ILRoute: A Graph-based Imitation Learning Method to Unveil Riders' Routing Strategies in Food Delivery Service

Tao Feng
Department of Electronic Engineering, Tsinghua University
Beijing, China

Huan Yan
Department of Electronic Engineering, Tsinghua University
Beijing, China

Huandong Wang
Department of Electronic Engineering, Tsinghua University
Beijing, China

Wenzhen Huang
Department of Electronic Engineering, Tsinghua University
Beijing, China

Yuyang Han
Meituan
Beijing, China

Hongsen Liao
Meituan
Beijing, China

Jinghua Hao
Meituan
Beijing, China

Yong Li
Department of Electronic Engineering, Tsinghua University
Beijing, China

ABSTRACT

Pick-up and delivery (PD) services such as online food ordering are playing an increasingly important role in serving people's daily demands. Accurate PD route prediction (PDRP) is important for service providers to efficiently schedule riders to improve service quality. It is crucial to model the decision-making process behind the route choice of riders for PDRP. Recent years have witnessed the success of utilizing imitation learning (IL) to model user decision-making process. Therefore, we propose to deploy an IL framework to solve the PDRP problem. However, there still exist three main challenges: (1) the rider's route decision is affected by multi-source and heterogeneous features and the complex relationships among these features make it hard to explore how they influence the rider's route decision-making; (2) the large route decision-making space make it easy to explore and predict unreasonable routes; (3) the rider's personalized preference is important in modeling the route decision-making process but cannot be fully explored. To tackle the above challenges, we propose *ILRoute*, a Graph-based imitation learning method for PDRP. *ILRoute* utilizes a multi-graph neural network (multi-GNN) to extract the multi-source and heterogeneous features and model their complex relationships. To address the large route decision-making space, *ILRoute* introduces a mobility regularity-aware constraint as prior route choice knowledge to reduce the exploration route decision-making space. To model the personalized preferences of the rider, *ILRoute* utilizes a personalized constraint mechanism to enhance the personalization of the rider's route decision-making process. Offline experiments conducted on three real-world datasets and online comparisons demonstrate the superiority of our proposed model.

CCS CONCEPTS

• **Computing methodologies → Planning for deterministic actions**.

KEYWORDS

Pick-up and delivery, Decision-making process, Imitation learning, Mobility regularity-aware constraint, Personalized constraint mechanism

ACM Reference Format:
Tao Feng, Huan Yan, Huandong Wang, Wenzhen Huang, Yuyang Han, Hongsen Liao, Jinghua Hao, and Yong Li. 2023. ILRoute: A Graph-based Imitation Learning Method to Unveil Riders' Routing Strategies in Food Delivery Service. 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/10.1145/3580305.3599844

1 INTRODUCTION

With the rapid development of internet and e-commerce, many pick-up and delivery (PD) services such as online food ordering are playing an increasingly important role in our daily life. According to Meituan's financial report [1] at the end of 2022, more than 687 million users use Meituan food delivery platform [4], covering tens of thousands of counties and cities.

A typical food ordering, pick-up, and delivery process is shown in Figure 1. Customers will first place orders on the online food delivery platform. After receiving the order information, the platform will push the information to the corresponding restaurant, and meanwhile dispatch the order to the riders. After the rider confirms the order, he/she will first go to the restaurant to pick up the orders, and then finally deliver the orders to the customer. In the food ordering and delivery process, customers hope their

[1]https://meituan.todayir.com/html/ir_news.php

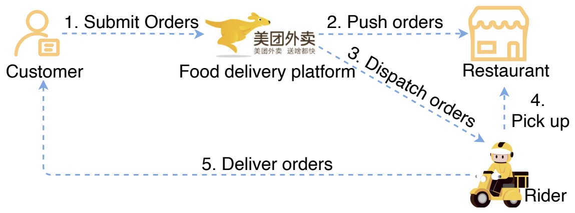


Figure 1: Food ordering and PD procedure.

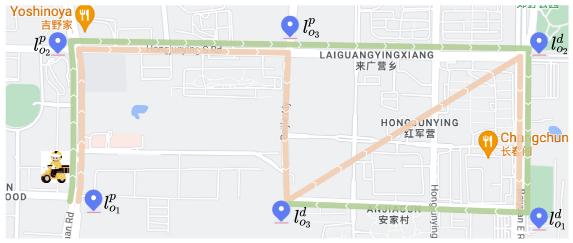


Figure 2: The decision-making process of a rider. The green line is the real PD route of rider and the orange line is the PD route predicted by a data-driven method. The $l^p_{o_i}$ and $l^p_{o_i}$ are the i-th pick-up node and delivery node respectively.

orders to be delivered on time, while riders usually want to choose the routes with minimum cost (total driving distance, driving time, etc.) to deliver the orders. To enhance customer satisfaction and rider delivery experience, many PD services providers [6, 15] model the riders' routing strategy and predict the delivery routes, so as to more reasonably dispatch the orders to riders. However, in reality, they find that the predicted rider routes are much different from the actual ones, leading to unreasonable order dispatch and route planing [27]. Therefore, it is of great importance to develop PD Route Prediction (PDRP) methods that can accurately unveil riders' routing strategies for route prediction.

PDRP has been studied in the academic community in recent years and can be mainly divided into two categories, i.e., rule-based methods and data-driven methods. Rule-based methods such as TimeRank and DisGreedy make route predictions based on distance or time. These methods do not consider the multi-source features and context information of the orders, thus failing to achieve good results. Data-driven methods like machine learning models [10] and deep learning models [6, 26, 27] design the neural network to capture the order features and context and predict the rider's future routes step by step. Although the data-driven models have achieved promising results, they ignore the decision-making process behind rider's routes. We take an example to illustrate it. As shown in Figure 2, we observe the rider's previous route decision will affect the subsequent route choice, because the rider can only deliver the order in the route after he/she has picked up the order in the restaurant. It can also be observed that a certain section of the route prediction deviation of the rider will affect the subsequent long-term route prediction, e.g., the data-driven method predicts the wrong route from $l^p_{o_3}$ to $l^d_{o_2}$, resulting in the incorrect predictions for the following routes.

Imitation learning (IL) is a powerful technology in modeling user decision-making processes. Therefore, in our paper, we aim to deploy an IL framework to solve the PDRP problem, which contains a reinforcement learning-based generator to model the route choice of the rider as a decision-making process and a discriminator to distinguish the generated routes of the generator. However, there still exist three main challenges in deploying IL.

- **Complex features influencing rider route decision**. The rider's route decision is affected by multi-source and heterogeneous features, such as the features of the order, and the context of the environment. Due to the complex relationship between these features, it is difficult to explore how they influence the rider's route decision-making.
- **Large route decision-making space**. Since a rider may have multiple orders with unfinished tasks at the same time, his/her route decision-making space grows exponentially with the number of route nodes. Meanwhile, there exist complex relationships between different tasks, thus it is easy to explore and predict unreasonable routes. How to address such a large route decision-making space to accurately infer the route is challenging.
- **Rider personalized preference**. The rider's personalized preference is important in modeling the route decision-making process. For example, a rider prefers to pick up the food for multiple orders due to the proximity in distance. However, existing deep learning-based methods often learn the common features of all the riders, ignoring the riders' personalized features in the decision-making process. How to learn the personalized preferences of riders is an open problem.

