Geography-Aware Sequential Location Recommendation

Defu Lian[1,3], Yongji Wu[1], Yong Ge[4], Xing Xie[5], Enhong Chen[1,2,3*]
[1]School of Computer Science and Technology, University of Science and Technology of China
[2]Institute of Smart City Research (Wuhu), University of Science and Technology of China
[3]School of Data Science, University of Science and Technology of China
[4]University of Arizona and [5]Microsoft Research Asia
{liandefu,cheneh}@ustc.edu.cn,{wuyongji317,ygestrive}@gmail.com,xingx@microsoft.com

ABSTRACT

Sequential location recommendation plays an important role in many applications such as mobility prediction, route planning and location-based advertisements. In spite of evolving from tensor factorization to RNN-based neural networks, existing methods did not make effective use of geographical information and suffered from the sparsity issue. To this end, we propose a Geography-aware sequential recommender based on the Self-Attention Network (GeoSAN for short) for location recommendation. On the one hand, we propose a new loss function based on importance sampling for optimization, to address the sparsity issue by emphasizing the use of informative negative samples. On the other hand, to make better use of geographical information, GeoSAN represents the hierarchical gridding of each GPS point with a self-attention based geography encoder. Moreover, we put forward geography-aware negative samplers to promote the informativeness of negative samples. We evaluate the proposed algorithm with three real-world LBSN datasets, and show that GeoSAN outperforms the state-of-the-art sequential location recommenders by 34.9%. The experimental results further verify significant effectiveness of the new loss function, geography encoder, and geography-aware negative samplers.

CCS CONCEPTS

• **Information systems** → **Collaborative filtering**; **Location based services**.

KEYWORDS

sequential recommendation, geography encoding, importance sampling, self-attention, location recommendation

1 INTRODUCTION

With the rapid development of information technology, it is much easier for human mobility behaviors to digitize and share with friends. Mobility behaviors can be used to understand and predict human mobility [9, 33], facilitating individual daily life in dining, transportation, entertainment, and so on. However, individual mobility is not always predictable [19, 28], due to data missing and sparsity. As the task of predicting a personalized ranking of locations given individual mobility history, sequential location recommendation plays an important role in improving the predictability of mobility over unseen locations, by exploiting the wisdom of the crowd. In addition to mobility prediction, sequential location recommendation is also useful in many other applications ranging from route planning to location-based advertisements.

The methods of sequential location recommendation has been evolving from tensor factorization and metric learning to RNN/CNN based neural networks in recent years. For example, Factorizing Personalized Markov Chains (FPMC) [26] were extended to address the sparsity issue of modeling personalized location transitions [3, 18]. Based on metric learning, Personalized Ranking Metric Embedding (PRME) was proposed to model personalized location transition [8], and further extended to incorporate geographical influence by multiplying travel distance with the estimated transition probability. To capture long-term dependence, Recurrent Neural Networks such as GRU and LSTM were extended to incorporate spatio-temporal information [5, 13, 22, 34, 40] by embedding travel distance, travel time and time of the week, or designing spatio-temporal gates for controlling information flow.

Among these existing methods, two important challenges are not well addressed. First, geographical information is still not effectively utilized. It is well known that the GPS position of location is important to describe physical proximity between locations and individual mobility history usually exhibits the spatial clustering phenomenon [16, 36]. Therefore, it is indispensable to encode the exact GPS positions of locations. Second, these methods may suffer from the sparsity issue. Note that users usually visit a small number of distinct locations [27], and negatively-preferred locations are mixed together with potentially positive ones in individual unvisited locations. These methods use either the BPR loss [25] or the binary cross-entropy loss for optimization by contrasting visited locations with random samples from unvisited locations. However, informativeness is different from sample to sample, so treating them equally in these loss functions is far from optimal.

To this end, we propose a Geography-aware sequential recommender based on the Self-Attention Network (GeoSAN for short) for location recommendation. In GeoSAN, in addition to embedding user, location and time, we also embed the exact GPS of the location with a novel geography encoder to address the first challenge. To be more specific, we first follow the tile map system to cut the world map into tiles (i.e., grids) of the same size at different levels of detail and use quadtree keys (quadkeys for short) for grid addressing. Then, given the specific level of detail, the GPS point is mapped into

*Enhong Chen is the corresponding author.

KDD '20, August 23–27, 2020, Virtual Event, CA, USA

ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403252

a grid and represented with the quadkey of the grid[1]. A quadkey at level l can be interpreted as a base-4 number of l digits. For example, the Eiffel Tower (latitude=48.858093 and longitude=2.294694) is represented with the quadkey "12022001101200033" at the 17-th level of detail. It is intuitive and straightforward to embed the quadkeys directly, but spatial proximity between grids is still not encoded. Note that each quadkey starts with the quadkey of its parent grid and the quadkeys of nearby grids are similar to each other. Therefore, we apply the self-attention network for encoding the n-gram of quadkeys, such that nearby grids are similarly represented.

To address the sparsity challenge, we propose a weighted binary cross-entropy loss based on importance sampling, so that informative negative samples are more weighted. Since the informative samples can contribute more to the gradient, the magnitude of gradient grows larger and the training course can be accelerated. The loss function can be adaptive to any negative samplers and can be optimized by any solvers. To make further use of geographical information, we propose geography-aware negative samplers, such that more informative negative locations are sampled with larger probability, promoting the informativeness of negative samples.

The contributions can be summarized as follows:

- We propose Geography-aware sequential recommender based on the Self-Attention Network (GeoSAN) for location recommendation, to capture long-term sequential dependence and to make fully effective use of geographical information.
- We propose the self-attention based geography encoder to represent the exact GPS positions of locations, such that spatial proximity between nearby locations can be captured. In this way, the spatial clustering phenomenon and distance-aware location transition can be implicitly modeled.
- We propose a new loss function based on importance sampling for optimizing GeoSAN, such that more informative negative samples are assigned larger weights. Geography-aware negative samplers are also proposed as the proposal distribution, such that the informativeness of negative samples is promoted.
- We evaluate GeoSAN with three real-world LBSN datasets. The results not only show that GeoSAN remarkably outperforms the state-of-the-art location sequential recommenders, but also show the advantage of the new loss function, and the effectiveness of two approaches of incorporating geographical information.

2 RELATED WORK

We first review recent literature of sequential location recommendation and then investigate recent advance of sequential recommendation, but concentrating on self-attention based methods.

2.1 Sequential Location Recommendation

Sequential location recommendation was modeled by pairwise interaction based tensor factorization [3, 18], personalized metric embedding [8], word embedding techniques [7], recurrent neural networks (with attention) [5, 13, 22, 34, 40]. Geographical information was incorporated by embedding travel distance [5, 13], distance-specific location transition [22], geography-aware uniform sampler [3] or designing a geography-aware gate in RNN [40] to reduce the effect of mobility behaviors at distant locations. Temporal information was also incorporated by embedding time interval [13], embedding time of week [13, 34], and controlling information flow with interval-aware gating [40]. These recommendation models are optimized with the BPR loss [3, 5, 8, 15, 18, 22], the cross entropy loss [13, 34, 40], and hierarchical softmax [7]. The differences of the proposed algorithm from these existing works lie in the methods of geographical modeling, the loss function and the use of self-attention network for capturing long-range dependence.

2.2 Geographical Modeling in Non-Sequential Location Recommendation

In the LBSN datasets, the spatial clustering phenomenon was observed and explained by the Tobler's first law of geography [36]. The spatial clustering phenomenon was then characterized by the distance between individual visited locations following the power-law distribution. In order to avoid the power-law distribution assumption, kernel density estimation was used to estimate the distribution of the distance between pairs of locations [37]. Since modeling the distance distribution may ignore the multi-center characteristics of individual visited locations [2], geo-clustering techniques were developed to cluster individual visited locations [2, 21]. Because it is difficult to estimate the number of clusters, 2-D KDE was adopted for modeling the spatial clustering phenomenon [14, 16, 17, 38].

