Location Prediction with Personalized Federated Learning

shuang wang (wangsh@mail.neu.edu.cn)
Northeastern University https://orcid.org/0000-0002-1533-1051
Bowei Wang
Northeastern University Software College
Shuai Yao
Northeastern University Software College
Jiangqin Qu
Northeastern University Software College
Yuezheng Pan
Northeastern University Software College

Research Article

Keywords: location prediction, self-attention, federated learning, spatial-temporal data analysis

Posted Date: November 11th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-1016044/v1

Location Prediction with Personalized Federated Learning

Shuang Wang*, Bowei Wang[1], Shuai Yao, Jiangqin Qu, Yuezheng Pan

Northeastern University Shenyang, P.R.China

Abstract

Location prediction has attracted wide attention in human mobility prediction because of the popularity of location-based social networks. Existing location prediction methods have achieved remarkable development in centrally stored datasets. However, these datasets contain privacy data about user behaviors and may cause privacy issues. A location prediction method is proposed in our work to predict human movement behavior using federated learning techniques in which the data is stored in different clients and different clients cooperate to train to extract useful users' behavior information and prevent the disclosure of privacy. Firstly, we put forward an innovative spatial-temporal location prediction framework(STLPF) for location prediction by integrating spatial-temporal information in local and global views on each client, and propose a new loss function to optimize the model. Secondly, we design a new personalized federated learning framework in which clients can cooperatively train their personalized models in the absence of a global model. Finally, the numerous experimental results on check-in datasets further show that our privacy-protected method is superior and more effective than various baseline approaches.

Keywords: location prediction, self-attention, federated learning, spatial-temporal data analysis

1. Introduction

The rapid development of digital technology has made it easier to digitize the trajectories of human behaviors, which has led to location-based social network (LBSNs) such as Foursquare and Gowalla. These data sets make it easy for researchers to predict human behavior patterns. Human mobility and mobility patterns are not always predictable because of their high degree of freedom and variety and it is challenging to capture human trajectories.

The traditional method uses Markov chain (MCS) to capture the human movement regularities [1, 2]. However, the mobility probability is defined by the model from the beginning, and only the impact of the last check-in will be considered. Recently, some location prediction work [3]use embedding and recurrent neural networks (RNNs) to predict human behavior patterns because of the strong learning ability of deep learning. Although deep learning improves performance, it also leads to the problem of gradient disappearance. In order to overcome this difficulty, scholars have done a lot of research, such as using the RNNs variants (LSTM [4] and GRU[5]) in the encoder-decoder framework. However, if the length of the check-in sequence is greater than 50，it is difficult for

*Corresponding author *Email addresses:* wangsh@mail.neu.edu.cn (Shuang Wang)

1 First author. 1971160@stu.neu.edu.cn (Bowei Wang)

LSTM to learn the long-term dependence between sequences. On the other hand, geographic information is sparse, and humans generally only visit a small part of the map. In [6], the author uses the tile map system to convert the latitude and longitude of geographic information into a unique quadkey to alleviate the sparsity problem. To the end, we propose a local prediction model based on self-attention to deal with the forgetting problem of long sequences. A quadkey encoding module is proposed in this model. We first convert the quadkey into a sequence using a sliding window, and then add a self-attention network to extract the relationship among the sequences.

Existing loss functions such as cross entropy [7, 8] and binary loss function [9, 10] only contain zero or one negative sample, which cannot fully utilize the information of many negative samples. We put forward a new loss function in our paper. It can fully extract the information of negative samples by mixing hard negative samples and easy negative samples.

The training of deep learning requires many data and the user's data is scattered in different institutions. Due to legal restrictions, institutions cannot disclose the user's privacy, otherwise it will face huge punishment. In recent years, federated learning(FL) [11, 12] has been put forward to deal with data silos and prevent privacy breaches. Existing federated learning methods [13, 14] most focus on training a powerful global model. In [15], the authors propose that federated learning can also converge on non-ii(independent and identically distributed) data. However, it cannot guarantee good performance for each client(institution) since data distribution is heterogeneity among institutions. It is necessary to train a personalized model for each client. The performance of the personalization model depends on the effectiveness of the collaboration among clients. In this paper, we come up with an adaptive coordination mechanism to generate a personalization model with good performance for each client. This method can adaptively discover the cooperation relationship between clients in each global communication, which enables clients with similar data distribution to have stronger cooperation ability than those clients with different data distribution.

The main contributions we put forward are as follows:

- We put forward a novel location prediction model, which uses self-attention to learn information between long sequences and alleviate the forgetting problem of long sequences in both local and global views. We design a coding module to convert the quadkey mapping into a vector to make full use of the quadkey.
- We put forward a new loss function that is adaptive to any negative sampling. It mixes hard negative samples and easy negative samples to obtain useful information in the negative samples, so as to optimize the model in a better direction.
- We propose an adaptive federated learning framework, which discovers the collaboration between client segments in each global aggregation. It enables clients with more similar gradients to have stronger writing ability and generates a personalized model with good performance for each client.
- We evaluate our location prediction model on two LBSN datasets and the results show that ours is better than those baseline methods and also demonstrate the superiority of our loss function. Then, we compare our proposed adaptive personalized federated learning framework with the most advanced methods, and ours is also better than theirs.

2. Related Work

We briefly review the works in this section in two ways: some representative works for location prediction and self-attention.

Self-attention. In 2017, the Google team proposed the text representation of self-attention [21] learning, which shows the superiority of self-attention in sequential tasks. Then self- attention is widely employed in machine translation[17], image classification[16] and NLP tasks[18, 19, 20]. In 2017, Atrank [22] used the attention mechanism to model the user behavior pattern, extracted the relationship between users and applied it to the recommendation system. Zhang et al. [23] used it to make the next decision by learning the relationship among historical tracks.