To tackle the above challenges, we propose *ILRoute*, a Graph-based imitation learning method for RDPR. Specifically, to address the first challenge, *ILRoute* proposes a multi-graph neural network (multi-GNN) to extract the multi-source and heterogeneous features. Multi-GNN contains a spatial GNN and a temporal GNN, which model the distance and order time relationships among orders respectively. In response to the second challenge, *ILRoute* introduces a mobility regularity-aware constraint to inspire the discriminator to distinguish the route sequences by considering the crucial spatial continuity patterns [7]. To solve the third challenge, *ILRoute* utilizes a personalized constraint mechanism to encourage the generator to generate the rider's routes with high mutual information with the rider's personalized features, thus enhancing the personalization of the rider's route decision-making process.

In summary, we make the following contributions in this study.

- This is the first work to investigate the PDRP task from the imitation learning perspective to the best of our knowledge. A graph-based imitation learning method *ILRoute* is proposed to solve the RDPR problem, which models the route choice of the rider as a decision-making process and unveils riders' routing strategies.
- We utilize a multi-GNN to extract the multi-source and heterogeneous features in the decision-making process of the rider, which supports the generator to generate reasonable routes. We also exploit a mobility regularity-aware constraint to reduce the rider's route decision-making space and a personalized constraint mechanism to enhance the

personalization in the route decision-making space, which improves the effectiveness and rationality of the routes generated by *ILRoute*.
- Offline experiments conducted on three real-world datasets from the Meituan delivery platform demonstrate that *ILRoute* significantly outperforms all the state-of-the-art baselines by reducing the metric of concordancy rate by over 7%. Moreover, compared with the online method in the Meituan food delivery platform, *ILRoute* improves the routes mean absolute error (MAE) metric by 4%.

2 PRELIMINARIES

2.1 Actor Critic (AC) Algorithm

The AC methods [13] leverage advantages from both value-based [16] and policy-based [23] methods. The AC methods include two estimators: a critic V_{π_θ} and an actor π_θ. The critic V_{π_θ} plays the role of the value-based method by estimating the value of the current state during training. It aims to minimize the TD δ_t error to precisely estimate the value of the current state:

$$\delta_t = (r(s_t, a_t) + V_{\pi_\theta}(s_{t+1}) - V_{\pi_\theta}(s_t))^2. \quad (1)$$

The actor π_θ plays the role of the policy-based method via interacting with the environment and generating actions according to the current policy. It utilizes an advantage function A_{π_θ} to make π_θ update more stable than policy gradient methods [28]:

$$A_{\pi_\theta}(s_t, a_t) = r(s_t, a_t) + V_{\pi_\theta}(s_{t+1}) - V_{\pi_\theta}(s_t). \quad (2)$$

Then the actor π_θ is updated through $J(\theta)$:

$$\nabla J(\theta) = E(\nabla_\theta log\pi_\theta(s_t, a_t)A_{\pi_\theta}(s_t, a_t)), \quad (3)$$

where E denotes the expectation value. In this work, we mainly combine our proposed *ILRoute* method with the AC algorithm to solve the PDRP problem.

2.2 Problem Formulation

Definition 1 (Order). *A set of orders $O = \{o_1, o_2, \cdots o_n\}$ is to be delivered by one rider $u \in U$ with his/her personalized features d_u. The orders can be divided into two categories. The orders in the first category have already been picked up from the restaurants by the rider and only have the delivery locations, which is denoted as $O_1 = \left\{o_1^1, o_2^1, \cdots o_n^1\right\}$. The second category represents the orders that have both pickup and delivery locations, which is denoted as $O_2 = \left\{o_1^2, o_2^2, \cdots o_n^2\right\}$. Therefore, we have $O = O_1 \cup O_2$ and $n = n_1 + n_2$. In addition, each order $o \in O$ is associated with order features E and context features V.*

Definition 2 (Pick-up and Delivery Route). *The set of pick-up and delivery locations are defined as $P = \left\{l_o^p | o \in O_1\right\}$ and $D = \left\{l_o^d | o \in O_2\right\}$ respectively. The pick-up and delivery route is the permutation of all the locations in $l_0 \cup P \cup D$, where l_0 denotes the rider's starting route node.*

Definition 3 (Pick-up and Delivery Route Prediction Problem). *Given a set of orders O, a set of riders U, the order features E, personalized features U_d , the context features V, and the label routes Tr, the goal of this problem is to train an effective route prediction model to predict the rider's future service routes.*

3 METHODOLOGY

3.1 System Overview

Figure 3 illustrates the proposed *ILRoute*, which is equipped with a graph-based route generator and a sequential discriminator. The graph-based route generator takes the rider features, route history, context features, and order features as input and converts them into the route choice of the rider. The sequential discriminator distinguishes the routes generated by the graph-based route generator and returns the reward to the generator to revise its policy.

3.2 Riders' Routing Choice as a Markov Decision Process

We transform the PDRP problem into a sequential decision-making problem, with the goal to maximize the accuracy of route prediction. Thus, we propose to formulate the problem using Markov Decision Process (MDP) [19] in an RL setting, which exists five parts:

- **Agent:** We consider the rider $u \in U$ as the RL agent, who observes the rider's personalized features d_u, a set of orders O, order features E, context features V, route history H, pick-up locations P and delivery locations D.
- **State:** Since the agent is the rider $u \in U$, we define state at time interval t as $s_t = (d_u, E_t, V_t, H_t)$, which contains the rider's personalized features, order features, context features and route history.
- **Action:** The action a_t at time interval t is selecting the next location that the rider will move to, which is also the output of the graph-based route generator and will be described in Sec 3.3.
- **Transition:** This component describes the process that the current state s_t will transit to the next state s_{t+1} after an action a_t is taken. Specifically, in our state setting, the rider's personalized features will remain unchanged. The order features, context features, and route history will change from s_t to s_{t+1} due to new route node choice and time changes.
- **Reward:** This reward function is to measure how similar the route generated by the graph-based route generator compared with the real rider's routes, which is the output of the sequential discriminator and will be discussed in Sec 3.4.

3.3 Graph-based Route Generator

As presented in Figure 3, the graph-based route generator G denoted as π_θ consists of two components: a multi-graph encoder and a pointer network-based route decoder. The multi-graph encoder is designed for extracting multi-source and heterogeneous features as spatio-temporal embedding and modeling the complex relationships among features influencing rider route decisions. The pointer network-based route decoder is to convert the spatio-temporal embedding into the action step by step.

