IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received January 28, 2020, accepted March 20, 2020, date of publication March 26, 2020, date of current version April 13, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2983555*

# Multi-Context-Aware Location Recommendation Using Tensor Decomposition

**JING LU AND MARTIN A. INDECHE**
School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China

Corresponding author: Jing Lu (jing.lu@usst.edu.cn)

This work was supported in part by the General Program of National Natural Science Foundation of China under Grant 61772342, and in part by the National Natural Science Foundation Youth Project under Grant 61703278.

**ABSTRACT** With the rapid growth of information generated by online social network platforms and the increased usage of Location-Based Social Networks, location recommendation research has attracted more attention both in academic and industry. However, the problem of data sparsity still posses a severe challenge to the existing location recommendation methods. Moreover, extracting and modeling multiple contextual information, which is one of the key factors that influences user check-in preferences, is another big challenge faced by the existing methods. Many of the existing location recommendation methods have low accuracy because they utilize limited contextual information when modeling user check-in behaviors. In this paper, we propose a Multi-Context-aware Location Recommendation using Tensor Decomposition (MCLR-TD) approach that incorporates multiple context information at different granularity scales in modeling user check-in behavior. We use a four mode tensor to model the relationship among the four dimensions: users, locations, time and weather. In order to reduce the data sparsity problem, we further construct four feature matrices that are collaboratively decomposed with the tensor. We carry out extensive experiments on two real-world datasets collected from Foursquare and Yelp and the results demonstrate the effectiveness of our approach.

**INDEX TERMS** Context information, contextual preference, location recommendation, tensor decomposition.

## I. INTRODUCTION

With the huge amount of information available on the web, many people regularly face the problem of ''choice paralysis''. It is very hard for them to make a satisfying decision on which locations to visit from the huge number of Point of Interests (POIs). POI Recommendation Systems try to solve this problem by utilizing users' historical check-in data available in Location Based Social Networks (LBSNs) and recommending POIs that users may be interested in. This is achieved by taking advantage of the increased usage of mobile devices that store and provide huge amount of users' check-in information alongside LBSNs like Foursquare, Brightkite and Yelp. POI recommendation can help users to explore interesting but unvisited locations of a given region and enrich their experience. POI recommendation is also very important to the location based advertisers as it can help them acquire more potential customers [1].

Human Check-in behavior is always influenced by multiple factors such as user preference [2]–[4], social influence [5]–[9] and geographical influence [2], [10]–[12]. Temporal context such as time is another key factor considered in many recommendation approaches [13]–[15]. People prefer checking in different POIs under different contextual scenarios [16], [17]. For example, users will prefer checking in outdoor activities during afternoon hours when the weather is favorable. Recommendation systems that incorporate contextual information have proved to perform better than those that does not consider it [18]. However, modeling user contextual preference is still a key problem in Context Aware Recommendation Systems (CARSs). This is because of two reasons; 1) Lack of context data: many of the available check-in datasets lack information about the exact contextual scenario under which a check-in instance happened. Thus the context information can only be acquired

The associate editor coordinating the review of this manuscript and approving it for publication was Yin Zhang.

through inference. 2) Incorporating multiple contexts requires high level modeling, hence many traditional recommendation methods stick to one context variable that is easy to handle. However, considering only one contextual factor leads to low accuracy in recommendation.

Although there has been a lot of research on location recommendation methods, many challenges still limit the performance of the existing POIs recommendation systems. For instance, one key challenge is the data sparsity problem [19], [20]. There is a huge number of locations available in LBSNs, however users have only visited a very limited number of locations, hence the user location matrix is very sparse. Another big challenge is how to effectively learn user's contextual preference by considering multiple contextual factors and combining information from heterogeneous sources.

To solve the problems discussed above, we propose a Multi-Context-aware Location Recommendation using Tensor Decomposition (MCLR-TD) approach. We model user check-in behaviors by considering two contextual information; time and weather. We first carry out extensive data analysis on the two datasets to learn how user check-in behavior is affected by the two contexts at different scales. We then construct a four mode tensor to model the relationship among users, locations, time and weather. In order to deal with data sparsity problem, we further construct four feature matrices that are collaboratively decomposed with the four mode tensor to obtain predicted scores. Finally, we carry out extensive experiments to compare the performance of MCLR-TD model with some existing POI recommendation methods.

Compared with the work in [21], [22] that also considered temporal context, we model temporal context at three granular levels namely, hourslot of the day, day of the week and season of the year. We also consider the influence of weather context on user check-in behaviors. Our approach further models the interaction between users, locations, time and weather by utilizing interaction matrices between these entities.

In summary, the main contributions of our work are as follows:

1) We carry out extensive data analysis on the two datasets to determine how user check-in preference is affected by the two contexts at different scales.
2) We construct a four mode tensor to model the relationship among users, locations, time and weather simultaneously.
3) We construct four feature matrices that are collaboratively decomposed with the four mode tensor to solve the data sparsity problem. The feature matrices models the interaction between different entities.
4) We evaluate our approach on two large-scale real world data sets collected from Yelp and Foursquare and the results obtained shows the effectiveness of our approach.

The remainder of this paper is organized as follows: Section II summarizes related works in location recommendation domain. Section III formulates location recommendation problem and describes the datasets used in the experiments. Section IV presents the proposed MCLR-TD approach. Section V discusses the experiment settings and the results. Section VI summarizes our work and presents some future research directions.

## II. RELATED WORK

POI recommendation has been widely researched by the academic community. In this paper, we mainly focus on previous studies that apply Matrix Factorization (MF) and Tensor Factorization (TF) based models.

### A. MATRIX FACTORIZATION BASED METHODS

MF models have widely been applied in location recommendation systems. MF models utilize two dimensional user-location check-in matrices as the input data when modeling user check-in preferences. Some of the existing MF based location recommendation methods are discussed below.

The work in [13] proposed a unified framework UTE+SE that extended user-based Collaborative Filtering (CF) by leveraging the time factor when computing the similarity between users. UTE+SE model integrated both temporal and spatial factors when modeling user's temporal check-in preference. However, UTE+SE model considered only hour of the day as the temporal context but ignored other time dimensions like day of the week, season and other context factors. The work in [14] divided user check-in data into 24 matrices based on two temporal properties of user's daily check-in preferences: non-uniformness and consecutiveness. They computed similarity among users before generating the recommendation list.

The work in [23] proposed two unified location recommendation models that incorporated spatial, textual and temporal information. Both models extend the item-based CF and user based CF model by utilizing the proximity factor when calculating user similarity. However, their models only considered the month of the year when modeling the temporal influence, thus may not fully capture the changes in user preference within a short period of time.

The work in [24] proposed GeoMF++ model that applied MF to jointly model the geographical influence and implicit feedback. The model used two dimensional kernel density estimation to capture the spatial clustering phenomenon from user's historical check-in data. However, GeoMF++ models user location preferences by only considering the spatial information, but ignores the influence of temporal context and other factors.

The work in [25] proposed TGSC-PMF method that utilized geographical distance, category information, location popularity and social relations to recommend locations to users. They applied Latent Dirichlet Allocation (LDA) model to learn user topic preferences and location topics. However, they never mentioned how to handle the data sparsity problem which may be prevalent when very few users leave comments on locations. They also ignore the influence of other context information like time, weather etc.

In summary, MF models have limited applications in CARSs because they can only handle data in two dimensions. Many models that apply matrix factorization approach ignore the influence of contextual factors like time and other implicit factors leading to poor performance.

### B. TENSOR FACTORIZATION BASED METHODS

Recently, tensor decomposition methods have been widely used in recommendation systems [26], [27]. This is due to their ability to handle multi-dimensional data. Using TF model can best solve the problem of matrix factorization because multiple dimensions can be modeled together to provide more useful information to the recommendation systems.