These methods for geographical modeling were integrated with location recommendation models in an ad-hoc way. Different from them, we propose the self-attention based geography encoder, which can be seamlessly integrated with the self-attention network for modeling the mobility history.

2.3 Sequential Item Recommendation with Self-Attention

Due to great business value, sequential item recommendation receives lots of attention recently and many algorithms have been proposed. Due to space limitations, we only discuss self-attention based sequential recommendation. For more sequential recommendation algorithms, please refer to the survey [24, 31].

Due to full parallelism and the capacity of capturing long-range dependence, the self-attention network [30] has been widely used in sequence modeling, and achieved the state-of-the-art performance in many NLP tasks. In recent two years, it was also used in sequential recommendation by optimizing the binary cross-entropy loss [11] based on inner product preference and triplet margin loss [39] based on Euclidean distance preference. The experimental results show that it significantly improved the performance of recommendation compared to RNN. Since time intervals between consecutive interactions may be different from each other, the self-attention network, which was originally designed for symbol sequences, does not use them for modeling sequential dependence. As such, the time interval aware self-attention network was proposed [12] by refining attention weights and values with time interval. Instead of the causality mask [12], a strategy (the Cloze objective) used for training BERT [6] was borrowed to avoid information leakage and improve recommendation performance [29].

The differences of GeoSAN from them lie in the geography encoder and the new loss function. It is possible to combine them to achieve further improvements in recommendation accuracy.

[1] https://en.wikipedia.org/wiki/Tiled_web_map

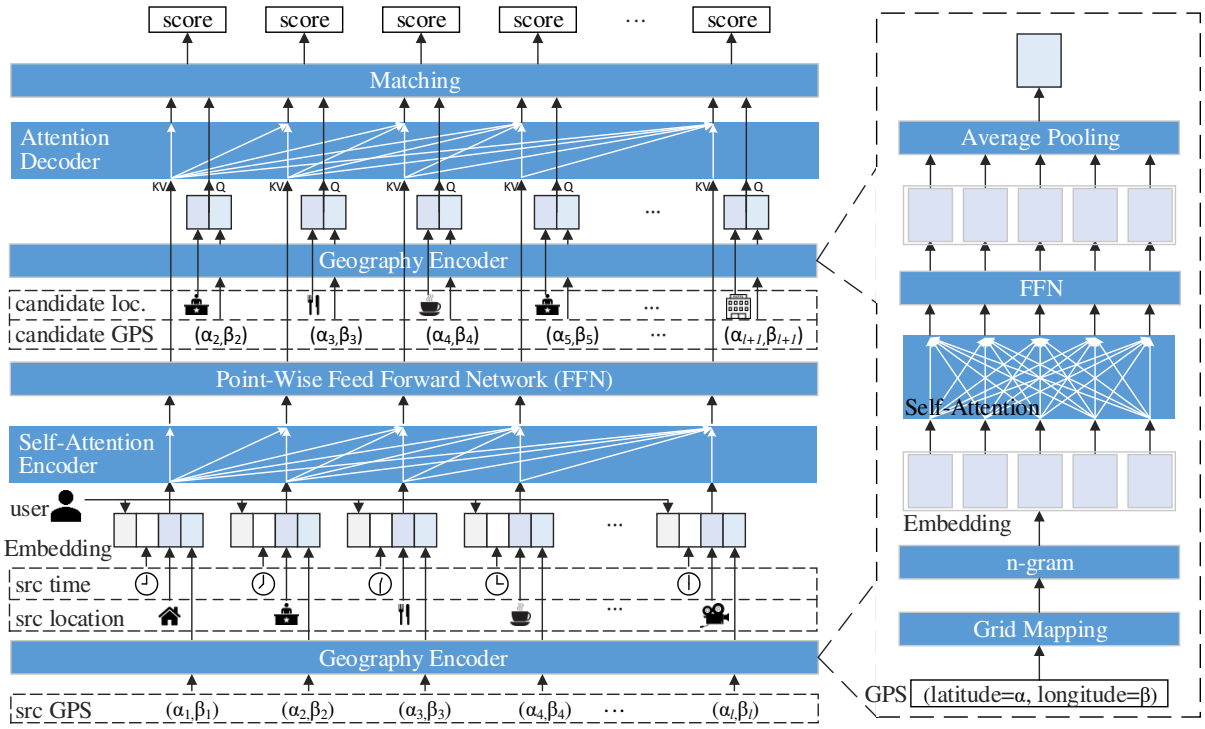


Figure 1: Framework of the proposed GeoSAN and the geography encoder

3 GEOGRAPHY-AWARE SEQUENTIAL RECOMMENDER

Sequential location recommender takes as input a user's mobility trajectory $S^u = r_1^u \to r_2^u \to \cdots \to r_n^u$, where $r_i^u = (u, t_i, l_i, p_i)$ indicates a behavior that user u visited location l_i with the exact position $p_i = (\text{latitude} = \alpha_i, \text{longitude} = \beta_i)$ at time t_i. Given the input trajectory, sequential location recommender predicts the next location l_{i+1} with the GPS position p_{i+1}. During the training process, as shown in Figure 1, we consider the trajectory except the last $r_1^u \to r_2^u \to \cdots \to r_{n-1}^u$ as input sequence and the trajectory except the first $r_2^u \to r_3^u \to \cdots \to r_n^u$ as output sequence. The proposed geography-aware sequential recommender is illustrated in Figure 1. Each part will be elaborated in the following sections.

3.1 Geography-aware Self-Attention Network

3.1.1 Embedding. The input sequence is first transformed into a fixed-length sequence. Given the maximum length m, if the input sequence length is greater than m, we split it from right to left into multiple sub-sequences. When the input sequence length is less than m, we repeatedly add a "padding" behavior to the right until the length grows to m. We then embed user, hour of week and location of each behavior, and encode the exact position with a novel geographical encoder, whose design will be elaborated later. These vectors are concatenated, forming the representation matrix $\boldsymbol{E} \in \mathbb{R}^{m \times d}$ of the sequence. For the padding behaviors, we use a constant zero vector for encoding. Since the self-attention encoder can not capture relative positions in the sequence like RNN, we follow [11] to add positional embedding $\boldsymbol{P}$ into $\boldsymbol{E}$, i.e., $\boldsymbol{E} = \boldsymbol{E} + \boldsymbol{P}$.

3.1.2 Self-Attention Encoder. To capture long-range dependence, we apply the self-attention encoder [30] for transforming the representation matrix $\boldsymbol{E}$ of the sequence. The self-attention encoder stacks multiple self-attention blocks, each of which consists of a self-attention layer and a point-wise feed-forward network (FFN). The self-attention layer takes the representation matrix $\boldsymbol{E}$ of the sequence as input and feeds them into an attention module after converting it through three distinct matrices $\boldsymbol{W}_Q, \boldsymbol{W}_K, \boldsymbol{W}_V \in \mathbb{R}^{d \times d}$,

$$\boldsymbol{S} = \text{SA}(\boldsymbol{E}) = \text{Attention}(\boldsymbol{E}\boldsymbol{W}_Q, \boldsymbol{E}\boldsymbol{W}_K, \boldsymbol{E}\boldsymbol{W}_V) \tag{1}$$

where the attention module used in the self-attention layer is the scaled dot-product attention, i.e.,

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}})\boldsymbol{V} \tag{2}$$

Note that the recommendation of the $n+1$-th location depends on not the future behaviors but only the preceding n behaviors. We achieve the causality constraint by a square mask, which is filled $-\infty$ in the upper triangle and 0 in other entries.

