# RLMob: Deep Reinforcement Learning for Successive Mobility Prediction

Ziyan Luo, Congcong Miao*

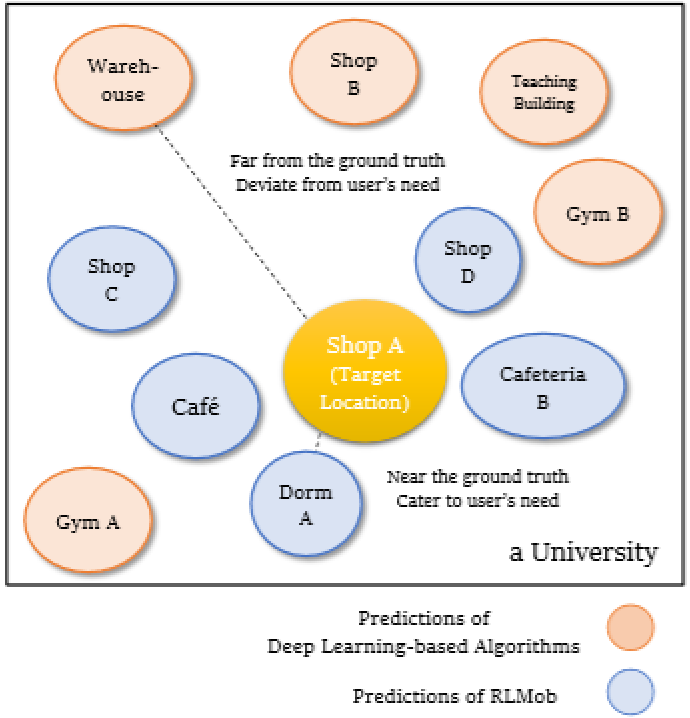
luo.ziyan@mila.quebec, mccmiao@163.com

*: Corresponding author

# Introduction

- Massive spatiotemporal trajectory data representing human mobility collected by miscellaneous devices
- *Successive mobility prediction* problem: predicting locations of the next few steps
- On university dataset and POI recommendation datasets

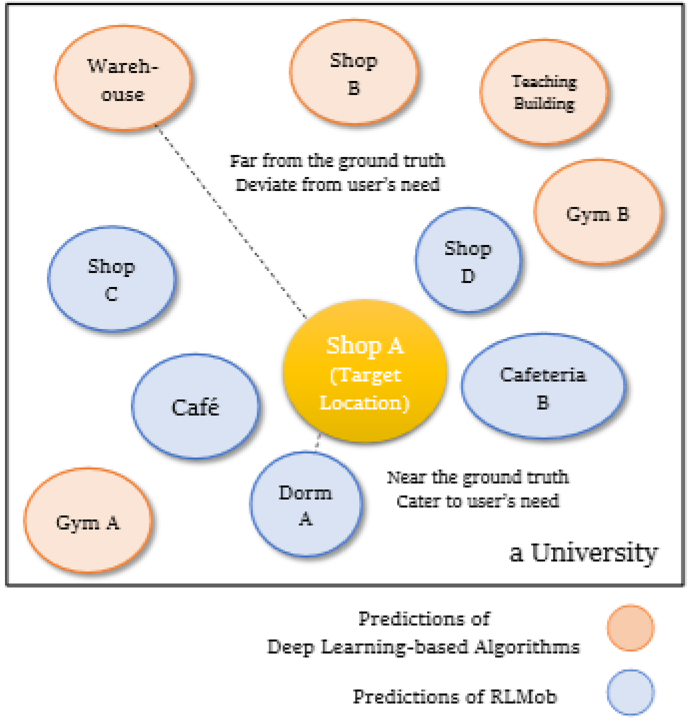
**When the algorithm predicted a wrong location**



# Introduction

- Four key challenges of existing supervised learning methods:
  - 1) disability to the circumstance that the optimizing target is non-differentiable
  - 2) difficulty to alter the recommendation strategy flexibly according to the changes in user needs
  - 3) error propagation and exposure bias issues when predicting multiple points
  - 4) cannot interactively explore user's potential interest that does not appear in the history

**When the algorithm predicted a wrong location**



# Problem Statement

- Trajectory Sequence: the aggregation of spatiotemporal points (a tuple of location point and timeslot)
- Trajectory: a sub sequence of trajectory sequence
- Problem (Successive Mobility Prediction): Given historical trajectory with length $m$ and the prediction timeslot set, the task of successive mobility prediction is to predict the next $n$ spatial points which is called the *target session*
- $m$, $n$ are variables (*historical trajectory*, *target session* are variable-length)

