SNPR: A Serendipity-Oriented Next POI Recommendation Model

Mingwei Zhang
Northeastern University
Shenyang, China
zhangmw@swc.neu.edu.cn

Yang Yang
Northeastern University
Shenyang, China
yangyang@stumail.neu.edu.cn

Rizwan Abbas
Northeastern University
Shenyang, China
swerizwan@gmail.com

Ke Deng
RMIT University
Melbourne, Australia
Ke.Deng@rmit.edu.au

Jianxin Li*
Deakin University
Melbourne, Australia
jianxin.li@deakin.edu.au

Bin Zhang
Northeastern University
Shenyang, China
zhangbin@mail.neu.edu.cn

ABSTRACT

Next Point-of-Interest (POI) recommendation plays an important role in location-based services. The state-of-the-art methods utilize recurrent neural networks (RNNs) to model users' check-in sequences and have shown promising results. However, they tend to recommend POIs similar to those that the user has often visited. As a result, users become bored with obvious recommendations. To address this issue, we propose Serendipity-oriented Next POI Recommendation model (SNPR), a supervised multi-task learning problem, with objective to recommend unexpected and relevant POIs only. To this end, we define the quantitative *serendipity* as a trade-off of *relevance* and *unexpectedness* in the context of next POI recommendation, and design a dedicated neural network with Transformer to capture complex interdependencies between POIs in user's check-in sequence. Extensive experimental results show that our model can improve *relevance* significantly while the *unexpectedness* outperforms the state-of-the-art serendipity-oriented recommendation methods.

CCS CONCEPTS

• **Information systems** → **Collaborative filterings**; **Location based Services**.

KEYWORDS

Point-of-Interest; Next POI Recommendation; Serendipity-Oriented Recommendation; Multi-Task Learning; Transformer

ACM Reference Format:
Mingwei Zhang, Yang Yang, Rizwan Abbas, Ke Deng, Jianxin Li, and Bin Zhang. 2021. SNPR: A Serendipity-Oriented Next POI Recommendation Model. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3482394

*Corresponding author.

CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia

ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3482394

1 INTRODUCTION

With the advancement of mobile technologies, location-based social networks (LBSNs) like Foursquare, Facebook place and Yelp are becoming pervasive in our daily lives [9]. Next POI recommendation in LBSNs can help users explore more interesting locations when they are in unfamiliar regions; on the other side, it can help shopkeepers to attract more customers. Recently, Next POI recommendation has become a hot research topic [10]. For catering to users' preferences, the majority of next POI recommendation methods analyze users' past behaviors to obtain their profiles and then adopt deep learning models to identify the most relevant POIs. As in the general recommendation, the well-known issue *over-specialization* has not been properly addressed such that the recommended POIs are expected to users and thus users will feel bored and corny, especially in their familiar regions. This situation motivates LBSNs to call for solutions on serendipity-oriented next POI recommendation.

The general serendipity-oriented recommendation has attracted many attentions [21, 24, 25]. However, the research in this field is still in its early stages and two open problems have not been completely addressed yet. First, there is no wide consensus on the definition and evaluation metric for serendipity in recommendation because serendipity includes the emotional dimension. But the principle recognized by researchers is that both *relevance* and *unexpectedness* are important for serendipity (e.g., [16]). The relevance refers to the consistency between the recommendations and user's preferences demonstrated in past behaviours, while the unexpectedness indicates how unlikely the users expect the recommendations. So far, no quantitative definition of serendipity is known in the literature in the field of next POI recommendation. Second, while the introduction of unexpectedness via serendipity mitigates the issue of tedious recommendation, the direct side effect is the *diminished relevance* of recommendation against the users' preferences [21]. It is critical to balance the unexpectedness and relevance in serendipity-oriented recommendation.

With objective to tackle the above two problems in next POI recommendation, this paper proposes a novel Serendipity-oriented

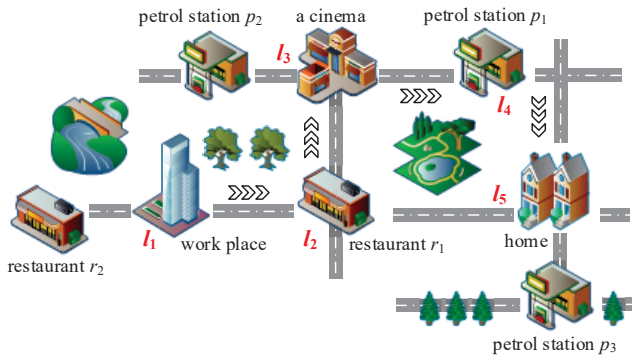

Figure 1: An example of complex interdependencies between POIs in user's check-in sequence.

Next POI Recommendation model (SNPR). SNPR is a supervised multi-task learning problem highlighted by two characters, i.e., (i) the quantitative definition of POI recommendation serendipity which leads to the desirable ground truth for training, and (ii) representing the complex relationship among POIs in user's check-in sequences with a dedicated deep neural network to improve recommendation relevance without loss of unexpectedness.

First, the quantitative definitions of unexpectedness and relevance have been deliberately designed so that the serendipity is sensible and intuitive in the context of next POI recommendation. From the past behaviours of a user, the unexpectedness and relevance of each candidate POI can be evaluated. Based on this, serendipitous next POI recommendation is modeled as a supervised multi-task learning problem. One task is to achieve high performance on unexpectedness, and the other is to achieve high performance on relevance.

Second, to improve relevance of next POI recommendation without loss of unexpectedness, SNPR adopts Transformer [39] to model the past behaviours of a user based on the user's check-in sequence. Transformer allows each visited POI to attend to all the other visited POIs in the check-in sequence. In contrast, the existing next POI recommendation methods utilize RNNs [38]. Compared with Transfomer, RNNs have an inherent limitation as they can only characterize the sequential dependency of POIs in the check-in sequence. For example, suppose $\{l_1, l_2, l_3, l_4, l_5\}$ is a user's POI check-in sequence as depicted in Figure 1, in which l_1 is the user's work place, l_2 and l_3 are a restaurant and a cinema respectively, l_4 is a petrol station, and l_5 is her home. Then, RNNs will model the sequence strictly in accordance with the temporal order. Take the case of l_2 in the user's check-in sequence, the user's choice "$l_2 = r_1$" (i.e., restaurant r_1 rather than r_2) would mainly depend on l_3, and would also depend on the starting place l_1 and the destination l_5. However, RNNs will only model the dependencies of l_2 on l_1. Take another case of l_4, the user's choice "$l_4 = p_1$" (i.e., petrol station p_1 rather than p_2 and p_3) would jointly depend on l_3 and l_5. However, RNNs will model the dependencies of l_4 on l_3, l_2 and l_1 with a deceasing trend. In contrast, all the potential dependencies among POIs in a check-in sequence can be modeled by Transfomer.

At last, we have conducted extensive experiments on two benchmark datasets. Our model outperforms the baseline serendipity-oriented recommendation methods significantly in various aspects. The major contributions of this paper are summarized as follows.

- We define a model of serendipity in next POI recommendation scenario, and provide a quantitative definition of *serendipity* with consideration of both *relevance* and *unexpectedness*. The model can flexibly make a trade-off between relevance and unexpectedness by adjusting a given hyper-parameter.
- We formulate the next serendipitous POI recommendation as a supervised multi-task learning problem, namely SPNR, with a dedicated neural network where Transformer is applied to capture the complex interdependencies among POIs.
- We have an interesting finding from experimental results that the task of unexpectedness can benefit the task of relevance when the recommendation list size is large. Our model has no *diminished relevance* issue that existing serendipity-oriented recommendation methods face.

2 RELATED WORK

Serendipity in Recommender Systems. There is a long history of accuracy-oriented recommender systems, where the more accurate they are, the more obvious over-specialization becomes, which accelerates the research of serendipity-oriented recommendation.

