Uncertainty-Aware Probabilistic Travel Time Prediction for On-Demand Ride-Hailing at DiDi

Hao Liu
The Hong Kong University of Science and Technology (Guangzhou)
The Hong Kong University of Science and Technology
liuh@ust.hk

Wenzhao Jiang
The Hong Kong University of Science and Technology (Guangzhou)
wjiang431@connect.hkust-gz.edu.cn

Shui Liu
Didichuxing Co. Ltd
liushui@didiglobal.com

Xi Chen
Didichuxing Co. Ltd
seanchenxi@didiglobal.com

ABSTRACT

Travel Time Estimation (TTE) aims to accurately forecast the expected trip duration from an origin to a destination. As one of the world's largest ride-hailing platforms, DiDi answers billions of TTE queries per day. The quality of TTE directly decides the customer's experience and the effectiveness of passenger-to-driver matching. However, existing studies mainly regard TTE as a deterministic regression problem and focus on improving the prediction accuracy of a single label, which overlooks the travel time uncertainty induced by various dynamic contextual factors. To this end, in this paper, we propose a probabilistic framework, ProbTTE, for uncertainty-aware travel time prediction. Specifically, the framework first transforms the single-label regression task to a multi-class classification problem to estimate the implicit travel time distribution. Moreover, we propose an adaptive local label-smoothing scheme to capture the ordinal inter-class relationship among soft travel time labels. Furthermore, we construct a route-wise log-normal distribution regularizer to absorb prior knowledge from large-scale historical trip data. By explicitly considering the travel uncertainty, the proposed approach not only improves the TTE accuracy but also provides additional travel time information to benefit downstream tasks in ride-hailing. Extensive experiments on real-world datasets demonstrate the superiority of the proposed framework compared with state-of-the-art travel time prediction algorithms. In addition, ProbTTE has been deployed in production at DiDi in late 2022 to empower various order dispatching services, and improves passenger and driver experiences significantly.

CCS CONCEPTS

• **Information systems → Spatial-temporal systems**; • **Computing methodologies → Machine learning**; • **Applied computing → Transportation**.

KEYWORDS

travel time estimation, order dispatching, deep neural networks, probabilistic forecasting

ACM Reference Format:
Hao Liu, Wenzhao Jiang, Shui Liu, and Xi Chen. 2023. Uncertainty-Aware Probabilistic Travel Time Prediction for On-Demand Ride-Hailing at DiDi. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3580305.3599925

1 INTRODUCTION

Recent years witnessed the rapid development of on-demand ride-hailing platforms (DiDi, Uber, Lyft, etc.), which deeply changed the landscape of modern transportation systems and urban mobility patterns. As one of the most essential services, Travel Time Estimation (TTE) plays an indispensable role in various ride-hailing tasks, such as route planning, order-dispatching, and dynamic pricing. In each day, the TTE service at DiDi answers over ten billion queries. Therefore, the quality of travel time prediction is not only important to the passenger's user experiences, but also plays a critical role to the driver and the business operator.

Due to its importance, travel time prediction has been extensively studied in industry and academia. To name a few, WDR [32] incorporates various spatiotemporal features and adopts a recurrent neural network to extract high-order feature interactions. Google Maps [4] leverages Graph Neural Networks (GNN) [14] to model spatial dependencies of the road network, which has been deployed to serve more than 20 major cities. As another example, ConST-GAT [6] proposes a 3D-graph attention mechanism to jointly capture spatial and temporal dependencies for travel time prediction. However, most existing approaches mainly focus on improving the single-label prediction accuracy (*i.e.*, the actual travel time), but overlook the uncertainty of travel time induced by various dynamic contextual factors (*e.g.*, congestion induced by unexpected incidents, ad hoc traffic control). As depicted in Figure 1, the travel time follows a long-tail distribution. In fact, the travel time of each route may significantly diverge under different traffic conditions. Seamlessly predicting the single-label travel time without considering such probabilistic nature may lead to biased predictions, which subsequently reduces the user's satisfaction as well as the overall

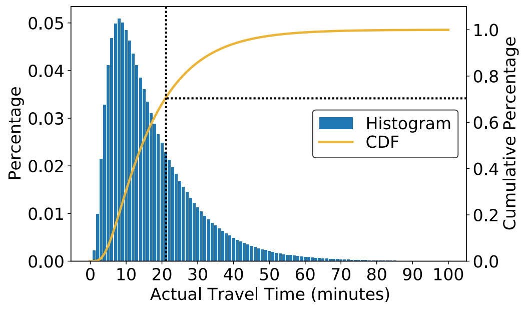


Figure 1: The long-tail travel time distribution of real-world historical trips in Beijing from May 2021 to June 2021 on DiDi's ride-hailing platform.

platform operating revenues. To this end, in this study, we investigate the uncertainty-aware probabilistic travel time prediction to improve various TTE-empowered ride-hailing tasks.

However, it is a nontrivial problem due to the following challenges. (1) *Unobservable travel uncertainty.* The travel time of a given route highly depends on its future travel context (*e.g.*, congestion, weather condition, unexpected traffic events). Despite existing approaches integrating various real-time contextual features to improve TTE accuracy, the future travel context is still unknown. One straightforward approach may be leveraging extra prediction models to infer future travel dynamics. However, such an approach may introduce extra noise and lead to severe error accumulation. Furthermore, the actual travel time of a certain trip collected by on-vehicle sensors (*e.g.*, GPS) is a scalar label. It is challenging to explicitly model the stochastic travel time distribution under the single-label supervision signal. (2) *Route-varying distributions.* In practice, the implicit travel time distribution of each route may be different. For example, the travel time on a highway in the suburb is usually much more stable than on an arterial road in the downtown area, not only because of the smooth traffic flow but also simpler travel context (e.g., fewer junctions and traffic lights). How to properly model the route-wise travel time distribution to improve the prediction performance is the second challenge. (3) *Service utility.* As a fundamental service, TTE supports diverse downstream ride-hailing tasks, such as order-dispatching and route recommendation [21]. The probabilistic travel time distribution provides extra information of the trip for downstream tasks, which improves the utility of the TTE service. However, the existing TTE service at DiDi involves a complicated infrastructure pipeline and highly optimized neural architecture, and it is prohibitively expensive and impractical to develop the probabilistic TTE service from scratch. Therefore, how to cost-effectively enable probabilistic travel time prediction with the consideration of the current system architecture is another challenge.

To tackle the above challenges, we propose a probabilistic travel time prediction framework (ProbTTE), which consists of three major modules. First, we propose a *Distributional Travel Time Encoding* (DTTE) module to transform the travel time prediction problem from conventional regression to a multi-class classification task. By discretizing the single-label travel time to a set of inter-related classes with the consideration of long-tail travel time patterns in ride-hailing, the TTE model automatically approximates the probabilistic travel time distribution we aim to capture. Second, we devise a tailored *Adaptive Local Label Smoothing* (ALLS) module to preserve the ordinal inter-class relationship in view of the local proximity between ground truth and nearby classes. Third, by analyzing large-scale historical trip data from DiDi, we reveal the travel time of each route follows the log-normal Gaussian distribution. Therefore, we propose a *Route-Wise Prior Regularization* (RWPR) module to further enhance the probabilistic travel time prediction by incorporating the prior knowledge of each route. It is worth mentioning that ProbTTE requires no extra manipulation of the underlying neural network architecture and the data pipeline, which makes it easy to integrate with the current TTE service. Since late 2022, the proposed framework has been deployed in production at DiDi to support multiple downstream order-dispatching services and reduces millions of bad cases per day.