Location Prediction. As a traditional position prediction method, the main idea of Markov chain (MC) model [2, 21] was to represent the individual moving trajectory as a Markov model. Then the state transition matrix was calculated and the next position was predicted according to the historical information. It only modeled short-term time series and its prediction ability was limited. Recently, recurrent neural networks and variants had good results in extracting long term sequence information. STRNN [15] used RNN model combined with spatial-temporal information[25] to predict the next location. This model can automatically capture the internal representation of spatial-temporal information and model human movement patterns. RNN model had poor ability to process long sequence data and RNN variant(LSTM, GRU) was proposed to deal with gradient disappearance recently. HST-LSTM model [26] used encoder and decoder and embeded spatial-temporal vector in LSTM to predict the next position. Attention was put forward to learn the long-term dependence between sequences. The DeepMove [27] used attention enhanced RNN to learn the user's mobility. VANext [28] made full use of CNN and attention to learn the law of historical tracks and predict the next location.

In the existing work, the extraction of time and coordinate information is limited. They cannot make full use of sparse spatial information. Some papers[26, 3] only calculated the time interval and spatial distance between two neighboring destinations, which limited model to learn the relationship between check-in locations from a local view. Different from these methods, our proposed model framework for location prediction considers both local and global views and have a good prediction performance.

The user's trajectory is privacy sensitive and the recent work on POI recommendation does not consider the issue of privacy protection. We added a machine learning technology called federated learning [12] to prevent privacy leakage. However, most of the existing federated learning methods [13, 14] focused on training a single global model and did not consider the heterogeneous data of each client. A single model was not guaranteed to be the optimal model for all clients. In this paper, we full captured the similarity of clients and train an effective personalized model for each client.

3. Preliminaries

In this section, we introduce the terminology and formal problems and the prior knowledge of the model.

3.1 Problem Statement

Let V and U denote POIs and users, $|U|$ and $|P|$ are users' and POIs' number.

Definition 1 (POI): POI $p = < v, c, l >$ is a location in the coordinate system, which contains

id v, category c and the geographical position information l(coordinates).

Definition 2 (CHECK-IN): The check-in is a tuple $rt_u = < u, p, t >$ denotes that user u visits the POI p at time t.

Definition 3 (TRAJECTORY): Given the user u's check-in records, trajectory sequence $tr^u = pt_i, ..., pt_L$, where L^u is the length, pt_i is the i-th check-in of user u.

Location Prediction Problem: Given a user's current trajectory sequence tr^u, the purpose is to predict the POI $p_{l_u+1} \in V$ in the $(L_u + 1)$-th time step of user u.

3.2 Tile Map System

Mercator projection is applied in tile map system [6], which maps the world into a plane. The initial plane's scale is 512*512 pixels. As the level increases, the scale doubles. A hierarchical gridding centered on the Leifeng Pagoda is shown in the figure 1. With a tile-map system and given a specific level we can map latitude and longitude to a unique id(quadkey). As shown in Figure 1, the quadkey of Leifeng Pagoda is "12022001101200013" when the level is 17.

Figure 1. Hierarchical mapping gridding based on Tile Map System

3.3 Federated Averaging

Federated averaging algorithm[12] involves multiple clients to train locally on their own data and communicate with the server to achieve a global model. The goal is to optimize the objective function.

$$min_w F(w) = \sum_{k=1}^{m} p_k F_k(w) \tag{1}$$

where $p_k \geq 0$, $\sum_{k=1}^{m} p_k = 1$ and m is all clients' amount. $F_k(w)$ is the local objective function and is defined in (2), where m_k is samples' amount of k-th client and $p_k = \frac{m_k}{m}$.

$$F_k(w) = \frac{1}{m_k} \sum_{j=1}^{m_k} l_i \, w \tag{2}$$

In the t-round iteration, local model parameters are reset to global model parameters by each client and then updates them locally for N times to obtain w_k^t and upload to the server. The central server then aggregates the client model using (3) to obtain the global model parameters.

$$w^{t+1} = \sum_{k=1}^{m} p_k \, w_k^t \tag{3}$$

4. Methodology

Firstly, we propose our local model, which consists of trajectory feature encoder and self-attention layer. Then we propose the loss function we designed. Finally, we combine federated learning with spatial-temporal model to protect users' privacy.

4.1 Spatial-temporal Location Prediction Framework (STLPF)

In this section we first elaborate the trajectory feature processing which contains a quadkey encoder and integrates the local spatial-temporal information which transmit to self-attention. Then we propose a self-attention layer which aggregates spatial-temporal information with local and global. The model's framework is described in Figure 3.

4.1.1 Trajectory Feature Processing

The location people visit is only a small part of the map and the longitude and latitude are sparse. First, we encode the longitude and latitude through the quadkey encoder to alleviate the sparsity. Then we embed each POI to the low dimensional vector representation.

(a) Quadkey Encoder

Given a specific level we can map latitude and longitude to a unique quadkey with the tile map system. We then embed quadkey directly with an embedding matrix. However, the closer the locations are and the more similar the quadkeys are and direct embedding cannot capture the similarity between them. We encode the quadkey through the self-attention network. Firstly, we transform quadkey q into a sequence s by sliding the window. Taking the level 10's quadkey '0122101332' as an example, 5 is the window size and 1 is the step size. The converted sequence is s = 01221->12210->22101->21013->10133->01332. Then, after embedding the sequence with an embedding matrix, the dependencies of the sequence are learned by self-attention. Last the encoded results are output through a connection and an average pooling layer.