# MDP Formulation

An MDP, a tuple of (S,A,P,R,γ)

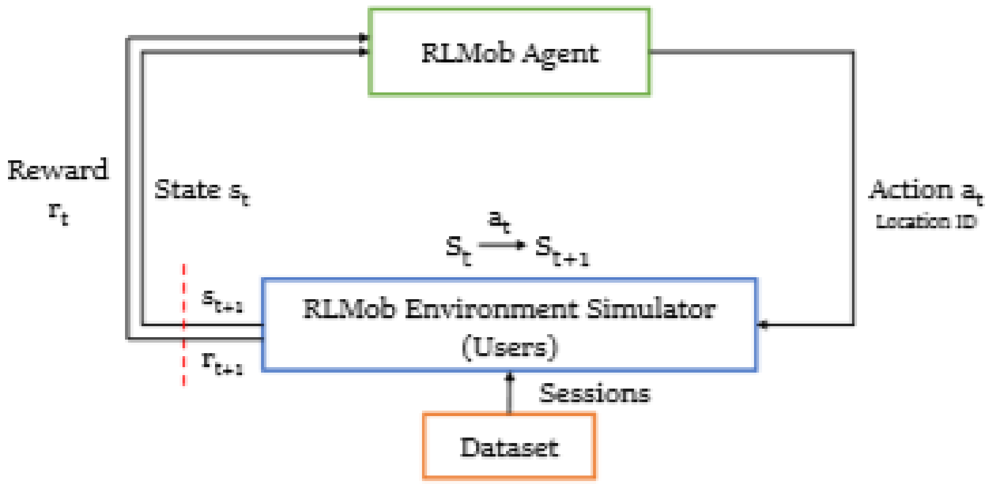
**State:** a fusion of user's characteristics, user historical trajectory, and the target timeslot

**Action (Discrete):** to recommend a location in the action space, the action space of the RLMob agent is all the available locations in the dataset

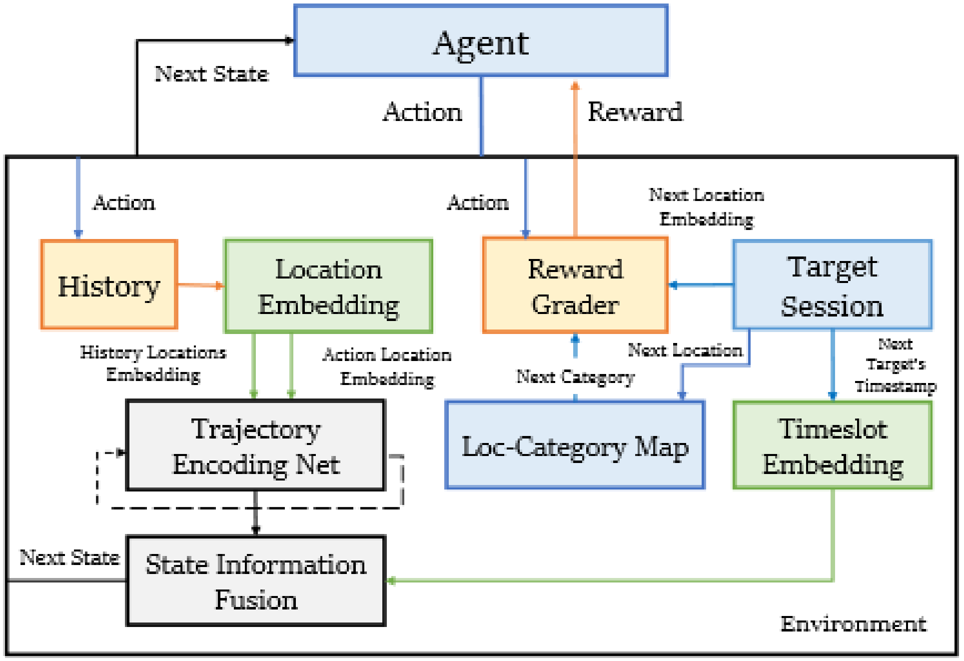
**Reward:** the RLMob environment simulator will give an immediate feedback *R(s,a)* to evaluate the action *a*

**Environment:** the RLMob environment simulator

The interactions between RLMob Agent and RL-Mob Environment in MDP



# Environment Simulator




- **Environment Init**: preprocessing, pretraining to get location and timeslot embedding (task: next location prediction)

