Hierarchical Reinforcement Learning for Point of Interest Recommendation

Yanan Xiao\textsuperscript{1,6}, Lu Jiang\textsuperscript{2}, Kunpeng Liu\textsuperscript{3}, Yuanbo Xu\textsuperscript{4,7}, Pengyang Wang\textsuperscript{5,8,\#} and Minghao Yin\textsuperscript{1,6,\#}

\textsuperscript{1}School of Computer Science and Information Technology, Northeast Normal University, China
\textsuperscript{2}Department of Information Science and Technology, Dalian Maritime University, China
\textsuperscript{3}Department of Computer Science, Portland State University, USA
\textsuperscript{4}College of Computer Science and Technology, Jilin University, China
\textsuperscript{5}Department of Computer and Information Science, University of Macau, China
\textsuperscript{6}Key Laboratory of Applied Statistics of MOE, Northeast Normal University, China
\textsuperscript{7}Mobile Intelligent Computing (MIC) Lab, Jilin University, China
\textsuperscript{8}The State Key Laboratory of Internet of Things for Smart City, University of Macau, China
\{xiaoy117, ymh\}@nenu.edu.cn, jianglk761@dlmu.edu.cn, yuanbox@jlu.edu.cn, kunpeng@pdx.edu, pywang@um.edu.mo

Abstract

With the increasing popularity of location-based services, accurately recommending points of interest (POIs) has become a critical task. Although existing technologies are proficient in processing sequential data, they fall short when it comes to accommodating the diversity and dynamism in users’ POI selections, particularly in extracting key signals from complex historical behaviors. To address this challenge, we introduced the Hierarchical Reinforcement Learning Preprocessing Framework (HRL-PRP), a framework that can be integrated into existing recommendation models to effectively optimize user profiles. The HRL-PRP framework employs a two-tiered decision-making process, where the high-level process determines the necessity of modifying profiles, and the low-level process focuses on selecting POIs within the profiles. Through evaluations of multiple real-world datasets, we have demonstrated that HRL-PRP surpasses existing state-of-the-art methods in various recommendation performance metrics.

1 Introduction

With the popularity of location-based services (LBS), point-of-interest (POI) recommendations have become an important tool for users to navigate [Wang \textit{et al.}, 2023b] and explore cities [Qin \textit{et al.}, 2023]. Recommendation systems need to extract learning from users’ historical location data to provide accurate recommendation services. However, given the diversity and ever-changing nature of users’ points of interest, it becomes a major challenge to filter out signals with predictive value from numerous historical data. Therefore, there is an urgent need to develop a new recommendation paradigm that can deeply understand adapt to users’ behavioral patterns to improve recommendation accuracy.

Early POI recommendation research focused on sequential behavioral influence, using models such as recurrent neural networks (RNN) [Xu \textit{et al.}, 2022; Huang \textit{et al.}, 2020], long short-term memory (LSTM) [Liu \textit{et al.}, 2020; Luo \textit{et al.}, 2021] and gated recurrent units (GRU) [Manotumruksa \textit{et al.}, 2020]. With the increasingly available location-based social networks (LBSNs), research began to fuse geographic [Sun \textit{et al.}, 2020], semantic [Wu \textit{et al.}, 2019], temporal [Doan \textit{et al.}, 2019], and other multidimensional information [Feng \textit{et al.}, 2020] to improve the understanding of user behavior, but introducing additional model complexity. However, the fidelity of the model is limited by a key assumption: all historical POIs have the same influence in estimating the similarity between user preferences and target POIs. This assumption may lead to ignoring the unique contributions of different historical POIs in the prediction process. Therefore, it becomes critical to introduce an attention mechanism to distinguish the impact of historical check-ins. For example, attention-based models such as NAIS [He \textit{et al.}, 2018] and NASR [Li \textit{et al.}, 2017] evaluate the attention coefficient of each historical POI to determine its importance in recommending target check-ins.