The FFN, being identically applied on each behavior representation, is used to endow the self-attention with nonlinearity and encode the interactions between dimensions. The FNN is a two-layer feed-forward network, whose output at step i is

$$\boldsymbol{F}_i = \text{FFN}(\boldsymbol{S}_i) = \max(0, \boldsymbol{S}_i\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2 \tag{3}$$

where $\boldsymbol{W}_1 \in \mathbb{R}^{d \times d_h}, \boldsymbol{W}_2 \in \mathbb{R}^{d_h \times d}$, s.t. $d_h > d$.

When stacking multiple self-attention blocks, residual connection and layer normalization are applied in FFN and the self-attention layer for stabilizing and speeding up the training process [30].

3.1.3 Target-aware Attention Decoder. The output of the l self-attention blocks is denoted by $\boldsymbol{F}_i^{(l)}$. Most existing self-attention based recommenders directly feed these outputs into the matching module, which may be suboptimal according to recent studies in [20, 41]. To improve the representation of the input sequence with respect to target locations, we introduce the target-aware attention decoder in GeoSAN as follows:

$$\boldsymbol{A} = \text{decoder}(\boldsymbol{F}^{(l)}|\boldsymbol{T}) = \text{Attention}(\boldsymbol{T}, \boldsymbol{F}^{(l)}\boldsymbol{W}, \boldsymbol{F}^{(l)}) \tag{4}$$

where $\boldsymbol{T} \in \mathbb{R}^{n \times d_T}$ is the representation matrix of the output sequence, obtained by concatenating embedding vectors of candidate locations and representation vectors of their geographical information. $\boldsymbol{W}$ is a matrix for mapping queries and keys into the same latent space. This can also be viewed as substituting the scaled dot-product attention with bilinear attention for computing attention weights between queries and keys. Note that the causality constraint is also required, achieved by the aforementioned mask.

3.1.4 Matching. Given the representation $\boldsymbol{A}_i$ of the input sequence at step i, we can compute preference score for each candidate location with any matching function f, like deep neural network [41], inner product [11] when representations of history and candidate are of the same dimension, and bilinear when they are of different dimensions. In particular, the preference score is written as follows:

$$y_{i,j} = f(\boldsymbol{A}_i, \boldsymbol{T}_j), \tag{5}$$

where $\boldsymbol{T}_j$ is the representation vector of candidate location j. Here we have to emphasize that the representation vector of each candidate location should consider the embedding of both location and its geographical information at the same time. Otherwise, geographical information can not take effect according to our empirical observation. Similar to [11], the embedding matrices and the geography encoder are shared between output sequences and input sequences.

3.2 Geography Encoder

The exact position of each location is usually described by latitude and longitude. Though they are continuous, it may not be suitable to feed them into the learning system directly. The reasons are two-fold. First, latitude and longitude can describe the whole earth's surface, but locations that human can access, are usually located in a very small region of the earth's surface. In other words, the learning system trained with human mobility suffers from the sparsity issue. Second, latitude and longitude strongly interact with each other since it is only possible to identify a location by making joint use of them. It may be difficult for the learning system to learn the strong interaction between them.

To this end, we propose the geography encoder, which embeds the exact position of location by first mapping latitude and longitude into a grid, and then encoding the unique id (quadkey) of the grid with a self-attention network.

3.2.1 Map Gridding and GPS Mapping. Before introducing how to map GPS points, we first divide the world map hierarchically into grids by following the Tile Map System. The tile map system has been widely used for quick navigation and interactive display in web map services, such as Google Maps and Bing Maps. In the system, the spherical form of the Mercator projection is used to project the entire world into a flat plane. The scale of the plane starts with 512x512 pixels and grows by a factor of 2 with the increase of levels. To improve the performance of map retrieval and display, the plane is cut into tiles (i.e. grids) of 256 x 256 pixels each. In this way, when the number of levels increases by one, each grid is divided into four sub-grids of the same size. The hierarchical gridding of a local plane around the Eiffel Tower (latitude=48.858093 and longitude=2.294694) is illustrated in Figure 2.

Figure 2: Hierarchical map gridding based on the Tile Map System, and the mapping of a location (the Eiffel Tower) into grids at level 16-18, whose quadkeys are annotated.

To map GPS points into grids, the latitude α and longitude β, assumed to be on the WGS (World Geodetic System) 84 datum, are converted into a Cartesian coordinate in the spherical-projected plane[2] given a specified level l, which is calculated as follow:.

$$\begin{aligned} x &= \frac{\beta + 180}{360} \times 256 \times 2^l \\ y &= \left(\frac{1}{2} - \frac{1}{4\pi} \log \frac{1 + \sin(\alpha \times \pi/180)}{1 - \sin(\alpha \times \pi/180)}\right) \times 256 \times 2^l \end{aligned} \tag{6}$$

The map from a Cartesian coordinate (x,y) to a grid is simply achieved by dividing them by 256. Because of grid division like quadtree, each grid can be identified with a quadtree key (quadkey for short), which can be interpreted as a base-4 number and whose length equals the level of detail. As illustrated in Figure 2, at the 17-th level of detail, the Eiffel Tower is mapped into a grid with the quadkey "12022001101200033". As such, the GPS mapping is achieved by making joint use of latitude and longitude, addressing the issue of the strong interaction between them. Different locations in a grid share the same quadkey as the grid, and grids without locations are directly ignored when embedding, so the sparsity issue is addressed to some extent.

3.2.2 Encoding Quadkeys. By considering the quadkeys of grids at the last level as a category variable, it is intuitive and straightforward to embed them with an embedding matrix. This method suffers from two issues. First, it is difficult to set an appropriate level of detail to stop map partition. The growing number of levels lead to the exponential increase number of grids, which is likely to better distinguish locations, but suffers from the sparsity issue. Second, the physical proximity between nearby grids is not modeled. In particular, nearby grids may have similar quadkeys. For

[2] https://en.wikipedia.org/wiki/Map_projection

example, in the leftmost of Figure 2, the quadkeys of the four grids share the same prefix "120220011012000". In other words, proximity relationships between nearby grids are partially hidden in their quadkeys.

To this end, we consider quadkeys as a character sequence, whose each character is in the set {"0", "1", "2", "3"}, and then apply the self-attention network for encoding the character sequence of the quadkey. However, the cardinality of the character set is small, and the meaning of characters at different positions may be different from each other, so such a character-level model can not fully encode the proximity between nearby grids. Therefore, we transform each quadkey into the sequence of n-grams first, such that the vocabulary size increases from 4 to 4^n. Taking the first 8 digits – "12022001" of the aforementioned quadkey as an example, the transformed sequence of trigrams is $120 \rightarrow 202 \rightarrow 022 \rightarrow 220 \rightarrow 200 \rightarrow 001$. After embedding the sequence of n-grams, we apply a stacked self-attention network for capturing sequential dependence, and then aggregate the sequence of n-gram representations via average pooling. Note that when the hyperparameter n equals the number of digits in the quadkeys, it would degenerate to the vanilla embedding. Hence, n controls the model capacity of the geography encoder, whose sensitivity will be analyzed in the experiments.

3.3 Loss Function with Importance Sampling

Recall that given the sequence S^u, the preference score at step i for a candidate location j is $y_{i,j}$. It is easy to understand that optimizing the cross-entropy loss [13, 34, 40] is not efficient when the number of candidate locations grows large. Existing sequential recommenders based on self-attention usually used the binary cross-entropy loss [11, 12], which is written as follows:

$$-\sum_{S^u \in S}\sum_{i=1}^{n}\left(\log \sigma(y_{i,o_i}) + \sum_{k \notin L^u} \log(1-\sigma(y_{i,k}))\right), \tag{7}$$

where S is a training set of mobility trajectories, o_i is the target location at step i and L^u is a set of visited locations by user u. Here we have already ignored the padding in the output sequence. In order to efficiently optimize the loss, one negative item is sampled from unvisited locations at each step based on the uniform distribution.