One line of research are specialized algorithms for serendipity-oriented recommendation. TANGENT [32] performed on a bipartite graph to detect groups of like-minded users and suggested items relevant to users from different groups. RWR-KI [5] suggested serendipitous items by random walk with restarts on an item similarity graph. StumbleOn [31] set up a framework consisting of a surprise component, a value component, and a learning component, which worked together to reason about what information is serendipitous and generate recommendations for users. SerRec [33] used transfer learning to recommend serendipitous items. SIRUP [28] was a model for serendipity-oriented recommendation, which was composed by two elements, i.e., novelty check and coping potential check, and inspired by curiosity theories. HAES [24] was a hybrid approach for recommending movies with elastic serendipity by quantifying the elasticity in recommendation system. DESR [25] was a model reinforcing the user preference direction and explainablity in serendipity-oriented recommendation.

The other line of research extends existing accuracy-oriented recommendation with serendipity. Kotkov et al. [20] applied a greedy strategy to improve recommendation serendipity through resorting the recommended lists. Karpus et al. [17] introduced ontology-based contextual pre-filtering to remove movies particularly familiar to users. Yang et al. [42] employed One-Class collaborative filtering to find serendipitous items by pre-filtering the unsatisfying and uninteresting items. Zhang et al. [43] implemented three basic algorithms to generate corresponding lists and used hybrid rank-interpolation to combine the outputs. Said et al. [36] proposed k-furthest neighbor (kFN) algorithm to recommend serendipitous items to users. To the best of our knowledge, there is only one research work on serendipitous POI recommendation [18]. It utilized the kFN model [36] to make serendipitous POI recommendation.

Different from the above methods, we model serendipity-oriented recommendation as a supervised multi-task learning problem, and integrate relevance and unexpectedness objectives into one dedicated neural network. Our model has no *diminished relevance* issue, and can even improve the performance on accuracy when the size of recommendation list is large.

Personalized Next POI Recommendation. The next POI recommendation has attracted many attentions of researchers due to its values in a wide range of applications. In the early stage, some pattern-based methods [26, 30] were proposed, which generally mined explicitly pre-defined patterns on dense trajectories, and had relatively limited performance in capturing mobility regularities about user movement. Then, some model-based methods were proposed, due to their ability to capture complex mobility regularities and fuse heterogeneous data. Early such studies [4, 11, 34] were based on Markov model, which generally predicted the probability of future visit by constructing a location transition matrix. However, Markov-based models aimed to learn transition probability between successive locations, thus failing to capture high order sequential regularity. Recent studies were based on RNNs, inspired by the strong capability of RNNs in sequential data modeling. For example, CARA [29] captured users' dynamic preferences by exploiting GRU's gate mechanism. DeepMove [8] designed a multi-modal RNN to capture the sequential transition. TMCA [23] adopted the LSTM-based framework and STGN [44] adopted the gated LSTM framework to learn spatial-temporal contexts respectively. LSTPM [38] designed a geo-dilated RNN to fully exploit the geographical relations among non-consecutive POIs.

Different from the state-of-the-art RNN-based next POI recommendation methods, we utilize Transformer to obtain the initial representations of POIs. Besides, we consider serendipity-oriented, not only accuracy-oriented next POI recommendation problem.

3 KEY CONCEPTS AND PROBLEM STATEMENT

To help readers to understand easily, Table 1 lists the notations used in the following sections.

Table 1: Notations.

Symbol	Description
$\mathcal{U}$	The set of all users $\{u_1, u_2, ..., u_{\|\mathcal{U}\|}\}$
$\mathcal{L}$	The set of all POIs $\{l_1, l_2, ..., l_{\|\mathcal{L}\|}\}$
C	The set of all Check-ins $\{c_1, c_2, ..., c_{\|C\|}\}$
$u.proc$	User profile: the set of observed check-ins of user u
$u.loc$	The set of visited POIs that appeared in $u.proc$
$u.cat$	The set of categories of the POIs in $u.loc$
$l.cat$	The category related to POI l
$\mathbf{T}$	A user's trajectory sequence $\{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_n\}$

3.1 The Key Concepts

Serendipity is a difficult concept to study, as it includes an emotional dimension. However, many related works indicate that relevance and unexpectedness are important for serendipity [1, 3, 6, 14, 16, 19, 21, 22]. To be serendipitous for a user, items must be relevant and unexpected in the context of recommender systems. As discussed in [3], *serendipity* emphasizes whether the user feels surprised when she sees a relevant recommendation, which implies that the item should not only be relevant to the user's interests, but also be unexpected (i.e., not intentionally looked for by the user). The above related works have different expressions and measurements on *relevance* and *unexpectedness* in different scenarios, which might cause confusion. Consequently, we propose the definitions of POI relevance and unexpectedness.

In the POI recommendation scenario, let $\mathcal{U} = \{u_1, u_2, ..., u_{|\mathcal{U}|}\}$ denote a set of LBSN users, and $\mathcal{L} = \{l_1, l_2, ..., l_{|\mathcal{L}|}\}$ be a set of POIs.

POI: A POI $l \in \mathcal{L}$ is represented by $l = (l_{cat}, l_{geo})$. l_{cat} indicates the POI category related to l, and l_{geo} indicates the location of l on the geographic surface represented by a longitude-latitude pair (lon_l, lat_l).

Check-in and User profile: A check-in c is a visit log consisting of three attributes: a user, a POI and the time of visit. User profile "$u.prof$" indicates the set of observed check-ins made by u. "$u.loc$" indicates the set of visited POIs appearing in the check-ins in "$u.prof$". Note that a user u may visit a POI l multiple times, and correspondingly, "$l \in u.loc$" may appear in "$u.prof$" multiple times. "$u.cat$" indicates the set of categories related to the POIs appearing in "$u.loc$".

POI relevance is equivalent to POI recommendation accuracy. Thus, a POI $l \in \mathcal{L}$ is considered to be relevant to user u if she has visited it, as general recommendation methods do.

Definition 1 (POI relevance)**.** Given a user u and a POI $l \in \mathcal{L}$, the *relevance* of l to u is defined as:

$$relevance(l, u) = \begin{cases} 1 & l \in u.loc \\ 0 & l \notin u.loc \end{cases} \tag{1}$$

where "$u.loc$" is the set of user u's visited POIs.

POI unexpectedness is modeled as the difference between a POI and a user profile in this work. A POI $l \in \mathcal{L}$ is considered to be unexpected to a user u if it is different from the user profile "$u.prof$". To well enhance the ability of difference evaluation in serendipitous POI recommendation, we will exploit the significance of location categories. For example, a user has often visited "*shopping center*" POIs and never gone to "*gym*" POIs, a recommended POI "*a specific yoga gym*" with the category of "*gym*" will be very unexpected to her. In addition, even two POIs l_i and l_j belong to the same category, there is also a difference between them to some extent. For example, two specific yoga gyms will be popular within different user groups and have different *unexpectedness* degree to a specific user. Therefore, we define POI unexpectedness by considering the difference between categories and the difference between POIs in a joint way.

Definition 2 (POI unexpectedness)**.** Given a user u and a POI $l \in \mathcal{L}$, the unexpectedness of l to u is defined as:

$$unexp(l, u) = \gamma * dif^{cat}(l, u) + (1 - \gamma) * dif^{POI}(l, u) \tag{2}$$

where $\gamma \in [0, 1]$ is a trade-off parameter between the category difference and the POI difference components.

The category difference of l to u is defined as:

$$dif^{cat}(l,u) = \min_{cat \in u.cat} dist(l.cat, cat) \quad (3)$$

$$dist(cat_i, cat_j) = 1 - J(cat_i, cat_j) = 1 - \frac{|\mathcal{U}_{cat_i} \cap \mathcal{U}_{cat_j}|}{|\mathcal{U}_{cat_i} \cup \mathcal{U}_{cat_j}|} \quad (4)$$

where "$l.cat$" indicates the category related to POI l, and $\mathcal{U}_{cat}$ indicates the set of users who have visited the POIs with category cat at least once. $dist(cat_i, cat_j)$ indicates the distance between categories cat_i and cat_j and is modeled as their Jaccard distance. Two categories with more same users like "shopping center"and "cinema" will have lower distance.

The POI difference of l to u is defined as:

$$dif^{POI}(l,u) = \frac{1}{|u.loc|} \sum_{i=1}^{|u.loc|} 1 - sim(l, l_i) \quad (5)$$

where "$l_i \in u.loc$" is a visited POI of u. $sim(l, l_i)$ can be any kind of similarity between l and l_i ($sim(l, l_i) \in [0, 1]$). For example, it might be the point-wise mutual information based similarity [15]. In this paper, we adopt cosine similarity[35].