(b) Embedding

Word2vec[29] is an extensible and effective embedded representation learning method. We use word2vec to embed the location v, category c and encoded quadkey q separately into a low-dimensinonal vector and then concat them. $e_{vi} = v_i W_v, e_{ci} = c_i W_c and e_{qi} = q_i W_q$ where v_i, c_i and q_i are the one-hot representation of the i-th location, category and encoded quadkey respectively. $W_v \in R^{|V|*d_v}, W_c \in R^{|C|*d_c}$ and $W_q \in R^{|Q|*d_e}$ are location id embedding matrix, category embedding matrix and encoded quadkey embedding matrix respectively. $|\bullet|$ represents the number of corresponding feature. The embedding vector is $p_i = concat(e_{vi}, e_{ci}, e_{qi}) \in R^{d_p}$.

In order to clearly simulate the relative access time difference between positions in the trajectory, we are encouraged by the paper[21] to encode timestamp to distinguish the order of the input sequence. The calculation formula is as (4).

$$\emptyset(t) = [cos(w_1 t), sin(w_1 t), \dots, cos(w_k t), sin(w_k t)] \tag{4}$$

$$w_k = 1/10000^{\frac{2k}{d}}$$

where function ϕ represents the temporal encoding, t is the absolute access timestamp of the POI, and d is the location coding vector's dimension. Then we rewrite the vector $p_i = concat(e_{vi}, e_{ci}, e_{qi} + \emptyset(t)) \in R^{d_p}$.

For a user sequence $tr^u = p_{t_i}, \dots, p_{t_L}$, the input of the attention layer is $I^u = I_1, \dots, I_L$ where $I_i = \{p_i, (lat_i, long_i), t_i\}$. p_i is i-th location's embedding vector of user u and t_i is the timestamp.

4.1.2 Attention Layer

(a) Local Spatial-temporal Attention. Self-attention can process longer sequences compared with RNN and we use it to learn the connection between spatial-temporal embedding vectors.

The trajectory length of each user is different, and those distant check-ins have little causal relationship with the user's preference, so we only consider the latest fixed length check-ins. Let L be a fixed length. The input of location vector is a L by d_p matrix as (5).

$$P^u=[p_1,\ldots,p_L]^T \tag{5}$$

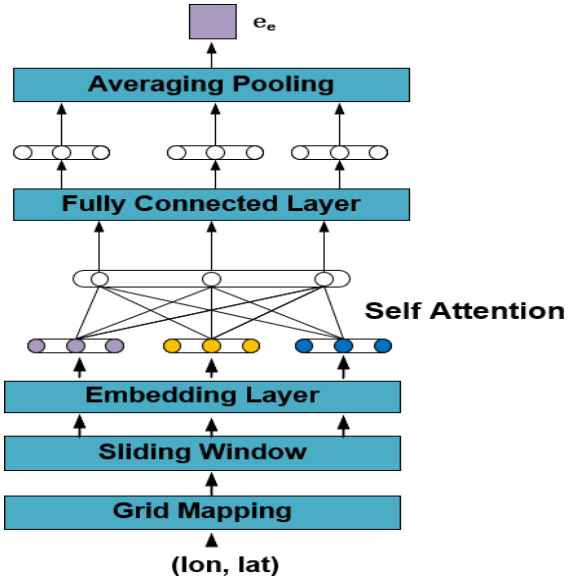

Figure 2. Quadkey Encoder

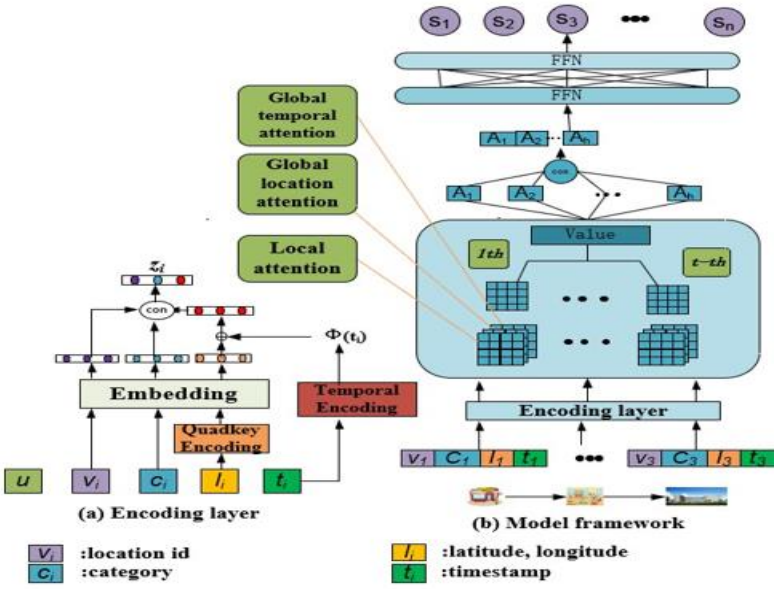

Figure 3. Framework of the proposed STLPF

To get user u's query, key and value , we use non-linear transformation to project P^u to three spaces and are shown in (6).

$$\begin{aligned} Query &= ReLU(P^u W^Q) \\ Key &= ReLU(P^u W^K) \\ Value &= ReLU(P^u W^V) \end{aligned} \tag{6}$$

where $W^Q \in d * d_Q, W^K \in d * d_Q \; and \; W^V \in d * d_Q$ are weight of query, key and value. ReLU is a non-linear activation function. Then, we use the formula (7) to calculate the score. The score measure the similarity between elements. The output is L by L and is shown in (7).

$$S_l^u = softmax(\frac{Query * Key^T}{\sqrt{d_Q}}) \tag{7}$$

By learning all users' check-in sequences, this module can capture spatial-temporal information. However, the spatial-temporal information captured by this module is in a local view because the input p_i of the module only considers its own temporal and spatial information and does not combine spatial-temporal information of other check-ins. In Figure 4 (a), we can see that the module thinks that l_2 is more similar to l_3, even if the distance between l_2 and l_1 is closer than l_3. Modules usually only consider whether the quadkeys of positions are similar rather than the distance between them.