Attention-based POI recommendation models, while improving performance, still face challenges such as the dilution effect caused by users’ diverse check-in histories. In a user’s check-in history, visits that truly reflect interest in a target POI may be obfuscated by a large number of irrelevant POIs, thus weakening the impact of key POIs. As shown in the Figure 1 illustrates the recommendation results of the NAIS model, where the score of each historical POI reflects its attention factor. Key historical POIs such as “Shopping Centers”, “Shoe Stores”, and “Jewelry Stores”, despite receiving high attention factors, are diluted in importance by other categories of POIs when all historical POI scores are aggregated. For example, POIs such as “Coffee Shops”, “In-
2 Definitions and Problem Formulation

This section provides foundational definitions to establish the context for our study on improving POI recommendation systems using hierarchical reinforcement learning.

2.1 Definitions

User Profile. Given user $u \in U = \{u_1, u_2, \ldots, u_L\}$, a series of POIs $P = \{p_1, p_2, \ldots, p_P\}$. Each user profile sequence is expressed as $E^u = (p^u_1, p^u_2, \ldots, p^u_{T_u})$. Where $p^u_t$ represents the POI visited by the specified user $u$ at time $t$, $p^u_{T_u}$ denotes the last visited POI.

User Profile.

Our contributions are threefold:

- We introduce an auxiliary POI recommendation model consisting of a profile modifier and a basic recommendation model. The joint training of these two models effectively removes the noisy POI from user profiles, resulting in higher prediction accuracy.

- A hierarchical reinforcement learning framework has been developed that is capable of recognizing and removing irrelevant information in an unsupervised manner, thus enhancing the recommendation process without the need for explicit labeling.

- After extensive experiments on a large number of real datasets, our model proves its effectiveness. When generalized to current mainstream recommender system models, the model achieves significant improvements in key performance metrics.

2.2 Problem Formulation

The primary objective of this research is to develop an optimization function that maximizes the probability of accurately predicting the next POI $p$ a user will visit. This involves refining a user’s original profile $E^u$ to focus more effectively on the POI that are most influential for future predictions. The refined profile can be expressed as

$$\hat{E}^u = F(E^u),$$

where $F$ comes from the process function aimed at refining the profile, and $\hat{E}^u$ represents the refined profile.

Given $E^u$, our task is to predict the POI $p^u_{T_u+1}$ that the user is most likely to visit next. Mathematically, this is formulated as

$$\arg\max_{p \in P} P(y = 1 | E^u, p^u_{T_u+1}),$$

where $P(y = 1 | E^u, p^u_{T_u+1})$ represents the conditional probability that the user visits POI $p$, $y = 1$ indicates that the prediction matches the actual next POI visited by the user. In contrast, $y = 0$ means that there is no match.

To evaluate the effectiveness of the predictive model, we use the following accuracy metric define as

$$\text{Accuracy} = \frac{1}{|U|} \sum_{u \in U} \arg\max_{p \in P} P(y = 1 | \hat{E}^u, p^u_{T_u+1}).$$

The efficacy of the function $F$ is quantified by calculating the average maximum probability that the predicted POIs correspond to the POIs actually visited by users.

3 Method

In this section, we first give an overview of the proposed model, then we introduce a hierarchal reinforcement learning algorithm to revise user profiles and finally present the training process of the entire model.
3.1 Overview Framework

Our model improves the fundamental recommendation system by refining user profiles. It identifies and removes "noisy" POIs—irrelevant check-ins that obscure the impact of significant POIs. This process relies on a hierarchical reinforcement learning algorithm, which divides the profile modification into sequential decision-making tasks at both high and low levels. The agent’s actions, guided by a modification policy, aim to optimize the user profile summary. After each modification cycle, the agent receives feedback from the environment, comprising the dataset and an initial recommendation model, to adjust its policy. Subsequently, the basic recommendation model is retrained based on the agent’s updated profile summaries. This joint training approach, which involves both the profile editor and the recommendation model, ensures more accurate POI recommendations. Figure 2 illustrates this framework.