Since only one negative item is sampled from unvisited locations, the binary cross-entropy loss can not make fully effective use of the large number of unvisited locations. In particular, after the loss is optimized for several epochs, positive samples may be easily distinguished from randomly sampled negative locations, so that the gradient of the loss is of small magnitude and the training course is slow. In other words, because informative negative samples are missing, the binary cross-entropy loss is hard to reduce. It is intuitive that unvisited locations with large preference scores $y_{i,j}$ can contribute more to gradient, so they are more informative and should be sampled with high probability. However, it is infeasible to consider top-k unvisited location with the largest preference scores as negative due to the false negative problem. It is also infeasible to directly sample negative locations with probability in proportion to the preference scores due to the efficiency issue. To this end, we propose to weight unvisited locations with the probability being negative, such that even with the uniform sampler, more informative locations can be more emphasized. In particular, the loss function is reformulated as follows:

$$-\sum_{S^u \in S}\sum_{i=1}^{n}\left(\log \sigma(y_{i,o_i}) + \sum_{k \notin L^u} P(k|i)\log(1-\sigma(y_{i,k}))\right), \tag{8}$$

where $P(k|i)$ is the probability of location k being negative given the mobility trajectory $r_1^u \rightarrow r_2 \rightarrow \cdots \rightarrow r_i^u$. We suggest to model the probability as follows

$$P(k|i) = \frac{\exp(r_{i,k}/T)}{\sum_{k' \notin L^u}\exp(r_{i,k'}/T)} \tag{9}$$

where T is a temperature parameter, controlling the divergence of the probability distribution from the uniform distribution. When T approaches ∞, it is equivalent to the uniform distribution.

However, this loss function still suffers from low efficiency of computing normalization in the probability. To improve the efficiency, by considering $\sum_{k \notin L^u} P(k|i)\log(1-\sigma(y_{i,k}))$ as computing expectation with respect to $P(k|i)$, we propose to approximate the expectation with importance sampling. Suppose the proposal distribution is $Q(k|i)$, from which it is easy to sample, and denote by $\tilde{Q}(k|i)$ the unnormalized probability of $Q(k|i)$. The loss is then approximated as follows according to [1] (also see Appendix A.3)

$$-\sum_{S^u \in S}\sum_{i=1}^{n}\left(\log \sigma(y_{i,o_i}) + \sum_{k=1}^{K} w_k \log\left(1-\sigma(y_{i,k})\right)\right), \tag{10}$$

where $w_k = \frac{\exp\left(r_{i,k}/T - \ln \tilde{Q}(k|i)\right)}{\sum_{k'=1}^{K}\exp\left(r_{i,k'}/T - \ln \tilde{Q}(k'|i)\right)}$ is the weight of the k-th sample. Therefore, among K locations, the locations with larger preference scores are assigned larger weight. When the proposal distribution is deviated from $P(k|i)$, the sample weight can compensate divergence between P and Q to some extent.

Note that only the unnormalized probability is used, being convenient for probability distribution over a subset of the whole locations L. When $Q(k|i)$ is a uniform distribution over $L \backslash L^u$, $\ln \tilde{Q}(k|i) \propto -\ln|L|$ and thus $w_k = \frac{\exp(r_{i,k}/T)}{\sum_{k'=1}^{K}\exp(r_{i,k'}/T)}$. $Q(k|i)$ can be approximated with the uniform distribution over L, due to extremely low probability of sampling locations in L^u as negative and the comparable accuracy of recommendation. When designing other proposal samplers, we also consider the distribution over L instead of $L \backslash L^u$ for simplification, which is also widely used in the field of NLP [23].

3.4 Geography-aware Negative Sampler

In the sequential location recommender, geographical information can also be effective to distinguish negative from potentially positive in unvisited locations. For example, when he/she visits the target location o_i at time t_i, the unvisited locations around o_i may be more likely to be negative. However, it may be computationally infeasible to directly sample locations based on GPS distance, such that closer locations are sampled with larger probability. Therefore, in the geography-aware negative samplers, we suggest to first retrieve K nearest locations to the target location and then randomly draw negative samples from these K candidates. We consider negative sampling based on the uniform distribution or a popularity-based distribution. In the popularity-based proposal, according to our empirical findings, it is better to use $\tilde{Q}(k|i) \propto \ln(c_k + 1)$, where c_k denotes occurring frequency in the mobility history.

4 EXPERIMENTS

4.1 Datasets

We use three publicly available real-world Location-Based Social Network datasets to evaluate our method: Gowalla, Brightkite [4], and Foursquare [35]. We remove users with fewer than 20 check-ins and locations which have been visited fewer than 10 times. Table 1 summarizes the statistics of the three datasets. For the check-in sequence of each user, we take the last check-in record on a previously unvisited location for evaluation, and all check-ins prior to that for training. The maximum sequence length is set to 100. Longer sequences will be divided into non-overlapping sub-sequences of length 100 from right to left, and the most recent 100 check-ins will be used in the evaluation.

Table 1: Dataset statistics.

	Gowalla	Brightkite	Foursquare
#users	31,708	5,247	12,695
#locations	131,329	48,181	37,344
#check-ins	2,963,373	1,699,579	1,941,959

4.2 Baselines

To show the effectiveness of our proposed method, we compare it with the following baselines:

- FPMC-LR [3] uses tensor factorization to model personalized location transitions and incorporates a localized region constraint in learning the transition tensor.
- PRME-G [8] utilizes metric learning to project users and locations into the sequential transition space and the user preference space to learn user-specific transition patterns. Geographical influence is incorporated by multiplying a travel-distance based weight.
- GRU is a simple baseline based on GRU4Rec [10], which adapts our framework by adopting a single-layer GRU for modeling sequences and the BPR loss [25] for optimization. We have also tried the popularity-based negative sampling in [10], but it leads to significant performance drop.
- SASRec [11] is a state-of-the-art sequential recommendation method based on self-attention mechanisms, but it only uses the most recent sequence of each user for training.
- STGN [40] enhances LSTM networks by introducing spatio-temporal gates to capture the spatio-temporal relationships between successive check-ins.

4.3 Metrics

The performance of recommendation is assessed by how well the target locations in the test set are ranked. We adopt two widely-used metrics of ranking evaluation: Hit Rate and NDCG [32]. Hit Rate at a cutoff k, denoted as HR@k, counts the fraction of times that the target location is among the top k. NDCG at a cutoff k, denoted as NDCG@k, rewards method that ranks positive items in the first few positions of the top-k ranking list. We report the two metrics at $k = 5$ and $k = 10$ in our experiments.

We consider a practical scenario that the recommendations are made based on the user's current GPS coordinates (consider how we get recommendations on where to go next when we open the map app on our mobile phones). For the sake of efficient evaluation, we first retrieve the nearest 500 previously-unvisited locations to the target location as negative candidates. In order to make the negative candidates more difficult to differentiate from the target ones, we train a Weighted Regularized Matrix Factorization (WRMF) model on the user-location interaction matrix by further dividing the training set into an 80% one for WRMF training, and a remaining 20% one for WRMF testing. Then we sort the 500 negative location candidates according to the scores predicted by the WRMF model. The top 100 locations are selected as the negative locations to rank with the target location. Hit Rate and NDCG can then be computed based on the ranking of these 101 locations. In addition to this recommendation scenario, we conduct evaluation in another scenario – next location recommendation, where each user's next physical position is unknown, and the negative samples have to be drawn from the vicinity of the immediately preceding check-in location. The details are discussed in the Appendix A.1.