3.3.1 Multi-Graph Encoder.

3.3.1 Multi-Graph Encoder. The multi-graph encoder first embeds the rider features d_u, route history H_t, context features V_t and order features E_t at time interval t. It then concatenates the embeddings

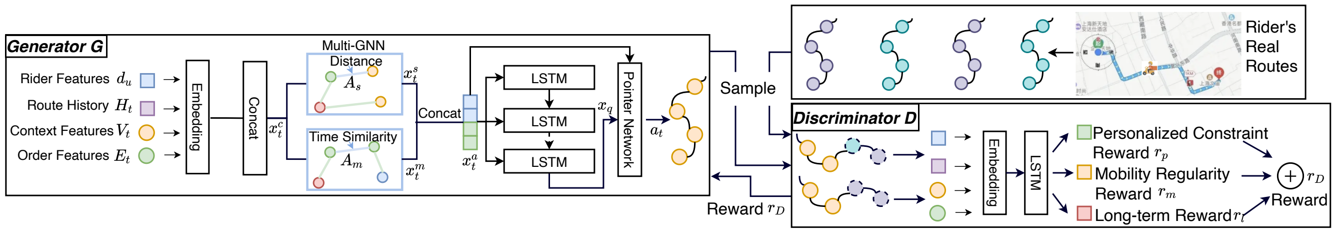


Figure 3: The framework of our system.

into a dense representation vector x^c, which can be derived by

$$x_t^c = Relu([d_u W_u, H_t W_h, V_t W_v, E_t W_e]), \tag{4}$$

where W_u, W_h, W_v, W_e are learnable parameters of embedding table, x_t^c is the vector representation of t-th time interval.

Inspired by GCN [12], multi-view graph representation [34] and GraphSAGE [8], we introduce a multi-GNN to extract the multi-source and heterogeneous features via a spatial-GNN and a temporal-GNN. The spatial-GNN models the spatial relationships among features through the distance between the order locations, denoted as A_s. And the temporal-GNN utilizes the due time of the orders to find the orders with similar time, so as to build the temporal relationship between the orders, denoted as A_m. What' more, the node feature in the two GNNs is vector representation x_t^c and the detailed layer-calculation of multi-GNN is as follows.

First, the spatial-GNN and the temporal-GNN convert x_t^c into two embeddings x_t^s and x_t^m with A_s and A_m,

$$x_t^s = \sigma(A_s x_t^c W_s + B_s), \tag{5}$$

$$x_t^m = \sigma(A_m x_t^c W_m + B_m), \tag{6}$$

where σ denotes the activation function, W_s, B_s, W_m and B_m are trainable parameters.

Then we concatenate them to obtain the hidden embedding x_t^a,

$$x_t^a = Concat(x_t^s, x_t^m). \tag{7}$$

3.3.2 Pointer Network-based Route Decoder. Based on the hidden embedding x_t^a extracted by multi-GNN, we first exploit a LSTM to convert the hidden embedding sequences $(x_1^a, x_2^a, \cdots, x_t^a)$ before time interval t into a hidden embedding vector x_q,

$$x_q = LSTM(x_1^a, x_2^a, \cdots, x_t^a). \tag{8}$$

Further, in order to obtain the route choice of the rider at each time interval t, we introduce a pointer network-based route decoder [24] to output the action. The detailed calculation of the decoder is as follows,

$$u_t = v^T tanh(W_q x_q + W_a x_t^a), \tag{9}$$

$$a_t = softmax(u_t) \tag{10}$$

where v^T, W_q and W_a are trainable parameters, $tanh$ and $softmax$ are activation functions.

3.4 Sequential Discriminator

The sequential discriminator is to distinguish the generation quality of the graph-based route generator compared with the real riders' routes. It also introduces a mobility regularity-aware constraint to

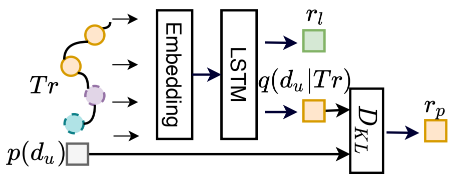


Figure 4: The personalized constraint mechanism in discriminator.

reduce the route choice exploration with prior spatial continuity knowledge and a personalized constraint mechanism to enhance the personalization of the rider's route decision-making process. As shown in Figure 3, the discriminator takes the whole route as input and utilizes an LSTM and a sigmoid function to convert the input into the long-term reward r_l, which is calculated as follows

$$r_l = sigmoid(LSTM(s_1, s_2, \cdots, s_T)), \tag{11}$$

where T is the length of the routes.

Mobility Regularity-Aware Constraint. While the standard discriminator generates learning signals for the generator by distinguishing the real and generated routes, it fails to capture the regularity and constraints of the rider route choice, which is the key point of high-quality route prediction. Rider's route choice shows a high degree of temporal and spatial regularity [7, 22], such as the significant probability to pick up or delivery the order in the nearby locations on the route sequence. Such mechanisms assist the discriminator to give a more effective signal, thus speeding up the generator to generate better rider routes. Specifically, we introduce a mobility regularity-aware constraint to add an auxiliary reward r_m, which assumes that riders will pick up or deliver the nearby orders first. The calculation of r_m is like this,

$$r_m = -\sum_{t=0}^{T-1} dis(l_t, l_{t+1}), \tag{12}$$

where l_t denotes the location of the order accessed by the rider at time interval t, dis denotes the Manhattan distance between two locations.

Personalized Constraint Mechanism. Personalized preference modeling is important in many fields such as traffic management [14] and courier order assignment [26]. Traditional discriminators often use a uniform model for all riders. This approach often models the rider's common route selection while ignoring the rider's personalization, which damages the prediction performance of the model. The InfoGAN [3] and InfoGAIL [3] that can generate personalized sequences by adding the mutual regulation between the condition variable c and the generated sequences. Inspired by this, we propose a personalized constraint mechanism to add the mutual regulation between our generated routes sequences Tr and the rider's personalized features d_u. To achieve this goal, we aim to maximize the mutual information $I(Tr; d_u)$, which can be calculated as follows:

$$I(Tr; d_u) = H(d_u) - H(d_u|Tr) = H(d_u) + E_{Tr}E_{d_u|Tr}logp(d_u|Tr), \tag{13}$$

where H denotes the entropy value, E denotes the expectation value and p denotes the probability.

Without access to the posterior $p(d_u|Tr)$, we cannot maximize the $I(Tr; d_u)$ directly. Here, we introduce $q(d_u|Tr)$ to approximate the true posterior $p(d_u|Tr)$,

$$logp(d_u|Tr) = logq(d_u|Tr) + log\frac{p(d_u|Tr)}{q(d_u|Tr)}. \tag{14}$$

Take equation (15) into equation (14), we can further observe that $E_{d_u|Tr}log\frac{p(d_u|Tr)}{q(d_u|Tr)}$ is always larger than 0 for denotes the KL divergence conditioned on Tr. Following previous work [3], we calculate the left part of equation (15) via the reparametrization trick, which can be expressed as follows:

$$I(Tr; d_u) \geq \int p(d_u)logq(d_u|Tr)dd_u + H(d_u) \equiv D_{KL}(p(d_u)||q(d_u|Tr)). \tag{15}$$

Therefore, we maximize $I(Tr; d_u)$ by maximizing $D_{KL}(p(d_u)$ $||q(d_u|Tr)$. Based on this, we add a personalized constraint reward $r_p = D_{KL}(p(d_u)||q(d_u|Tr)$ to enhance the personalization of the rider's route decision-making process. Therefore, for the discriminator, we can obtain its reward r_D,

$$r_D = r_l + \beta r_m + \gamma r_p, \tag{16}$$

where β and γ are hyperparameters.

