$

where $\hat{y}^{(i)}$ is the estimated travel time of trip i, $y^{(i)}$ denotes the ground truth, and N is the number of trips.

D. Baselines

We compare the proposed model with the following ETA methods:

- **Historical Average (HA):** HA estimates travel time by calculating the average speed during a specific time from historical records. As the departure time is divided into 5-minute time slices in this dataset, we average the trips in the same time slice.
- **TEMP:** TEMP [8] is a route-free baseline that estimates travel time by querying and averaging all the neighboring

TABLE II
PERFORMANCE COMPARISON OF DIFFERENT APPROACHES. MAPE AND SR ARE REPORTED IN PERCENTAGE (%)

	Total			Short (< 10 min)			Medium (10–20 min)			Long (> 20 min)		
	MAPE	MAE (s)	SR	MAPE	MAE (s)	SR	MAPE	MAE (s)	SR	MAPE	MAE (s)	SR
HA	24.24	221.84	38.32	22.59	95.24	40.19	23.70	202.62	38.50	28.59	506.53	34.34
TEMP	20.58	136.85	48.65	23.01	92.27	45.74	18.06	148.67	52.21	19.73	312.99	47.16
XGBoost	17.76	142.87	52.19	21.35	84.65	45.27	15.71	132.58	56.72	15.50	276.63	55.30
Route ETA	16.12	140.31	53.78	16.31	67.66	54.07	15.71	133.70	54.56	16.67	292.81	51.52
DeepTTE (merge)	14.97	125.00	58.86	16.40	67.84	54.58	14.51	121.89	59.88	13.29	240.95	64.75
WDR	15.73	124.36	58.29	20.15	77.37	47.33	13.24	110.87	64.00	12.90	240.30	66.28
CatETA (no merge)	15.01	125.07	58.78	16.60	69.26	53.78	14.50	121.60	59.95	13.09	239.25	65.71
CatETA (regression)	15.09	122.57	59.65	17.81	71.19	52.16	13.38	113.42	64.37	13.68	240.97	63.49
CatETA	**13.92**	**116.07**	**62.52**	**15.83**	**65.31**	**55.92**	**12.94**	**109.02**	**65.79**	**12.46**	**227.85**	**67.81**

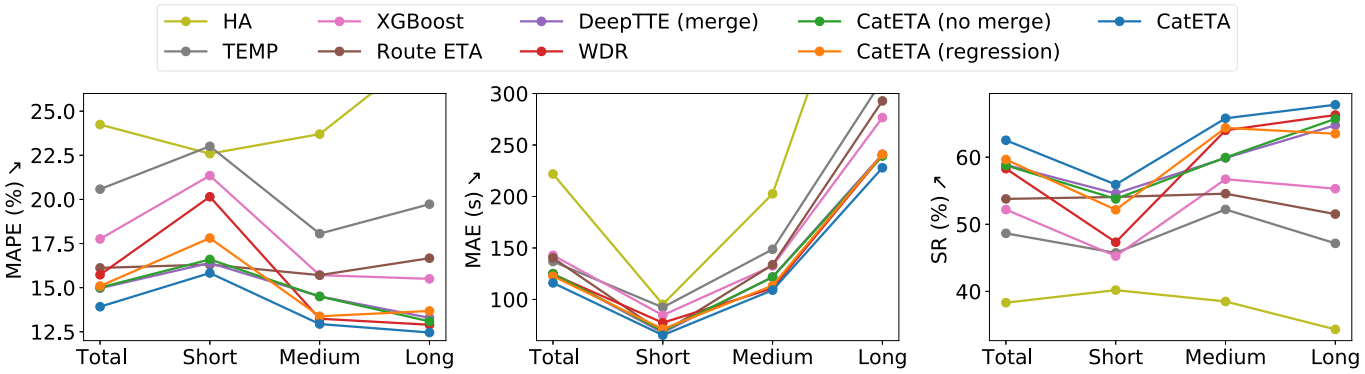


Fig. 6. Performance comparison of different approaches. MAPE and SR are reported in percentage (%).

trips. Trips with the same or nearby starting and destination links are considered as neighboring trips. In this work, we retrieve ten neighboring trips for each query and average the travel time, which follows the setting in previous study [14].