The work by [28] proposed a method for location recommendation by incorporating time context using TF model. They modeled user-location-time relations using a three-mode tensor and extracted user and location similarity matrices. In order to increase efficiency of their model, they also proposed Threshold Algorithm (TA). However, their model divided time into 24 hour slots which is hard to capture users temporal preference because a user will have very limited or no check-ins within one hour. Furthermore, their model did not utilize other context factors.

The work in [29] proposed a model that exploited category information using a TF model. They obtained latent factors using Stochastic Gradient Descent. However, they did not consider the influence of temporal context on category transitions. The work by [30] proposed a location recommendation approach that modeled user's time aware topic preferences using TF. They used LDA model to extract user topic distribution from comments and constructed a three mode tensor to represent temporal user topic distribution. They converted user topic preferences to user POI preference to generate recommendation list. However, they modeled different scales of temporal context separately and ignored the influence of other context factors such as weather.

The work in [31] proposed an Aggregated Temporal Tensor Factorization (ATTF) method that modeled three temporal aspects of human check-in behavior namely: periodicity, consecutiveness, and non-uniformness. ATTF model aggregated the contribution of different temporal latent features at different time scales i.e. hour, week, day, and month through linear combination. However, this model does not consider the effect of other context such as weather and also faces data sparsity problem due to limited check-ins within one hour.

The work in [21] proposed a Collaborative Filtering approach using a three mode tensor to model user check-in preferences by incorporating spatial influence, temporal dependency and social constraints. However, their method only considered temporal information based on hour of the day granularity and ignored the effect of other time dimensions such as day of the week and season of the year and also other contexts information like weather and category influence.

The work in [32] proposed a Collaborative Tensor Factorization (CTF) method for recommending points of interest (POIs) using a 3-mode tensor with three feature matrices. They used an element-wise gradient descent optimization algorithm to solve the formulation problem. The work in [22] improved the CTF model by proposing a Partition-based Collaborative Tensor Factorization (PCTF). PCTF partitioned the original tensor into small sized tensors using a clustering algorithm before carrying out decomposition. However, both works divided a day into 24 time slots leading to high data sparsity because. Moreover, they do not consider other time dimensions like day of the week and seasons which also influence user's mobility patterns.

Compared with the existing location recommendation methods, our work differentiates itself by using a four mode tensor to model user check-in behavior under multiple contexts i.e. time and weather simultaneously. The two contexts are modeled at different scales of granularity. We capture the influence of each context on both locations and users by carrying out collaborative decomposition of the tensor and the feature matrices. We also carry out separate experiments to determine the impact of each feature matrix on our model performance. Our model could also be easily extended to incorporate more contextual dimensions without loss of generality.

## III. PRELIMINARIES

### A. PROBLEM STATEMENT

Given a set of users $U = \{u_1, u_2, \ldots, u_N\}$, a set of locations $L = \{l_1, l_2, \ldots, l_M\}$, and the current context $C = \{c_1, c_2, \ldots, c_K\}$, where $N, M$ and $K$ denote the number of users,locations and contexts respectively, the recommendation system should return a list of top-K $\{1, 2, \ldots, K\}$ locations to the given user based on the user's current context. The ranking order of locations in the final output list must reflect user's preference in the current context. The Top-K locations are defined as shown in equation 1.

$$Top(u, c, K) = arg^K_{l \in L} MaxR_{(u,l,c)} \tag{1}$$

where $R$ denotes the predicted score of user $u$ checking in location $l$ under context $c$. $K$ denotes the number of locations to be recommended.

### B. DATASET DESCRIPTION

We used two real-world datasets of two cities collected from Yelp[1] and Foursquare [33] to carry out extensive experiments on our proposed model. Each check-in record in the datasets contains a unique identifier of the user, POI with longitude

[1] https://www.yelp.com/dataset

and latitude coordinates and the check-in timestamp. A sample check-in record obtained from yelp dataset is as follows.

{userID: 897;
LocationID: 4bf58dd8d48988d1e0931735;
Category: Coffee Shop,
Longitude: −73.974;
Latitude: 40.752;
Time:Tue April 03 18:04:38 2012}.

Table 1 shows the summarized statistical information of the check-in dataset. Historical weather details for the two datasets were crawled from the World Weather Online API[2] for each of the {latitude,longitude,time} triplet available in the dataset. The weather data collected include temperature, humidity, pressure, wind speed, cloud cover, moon phase and heat index. We also collected daily weather summary tags. They include overcast, sunny, cloudy, rainy, drizzle, fog and mist.

**TABLE 1. Dataset statistics summary.**

| Dataset | #Users | #POIs | #Check-ins | Region | Density |
|---|---|---|---|---|---|
| Foursquare | 1083 | 38648 | 227428 | Tokyo | 0.0729 |
| Yelp | 2293 | 555 | 573703 | New York | 0.873 |

## IV. MCLR-TD METHODOLOGY

Figure 1 shows the framework of our proposed MCLR-TD approach. It consists of four main parts: 1) Context inference and modeling; 2) tensor and feature matrix construction; 3) Collaborative tensor decomposition; 4) POI recommendation.

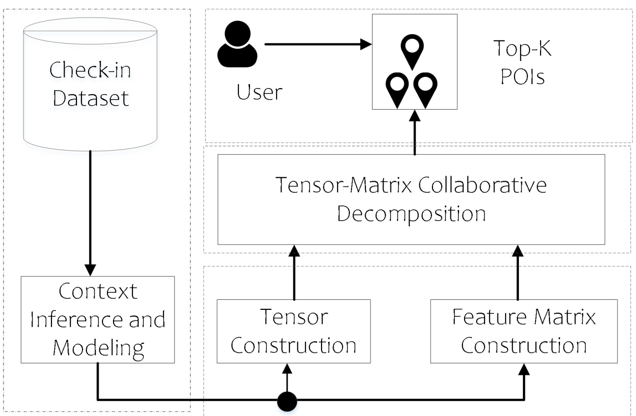



**FIGURE 1. MCLR-TD system framework.**

### A. CONTEXT INFERENCE AND MODELING

Context inference and modeling is a very critical stage in CARSs. Users in LBSNs check-in different POIs under different contexts. We study the distribution of user check-ins in different time and weather context scenarios and select context factors that display high check-in distribution variance. This improves the performance of our system because incorporating all context factors without determining which factors are relevant may adversely affect the accuracy of the recommendation system and increase the number of computations.

#### 1) TIME CONTEXT MODELING

User activities are periodical. Users normally visit different locations during different time periods [13]. For example, people usually go to restaurants during lunch and dinner hours, and they go to nightlife spots like bars at night. In order to capture user check-ins preference in different time periods, we split time context into three different levels namely, season, day of the week and hourslot of the day. Every user check-in instance has a unique timestamp, hence, there are very huge numbers of timestamps in our check-in dataset. In order to reduce the time dimension, we encode check-in timestamp into timeslots that represent both season, day of the week and hourslot.

##### *a: SEASON*

Different users prefer visiting different locations in different seasons of the year. Similarly, locations also experience varying check-in distribution in different seasons. For example, outdoor spots like parks will experience higher check-ins during summer and spring compared to winter due to the good weather. Based on the seasonal check-in distribution in different categories shown in figure 2, it is clear that season context influences user check-in preferences. Hence, we incorporate season information in our location recommendation approach.

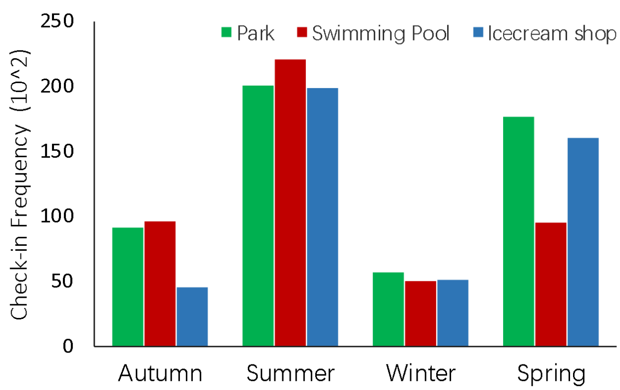



**FIGURE 2. Seasonal distribution of user check-ins in different categories.**

##### *b: DAY OF THE WEEK*

Users in LBSNs have varying check-in behaviors on different days of the week [15], [31]. They check in different location categories during weekdays and weekends. For example, work places have higher check-ins during weekdays while leisure spots have higher check-ins on weekends, as shown in figure 3. Hence, we divide a week into weekday (Monday-Friday) and weekends (Saturday and Sunday).