4.4 Settings

We set the dimension of location embedding to 50 for all methods, the other parameters of the baselines to the default values. In our GeoSAN model, we use 6-gram tokens to represent quadkeys at the 17-th level of detail. We train our model using the Adam optimizer with a learning rate of 0.001 and set the dropout ratio to 0.5. The number of training epochs is set to 50 for Gowalla and Foursquare, and 20 for Brightkite. We use two layers of self-attention modules for both the check-in sequence encoder and the geography encoder, and set the temperature T in our new loss function to 1.0 for Gowalla and Foursquare, and 100.0 for Brightkite. For each location in the dataset, we retrieve its nearest K=2000 locations for the kNN-uniform negative sampler to sample from. We set the number of negative samples for training to 5 for Gowalla and Foursquare, and 9 for the Brightkite dataset. Our model is implemented in PyTorch and available at github[3].

4.5 Comparison with Baselines

The results are summarized in Table 2. We observe that our proposed model consistently outperforms all compared baselines on all three datasets. Our proposed method archives up to 43.0% and 50.5% improvements over the best-performing baseline in terms of HR@5 and NDCG@5. SASRec is a strong baseline with decent performance on all three datasets, validating the effectiveness of the self-attention architecture. Interestingly, SASRec can be further improved when being adapted to our framework, i.e., dividing long sequences into multiple sub-sequences of length 100, instead of only using the most recent sequence of length 100. This also explains the good performance of GRU, which adapts our framework by simply adopting GRU and the BPR loss. Thanks to geographical modeling and ranking evaluation based on nearby locations of the target location, PRME-G even shows higher accuracy than GRU and SASRec. Though STGN also incorporates geographical information by designing spatial-aware gates, STGN performs poor in all three datasets. The main reason may lie in insufficient modeling of geographical information.

[3] https://github.com/libertyeagle/GeoSAN

Table 2: Comparison with baselines

	Gowalla				Brightkite				Foursquare			
	HR@5	NDCG@5	HR@10	NDCG@10	HR@5	NDCG@5	HR@10	NDCG@10	HR@5	NDCG@5	HR@10	NDCG@10
FPMC-LR	0.1486	0.1061	0.2202	0.1291	0.1427	0.1021	0.2203	0.1270	0.2191	0.1516	0.3150	0.1825
PRME-G	0.4008	0.3148	0.5029	0.3477	0.3678	0.2680	0.4904	0.3075	0.2615	0.1870	0.3607	0.2190
GRU	0.3815	0.2935	0.4883	0.3279	0.3528	0.2662	0.4675	0.3033	0.2838	0.2007	0.3900	0.2350
STGN	0.2299	0.1636	0.3311	0.1962	0.1997	0.1420	0.3051	0.1757	0.1979	0.1331	0.2905	0.1629
SASRec	0.3814	0.2926	0.4930	0.3286	0.3333	0.2514	0.4446	0.2874	0.2929	0.2081	0.4014	0.2431
GeoSAN	**0.4951**	**0.3898**	**0.6028**	**0.4245**	**0.5258**	**0.4034**	**0.6425**	**0.4409**	**0.3735**	**0.2714**	**0.4867**	**0.3080**

4.6 Ablation Study

To analysis the effectiveness of the various components in our method, we conduct an ablation study. Our base model (denoted as Original) does not use user embedding, time embedding, and target-aware attention decoder. We consider the following variants of our model:

- I. *US (Uniform Sampler)*: We use the uniform negative sampler over all locations, instead of the kNN-uniform negative sampler.
- II. *BCE (Binary Cross-Entropy) Loss*: We use the binary cross-entropy loss, without assigning weights to negative samples.
- III. *Remove GE (Geography Encoder)*: We remove the geography encoder and only use location embedding for representation.
- IV. *Remove PE (Positional Embedding) in GE*: We remove positional embedding used in the geography encoder, which injects relative position information of each n-gram token within a quadkey.
- V. *Add UE (User Embedding)*: We add a user embedding into the check-in sequence encoder by concatenating user embedding with location embedding and geographical representation from the geography encoder. A linear layer is then used for transformation to match the dimension of candidate representation.
- VI. *Add TE (Time Embedding)*: We add a time embedding in a similar way to V. The timestamp of each check-in record is first mapped to a one-hour interval within a week (i.e., there are $24 \times 7 = 168$ time intervals) and then fed to an embedding layer.
- VII. *Add TAAD (Target-Aware Attention Decoder)*: We add the target-aware attention decoder in this variant. We add a residual connection between the output of the sequence encoder and the output of the decoder, followed by layer normalization.

The results are summarized in Table 3. From this table, we can have the following findings:

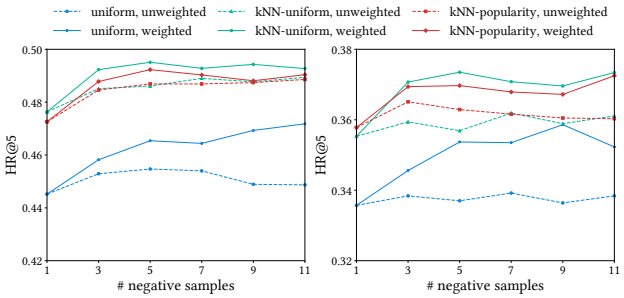
- *Finding 1: kNN-uniform sampler dramatically boosts the performance by drawing relatively "hard" negative samples.* We can see that using kNN-uniform sampler instead of the global uniform sampler leads to an improvement of 6.8%, 3.2%, 7.0% on the three datasets, with respect to NDCG@5. kNN-uniform sampler only samples from unvisited locations in the vicinity of the target location. The "localized" negative locations are more difficult to distinguish from the target location than the negative samples generated by the global sampler. The global uniform sampler has a high probability of generating locations far away from the user's current position, which the user is more unlikely to visit next (e.g., the sampler generates a location in China for a user living in Australia). Compared to the "easy" negative samples, the "hard" negative samples are more likely to improve the discriminating ability of the model. Besides, the kNN-uniform sampler is consistent with the evaluation setting by using nearby locations of the target location for ranking.
- *Finding 2: The proposed new loss function proves to be helpful.* Compared to the unweighted binary cross-entropy loss, using weighted loss improves the performance by 2.3%, 2.1% and 5.2% on the three datasets in terms of NDCG@5. This is because the weighted loss function pays more attention to the generated samples with the higher negative probability.
- *Finding 3: Incorporating the geographical information by the geography encoder dramatically improves the accuracy of recommendation.* The improvements are 12.0%, 24.5%, 4.3% on the three datasets in terms of NDCG@5. This implies that the spatial relations between locations are of vital use, and must be well modeled in order to make precise recommendation. We find that the geography encoder works especially well on Gowalla and Brightkite. This is because they contain worldwide data while the Foursquare dataset only focuses on a single region. The comparison with the variant IV indicates that there is no significant effect in adding positional information in geography encoder.
- *Finding 4: Adding user embedding or time embedding does not lead to performance improvement.* This may contribute to the mismatch between the check-in embedding space and the candidate location embedding space. In these two variants (V, VI), the added embedding is first concatenated with the location embedding and the geography encoding, and then goes through a linear layer. This may lead to a deviation from the candidate embedding space (location embedding $\oplus$ geography encoding). We have also tried to implement this linear projection on the output of the sequence encoder (late-fusion approach), but this does also not lead to improvement of recommendation accuracy.
- *Finding 5: Using a target-aware attention decoder is helpful in certain circumstances.* Adding a decoder to attend to historical check-ins relevant to the target location leads to improvement of recommendation accuracy on the Foursquare dataset, but leads to a slight performance decrease on Gowalla and Brightkite.