Our main contributions are summarized as follows. (1) To our knowledge, this is the first attempt to investigate the probabilistic travel time estimation problem for the ride-hailing platform. (2) We propose ProbTTE, a probabilistic framework to improve the accuracy and utility of the TTE service. ProbTTE is agnostic to the prediction model and can be easily integrated with existing TTE alternatives. (3) We deploy ProbTTE at DiDi's ride-hailing platform, which successfully improves the travel time prediction accuracy and order-dispatching quality. We share the deployment and integration experience to benefit the community. (4) Extensive experiments on the real-world TTE datasets and two order-dispatching services demonstrate the effectiveness of our proposed framework against five baselines and the previous order-dispatching strategy.

2 PRELIMINARIES

In this section, we first present the formal problem definition and then briefly introduce the current TTE service at DiDi.

2.1 Definitions and Problem Statement

This paper focuses on vehicle travel time on the road network for ride-hailing. Thus, we first define the road network as a directed weighted graph $\mathcal{G} = (V, E)$, where $v_i \in V$ denotes road intersections and $e_{ij} \in E$ is the road segments with the endpoint v_i and v_j. In this way, a route can be represented as a sequence of road segments $r = \{e_1, e_i, \dots, e_k\}$, where k is the total number of road segments in the route. With the above notations and concepts, we define the trip query and the problem we aim to address.

Definition 1. ***Trip query***. *A trip query is defined as a 4-tuple $q = (l_o, l_d, t, r)$, where l_o and l_d denote the origin and destination of the trip, t is the trip start time, r is the planned travel route of the trip.*

We use y_i to denote the actual travel time of the corresponding trip query q_i. Note on the ride-hailing platform, the origin l_o and the destination l_d are defined by geographical locations (*i.e.*, longtitude and latitude) and associated with specific Point-of-Interests (POIs).

Problem 1. *Given a trip query q, we aim to estimate the probabilistic travel time distribution $\hat{y}$ from the origin l_o to the destination l_d along the route r based on $\hat{y} = \mathcal{F}(q, \mathcal{G})$, where $\mathcal{G}$ denotes the road network and $\mathcal{F}(\cdot)$ is the mapping function we aim to learn based on a set of historical queries $Q = \{q_1, q_2, q_3, \dots\}$.*

2.2 The Travel Time Estimation Service at DiDi

As an essential service of ride-hailing, DiDi continuously upgrades the TTE framework in the past years. We briefly introduce the evolution of DiDi's TTE service.

2.2.1 Machine-learning based approach. As early as 2016, DiDi launched the first version of machine-learning based TTE service. In this version, we first extract various route-related features and trip contextual features. To be specific, we construct route features including the segment ID, segment attributes (*e.g.*, length, width, speed limitation, lanes, *etc.*), the POI distribution around the segment (*e.g.*, the number of nearby gas stations), and traffic features (*e.g.*, real-time traffic condition and various statistical traffic speeds) to quantify the complex real-time travel context. Besides, we construct trip contextual features including the origin-destination features (*e.g.*, distance, geohash [23]), temporal features (*e.g.*, time of day, day of week, and holidays), driver features (*e.g.*, driver profile and vehicle profile) and weather condition features. The rich feature set not only encodes the complex real-time travel context but also various external factors related to each trip. After that, a variant of Gradient Boosting Regressor Tree (GBRT) is devised to predict the single-value travel time. Formally, given a trip query q_i, the corresponding feature vector $\mathbf{x}_i$ and the ground truth travel time y_i, we sequentially learn a set of tree regressors $\mathcal{F}(\cdot) = \{f_1(\cdot), f_2(\cdot), \dots, f_k(\cdot)\}$ and generate the prediction by ensembling each regressor,

$$\hat{y}_i = \mathcal{F}(\mathbf{x}_i) = \sum_{j=1}^{k} f_j(\mathbf{x}_i), \tag{1}$$

where $\hat{y}_i$ is the estimated scalar travel time and $f_j(\cdot)$ is the regression tree. The training objective is to minimize the Mean Average Error (MAE) loss,

$$\mathcal{L}_{reg} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} |y_i - \hat{y}_i|, \tag{2}$$

where $|Q|$ is the number of historical trips for training. In a nutshell, GBRT captures the linear interactions among high-dimensional handcrafted features and is lightweight for online deployment.

2.2.2 Deep-learning based approach. Recently deep learning has shown great superiority on extracting high-order features, which leads to breakthroughs in various domains [11, 33]. In early 2018, DiDi successfully upgraded the TTE service base on deep neural networks. Specifically, DiDi proposes a Wide-Deep-Recurrent (WDR) neural network to capture high-order feature interactions and the sequential route dependencies from large-scale historical trip data. More in detail, the WDR consists of three modules, the wide network, the deep network, and the recurrent network. The wide and deep network is first proposed for recommender systems [3] to improve the model expressive power by encoding sparse features to low-dimensional embeddings. By jointly learning linear feature combinations and higher-order feature interactions, the wide and deep network improves the memorization and generalization ability of the travel time prediction model. In particular, the wide part is designed for data memorization,

$$\mathbf{z}_i^w = \mathbf{w}^\top \mathbf{x}_i + b, \tag{3}$$

where $\mathbf{w}$ is the learnable parameter and b is the bias. The deep part first converts sparse features to dense embedding vectors and then extracts high-order knowledge by stacking multiple fully connected neural networks, *a.k.a.* the Multi-Layer Perception (MLP),

$$\mathbf{z}_i^d = MLP(\mathbf{x}_i). \tag{4}$$

Besides modeling feature interactions, the recurrent network is further introduced the capture the sequential dependency among segments in each route, which is initially parameterized by a Long-Short Term Memory (LSTM) [16]. Recently, we upgraded the recurrent network to Transformer [29] to further enhance the capability of preserving long-term dependency along the segment sequence

$$\mathbf{z}_i^r = Transformer(\mathbf{x}_i). \tag{5}$$

The final output is a combination of three components,

$$\hat{y}_i = \mathbf{z}_i^w + \mathbf{z}_i^d + \mathbf{z}_i^r. \tag{6}$$

The objective is the same as GBRT as defined in Equation (2).

Despite the current model delivering effective TTE results, the model outputs deterministic predictions under the supervision of single-label travel time. Next, we introduce our proposed probabilistic travel time forecasting framework.

3 THE PROPOSED APPROACH

3.1 Overview

Figure 2 depicts the overall framework of ProbTTE, which consists of three major tasks, (1) transforming the single-label travel time prediction to multi-class classification, (2) preserving the ordinal relationship among travel time classes, and (3) incorporating route-wise prior knowledge to probabilistic travel time prediction. For the first task, we propose a *Distributional Travel Time Encoding* (DTTE) module to discretize the continuous real-valued travel time to a set of tailored discrete classes, which enables the model to learn the probabilistic distribution we intend to match. For the second task, we propose an *Adaptive Local Label Smoothing* (ALLS) module to automatically learn the inter-class relationship without requiring explicit distribution signals. For the third task, we further propose a *Route-Wise Prior Regularization* (RWPR) module to guide the model optimization direction by absorbing the distribution knowledge from large-scale historical trip data. In this work, we deploy ProbTTE based on the current TTE service as described in Section 2.2.

3.2 Distributional Travel Time Encoding

Existing TTE models [9, 17, 30] mainly focus on capturing high-order feature interactions and sophisticated spatiotemporal dependencies under the guidance of deterministic single-label, which limits the model's performance and utility. In this module, we first transform the original regression task to a multi-class classification task, where the goal is to predict the discrete travel time distribution as the general classification.