(b) Global Location Attention. In order to make up for this shortcoming, we propose to learn spatial information from a global perspective. We calculate the distance between the current position and all available positions and store it in matrix Dis. Given the coordinates of two POIs of p_i and p_j , the distance between them dis(p_i, p_j) is calculated as (8).

$$\varphi(\frac{dis(p_i,p_j)}{R}) = \varphi(\gamma_i - \gamma_j) + cos(\gamma_i) + cos(\gamma_j)\varphi(\Delta\gamma) \tag{8}$$

where $\varphi(\theta) = (1 - cos(\theta))/2, R \; is \; earth's \; radius, \quad \gamma_i, \gamma_j$ are two locations' latitude and $\Delta\gamma$ is the longitude difference.The L by L matrix *Dis* is calculated in (9).

$$Dis^u = Dis[I^u] \tag{9}$$

Dis_{ij}^u denotes the i-th row and j-th column of Dis^u. It is the spatial distance of γ_i and γ_j of user u.

(c) Global Temporal Attention. The user's check-in behavior is related to time. Therefore, we characterize the time interval to learn the time law of users. The input of temporal attention is T^u and is shown in (10).

$$T^u = [t_1, \ldots, t_L]^T \tag{10}$$

The global temporal attention matrix is seted in (11).

$$\begin{aligned} S_t^u &= MinMaxNorm(D_t^u) \\ D_t^u &= F_T(T^u) \\ D_t^u(i,j) &= t_j - t_i \end{aligned} \tag{11}$$

where D_t^u is a L by L's matrix, MinMaxNorm () is a function that maps the input matrix to a closed

interval [0, 1] and $F_T(T^u)$ is a function to map T^u to D_t^u.

(a)Local View: Learning similarities from Tile Map System (b) **Global View: Learning similarities from distances**

Figure 4. Learning spatial information from position coordinates

We linearly combine the local attention score with the temporal attention score.

$$S^u = S_l^u + \beta S_t^u + \gamma Dis^u \tag{12}$$

The output S^u is a matrix about location vector affected by time feature and hyperparameter β indicates the influencing degree. Then, the output from the attention layer is shown in (13).

$$A^u = softmax(S^u) \cdot Value \tag{13}$$

where A^u is a matrix and its shape is L by d_Q and $Value$ is calculated in formula(6). We normalize the softmax function to change the score into a probability and highlight important elements. The multi-head attention is used by us to form m subspaces to fully explore the hidden information. The results we obtained are $A_1^u, \ldots, A_m^u$.

4.1.3 Prediction Layer

Finally, we concatenate the m results. We apply the linear transformation and result is calculated in (14):

$$R^u = concat(A_1^u, \ldots, A_m^u) \cdot W^A \tag{14}$$

where $W^A \in R^{md_Q * d}$ and d_Q=d/m. The output R^u is a L by d matrix. The prediction results are defined in (15).

$$y(v_{L+1}^u | R^u) = softmax(R^u W_p^T + b) \tag{15}$$

where $y(v_{L+1}^u | R^u)$ is a probability prediction of next location. Compared with previous work, we not only share weights W_p, but also dynamically represent historical trajectories. This method not only reduces over fitting, but also retains more information.

4.2 Loss Function

Given the trajectory $Traj_m$, where $Traj_m$ represents the trajectory sequence of the m-th user, the probability score of user m visiting the k-th location at the j-th moment is denoted as $y_{j,k}$. In [7, 8], the objective function is cross-entropy loss. The cross-entropy loss is not very efficient in which it only considers positive samples. The performance decreases as the number of candidate locations increases. Existing location recommendation algorithms usually use binary cross-entropy [9, 10] defined in (16).

$$-\sum_{u_i \in U} \sum_{j=1}^{L^{u_i}-1} (logs(y_j, o_j) + log(1 - sy_{j,k})) \tag{16}$$

where U is users' training set, o_j is the target location at the j-th moment and L^{u_i} is the trajectory's length of user i.

The binary cross-entropy loss function contains a positive sample and a randomly negative sample. The negative sample is sampled from the unvisited location and there is a high probability that negative samples with important information are ignored, so it is hard to reduce the loss of binary cross-entropy.

We can see that locations unvisited with large preference scores $y_{j,k}$ contribute more to the gradient and we refer to these locations as hard negative samples. However, it is not feasible to consider the top-k unvisited sites with the highest preferred score as a hard negative sample because of false negative. Previous research has shown that users' check-in behavior is spatially clustered and users prefer to visit nearby places than to travel long distances. We use the k-nearest neighbor algorithm to select hard negative samples.[30] proposed models trained with hard negative samples is not the best method and the use of easy negative samples is still necessary. It is beneficial to mix easy and hard negative samples in training. We can improve the model's accuracy by increasing the ratio of easy and hard negative sample. Our loss function is defined in (17).

$$-\sum_{u_i \in U} \sum_{j=1}^{L^{u_i}-1} (logs(y_j, o_j) + \sum_{k=1}^{K} log(1 - sy_{j,k}) + \sum_{e=1}^{E} log(1 - sy_{j,e})) \tag{17}$$

where the second term is for picking a hard negative sample set and the third term is for picking a easy negative sample set. Inspired by the paper[6],we weighted the sample fraction of the loss function. We redefine the loss function in (18).

$$-\sum_{u_i \in U} \sum_{j=1}^{L^{u_i}-1} w_k (logs(y_j, o_j) + \sum_{k=1}^{K} log(1 - sy_{j,k}) + \sum_{e=1}^{E} w_e log(1 - sy_{j,e})) \tag{18}$$

where $w_e = \frac{exp(r_{j,e})}{\sum_{e=1}^{E} exp(r_{j,e})}$ and $w_k = \frac{exp(r_{j,k})}{\sum_{k=1}^{K} exp(r_{j,k})}$ are the weight of e-th and k-th sample. Among the samples, the larger the preference score and the greater the weight.