4.7 Performance w.r.t. Negative Sampling and Loss Function

4.7.1 Settings. We investigate the effect of different combinations of negative samplers and weighted loss (our new loss function) / unweighted loss (the vanilla binary cross-entropy loss). Besides the uniform sampler and the kNN-uniform sampler, we also consider a kNN-popularity sampler, which samples from the target location's k-nearest neighbors according to location popularity. For each combination of the negative sampler and the loss function, we vary the

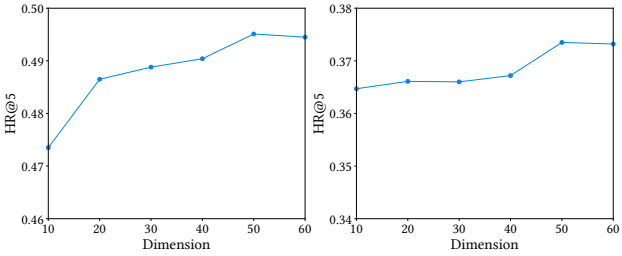
Table 3: Ablation analysis. Performance better than the original model is boldfaced.

	Gowalla				Brightkite				Foursquare			
	HR@5	NDCG@5	HR@10	NDCG@10	HR@5	NDCG@5	HR@10	NDCG@10	HR@5	NDCG@5	HR@10	NDCG@10
Original	0.4951	0.3898	0.6028	0.4245	0.5258	0.4034	0.6425	0.4409	0.3735	0.2714	0.4867	0.3080
I. *US*	0.4654	0.3650	0.5743	0.4001	0.4988	0.3908	0.6181	0.4295	0.3537	0.2537	0.4712	0.2917
II. *BCE Loss*	0.4861	0.3812	0.6010	0.4184	0.5091	0.3951	0.6312	0.4346	0.3569	0.2580	0.4738	0.2958
III. *-GE*	0.4472	0.3481	0.5555	0.3830	0.4197	0.3240	0.5256	0.3581	0.3572	0.2602	0.4735	0.2977
IV. *-PE*	0.4791	0.3782	0.5919	0.4146	0.5146	0.4030	0.6402	**0.4435**	**0.3774**	**0.2750**	**0.4904**	**0.3115**
V. *+UE*	0.4496	0.3521	0.5618	0.3884	0.4967	0.3857	0.6101	0.4223	0.3416	0.2454	0.4573	0.2826
VI. *+TE*	0.4794	0.3784	0.5899	0.4140	0.5045	0.3911	0.6261	0.4304	0.3631	0.2646	0.4789	0.3020
VII. *+TAAD*	0.4910	0.3871	0.5987	0.4219	0.4925	0.3821	0.6143	0.4214	**0.3837**	**0.2788**	**0.5013**	**0.3169**



(a) Gowalla (b) Foursquare

Figure 3: The impact of using different negative samplers, number of negative samples and loss functions.



(a) Gowalla (b) Foursquare

Figure 4: The impact of geography encoding dimension

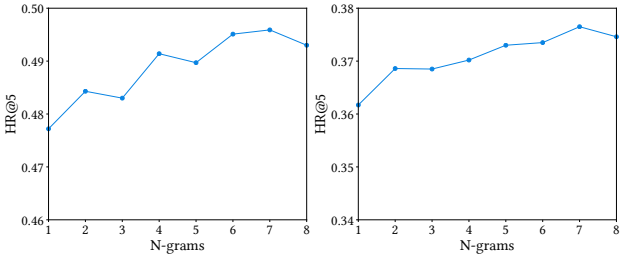
number of negative samples from 1 to 11 with a step 2. The results on the Gowalla and Foursquare dataset are shown in Figure 3.

4.7.2 Findings. First, *the models trained with the weighted loss consistently outperform the ones trained with the unweighted loss.* When using the weighted loss, the model trained with the uniform sampler improves a lot as the number of negative samples grows, while there is no notable variation in performance when using the unweighted loss. Second, *when using the kNN-based sampler, only a small number of negative samples could enable remarkable performance improvements.* This further verifies the effectiveness of sampling from nearby locations, and consistent with the next finding, *the models trained with the kNN-based sampler are consistently better than the ones trained with the uniform sampler.* We initially believe that the kNN-popularity sampler would further leads to improvement, but it takes little effect according to Figure 2. We use the uniform sampler and the popularity sampler to draw locations from the 500 negative evaluation candidates on Gowalla. The probability that the samples from the popularity sampler falls into the final 100 locations given by WRMF is very close to that from the uniform sampler, indicating the popularity sampler can not capture the behaviors of WRMF much more than the uniform sampler.

4.8 Sensitivity w.r.t. Geography Encoding Dimension

4.8.1 Settings. We vary the dimension used in the geography encoder from 10 to 60 with a step 10. Figure 4 shows the results.

4.8.2 Findings. We find that the performance gets much worse when using a small geography encoding dimension, which fails to capture the intricate geographical relations among different locations. A medium embedding size of 50 leads to peak recommendation accuracy on both datasets, which can adequately express the intrinsic geographical meaning of each n-gram token in the quadkeys. Further increasing the embedding dimension instead hurts the recommendation accuracy, as the number of possible n-gram tokens is limited. Specifically, we use 6-gram in our experiment, which leads to a vocabulary of size $4^6 = 4096$.



(a) Gowalla (b) Foursquare

Figure 5: The impact of N-grams

4.9 Sensitivity w.r.t. N-gram

4.9.1 Settings. We vary the n-grams used in the geography encoder from $n = 1$ to $n = 8$. The results are shown in Figure 5.

4.9.2 Findings. A small n results in a limited cardinality of the vocabulary set and limited representation power of tokens, resulting in poor recommendation performance. The performance will gradually reach a stable peak point with the increase of n. A 4-gram vocabulary, which includes $4^4 = 256$ possible tokens, makes the model capable enough to capture the hidden geographical information in quadkeys.

5 CONCLUSIONS

In this paper, we propose a geography-aware self-attentive sequential location recommender. In this recommender, we put forward a new loss function based on importance sampling, such that informative negative samples are better used, and design a new geography encoder to incorporate geographical information, such that the spatial clustering phenomenon and distance-aware location transition can be implicitly captured. We also develop geography-aware negative samplers to improve the informativeness of sampled negative locations. We then evaluate the proposed algorithm with three real-world datasets. The experimental results show that the proposed algorithm outperforms the state-of-the-art sequential location recommender by 34.9% on average. Through ablation study and sensitivity analysis, we also show the significant effect of the new loss, the new geography encoder, and the geography-aware negative samplers at improving recommendation performance. Future work can includes the pretraining of the geography encoder and the designing of more informative negative samplers.

ACKNOWLEDGMENTS

The work was supported by grants from the National Natural Science Foundation of China (No. 61976198, 61727809 and 61832017), Municipal Program on Science and Technology Research Project of Wuhu City (No. 2019yf05), and the Fundamental Research Funds for the Central Universities.