- **XGBoost:** XGBoost [48] is an established ensembling method. Since the input of XGBoost must have fixed length, we aggregate the link sequence of each trip to a fixed length and then concatenate it to the head part. The length is set to 20, which is close to the average length of the merged link sequences.
- **RouteETA:** RouteETA estimates travel time by adding the average travel time of each link at the trip departure time. The real-time travel time of each link is collected via online map services. RouteETA is used as an external attribute in the proposed methods. For a fair comparison, RouteETA is used as input to all other baselines as well.
- **DeepTTE (merge):** DeepTTE [1] is among the representative trajectory-based studies. It leverages a neural network to convert each raw GPS trajectory into a series of features, before applying an RNN to capture spatio-temporal dependencies. Since this model is not applicable to link-based trip records, we replace the feature extraction layer with our proposed link merging method.
- **WDR:** WDR [6] is a widely deployed link-based ETA method which combines wide, deep, and recurrent neural networks, achieving highly competitive prediction accuracy.
- **CatETA (no merge):** CatETA (no merge) is a simplified version of the proposed model, in which link merging in the spatio-temporal learning component is removed.
- **CatETA (regression):** CatETA (regression) is a regression version of CatETA that uses the same network structure and link merging method. Unlike CatETA, the loss function of the regression version is the mean square error and travel time is normalized by z-score.

E. Performance Evaluation

We first evaluate the performance of the proposed CatETA against the selected baselines. The results, summarized in Table II and Fig. 6, clearly demonstrate that CatETA significantly outperforms all other methods on the three evaluation metrics. Furthermore, both CatETA (no merge) and CatETA (regression) achieve competitive results, illustrating the efficacy of the proposed link merging and classification approximation methods. It is worth noting that our classification scheme (13.92%) outperforms the pure regression scheme (15.09%) by 1.17% with the same model structure and link merging method. Meanwhile, DeepTTE (merge) also provides compelling results in favor of the superior performance of link merging. While DeepTTE is designed for GPS trajectories, it can still produce competitive results with minor modifications. In summary, the above results show that CatETA is highly effective for link-based ETA and achieves state-of-the-art results.

Table II also shows the performance of each method over a range of trip lengths. We categorize trips into three groups

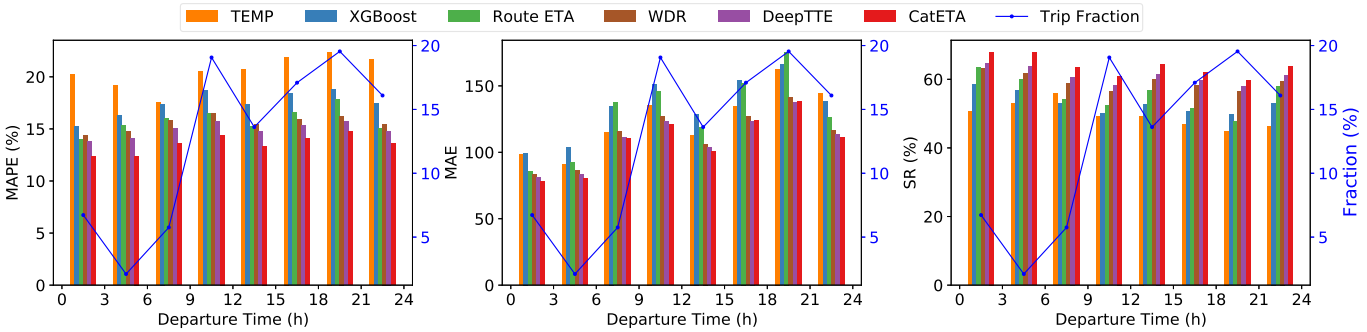


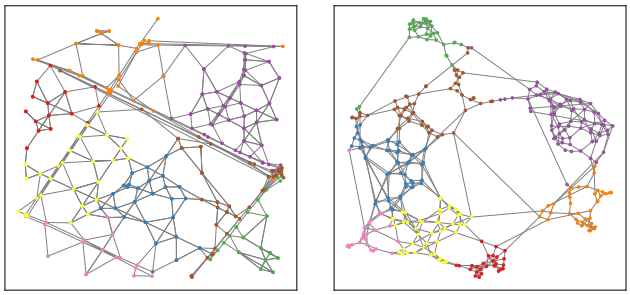
Fig. 7. Performance comparison for trips with different departure times.