3.2 The Basic Recommendation Model

The key element of recommendation is to accurately characterize the user’s preferences based on higher profile $E^u$. The general idea is that we represent each historical POI $p^u_i$ as a revalued low-dimensional embedding vector $v^u_i$ and summarize the embeddings of all historical POIs $v^u_1, v^u_2, \ldots, v^u_t$ to denote the preference of the user $u$’s $v_u$. If we also denote the target POI $p_t$ as the embedding vector $v_t$, the probability of recommending the POI $p_t$ to user $u$, i.e., $P(y = 1|E^u, p_t)$, can be represented as

$$P(y = 1|E^u, p_t) = \sigma(v^T_u v_t), \quad (4)$$

where $y = 1$ indicates that $p_t$ is recommended to the user $u$ and $\sigma$ is the sigmoid function to transform the input into a probability. Then the key question is how to obtain the aggregated embedding $v_u$. One straightforward way is to average the embeddings of all the historical POIs, i.e., $v_u = \frac{1}{t_u} \sum_{i=1}^{t_u} p^u_i$. However, equally treating all the POIs’ contributions may impact the representation of a user’s real interest in the target POI. Thus, as NAIS does, we can adopt the attention mechanism to estimate an attention coefficient $a_{it}^u$ for each historical POI $p^u_i$ when recommending $p_t$. Specifically, we parameterize the attention coefficient $a_{it}^u$ as a function with $v^u_i$ and $v_t$ as input and then aggregate the embeddings according to their attentions defined as

$$\hat{v}_u = \sum_{i=1}^{t_u} a_{it}^u p^u_i = f(v^u_i, v_t), \quad (5)$$

where $f$ can be instantiated by a multi-layer perceptron on the concatenation or the element-wise product of the two embeddings $v^u_i$ and $v_t$.

In addition to NAIS, our framework is compatible with various types of existing attention mechanisms and recommendation models. To ensure the generalizability and consistency of the study, we chose NAIS as the main mechanism for the computation of attention in our experiments. This choice not only demonstrates the flexibility of the framework, but also provides a unified benchmark for comparing different models.

3.3 Profile Reviser

As mentioned above, the purpose of the profile reviser is to remove noisy processes that do not contribute much to the prediction. Inspired by the theory of hierarchical abstract machines [Parr and Russell, 1997], we describe the task of profile revision as a hierarchical Markov Decision Process (MDP). In general, we decompose the entire MDP task $M$ into two classes of subtasks $M^p$ and $M^l$, where $M^p$ is a high-level abstract task in the hierarchy, and solving it solves...
the entire MDP \( M \), and \( M^l \) is a low-level prototask in the hierarchy. Each kind of task is defined as a 4-tuple MDP \((S, A, T, R)\), where \( S \) is a set of states, \( A \) is a set of actions, \( T \) is a transition model mapping \( S \times A \times S \) into probabilities in \([0,1]\), and \( R \) is a reward function mapping \( S \times A \times S \) into real-valued rewards.

We formulate our task by a high-level task and a low-level task. Specifically, given a sequence of historical POIs \( E^u := (p_1^u, p_2^u, ..., p_n^u) \) of user \( u \) and target POI \( p \), the agent performs a high-level task of one binary action to determine whether to revise the profile \( E^u \) or not. If it decides to revise \( E^u \), the agent performs a low-level task of multiple actions to determine whether to remove each historical POI \( p_i^u \) or not. After the low-level task is finished, the overall task is finished. If the high-level task decides to make no revision, the low-level task will not be executed and the overall task is directly finished.

We formulate the profile reviser as two-level MDPs because some of the user profiles are discriminative and can already be correctly predicted by the basic recommendation model. We can simply keep those profiles as the original ones and only revise the ones that result in false recommendations. Out of this consideration, we design a high-level task to decide whether to revise the profile of a user or not, and a low-level task to decide which POI in the profile should be removed. We will introduce the details of how to design the state, action, and reward for the two-level tasks.

**State.** The high-level task takes an action according to the state of the whole profile \( E^u \) and the low-level task takes a sequence of actions according to the state of each POI \( p_i^u \in E^u \). We define different state features for the two tasks.