In the selection of hard negative samples for sequence position prediction, we first use the K-nearest neighbor algorithm to select K positions which are closest to the target position, and then conduct random sampling from these K positions to form hard negative samples. Experiments show that the increase of hard negative samples does not improve the performance of prediction. Then we select easy negative samples from other regions according to a certain ratio factor.

4.3 Adaptive Personalized Federated Learning Framework(APF)

Federated averaging focuses on training a single global model, but an optimized global model cannot guarantee the best performance in every client and federated averaging algorithm cannot obtain a global model with good performance when the client data is heterogeneous. Moreover, due to the nature of federated average algorithm, the gradient of the global model will be biased to

the client with large amount of data. Therefore, it is necessary for each client to obtain a personalization model with good performance by collaborating with other clients.

We rewrite (3) in (19).

$$s_k^t = \gamma_{t-1,1} w_1^{t-1} + \ldots + \beta_{t-1,k} w_1^{t-1} + \ldots + \gamma_{t-1,m} w_m^{t-1} \tag{19}$$

where $\gamma_{t-1,1} + \ldots + \beta_{t-1,k} + \ldots + \gamma_{t-1,m} = 1$ and s_k^t represents the personalization model of the t-th round's k-th client. It is obvious that the personalization model parameter of the k-th client can be viewed as a linear weight aggregation of all the other model parameters. Inspired by [31], $\gamma_{t-1,j}$ is denoted in (20)

$$\gamma_{t-1,j} = \frac{A(w_k^{t-1}, w_j^{t-1}) \cdot (1 - \beta_{t-1,k})}{\sum_{j \neq k}^{n} A(w_k^{t-1}, w_j^{t-1})} \tag{20}$$

where $j \neq k$ and $A(w_k^{t-1}, w_j^{t-1})$ is a function to measure the relationship between w_k^{t-1} and w^{t-1}.In our algorithm, we use the similarity function to measure the relationship between the two gradients. The more similar the two gradients are and the greater the weight. We encourage clients with more similar data distributions to have greater internal collaboration capabilities. Considering that the dimensions of the model are large, there is no advantage in using Euclidean distance to measure similarity. We adopt pearson similarity $p(w_k^{t-1}, w_j^{t-1})$ to measure the relationship between two gradients and add an exponential term to this function.

We rewrite (20) in (21).

$$s_k^t = \beta_{t-1,k} w_k^{t-1} + \ldots + (1 - \beta_{t-1,k}) z_k^{t-1} \tag{21}$$

We view the personalization model s_k^t as a weight combination of client k and other clients, and z_k^{t-1} is an intermediate value that is weighted by other model parameters other than k. We define z_k^{t-1} in (22).

$$z_k^{t-1} = \frac{A(w_k^{t-1}, w_j^{t-1}) \cdot w_j^{t-1}}{\sum_{j \neq k}^{n} A(w_k^{t-1}, w_j^{t-1})} \tag{22}$$

The details of our algorithm are summarized in Algorithm 1. In the t-th iteration, the clinet k initializes the parameters to s_k^t sent by the server and performs random gradient descent locally. After the local model converges, the parameters are sent to server. The central server calculates the corresponding personalized parameters for each client through (21) and (22) and sends them to each client. The algorithm will be iterative until the communication rounds reaches T.

5. Experiments

Firstly, we describe the data sets required for the experiment firstly and then introduce the baseline methods to be compared, the evaluation indicators and the experimental environment. In addition, we compare the location prediction method with the baseline and then compare the loss function and finally compare our personalization framework with the federated average algorithm.

Algorithm 1: APF:Adaptive Personalized Federated Learning Framework

Input: The total number of clients m, $w_1^0 = ... = w_m^0 \in R^d$, total communication rounds T, number of client local iterations N

Output: $w_1^T, ..., w_m^T$

for *t=1,...,T* **do**
  if *t=1* **then**
    for *client i=1,...,m* **do**
      $w_k^1 = \arg\min_{w \in R^d} F_k(w)$
      client i uploads the local model w_k^1 to the central server
    end
  else
    for *every client i=1,...,m, central server* **do**
      server computes the aggregation of w_k^{t-1} by

$$z_k^{t-1} = \frac{\sum_{j \neq k}^{m} e^{\sigma r(w_k^{t-1}, w_j^{t-1})}) \cdot w_k^{t-1}}{\sum_{j \neq k}^{m} e^{\sigma r(w_k^{t-1}, w_j^{t-1})}}$$

      server computes the personalized model s_k^t by

$$s_k^t = \beta_{t-1,k} w_k^{t-1} + (1 - \beta_{t-1,k}) z_k^{t-1}$$

      server sends the model s_k^t to the client k
    end
    for *client k=1,...,m* **do**
      client k iterates N times to computes its local model w_k^t by $w_k^t = \min_w F_i(w) + \lambda ||w - s_k^t||^2$
      client k uploads the local model w_k^t to the central server
    end
  end
end

5.1 Datasets

We evaluate ours with state-of-the-art methods on two check-ins datasets from the Foursquare dataset [32]. This dataset contains check-ins in NYC and Tokyo collected for about 10 months (from 12 April 2012 to 16 February 2013). It contains 227,428 check-ins in New York City and 573,703 check-ins in Tokyo. For each city dataset, we removed users who's check-in less than 25. Table 1 shows the basic information of data preprocessing. We use the first 80% of each dataset as the training set of our method and last 20% as the testing set.