3.2 Problem Formulation

For each user $u \in \mathcal{U}$, we can obtain a trajectory sequence represented by $\mathbf{T} = \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_n\}$ from user profile "$u.prof$", where n is the index of the current trajectory. Each trajectory $\mathcal{T}_i \in \mathbf{T}$ consists of a sequence of POIs visited by user u in a consecutive order, i.e., $\mathcal{T}_i = \{l_1, l_2, ..., l_{|\mathcal{T}_i|}\}(l \in \mathcal{L})$. In this paper, we treat user's all check-ins in one day as a single trajectory.

As discussed in Definition 1 and Definition 2, we embody the relevance as the prediction accuracy of next POI recommendation and the unexpectedness as difference between POIs and user profiles. Based on these conditions, a POI $l \in \mathcal{L}$ is considered to be serendipitous to a user u if it is both relevant and unexpected. So, we model the POI serendipity as:

$$serendipity(l,u) = \lambda * relevance(l,u) + (1-\lambda) * unexp(l,u) \quad (6)$$

where $\lambda \in [0, 1]$ is a trade-off parameter between the relevance and unexpectedness components.

We formulate the serendipity-oriented next POI recommendation problem as follows. Given the set of POIs $\mathcal{L}$, the set of users $\mathcal{U}$ and their historical check-in data C, we aim to predict whether user u will have a potential feeling of serendipity with POI l, and recommend the top-K POIs with predicted serendipity scores to u at the next timestamp. Our task can be formulated to learn a prediction function $serendipity^*(l,u) = \mathcal{F}(l,u|\theta, \mathcal{U}, \mathcal{L}, C)$, where $serendipity^*(l,u)$ denotes the degree that u will feel serendipitous with l, and θ denotes the model parameters of function $\mathcal{F}$.

4 THE PROPOSED MODEL

In this section, we present our designed dedicated neural network for serendipitous POI recommendation. Figure 2 depicts the architecture of SNPR. It mainly consists of four components: a) *Trajectory encoding*: it encodes each POI in every trajectory of a user u by Transformer encoder; b) *Unexpectedness-oriented modeling*: it models user u for the purpose of predicting the *unexpectedness* of candidate POIs to u (corresponding to the blue part in Figure 2); c) *Relevance-oriented modeling*: it models user u for the purpose of predicting the *relevance* of candidate POIs to u (corresponding to the yellow part in Figure 2); d) *Serendipitous POI recommendation and network training*: it generates top-K serendipitous POIs for user u according to the predicted *serendipity* of candidate POIs to u.

4.1 Trajectory Encoding

Given a user u, $\mathbf{T} = \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_n\}$ is her trajectory sequence. We first learn the optimal representations of all POIs (denoted by $\{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_{|\mathcal{T}_i|}\}$) in each of her trajectories $\mathcal{T}_i = \{l_1, l_2, ..., l_{|\mathcal{T}_i|}\}$ ($\mathcal{T}_i \in \mathbf{T}$) as illustrated in Figure 2. They are used as a basis for modeling *unexpectedness* and *relevance* of POIs to user u.

As discussed in Section 1, we utilize Transformer[39] to encode POIs rather than RNNs. The Transformer POI encoder does not require that the sequential check-in data are processed in order, and can capture the dependencies between representation pairs without regard to their distance in the sequences.

Here we start by describing a single layer of the Transformer POI encoder, and then discuss how to generalize it to multiple layers. One single layer contains two sub-layers, a multi-head self-attention layer and a position-wise feed-forward network.

Multi-head self-attention layer. Given a trajectory $\mathcal{T}_i$ of user u, $\mathbf{z}_j^l$ denotes the hidden representation of a POI $l_j \in \mathcal{T}_i$ ($1 \le j \le |\mathcal{T}_i|$) at layer l. The l-th layer computes the hidden representation $\mathbf{z}_j^{l+1}$ for each POI l_j at layer $l+1$. The multi-head self-attention sublayer let $\mathbf{z}_j^{l+1}$ be optimized by attending to the hidden representations of all POIs at layer l. We stack the hidden representation $\mathbf{z}_j^l \in \mathbb{R}^d$ together into matrix $\mathbf{Z}^l \in \mathbb{R}^{|\mathcal{T}_i| \times d}$ since we compute attention function on all positions simultaneously in practice. Please note that item embeddings are represented by row vectors in this paper following [39], although it's not common.

Multi-head attention projects $\mathbf{Z}^l$ into h subspaces (i.e., h heads) parallelly, which are concatenated and then linearly projected to obtain the final output of this sub-layer.

$$\mathrm{MH}(\mathbf{Z}^l) = \mathrm{Concat}(head_1, head_2, ..., head_h)\mathbf{W}^O \quad (7)$$

where $\mathbf{W}^O \in \mathbb{R}^{d \times d}$ is learnable parameters. $head_i \in \mathbb{R}^{|\mathcal{T}_i| \times d/h}$ is the ith attention head, which is computed as:

$$head_i = \mathrm{Attention}(\mathbf{Z}^l\mathbf{W}_i^Q, \mathbf{Z}^l\mathbf{W}_i^K, \mathbf{Z}^l\mathbf{W}_i^V) \quad (8)$$

$$\mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d/h}}\right)\mathbf{V} \quad (9)$$

where $\mathbf{W}_i^Q \in \mathbb{R}^{d \times d/h}$, $\mathbf{W}_i^K \in \mathbb{R}^{d \times d/h}$, $\mathbf{W}_i^V \in \mathbb{R}^{d \times d/h}$ are learnable parameters. The Attention function is *Scaled Dot-Product Attention*, where *query* $\mathbf{Q}$, *key* $\mathbf{K}$ and *value* $\mathbf{V}$ are all projected from the same matrix $\mathbf{Z}^l$ with different learned projection matrices $\mathbf{W}_i^Q$, $\mathbf{W}_i^K$ and $\mathbf{W}_i^V$. The scaling factor $\frac{1}{\sqrt{d/h}}$ is introduced to produce a softer attention distribution for avoiding extremely small gradients.

Position-wise feed-forward network. A position-wise feed-forward network to the outputs of the self-attention sub-layer is applied separately and identically at each position to endow the model with nonlinearity. It consists of two affine transformations with a Gaussian Error Linear Unit (GELU) activation in between:

$$\mathrm{FFN}(\mathbf{X}) = \mathrm{GELU}(\mathbf{X}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2 \quad (10)$$