- **Low-level task:** When determining to remove a historical POI \( p_i^u \in E^u \), we define the state features \( s_i^l \). As the cosine similarity between the embedding vectors of the current historical POI \( p_i^u \) and the target POI \( p \), the element-wise product between them, and also the average of the two previous features over all the reserved historical POIs, where the embedding vector of a POI \( p_i \) can be provided by a pre-trained basic recommendation model. We also treat the user’s level of interest in the POI as an additional state feature that enhances the contribution of \( p_i^u \) to \( p \), in addition to the similarity-based features. For simplicity, we omit the superscript \( u \) in all the notations on the state features.

- **High-level task:** When determining to revise a whole profile \( E^u \), we define the state features \( s^h \) as the average cosine similarity between the embedding vectors of each historical POI in \( E^u \) and the target POI and the average element-wise product between them. We also define an additional state feature as the probability \( P(y = 1|E^u, p_i) \) of recommending \( p_i \) to user \( u \) by a basic recommendation model. The probability reflects the credibility of the POI \( p_i \) recommended based on the profile \( E^u \). The lower the probability of recommendation, the more effort should be put into revising \( E^u \). Note we train the profile reviser only based on the positive instances, i.e., a user profile paired with a real target POI, as negative instances with random target POIs can hardly guide the agent to select the contributing POIs to the target POI. Thus \( P(y = 0|E^u, p_i) \) for a negative instance is not calculated.

**Action and Policy.** We define the high-level action \( a^h \in \{0,1\} \) as a binary value to represent whether to revise the whole profile of a user or not and define a low-level action a \( d^l \in \{0,1\} \) as a binary value to represent whether to remove the historical POI \( p_i^u \) or not. We perform a low-level action \( a_i^l \) according to the policy function defined as

\[
H^l = \text{ReLU}(W^l s^l + b^l),
\]

\[
\pi(s_i^l, a_i^l) = P(a_i^l|s^l_i, \Theta^l) = a_i^l \sigma(W_2^l H^l) + (1 - a_i^l)(1 - \sigma(W_2^l H^l)),
\]

where \( W_1^l \in \mathbb{R}^{d_1^l \times d_2^l}, W_2^l \in \mathbb{R}^{d_2^l \times 1} \) and \( b^l \in \mathbb{R}^{d_2^l} \) are the parameters to be learned with \( d_1^l \) as the number of the state features and \( d_2^l \) as the dimension of the hidden layer. Notation \( H^l \) represents the embedding of the input state. We denote \( \Theta^l = \{W_1^l, W_2^l, b^l\} \). The sigmoid function \( \sigma \) is used to transform the input into a probability. The high-level action is performed according to the similar policy function with different parameters \( \Theta^h = \{W_1^h, W_2^h, b^h\} \).

**Reward.** The reward is a signal to indicate whether the performed actions are reasonable or not. We assume that every low-level action in the low-level task has a delayed reward after the last action \( a_i^l \) is performed for the last POI \( p_i^u \in E^u \). In other words, the immediate reward for a low-level action is zero except for the last low-level action. Thus, we define the reward for each low-level action define as

\[
R(a_i^l, s_i^l) = \begin{cases} 
\log p(E^u, p_i) - \log p(E^u, p_i), & \text{if } t = t_u; \\
0, & \text{otherwise,}
\end{cases}
\]

where \( p(E^u, p_i) \) is an abbreviation of \( P(y = 1|E^u, p_i) \) and \( E^u \) is the revised profile, which is a subset of \( E^u \). For the special case \( E^u = \phi \), i.e., all the historical POIs are removed, we randomly select a POI from the original set \( E^u \). The reward is defined as the difference between the log-likelihood after and before the profile is revised. A positive difference indicates a positive utility gained by the revised profile.

If the high-level task chooses the revising action, it calls the low-level task and receives the same delayed reward \( R(a_i^l, s_i^l) \) after the last low-level action is performed. Otherwise, it keeps the original profile and obtains a zero reward as \( \log p(E^u, p_i) \) is not changed.