(short, medium, and long) based on their travel time, with each group accounting for 37%, 43%, and 20% of the test set. It appears that all methods perform well in the long trip scenario, and the performance of the model degrades as the trip duration decreases. However, CatETA shows significant advantages for short trips over regression-based methods, i.e., CatETA (regression), XGBoost, DeepTTE, and WDR. This is because our classification scheme eliminates the adverse effect of extremely long trips in this skewed distribution. On the other hand, the regression model is easily influenced by long trips, undermining its performance for short trips. This result indicates that our approach is more practical than the baselines in real-world scenarios that involve a non-negligible amount of short trips.

Due to the fact that the distribution of trip volume varies throughout the day, we further investigate the performance of CatETA at various time intervals. We divide the day into eight three-hour time windows according to departure time and examine the performance of different models during each period. As shown in Fig. 7, increasing trip volume has a detrimental effect on the performance of CatETA, which is corroborated by the results of other models. The reason is that increased trips also mean more vehicles on the road, causing traffic congestion and therefore complexity that ultimately impairs model performance. There are two peaks in a day, 9:00–12:00 and 18:00–21:00, which correspond to the peak periods for commuting to and from work, respectively. Correspondingly, performance during these periods is slightly lower than during others. Nevertheless, CatETA can still produce better estimates than the evaluated baselines for all time windows throughout the day.

F. Hyperparameter Sensitivity Test

In this section, we first investigate the impact of the number of clusters M, which determines the performance of model and computational requirements. Fig. 8 shows the projection of the graph-embedded learning representation within a region. We can observe that the graph embedding approach is able to cluster links with similar embedding representations in the road network. Recall that there are $882,718$ links in the entire network; we set the number of clusters $M \in \{160000, 80000, 16000, 8000\}$, which is approximately $\{20\%, 10\%, 2\%, 1\%\}$ of the original value, respectively. We also provide the average number of merged links

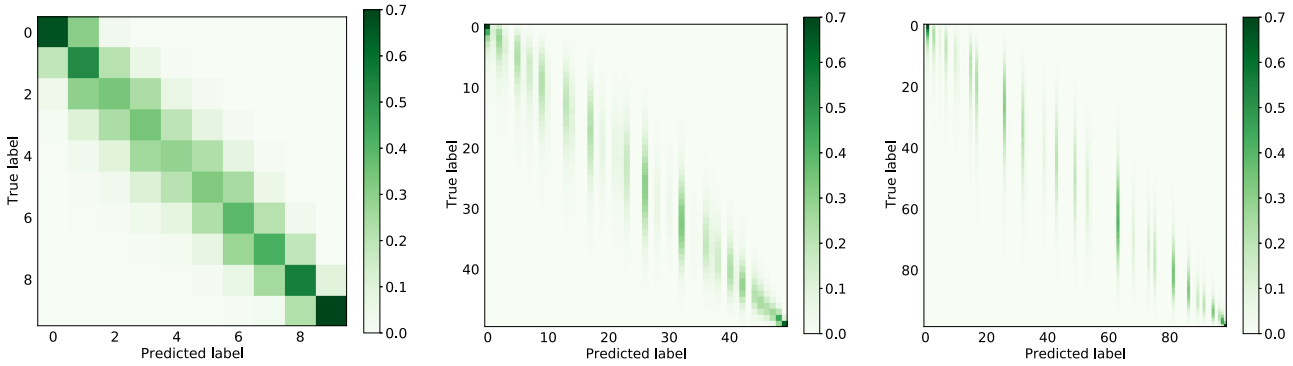


(a) Network Topology with Graph Embedding. (b) The link clusters produced by Graph Embedding

Fig. 8. Visualization of graph clustering. The same color indicates that the links are in the same cluster. Links with similar embedding representations can be clearly separated.