Figure 4 shows the specific implementation process. The $q(d_u|Tr)$ is a head of the discriminator trained by maximum likelihood and the $p(d_u)$ is calculated from generated routes. Further, we denote the discriminator as D_ϕ, which is parameterized by ϕ and is optimized based on the following loss function

$$\mathcal{L}_D(\phi_n) = -\mathbb{E}_{\pi_{Tr}}[\log D_\phi(Tr)] - \mathbb{E}_{\pi_\theta}[\log(1-D_\phi(Tr)] - \mathbb{E}_{\pi_\theta}[\log q(d_u|Tr)], \tag{17}$$

where $\mathbb{E}_{\pi_\theta}$ represents the expectation with respect to the routes generated by generator π_θ. In addition, $\mathbb{E}_{\pi_{Tr}}$ represents the expectation with respect to the real riders' routes.

3.5 Model Training

We summarize details of the training process of *ILRoute* in Algorithm 1 of Appendix A.1. From the algorithm, we can first observe that a batch of generated route sequences and real-world route sequences are sampled to train the discriminator (lines 6-8). The generated sequences are regarded as negative samples and real-world sequences are regarded as positive samples to train discriminator via an Adam Optimizer [20] (line 8). Then, we calculate a batch of rewards for the generated routes (line 5). Finally, we train the generator by maximizing the expectation of reward via a reinforcement learning algorithm called actor-critic (AC) (line 12).

Table 1: Statistic of Datasets.

Datasets	*Beijing*	*Fuzhou*	*Guiyang*
Location	Beijing	Fuzhou	Guiyang
Time Span	08/15/2022-09/30/2022	09/15/2022-10/30/2022	10/15/2022-11/15/2022
Number of Riders	7000	60000	5000

Table 2: The features one sample contains.

Feature	Description
Driver info	Start location, Average speed, Punctuality
Context info	Period of a day, Day of a week, Index of city
Order info	Due time, Earliest pickup time, Pickup location, Drop-off duration, Delivery locations
Label	Actual delivery route

4 EVALUATION

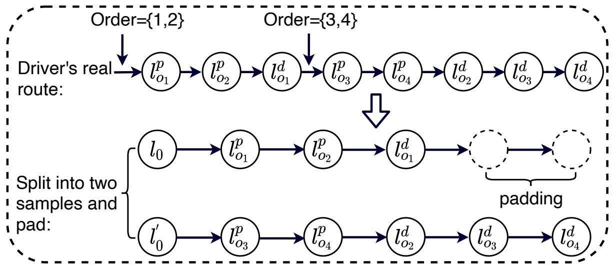

Order={1,2}
Order={3,4}
Driver's real route:
$l^p_{o_1}$ $l^p_{o_2}$ $l^d_{o_1}$ $l^p_{o_3}$ $l^p_{o_4}$ $l^d_{o_2}$ $l^d_{o_3}$ $l^d_{o_4}$
Split into two samples and pad:
l_0 $l^p_{o_1}$ $l^p_{o_2}$ $l^d_{o_1}$
padding
l'_0 $l^p_{o_3}$ $l^p_{o_4}$ $l^d_{o_2}$ $l^d_{o_3}$ $l^d_{o_4}$

Figure 5: An example to split the driver's real route into samples and pad.

In this section, we conduct both online and offline experiments to verify the effectiveness of *ILRoute* framework. We aim to answer the following Research Questions (RQs) through experimental studies.

- **RQ1:** Compared with the state-of-the-art route prediction techniques, can *ILRoute* achieve comparable results across different cities?
- **RQ2:** How is the performance of *ILRoute*'S variants with different combinations of its key components?
- **RQ3:** How do the hyper-parameter settings affect the performance of our *ILRoute*?
- **RQ4:** Compared with the online route prediction techniques in the Meituan food delivery platform, can *ILRoute* achieve comparable results?

4.1 Offline Experiments

4.1.1 Datasets.

Descriptions. To show the generality of our model, we conduct experiments on three real-world route datasets of three cities from the Meituan food delivery platform, whose descriptions are summarized in Table 1. The details of each dataset are as follows:

- **Beijing Dataset**. The dataset contains the pick-up and delivery records of 7000 riders from 45 days in Beijing, China. This dataset is to verify the generalization of *ILRoute* in developed cities.
- **Fuzhou Dataset**. The dataset contains the pick-up and delivery records of 5000 riders from 30 days in Fuzhou,

Table 3: Performance comparison over all baselines in five scenarios Bold denotes best results and underline denotes the second-best results.

City	**Beijing**						**Fuzhou**						**Guiyang**					
Method	DMAE	SR@1	SR@200	KRC	BR@500	BR@1000	DMAE	SR@1	SR@200	KRC	BR@500	BR@1000	DMAE	SR@1	SR@200	KRC	BR@500	BR@1000
TimeRank	1855.58	0.086	0.15	0.14	0.53	0.33	1698.45	0.098	0.16	0.13	0.53	0.30	1645.91	0.098	0.18	0.14	0.50	0.29
DisGreedy	1338.74	0.11	0.21	0.63	0.52	0.30	1206.56	0.12	0.22	0.64	0.49	0.25	1354.27	0.10	0.21	0.60	0.53	0.30
OSquare	575.45	0.58	0.68	0.84	0.15	0.099	563.54	0.59	0.71	0.85	0.13	0.068	512.35	0.62	0.73	0.87	0.12	0.062
DeepRoute	550.76	0.62	0.73	0.88	0.11	0.033	524.12	0.64	0.75	0.89	0.11	0.030	470.88	0.67	0.78	0.90	0.087	0.028
FDNet	505.34	0.64	0.75	0.89	0.13	0.066	496.42	0.65	0.76	0.89	0.14	0.058	450.28	0.68	0.79	0.91	0.103	0.065
Graph2Route	473.82	0.66	0.79	0.90	0.096	0.052	405.36	0.67	0.78	0.89	0.088	0.027	412.78	0.70	0.80	0.91	0.092	0.045
IRL	565.36	0.60	0.69	0.86	0.14	0.074	554.29	0.61	0.72	0.84	0.12	0.059	492.33	0.64	0.74	0.88	0.14	0.079
GAIL	500.13	0.65	0.76	0.89	0.11	0.059	468.96	0.66	0.76	0.89	0.095	0.047	438.48	0.69	0.79	0.89	0.101	0.059
ILRoute (Ours)	**404.63**	**0.70**	**0.82**	**0.92**	**0.059**	**0.015**	**346.68**	**0.71**	**0.82**	**0.92**	**0.053**	**0.011**	**357.95**	**0.73**	**0.83**	**0.93**	**0.054**	**0.014**

China. This dataset is to verify the effectiveness of *ILRoute* in quasi-developed cities.

- **Guiyang Dataset**. The dataset contains the pick-up and delivery records of 6000 riders from Sept. 15, 2022 to Oct. 15, 2022 (30 days) in Guiyang, China. This dataset aims to the generalization of *ILRoute* in underdeveloped cities.