[2]https://www.worldweatheronline.com/

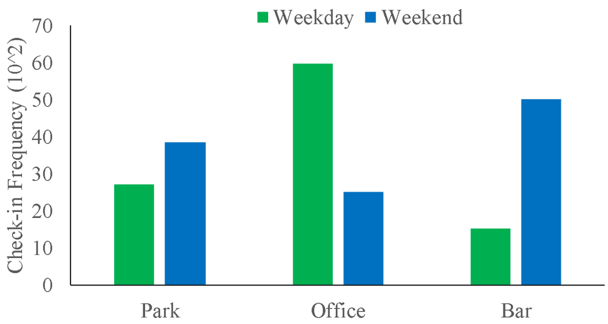



**FIGURE 3.** Daily distribution of user check-ins in different location categories.

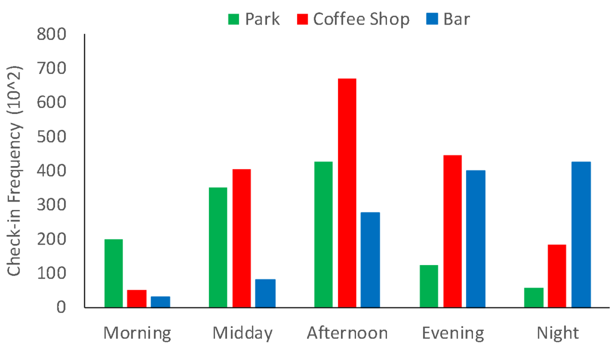



**FIGURE 4.** Hourslot check-in distribution in different categories.

### c: HOURSLOT

Different users prefer visiting different locations during different hours of the day. For example, a user may prefer visiting an animal park in the afternoon and visiting a bar at night. Figure 4 shows the distribution of check-ins of different categories in different hourslots. Based on the figure 4, it is evident that different location categories have different check-in distribution during different hours of the day.

Compared to [13], [34] that divided time into 24 hourslots which may lead to sparsity problem because users have limited number of check-ins within an hour; we divide a day into 5-hour slots as shown in table 2 below.

With longer time frames, more information and more check-in data will be integrated into the tensor and the feature matrices making them more dense, leading to better performance [21].

Because, season, day of the week and hour slot represent time periods of different granularities, we represent them

**TABLE 2.** Hour slot range.

| Houslot | Range |
|---|---|
| Morning | 05:00 am-10:00 am |
| Midday | 10: 00 am-13: 00 pm |
| Afternoon | 13: 00 pm-18:00 pm |
| Evening | 18: 00 pm-23:00 pm |
| Night | 23: 00 pm-05:00 am |

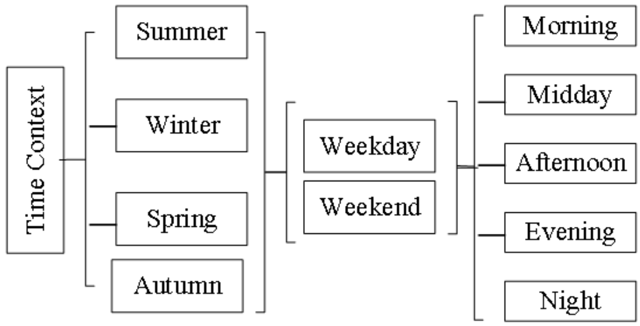



**FIGURE 5.** Time hierarchy representation.

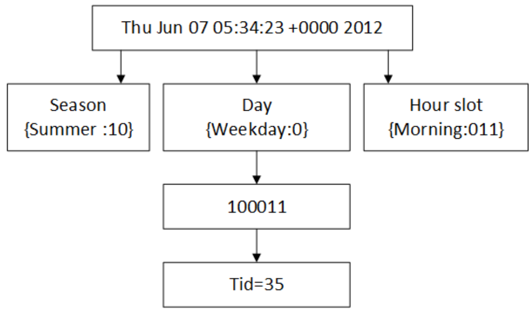



**FIGURE 6.** Time Hashing example.

using a hierarchical model, as shown in figure 5. For example, a check-in that happened on July 2013 at 17 pm will be represented as "*Summer_weekday_Afternoon*". Since there are 4 seasons, 2 types of days in a week and each day has 5-hour slots, we encode them to obtain a total of 40 unique timeslots. Thus, our MCLR-TD approach is able to model user check-in behavior in 40 different hierarchical time periods.

For ease of representation, we encode each timeslot using 2 bits to represent season information, 1 bit to represent the day of the week and 3 bits to represent the hour slot. The binary representation is then converted into a unique decimal digit as the timeslot id. Figure 6 shows an encoding example of sample check-in timestamp "Thu Jun 07 05:34:23 +0000 2012" extracted from the Yelp dataset.

### 2) WEATHER CONTEXT MODELING

Users check-in behavior are always influenced by weather conditions [35]. Users will prefer checking in different location categories in different weather conditions. For example, during favorable weather conditions, many users will prefer visiting outdoor places like parks, when the weather is not favorable they will prefer visiting indoor places like museums or stay at home. Based on figure 7, pressure, precipitation and visibility have a skewed check-in distribution while temperature and moon illumination have a normal distribution.

A location will experience multiple weather conditions simultaneously, for example, in winter it may be sunny but freezing or windy. Thus, we also represent weather context using a hierarchical model. To reduce data sparsity caused by directly using specific temperature values, we split the temperature values into four intervals, namely freezing

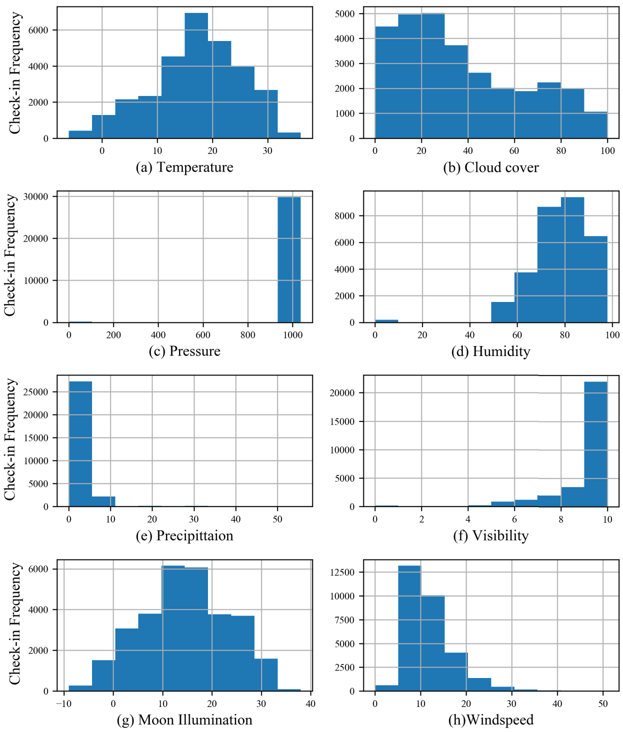



**FIGURE 7.** **Check-in distribution in different weather conditions.**

(below 0 degrees), cold (0-10 degrees), warm (10-25 degrees and hot (above 25 degrees). We combine the seven daily weather summary tags (overcast, sunny, cloudy, rainy, drizzle, fog and mist) with the four temperature slots to obtain a total of 28 weather context dimensions that are used in tensor construction stage. An example of weather context can be represented as $\{overcast\_hot\}$. In order to generate unique ids for each weather context, we use 2 bits to represent the temperature range and 3 bits to represent the daily weather summary tags.