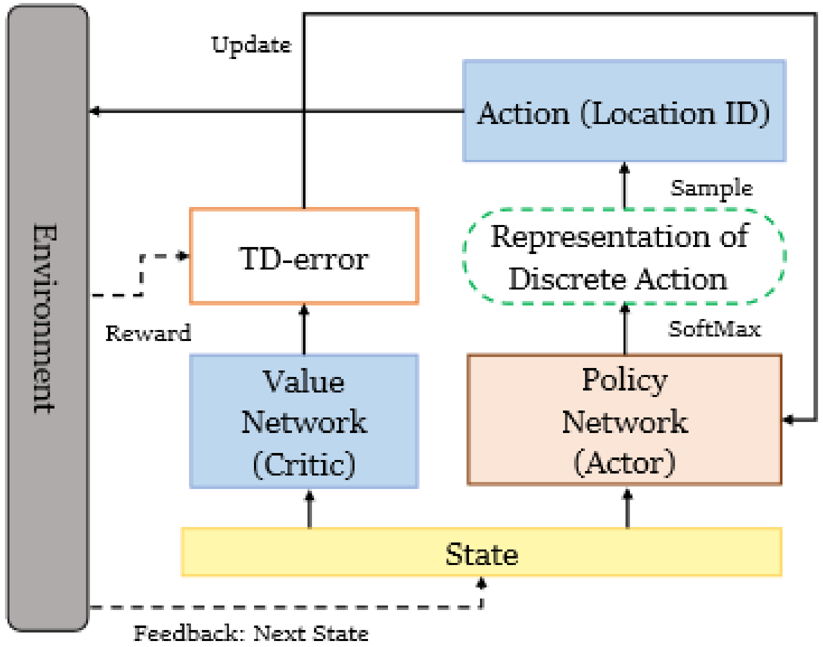
- **Reward Design (*R*)**:

$$r(s_t, a_t) = PS(s_t, a_t) + CS(s_t, a_t) + L2\text{-}Dist(s_t, a_t)$$

$$PS(s_t, a_t) = \begin{cases} k, & a_t = l^*(s_t) \\ 0, & a_t \neq l^*(s_t) \end{cases} \qquad CS(s_t, a_t) = \begin{cases} b, & C(a_t) = c^*(s_t) \\ 0, & C(a_t) \neq c^*(s_t) \end{cases}$$

$$L2\text{-}Dist(s_t, a_t) = \alpha \frac{\sqrt{\sum_{i=1}^{|D|} (e_L^i(a_t) - e_{l^*}^i(s_t))^2}}{|D|} \qquad R(s_t, a_t) = r(s_t, a_t) - r(s_t, a_t')$$

# Architecture of RLMob Agent




## Challenges

- The heterogeneity of user trajectory with unfixed length
- Diversified of preference and even some serendipity
- => Large variance of state encoding

## RLMob Agent:

- 1. Adopt the actor-critic architecture (shown in the figure), which empirically accelerates convergence
- 2. Proximal Policy Optimization (PPO) and GAE

$$J_{PPO-Clip}(\theta) = \sum_{(s_t,a_t)} \min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t,a_t),\right.$$
$$\left.\mathrm{clip}\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}, 1-\epsilon, 1+\epsilon\right) A^{\pi_{\theta_k}}(s_t,a_t)\right)$$

$$A(s_t, a_t) = \sum_{l=0}^{L} (\gamma\lambda)^l \delta^V_{t+l}$$

- 3. Networks pretraining: use the trained parameter of GRU4Rec

# Training and testing

**Algorithm 1** RLMob Training Stage

**input:** ◇ Initial parameters $\theta$, $\phi$
  **for** each episode **do**
    ▲ Initialize the environment with a random trajectory
    ▲ Make the initial state $s_0$

    **for** horizon $L$ **do**
      **for** each environment step **do**
        ◇ Sample action from the policy  $a_t \sim \pi_\theta(a_t|s_t)$
        ▲ Compute the reward $r_t$ (See Section 4.1.3)
        ▲ Combine $a_t$ to the user historic trajectory
        ▲ Make the next state $s_{t+1}$ (See Section 4.1.2)
        ◇ Store the transition  $(s_t, a_t, r_t, s_{t+1}, done)$
      **end for**

      **for** gradient step $K$ **do**
        ◇ Take out the transition(s)  $(s_t, a_t, r_t, s_{t+1}, done)$
        ◇ Update the critic parameters  $\phi \leftarrow \phi - \lambda_{V_\phi} \nabla_\phi J(\phi)$
        ◇ Update policy weights  $\theta \leftarrow \theta - \lambda_{\pi_\theta} \nabla_\theta J(\theta)$
      **end for**
    **end for**
  **end for**
**output:** ◇ Optimized parameters $\theta$, $\phi$