Specifically, we encode the real-valued deterministic travel time $y \in \mathbb{R}_{>0}$ into a one-hot vector, where each category corresponds to a small travel time interval. Instead of uniformly discretizing the travel time to equal-length time bins, we propose to split the label space with the consideration of travel time distributions. As

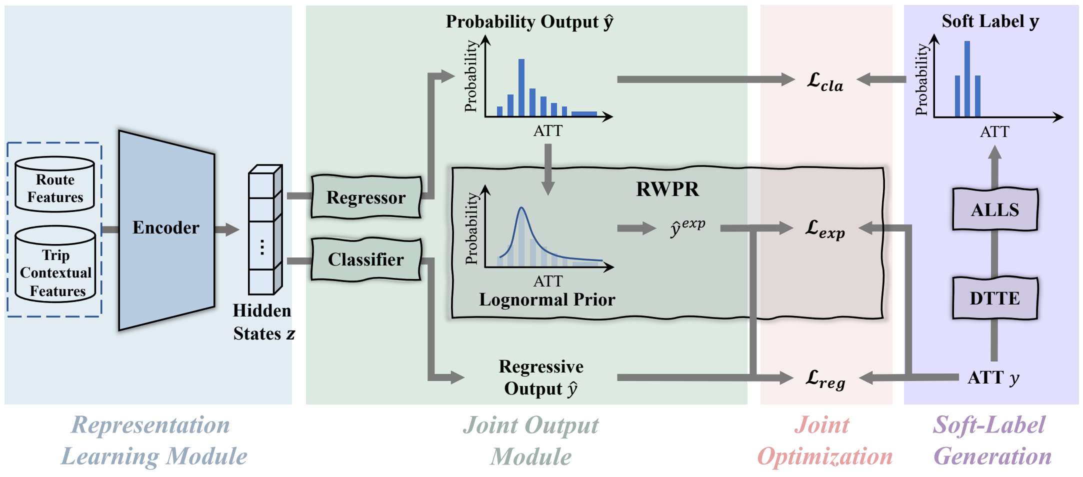


Figure 2: Overall framework of ProbTTE.

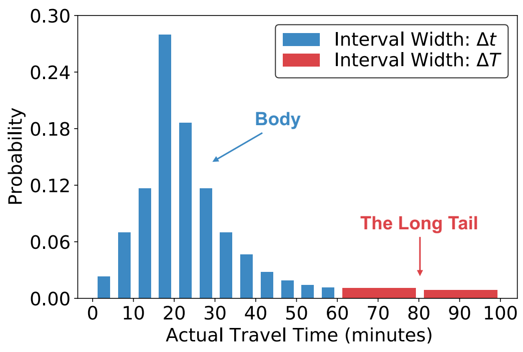


Figure 3: An illustrative example of travel time label discretization. DTTE encodes long-tail classes in more coarse-grained intervals, i.e., $\Delta t \ll \Delta T$.

depicted in Figure 1, the distribution of trip queries at DiDi shows a strong long-tail pattern. Intuitively, a fixed prediction error (*e.g.*, 10 seconds) is severer to user experience in short trips than a longer one. Evenly splitting the time interval may also let the model pay over-strengthened attention to minority classes. Therefore, as depicted in Figure 3, we define the boundaries of each travel time interval by considering the trip duration and density,

$$b_j = \begin{cases} j \cdot \Delta t, & \text{if } 0 \le j \le M \\ M \cdot \Delta t + j \cdot \Delta T, & \text{if } M+1 \le j \le N \\ +\infty, & \text{if } j = M+N+1 \end{cases}, \tag{7}$$

where Δt and ΔT are pre-defined window sizes of ordinary travel time interval and long-tailed travel time interval, respectively. M and N are hyperparameters controlling the boundary of ordinary and long-tail travel time intervals. Note that the aforementioned parameters can be empirically determined according to the empirical distribution of all the historical trips. In practice, we set $\Delta t \ll \Delta T$.

Given a trip query q and the ground truth y, we can derive the one-hot label $\mathbf{y} \in \mathbb{R}^{|C|}$ for query q, where C is the set of travel time classes, and the class $c_j \in C$ is set to one if $y \in [b_j, b_{j+1})$ and zero otherwise. Under the supervision of the one-hot label vector $\mathbf{y}$, we define the output layer for multi-class prediction,

$$\hat{\mathbf{y}} = \frac{e^{\mathbf{w}_i^\top \mathbf{x}}}{\sum_{j=1}^{|C|} e^{\mathbf{w}_j^\top \mathbf{x}}}, \tag{8}$$

where $\mathbf{w}_i$ are learnable parameters of the class i. The prediction model can be trained via the cross-entropy classification loss:

$$\mathcal{L}_{cla} = -\frac{1}{|Q|}\sum_{i=1}^{|Q|} \mathbf{y}_i \cdot \log(\hat{\mathbf{y}}_i). \tag{9}$$

By enforcing the model to mimic the one-hot label, the output $\hat{\mathbf{y}}$ represents the probability distribution of the original ground truth label we intend to match.

Briefly, the advantage of DTTE is twofold. (1) The multi-class classification objective preserves more information during model training, which improves the travel time prediction accuracy. (2) The distributional output provides additional information, such as the confidence of the prediction, which is useful in downstream ride-hailing tasks.

3.3 Adaptive Local Label Smoothing

Despite DTTE enabling the prediction model to output travel time distributions with the guidance of the actual travel time, recent study [25] reveals that training the neural network with one-hot labels tends to make the model over-confident on the targeted class. To alleviate the above problem, one straightforward approach may be adopting the Label Smoothing (LS) technique [25], which assigns soft label $\frac{\epsilon}{|C|-1}$ to each class except the correct one by reducing the ground truth probability from 1 to $1-\epsilon$. Here ϵ is a small constant. Such an approach has been widely applied to improve the independent multi-class classification problem in various computer vision and natural language tasks [10, 39]. However, simply assigning each class with equal importance overlooks the inter-class relationship in the multi-class travel time prediction problem. In fact, the discrete classes generated by DTTE follow a natural order. Predicting the travel time to a nearby class of the ground truth

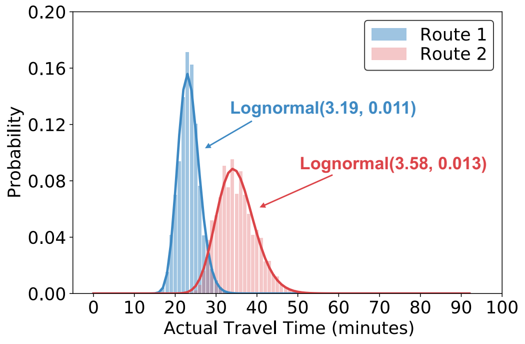


Figure 4: The probabilistic distribution of two different routes, which can be well fitted by two different log-normal prior distributions.

is more accurate than distant ones. Furthermore, as DTTE may generate hundreds of classes, assigning non-zero soft weights to classes distant from the ground truth may introduce extra bias for the model to converge. To address the above limitations, for a trip query q, we amend the $|C|$-dimensional one-hot vector $\mathbf{y}$ as below,

$$\mathbf{y}[j] = \begin{cases} p, & \text{if } |j-c| = 0, \\ \frac{1-p}{2\tau}, & \text{if } 1 \le |j-c| \le \tau, \\ 0, & \text{else,} \end{cases} \tag{10}$$

where $\mathbf{y}[j]$ denotes the j-th class of $\mathbf{y}$, c denotes the ground truth travel time class, p is a hyperparameter controlling the degree of label smoothing, τ is another hyperparameter controlling the width of local distribution support. Appendix A further explains the difference between common label smoothing and ALLS. The major modification of ALLS is that the soft label is only assigned to 2τ-nearest neighbor classes with the awareness of the ground truth class location. In particular, we set

$$\tau = \frac{y \cdot \alpha}{\Delta t}, \; p = \frac{\Delta t}{\Delta t + y \cdot \beta}, \tag{11}$$

where α and β are parameters of the prior travel time distribution empirically derived from the historical trip data. Moreover, to ensure the probability of the ground truth class larger than other classes, we constraint α and β satisfy $\frac{1-p}{2\tau} \le p$, which is equivalent to the following simple form:

$$\beta \le 2\alpha. \tag{12}$$

In such a way, the model is enforced to pay more attention to nearby classes around the ground truth with an adaptive time window.