REFERENCES

[1] Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer-Verlag New York, Inc.
[2] Chen Cheng, Haiqin Yang, Irwin King, and Michael R Lyu. 2012. Fused matrix factorization with geographical and social influence in location-based social networks. In *Proceedings of AAAI'12*. 17–23.
[3] 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'13*. AAAI Press, 2605–2611.
[4] E. Cho, S.A. Myers, and J. Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *Proceedings of KDD'11*. 1082–1090.
[5] Qiang Cui, Yuyuan Tang, Shu Wu, and Liang Wang. 2019. Distance2Pre: Personalized Spatial Preference for Next Point-of-Interest Prediction. In *Proceedings of PAKDD'19*. Springer, 289–301.
[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
[7] Shanshan Feng, Gao Cong, Bo An, and Yeow Meng Chee. 2017. Poi2vec: Geographical latent representation for predicting future visitors. In *Proceedings of AAAI'17*.
[8] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. 2015. Personalized ranking metric embedding for next new POI recommendation. In *Proceedings of IJCAI'15*. AAAI Press, 2069–2075.
[9] M.C. Gonzalez, C.A. Hidalgo, and A.L. Barabasi. 2008. Understanding individual human mobility patterns. *Nature* 453, 7196 (2008), 779–782.
[10] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. In *International Conference on Learning Representations (ICLR)*.
[11] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *Proceedings of ICDM'18*. IEEE, 197–206.
[12] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time Interval Aware Self-Attention for Sequential Recommendation. In *Proceedings of WSDM'20*. 322–330.
[13] Ranzhen Li, Yanyan Shen, and Yanmin Zhu. 2018. Next point-of-interest recommendation with temporal and multi-level context attention. In *Proceedings of ICDM'18*. IEEE, 1110–1115.
[14] Defu Lian, Yong Ge, Fuzheng Zhang, Nicholas Jing Yuan, Xing Xie, Tao Zhou, and Yong Rui. 2018. Scalable Content-Aware Collaborative Filtering for Location Recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2018). https://doi.org/10.1109/TKDE.2018.2789445
[15] Defu Lian, Qi Liu, and Enhong Chen. 2020. Personalized Ranking with Importance Sampling. In *Proceedings of The Web Conference 2020*. 1093–1103.
[16] Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. 2014. GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *Proceedings of KDD'14*. ACM, 831–840.
[17] Defu Lian, Kai Zheng, Yong Ge, Longbing Cao, Enhong Chen, and Xing Xie. 2018. GeoMF++ Scalable Location Recommendation via Joint Geographical Modeling and Matrix Factorization. *ACM Transactions on Information Systems (TOIS)* 36, 3 (2018), 1–29.
[18] Defu Lian, Vincent Wenchen Zheng, and Xing Xie. 2013. Collaborative Filtering Meets Next Check-in Location Prediction. In *Proceedings of WWW'13 Companion*. ACM, 231–232.
[19] Defu Lian, Yin Zhu, Xing Xie, and Enhong Chen. 2014. Analyzing Location Predictability on Location-Based Social Networks. In *Proceedings of PAKDD'14*.
[20] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems. In *Proceedings of KDD'18*. ACM, 1754–1763.
[21] Bin Liu, Yanjie Fu, Zijun Yao, and Hui Xiong. 2013. Learning geographical preferences for point-of-interest recommendation. In *Proceedings of KDD'13*. ACM, 1043–1051.
[22] 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'16*. AAAI Press, 194–200.
[23] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[24] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. 2018. Sequence-aware recommender systems. *ACM Computing Surveys (CSUR)* 51, 4 (2018), 1–36.
[25] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of UAI'09*. AUAI Press, 452–461.
[26] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of WWW'10*. ACM, 811–820.
[27] C. Song, T. Koren, P. Wang, and A.L. Barabási. 2010. Modelling the scaling properties of human mobility. *Nature Physics* 6, 10 (2010), 818–823.
[28] C. Song, Z. Qu, N. Blumm, and A.L. Barabási. 2010. Limits of predictability in human mobility. *Science* 327, 5968 (2010), 1018–1021.
[29] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of CIKM'19*. 1441–1450.
[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.
[31] Shoujin Wang, Longbing Cao, and Yan Wang. 2019. A survey on session-based recommender systems. *arXiv preprint arXiv:1902.04864* (2019).
[32] Markus Weimer, Alexandros Karatzoglou, Quoc Viet Le, and Alex Smola. 2007. Maximum margin matrix factorization for collaborative ranking. *Proceedings of NIPS'07* (2007), 1–8.
[33] Yongji Wu, Defu Lian, Shuowei Jin, and Enhong Chen. 2019. Graph convolutional networks on user mobility heterogeneous graphs for social relationship inference. In *Proceedings of IJCAI'19*. AAAI Press, 3898–3904.
[34] Cheng Yang, Maosong Sun, Wayne Xin Zhao, Zhiyuan Liu, and Edward Y Chang. 2017. A neural network approach to jointly modeling social networks and mobile trajectories. *ACM Transactions on Information Systems (TOIS)* 35, 4 (2017), 36.
[35] Dingqi Yang, Bingqing Qu, Jie Yang, and Philippe Cudre-Mauroux. 2019. Revisiting user mobility and social relationships in lbsns: A hypergraph embedding approach. In *The World Wide Web Conference*. 2147–2157.
[36] Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik-Lun Lee. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In *Proceedings of SIGIR'11*. ACM, 325–334.
[37] Jia-Dong Zhang and Chi-Yin Chow. 2013. iGSLR: Personalized Geo-Social Location Recommendation-A Kernel Density Estimation Approach. In *Proceedings of GIS'13*.
[38] Jia-Dong Zhang, Chi-Yin Chow, and Yanhua Li. 2014. iGeoRec: A personalized and efficient geographical location recommendation framework. *IEEE Transactions on Services Computing* 8, 5 (2014), 701–714.
[39] Shuai Zhang, Yi Tay, Lina Yao, and Aixin Sun. 2018. Next item recommendation with self-attention. *arXiv preprint arXiv:1808.06414* (2018).
[40] 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'19*, Vol. 33. 5877–5884.
[41] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of KDD'18*. ACM, 1059–1068.

Table 4: Comparison with baselines when users' next physical positions are unknown.

	Gowalla				Brightkite				Foursquare			
	HR@5	NDCG@5	HR@10	NDCG@10	HR@5	NDCG@5	HR@10	NDCG@10	HR@5	NDCG@5	HR@10	NDCG@10
FPMC-LR	0.1462	0.1042	0.2169	0.1270	0.1500	0.1059	0.2293	0.1312	0.2177	0.1506	0.3111	0.1808
PRME-G	0.3126	0.2476	0.3899	0.2726	0.1481	0.0984	0.2239	0.1229	0.1223	0.0869	0.1738	0.1035
GRU	0.3459	0.2638	0.4473	0.2965	0.3394	0.2554	0.4420	0.2884	0.1920	0.1318	0.2784	0.1597
STGN	0.2438	0.1782	0.3390	0.2089	0.1955	0.1344	0.2973	0.1670	0.2012	0.1443	0.2886	0.1725
SASRec	0.3426	0.2591	0.4421	0.2912	0.3139	0.2279	0.4231	0.2631	0.1922	0.1329	0.2830	0.1620
GeoSAN	**0.4574**	**0.3571**	**0.5630**	**0.3911**	**0.4479**	**0.3488**	**0.5657**	**0.3868**	**0.2967**	**0.2107**	**0.3981**	**0.2433**

A APPENDIX

A.1 Another Evaluation Scheme

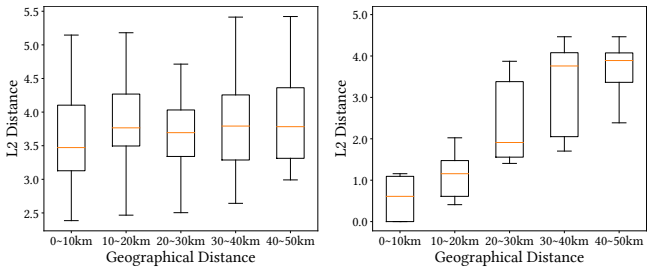
A.1.1 Settings. We consider another evaluation setting when users' next physical positions are unknown. In this case, we generate negative ranking samples based on users' last check-in locations. For each user, we first retrieve the 500 nearest locations to his/her last check-in location and then select the top 100 locations according to the scores predicted by the WRMF model. These 100 locations are considered as negative samples to rank with the target location, just like what we do when users' current positions are known. The results are reported in Table 4.

A.1.2 Findings. We see that the proposed model still outperforms all competing baselines on all three datasets in this particular setting. Concretely, the improvements compared to the best-performing baseline are 35.4%, 37.9%, 39.9% in terms of NDCG@5 on the dataset of Gowalla, Brightkite and Foursquare, respectively. This further validates the effectiveness of the proposed algorithm. The overall recommendation performance of all algorithms is relatively smaller than the other setting, and PRME-G degrades more than other algorithms in this setting, as it may be inaccurate to consider a user's last check-in location as his/her next position.