## B. CONSTRUCTION OF TENSOR AND FEATURE MATRICES

### 1) FOUR-MODE TENSOR CONSTRUCTION

From every check-in data quartet (user, location, time, weather), we construct a four mode tensor $A \in R^{(N\times M\times K\times L)}$ to model user check-in preference to given locations under different contextual scenarios. Where N, M, K and L represent the number of users, POIs, timeslots and weather context respectively. The four modes are summarized as follows:

1) Mode-1 (Users): $U = [u_1, u_2, \cdots, u_N]$ denotes N different users
2) Mode-2 (POIs): $L = [l_1, l_2, \cdots, l_M]$ denotes M different POIs where users checked in.
3) Mode-3 (Hashed Timeslots) $T = [t_1, t_2, \cdots, t_K]$ denotes K different hashed timestamps when check-in instance occurred.
4) Mode-4 (Hashed Weather) $W = [w_1, w_2, \cdots, w_L]$ represents L hashed weather conditions under which the check-in instance occurred.

We represent user check-in preferences based on check-in frequency. The higher the check-in frequency, the more a user prefers that location under the given context. Each entry $A(i, j, k, l)$ of the tensor $A$ stores the number of times that user $i$ checked in location $j$ in time slot $k$ and under weather context $w$. Thus, we can access the distribution of user check-ins in different locations under different contexts by accessing the vector $A(i, j, k, l)$. If a user has no check-in record in a given location, then $A(i, j, k, l) = 0$.

Figure 8 below shows the schematic representation of the constructed four mode tensor. The tensor construction process can be interpreted as constructing a three mode tensor with dimensions (user,location,time) for every weather context in our weather dimension.

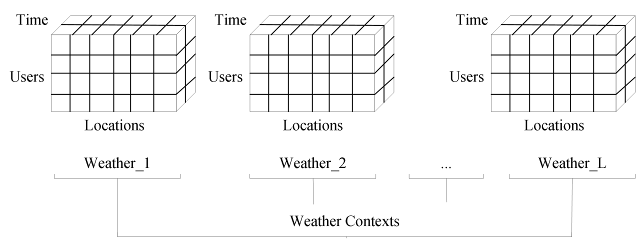



**FIGURE 8.** **Schematic representation of 4-mode tensor.**

### 2) CONSTRUCTION OF FEATURE MATRICES

User check-ins in given contexts are always limited to a small number of locations. The tensor constructed above is very sparse with very few non-zero values. Filling the missing entries by only utilizing the non-zero entries of the sparse tensor is not efficient enough [32].

In order to reduce the sparsity problem, we further construct four feature matrices that are factorized collaboratively with the tensor. The matrices constructed shares atleast one mode in common with the four mode tensor. The coupled factorization of tensor with matrices can best recover the missing entries of the sparse tensor [36]. The feature matrices constructed can also aide in recommendation of locations to new users commonly referred as ‘‘cold start’’ by recommending most popular locations in specified context scenarios. The four matrices are explained below.

#### *a: TIME-CATEGORY MATRIX $M_1$*

Every location belongs to a specific category. Different location categories will experience varying check-in distribution in different time slots, as discussed earlier in section IV-A1. Thus, modeling the distribution of check-ins in different categories in different time slots can help infer which categories are most preferred by the users in a given time slot i.e. category popularity. This is more efficient than modeling the distribution of check-ins in single locations directly with the huge number of locations in the datasets. To generate the time-category matrix, we first cluster check-in data into 144 categories available in our dataset. Check-ins in every category are further clustered into 40 unique timeslots.

Every entry $M_1(i,j)$ of the matrix stores the total counts of check-ins of time slot $t_i$ in category $c_j$. Sample time category matrix is as shown below.

$$\begin{array}{cc} & \begin{array}{ccccc} c_1 & c_2 & c_3 & \cdots & c_M \end{array} \\ \begin{bmatrix} t_1 \\ t_2 \\ t_3 \\ \vdots \\ t_N \end{bmatrix} & \begin{bmatrix} 5 & 0 & 2 & \cdots & 7 \\ 4 & 8 & 11 & \cdots & 6 \\ 3 & 0 & 7 & \vdots & 15 \\ \vdots & \vdots & & \ddots & \vdots \\ 13 & 0 & 5 & \cdots & 8 \end{bmatrix} \end{array}$$

#### b: LOCATION SIMILARITY MATRIX $M_2$

Users tend to prefer visiting locations similar to the ones they visited in the past [37]. Locations with similar check-in distribution in different time periods are considered to be more similar than those with different check-in distribution. For example, a bar is more related to a night club than a coffee shop. We calculate the similarity between two locations based on user check-in distribution in different timeslots using the cosine similarity. The similarity is computed using equation 2.

$$cosSim(l_i, l_j, C) = \frac{\sum_{c_k \in C}(r_{ki} \times r_{kj})}{\sqrt{\sum_{c_k \in C}(r_{ki}^2)} \times \sqrt{\sum_{c_k \in C}(r_{kj}^2)}} \tag{2}$$

where $r_{ki}$ and $r_{kj}$ denotes the check-in frequency of timeslot $t_k$ in location $l_i$ and $l_j$ respectively. $C$ is the context scenario.

Using the location similarity information, we construct a location- location matrix $M_2$, where each row and column of the matrix denotes a location. Each entry $M_2(i,j)$ stores the similarity between location $l_i$ and $l_j$.

$$\begin{array}{cc} & \begin{array}{ccccc} l_1 & l_2 & l_3 & \cdots & l_M \end{array} \\ \begin{bmatrix} l_1 \\ l_2 \\ l_3 \\ \vdots \\ l_N \end{bmatrix} & \begin{bmatrix} 1 & 0.7 & 0.5 & \cdots & 0.3 \\ 0.7 & 1 & 0.3 & \cdots & 0.5 \\ 0.5 & 0.1 & 1 & \vdots & 0.6 \\ \vdots & \vdots & & \ddots & \vdots \\ 0.4 & 0 & 0.5 & \cdots & 1 \end{bmatrix} \end{array}$$

#### c: LOCATION-WEATHER MATRIX $M_3$

As discussed in section IV-A2, locations have varying check-in distribution in different weather contexts. Knowing locations that most users prefer visiting under different weather conditions can help infer the locations a user may be willing to visit in a given weather condition. Matrix $M_3$ measures the popularity of a location in a specific weather context. For every location in our check-in dataset, we cluster the check-ins into 28 clusters based on the weather context. Each element of the matrix stores the total count of check-ins in the given weather context. A sample location weather matrix is shown below.

$$\begin{array}{cc} & \begin{array}{ccccc} w_1 & w_2 & w_3 & \cdots & w_M \end{array} \\ \begin{bmatrix} l_1 \\ l_2 \\ l_3 \\ \vdots \\ l_N \end{bmatrix} & \begin{bmatrix} 11 & 3 & 17 & \cdots & 9 \\ 3 & 8 & 12 & \cdots & 6 \\ 13 & 0 & 7 & \vdots & 15 \\ \vdots & \vdots & & \ddots & \vdots \\ 5 & 0 & 9 & \cdots & 4 \end{bmatrix} \end{array}$$

#### d: USER CATEGORY MATRIX $M_4$

Different users have a varying preference for locations belonging to different categories [38]. For example, some users may prefer checking in shopping malls while others may prefer checking in nightlife spots. Thus, knowing the location category that a given user prefers checking in most can help infer the locations that a given user is most interested in. For every user in our check-in data, we count the check-in frequency in every category and store in matrix $M_4$, where each row denotes a user, while each column denotes a location category. Algorithm 1 summarizes the process of constructing user-category matrix. A sample user-category matrix is as shown below.

$$\begin{array}{cc} & \begin{array}{ccccc} c_1 & c_2 & c_3 & \cdots & c_M \end{array} \\ \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_N \end{bmatrix} & \begin{bmatrix} 3 & 0 & 0 & \cdots & 5 \\ 0 & 6 & 1 & \cdots & 2 \\ 0 & 2 & 0 & \vdots & 1 \\ \vdots & \vdots & & \ddots & \vdots \\ 1 & 0 & \cdots & 4 & 0 \end{bmatrix} \end{array}$$