in each trip and the training time per epoch to demonstrate how our scheme can significantly reduce computational costs. As shown in Table III, we can see that the average length of sequences (links in trips) and the epoch time decreases with M. Compared to using the original sequence directly, the link merging method improves the estimated MAPE from 15.01% to 13.92% while reducing the computation time by half. This is because lengthy sequences introduce a lot of extraneous noise. The main link state information can be preserved by merging the links, thereby improving the model accuracy. In addition, our link merging scheme is capable of merging links with similar embedding representations. The computational cost of BiGRU is proportional to sequence length, thus significantly reducing the training cost. However, the adage "smaller is better" is not true for M, as merging too many links may discard too much link information. This can also be verified from the results displayed in Table III, where the average length and epoch time are reduced despite the fact that accuracy decreases.

To quantify the impact of two critical hyperparameters in the proposed categorical approximate approach, i.e., the travel time class number C and the categorical approximating k, we choose $C \in \{10, 50, 100\}$ and set $k \in \{1, 3, 5\}$, $k \in \{1, 5, 10\}$ and $k \in \{1, 10, 20\}$ for each C value, respectively. The simulation results summarized in Table IV demonstrate that both a large and a small C can affect model performance, and that $C = 50$ is the best among the three settings. This



(a) $C = 10$ (b) $C = 50$ (c) $C = 100$

Fig. 9. Normalized confusion matrix.

TABLE III
PERFORMANCE OF THE PROPOSED MODEL OVER NUMBER OF CLUSTERS

# Cluster M	# Links per Trip	Epoch Time (s)	MAPE (%)
882,718	87.83	81.77	15.01
160,000	41.13	49.28	14.48
80,000	28.21	41.97	**13.92**
16,000	15.34	34.18	14.47
8,000	12.99	30.31	14.32

TABLE IV
PERFORMANCE OF THE PROPOSED MODEL OVER NUMBER OF CLASSES C AND APPROXIMATING TOP-k

C	k	MAPE(%)	MAE(s)	SR(%)	Acc(%)
10	1	16.51	146.10	54.19	46.19
	3	15.86	135.86	56.90	87.04
	5	15.98	135.18	56.97	97.27
50	1	14.30	119.91	61.31	13.76
	5	**13.92**	116.07	62.52	49.25
	10	14.11	**115.28**	**62.61**	70.73
100	1	15.32	126.71	57.85	6.84
	10	14.86	122.31	59.42	43.03
	20	14.97	121.55	59.58	69.79

is explained by a trade-off in categorizing travel time into few classes, which can produce a coarse-grained approximation result, versus having too many classes that are hard to classify, thus reducing the accuracy of the model. Indeed, one may observe that as C increases, the corresponding accuracy decreases. In addition, the value of k is also important since a larger k approximates the ETA with more travel time categories. However, MAPE and SR do not always improve as k increases, indicating that using a k value that is too large may introduce noise and thus reduce the approximation precision. As such, it is recommended to choose a medium number of classes, and we use $k = 0.1 \times C$ as a guideline following the previous experiments.

Finally, we present the normalized confusion matrix when categorizing travel time into different number of classes. As shown in Fig. 9, the classifications of CatETA are typically accurate with all tested number of classes C; as C increases, even if the top-1 prediction accuracy decreases, the prediction is still likely to fall into neighboring labels of the ground truth. Moreover, since neighboring categories tend to have similar confidence, using top-k can greatly improve the prediction accuracy. Therefore, we use top-k for approximation in order to transform the class into a continuous numerical prediction and maximize the reliability.

TABLE V
PERFORMANCE COMPARISON OF DIFFERENT APPROACHES ON KDD CUP 2020 DATASET. MAPE AND SR ARE REPORTED IN PERCENTAGE (%)

	MAPE	MAE (s)	SR
TEMP	22.60	145.64	44.94
XGBoost	21.43	134.37	46.32
Route ETA	22.49	151.15	41.43
DeepTTE (merge)	16.77	112.74	56.38
WDR	18.45	134.66	51.58
CatETA	**15.82**	**111.28**	**58.83**

G. Generality of CatETA

As mentioned previously, the case studies are conducted on the only public link-based ETA dataset. In this section, we construct a link-based ETA dataset from GPS trajectories to analyze the generality of CatETA. Specifically, the GPS trajectories are from KDD CUP 2020 dataset[2] which is collected by ride-sharing vehicles in Chengdu, China, from Nov 1st, 2016 to Nov 30th, 2016. It records approximately 20 thousand trips and 30 million GPS points a day with a sampling interval of 2–4 s. To project GPS trajectories onto the road network, we first obtain the network topology from OpenStreeMap[3] and then perform map matching by hidden Markov model [49]. With GPS trajectories matched to the map, consecutive GPS points on the same road segment are merged as a traversed link, and a trip can be represented by a sequence of links.