$$\mathrm{GELU}(x) = x\Phi(x) \quad (11)$$

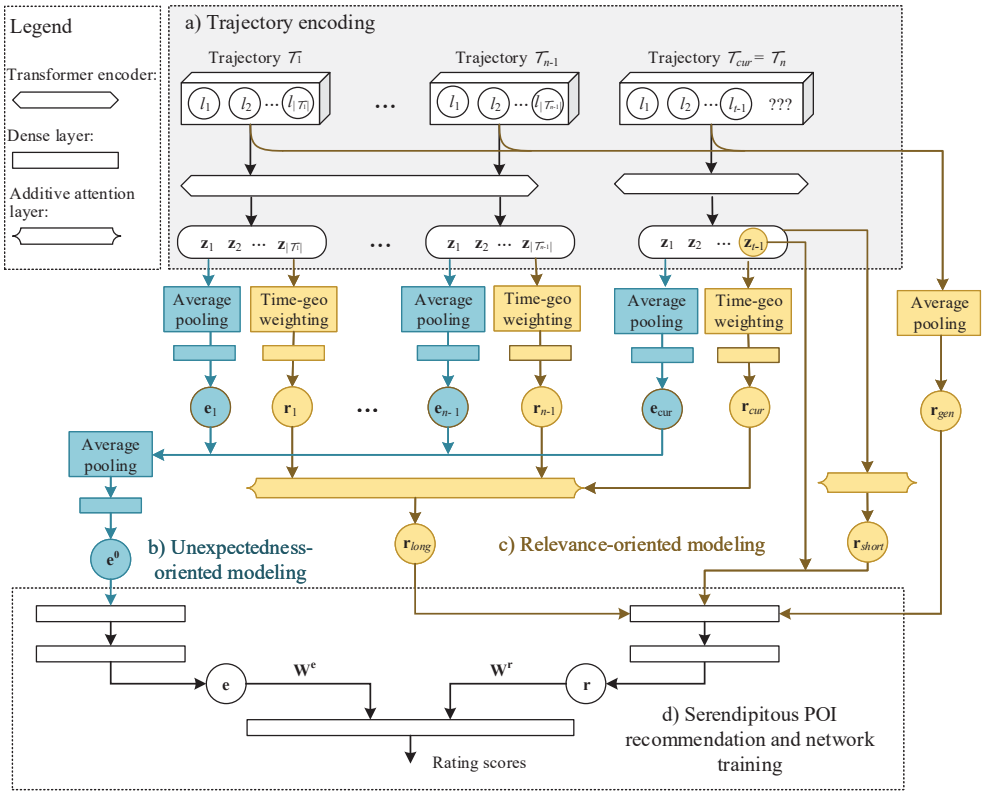

Figure 2: Overview of the neural network architecture of SNPR. $\mathbf{z}_*$ denotes the hidden representations of POIs. $\mathbf{e}_*$ denotes the hidden representations related to unexpectedness. $\mathbf{r}_*$ denotes the hidden representations related to relevance. $*$ is the asterisk wildcard.

where $\mathbf{X}$ is the output of the multi-head attention layer; $\Phi(x)$ is the cumulative distribution function of the standard Gaussian distribution; $\mathbf{W}_1 \in \mathbb{R}^{d\times 4d}$, $\mathbf{b}_1 \in \mathbb{R}^{4d}$, $\mathbf{W}_2 \in \mathbb{R}^{4d\times d}$, $\mathbf{b}_2 \in \mathbb{R}^{d}$ are learnable parameters and shared across all positions. In this work, we use a smoother GELU [13] activation rather than the standard ReLu activation following BERT [7].

To avoid overfitting and learn meaningful features hierarchically, we apply dropout [37] to the output of each of the two sub-layers (i.e., the multi-head self-attention sub-layer and the position-wise feed-forward network sub-layer). Moreover, we also employ a residual connection [12] around each sub-layer, followed by layer normalization [2].

Stacking Transformer layers. Recent work shows that different layers capture different types of features. Therefore, we stack the Transformer encoder layers to further model the complex POI relations underlying the trajectories. In summary, the layer-by-layer iterative computation method is defined as:

$$\mathbf{Z}^{l+1} = \mathrm{TE}(\mathbf{Z}^{l}) \forall l \in [1, 2, ...L] \tag{12}$$

$$\mathrm{TE}(\mathbf{Z}^{l}) = \mathrm{LN}(\mathbf{A}^{l} + \mathrm{Dropout}(\mathrm{FFN}(\mathbf{A}^{l}))) \tag{13}$$

$$\mathbf{A}^{l} = \mathrm{LN}(\mathbf{Z}^{l} + \mathrm{Dropout}(\mathrm{MH}(\mathbf{Z}^{l}))) \tag{14}$$

where TE denotes one Transformer encoder layer, LN denotes layer normalization function, and Dropout denotes dropout function. For each POI and each category, we first initialize their representations by uniform distribution. Then, we obtain the initial vector $\mathbf{z}^0$ of each POI l in a trajectory by concatenating its primal representation and the primal representation of its category. We stack them together to form the initial representation matrix $\mathbf{Z}^0$ of the trajectory. For the sake of efficiency, we only tried small L, i.e., $L = \{1, 2, 3\}$.

Given a trajectory $\mathcal{T}_i = \{l_1, l_2, ..., l_{|\mathcal{T}_i|}\}$ of a specific user u, the trajectory encoding component outputs the hidden representation $\mathbf{z}_j$ for each POI $l_j \in \mathcal{T}_i$. It is worth noting that trajectory $\mathcal{T}_{cur}$ is deployed a separate Transformer encoder to explicitly model the recently visited POIs, which will later serve the goal of short-term preference modeling.

4.2 Unexpectedness-oriented Modeling

Given a user u, this component models her profile "$u.prof$" to a vector $\mathbf{e}^0$ as depicted in Figure 2, with the purpose of predicting the *unexpectedness* of candidate POIs to u.

According to Definition. 2, POI unexpectedness to a user is irrelevant to the temporal and geographical information of her check-ins. Therefore, we simply represent each trajectory by an average pooling followed by a dense layer as:

$$\mathbf{e}_i^{avg} = \frac{1}{|\mathcal{T}_i|} \sum_{j=1}^{|\mathcal{T}_i|} \mathbf{z}_j \tag{15}$$

$$\mathbf{e}_i = \text{GELU}(\mathbf{e}_i^{avg}\mathbf{W}^{avg} + \mathbf{b}^{avg}) \tag{16}$$

where GELU is the activation function, $\mathbf{W}^{avg} \in \mathbb{R}^{d\times d}$, $\mathbf{b}^{avg} \in \mathbb{R}^d$ are learnable parameters.

we can obtain each of the trajectory representation $\mathbf{e}_i$ ($1 \le i \le cur$) for user u by Eq. (16). Then, we model the unexpectedness-oriented user vector (i.e., $\mathbf{e}^0$) for u by leveraging another average pooling to her trajectory embeddings $\{\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_{cur}\}$ followed by another dense layer. The equation is like Eq. (16) and we will not detail it anymore.

4.3 Relevance-oriented Modeling

Given a user u, this component aims to predict the *relevance* of candidate POIs to u. It models user profile to four vectors: the general preference $\mathbf{r}_{gen}$, the current preference $\mathbf{z}_{t-1}$, the long-term preference $\mathbf{r}_{long}$ and the short-term preference $\mathbf{r}_{short}$. We obtain the relevance-oriented user vector $\mathbf{r}^0$ by concatenating these four vectors, i.e., $\mathbf{r}^0 = \mathbf{r}_{gen}\|\mathbf{z}_{t-1}\|\mathbf{r}_{long}\|\mathbf{r}_{short}$, where "$\|$" denotes the concatenation operation. $\mathbf{r}_{gen}$ models the general behaviors of user u, and is computed simply by leveraging an average pooling on the initial representation of each POI. The last POI $\mathbf{z}_{t-1}$ has distinctly importance for next POI recommendation and is computed through the trajectory encoding component. Therefore, this component mainly consists of two modules: the user's long-term preference modeling and short-term preference modeling.

Long-term preference modeling. Temporal and geographical contextual information plays a key role for analyzing user behaviors, and is helpful for predicting where she will go next in the POI recommendation scenario [27, 38, 45]. Hence, the hidden representation of each trajectory $\mathcal{T}_i$ (denoted as $\mathbf{r}_i$) should incorporate such information to capture the time-sensitive property and geography-sensitive property. For each trajectory $\mathcal{T}_i$, we develop 1) a *time-weighted operation* and 2) a *geography-weighted operation* followed by 3) a *dense layer* to generate its relevance-oriented embedding $\mathbf{r}_i$. Then we model the long-term preference $\mathbf{r}_{long}$ of user u by 4) an *additive attention layer* on all trajectory embeddings $\{\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_{cur}\}$ of u.

(1) *Time-weighted operation.* Specifically, we map one day into 6 time slots (i.e., every 4 hours a slot, from $(22:00 \mapsto 02:00)$ to $(18:00 \mapsto 22:00)$) and one week into $6 \times 7 = 42$ time slots. For each slot i, we construct a vector $\mathcal{N}^i = (n_1^i, n_2^i, ..., n_{|\mathcal{L}|}^i)$ where n_k^i ($k = 1, 2, ..., |\mathcal{L}|$) is the number of check-ins of POI l_k in time slot i. Then, we utilize cosine distance to measure the temporal similarity $sim_{i,j}^{\tau}$ between the i-th and j-th time slots as:

$$sim_{i,j}^{\tau} = \frac{\mathcal{N}^i \cdot \mathcal{N}^j}{\|\mathcal{N}^i\|\|\mathcal{N}^j\|} = \frac{\sum_{k=1}^{|\mathcal{L}|} n_k^i \times n_k^j}{\sqrt{\sum_{k=1}^{|\mathcal{L}|} {n_k^i}^2}\sqrt{\sum_{k=1}^{|\mathcal{L}|} {n_k^j}^2}} \tag{17}$$

The resulting similarity ranges from 0 (meaning that two time slots don't have any POI which has check-ins in both slots), to 1 (meaning that two slots exactly have the same check-in numbers on each POI).