3.4 Route-Wise Prior Regularization

As discussed before, each route's probabilistic travel time distribution may vary. Figure 4 illustrates the diverse distribution of real-world travel routes. In the previous study, it is difficult to quantify such route-wise distribution due to the data sparsity. However, as one of the world's largest ride-hailing platforms, DiDi has accumulated large-scale trip data, which provides us with a new opportunity to improve the probabilistic travel time prediction. To this end, we propose an additional regularization module to incorporate such route-wise prior knowledge.

In contrast to the general normal distribution, we introduce a parameterized log-normal distribution to fit the route-wise long-tail prior. Specifically, we first define the random variable Y_i to denote the travel time that follows the distribution $p_i(y)$ of trip query q_i,

$$Y_i \sim p_i(y) = \frac{1}{y\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(\log y - \mu_i)^2}{2\sigma_i^2}\right), \tag{13}$$

where μ_i and σ_i^2 denote the expectation and variance of $\log Y_i$, respectively. Let $\mathbf{p}_i \in \mathbb{R}^{|C|}$ denote the vector satisfying

$$\mathbf{p}_i[j] = \begin{cases} p_i(\frac{b_j+b_{j+1}}{2}) \cdot (b_{j+1} - b_j), & \text{if } 0 \le j \le |C|-2 \\ p_i(\frac{2b_j+\Delta T}{2}) \cdot \Delta T, & \text{if } j = |C|-1 \end{cases}, \tag{14}$$

where $\mathbf{p}_i[j]$ denotes the j-th element of $\mathbf{p}_i$. Intuitively, $\mathbf{p}_i$ is a discrete representation of the continuous distribution $p_i(y)$ defined in Equation (13). We minimize the cross-entropy between $\hat{\mathbf{y}}_i$ and $\mathbf{p}_i$ to obtain the lognormal distribution that best approximates the model's output distribution $\hat{\mathbf{y}}_i$ *i.e.*,

$$\min_{\mu_i,\sigma_i} \mathcal{L}_{prior} = -\hat{\mathbf{y}}_i \cdot \log(\mathbf{p}_i). \tag{15}$$

Equation (15) is a smooth convex problem and the optimum satisfies $\nabla\mathcal{L}_{prior} = \mathbf{0}$, solving which obtained the analytic solution

$$\mu_i = \hat{\mathbf{y}}_i \cdot \log(\mathbf{b}), \; \sigma_i^2 = \hat{\mathbf{y}}_i \cdot \Big(\log(\mathbf{b}) - \mu_i \cdot \mathbf{1}_{|C|}\Big)^2, \tag{16}$$

where $\mathbf{b} = \left(\frac{b_0+b_1}{2}, ..., \frac{b_{|C|-2}+b_{|C|-1}}{2}, \frac{2b_{|C|-1}+\Delta T}{2}\right)$ and $\mathbf{1}_{|C|}$ denote the all 1 vector of size $|C|$.

With the estimated log-normal distribution, we can directly obtain the most likely travel time and the expected travel time,

$$\hat{y}_i^{max} := \arg\max_y p_i(y) = e^{\mu_i - \sigma_i^2}, \tag{17}$$

$$\hat{y}_i^{exp} := \mathbb{E}[Y_i] = e^{\mu_i + \sigma_i^2/2} = \hat{y}_i^{max} \cdot e^{\frac{3}{2}\sigma_i^2}. \tag{18}$$

Both $\hat{y}_i^{max}$ and $\hat{y}_i^{exp}$ can be used as the prediction result. Note $\hat{y}_i^{exp}$ exhibits an ideal calibration by enlarging $\hat{y}_i^{max}$ with a factor $e^{\frac{3}{2}\sigma_i^2}$, which is more resistant to the travel time underestimate issue under extreme conditions (*e.g.*, heavy traffic jam). In practice, we choose the expectation $\hat{y}_i^{exp}$ to be one of the joint outputs of the probabilistic prediction model.

We introduce another objective to let the model self-adaptively balance between the most likely travel time and the calibrated one,

$$\mathcal{L}_{exp} = \frac{1}{|Q|}\sum_{i=1}^{|Q|} |\hat{y}_i^{exp} - y_i|. \tag{19}$$

3.5 Joint Optimization

To sum up, we train the ProbTTE framework in an end-to-end way by jointly optimizing the following objective,

$$\mathcal{L} = \mathcal{L}_{reg} + \lambda_1\mathcal{L}_{cla} + \lambda_2\mathcal{L}_{exp}, \tag{20}$$

where $\mathcal{L}_{reg}$ is the original regression loss defined in Equation (2), $\mathcal{L}_{cla}$ and $\mathcal{L}_{exp}$ are joint losses defined in Equation (9) and Equation (19). λ_1 and λ_2 are hyperparameters controlling the importance of the regularizers.

4 APPLICATIONS AND DEPLOYMENT

4.1 Applications in Ride-Hailing Service

We demonstrate two applications of ProbTTE, (1) the basic prediction of travel time, and (2) the assessment of trip distribution scores. The first application plays an important role in route planning and the second application provides an effective indicator for order-dispatching services.

4.1.1 Travel time prediction. As aforementioned, the ProbTTE framework is jointly optimized with the current TTE model described in Section 2.2. We further leverage the probabilistic output of the of ProbTTE to improve the TTE service. Specifically, based on the regression output $\hat{y}_i$ of Equation (6) and the expectation of the probabilistic output $\hat{y}_i^{exp}$ of Equation (19), we adjust the final predicted travel time by ensembling the estimations from different modules,

$$\hat{y}_i^f = \lambda \hat{y}_i + (1 - \lambda)\hat{y}_i^{exp}, \tag{21}$$

where λ is the hyperparameter controlling the importance of $\hat{y}_i$ and $\hat{y}_i^{exp}$.

4.1.2 Trip distribution score for order dispatching. Order dispatching directly decides the effectiveness of passenger-to-driver matching. The failure of dispatching seriously hurts the user experience and reduces the operating revenue. In real-world scenarios, most of the dispatching failure is induced by the extreme deviation of the predicted pickup time, which results in a longer waiting time than the passenger expected. Therefore, we define an indicator based on the probabilistic output of ProbTTE to help reduce bad cases in the order dispatching service. Please refer to Appendix B for the formal definition of the trip distribution score.

The distribution score reflects the uncertainty of the travel time predictions by leveraging the additional information encoded in the distributional output of ProbTTE. Intuitively, a more concentrated distribution with a higher maximal probability and better monotonicity on both sides of the maximal probability will have a larger distribution score, which indicates less travel time uncertainty. Thus, the distribution score can be directly adopted by the order-dispatching service to filter out potential drivers with higher pickup time uncertainty and prevent from dispatching orders with higher travel time uncertainty.

4.2 System Deployment

Since late 2022, the ProbTTE has been deployed at DiDi to support various ride-hailing services. In this subsection, we present the deployment and implementation details, including the model training pipeline, the asynchronous model update mechanism, and the online serving strategy.

4.2.1 Model training pipeline. In the production environment, the historical trip data is stored in the distributed file system (*i.e.*, HDFS). We periodically acquire and update the time-varying contextual features (*e.g.*, weather) through live APIs. For feature construction, we employ Spark [37], a distributed in-memory engine, to efficiently extract traffic and route-related features. Once the dataset is ready, we copy the training data to local GPU servers for model training. The model is trained on a powerful server with 4 Intel Xeon E5-2630 V4 CPUs, 45 GB memory, and one Nvidia Tesla P40 GPU. Note ProbTTE requires no additional data input or feature processing compared with the current TTE model, which means no additional development effort is required on the training pipeline.