In addition, we define an internal reward \( G(a_i^l, s_i^l) \) which is used only inside the low-level task to speed up its local learning and does not propagate to the high-level task. Specifically, we first calculate the average cosine similarity between each historical POI and the target POI after and before the profile is revised, and then use the difference between them as the internal reward \( G(a_i^l, s_i^l) \). The internal reward encourages the agent to select the most relevant POIs to the target POI. Finally, we sum \( G(a_i^l, s_i^l) \) and \( R(a_i^l, s_i^l) \) as the reward for the low-level task.

**Objective Function.** We aim at finding the optimal parameters of the policy function defined in Eq. 6 to maximize the
expected reward defined as
\[ \Theta^* = \arg\max_\Theta \sum_\omega P_\omega(\tau; \Theta)R(\tau), \quad (8) \]
where \( \Theta \) represents either \( \Theta^h \) or \( \Theta^l \), \( \tau \) is a sequence of the sampled actions and the transited states, \( P_\omega(\tau; \Theta) \) denotes the corresponding sampling probability and \( R(\tau) \) is the reward for the sampled sequence \( \tau \). The sampled sequence \( \tau \) can be \( \{s^0_1, a^1_1, s^1_2, \ldots, a^l_{t-1}, s^l_t\} \) for the low-level task and \( \{s^h, a^h\} \) for the high-level task.

Since there are too many possible action-state trajectories for the entire sequences of the two tasks, we adopt the policy gradient theorem and the Monte Carlo policy gradient methods [Sutton and Barto, 2018; Thomas and Brunskill, 2017] to sample \( M \) action-state trajectories. The index \( m \) denotes the \( m \)-th trajectory from these samples, where \( M \) is the total number of sampled trajectories. Based on these samples, we calculate the gradient of the parameters for the low-level policy function defined as
\[ \nabla_\theta = \frac{1}{m} \sum_{m=1}^M \sum_{t=1}^{t_u} \nabla_\theta \log \pi_\theta(s^m_t, a^m_t) \left( R(a^m_t, s^m_t) + G(a^l_{t-1}, s^l_{t-1}) \right) \quad (9) \]
where the reward \( R(a^m_t, s^m_t) + G(a^l_{t-1}, s^l_{t-1}) \) for each action-state pair in sequence \( \tau^{(m)} \) is assigned the same value and equals to the terminal reward \( R(a^m_T, s^m_T) + G(a^l_{T-1}, s^l_{T-1}) \). The gradient for the high-level policy function is defined as
\[ \nabla_\theta J = \frac{1}{m} \sum_{m=1}^M \nabla_\theta \log \pi_\theta(s^m, a^m) R(a^m, s^m) \quad (10) \]
where the reward \( R(a^m, s^m) \) is assigned as \( R(a^m_t, s^m_t) \) when \( a^m = 1 \), or 0. We omit the superscript \( h \) and \( l \) in Eq. 8 and Eq. 9 for simplicity.

### 3.4 Model Training

The two models of the profile reviser and the basic recommendation model are interleaved together, and we need to train them jointly. The training process is shown in Algorithm 1, where we first pre-train the basic recommendation model based on the original dataset, then we fix the parameters of the basic recommendation model and pre-train the profile reviser to automatically revise the user profiles; finally, we jointly train the models together. Same as the settings, to have a stable update, each parameter is updated by a linear combination of its old version and the new old version, i.e. defined as
\[ \Theta_{new} = \lambda \Theta_{new} + (1 - \lambda) \Theta_{old}, \quad (11) \]
where \( \lambda \ll 1 \). The time complexity is \( O(L(NuM)) \), where \( L \) is the number of epochs, \( N \) is the number of instances, \( T_I \) is the average number of historical courses and \( M \) is the Monte Carlo sampling time.