Given a trajectory $\mathcal{T}_i$ of user u, we can derive a sequence of check-in time slots $(\tau_1, \tau_2, ..., \tau_{|\mathcal{T}_i|})$ according to the check-in time of each POI in $\mathcal{T}_i$, where $\tau_i \in \{1, 2, ..., 42\}$ ($1 \le i \le |\mathcal{T}_i|$). With the target user's current time slot τ_{cur}, the time-weighted representation $\mathbf{r}_i^{\tau}$ of $\mathcal{T}_i$ is calculated as:

$$\mathbf{r}_i^{\tau} = \sum_{t=1}^{|\mathcal{T}_i|} w_t \mathbf{z}_t, w_t = \frac{\exp\left(sim_{\tau_{cur},\tau_t}^{\tau}\right)}{\sum_{j=1}^{|\mathcal{T}_i|}\exp\left(sim_{\tau_{cur},\tau_j}^{\tau}\right)} \tag{18}$$

where $sim_{\tau_{cur},\tau_j}^{\tau}$ is the temporal similarity between the current time slot τ_{cur} and the time slot of the j-th visited POI in $\mathcal{T}_i$.

(2) *Geography-weighted operation.* Intuitively, for next-POI recommendation, the target user's next move will be highly dependent on her current geographic location. Thus, the geography-weighted representation $\mathbf{r}_i^{geo}$ for each trajectory $\mathcal{T}_i$ of user u is calculated based on the geographical similarity between the current location l_{cur} and the POI locations in $\mathcal{T}_i$ as:

$$\mathbf{r}_i^{geo} = \sum_{t=1}^{|\mathcal{T}_i|} w_t \mathbf{z}_t, w_t = \frac{\exp\left(sim_{l_{cur},l_t}^{geo}\right)}{\sum_{j=1}^{|\mathcal{T}_i|}\exp\left(sim_{l_{cur},l_j}^{geo}\right)} \tag{19}$$

$$sim_{l_{cur},l_j}^{geo} = \frac{1}{\sqrt{(lon_{l_{cur}} - lon_{l_j})^2 + (lat_{l_{cur}} - lat_{l_j})^2}} \tag{20}$$

where sim_{l_{cur},l_j}^{geo} is the similarity between the current location l_{cur} (i.e., l_{t-1}) and the location l_j in $\mathcal{T}_i$. It's modeled as the reciprocal of the geographical distance between these two locations.

(3) *Dense layer for relevance-oriented trajectory representation.* So far, we can obtain the time-weighted representation $\mathbf{r}_i^{\tau}$ and the geography-weighted representation $\mathbf{r}_i^{geo}$ for each trajectory $\mathcal{T}_i$ of user u. Then, a dense layer is applied to endow the model with nonlinearity as:

$$\mathbf{r}_i = \text{GELU}((\mathbf{r}_i^{\tau}\|\mathbf{r}_i^{geo})\mathbf{W}^{den} + \mathbf{b}^{den}) \tag{21}$$

where $\mathbf{W}^{den} \in \mathbb{R}^{2d\times d}$, $\mathbf{b}^{den} \in \mathbb{R}^d$ are learnable parameters.

(4) *Additive attention layer for long-term preference representation.* The long-term preference of a user u can be clearly revealed by their historical trajectories. Besides, different trajectories may have varied importance for modeling user interests. Thus, we apply an additive attention mechanism to aggregate relevance-oriented trajectory embeddings (i.e., $\{\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_{cur}\}$) for user long-term preference modeling as:

$$\mathbf{r}_{long} = \sum_{i=1}^{cur-1} \alpha_i \mathbf{r}_i \tag{22}$$

$$\alpha_i = \frac{\exp\left(tanh\left((\mathbf{r}_{cur}\|\mathbf{r}_i)\mathbf{W}_a^{long}\right)\mathbf{q}_a^{long}\right)}{\sum_{j=1}^{cur-1}\exp\left(tanh\left((\mathbf{r}_{cur}\|\mathbf{r}_j)\mathbf{W}_a^{long}\right)\mathbf{q}_a^{long}\right)} \tag{23}$$

where $\mathbf{W}_a^{long} \in \mathbb{R}^{2d\times d}$, $\mathbf{q}_a^{long} \in \mathbb{R}^d$ are learnable parameters.

In summary, the comprehensive long-term preference representation $\mathbf{t}_{long}$ considers the temporal and geographical associations between each historical POIs and the current POI while calculating each trajectory embeddings. Furthermore, it considers different importance of historical trajectories according to the last trajectory representation by an additive attention layer.

Short-term preference modeling. Comparing with historical trajectories, the current trajectory $\mathcal{T}_{cur}$ is distinctly important for the next POI recommendation. Therefore, we only exploit the POI embeddings $\{\mathbf{z}_1, \mathbf{z}_1, ..., \mathbf{z}_{t-1}\}$ of $\mathcal{T}_{cur}$ obtained in Section 4.1 to

model user's short-term preference $\mathbf{r}_{short}$. It is calculated also by applying an additive attention mechanism. The computing equations are similar to Eq. (22) and Eq. (23), and not detailed any more.

4.4 Serendipitous POI Recommendation and Network Training

Given a user u, we can obtain her unexpectedness-oriented representation $\mathbf{e}^0$ in Section 4.2 and relevance-oriented representation $\mathbf{r}^0$ in Section 4.3. Then, we put them into a two-layer fully connected neural network to obtain their final representations $\mathbf{e}$ and $\mathbf{r}$ respectively. At last, we compute the rating score distribution $\mathbf{serendipity}^*(u)$ over the $|\mathcal{L}|$ POIs of user u via the following:

$$\mathbf{serendipity}^*(u) = \lambda \times \text{softmax}(\mathbf{r}\mathbf{W}^{\mathbf{r}} + \mathbf{b}^{\mathbf{r}}) \oplus (1-\lambda) \times \text{sigmoid}(\mathbf{e}\mathbf{W}^{\mathbf{e}} + \mathbf{b}^{\mathbf{e}}) \tag{24}$$

where $\oplus$ denotes the element-wise addition operation. $\lambda \in [0, 1]$ is the relevance-unexpectedness trade-off factor. $serendipity^*(l_c, u) \in \mathbf{serendipity}^*(u)$ ($l_c \in \mathcal{L}$) denotes the predicted *serendipity* score of candidate POI l_c to user u. The model will recommend the top-K predicted serendipitous POIs to the target user.

During training, we first calculate the serendipity ground truth according to the model defined in Section 3.1. Then, we define the objective function as the squared error with L2 norm regularization to train the network:

$$\min_{\theta} \sum_{u \in \mathcal{U}} \sum_{l_c \in Trn^u} \|serendipity^*(l_c, u) - serendipity(l_c, u)\|_2^2 + \mu \|\theta\|_2^2 \tag{25}$$

where θ denotes all the parameters contained in the network model. Trn^u is the set of u's all training instances.

5 EXPERIMENTS

We conduct comprehensive experiments on real-world datasets to comparatively evaluate the effectiveness of the proposed model—SNPR.

5.1 Experimental Setup

5.1.1 Datasets and Data Preprocessing. As category information of POIs is indispensable in our model, some classic POI recommendation datasets like Gowalla can't be adopted. A series of datasets[1] were collected from one of the most popular LBSNs—Foursquare by Yang et al.. We extracted two datasets from them for our experimental evaluation. One was the New York dataset collected from April 2012 to February 2013 [41]. The other was the United Kingdom dataset (denoted as UK) collected from April 2012 to September 2013[40].