Preprocessing. We first filter out outliers in the data, such as data points where the driver's driving speed is greater than a threshold. Then, since the driver may receive new orders during the delivery process which will affect his/her subsequent delivery choice, we split the delivery route of one driver into samples following previous work [27]. Taking Figure 5 as an example, the driver receives new orders o_3, o_4 after delivering order o_1, then he/she chooses to fetch order o_3 affected by the dispatching of new order and continues to deliver the other unfinished orders. In this situation, we split the driver's delivery route into two samples from the route node where the new orders arrive. The first sample takes $\{o_1, o_2\}$ as input and the corresponding label is $\left\{l_0, l^p_{o_1}, l^p_{o_2}, l^d_{o_1}\right\}$. The input of the second sample is $\{o_2, o_3, o_4\}$ and its label is $\left\{l'_0, l^p_{o_3}, l^p_{o_4}, l^d_{o_2}, l^d_{o_3}, l^d_{o_4}\right\}$. What's more, for the convenience of training, we also pad the samples to the same length. Note that the route splitting and padding data is utilized for training and evaluating in the offline environment. When we exploit *ILRoute* for prediction, it will generate the complete delivery route. In addition, we also summarize the features included in each sample shown in Table 2, which contain driver info, context info, order info, and label.

4.1.2 *Experimental Setup.*

Baseline Methods. To make a comprehensive comparison, eight baseline methods are adopted in performance evaluation.

First, we compare *ILRoute* with two rule-based baselines, including:

- **TimeRank [26].** It predicts the route by selecting the route node with the least remaining time each time.
- **DisGreedy [26].** It predicts the route by choosing the route node with the nearest distance each time.

Second, we adopt four route prediction methods based on machine learning or deep learning, including:

- **OSquare [37].** It is a node-wise ranking method that trains the prediction model based on classical machine learning methods like LightGBM [10]. It generates the whole route sequences step by step.
- **DeepRoute [27].** It is a deep learning-based model with a Transformer encoder and an attention-based decoder to select the next route node the rider should go. Note that DeepRoute itself cannot handle the pick-up then delivery route prediction problem. Therefore, we add the action mask mechanism to its decoder to guarantee that its generated route sequences meet the constraints of our datasets.
- **FDNet [6].** It is also a deep learning-based model utilizing an LSTM encoder and a Pointer Network decoder to determine the next route nodes of riders. In order to improve the route prediction performance, it introduces the arrival time prediction as the auxiliary task into its model.
- **Graph2Route [26].** It is the state-of-art method based on a dynamic spatial-temporal graph neural network, which is equipped with a graph-based encoder and a graph-based personalized route decoder.

At last, two representative imitation learning baselines are compared, including:

- **IRL [5].** It models the riders' route choice as a decision-making process and generates the route based on a reinforcement learning-based generator, which is trained by maximizing the entropy. Note that here it implements inverse reinforcement learning based on deep neural networks.
- **GAIL [9].** This model generates riders' route choices by modeling the sequence generation as a human decision-making process. In general, a GAIL model includes a discriminator module and a policy module. The policy is to generate the route sequences and the discriminator is to train the policy module by judging the quality of generated route sequences.

Evaluation Metrics. We compare the performance of different models based on the following six metrics of three categories.

First, we adopt the following metric to measure the route accuracy:

- **DMAE** [29]. It denotes the mean absolute error of the distance differences between generated routes and real routes. It measures how far the generated routes deviate from the real routes in terms of spatial distance.

Second, we choose the following three metrics to calculate the concordancy of the generated routes:

- **SR@k**. It represents the relaxed concordancy rate of generated routes compared with real routes. It first compares the route nodes of the generated routes and the real routes and calculates the distance between them. If the distance is less than k meters, the two route nodes are considered to be consistent. Then count the number of such consistent nodes in the routes and divide it by the length of the routes to obtain this metric. It is to relax the distance in consideration of the statistical error in the case of riders picking-up at the same store. In our experiments, we choose SR@1 and SR@200 as our metrics.

- **KRC [26]**. Kendall Rank Correlation (KRC) [11] is a statistical metric to measure the ordinal association between two sequences. In this paper, we utilize it to quantify the difference between two route ranks. Let $(g_1, \hat{o}_1, o_1)$, $(g_2, \hat{o}_2, o_2)$, $\cdots$, $(g_n, \hat{o}_n, o_n)$ be a set of observation sequences, i.e., g_i is the i-th unfinished pick-up or delivery node of the driver, $\hat{o}_i$ is the order of g_i in the predicted route and o_i is the order of g_i in the real route. For any pair of $(g_i, \hat{o_i}, o_i)$ and $(g_j, \hat{o}_j, o_j)$, if both $\hat{o}_i > \hat{o}_j$ and $o_i > o_j$ or both $\hat{o}_i < \hat{o}_j$ and $o_i < o_j$, the pair can be regarded as concordant. Otherwise, it is regarded as a discordant pair. Then the KRC is defined as:

$$KRC = \frac{n_c - n_d}{n_c + n_d}, \tag{18}$$

where n_c is the number of concordant pairs, and n_d is the number of discordant pairs.

Last, we utilize the following metric to measure how far the generated routes deviate from the real routes in terms of route concordancy:

- **BR@k**. It first calculates the distances between the generated route nodes and the real route nodes. Then we count the number of node pairs whose distance exceeds k meters and finally divide the number of these node pairs by the route length to get this metric. In our experiments, we choose BR@500 and BR@1000 as our metrics.

In summary, DMAE measures the distance similarity of the predicted route and the real route, while SR@k, KRC, and BR@k calculate their similarity from the perspective of route concordancy. Higher DMAE, SR@k, KRC, and lower BR@k mean better performance of the algorithm.

4.1.3 Evaluation Results (RQ1). We conduct rider's route prediction experiments in three cities with different development levels to verify the effectiveness and generalization ability of the *ILRoute* method. Each experimental result is the average value over 5 runs with different seeds. Table 3 summarizes the overall performance comparison, where *ILRoute* significantly outperforms all baseline methods. On Beijing dataset, it improves the performance of the best baseline by 19.2% at DMAE, by 7.7% at SR@1, and by 46.3% at BR@500.

Specifically, in comparison with two rule-based baselines, *ILRoute* takes into account the multi-source information of the rider as well as the context information into the generator, and thus achieves much better results. Compared to four route prediction methods based on machine learning or deep learning, *ILRoute* models the rider's decision-making process and considers the cumulative error of the rider's generated routes based on the reward function. Compared with two representative imitation learning baselines, *ILRoute* takes the rider personalization mechanism into the route generation model, so as to generate routes that are more in line with the real riders' behaviors. In the four route prediction methods based on machine learning or deep learning, Graph2Route performs best because it models the graph neural network into the decoder, which can greatly reduce the rider's route selection space and generate more reasonable routes. OSquare has the worst performance because it does not model the sequence information of the rider's choice well. In the two representative imitation learning baselines, GAIL performs much better than IRL because the restriction that the reward function is a linear relationship with rider's features limits the ability of IRL.

4.1.4 Ablation Study (RQ2). To provide a comprehensive understanding of the key components of *ILRoute*, we conduct a series of experiments to investigate the effect of different components.