### 4 Experiment

We evaluate the performance of HRL-PRP in the next location prediction task. We aim to answer the following five main research questions:

- **Q1**: How does HRL-PRP perform in the next-location prediction task?
- **Q2**: What is the effectiveness of the roles of high-level and low-level agents in the decision-making process?
- **Q3**: How does reward design affect the performance of HRL-PRP?
- **Q4**: How can the necessity and rationality of the framework be analyzed through examples?
- **Q5**: How hyperparameters settings affect recommendation performance?
<table>
<thead>
<tr>
<th>Methods</th>
<th>Revised profile or the learned attentions</th>
<th>The target POI</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRL-PRP</td>
<td>Cafes, Cake Stores, Clothing Store, Airports, Train Stations</td>
<td>Baker’s Store</td>
</tr>
<tr>
<td>NAIS</td>
<td>Cafes (13.22), Cake Stores (9.41), Clothing Store (9.55), Airports (5.42), Train Stations (7.14)</td>
<td>Baker’s Store</td>
</tr>
<tr>
<td>HRL-PRP</td>
<td>Student Center, Gym/Fitness Center, Bar, Library</td>
<td>Student Apartment</td>
</tr>
<tr>
<td>NAIS</td>
<td>Student Center (15.20), Gym/Fitness Center (15.78), Bar (7.22), Library (12.41)</td>
<td>Student Apartment</td>
</tr>
</tbody>
</table>

Table 2: Case studies of the profiles revised by HRL-PRP and the attention coefficients learned by NAIS.

4.1 Experiment Settings

Datasets. In our study, we additionally included other datasets from Tokyo, Brightkite, Instagram, and Gowalla, which are widely used benchmarks in POI recommendation studies. Each dataset contains a series of historical POIs and a specific target POI. During the training phase, the last POI of the sequence is set as the target, and the rest constitutes the historical context. When generating negative samples, the target POI is replaced by four random POIs. In the testing phase, each check-in in the test set is considered as a target event and 99 random negative instances are paired to fully evaluate the model performance.

Baseline Methods. We compare HRL-PRP with eight base algorithms for comparison, including (1) FPMC [Rendle et al., 2010], (2) LSTM [Memory, 2010], (3) ST-RNN [Xu et al., 2022], (4) HST-LSTM [Kong and Wu, 2018], (5) SERM [Yao et al., 2017], (6) Deepmove [Feng et al., 2018], (7) LSTPM [Sun et al., 2020], (8) STAN [Luo et al., 2021], (9) CARA [Manotumruksa et al., 2020], (10) ATST-LSTM [Huang et al., 2019], (11) GeoSan [Lian et al., 2020].

Evaluation Metrics. Our evaluation uses the following metrics: hit rate (HR), normalized discounted cumulative gain (NDCG), recall, F1 score, and mean accuracy (MAP).

Implementation Details. For the profile reviser, sampling time $M$ is set as 3, and the learning rate is set as $0.001/0.0005$ at the pre-training and joint-training stages respectively. In the policy function, the dimensions of the hidden layer $d_2$ and $d_3$ are both set as 8. For the basic recommender, the dimension of the POI embeddings is set to 128, the learning rate is $0.001$ at both the pre-training and joint-training stages, and the size of the minibatch is 128. The delayed coefficient $\lambda$ for the joint training is $0.0005$.

4.2 Overall Performance (Q1)

Table 1 shows the performance metrics of the HRL-PRP framework before and after combining it with the baseline recommendation model. The framework outperforms the baseline methods on key metrics, especially on sparse datasets, compared to FPMC methods. Serialization methods such as LSTM and LSTPM underperform in the distinction of historical POIs. While methods such as STAN and CARA, which fuse multidimensional information, are improved, they are affected by data noise and have different effects. Overall, existing recommender systems have limitations in handling diverse user interests. HRL-PRP more accurately reflects user preferences by effectively removing noisy POIs, thus achieving significant improvement in recommendation accuracy.