For performance evaluation, we choose TEMP, XGBoost, RouteETA, DeepTTE (merge) and WDR as the baselines

[2]This dataset can be downloaded (following approval of access request) at https://outreach.didichuxing.com/app-vue/KDD_CUP_2020

[3]https://www.openstreetmap.org/

(c.f. V-D). All the experimental settings are kept the same as in Section V-B for a fair comparison. As summarized in Table V, CatETA has the best performance on three evaluation metrics compared to other baselines, which follows the same trend as the results in Table II. The results indicate that CatETA can be generalized broadly when applied to other datasets.

VI. Conclusion

In this paper, we propose a novel categorical approximate approach for estimating the time of arrival based on a series of traversed links. Concretely, we introduce CatETA to jointly learn spatio-temporal features from link sequences and external attributes in a set of global information. In the proposed approach, to prevent the regression model from being skewed, we formulate travel time as a categorical value and design a categorical approximate approach. In particular, we estimate the travel time via the weighted average of different classes to eliminate bias in categorical labeling. Additionally, we merge spatially adjacent links according to graph embedding, thereby reducing the computational burden of long link sequences.

To evaluate the performance of CatETA, we conduct a series of comprehensive case studies on a real-world link-based ETA dataset. The simulation results demonstrate that the proposed method significantly outperforms state-of-the-art baselines in all scenarios. Subsequently, our hyperparameter sensitivity tests show that categorical approximation effectively help avoid skewing the regression model and that merging neighboring links can considerably alleviate computational costs. Furthermore, we construct a link-based dataset from GPS trajectories which reveals the generality of our model on other cities and datasets. In future work, we will explore ETA models that can be adapted to special scenarios, such as in the event of a traffic accident, that are closer to real applications.

References

[1] 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.*, New Orleans, AK, USA, Apr. 2018, vol. 32, no. 1, pp. 2500–2507.

[2] Y. Wang, Y. Zheng, and Y. Xue, "Travel time estimation of a path using sparse trajectories," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, 2014, pp. 25–34.

[3] C.-H. Wu, J.-M. Ho, and D.-T. Lee, "Travel-time prediction with support vector regression," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 276–281, Dec. 2004.

[4] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: Concepts, methodologies, and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 5, no. 3, Sep. 2014, pp. 1–55.

[5] H. Hong *et al.*, "HetETA: Heterogeneous information network embedding for estimating time of arrival," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 2444–2454.

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

[7] S. Maiti, A. Pal, A. Pal, T. Chattopadhyay, and A. Mukherjee, "Historical data based real time prediction of vehicle arrival time," in *Proc. 17th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Qingdao, China, Oct. 2014, pp. 1837–1842.

[8] 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, 2019.

[9] W.-C. Lee, W. Si, L.-J. Chen, and M. C. Chen, "HTTP: A new framework for bus travel time prediction based on historical trajectories," in *Proc. 20th Int. Conf. Adv. Geographic Inf. Syst. (SIGSPATIAL)*. Redondo Beach, CA, USA, Association for Computing Machinery, 2012, pp. 279–288.

[10] C. de Fabritiis, R. Ragona, and G. Valenti, "Traffic estimation and prediction based on real time floating car data," in *Proc. 11th Int. IEEE Conf. Intell. Transp. Syst.*, Beijing, China, Oct. 2008, pp. 197–203.

[11] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," *SIGMOD Rec.*, vol. 29, no. 2, pp. 1–12, 2000.

[12] 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*. New York, NY, USA: Association for Computing Machinery, 2013, pp. 713–724.