### C. CONTEXT-AWARE COLLABORATIVE TENSOR DECOMPOSITION

The ultimate goal of tensor decomposition is to supplement missing values of a tensor [39]. The most commonly used decomposition methods are Tucker decomposition (TD) and Canonical Polyadic decomposition (CP) [40]. In this paper, we apply CP decomposition method since it is more computationally flexible to deal with large datasets [39]. The CP decomposition factorizes a tensor into a sum of rank-one

**Algorithm 1** Construction of User-Category Matrix

**Input:** Pre-processed check-in data
**Output:** Matrix $M_4$

data = Load check-in data
Obtain the number of unique users and unique categories
Initialize $M_4$ with zeroes
**for** each user $i$ in data **do**
  **for** each category $j$ in data **do**
    *freq*=checkinCount(i,j)
    $M_4(i,j) = freq$
  **end for**
**end for**
Return $M_4$

tensors. For our case, the 4-mode tensor $A \in R^{(N\times M\times K\times L)}$ is decomposed into four low dimensional factor matrices $U \in R^{(u\times k)}, L \in R^{(l\times k)}, T \in R^{(t\times k)}$ and $W \in R^{(w\times k)}$. The decomposition can be represented as shown in equation 3.

$$A \approx [U, L, T, W] = \sum_{r=1}^{R} \lambda_r u_r \cdot l_r \cdot t_r \cdot w_r \tag{3}$$

where $U, L, T, W$ are termed as the factor matrices which is the combination of the vectors from the rank-one components. $u_r \in R^I, l_r \in R^J, t_r \in R^K$ and $w_r \in R^L$ for $r = [1, 2, \cdots, R]$ represent the rank-one tensors of the four modes users, locations, timeslots and weather respectively. R is a positive integer that denotes the number of components or the rank i.e. the number of rank-one tensors needed to approximate tensor A as their sum. The rows of the four factor matrices correspond to each dimension of the tensor while the columns corresponds to the rank R. Elementwise, equation 3 can be rewritten as shown in equation 4.

$$A_{(rec)} = \sum_{i=1}^{m} (u_{ri} \cdot l_{rj} \cdot t_{rk} \cdot w_{rl}) \tag{4}$$

The four feature matrices $M_1, M_2, M_3, M_4$ are factorized using matrix factorization into their respective factor matrices. For general case, factorization of matrix can be represented using equation 5.

$$M = AB^T M \in R^{n\times m}, \quad A \in R^{n\times r},\ B^T \in R^{r\times m} \tag{5}$$

where r denotes the rank of the factorization. The factorization explains matrix M through r different latent factors, which are encoded in the matrices A and $B^T$.

In order to perform multiplications of a tensor and matrices, a tensor must first be matricized into one of its dimensions as a matrix. Matricization refers to reordering the elements of N-way array into a matrix. Mode-k unfolding of the tensor is obtained by assembling all the mode-k fibers into a matrix. The product is calculated by multiplying each mode-n fiber by the matrix. For example, with our four mode tensor, the product of tensor $X \in R^{(I\times J\times K\times L)}$ with a matrix $B \in R^{(M\times J)}$ denoted as $Y = X\times_n B \in R^{(I\times M\times K\times L)}$ can be represented as shown in equation 6; where the $X_n$ denotes the mode-n product,for our case $n = 2$.

$$Y(i, m, k, l) = \sum_{j} x_{(i,j,k,l)} b_{(m,j)} \tag{6}$$

The missing values of the tensor are imputed by extracting common factor matrices with respect to similar modes of the tensor and the matrices. This is achieved by taking first order partial derivatives during optimization. With the collaborative factorization of tensor and matrix, the factor matrices share parameters when an entity participates in multiple relations. For example, matrix $M_1$ shares time dimension with tensor A, Matrix $M_2$ and $M_3$ shares location dimension with the tensor and matrix $M_4$ shares user dimension with the tensor. Thus, we can propagate knowledge from the four feature matrices into our tensor by collaborative tensor decomposition.

### D. FORMULATION OF THE OBJECTIVE FUNCTION

Given tensor $A \in R^{(I\times J\times K\times L)}$ and the four feature matrices $M_1, M_2, M_3, M_4$, our objective is to find the components $U, L, T, W, C$ that minimizes the equation below:

$$\begin{aligned} F(&U, L, T, W, C) \\ &= \frac{1}{2}||Z \times (A - U\circ L\circ T\circ W)||_F^2 \\ &\quad + \frac{\lambda_1}{2}||M_1 - TC^T||_F^2 + \frac{\lambda_2}{2}||M_2 - LL^T||_F^2 \\ &\quad + \frac{\lambda_3}{2}||M_3 - LW^T||_F^2 + \frac{\lambda_4}{2}||M_4 - UC^T||_F^2 \\ &\quad + \frac{\lambda_5}{2}(||U||^2 + ||L||^2 + ||T||^2 + ||W||^2) \end{aligned} \tag{7}$$

where $\frac{1}{2}||Z \times (A - U\circ L\circ T\circ W)||^2$ is the least squared error loss function for the decomposition of the 4-mode tensor into four factor matrices $U, L, T, W$. $Z$ is the non-negative weighting tensor that stores 1 for known entries and 0 for unknown entries of tensor. $A$. $||.||^2$ denotes the Frobenius norm calculated as follow:.

$$||A||_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} ||a_{i,j}||^2} \tag{8}$$

$M_1 - TC^T$ is the least square error for the factorization of matrix $M_1$ into factor matrices T and C. $M_2 - LL^T$ is the least square error for the factorization of matrix $M_2$ into factor matrices $L \in R^{(l\times k)}$ and $L^T \in R^{(k\times l)}$. $M_3 - LW^T$ is the least square error for the factorization of matrix $M_3$ into factor matrices $L \in R^{(l\times k)}$ and $W \in R^{(w\times k)}$. $M_4 - UC^T$ denotes the least square error for the factorization of matrix $M_4$ into factor matrices $U \in R^{(u\times k)}$ and $C \in R^{(c\times k)}$. $\frac{\lambda_5}{2}(||U||^2+||L||^2+||T||^2+||W||^2)$ is the regularization term for avoiding overfitting. $\lambda_1$, $\lambda_2$, $\lambda_3$, $\lambda_4$ and $\lambda_5$ are the model parameters that controls the weights of different parts of the objective function during the decomposition process. Due to the drawbacks of using Alternating Least Squares (ALS) and Gradient Descent (GD) [26], we apply Stochastic Gradient Descent (SGD) [41] to solve the optimization problem by taking the derivatives of the objective function with respect to each of the four factors user, location, time and weather.

The optimization algorithm is summarized in Algorithm 2. It takes the four mode tensor A and the four feature matrices $M_1, M_2, M_3, M_4$ as the input. We then initialize the four factor matrices with small random values and run SGD based on non- zero elements to find the minimum values. We then take the partial derivatives where $Z^i$ and $A^i$ denotes the mode-i tensor unfolding of the tensors Z and A respectively. Notation * denotes the Khatri-Rao product. The final outputs are the four factor matrices which are used in reconstructing a dense tensor.