- **w/o d-loss**: This view evaluates the effectiveness of adding mobility regularity-aware mechanism to constrain the strategy space of route prediction. It removes the mobility regularity-aware mechanism from the framework of *ILRoute*.
- **w/o c-loss**: It evaluates the effectiveness of utilizing personalized constraint to model the personalized features of riders in route prediction. It removes personalized constraint mechanism from the framework of *ILRoute*.
- **w/o multi-GNN**: This view aims to evaluates the effectiveness of extracting Rrider's multi-source features and contexts via multi-GNN. It removes muti-GNN from the framework of *ILRoute*.

We report the evaluation results on Beijing dataset in Figure 6. It can be observed that removing mobility regularity-aware mechanism will make *ILRoute* easy to explore many unreasonable routes, thus degrading the performance. Removing personalized constraint makes the generated rider route selection behavior loss of diversity„ which damages the performance of *ILRoute*. We can also observe that removing multi-GNN makes it difficult for *ILRoute* to extract Rrider's multi-source features and contexts, therefore achieving a worse performance. In a nutshell, each component of *ILRoute* improves the route prediction gradually and finally the full version achieves the best performance.

4.1.5 Hyper-parameters Study (RQ3). To investigate the robustness of our framework, we examine how key parameters affect the performance of *ILRoute* in the route prediction performance and report the results in Figure 7. The results is performed on SR@1 since it is the most important metric to evaluate the precision of generated routes. Figure 7(a) and 7(b) show the effect of the hyper-parameter β on two datasets. In *ILRoute*, β denotes the weight of r_m in equation (16). It is an important parameter, since it balances the personalization and commonality of the generated routes. As we can observe, in both two datasets, SR@1 first increases and then decreases when β increases from 0 to 1. To achieve the best performance, we choose $\beta = 0.4$ and $\beta = 0.6$ for Beijing and Fuzhou datasets respectively. Fig. 7(c) and 7(d) show the effect of γ, a weight of r_p in the equation (16). By adjusting this parameter, we can control the exploration scope of the generator to avoid generating

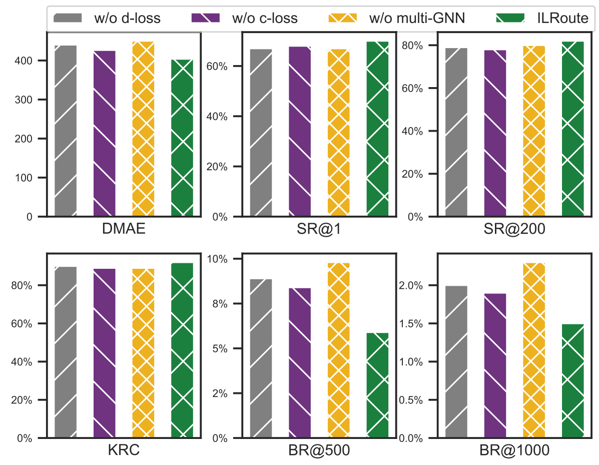
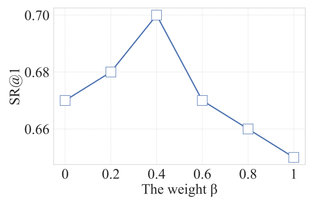
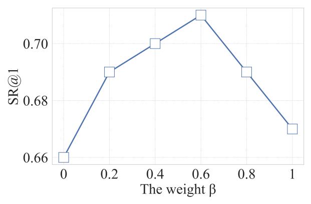


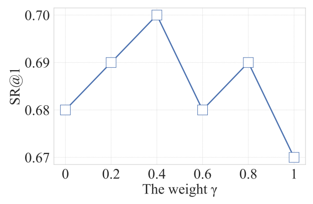
Figure 6: Ablation study of six metrics on Beijing dataset.



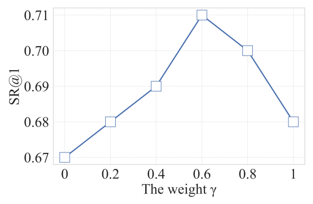
(a) Effect of hyper-parameter β on Beijing dataset.



(b) Effect of hyper-parameter β on Fuzhou dataset.



(c) Effect of hyper-parameter γ on Beijing dataset.



(d) Effect of hyper-parameter γ on Fuzhou dataset.

Figure 7: Effects of hyper-parameters β and γ to metric SR@1 on Beijing and Fuzhou datasets.

extreme routes. We determine $\gamma = 0.4$ and $\gamma = 0.6$ on two datasets respectively when maximum SR@1 is achieved.

4.2 Online Comparison (RQ4)

Motivated by the encouraging offline evaluation results, we carry out a lot of feature engineering and parameter experiments to compare *ILRoute* with the online methods applied in the Meituan food delivery platform. The experiment results show that *ILRoute* improves the DMAE by 4%. Since *ILRoute* aims to serve millions of riders and billions of customers, a small increase in the value of the metrics means a huge improvement of the model, and can generate enormous economic benefits.

4.3 Case Study (RQ1)

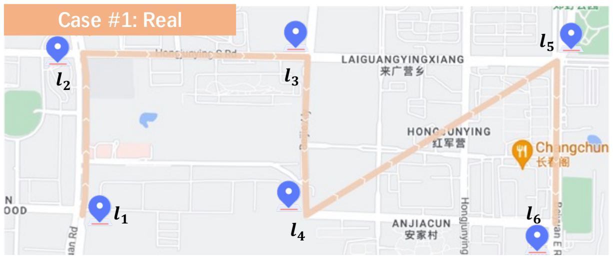
To analyze the performance of *ILRoute* more intuitively, we conduct an empirical case study shown in Figure 8. We analyze cases to illustrate the advantages of *ILRoute* over Graph2Route algorithm (best baseline), which can direct further improvements.

Case #1. Figure 8(a), 8(b) and 8(c) show three routes generated by the real case, *ILRoute* and Graph2Route respectively. In this case, we can observe that both *ILRoute* and Graph2Route predict the wrong route from l_3 to l_4 by comparing with the real route. However, *ILRoute* adjusts the deviation in time in the generation of subsequent route segments (from l_4 to l_6), so that the subsequent route segment predictions do not deviate too much from the real route segments. In contrast to this, Graph2Route is affected by the prediction deviation of previous route segments, resulting in increasingly large deviations in subsequently generated route segments. The reason for the above observation is that *ILRoute* models the rider's route selection via RL and models the cumulative error of route prediction based on the long-term reward function. Therefore, when there is a previous route prediction deviation, *ILRoute* will consider its cumulative impact and correct these errors in subsequent route generation.