[13] H. Zhang, H. Wu, W. Sun, and B. Zheng, "DeepTravel: A neural network based travel time estimation model with auxiliary supervision," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Stockholm, Sweden, 2018, pp. 3655–3661.

[14] 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.

[15] J. Weng, C. Wang, H. Huang, Y. Wang, and L. Zhang, "Real-time bus travel speed estimation model based on bus GPS data," *Adv. Mech. Eng.*, vol. 8, no. 11, Nov. 2016, Art. no. 168781401667816.

[16] B. Yang, C. Guo, and C. S. Jensen, "Travel cost inference from sparse, spatio temporally correlated time series using Markov models," *Proc. VLDB Endowment*, vol. 6, no. 9, pp. 769–780, Jul. 2013.

[17] M. As and T. Mine, "Dynamic bus travel time prediction using an ANN-based model," in *Proc. 12th Int. Conf. Ubiquitous Inf. Manage. Commun.* Langkawi, Malaysia: Association for Computing Machinery, 2018, pp. 1–8.

[18] K. Fu, F. Meng, J. Ye, and Z. Wang, *CompactETA: A Fast Inference System for Travel Time Prediction*. Virtual Event, CA, USA: Association for Computing Machinery, 2020, pp. 3337–3345.

[19] T.-Y. Fu and W.-C. Lee, "DeepIST: Deep image-based spatio-temporal network for travel time estimation," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Nov. 2019, pp. 69–78.

[20] M. Kormaksson, L. Barbosa, M. R. Vieira, and B. Zadrozny, "Bus travel time predictions using additive models," in *Proc. IEEE Int. Conf. Data Mining*, Shenzhen, China, Dec. 2014, pp. 875–880.

[21] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, "Preserving privacy in GPS traces via uncertainty-aware path cloaking," in *Proc. 14th ACM Conf. Comput. Commun. Secur. (CCS)*. New York, NY, USA: Association for Computing Machinery, 2007, pp. 161–171.

[22] Y. Zhu, Y. Liu, J. J. Q. Yu, and X. Yuan, "Semi-supervised federated learning for travel mode identification from GPS trajectories," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 1–12, Mar. 2021.

[23] M. Asghari, T. Emrich, U. Demiryurek, and C. Shahabi, "Probabilistic estimation of link travel times in dynamic road networks," in *Proc. 23rd SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, Seattle, WA, USA: Association for Computing Machinery, Nov. 2015, pp. 1–10.

[24] E. Jenelius and H. N. Koutsopoulos, "Travel time estimation for urban road networks using low frequency probe vehicle data," *Transp. Res. B, Methodol.*, vol. 53, pp. 64–81, Jul. 2013.

[25] 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)*, Hague, The Netherlands, Oct. 2013, pp. 2292–2297.

[26] J. Wang, Q. Gu, J. Wu, G. Liu, and Z. Xiong, "Traffic speed prediction and congestion source exploration: A deep learning method," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, Barcelona, Spanish, Dec. 2016, pp. 499–508.

[27] S. Xu, R. Zhang, W. Cheng, and J. Xu, "MTLM: A multi-task learning model for travel time estimation," *GeoInformatica*, vol. 26, pp. 1–17, Aug. 2020.

[28] Y. Sun *et al.*, "CoDriver ETA: Combine driver information in estimated time of arrival by driving style learning auxiliary task," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 5, pp. 1–12, May 2020.

[29] Y. Sun, K. Fu, Z. Wang, C. Zhang, and J. Ye, "Road network metric learning for estimated time of arrival," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Virtual Event, Milan, Italy, Jan. 2021, pp. 1820–1827.

[30] C. Zhang, J. J. Q. Yu, and Y. Liu, "Spatial-temporal graph attention networks: A deep learning approach for traffic forecasting," *IEEE Access*, vol. 7, pp. 166246–166256, 2019.

[31] J. J. Q. Yu, C. Markos, and S. Zhang, "Long-term urban traffic speed prediction with deep learning on graphs," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 1–12, Apr. 2021.

[32] 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.

[33] 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*. New York, NY, USA: Association for Computing Machinery, Aug. 2020, pp. 2697–2705.