**Algorithm 2** Collaborative Tensor Factorization

**Input:** Tensor A, matrices $M_1, M_2, M_3, M_4$;
**Output:** Factor matrices U, L, T, W

**for** $n = 1 \in size(A)$ **do**
  Initialize $U, L, T, W$ with small random values
**end for**
**while** not converged **do**
  Compute step length $\alpha_i$
  Compute the gradients as follows:
  $\nabla_U F = (Z^1 - A^1)(L * T * W) + \lambda_5 U + \lambda_4(M_4 - UC^T)$
  $\nabla_L F = (Z^2 - A^2)(U * T * W) + \lambda_2(M_2 - LL^T) + \lambda_3(M_3 - LW^T) + \lambda_5 L$
  $\nabla_T F = (Z^3 - A^3)(U * T * W) + \lambda_2(M_1 - TC^T) + \lambda_5 T$
  $\nabla_W F = (Z^4 - A^4)(U * L * T) + \lambda_3(M_3 - LW^T) + \lambda_3(M_3 - LW^T) + \lambda_5 L$
  Update values based on new gradients
  $U_{i+1} = U_i - \alpha_i \nabla_U F_i$
  $L_{i+1} = L_i - \alpha_i \nabla_L F_i$
  $T_{i+1} = T_i - \alpha_i \nabla_T F_i$
  $W_{i+1} = W_i - \alpha_i \nabla_W F_i$
  Compute $F_{i+1}$
**end while**
Return $U, L, T, W$

### E. GENERATION OF TOP-K LOCATIONS

After the decomposition process, we can recover missing values of our sparse tensor A by taking the outer product of the four factor matrices of our output using equation 9.

$$A_{rec} = U \cdot L \cdot T \cdot W \tag{9}$$

The reconstructed tensor measures the associations among the users, locations, timeslots and weather. Each element of $A_{rec}$ represent a relation $u, l, t, w, s$, where s is the predicted score of user u visiting location l under context $(t, w)$. The process of generating top-k locations is summarized in Algorithm 3.

**Algorithm 3** Top-K Locations Generation Process

**Input:**Factor Matrices, userID, timeslotID, weatherID, K
**Output:**Ranked list of K locations

Reconstruct tensor $A$ using equation 9
locationList=[]
**for** $useru \in U$ **do**
  Select all location based on userID and context ID
  locationList.append($A_{rec}[uid, :, tid, wid]$)
**end for**
Rank locations based on the predicted visiting score.
$top\_k$ =select($ranked\_list$,K)
Return $top\_k$

We access the list of candidate locations to recommend to user $u$ in a given context $(t, w)$ by selecting $N$ corresponding locations with the highest predicted weights from the reconstructed tensor. Locations are ranked based on the values of the predicted scores and the Top-K locations are recommended to the user.

## V. EXPERIMENTS AND EVALUATION

In this section, we conduct extensive experiments to demonstrate the effectiveness of MCLR-TD approach on location recommendation.

### A. EXPERIMENT SETUP

Statistics of the two check-in datasets used in this paper is summarized in table 1. Datasets from two different LBSNs collected from different cities New York and Tokyo are used so as to prove the resiliency of our model on variant datasets. We only selected check-ins that belong to five primary categories namely: food, big outdoors, nightlife, entertainment and shopping. This is aimed at getting rid of residential areas like homes. Furthermore, we only retained users who have check-ins in at least 10 distinct POIs and locations with at least 5 check-in records.

We implemented our approach in MATLAB using MATLAB Tensor Toolbox.[3] For each user in our dataset, we first rank their check-ins based on check-in timestamp and select 30% of their most recent check-ins as the testing data; the remaining records are used for training and validation. The input files are text files with one line for each non zero value in the tensor and matrix. Each line consists of integers that give the indexes of the non-zero value in the tensor, followed by a real number that represents the check-in frequency. For example, for the tensor construction, the text file consists of five values $\{uid, Pid, tid, wid, freq\}$ in each line, where $freq$ represents the check-in frequency.

### B. BASELINE METHODS

In order to measure the effectiveness of our approach, we compare it with the following approaches:

1) Standard CP: Takes only the four dimensional tensor as the input without considering the four feature matrices. Its objective function is obtained by setting $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ of equation 7 to zeros as shown in equation 10.

$$F(U, L, T, W) = \frac{1}{2}||Z \times (A - U \circ L \circ T \circ W)||_F^2 + \frac{\lambda_5}{2}(||U||^2 + ||L||^2 + ||T||^2 + ||W|^2) \tag{10}$$

2) GeoMF++ [24]. A joint geographical and Matrix factorization model that exploits user mobility data and location's spatial information in making location recommendation.
3) ATTF [31]: A three dimensional tensor based model that models temporal context in different time scales; hour, day,week and month but does not incorporate weather context data.
4) UZT [30]: Content based model that applied tensor factorization in modeling user check-in preferences using topic probability distribution learned from user comments data.

[3]https://www.sandia.gov/ tgkolda/TensorToolbox/index-2.6.html

5) ST-DME [34]: Utilized the spatial temporal contexts in making location recommendation. It incorporates day of the week and the 24 hours of a day as the time context.
6) PCTF [22]: A three mode tensor based method that considers 24 hour timeslots and also incorporates feature matrices when modeling user check-in preferences.

### C. EVALUATION METRICS

We evaluate the performance of MCLR-TD model using three standard performance evaluation metrics. Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) measure the deviation between the true and the predicted values and are calculated using equation 11 and 12 respectively.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(A_{(ijkl)} - A'_{(ijkl)})^2} \tag{11}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|A_{(ijkl)} - A'_{(ijkl)}| \tag{12}$$

where $A_{(ijkl)}$ and $A'_{(ijkl)}$ denotes the real value and the predicted value respectively. N is the number of POIs in the recommendation list.

Because both RMSE and MAE may not be accurate enough when evaluating the ranked list of locations recommended to the user, we used Mean Reciprical Rank (MRR) to evaluate the quality of the final ranked list of POIs recommended to the user. MRR is calculated using equation 13.

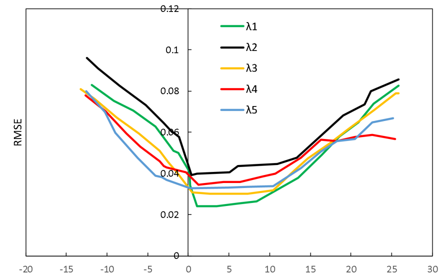
$$MRR = \frac{1}{|N|}\sum_{i=1}^{N}(\frac{1}{rank_i}) \tag{13}$$

### D. EXPERIMENTAL RESULTS

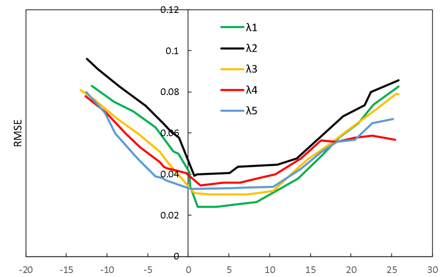
In this subsection we discuss the results of our experiments in five parts as follows. (1) The first part compares the performance of our model on varying parameter values and selects the optimal values for the parameters. (2) The second part discusses the effect of multiple contexts on recommendation performance. (3) The fourth part investigates the impact of feature matrices on the performance of MCLR-TD model. (4) Measures the runtime for Standard-CP and MCLR-TD methods. (5) The last part compares the performance of MCLR-TD model with baseline approaches.

#### 1) MODEL PARAMETER TUNING

In order to find optimal values for our model parameters, we used the control variable method in learning the parameters. 70% of the training dataset was used for learning the parameters, while the remaining 30% was used for validation. The five parameters $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5$ are all set to 0.0001 as the initial value and then we ran parameter sweeping experiments by varying the values of one parameter at a time. We computed RMSE on the validation dataset and picked the optimal parameters with the least RMSE.




**(a)** Yelp Dataset




**(b)** Foursquare Dataset

**FIGURE 9.** Variation of RMSE for different values of model parameters on validation dataset.

Figure 9 above shows the results obtained for different values of the parameters on the two validation datasets. We observed that RMSE values varies with different values of all the five parameters. The RMSE increases when the value of the parameters is too small. This is because when the parameter value is too small, the MCLR-TD model cannot fully utilize the information from the corresponding feature matrices. Similarly, RMSE increases when the value of the parameters is too large, this is because the model over utilized the information from the feature matrices and less information from the four mode tensor leading to poor performance. Hence, selecting optimal parameters is very critical for the performance of MCLR-TD model.