4.2.2 Asynchronous model update. For model optimization, we implement the proposed framework using TensorFlow [1]. We normalize all numerical features via Z-score normalization and project all categorical features to 8-dimensional embeddings. We encode all ID features into 64 dimensions. We employ the LeakyReLU ($\alpha = 0.2$) as the activation function. We set the local window size $\alpha = 4.2$, the interval of travel time classes $\Delta t = 30$, the loss weights $\lambda_1 = 4 \times 10^4$ and $\lambda_2 = 1.0$. The learning rate is set to 0.0003, and the batch size is fixed to 512. Just like most internet services [20, 21], the machine learning model is required to handle large-scale and real-time streaming data. This requires periodical model updates (usually per day or per hour) to incorporate the critical knowledge in the latest data. However, periodically learning new models from scratch from the whole training data is prohibitively expensive [34, 36]. To preserve useful knowledge in the previous model and avoid retraining the model, we propose to incrementally update the TTE model in an asynchronous way. Recall the ProbTTE is extended based on the WDR model, we update the wide part and the deep-recurrent part separately. For the wide part, we follow FTRL [24], a regularization based learning strategy, to update the parameters. FTRL preserves useful knowledge and eases model optimization by enforcing the parameters to be sparse. For the deep and recurrent parts, we adopt the fine-tuning strategy to preserve knowledge encoded in the neural network. The fine-tuning strategy absorbs new knowledge from the latest training data through the incremental update of neural network parameters. Also note the ProbTTE does not request extra manipulation of the asynchronous model update, which is designated for the current TTE service. Once the model is incrementally updated, we push the well-trained model to the online environment to serve various ride-hailing services.

4.2.3 Online serving. The online service requires scalable and low-latency probabilistic inference to guarantee the user experience. To this end, we implement the online serving pipeline in C++. The feature extraction is optimized through several open-source vector libraries such as Eigen[1]. For the TTE service, we solely use the real-valued regression output and freeze the computation of other output heads at the inference time. For other downstream tasks, we only output the travel time distribution and send it to other services through the remote procedure call.

5 EXPERIMENTS

In this section, we conduct extensive experiments to evaluate our proposed framework on the basis of DiDi's current TTE service as introduced in Section 2.2. We report the performance gain by integrating ProbTTE into the existing ride-hailing services, including the vanilla travel time prediction and the downstream order-dispatching service. To be more specific, we aim to answer the following research questions. **RQ 1**: How does ProbTTE perform compared with the state-of-the-art TTE models? **RQ 2**: How do different modules in the framework affect the probabilistic distribution

[1]https://eigen.tuxfamily.org

Table 1: Statistics of two real-world datasets.

Data type	Beijing	Shanghai
Time span	2021.4.28-2021.6.22	
# of segments	2,292,963	2,317,394
# of training trip queries	28,465,664	28,900,352
# of testing trip queries	4,963,328	4,438,016

prediction? **RQ 3**: How robust is ProbTTE with respect to different hyper parameters? **RQ 4**: Can ProbTTE benefits downstream ride-hailing tasks in the online production environment?

5.1 Experimental Setup

5.1.1 Data description. We conduct extensive experiments on two metropolises, Beijing and Shanghai. Both datasets are collected from DiDi's ride-hailing platform, ranging from April 28, 2021 to June 22, 2021. For each dataset, we split data from April 28, 2021 to June 16, 2021 as the training set and the rest as the test set. In the ride-hailing platform, each trip query corresponds to a travel record. We also remove abnormal records such as travel time less than one minutes and travel speed over 120 Km/h. The average number of road segments in each route is 115 and 87, respectively. The average length of each route is 5.55 Km and 4.27 Km, respectively. The dynamic road features (*e.g.*, traffic speed) are changed every 5 minutes. For each record, the historical features (*e.g.*, average speed of the vehicle) are derived from the past seven weeks. The statistics of two datasets are reported in Table 1.

5.1.2 Metrics. We adopt five metrics for evaluation. For the travel time prediction task, we use Mean Average Error (MAE), Root Mean Squared Error (RMSE), Mean Average Percentage Error (MAPE) and Satisfaction Rate (SR) for evaluation [22, 40]. In particular, the SR is defined as the ratio of trips with MAPE lower than 10%, which reflects the user's experience by using the service [2]. For the order-dispatching task, we use the Bad Case Rate (BCR) for evaluation, which is defined as the ratio of dispatches where the actual travel time exceeds the estimated travel time by an unacceptable margin.

5.1.3 Baselines. We compare our proposed framework with the following state-of-the-arts as baselines, including a rule-based TTE strategy **RouteETA**, a tree-based prediction model **GBRT** [7], the current TTE model **WDR** [32] deployed at DiDi as described in Section 2.2, and a Graph Neural Network (GNN) based TTE model **HierETA** [2]. Please refer to Appendix C for more baseline details.

5.2 TTE Performance Comparison (RQ 1)

Table 2 reports the overall results of ProbTTE and all the compared baselines with respect to four evaluation metrics. Since the performance is very close in the offline and online environments, we only report the offline results on two datasets. We can make the following observations. First, the proposed framework outperforms all baselines in terms of all metrics on two datasets. Specifically, ProbTTE achieves (2.16%, 2.24%, 3.15%, 1.18%) improvements compared with the state-of-the-art baseline using MAE, RMSE, MAPE, and SR on Beijing. The improvement on Shanghai are (1.36%, 1.17%, 6.29%, 1%), respectively. Such improvement is significant in the highly-optimized industrial machine learning system. More importantly, it is worth mentioning that ProbTTE achieves consistent improvements compared with the current model WDR used in DiDi's TTE service, without modifying the original neural network architecture. Furthermore, GBRT outperforms RouteETA by a large margin but performs worse than deep learning models. This is because the GBRT memorized feature combinations in historical data, but can not capture high-order feature interactions and complex spatiotemporal dependencies. We also observe WDR outperforms the state-of-the-art HierETA in terms of multiple metrics, which is because WDR is highly optimized in the production environment. This also proves the importance of engineering optimization of machine learning systems for practical deployment. Finally, we observe all models achieve better MAE and RMSE but worse MAPE and SR in Shanghai than in Beijing. This is perhaps because of the longer average trip distance in Beijing but the more complicated travel context in Shanghai.

5.3 Effectiveness of Joint Probabilistic Optimization (RQ 2)

To validate the effectiveness of each module in ProbTTE, we conduct an ablation study on two datasets using four metrics. Specifically, we compare the following variants. (1) ProbTTE *-WoSR* removes the adaptive local label smoothing and route-wise prior regularization module. (2) ProbTTE *-WLS* replaces the locally adaptive label smoothing with the vanilla label smoothing scheme. (3) ProbTTE *-WoR* removes the route-wise prior regularization module. (4) ProbTTE *-WND* replaces the log-normal distribution with the conventional normal distribution.

As reported in Table 3, we make the following observations. First, all three modules contribute to the overall performance gain. We observe performance degradation by removing any one of the modules. This validates the effectiveness of the probabilistic joint optimization framework. Second, we observe a performance degradation by replacing the adaptive local label smoothing with the vanilla label smoothing scheme, which demonstrates the benefit of constraining soft labels to a local interval around the actual travel time. Besides, we also observe performance improvement by replacing the general normal distribution with the log-normal distribution, which validates the unique distribution pattern of ride-hailing trips and the necessity of route-wise prior regularization. Overall, the above results verify the effectiveness of the proposed joint optimization framework.

5.4 Study of Parameter Sensitivity (RQ 3)

Then we study the parameter sensitivity of our proposed approach. We report the class window size parameter α, the interval of travel time classes Δt, the weight of the classification loss λ_1 and the weight of the route-wise regularizer λ_2 on the two datasets using the MAE metric. The results on the other three metrics are similar.