Table 1. Statistics of datasets.

	Foursquare	
City	**New York**	**Tokyo**
Users	**1082**	**2293**
Locations	**34440**	**56106**
Check-ins	**184509**	**449640**

5.2 Baselines(Location prediction)

To verify the validity of our location prediction model, we compared it to some baselines:

GRU [5], which encodes the trajectory and context information, and uses the sequence model GRU to learn the correlation between trajectories, and then predict the next location probability.

ST-RNN [15], which is a spatial temporal RNN model to predict next location. It can make use of different time intervals and different geographical distances to model Distance- specific transition matrices.

MCARNN [33], which is a multi-task context aware recurrent neural network. It integrates sequential dependence and temporal regularity of spatial activities to conduct location prediction .

- **VANext [28]**, which is based on variational attention to conduct the next POI prediction. It captures the underlying characteristics of recent mobility and then searches for similar historical traces of periodic patterns.

- **STSAN [34]**, which is based on self-attention to predict the next POI. It designs trajectory attention module to capture dynamic temporal and spatial information.

5.3 Evaluation Metric

Give a user's trajectory, location prediction aims at predicting the user's next position. Given the user set U and the candidate locations set V, for each user, given the previous locations, we predicted the next POI. The prediction accuracy is calculated in formula(23).

$$Acc@k = \frac{1}{|U|}\sum_{u_i \in U}\frac{1}{L^{u_i}-1}\sum_{t=2}^{L^{u_i}} l(v_t \in s_t^{u_i}(k)) \tag{23}$$

where $s_t^{u_i}(k)$ is a set of top-k locations of user u_i in time step t, u_t is real visited location for user u_i in time step t, L^{u_i} is user u_i 's trajectory length. The value of $l(v_t \in s_t^{u_i}(k)$ is 1 when se $s_t^{u_i}(k)$ contains v_t and the value is 0 when v_t is not in set $s_t^{u_i}(k)$.

Table 2 .Performance compared with the baselines

	Metric	GRU	ST-RNN	MCARNN	VANext	STSAN	Ours	Improv.
NYC	*Acc@1*	0.2347	0.2518	0.2963	0.3578	0.3653	0.4067	10.18%
	APR	0.9218	0.9325	0.9673	0.9765	0.9795	0.9893	1.00%
Tokyo	*Acc@1*	0.2213	0.2458	0.2796	0.3479	0.3561	0.3887	9.15%
	APR	0.9146	0.9358	0.9529	0.9712	0.9786	0.9856	0.72%

We also apply Average Percentage Rank (APR) to assessment the model's performance and is calculated in formula(24).

$$APR = \frac{1}{|U|}\sum_{u_i \in U}\frac{1}{L^{u_i}-1}\sum_{t=2}^{L^{u_i}}\frac{|V|-rank^{u_i}(v_t)+1}{|V|} \tag{24}$$

where $rank^{u_i}(v_t)$ denotes candidate location $v_t's\ rank\ for\ user\ u_i$ in time step t.

5.4 Setup

For each user, we use the first 80% sub-trajectory as the training data, and the remaining 20% as the test data. We implemented our model with Pytorch. All experiments were conducted on an NVIDIA 2080ti GPU on the Ubuntu system. In our local prediction model, we set the level of the quadkeys to 17, and we first generate sequences of the quadkeys with a sliding window method of 7 window size and 1 step size. We use *RmsProp* [35] optimization algorithm. In our new loss function, we sampled 4 hard negative samples and then sampled 28 easy negative samples in a ratio of 1:7.

5.5 Performance comparison of local models

We evaluate our model with the baselines on two check-ins datasets. The experimental results are summarized in Table 2. We found that ours outperformed all of the comparative baselines on the two datasets. We improved by 10.18% and 1.00% over the best baseline at *Acc*@1 and *APR* on New York dataset, respectively. In theory, RNN can capture long-term time dependencies. However, there may be a problem of gradient disappearance or outbreak in the actual long-term time, GRU is used to alleviate this problem. ST-RNN is a spatiotemporal recursive neural network, which integrates spatial-temporal and spatial-temporal context information. However, it does not explicitly model the user's historical access patterns. STRNN uses recursive neural networks to capture long-term POI dependencies, but because of the gradient disappearance problem inherent in recursive networks, it cannot handle very long historical trajectories well. MCARNN outperforms the previous two models in which it can handle decaying sequence dependencies and incorporate temporal contexts. Vanext is a strong baseline that performs well on both datasets, demonstrating the effectiveness of the variational attention architecture. Vanext effectively captures short-term human movement patterns and utilizes the CNN network and variational attention to generate better representations of his- torical trajectories. STSAN designs a trajectory attention module to extract dynamic temporal and spatial information. But it does not solve the sparsity problem of longitude and latitude. It achieves the best performance over previous methods.

5.6 Performance comparison of loss function

For hard negative samples, we used KNN uniform sampler to sample. We sampled 4 hard negative samples and then uniformly sampled 28 easy negative samples with a scale factor of 1:7. We compared our loss function with NLL loss function and binary loss function in the same experimental environment. We found that our loss function was superior to the other two loss functions. The experimental results are shown in Figure 5.