We preprocessed both of the datasets. For POIs, we eliminated the unpopular ones that were visited by less than 10 users. For users, we treated their all check-ins in one day as a single trajectory as illustrated in Section 3.2. Then, we removed trajectories having less than three check-ins, and filtered out the inactive users with less than 5 trajectories. Basic statistics of both datasets after preprocessing are shown in Table 2.

[1] https://sites.google.com/site/yangdingqi/home/foursquare-dataset

Table 2: Statistics of datasets.

Item	Statistic	
	New York	United Kingdom
# users	637	315
# POIs	4601	1827
# check-ins	69709	12788
# trajectories	14348	3114
spasity	2.378%	2.222%

5.1.2 Metrics. We first adopted two widely-used metrics for *relevance* evaluation: *Precision@K* and Normalized Discounted Cumulative Gain (*NDCG@K*), where K indicates recommending top-K ranked POIs. *Precision@K* don't care about rank position in the recommendation list, while *NDCG@K* is a full position-aware metric which assigns larger weights on higher positions.

Then, we define a metric *UNE@K* for *unexpectedness* evaluation as follows.

$$UNE@K = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \sum_{i=1}^{K} unexp(l_i, u) \tag{26}$$

where $unexp(l_i, u)$ is the unexpectedness of the i-th recommended POI to user u, and is calculated according to Eq. (2).

Finally, we defined a metric *SER@K* to compare the recommendation performance on *serendipity* comprehensively by adopting F-measure and min-max scaling to balance *relevance* and *unexpectedness* evaluation as follows.

$$SER@K = \frac{2 * \text{MM}(NDCG@K) * \text{MM}(UNE@K)}{\text{MM}(NDCG@K) + \text{MM}(UNE@K)} \tag{27}$$

$$\text{MM}(x) = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{28}$$

where *NDCG@K* and *UNE@K* are typical metrics for *relevance* and *unexpectedness* respectively. *NDCG@K* is adopted to consider *relevance* factor because it's more comprehensive than *Precision@K*. MM is the min-max scaling function. It is adopted to solve the issue that different metrics usually have great different ranges.

For all of the above metrics, higher values indicate better recommendation performance.

5.1.3 Baselines and Implementation Details. To evaluate the performance of the proposed model, we selected the following six recommendation models: two naive baselines commonly used for the evaluation of serendipity-oriented recommendation methods; one vanilla and two state-of-the-art serendipity-oriented recommendation models; and one state-of-the-art accuracy-oriented POI recommendation models.

- **POP**. Popularity-based recommendation method generates results according to the prevalence of POIs, where we adopt the number of Check-ins that POIs have to measure their popularity.
- **RAND**. Random-based method recommends POIs randomly among the candidates that the target user hasn't visited.
- **KFN** [36]. It is a vanilla serendipity-oriented recommendation method. Starting with the core idea of kNN and turning

Table 3: The performance comparison of all methods. The best results are starred, and the second-best results are listed in bold. *P@K, N@K, U@K* and *S@K* are the abbreviations for *Presion@K, NDCG@K, UNE@K* and *SER@K*. All the results are expressed as percentages (%).

Datasets		New York								United Kingdom							
top-K	Metrics	Pop	RAND	KFN	HAES	DESR	LSTPM	SNPR^{NU}	SNPR	Pop	RAND	KFN	HAES	DESR	LSTPM	SNPR^{NU}	SNPR
3	P@K	0.675	0.026	0.261	0.853	0.111	**8.237**	**8.580***	5.695	1.353	0.053	0.162	0.506	0.107	**5.080**	**5.863***	4.346
	N@K	1.494	0.039	0.697	1.886	0.181	**20.34**	**21.42***	12.02	2.942	0.100	0.338	1.293	0.169	**12.19**	**14.29***	10.21
	U@K	56.02	**65.17**	63.64	35.14	58.60	23.00	35.02	**73.34***	35.10	**50.23**	45.49	36.85	38.62	15.12	22.14	**60.68***
	S@K	12.33	0.000	5.928	12.72	1.316	0.000	**38.56**	**71.83***	**27.50**	0.000	3.271	14.29	0.963	0.000	26.69	**83.19***
5	P@K	0.510	0.035	0.214	0.627	0.067	**5.723**	**5.970***	5.325	1.418	0.048	0.193	0.335	0.071	3.635	**3.899***	**3.842**
	N@K	1.706	0.081	0.814	2.120	0.181	**21.95**	**23.13***	15.94	4.179	0.134	0.522	1.355	0.182	13.40	**15.08***	**14.74**
	U@K	56.90	**65.29**	63.48	35.18	58.66	24.76	38.55	**74.20***	34.41	**50.01**	45.78	37.43	38.24	15.18	22.51	**61.43***
	S@K	12.72	0.000	6.111	12.46	0.862	0.000	**43.61**	**81.53***	**32.78**	0.000	4.995	13.97	0.638	0.000	27.36	**98.84***
10	P@K	0.348	0.018	0.160	0.432	0.046	3.406	**3.443**	**4.829***	1.057	0.024	0.100	0.267	0.042	2.274	**2.342**	**3.651***
	N@K	2.017	0.083	0.982	2.495	0.218	23.72	**24.62***	**24.35**	5.287	0.134	0.532	1.672	0.202	14.86	**16.34**	**17.04***
	U@K	59.43	**65.18**	63.62	35.05	58.91	26.70	43.41	**74.32***	34.74	**49.56**	45.46	38.32	38.76	15.66	23.98	**62.60***
	S@K	14.14	0.000	6.997	12.60	1.092	0.000	**51.97**	**99.44***	**34.85**	0.000	4.541	15.32	0.798	0.000	29.94	**1.000***
20	P@K	0.228	0.025	0.143	0.350	0.034	1.899	**1.904**	**3.425***	0.834	0.067	0.066	0.200	0.049	1.462	**1.469**	**2.502***
	N@K	2.301	0.159	1.299	3.177	0.275	24.71	**25.52**	**27.03***	6.829	0.413	0.611	2.012	0.350	16.50	**17.80**	**20.54***
	U@K	61.06	**65.19**	63.62	35.19	58.98	28.36	49.50	**74.24***	37.67	**49.46**	45.60	38.49	38.53	16.78	25.46	**61.05***
	S@K	14.34	0.000	8.042	12.81	0.858	0.000	**61.92**	**1.000***	**38.21**	0.622	2.536	14.10	0.000	0.000	31.98	**1.000***

kNN inside out, this approach creates neighbors of maximally dissimilar users and then recommends items that a user's neighbors are most likely to dislike.

- **HAES** [24]. It's a state-of-the-art serendipity-oriented recommendation model. HAES defines a novel concept of elasticity both for users and items, to adjust the level of serendipity flexibly and reach a trade-off between accuracy and serendipity.
- **DESR** [25]. It's another state-of-the-art serendipity-oriented recommendation model. DESR trys to combine user-item relevance with user preference direction to recommend serendipitous items, by managing to infer users' various long-term preferences and short-term demands as the basis of user preference direction.
- **LSTPM** [38]. It's a state-of-the-art general next POI recommendation model. LSTPM develops a context-aware nonlocal network structure to model users' long-term preferences and a geo-dilated RNN to model users' short-term preferences.
- **SNPR-NU**. SNPR-No-Unexpectedness, is a simplified version of our model without the component for unexpectedness modeling, which recommended POIs only considering relevance.

For each user, we held the first 80% of check-ins as the training set and the next 20% of check-ins as the test set for reporting model performance. The hyperparameters are set as follows. The embedding size of POIs is 64, the numbers of Transformer encoder layers and heads are both 2, the trade-off factor γ between category difference and POI difference is 0.5, the *relevance-unexpectedness* trade-off factor λ is 0.7, epoch is 30, and learning rate is 0.0001. All the experimental results of our model are achieved by using the above hyperparameter configuration settings if no specific situations are provided. For all the baselines, we set respective optimal parameters either according to corresponding references or based on our experiment results. We adopted Adam optimizer to train the models.

5.2 Overall Comparison

The experimental results of all the methods are illustrated in Table 3. We have the following observations.