First, we vary α from 0.0 to 8.4, as reported in Figure 5(a). Overall, the MAE variation trend on two datasets are similar. Our approach yields the best performance when $\alpha = 4.2$. We use 4.2 as the default setting in the evaluation.

Second, we vary Δt from 15 to 75, as illustrated in Figure 5(b). Overall, when increasing Δt, the MAE first decreases then increases.

Table 2: Overall performance of travel time prediction on Beijing and Shanghai datasets.

Model	Beijing				Shanghai			
	MAE (sec)↓	RMSE (sec)↓	MAPE↓	SR(%)↑	MAE (sec)↓	RMSE (sec)↓	MAPE↓	SR(%)↑
RouteETA	153.29	250.06	0.1512	43.07	137.52	203.88	0.1728	38.64
GBDT	140.23	230.70	0.1401	46.22	125.02	189.19	0.1557	42.13
WDR	113.52	191.92	0.1258	55.82	97.35	146.79	0.1284	52.16
HierETA	115.59	194.62	0.1146	55.14	98.92	151.27	0.1579	51.76
ProbTTE	**111.12**	**187.01**	**0.1111**	**56.48**	**96.04**	**145.09**	**0.1208**	**52.68**

Table 3: Ablation study on Beijing and Shanghai datasets.

Model variants	Beijing				Shanghai			
	MAE (sec)↓	RMSE (sec)↓	MAPE↓	SR(%)↑	MAE (sec)↓	RMSE (sec)↓	MAPE↓	SR(%)↑
ProbTTE -*WoSR*	112.46	190.67	0.1367	56.24	96.44	146.6	0.1453	52.49
ProbTTE -*WLS*	112.16	189.76	0.1372	56.19	97.07	147.64	0.1482	52.23
ProbTTE -*WoR*	112.42	190.56	0.1367	56.20	96.75	147.59	0.1461	52.40
ProbTTE -*WND*	111.68	189.03	0.1369	56.32	96.33	146.42	0.1461	52.55
ProbTTE	**111.12**	**187.01**	**0.1111**	**56.48**	**96.04**	**145.09**	**0.1208**	**52.68**

Our approach yields the best performance when $\Delta t = 30$. Intuitively, too small Δt may induce the model hard to converge and too large Δt leads to information loss.

Third, we vary λ_1 from 2 to 6. As depicted in Figure 5(c), our approach achieves the best performance when $\lambda_1 = 4$. Increasing or decreasing λ_1 may induce unwilling performance variation.

Fourth, we vary λ_2 from 0.8 to 1.2. As shown in Figure 5(d), we observe the MAE of ProbTTE first decreases then increases. The model achieves the best performance when $\lambda_2 = 1.0$. We use 1.0 as the default hyperparameter in our model.

Overall, adjusting the above hyperparameters induce performance variation in a reasonable range on two datasets, which demonstrates the robustness of our approach.

5.5 Utility on Order Dispatching (RQ 4)

Besides the experiments on travel time prediction, we further justify the utility of the proposed probabilistic prediction framework on downstream tasks. More specifically, we conduct online A/B tests on five cities from September 16, 2022 to September 29, 2022, involving 3 million order-dispatching requests in total. We compare the current order-dispatching strategy without distribution score (*WoDS*) and the optimized order-dispatching strategy incorporating distribution score (*WDS*) on two ride-hailing services deployed at DiDi, namely service I and service II. In the experiments, we randomly sample the ride-hailing request to different order-dispatching strategies for fairness concerns.

Table 4 reports the Bad Case Rate of the current strategy with and without using our distribution score. As can be seen, integrating the distribution score consistently reduces the BCR on two services (2.8% on service I and 9.16% on service II) with statistical significance. In the real-world scenario, such optimization directly improves the user experience and driver income for millions of orders per day by reducing dispatching failure. The above results demonstrate the utility of ProbTTE in improving downstream ride-hailing tasks. We additionally provide a case study to show the practicability of the output travel time distribution in alleviating underestimation issue in Appendix D.

Table 4: Improvement on order dispatching services.

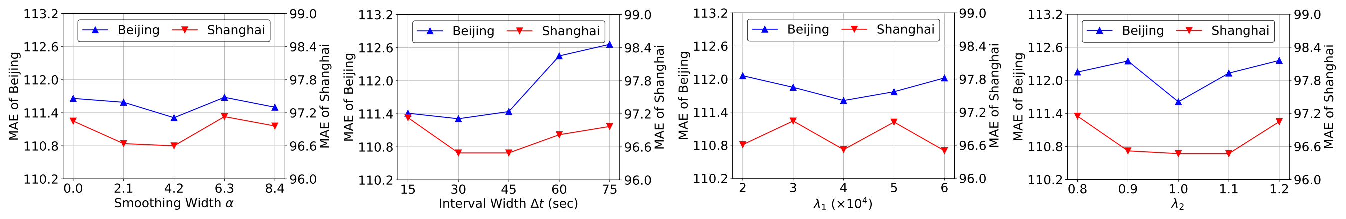
	Service I	Service II
WoDS	3.04%	4.08%
WDS	2.96%	3.7%
Improvement	2.8%	9.16%
P-value	0.03	0.01

6 RELATED WORK

6.1 Travel Time Estimation

Due to its importance to various location-based services (*e.g.*, ride-hailing [32], navigation [6], and food delivery [43]), TTE has been extensively studied in the literature. In this work, we focus on the TTE with the predefined travel route. Overall, existing machine learning based TTE models can be categorized into two classes, *i.e.*, the segment-based approach and the route-based approach.

For the segment-based approach [30, 31], the TTE service first estimates the travel time of each road segment independently and then aggregates the overall travel time by incorporating extra waiting time at road intersections. For instance, the GBRT-based TTE service previously deployed at DiDi manually extracts features for each road segment and then predicts the travel time based on the feature set of each segment. By leveraging the feature extraction capability of deep learning, eRCNN [30] learns correlations between nearby segments to improve the speed prediction accuracy. Recently, HetETA [17] leverages graph neural networks [14] to enhance the representation of road segments. As another example, CompactETA [9] further encodes spatiotemporal road network dependencies via a graph attention network and provides low-latency online TTE inference.



(a) Effect of α. (b) Effect of Δt. (c) Effect of λ_1. (d) Effect of λ_2.

Figure 5: Parameter sensitivity analysis on two datasets.

Route-based approaches focus on capturing road segment dependencies in each route to deliver end-to-end travel time prediction. One straightforward approach is to encode the whole route by using sequential models, *e.g.*, LSTM [16] and Bi-LSTM [38]. WDR[32] combines the sequential model and a deep&wide network to capture both the route dependencies and contextual information. ConSTGAT [6] integrates the road segment approach and route approach in the multi-task learning paradigm. However, existing TTE approaches are mainly trained under the supervision of a real-valued single label, which overlooks the probabilistic nature of travel time under various travel uncertainties.

6.2 Travel Time Distribution Estimation

Another line of research focuses on estimating the Travel Time Distribution (TTD) rather than scalar travel time. Related studies can be broadly categorized into statistical models, Machine Learning (ML) models, and Deep Learning (DL) models.

Statistical models identify distributions that best fit with the observed TTD. Guessous [12] discusses the fitness of six distributions on motorways. Li [18] explores the reliability of four distributions on various types of roads with varied traffic conditions. However, these studies rely on prior model assumptions, limiting flexibility. ML models such as variance-entropy-based clustering [35] and Bayesian networks [26] can capture TTDs in different situations without strict assumptions. However, they may fail to capture complex spatiotemporal dependencies in real-world traffic data.