Our loss function mixes hard and easy negative samples and makes full use of the information of negative samples. The NLL loss function only focuses on positive samples, and its optimization ability will gradually decrease with the increase of locations. The binary loss function samples only one negative sample from the unvisited location, and it cannot make full use of the large amount of information from the unvisited location.

5.7 Performance comparison of federated learning framework

We compare our adaptive personalized federated learning framework to federated average and non-federated learning. Non-federated learning, where data is gathered together for training, represent the performance upper bound of federated learning methods for location

prediction tasks. We compare $ACC@1$ and APR with federated averaging algorithm and non-federated learning respectively. The comparison of losses and $Acc@1$ are shown in Figure 6. Our personalized federated learning framework has better performance on $ACC@1$ compared to the federated average. Federated averaging algorithm focuses on the training of a single global model, the gradient of the global model will be biased to the client with large amount of data, and it cannot adapt to the situation of heterogeneous data. Our personalized federated learning framework improves collaboration efficiency between clients by encouraging stronger collaboration among clients with similar data distributions and performs better than federated averaging algorithms on heterogeneous data.

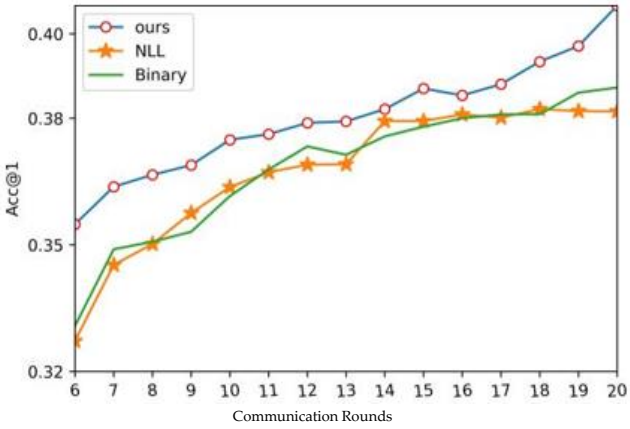

Figure 5. The $Acc@1$ of using different loss functions in New York dataset

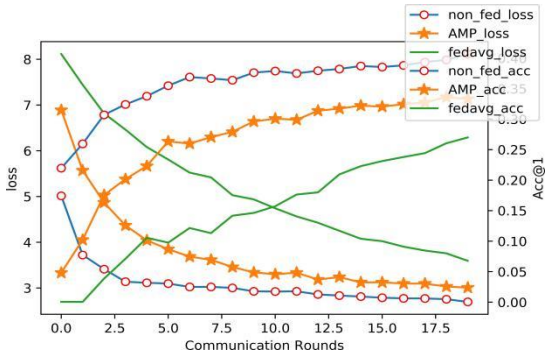

Figure 6. The $Acc@1$ and $loss$ of using different federated learning framework in New York dataset

6. Conclusion

In this paper, we first propose a spatial-temporal prediction network, which can handle the sparsity problem of geographic information well and make full use of the local and global temporal and spatial information to mine the user's trajectory and alleviate the data sparsity problem. Experimental results show that this method is superior to other baseline methods and has good

performance on sparse data sets. Secondly, we propose a new loss function that optimizes the model by combining hard samples with easy samples. Finally, we propose an adaptive personalized federated learning framework, which address the heterogeneity of data and protects the privacy of clients by encouraging stronger collaboration among clients with similar data distributions. Future work will include the pretraining of the quadkey encoder and the designing of automatically selecting the ratio of easy negative samples and hard negative samples.

7. Acknowledgement

This research is partially funded by Natural Science Foundation of Liaoning Province (Grant no.2019-MS-111) and the National Natural Science Foundation of China (Grant no.61602102, 61872069).

Compliance with ethical standards

Conflict of interest. The authors declare that there is no conflict of interest regarding the publication of this paper.

Ethical approval. This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent. Informed consent was obtained from all individual participants included in the study.

References

[1] A. Asahara, K. Maruyama, A. Sato, K. Seto, Pedestrian-movement prediction based on mixed markov-chain model, in: Proceedings of the 19th ACM SIGSPATIAL international conference on advances in geographic information systems, 2011, pp. 25–33. doi:10.1145/2093973.2093979.

[2] S. Rendle, C. Freudenthaler, L. Schmidt-Thieme, Factorizing personalized markov chains for next-basket recommendation, in: Proceedings of the 19th international conference on World wide web, 2010, pp. 811–820. doi:10.1145/1772690.1772773.

[3] Q. Liu, S. Wu, L. Wang, T. Tan, Predicting the next location: A recurrent model with spatial and temporal contexts, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 30, 2016.

[4] Hochreitersepp, Schmidhuberjurgen, Long short-term memory, Neural Computation 9 (8) (1997) 1735–1780. doi:10.1162/neco.1997.9.8.1735.

[5] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, arXiv preprint arXiv:1412.3555 (2014).

[6] D. Lian, Y. Wu, Y. Ge, X. Xie, E. Chen, Geography-aware sequential location recommendation, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Vol. abs/1412.3555, 2020, pp. 2009–2019.

[7] R. Li, Y. Shen, Y. Zhu, Next point-of-interest recommendation with temporal and multi- level

context attention, in: 2018 IEEE International Conference on Data Mining (ICDM), IEEE Computer Society, 2018, pp. 1110–1115. doi:10.1109/ICDM.2018.00144.