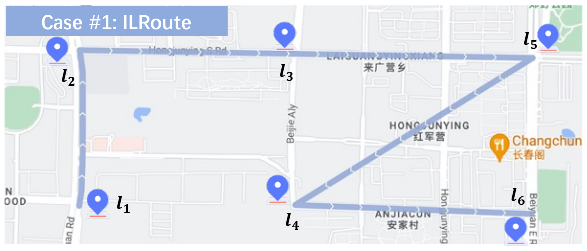
Case #2. Figure 8(d), 8(e) and 8(f) show three routes generated by the real case, *ILRoute* and Graph2Route respectively. In this case, it can be observed that the routes generated by *ILRoute* are the same as real routes while Graph2Route generates different routes. We can also observe that the routes generated by Graph2Route seem to be more convenient and reasonable and can save a lot of time for the rider. After checking the local traffic conditions, we found that the red dotted line area in Figure 8(d) is a location with a high incidence of accidents. What's more, by analyzing the personalized features of this rider, such as historical average speed and historical average punctuality, we can infer that this rider does not completely pursue efficiency but pays more attention to safety. The reason for the above observation is that *ILRoute* introduces the personalized constraint mechanism into the model and generates personalized routes for the rider. In contrast to this, Graph2Route does not specifically model the personalization of riders, therefore the generated routes are convenient and fast paths that most riders will take.

5 RELATED WORKS

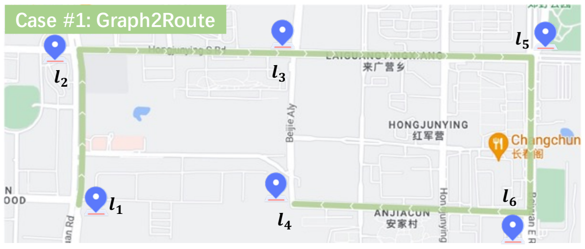
Imitation Learning. The goal of imitation learning is to learn the generator, which gives the action to be executed based on the current state [1, 2, 38, 39, 39]. The most success imitation learning method is the inverse reinforcement learning, which can be regarded as a special case of reinforcement learning with the unknown reward function to be learned from expert data [9, 39]. With the emergence of deep learning techniques, generative adversarial imitation learning (GAIL) is proposed to utilize the non-linear neural network to model the reward function and policy function to solve the imitation learning problem, which has been adopted in numerous practical applications, including dynamic treatment regimes [25], traffic signal control [31], and human drive behavior analysis [17, 18, 21, 30, 32, 33, 35, 36], etc. In this paper, we utilize imitation learning techniques to solve the PDRP problem, which models the rider's route choice as a decision-making process.



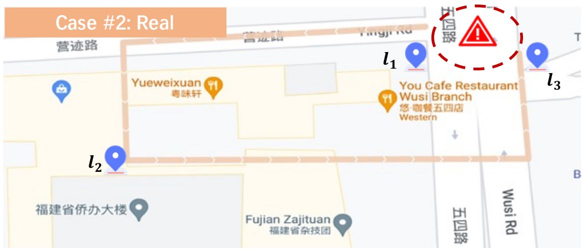
(a) The real route of rider in Case #1.



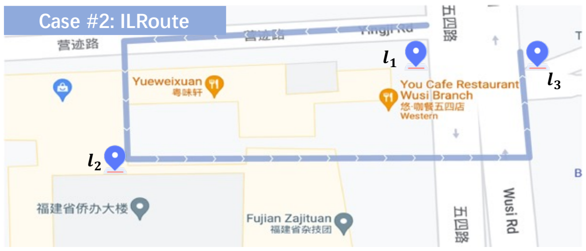
(b) The route generated by *ILRoute* in Case #1.



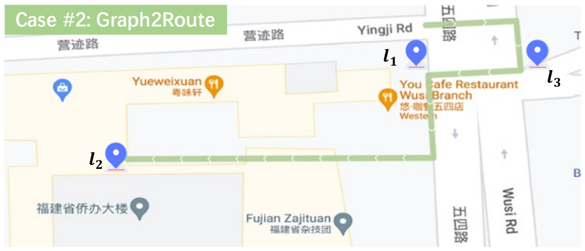
(c) The route generated by Graph2Route in Case #1.



(d) The real route of rider in Case #2. Note that the location of the red dotted line is the road section with a high incidence of accidents.



(e) The route generated by *ILRoute* in Case #2.



(f) The route generated by Graph2Route in Case #2.

Figure 8: Case Study. The orange lines, blue lines and green lines are real routes, routes generated by *ILRoute* and routes generated by Graph2Route respectively. l means the visit location of the rider.

Pick-up and Delivery Route Prediction. Pick-up and Delivery Route Prediction (PDRP) has been studied in the academic community in recent years and can be mainly divided into two categories, i.e., rule-based methods and data-driven methods. Rule-based methods such as TimeRank and DisGreedy make route predictions based on distance or time. These methods do not consider the multi-source features and context information of the orders, thus failing to achieve good results. Data-driven methods like machine learning models [10] and deep learning models [6, 26, 27] design the neural network to capture the order features and context and predict the rider's future routes step by step. Although the data-driven models have achieved promising results, they ignore the decision-making process behind rider's routes. Imitation learning (IL) is a powerful technology in modeling user decision-making processes. Therefore, in our paper, we aim to deploy an IL framework to solve the PDRP problem.

6 CONCLUSION

In this paper, we proposed to predict the rider's routes via a graph-based imitation learning method called *ILRoute*, which contains a reinforcement learning-based generator to model the route choice of the rider as a decision-making process and a discriminator to distinguish the generated routes of the generator. We first utilize a multi-GNN to extract the multi-source and heterogeneous features in the decision-making process of the rider. Then we exploit a mobility regularity-aware constraint to reduce the rider's route decision-making space and a personalized constraint mechanism to enhance the personalization in the route decision-making space. Offline experiments conducted on three real-world datasets and online comparison demonstrate the superiority of our proposed model.

ACKNOWLEDGEMENTS

This work was supported in part by The National Key Research and Development Program of China under grant 2022YFB3104702, the National Nature Science Foundation of China under 62272260, 62272262, 62171260, a grant from the Guoqiang Institute, Tsinghua University under 2021GOG1005, and China Postdoctoral Science Foundation under grant 2022M721833.

REFERENCES

[1] Michael Bain and Claude Sammut. 1995. A Framework for Behavioural Cloning.. In *Machine Intelligence 15*. 103–129.

[2] Abdeslam Boularias, Jens Kober, and Jan Peters. 2011. Relative entropy inverse reinforcement learning. In *Proc. AISTATS*. 182–189.

[3] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in neural information processing systems* 29 (2016).

[4] Xuetao Ding, Runfeng Zhang, Zhen Mao, Ke Xing, Fangxiao Du, Xingyu Liu, Guoxing Wei, Feifan Yin, Renqing He, and Zhizhao Sun. 2020. Delivery scope: A new way of restaurant retrieval for on-demand food delivery service. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3026–3034.

[5] Chelsea Finn, Sergey Levine, and Pieter Abbeel. 2016. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*. PMLR, 49–58.

[6] Chengliang Gao, Fan Zhang, Guanqun Wu, Qiwan Hu, Qiang Ru, Jinghua Hao, Renqing He, and Zhizhao Sun. 2021. A Deep Learning Method for Route and Time Prediction in Food Delivery Service. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2879–2889.

[7] M. C. Gonzalez, CA Hidalgo, and Albert Laszlo Barabasi. 2008. Understanding individual human mobility patterns. *Nature* 453, 7196 (2008), p.779–782.