**Algorithm 2** RLMob Test Stage

  **for** each episode (item in test dataset) **do**
    ▲ Initialize the environment with the historic trajectory
    ▲ Make the initial state $s_0$

    **for** each environment step **do**
      ◇ Get action from the policy  $a_t = \arg\max_{a_t} \pi_\theta(a_t|s_t)$
      ▲ Compute the reward $r_t$ (See Section 4.1.3)
      ▲ Combine $a_t$ to the user historic trajectory
      ▲ Make the next state $s_{t+1}$ (See Section 4.1.2)
      ▲◇ Make statistics on $a_t$
    **end for**
  **end for**

After some episodes of training, the performance is tested,

and then the framework continues to alternate between the training state and the test stage

# Experiments

- Build a dataset based on the Wi-Fi data collected at a university and use two publicly available datasets to test our agent
- Filter and split them into training and test datasets
- Perform various comparison experiments to study the effectiveness of the purposed method

# Experiments

The performance of all comparison approaches

**Table 2: The performance of all comparison approaches. Improvement indicates the improvement of our method compared with GRU4Rec because GRU4Rec is the strongest baseline in general. This table only shows best results of all methods.**

| Dataset | Metric/Method | MC | MF | MLP | GRU4Rec | RLMob-REINFORCE | **RLMob-Proposed** | Improvement |
|---|---|---|---|---|---|---|---|---|
| Univ-WIFI | **Episode Final Return** | -8.315 | -20.93 | -0.6380 | 0.0000 | 0.2329 | 2.191 | - |
| | Acc@1 | 0.1350 | 0.0118 | 0.2079 | 0.2110 | 0.2127 | 0.2291 | 8.58% |
| | Macro-F1 | 0.0472 | 0.0093 | 0.2015 | 0.1565 | 0.1564 | 0.1664 | 5.95% |
| | CoCiN | 0.1093 | 0.2745 | 0.1489 | 0.1568 | 0.1590 | 0.1745 | 10.14% |
| | L2-Dist | 1.142 | 1.861 | 1.223 | 1.189 | 1.186 | 1.185 | 0.34% |
| F-TKY | **Episode Final Return** | -6.730 | -22.73 | -0.5780 | 0.0000 | 0.2229 | 2.397 | - |
| | Acc@1 | 0.2705 | 0.1842 | 0.3381 | 0.3931 | 0.3948 | 0.4150 | 5.57% |
| | Macro-F1 | 0.2551 | 0.1535 | 0.3266 | 0.3532 | 0.3449 | 0.3602 | 1.98% |
| | CoCiN | 0.3023 | 0.0000 | 0.2664 | 0.0031 | 0.0036 | 0.0036 | 16.13% |
| | L2-Dist | 0.7933 | 1.059 | 0.7714 | 0.7026 | 0.6939 | 0.6623 | 5.74% |
| F-NYK | **Episode Final Return** | -3.907 | -39.59 | -8.027 | 0.0000 | 0.0800 | 0.4402 | - |
| | Acc@1 | 0.3977 | 0.0755 | 0.3599 | 0.4362 | 0.4369 | 0.4401 | 0.73% |
| | Macro-F1 | 0.4266 | 0.0781 | 0.3853 | 0.4377 | 0.4407 | 0.4469 | 0.89% |
| | CoCiN | 0.0170 | 0.0299 | 0.0211 | 0.0036 | 0.0037 | 0.0040 | 7.5% |
| | L2-Dist | 1.250 | 1.944 | 1.334 | 1.185 | 1.185 | 1.177 | 0.68% |

- **CoCiN** stands for “Correction of Category in Negative results”
- Episode Final Return means the average sum of all rewards in an episode on the test data points