[34] X. Fang, J. Huang, F. Wang, L. Liu, Y. Sun, and H. Wang, "SSML: Self-supervised meta-learner for en route travel time estimation at Baidu maps," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA: Association for Computing Machinery, Aug. 2021, pp. 2840–2848.

[35] J. J. Q. Yu, "Citywide estimation of travel time distributions with Bayesian deep graph learning," *IEEE Trans. Knowl. Data Eng.*, early access, Oct. 6, 2021, doi: 10.1109/TKDE.2021.3117986.

[36] H. Cai, V. W. Zheng, and K. C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1616–1637, Feb. 2018.

[37] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA, Aug. 2014, pp. 701–710.

[38] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proc. 1st Int. Conf. Learn. Represent. (ICLR)*, Scottsdale, AZ, USA, 2013, pp. 1–12.

[39] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*.

[40] K. Cho *et al.*, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," 2014, *arXiv:1406.1078*.

[41] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–16.

[42] F. Wu and L. Wu, "DeepETA: A spatial–temporal sequential neural network model for estimating time of arrival in package delivery system," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, no. 1, pp. 774–781.

[43] Y. Wang *et al.*, "Tacotron: Towards end-to-end speech synthesis," 2017, *arXiv:1703.10135*.

[44] 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. Discovery Data Mining*. London, U.K.: Association for Computing Machinery, Jul. 2018, pp. 1695–1704.

[45] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Advances in Neural Information Processing Systems*, vol. 29, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds. Barcelona, Spain: Curran Associates, 2016, pp. 1019–1027.

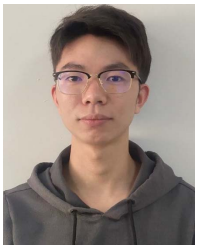
[46] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, 2015, pp. 1–15.

[47] A. Paszke *et al.*, "Automatic differentiation in PyTorch," in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, 2017, pp. 1–4.

[48] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. San Francisco, CA, USA: Association for Computing Machinery, Aug. 2016, pp. 785–794.

[49] C. Yang and G. Gidófalvi, "Fast map matching, an algorithm integrating hidden Markov model with precomputation," *Int. J. Geograph. Inf. Sci.*, vol. 32, no. 3, pp. 547–570, 2018.

Yongchao Ye (Student Member, IEEE) received the B.Eng. degree in computer science and technology from Ningbo University, Ningbo, China, in 2020. He is currently pursuing the master's degree with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. His research interests include spatio-temporal data mining in smart city, intelligent transportation systems, and federated learning.



Yuanshao Zhu (Member, IEEE) received the B.Eng. degree in communication engineering from Shandong University, Weihai, China, in 2019, and the M.S. degree in electrical science and technology from the Southern University of Science and Technology, Shenzhen, China, in 2022. He is currently pursuing the Ph.D. degree with the Southern University of Science and Technology and the City University of Hong Kong joint program. His research interests include deep learning in smart city, intelligent transportation systems, and federated learning.

Christos Markos received the Ph.D. degree in electrical and data engineering from the University of Technology, Sydney, in 2022, and the integrated master's degree in electrical and computer engineering from the University of Thessaly, Greece, in 2017. He is currently a Visiting Researcher at the Southern University of Science and Technology, Shenzhen, China. His research interests include deep learning, intelligent transportation systems, and trajectory data mining.

James J. Q. Yu (Senior Member, IEEE) received the B.Eng. and Ph.D. degrees in electrical and electronic engineering from The University of Hong Kong, Pokfulam, Hong Kong, in 2011 and 2015, respectively. He was a Post-Doctoral Fellow at The University of Hong Kong from 2015 to 2018. He is an Assistant Professor at the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China; and an Honorary Assistant Professor at the Department of Electrical and Electronic Engineering, The University of Hong Kong. His research interests are in smart city and urban computing, deep learning, intelligent transportation systems, and smart energy systems. His current work is mainly on forecasting and decision making of future transportation systems and basic artificial intelligence techniques for industrial applications. He was ranked World's Top 2% Scientists of 2019 and 2020 by Stanford University. He is an Editor of the *IET Smart Cities* journal.