A.2 Effectiveness of the Geography Encoder in Capturing Geographical Relations

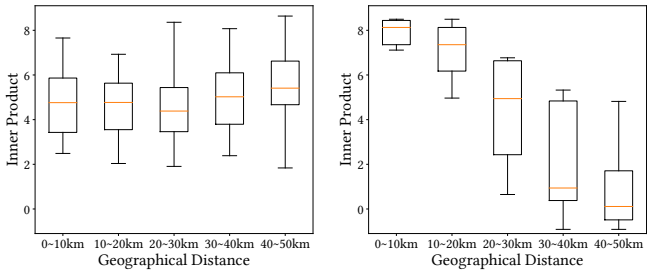
We propose the geography encoder to embed quadkeys using self-attention mechanism, and the geographical relations between different locations can then be implicitly captured in its latent space. We further validate whether quadkey representations learned by the geography encoder indeed imply certain underlying geographical relations. The underlying intuition is that Euclidean distance between quadkey representations of two distant locations should be larger than close ones.

A.2.1 Settings. We conduct an empirical analysis on the Foursquare dataset to validate our above intuition. We first find out the location (denoted as L) which has the most neighbors within a 50km radius. The neighboring locations are divided into 5 groups according to the distance from the location L, each of which corresponds to one of these 5 intervals: (0km, 10km], (10km, 20km], (20km 30km], (30km, 40km], (40km, 50km]. We then sample 30 locations from locations of each group, and compute Euclidean distances between the geography encoding of the 30 samples of each group and the geography encoding of L. As a comparison, we do the same for location embeddings (the embedding matrix used in our model to map location IDs to embedding vectors). We show box plots of the two types of embeddings in Figure 6.



(a) Location Embedding (b) Geography Encoding

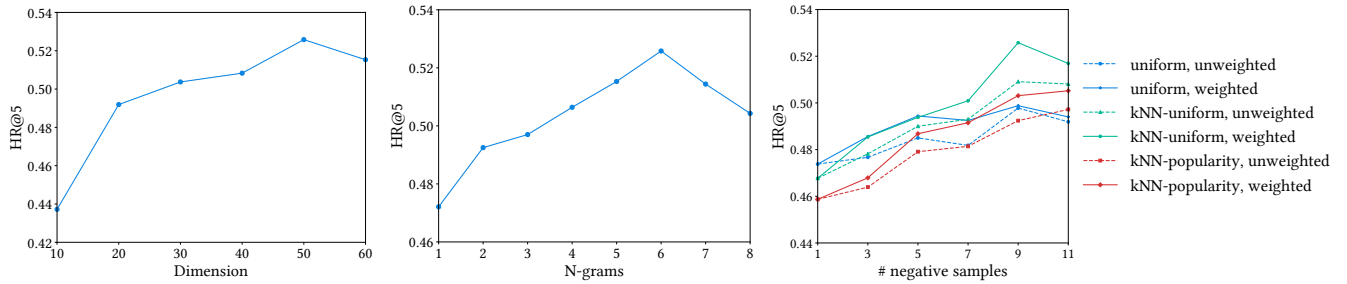
Figure 6: Box plots of the Euclidean distances between L's neighbors and L under geography encoding and location embedding.



(a) Location Embedding (b) Geography Encoding

Figure 7: Box plots of the inner product between L's neighbors and L under geography encoding and location embedding.

A.2.2 Findings. We see from Figure 6(b), locations farther away from L generally have greater Euclidean distances under the geography encoding. However, there is no significant relationship between Euclidean distances of the embedding vectors and geographical distances for locations, as shown in Figure 6(a). It suggests that the geography encoder indeed can capture geographical relations implicitly. We also show box plots of the inner product in Figure 7. It shows that locations farther away from L generally have smaller inner product under the geography encoding. And there is also no significant relationship between inner product of the embedding vectors and geographical distances for locations.



(a) The impact of geo-encoder dim. (b) The impact of N-grams. (c) The impact of negative sampler, negative sample number & loss functions.

Figure 8: Parameter sensitivity analysis on Brightkite.

A.3 Derivation of the Loss based on Importance Sampling

Given the probability $P(k|i) = \frac{\exp(r_{i,k}/T)}{\sum_{k' \notin L^u} \exp(r_{i,k'}/T)}$, we would like to approximate the expectation $\sum_{k \notin L^u} P(k|i) \log(1 - \sigma(y_{i,k}))$ by the proposal distribution $Q(k|i)$ based on importance sampling. In particular, given the K samples drawn from the proposal $Q(k|i)$,

$$\sum_{k \notin L^u} P(k|i) \log\left(1 - \sigma(y_{i,k})\right) \approx \frac{1}{K} \sum_{k} \frac{P(k|i)}{Q(k|i)} \log\left(1 - \sigma(y_{i,k})\right).$$

Since it is time-consuming to compute the normalization term, it is also approximated with these K samples, i.e.,

$$Z_P = \sum_{k \in L \setminus L^u} \exp(r_{i,k}/T) \approx Z_Q \frac{1}{K} \sum_{k} \exp\left(r_{i,k}/T - \ln \tilde{Q}(k|i)\right).$$

where $Q(k|i) = \frac{1}{Z_Q} \tilde{Q}(k|i)$. Then we define the weight of the k-th sample as follows:

$$w_k = \frac{1}{K} \frac{P(k|i)}{Q(k|i)} = \frac{Z_Q}{K Z_P} \frac{\tilde{P}(k|i)}{\tilde{Q}(k|i)} = \frac{\exp\left(r_{i,k}/T - \ln \tilde{Q}(k|i)\right)}{\sum_k \exp\left(r_{i,k}/T - \ln \tilde{Q}(k|i)\right)}.$$

Using the sample weight, the expectation can be simplified:

$$\sum_{k \notin L^u} P(k|i) \log\left(1 - \sigma(y_{i,k})\right) \approx \sum_{k} w_k \log\left(1 - \sigma(y_{i,k})\right).$$

A.4 Discussions of Capturing the Spatial Clustering Phenomenon in GeoSAN

According to Section A.2, the geography encoder indeed encodes geographical information successfully. Recall that the work [36] exploited geographical influence in the following way to capture the spatial clustering phenomenon,

$$\log P(k|L^u) = \sum_{l \in L^u} \log P(d(k,l)).$$

Considering a simplified GeoSAN, which only uses the geography encoder and the attention decoder, and computes preference scores based on inner product, it is then formulated as

$$y_{i,k} = \sum_{l=1}^{i} \alpha_{kl} \boldsymbol{E}_l^T \boldsymbol{E}_k,$$

where $\boldsymbol{E}_l$ is the output of the geography encoder by feeding the GPS of location l as input, and α_{kl} is the attention weight. According to Section A.2, $\boldsymbol{E}_l^T \boldsymbol{E}_k$ is negatively correlated with the physical distance $d(k,l)$ between location k and location l. Therefore, the preference score function in the simplified GeoSAN is strongly connected with [36], indicating that GeoSAN can implicitly capture the spatial clustering phenomenon without fitting the density of distance between any two visited locations. When only exploiting the last check-in location for computing the preference score $y_{i,k}$, the preference score function intrinsically models distance-aware location transition.

A.5 Parameter Sensitivity Analysis on the Brightkite Dataset

Here we present the results of parameter sensitivity analysis on the Brightkite dataset in Figure 8. Figure 8(a) and Figure 8(b) show the impact of using different geography encoding dimensions and using different n-grams. We see that the overall trend is similar to the other two datasets. Figure 8(c) illustrates the impact of using different combinations of negative samplers and the loss function when the number of negative samples varies. Again, we see that the proposed weighted loss function consistently outperforms the unweighted one, and the proposed kNN-based negative samplers is better than the uniform negative sampler.