For the Yelp dataset, the optimal parameter combination for the five parameters $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5$ for our model was set to [1.25, 0.473, 0.883, 1.153, 0.438] respectively. For the Foursquare dataset, the best parameters combination was [1.351, 0.689, 0.994, 1.607, 0.0.479]. The optimal parameters for Foursquare dataset are slightly higher than those for Yelp dataset. This because foursquare dataset is less dense, hence, learns more information from the feature matrices. Based on the results, it is evident that all the parameters affected the performance of the MCLR-TD model. However, $\lambda_1, \lambda_3, \lambda_4$ were more influential than $\lambda_2, \lambda_5$. This is because $\lambda_1, \lambda_3, \lambda_4$ controls the weights of time, weather and category contextual factors of our model.

#### 2) IMPACT OF MULTIPLE CONTEXTS

In this subsection, we carry out experiments to compare the performance of our model when it incorporates one context factor at a time and when both contexts are modeled

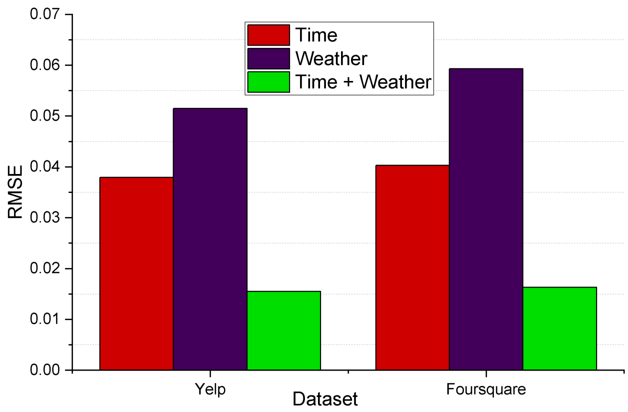



**FIGURE 10. Impact of contexts on performance.**

simultaneously. The first model considers only time context information (user, location, time); the second model considers only weather context information (user, location, weather) and the third model incorporates both time and weather contexts (user, location, time, weather). We carried out separate experiments for the three models on both datasets and computed the RMSE. Figure 10 show the results for the three models.

Based on the results, we observed that the model that considers both temporal context (time) and weather context simultaneously outperforms models that consider only one context factor. This is confirms that users mobility pattern is influenced by multiple contexts, a model that considers multiple contexts can best capture users' contextual preferences compoared to those that consider only one context factor. Hence, incoprorating multiple contexts in a recommendation system help improve its performance. We also observed that the model that only consider time context performed better than the model that consider weather context. This proves that temporal context is a key factor in location recommendation systems, as also proved in previous works [13], [42]. Thus, modeling the two contexts time and weather using a four mode tensor boosts the performance of our model.

### 3) IMPACT OF FEATURE MATRICES

This experiment was aimed at measuring the effect of each feature matrix incorporated in our model. We carry out five separate experiments on collaborative tensor-matrix decomposition by incorporating only one feature matrix at a time on both datasets. These models are compared with the Standard CP (SCP) model which does not include any feature matrix and the MCLR-TD model that incorporate all the feature matrices. The results obtained are summarized in Table 3.

Based on the results, we observed that models that incorporate feature matrices $M_1$ and $M_4$ has better performance than model that incorporate matrices $M_2$ and $M_3$. This is because matrices $M_1$ and $M_4$ stores information about time and category contexts that have higher influence on user check-in behavior. We also observed that the results obtained by the models that consider only one feature matrix does not vary very much, this is because feature matrices only share one or two modes with the four mode tensor, hence not effective enough when recovering the missing values of the original tensor during decomposition process because the model can only learn more information from the shared dimension. For example, matrix $M_1$ only shares the time dimension with the tensor. When all the 4 feature matrices are combined together using optimal parameter values, our model achieves much better performance compared to Standard-CP and the other models that only incorporate only one feature matrix. Standard CP method has worst performance because it does not utilize any external information learned from the feature matrices. Hence, incorporating all the four feature matrices in our recommendation model with optimal parameter values improves its performance by helping reduce the data sparsity problem.

**TABLE 3. Impact of feature matrices on performance.**

| Approach | Yelp | | | Foursquare | | |
|---|---|---|---|---|---|---|
| | RMSE | MAE | MRR | RMSE | MAE | MRR |
| SCP | 0.0496 | 0.0473 | 0.109 | 0.0513 | 0.0481 | 0.103 |
| M1 | 0.0386 | 0.0371 | 0.157 | 0.0398 | 0.0399 | 0.149 |
| M2 | 0.0403 | 0.0385 | 0.135 | 0.0453 | 0.0387 | 0.128 |
| M3 | 0.0399 | 0.0391 | 0.149 | 0.0411 | 0.0402 | 0.137 |
| M4 | 0.0375 | 0.0362 | 0.168 | 0.0397 | 0.0365 | 0.148 |
| MCLR-TD | **0.0194** | **0.0165** | **0.437** | **0.0215** | **0.0207** | **0.409** |

### 4) RUNTIME COMPARISON

We carry out experiments to measure the time taken to converge by the Standard-CP method and the MCLR-TD method on both datasets. The results obtained are summarized in table 4. Because the objective function is minimized based on the known entries of the tensor, we observed that Standard-CP method took longer time to converge compared to MCLR-TD method. MCLR-TD can obtain missing values from the feature matrices, hence took less time to converge. Both methods took longer time to converge on Foursquare dataset because it's more sparse compared to Yelp dataset.

**TABLE 4. Runtime comparison.**

| Approach | Time (in seconds) | |
|---|---|---|
| | Yelp | Foursquare |
| Standard-CP | 12500 | 14080 |
| MCLR-TD | 5800 | 6000 |

### 5) COMPARISON WITH BASELINE METHODS

Tables 5 show the results achieved by our approach and the baseline methods on the two test datasets for the three evaluation metrics: RMSE, MAE and MRR.

Based on the results, we observed that MCLR-TD outperforms the five baseline approaches on both datasets. This implies that the incorporation of multiple context factors

**TABLE 5. Performance comparison with baseline methods.**

| Approach | Yelp | | | Foursquare | | |
|---|---|---|---|---|---|---|
| | RMSE | MAE | MRR | RMSE | MAE | MRR |
| MCLR-TD | **0.0198** | **0.0157** | **0.5193** | **0.0215** | **0.0187** | **0.5016** |
| PCTF | 0.0375 | 0.0351 | 0.3173 | 0.0394 | 0.0383 | 0.3025 |
| ATTF | 0.0308 | 0.0297 | 0.3694 | 0.0341 | 0.0325 | 0.3319 |
| UZT | 0.0437 | 0.0409 | 0.3015 | 0.0503 | 0.0496 | 0.2793 |
| ST-DME | 0.0498 | 0.0435 | 0.2738 | 0.0518 | 0.0468 | 0.2095 |
| GeoMF++ | 0.0533 | 0.0508 | 0.1863 | 0.0587 | 0.0549 | 0.1582 |

modeled at different scales is effective for POI recommendation. Apart from incorporating time and weather contexts, MCLR-TD approach also incorporates the influence of location category information by incorporating user category transition matrix. MCLR-TD is able to deal with the data sparsity problem by incorporating four additional feature matrices that are collaboratively decomposed together with the tensor, hence more accurate in predicting the missing values.

ATTF model performs better than PCTF model because it incorporates temporal context in three different dimensions while PCTF only consider 24 hours of the day. UZT and ATTF performs poorer compared to MCLR-TD because they only utilize the sparse tensor as the input. UZT model also only considers hour of the day time granularity, hence the model is not able to fully capture the temporal influence on user check-in behaviors. GeoMF++ approach has the worst performance because it only considers the spatial influence when modeling user check-in behavior, hence it is unable to model the temporal changes in user check-in preference. Furthermore, PCTF, UZT, ATTF and ST-DME approaches only model time context but ignore the influence of other contextual factors like weather and category information leading to poor performance. We also observed that all methods have better performance on the Yelp dataset compared with Foursquare dataset. This is because the Yelp dataset has a higher check-in density and more POI compared to Foursquare dataset; hence low sparsity.