[8] P. Zhao, A. Luo, Y. Liu, F. Zhuang, J. Xu, Z. Li, V. S. Sheng, X. Zhou, Where to go next: A spatio-temporal gated network for next poi recommendation, IEEE Transac- tions on Knowledge and Data Engineering (2020) 5877–5884doi:10.1609/aaai.v33i01. 33015877.

[9] W.-C. Kang, J. McAuley, Self-attentive sequential recommendation, in: 2018 IEEE International Conference on Data Mining (ICDM), IEEE Computer Society, 2018, pp. 197–206. doi:10.1109/ICDM.2018.00035.

[10] J. Li, Y. Wang, J. McAuley, Time interval aware self-attention for sequential recommendation, in: Proceedings of the 13th International Conference on Web Search and Data Mining, ACM, 2020, pp. 322–330. doi:10.1145/3336191.3371786.

[11] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al., Advances and open problems in federated learning, arXiv preprint abs/1912.04977 (2019).

[12] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial Intelligence and Statistics, Vol. 54, PMLR, 2017, pp. 1273–1282.

[13] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, V. Smith, Federated optimization in heterogeneous networks (2020).

[14] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, V. Chandra, Federated learning with non-iid data, arXiv preprint arXiv:1806.00582 (2018).

[15] X. Li, K. Huang, W. Yang, S. Wang, Z. Zhang, On the convergence of fedavg on non-iid data, arXiv preprint arXiv:1907.02189 (2019).

[16] V. Mnih, N. Heess, A. Graves, K. Kavukcuoglu, Recurrent models of visual attention, arXiv:1406.6247 (2014).

[17] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473 (2014).

[18] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, B. Xu, Attention-based bidirectional long short-term memory networks for relation classification, in: Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers), The Association for Computer Linguistics, 2016, pp. 207–212. doi:10.18653/v1/p16-2034.

[19] M.-T. Luong, H. Pham, C. D. Manning, Effective approaches to attention-based neural machine translation (2015) 1412–1421 doi:10.18653/v1/d15-1166.

[20] W. Yin, H. Schütze, B. Xiang, B. Zhou, Abcnn: Attention-based convolutional neural network for modeling sentence pairs, Transactions of the Association for Computational Linguistics 4 (2016) 259–272.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser,I. Polosukhin, Attention is all you need, arXiv:1706.03762 (2017).

[22] C. Zhou, J. Bai, J. Song, X. Liu, Z. Zhao, X. Chen, J. Gao, Atrank: An attention-based user behavior modeling framework for recommendation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, 2018.

[23] S. Zhang, Y. Tay, L. Yao, A. Sun, Dynamic intention-aware recommendation with self-attention, arXiv:1808.06414 (2018).

[24] S. Feng, X. Li, Y. Zeng, G. Cong, Y. M. Chee, Personalized ranking metric embedding for next new poi recommendation, in: IJCAI'15 Proceedings of the 24th International Conference on Artificial Intelligence, ACM, 2015, pp. 2069–2075.

[25] A. Al-Molegi, M. Jabreel, B. Ghaleb, Stf-rnn: Space time features-based recurrent neural network for predicting people next location, in: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2016, pp. 1–7. doi:10.1109/SSCI.2016.7849919.

[26] D. Kong, F. Wu, Hst-lstm: A hierarchical spatial-temporal long-short term memory network for location prediction., in: IJCAI, Vol. 18, ijcai.org, 2018, pp. 2341–2347. doi:10.24963/ijcai.2018/324.

[27] J. Feng, Y. Li, C. Zhang, F. Sun, F. Meng, A. Guo, D. Jin, Deepmove: Predicting human mobility with attentional recurrent networks, in: Proceedings of the 2018 world wide web conference, ACM, 2018, pp. 1459–1468. doi:10.1145/3178876.3186058.

[28] Q. Gao, F. Zhou, G. Trajcevski, K. Zhang, T. Zhong, F. Zhang, Predicting human mobility via variational attention, in: The World Wide Web Conference, ACM, 2019, pp. 2750– 2756. doi:10.1145/3308558.3313610.

[29] G. C. Tomas Mikolov, Kai Chen, J. Dean, Distributed representations of words and phrases and their compositionality, in Proceedings of the Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013 (2013) 3111–3119.

[30] J.-T. Huang, A. Sharma, S. Sun, L. Xia, D. Zhang, P. Pronin, J. Padmanabhan, G. Ot- taviano, L. Yang, Embedding-based retrieval in facebook search, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, ACM, 2020, pp. 2553–2561. doi:10.1145/3394486.3403305.

[31] Y. Huang, L. Chu, Z. Zhou, L. Wang, J. Liu, J. Pei, Y. Zhang, Personalized cross-silo federated learning on non-iid data, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2021.

[32] D. Yang, D. Zhang, V. W. Zheng, Z. Yu, Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns, IEEE Transactions on Systems, Man, and Cybernetics: Systems 45 (1) (2015) 129–142. doi:10.1109/TSMC.2014.2327053.

[33] D. Liao, W. Liu, Y. Zhong, J. Li, G. Wang, Predicting activity and location with multi- task

context aware recurrent neural network, in: IJCAI, ijcai.org, 2018, pp. 3435–3441. doi:10.24963/ijcai.2018/477.

[34] S. Wang, A. Li, S. Xie, W. Li, B. Wang, S. Yao, M. Asif, A spatial-temporal self-attention network (stsan) for location prediction, Complexity 2021 (2021) 6692313:1–6692313:13. doi:10.1155/2021/6692313.

[35] Basu, A., Soham D, Anirbit M, Enayat U. Convergence guarantees for RMSProp and ADAM in non-convex optimization and their comparison to Nesterov acceleration on autoencoders. CoRR abs/1807.06766(2018).