Recently, a few efforts have been made in developing DL methods for modeling TTDs. DeepGTT [19] uses a deep generative model to estimate the TTD for a given route, while GCGTTE [28] employs generative adversarial networks for segment-wise TTD approximation. RTAG [42] integrates DeepGTT into a multi-task learning framework for route TTD estimation. These methods assume inverse Gaussian [19, 42] or Gaussian distributions [28] for travel time and estimate prior distribution parameters using deep learning models. However, they are still limited by distribution assumptions and might not fully exploit the capacity of DL models.

6.3 Probabilistic Forecasting

Probabilistic forecasting aims to predict the probability distribution of future quantities or events. Based on the model assumption, parametric and non-parametric are two major probability forecasting directions. In particular, the parametric approach estimates the distribution based on a prior of the distribution shape (*e.g.*, Gaussian and Gamma). For instance, Harvey [15] introduced a structural model for time series forecasting. DeepAR [27] estimates the probabilistic distribution for business process optimization via an auto-regressive recurrent neural network. Non-parametric approach, on the other hand, estimates the distribution without an explicit definition of the data distribution. For instance, quantile regression and ensemble are widely used for non-parametric forecasting [41]. Another line of research closely related to our work is ordinal regression [13], where the labels follow a natural order. The regression problem is usually transformed into a classification task to output the potential distribution via a Softmax classifier [5, 8]. In this work, we propose a non-parametric framework for probabilistic travel time prediction with the regularization of the parameterized route-wise prior distribution.

7 CONCLUSION

In this paper, we presented ProbTTE, a probabilistic prediction framework for uncertainty-aware travel time estimation. By explicitly quantifying and incorporating the travel time distribution into existing TTE services, the proposed framework improves the accuracy of travel time prediction and benefits various downstream ride-hailing tasks. Specifically, we transformed the single-label regression task into a multi-class classification task via a distributional travel time encoder. A adaptive local label smoothing module is then devised to alleviate the over-confidence problem by considering the inter-class ordinal relationship. Moreover, by leveraging the large-scale historical trip data, we proposed a route-wise prior regularizer to enforce the model absorbing prior distribution knowledge. Extensive experiments on real-world datasets demonstrated the superiority of the proposed framework. Since 2022, ProbTTE has been deployed and tested on DiDi's ride-hailing platforms. The probabilistic framework improves the travel time estimation service and benefits multiple TTE-empowered tasks in terms of accuracy and utility. It is also worth mentioning that the framework is model-agnostic, which means it can be easily integrated with other TTE models and extended to support general regression problems. In the future, we plan to apply ProbTTE to optimize more downstream services and improve other critical spatiotemporal prediction tasks.

ACKNOWLEDGMENTS

This research was supported in part by the National Natural Science Foundation of China under Grant No.62102110, Guangzhou Science and Technology Plan Guangzhou-HKUST(GZ) Joint Project No. 2023A03J0144, and Foshan HKUST Projects (FSUST21-FYTRI01A, FSUST21-FYTRI02A).

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A System for Large-Scale Machine Learning. In *12th USENIX Symposium on Operating Systems Design & Implementation*. 265–283.

[2] Zebin Chen, Xiaolin Xiao, Yue-Jiao Gong, Jun Fang, Nan Ma, Hua Chai, and Zhiguang Cao. 2022. Interpreting trajectories from multiple views: A hierarchical self-attention network for estimating the time of arrival. In *Proceedings of the 28th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2771–2779.

[3] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. 7–10.

[4] Austin Derrow-Pinion, Jennifer She, David Wong, Oliver Lange, Todd Hester, Luis Perez, Marc Nunkesser, Seongjae Lee, Xueying Guo, Brett Wiltshire, et al. 2021. Eta prediction with graph neural networks in google maps. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 3767–3776.

[5] Raul Diaz and Amit Marathe. 2019. Soft labels for ordinal regression. In *Proceedings of the 32nd IEEE/CVF Conference on Computer Vision & Pattern Recognition*. 4738–4747.

[6] Xiaomin Fang, Jizhou Huang, Fan Wang, Lingke Zeng, Haijin Liang, and Haifeng Wang. 2020. Constgat: Contextual spatial-temporal graph attention network for travel time estimation at baidu maps. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2697–2705.

[7] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[8] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. 2018. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the 31st IEEE/CVF Conference on Computer Vision & Pattern Recognition*. 2002–2011.

[9] Kun Fu, Fanlin Meng, Jieping Ye, and Zheng Wang. 2020. Compacteta: A fast inference system for travel time prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3337–3345.

[10] Yingbo Gao, Weiyue Wang, Christian Herold, Zijian Yang, and Hermann Ney. 2020. Towards a better understanding of label smoothing in neural machine translation. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics & the 10th International Joint Conference on Natural Language Processing*. 212–223.

[11] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press.

[12] Younes Guessous, Maurice Aron, Neila Bhouri, and Simon Cohen. 2014. Estimating travel time distribution under different traffic conditions. *Transportation Research Procedia* 3 (2014), 339–348.

[13] Tianchu Guo, Hui Zhang, ByungIn Yoo, Yongchao Liu, Youngjun Kwak, and Jae-Joon Han. 2021. Order regularization on ordinal loss for head pose, age and gaze estimation. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*. 1496–1504.

[14] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems 30*. 1024–1034.

[15] Andrew C Harvey. 1990. *Forecasting, structural time series models and the Kalman filter*. Cambridge University Press.

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[17] Huiting Hong, Yucheng Lin, Xiaoqing Yang, Zang Li, Kung Fu, Zheng Wang, Xiaohu Qie, and Jieping Ye. 2020. HetETA: Heterogeneous information network embedding for estimating time of arrival. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2444–2454.

[18] Ruimin Li, Huajun Chai, and Jin Tang. 2013. Empirical study of travel time estimation and reliability. *Mathematical Problems in Engineering* 2013 (2013).

[19] Xiucheng Li, Gao Cong, Aixin Sun, and Yun Cheng. 2019. Learning Travel Time Distributions with Deep Generative Model. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*. ACM, 1017–1027.

[20] Hao Liu, Ying Li, Yanjie Fu, Huaibo Mei, Jingbo Zhou, Xu Ma, and Hui Xiong. 2020. Polestar: An Intelligent, Efficient and National-Wide Public Transportation Routing Engine. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2321–2329.

[21] Hao Liu, Yongxin Tong, Panpan Zhang, Xinjiang Lu, Jianguo Duan, and Hui Xiong. 2019. Hydra: A Personalized and Context-Aware Multi-Modal Transportation Recommendation System. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2314–2324.

[22] Hao Liu, Qiyu Wu, Fuzhen Zhuang, Xinjiang Lu, Dejing Dou, and Hui Xiong. 2021. Community-Aware Multi-Task Transportation Demand Prediction. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*. 320–327.

[23] Jiajun Liu, Haoran Li, Yong Gao, Hao Yu, and Dan Jiang. 2014. A geohash-based index for spatial data management in distributed memory. In *2014 22nd International Conference on Geoinformatics*. 1–4.

[24] H Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad click prediction: a view from the trenches. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1222–1230.

[25] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? *Advances in Neural Information Processing Systems 32*.

[26] Anatolii Prokhorchuk, Justin Dauwels, and Patrick Jaillet. 2019. Estimating travel time distributions by Bayesian network inference. *IEEE Transactions on Intelligent Transportation Systems* 21, 5 (2019), 1867–1876.

[27] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191.

[28] Xiaozhuang Song, Chenhan Zhang, and James J. Q. Yu. 2021. Learn Travel Time Distribution with Graph Deep Learning and Generative Adversarial Network. In *24th IEEE International Intelligent Transportation Systems Conference*. IEEE, 1385–1390.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems 30*.

[30] Jingyuan Wang, Qian Gu, Junjie Wu, Guannan Liu, and Zhang Xiong. 2016. Traffic speed prediction and congestion source exploration: A deep learning method. In *2016 IEEE 16th International Conference on Data Mining*. 499–508.