# Experiments

## The performance of all comparison approaches

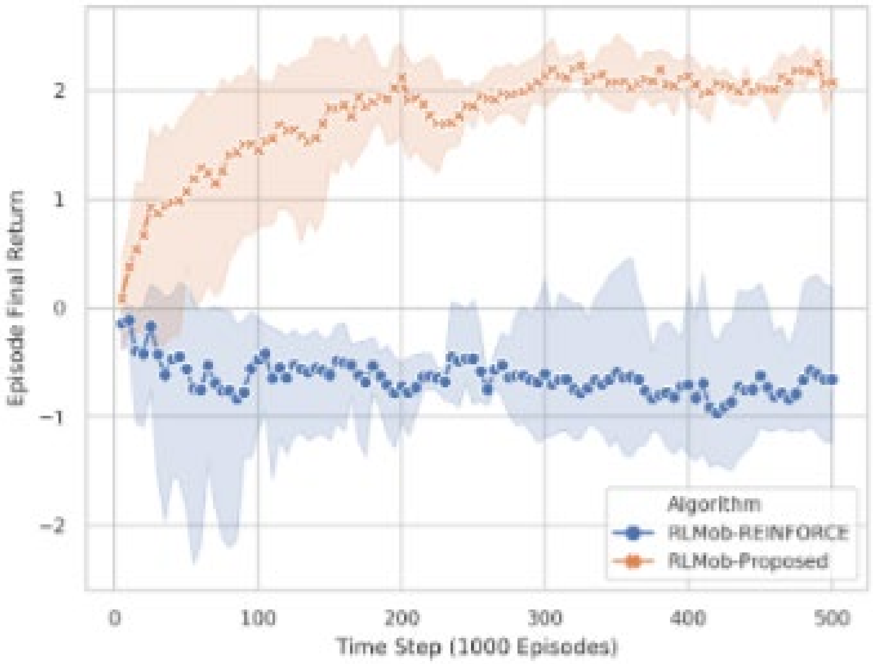
**Table 2: The performance of all comparison approaches. Improvement indicates the improvement of our method compared with GRU4Rec because GRU4Rec is the strongest baseline in general. This table only shows best results of all methods.**

| Dataset | Metric/Method | MC | MF | MLP | GRU4Rec | RLMob-REINFORCE | **RLMob-Proposed** | Improvement |
|---|---|---|---|---|---|---|---|---|
| Univ-WIFI | **Episode Final Return** | -8.315 | -20.93 | -0.6380 | 0.0000 | 0.2329 | 2.191 | - |
| | Acc@1 | 0.1350 | 0.0118 | 0.2079 | 0.2110 | 0.2127 | 0.2291 | 8.58% |
| | Macro-F1 | 0.0472 | 0.0093 | 0.2015 | 0.1565 | 0.1564 | 0.1664 | 5.95% |
| | CoCiN | 0.1093 | 0.2745 | 0.1489 | 0.1568 | 0.1590 | 0.1745 | 10.14% |
| | L2-Dist | 1.142 | 1.861 | 1.223 | 1.189 | 1.186 | 1.185 | 0.34% |
| F-TKY | **Episode Final Return** | -6.730 | -22.73 | -0.5780 | 0.0000 | 0.2229 | 2.397 | - |
| | Acc@1 | 0.2705 | 0.1842 | 0.3381 | 0.3931 | 0.3948 | 0.4150 | 5.57% |
| | Macro-F1 | 0.2551 | 0.1535 | 0.3266 | 0.3532 | 0.3449 | 0.3602 | 1.98% |
| | CoCiN | 0.3023 | 0.0000 | 0.2664 | 0.0031 | 0.0036 | 0.0036 | 16.13% |
| | L2-Dist | 0.7933 | 1.059 | 0.7714 | 0.7026 | 0.6939 | 0.6623 | 5.74% |
| F-NYK | **Episode Final Return** | -3.907 | -39.59 | -8.027 | 0.0000 | 0.0800 | 0.4402 | - |
| | Acc@1 | 0.3977 | 0.0755 | 0.3599 | 0.4362 | 0.4369 | 0.4401 | 0.73% |
| | Macro-F1 | 0.4266 | 0.0781 | 0.3853 | 0.4377 | 0.4407 | 0.4469 | 0.89% |
| | CoCiN | 0.0170 | 0.0299 | 0.0211 | 0.0036 | 0.0037 | 0.0040 | 7.5% |
| | L2-Dist | 1.250 | 1.944 | 1.334 | 1.185 | 1.185 | 1.177 | 0.68% |

- As expected, our method performs the best generally
- Results on *Univ-WIFI* dataset is better than Foursquare datasets, which may be owing to the sparse nature (the locations in a trajectory is sparse over time, making it “incomplete”)

# Experiments

Results on RL-based approaches



# Summary

- Attack the successive mobility prediction problem, and innovatively leverage DRL to solve the problem
- Design the RLMob framework and describe the interaction flow between the framework and the simulated environment
- Some advanced DRL algorithms that can be applied to this framework like PPO are introduced
- In our experiment, our method is performant

Thank you
for your careful listening!