(1) On the relevance-related metrics (i.e., *Presion@K*, *NDCG@K*), our model SNPR, our simplified model SNPR^{NU} and the general state-of-the-art next POI recommendation model LSPTM always yield the top three performance and SNPR^{NU} always outperforms LSPTM whatever the size of recommendation list K is. Specifically, when K is 3, SNPR^{NU} ranks the first, LSTPM ranks the second, and SNPR ranks the third. When K is 5, SNPR^{NU} still ranks the first, LSTPM ranks the second on the New York dataset, and SNPR ranks the second on the United Kingdom dataset. When K is 10, SNPR ranks the first expect on *NDCG@K* on the New York dataset, and LSTPM all ranks the third. When K is 20, SNPR ranks the first, SNPR^{NU} ranks the second, and LSTPM ranks the third. From the above analysis, we can find an interesting conclusion that the objective of *unexpectedness* can benefit the objective of *relevance* when the size of recommendation list is larger than 10. This may be because users tend to find novel POIs in their check-in data, and more relevant POIs can be involved by considering *unexpectedness* while K is large. Comparing with serendipity-oriented methods, SNPR can have an order of magnitude improvement on *Presion@K* and *NDCG@K*.

(2) On the unexpectedness metric *UNE@K*, our model SNPR achieves the best performance and the state-of-the-art general next

Table 4: Influence of key hyperparameters (i.e., the relevance-unexpectedness trade-off factor λ, the embedding size of POIs, the number of heads of POI encoder, and the number of layers of POI encoder) on the performance of our model SNPR.

Datasets	λ	P@K	N@K	U@K	size	P@K	N@K	U@K	heads	P@K	N@K	U@K	layers	P@K	N@K	U@K
New York	0	0.024	0.086	**74.32***	16	3.670	14.72	70.24	1	**4.933***	**24.71***	74.30	1	4.561	22.33	**74.33**
	0.25	4.177	17.45	69.42	32	4.120	17.20	70.28	2	**4.829**	**24.35**	**74.32**	2	**4.829***	**24.35***	74.32
	0.5	4.619	22.56	72.15	64	**4.829***	**24.35***	**74.32***	4	4.497	21.77	74.30	3	**4.743**	**24.19**	**74.33***
	0.75	**4.713**	**23.45**	**74.25**	128	4.334	19.11	70.64	8	4.492	22.48	74.30				
	1	**4.799***	**23.82***	62.17	256	**4.662**	**20.84**	**72.31**	16	4.711	23.31	**74.33***				
United Kingdom	0	0.032	0.123	**62.50**	16	1.592	5.507	60.01	1	2.734	10.61	61.53	1	2.364	8.446	**62.33**
	0.25	2.335	9.367	62.01	32	2.175	8.061	**62.40**	2	**3.651**	**17.04**	**62.56**	2	**3.651***	**17.04***	**62.56***
	0.5	**3.280**	**17.24***	**63.50***	64	**3.651***	**17.04***	**62.56***	4	3.119	15.17	61.65	3	**2.958**	**12.21**	59.46
	0.75	**3.596***	**14.66**	61.48	128	3.306	12.58	57.83	8	**3.992***	**19.66***	**62.70***				
	1	3.198	13.98	38.09	256	**3.483**	**15.51**	60.76	16	3.207	13.25	60.47				

POI recommendation model LSTPM yields the lowest performance. The naive method RAND achieves the second-best performance, but it yields the lowest performance on relevance-related metrics in most cases.

(3) On the serendipity metric *SER@K*, Our model consistently achieves the best performance, and our simplified model SNPR-NU achieves the second-best performance on the New York dataset. The *SER@K* values of RAND and LSTPM are 0 in most cases because they yields the lowest performance on relevance and on unexpectedness respectively. It demonstrates that SNPR can improve the *serendipity* for next POI recommendation significantly compared with no matter the state-of-the-art serendipity-oriented models and general next POI recommendation models.

5.3 Effects of Hyperparameters

To get deep insights on the proposed model SNPR, we investigated its sensitivity on some core hyperparameters. Table 4 lists the experimental results when K=10. From it we can observe that:

(1) The hyperparameter values with the best performance are not all the same on different datasets. So, we analyze the effects of hyperparameters averagely on the two datasets.

(2) The performance on relevance decreases sharply when the relevance-unexpectedness trade-off factor λ changes from 0.25 to 0, and the performance on unexpectedness decreases obviously when λ changes from 0.75 to 1. Thus, it's reasonable to tune λ from 0.25 to 0.75. We set $\lambda = 0.7$ in other experiments.

(3) The performance on *UNE@K* changes slightly over different POI embedding size, number of heads of POI encoder, and number of layers of POI encoder. The performance on relevance also changes slightly over different numbers of heads and layers of POI encoder. However, it improves obviously when the embedding size inceases from 16 to 64. Then, it changes slightly. Thus, we set the embedding size to 64, the number of heads to 2 and the number of layers to 2 as default values, which are relatively optimal.

5.4 Ablation Analysis

To assess the effectiveness of Transformer POI encoder of SNPR, we conducted an ablation test by comparing with the LSTM POI encoder. Figure 3 presents the results of the ablation test on the three metrics: *Precision@K*, *NDCG@K* and *UNE@K*. From it we can see that the performance of SNPR based on Transformer POI encoder is better than the performance based on LSTM POI encoder no matter on which metric and dataset. Specifically, the improvements on relevance-related metrics (i.e., *Precision@K* and *NDCG@K*) are more obvious than on unexpectedness-related metric (i.e., *UNE@K*). These results show that Transformer encoder can capture the complex interdependencies between POIs and improve recommendation performance especially on relevance.

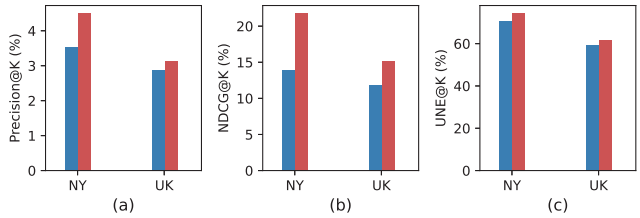

Figure 3: Ablation analysis on POI encoder. NY denotes the New York dataset, UK denotes the United Kingdom dataset. The blue bar denotes the LSTM POI encoder, and the red bar denotes the Transformer POI encoder.

6 CONCLUSION

This paper has introduced serendipity to next POI recommendation such that the recommendations are ensured to satisfy user's preference evidenced by the past behaviours and user's curiosity to explore new locations. It overcomes the common issue of existing solutions in the field of next POI recommendation solutions which concerns the user's preference only and thus makes tedious recommendations. The extensive experiments evidence the effectiveness of the proposed SNPR model in next POI recommendation with consideration of relevance and unexpectedness at the same time following the quantitative serendipity. By adjusting relevance-unexpectedness trade-off hyperparameter, the proposed solution allows the next POI recommendation bias to relevance or unexpectedness flexibly.

7 ACKNOWLEDGMENTS

This work was supported by Australian Research Council Linkage Project (No. LP180100750) and Discovery Project (No. DP210100743).

REFERENCES

[1] Panagiotis Adamopoulos and Alexander Tuzhilin. 2014. On Unexpectedness in Recommender Systems: Or How to Better Expect the Unexpected. *ACM Trans. Intell. Syst. Technol.* 5, 4 (2014), 54:1–54:32.

[2] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer Normalization. *CoRR* abs/1607.06450 (2016).

[3] Li Chen, Yonghua Yang, Ningxia Wang, Keping Yang, and Quan Yuan. 2019. How Serendipity Improves User Satisfaction with Recommendations? A Large-Scale User Evaluation. In *Proceedings of WWW*. 240–250.

[4] Chen Cheng, Haiqin Yang, Michael R. Lyu, and Irwin King. 2013. Where You Like to Go Next: Successive Point-of-Interest Recommendation. In *Proceedings of IJCAI*. 2605–2611.

[5] Marco de Gemmis, Pasquale Lops, Giovanni Semeraro, and Cataldo Musto. 2015. An investigation on the serendipity problem in recommender systems. *Inf. Process. Manag.* 51, 5 (2015), 695–717.

[6] Marco de Gemmis, Pasquale Lops, Giovanni Semeraro, and Cataldo Musto. 2015. An investigation on the serendipity problem in recommender systems. *Inf. Process. Manag.* 51, 5 (2015), 695–717.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. [n.d.]. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT*. 4171–4186.