[31] Yilun Wang, Yu Zheng, and Yexiang Xue. 2014. Travel time estimation of a path using sparse trajectories. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 25–34.

[32] Zheng Wang, Kun Fu, and Jieping Ye. 2018. Learning to estimate the travel time. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 858–866.

[33] Yang Yang, Yi-Feng Wu, De-Chuan Zhan, Zhi-Bin Liu, and Yuan Jiang. 2018. Complex Object Classification: A Multi-Modal Multi-Instance Multi-Label Deep Network with Optimal Transport. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2594–2603.

[34] Yang Yang, Da-Wei Zhou, De-Chuan Zhan, Hui Xiong, and Yuan Jiang. 2019. Adaptive Deep Models for Incremental Learning: Considering Capacity Scalability and Sustainability. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 74–82.

[35] Jing Yuan, Yu Zheng, Xing Xie, and Guangzhong Sun. 2011. T-drive: Enhancing driving directions with taxi drivers' intelligence. *IEEE Transactions on Knowledge and Data Engineering* 25, 1 (2011), 220–232.

[36] Zixuan Yuan, Hao Liu, Junming Liu, Yanchi Liu, Yang Yang, Renjun Hu, and Hui Xiong. 2021. Incremental Spatio-Temporal Graph Learning for Online Query-POI Matching. In *Proceedings of the Web Conference 2021*. 1586–1597.

[37] Matei Zaharia, Reynold S Xin, Patrick Wendell, Tathagata Das, Michael Armbrust, Ankur Dave, Xiangrui Meng, Josh Rosen, Shivaram Venkataraman, Michael J Franklin, et al. 2016. Apache spark: a unified engine for big data processing. *Commun. ACM* 59, 11 (2016), 56–65.

[38] Hanyuan Zhang, Hao Wu, Weiwei Sun, and Baihua Zheng. 2018. Deeptravel: A Neural Network Based Travel Time Estimation Model with Auxiliary Supervision. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. 3655–3661.

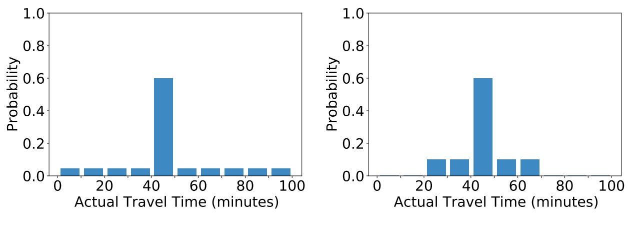
[39] Kaihua Zhang, Tengpeng Li, Bo Liu, and Qingshan Liu. 2019. Co-saliency detection via mask-guided fully convolutional networks with multi-scale label smoothing. In *Proceedings of the 32nd IEEE/CVF Conference on Computer Vision & Pattern Recognition*. 3095–3104.

[40] Weijia Zhang, Hao Liu, Lijun Zha, Hengshu Zhu, Ji Liu, Dejing Dou, and Hui Xiong. 2021. MugRep: A multi-task hierarchical graph representation learning framework for real estate appraisal. In *Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3937–3947.

[41] Yao Zhang, Jianxue Wang, and Xifan Wang. 2014. Review on probabilistic forecasting of wind power generation. *Renewable and Sustainable Energy Reviews* 32 (2014), 255–270.

[42] Wanyi Zhou, Xiaolin Xiao, Yue-Jiao Gong, Jia Chen, Jun Fang, Naiqiang Tan, Nan Ma, Qun Li, Hua Chai, Sang-Woon Jeon, and Jun Zhang. 2023. Travel Time Distribution Estimation by Learning Representations Over Temporal Attributed Graphs. *IEEE Transactions on Intelligent Transportation Systems* 24, 5 (2023), 5069–5081.

[43] Lin Zhu, Wei Yu, Kairong Zhou, Xing Wang, Wenxing Feng, Pengyu Wang, Ning Chen, and Pei Lee. 2020. Order fulfillment cycle time estimation for on-demand food delivery. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2571–2580.



(a) Soft label via LS (b) Soft label via ALLS

Figure 6: An illustrative example of adaptive local label smoothing. Compared with conventional label smoothing, ALLS assigns soft labels in a local window around the actual travel time class.

A DETAILS OF ADAPTIVE LOCAL LABEL SMOOTHING

The difference between ALLS and common soft label is illustrated in Figure 6.

B DEFINITION OF TRIP DISTRIBUTION SCORE FOR ORDER DISPATCHING

Definition 2. ***Distribution score****. Given a probability vector $\hat{\mathbf{y}} \in \mathbb{R}^{|C|}$ and the index of the maximal probability class m, the distribution score $s(\hat{\mathbf{y}})$ of $\hat{\mathbf{y}}$ is defined as*

$$s(\hat{\mathbf{y}}) = \sum_{0 \le i < j \le m} (\hat{\mathbf{y}}[j] - \hat{\mathbf{y}}[i])(j - i) + \sum_{m \le i < j \le |C|-1} (\hat{\mathbf{y}}[i] - \hat{\mathbf{y}}[j])(j - i).$$

C BASELINE DETAILS

We detail all the compared baselines as follows.

- **RouteETA** is a rule-based strategy for travel time prediction. Given a route, the RouteETA calculates the overall travel time by aggregating the historical average travel time of each road segment and the expected waiting time at each intersection. While the accuracy is unsatisfactory, the latency of RouteETA is extremely low.
- **GBRT** [7] is a tree-based prediction model, which is the initial machine learning based travel time prediction model deployed at DiDi. In this work, we use the XGBoost library to implement the model. In particular, the minimal child weight is set to 5, the maximum tree depth is set to 3, and the learning rate is fixed at 0.1.
- **WDR** [32] is the current TTE model used at DiDi as described in Section 2.2, where the recurrent part is a Transformer [29]. The deep network projects features into 20-dimensional vector via a three hidden layer MLP, where the size of each layer is 256. For the recurrent part, we set the hidden layer dimension, the embedding output, the number of attention layers, the number of attention heads to 128, 64, 2, and 8. We adopt LeakyReLU as the activation function with $\alpha = 0.2$.
- **HierETA** [2] is a Graph Neural Network (GNN) based travel time prediction model. HierETA devises a hierarchical attention network to capture local and global route patterns simultaneously. In particular, we set the number of parameters in all hidden recurrent layers to 128, the number of parameters in the attention layer to 256, and the embedding size to 8.

D EXTENDED CASE STUDY

We carried out a case study to validate the utility and effectiveness of ProbTTE's output travel time distribution in reducing the travel time underestimation. Specifically, given a TTE query q with actual travel time $y = 5245.00$ sec, we compared the travel time estimated by WDR and ProbTTE deployed at DiDi. The results showed that the regressive TTE of WDR is $\hat{y}_{WDR} = 4613.38$ sec, which underestimates the travel time by a large margin. Although the regressive TTE output by ProbTTE, $\hat{y}_{ProbTTE} = 4537.52$ sec, is quite close to $\hat{y}_{WDR}$, the expected TTE calculated by Equation (18), $\hat{y}^{exp}_{ProbTTE} = 5198.11$ sec, significantly mitigates the underestimation issue. In practice, adopting bagging introduced in Equation (21) will still show a prominent improvement. This case further demonstrates the model's ability to strike a balance between the estimated most likely travel time and the expected travel time. Excessive optimization of the loss function, $\mathcal{L}_{reg}$, in this extreme scenario could potentially impact the model's performance under normal conditions. Hence, the model is primarily trained to minimize $\mathcal{L}_{exp}$, placing greater emphasis on maintaining satisfactory performance in the majority of cases, while still producing a well-calibrated estimation, $\hat{y}^{exp}$, for particular extreme cases like this.