[8] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in neural information processing systems*. 1024–1034.

[9] Jonathan Ho and Stefano Ermon. 2016. Generative adversarial imitation learning. In *Proc. NeurIPS*.

[10] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems* 30 (2017).

[11] Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika* 30, 1/2 (1938), 81–93.

[12] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[13] Vijay R Konda and John N Tsitsiklis. 2000. Actor-critic algorithms. In *Advances in neural information processing systems*. 1008–1014.

[14] Tong Li, Ahmad Alhilal, Anlan Zhang, Mohammad A Hoque, Dimitris Chatzopoulos, Zhu Xiao, Yong Li, and Pan Hui. 2019. Driving big data: A first look at driving behavior via a large-scale private car dataset. In *2019 IEEE 35th International Conference on Data Engineering Workshops (ICDEW)*. IEEE, 61–68.

[15] Yiyao Li and William Phillips. 2018. Learning from route plan deviation in last-mile delivery. (2018).

[16] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[17] Menghai Pan, Weixiao Huang, Yanhua Li, Xun Zhou, Zhenming Liu, Jie Bao, Yu Zheng, and Jun Luo. 2020. Is reinforcement learning the choice of human learners? a case study of taxi drivers. In *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*. 357–366.

[18] Menghai Pan, Weixiao Huang, Yanhua Li, Xun Zhou, and Jun Luo. 2020. xGAIL: Explainable Generative Adversarial Imitation Learning for Explainable Human Decision Analysis. In *Proc. KDD*.

[19] Martin L Puterman. 1990. Markov decision processes. *Handbooks in operations research and management science* 2 (1990), 331–434.

[20] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. 2019. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237* (2019).

[21] Hwasoo Yeo Seongjin Choi, Jiwon Kim. 2020. "TrajGAIL: Generating Urban Vehicle Trajectories using Generative Adversarial Imitation Learning". In *arXiv preprint*.

[22] Chaoming Song, Zehui Qu, Nicholas Blumm, and Albertlaszlo Barabasi. 2010. Limits of Predictability in Human Mobility. *Science* 327, 5968 (2010), 1018–1021.

[23] Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. 1999. Policy gradient methods for reinforcement learning with function approximation.. In *NIPs*, Vol. 99. Citeseer, 1057–1063.

[24] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. *Advances in neural information processing systems* 28 (2015).

[25] Lu Wang, Wenchao Yu, Xiaofeng He, Wei Cheng, and Hongyuan Zha. 2020. Adversarial Cooperative Imitation Learning for Dynamic Treatment Regimes. In *Proc. WWW*.

[26] Haomin Wen, Youfang Lin, Xiaowei Mao, Fan Wu, Yiji Zhao, Haochen Wang, Jianbin Zheng, Lixia Wu, Haoyuan Hu, and Huaiyu Wan. 2022. Graph2Route: A Dynamic Spatial-Temporal Graph Neural Network for Pick-up and Delivery Route Prediction. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4143–4152.

[27] Haomin Wen, Youfang Lin, Fan Wu, Huaiyu Wan, Shengnan Guo, Lixia Wu, Chao Song, and Yinghui Xu. 2021. Package pick-up route prediction via modeling couriers' spatial-temporal behaviors. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2141–2146.

[28] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3-4 (1992), 229–256.

[29] Cort J Willmott and Kenji Matsuura. 2005. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate research* 30, 1 (2005), 79–82.

[30] Guojun Wu, Yanhua Li, Shikai Luo, Ge Song, Qichao Wang, Jing He, Jieping Ye, Xiaohu Qie, and Hongtu Zhu. 2020. A Joint Inverse Reinforcement Learning and Deep Learning Model for Drivers' Behavioral Prediction. In *Proc. CIKM*.

[31] Yuanhao Xiong, Guanjie Zheng, Kai Xu, and Zhenhui Li. 2019. Learning Traffic Signal Control from Demonstrations. In *Proc. CIKM*.

[32] Yuan Yuan, Jingtao Ding, Huandong Wang, Depeng Jin, and Yong Li. 2022. Activity Trajectory Generation via Modeling Spatiotemporal Dynamics. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4752–4762.

[33] Yuan Yuan, Huandong Wang, Jingtao Ding, Depeng Jin, and Yong Li. 2023. Learning to Simulate Daily Activities via Modeling Dynamic Human Needs. In *Proceedings of the ACM Web Conference 2023*. 906–916.

[34] Mingyang Zhang, Tong Li, Yong Li, and Pan Hui. 2021. Multi-view joint graph representation learning for urban region embedding. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*. 4431–4437.

[35] Xin Zhang, Yanhua Li, Xun Zhou, and Jun Luo. 2019. Unveiling taxi drivers' strategies via cgail: Conditional generative adversarial imitation learning. In *Proc. ICDM*.

[36] Xin Zhang, Yanhua Li, Xun Zhou, Ziming Zhang, and Jun Luo. 2020. TrajGAIL: Trajectory Generative Adversarial Imitation Learning for Long-Term Decision Analysis. In *2020 IEEE International Conference on Data Mining (ICDM)*.

[37] Yan Zhang, Yunhuai Liu, Genjian Li, Yi Ding, Ning Chen, Hao Zhang, Tian He, and Desheng Zhang. 2019. Route prediction for instant delivery. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 3 (2019), 1–25.

[38] Brian D Ziebart, J Andrew Bagnell, and Anind K Dey. 2010. Modeling interaction via the principle of maximum causal entropy. In *ICML*.

[39] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. 2008. Maximum entropy inverse reinforcement learning.. In *Proc. AAAI*, Vol. 8. 1433–1438.

A APPENDIX

A.1 Algorithm

We summarize details of the training process of *ILRoute* in Algorithm 1. From the algorithm, we can first observe that a batch of generated route sequences and real-world route sequences are sampled to train the discriminator (lines 6-8). The generated sequences are regarded as negative samples and real-world sequences are regarded as positive samples to train discriminator via an Adam Optimizer [20] (line 8). Then, we calculate a batch of rewards for the generated routes (line 5). Finally, we train the generator by maximizing the expectation of reward via a reinforcement learning algorithm called actor-critic (AC) (line 12).

Algorithm 1: ILRoute

```
Input: Real-world rider's route sequences $T^r$.
1:  Initialize $\pi_\theta$, $D_\phi$ with random weights for generator and
    discriminator;
2:  for $i$=0,1,2... do
3:      Generate a batch of sequences $T_i^g \sim\pi_\theta$;
4:      for $k = 0, 1, 2...$ do
5:          Calculate a batch of rewards $r_D$ based on (17) for each
            state-action pair in sequences $T_i^g$;
6:          Sample generated sequences $T_{ki}^g$ based on $\pi_\theta$;
7:          Sample real-world sequences $T_k^r$ from $T^r$ with same batch
            size;
8:          Update $D_\phi$ based on (18) Adam Optimizer with the
            positive samples $T_k^r$ and negative samples $T_{ki}^g$.
9:      end for
10:     Update $\pi_\theta$ by maximizing the expectation of reward
        $\mathbb{E}_{\pi_\theta}(r_D)$ via the AC method;
11: end for
```