[8] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. 2018. DeepMove: Predicting Human Mobility with Attentional Recurrent Networks. In *Proceedings of WWW*. 1459–1468.

[9] Qing Guo, Zhu Sun, Jie Zhang, and Yin-Leng Theng. 2020. An Attentional Recurrent Neural Network for Personalized Next Location Recommendation. In *Proceedings of AAAI*. 83–90.

[10] Peng Han, Zhongxiao Li, Yong Liu, Peilin Zhao, Jing Li, Hao Wang, and Shuo Shang. 2020. Contextualized Point-of-Interest Recommendation. In *Proceedings of IJCAI*. 2484–2490.

[11] Jing He, Xin Li, Lejian Liao, Dandan Song, and William K. Cheung. 2016. Inferring a Personalized Next Point-of-Interest Recommendation Model with Latent Behavior Patterns. In *Proceedings of AAAI*. 137–143.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of CVPR*. 770–778.

[13] Dan Hendrycks and Kevin Gimpel. 2016. Bridging Nonlinearities and Stochastic Regularizers with Gaussian Error Linear Units. *CoRR* abs/1606.08415 (2016).

[14] Leo Iaquinta, Marco de Gemmis, Pasquale Lops, Giovanni Semeraro, Michele Filannino, and Piero Molino. 2008. Introducing Serendipity in a Content-Based Recommender System. In *Proceedings of HIS*. 168–173.

[15] D. Kaminskas, M. Bridge. 2014. Measuring surprise in recommender systems. In *the Workshop on Recommender Systems Evaluation: Dimensions and Design*. 1–6.

[16] Marius Kaminskas and Derek Bridge. 2017. Diversity, Serendipity, Novelty, and Coverage: A Survey and Empirical Analysis of Beyond-Accuracy Objectives in Recommender Systems. *ACM Trans. Interact. Intell. Syst.* 7, 1 (2017), 2:1–2:42.

[17] Aleksandra Karpus, Iacopo Vagliano, and Krzysztof Goczyla. 2017. Serendipitous Recommendations Through Ontology-Based Contextual Pre-filtering. In *Proceedings of BDAS*. 246–259.

[18] Samira Khoshahval, Mahdi Farnaghi, Mohammad Taleai, and Ali Mansourian. 2018. A Personalized Location-Based and Serendipity-Oriented Point of Interest Recommender Assistant Based on Behavioral Patterns. In *Proceedings of AGILE*. 271–289.

[19] Denis Kotkov, Jari Veijalainen, and Shuaiqiang Wang. 2016. Challenges of Serendipity in Recommender Systems. In *Proceedings of WEBIST*. 251–256.

[20] Denis Kotkov, Jari Veijalainen, and Shuaiqiang Wang. 2020. How does serendipity affect diversity in recommender systems? A serendipity-oriented greedy algorithm. *Computing* 102, 2 (2020), 393–411.

[21] Denis Kotkov, Shuaiqiang Wang, and Jari Veijalainen. 2016. A survey of serendipity in recommender systems. *Knowl. Based Syst.* 111 (2016), 180–192.

[22] Denis Kotkov, Shuaiqiang Wang, and Jari Veijalainen. 2016. A survey of serendipity in recommender systems. *Knowl. Based Syst.* 111 (2016), 180–192.

[23] Ranzhen Li, Yanyan Shen, and Yanmin Zhu. 2018. Next Point-of-Interest Recommendation with Temporal and Multi-level Context Attention. In *Proceedings of ICDM*. 1110–1115.

[24] Xueqi Li, Wenjun Jiang, Weiguang Chen, Jie Wu, and Guojun Wang. 2019. HAES: A New Hybrid Approach for Movie Recommendation with Elastic Serendipity. In *Proceedings of CIKM*. 1503–1512.

[25] Xueqi Li, Wenjun Jiang, Weiguang Chen, Jie Wu, Guojun Wang, and Kenli Li. 2020. Directional and Explainable Serendipity Recommendation. In *Proceedings of WWW*. 122–132.

[26] Zhenhui Li, Bolin Ding, Jiawei Han, Roland Kays, and Peter Nye. 2010. Mining periodic behaviors for moving objects. In *Proceedings of SIGKDD*. 1099–1108.

[27] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the Next Location: A Recurrent Model with Spatial and Temporal Contexts. In *Proceedings of AAAI*. 194–200.

[28] Valentina Maccatrozzo, Manon Terstall, Lora Aroyo, and Guus Schreiber. 2017. SIRUP: Serendipity In Recommendations via User Perceptions. In *Proceedings of IUI*. 35–44.

[29] Jarana Manotumruksa, Craig Macdonald, and Iadh Ounis. 2018. A Contextual Attention Recurrent Architecture for Context-Aware Venue Recommendation. In *Proceedings of SIGIR*. 555–564.

[30] Anna Monreale, Fabio Pinelli, Roberto Trasarti, and Fosca Giannotti. 2009. WhereNext: a location predictor on trajectory pattern mining. In *Proceedings of SIGKDD*. 637–646.

[31] Xi Niu. 2018. An Adaptive Recommender System for Computational Serendipity. In *Proceedings of SIGIR*. 215–218.

[32] Kensuke Onuma, Hanghang Tong, and Christos Faloutsos. 2009. TANGENT: a novel, 'Surprise me', recommendation algorithm. In *Proceedings of SIGKDD*. 657–666.

[33] Gaurav Pandey, Denis Kotkov, and Alexander Semenov. 2018. Recommending Serendipitous Items using Transfer Learning. In *Proceedings of CIKM*. 1771–1774.

[34] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of WWW*. 811–820.

[35] Francesco Ricci, Lior Rokach, and Bracha Shapira (Eds.). 2015. *Recommender Systems Handbook*. Springer.

[36] Alan Said, Ben Fields, Brijnesh J. Jain, and Sahin Albayrak. 2013. User-centric evaluation of a K-furthest neighbor collaborative filtering recommender algorithm. In *Proceedings of CSCW*. 1399–1408.

[37] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 15, 1 (2014), 1929–1958.

[38] Ke Sun, Tieyun Qian, Tong Chen, Yile Liang, Quoc Viet Hung Nguyen, and Hongzhi Yin. 2020. Where to Go Next: Modeling Long- and Short-Term User Preferences for Point-of-Interest Recommendation. In *Proceedings of AAAI*. 214–221.

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Proceedings of NIPS*. 5998–6008.

[40] Dingqi Yang, Daqing Zhang, and Bingqing Qu. 2016. Participatory Cultural Mapping Based on Collective Behavior Data in Location-Based Social Networks. *ACM Trans. Intell. Syst. Technol.* 7, 3 (2016), 30:1–30:23.

[41] Dingqi Yang, Daqing Zhang, Vincent W. Zheng, and Zhiyong Yu. 2015. Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs. *IEEE Trans. Syst. Man Cybern. Syst.* 45, 1 (2015), 129–142.

[42] Yongjian Yang, Yuanbo Xu, En Wang, Jiayu Han, and Zhiwen Yu. 2018. Improving Existing Collaborative Filtering Recommendations via Serendipity-Based Algorithm. *IEEE Trans. Multimedia* 20, 7 (2018), 1888–1900.

[43] Yuan Cao Zhang, Diarmuid Ó Séaghdha, Daniele Quercia, and Tamas Jambor. 2012. Auralist: introducing serendipity into music recommendation. In *Proceedings of WSDM*. 13–22.

[44] Pengpeng Zhao, Haifeng Zhu, Yanchi Liu, Jiajie Xu, Zhixu Li, Fuzhen Zhuang, Victor S. Sheng, and Xiaofang Zhou. 2019. Where to Go Next: A Spatio-Temporal Gated Network for Next POI Recommendation. In *Proceedings of AAAI*. 5877–5884.

[45] Pengpeng Zhao, Haifeng Zhu, Yanchi Liu, Jiajie Xu, Zhixu Li, Fuzhen Zhuang, Victor S. Sheng, and Xiaofang Zhou. 2019. Where to Go Next: A Spatio-Temporal Gated Network for Next POI Recommendation. In *Proceedings of AAAI*. 5877–5884.