4.3 The Study of HRL-PRP Agents (Q2)

The design of High-level Agent. To demonstrate the effectiveness of high-level tasks, we compare the HRL architecture with the single-tier RL architecture (Deep Q-Learning, which directly decides whether to delete each POI mainly through low-level tasks) on several evaluation metrics. The results show that HRL outperforms single-tier RL on all metrics, highlighting the importance of high-level agents in maintaining and adjusting the diversity of user profiles. For example, the #Categories/#POIs values of HRL-modified profiles averaged 0.62 and 0.64, which were lower than those of single-tier RL at 0.68 and 0.72, suggesting that HRL produced more consistent profiles. This demonstrates the effectiveness of HRL-PRP’s strategy of efficiently deciding to retain or modify profiles through high-level tasks.

The design of Low-level Agent. We compare the proposed HRL with the greedy correction algorithm. First, if $\log P(y = 1|\mathcal{E}^u, p_l < \mu_1)$ decides to modify the whole contour $\mathcal{E}^u$, and if the cosine similarity between $e^u_i \in \mathcal{E}^u$ and $p_l$ is less than $\mu_2$, then $e^u_i \in \mathcal{E}^u$ is further deleted. $e^u_i \in \mathcal{E}^u$. In Figure 4, we tune $\mu_1$ from 0.1 to 0.6 with an interval of 0.5, and tune $\mu_2$ from -0.1 to 0.1 with an interval of 0.5, and get the best results for the two data when $\mu_1 = 0.4$ and $\mu_2 = 0.5$, which are 1.43% and 1.22% less than HRL-PRP. Note that the best performance is obtained when the number of remaining POIs is almost the same as HRL-PRP.
4.4 The Study of Reward (Q3)

We investigated the performance of different attention mechanisms driven by intrinsic rewards. As shown in Figure 5, among the multiple attention models examined, NAIS exhibits the best performance, while RNN and LSTM perform weakly. This performance difference mainly stems from the fact that RNN and LSTM do not sufficiently consider the episodic and non-sequential nature of user behavior in POI recommendation. This point highlights how the reward mechanism affects attention allocation in different models and how this allocation significantly affects the overall recommendation effectiveness.

4.5 The Study of Case Performance (Q4)

As shown in Table 2, the 2 cases of profiles corrected by the proposed HRL-PRP. The cases show that HRL-PRP is effective in removing spurious interest points that are not related to the target interest points. In contrast, while NAIS assigns high attention to contributing historical points of interest, some irrelevant points of interest do not receive significantly different or even higher attention than relevant points of interest, resulting in a weakened effect of truly contributing points of interest when aggregating all historical points of interest. As a result, the performance of recommendation models based on such differentiated revised profiles is improved.

4.6 The Study of Hyperparameters (Q5)

In reinforcement learning, experimental performance is extremely sensitive to parameter selection. As shown in Figure 6, we conducted an extensive learning rate analysis, scrutinizing the effects of varying embedding dimensions and training batch sizes. Furthermore, the impact of different learning rates on joint training was also meticulously scrutinizing the effects of varying embedding dimensions extremely sensitive to parameter selection. As shown in Figure 5,

6 Conclusion

This study proposes a hierarchical reinforcement learning preprocessing framework (HRL-PRP). We categorize the tasks into a two-level decision-making process: the high-level task is responsible for determining whether or not the current user’s profile needs to be modified; the low-level task focuses on deciding which POIs to modify specifically. By jointly training the user profile modifier and the underlying recommendation model, we aim to improve the overall recommendation accuracy. The model simplifies the complexity of the recommendation process by efficiently filtering irrelevant information while focusing on key content without explicitly supervising the signal. In the future, we plan to apply this model to other recommendation domains, to explore its potential for processing key items in users’ historical data.
Acknowledgments

This work is supported by NSFC(under Grant No. 61976050, 62376048), Jilin Province Science and Technology Department Project (under Grant No. YDZ202201ZYTS415, 20240602005RC), Jilin Education Department Project No.JJKH20231319KJ. This work is supported by the Science and Technology Development Fund (FDCT), Macau SAR (file no. 0123/2023/RIA2, 001/2024/SKL), the Start-up Research Grant of University of Macau (File no. SRG2021-00017-IOTSC). This work is supported by the Fundamental Research Funds for the Central Universities”under Grant No.3132024264.

References