In summary, the results obtained shows the effectiveness of utilizing multiple contextual factors in a recommendation system using tensor based approach. The integration of the tensor and the feature matrices improves the performance of our model by making it more robust in dealing with sparse datasets.

## VI. CONCLUSION AND FUTURE WORK

In this paper, we proposed MCLR-TD approach that utilizes multiple contextual information of time and weather in making location recommendation using collaborative tensor-matrix decomposition. We first carried out an extensive data analysis on the two datasets to determine how human check-in behaviors are affected by contextual factors. Based on the intuitions obtained from data analysis, we constructed a four mode tensor to model the relationship among users, locations, time and weather. Our approach further incorporates four feature matrices constructed using data from different dimensions. The four mode tensor and the four feature matrices decomposed together hence reducing the data sparsity problem. The evaluation results on two real world check-in datasets showed that MCLR-TD performs better than the baseline methods due to it's ability to model multiple contexts and deal with data sparsity problem by incorporating information from different dimensions. In future, we intend to work on scaling to large and more datasets collected from different LBSNs. We will also work on how to incorporate other contexts factors such as trip purpose and mode of transport.

## REFERENCES

[1] F. Yu and S. Jiang, "Mining location influence for location promotion in location-based social networks," *IEEE Access*, vol. 6, pp. 73444–73456, 2018.
[2] B. Chen, S. Yu, J. Tang, M. He, and Y. Zeng, "Using function approximation for personalized point-of-interest recommendation," *Expert Syst. Appl.*, vol. 79, pp. 225–235, Aug. 2017.
[3] S. Wang, M. Gong, C. Qin, and J. Yang, "A multi-objective framework for location recommendation based on user preference," in *Proc. 13th Int. Conf. Comput. Intell. Secur. (CIS)*, Dec. 2017, pp. 39–43.
[4] X. Jiao, Y. Xiao, W. Zheng, H. Wang, and Y. Jin, "R2SIGTP: A novel real-time recommendation system with integration of geography and temporal preference for next Point-of-Interest," in *Proc. World Wide Web Conf. (WWW)*, May 2019, pp. 3560–3563.
[5] R. Gao, J. Li, X. Li, C. Song, J. Chang, D. Liu, and C. Wang, "STSCR: Exploring spatial-temporal sequential influence and social information for location recommendation," *Neurocomputing*, vol. 319, pp. 118–133, Nov. 2018.
[6] A. S. Dokuz and M. Celik, "Discovering socially important locations of social media users," *Expert Syst. Appl.*, vol. 86, pp. 113–124, Nov. 2017.
[7] R. Yue-Qiang, W. Ze, S. Xiao-Na, and S. Shi-Min, "A multi-element hybrid location recommendation algorithm for location based social networks," *IEEE Access*, vol. 7, pp. 100416–100427, 2019.
[8] Z. Pan, L. Cui, X. Wu, Z. Zhang, X. Li, and G. Chen, "Deep potential Geo-social relationship mining for Point-of-Interest recommendation," *IEEE Access*, vol. 7, pp. 99496–99507, 2019.
[9] R. Logesh and V. Subramaniyaswamy, "A reliable point of interest recommendation based on trust relevancy between users," *Wireless Pers. Commun.*, vol. 97, no. 2, pp. 2751–2780, Nov. 2017.
[10] W. Cai, Y. Wang, R. Lv, and Q. Jin, "An efficient location recommendation scheme based on clustering and data fusion," *Comput. Electr. Eng.*, vol. 77, pp. 289–299, Jul. 2019.
[11] W. Liu, J. Wang, A. K. Sangaiah, and J. Yin, "Dynamic metric embedding model for point-of-interest prediction," *Future Gener. Comput. Syst.*, vol. 83, pp. 183–192, Jun. 2018.
[12] C. Cheng, H. Yang, I. King, and M. R. Lyu, "Fused matrix factorization with geographical and social influence in location-based social networks," in *Proc. 26th AAAI Conf. Artif. Intell. (AAAI)*, 2012, pp. 17–23.
[13] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann, "Time-aware point-of-interest recommendation," in *Proc. 36th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2013, pp. 363–372.
[14] H. Gao, J. Tang, X. Hu, and H. Liu, "Exploring temporal effects for location recommendation on location-based social networks," in *Proc. 7th ACM Conf. Recommender Syst. RecSys*, 2013, pp. 93–100.
[15] S. Zhao, T. Zhao, H. Yang, M. R. Lyu, and I. King, "Stellar: Spatial-temporal latent ranking for successive point-of-interest recommendation," in *Proc. 30th AAAI Conf. Artif. Intell.*, Feb. 2016, pp. 315–321.
[16] H. Bagci and P. Karagoz, "Context-aware location recommendation by using a random walk-based approach," *Knowl. Inf. Syst.*, vol. 47, no. 2, pp. 241–260, May 2016.
[17] G. Adomavicius, B. Mobasher, F. Ricci, and A. Tuzhilin, "Context-aware recommender systems," *AI Mag.*, vol. 32, no. 3, pp. 67–80, Jan. 2011.

[18] S. Setiowati, T. B. Adji, and I. Ardiyanto, "Context-based awareness in location recommendation system to enhance recommendation quality: A review," in *Proc. Int. Conf. Inf. Commun. Technol. (ICOIACT)*, Mar. 2018, pp. 90–95.
[19] J. Bao, Y. Zheng, D. Wilkie, and M. Mokbel, "Recommendations in location-based social networks: A survey," *GeoInformatica*, vol. 19, no. 3, pp. 525–565, Jul. 2015.
[20] M. Aliannejadi and F. Crestani, "Personalized context-aware point of interest recommendation," *ACM Trans. Inf. Syst.*, vol. 36, no. 4, pp. 1–28, Oct. 2018.
[21] L. Yao, Q. Z. Sheng, X. Wang, W. E. Zhang, and Y. Qin, "Collaborative location recommendation by integrating multi-dimensional contextual information," *ACM Trans. Internet Technol.*, vol. 18, no. 3, pp. 1–24, Feb. 2018.
[22] W. Luan, G. Liu, C. Jiang, and L. Qi, "Partition-based collaborative tensor factorization for POI recommendation," *IEEE/CAA J. Automatica Sinica*, vol. 4, no. 3, pp. 437–446, Jul. 2017.
[23] P. Kefalas and Y. Manolopoulos, "A time-aware spatio-textual recommender system," *Expert Syst. Appl.*, vol. 78, pp. 396–406, Jul. 2017.
[24] D. Lian, K. Zheng, Y. Ge, L. Cao, E. Chen, and X. Xie, "GeoMF++: Scalable location recommendation via joint geographical modeling and matrix factorization," *ACM Trans. Inf. Syst.*, vol. 36, no. 3, pp. 1–29, Mar. 2018.
[25] H. Gao, J. Tang, X. Hu, and H. Liu, "Content-aware point of interest recommendation on location-based social networks," in *Proc. 29th AAAI Conf. Artif. Intell.*, Feb. 2015, pp. 1721–1727.
[26] F. Yang, F. Shang, Y. Huang, J. Cheng, J. Li, Y. Zhao, and R. Zhao, "LFTF: A framework for efficient tensor analytics at scale," *Proc. VLDB Endowment*, vol. 10, no. 7, pp. 745–756, Mar. 2017.
[27] B. Zou, C. Li, L. Tan, and H. Chen, "GPUTENSOR: Efficient tensor factorization for context-aware recommendations," *Inf. Sci.*, vol. 299, pp. 159–177, Apr. 2015.
[28] X. Zhu and R. Hao, "Context-aware location recommendations with tensor factorization," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Jul. 2016, pp. 1–6.
[29] S. Maroulis, I. Boutsis, and V. Kalogeraki, "Context-aware point of interest recommendation using tensor factorization," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2016, pp. 963–968.
[30] G. Liao, S. Jiang, Z. Zhou, C. Wan, and X. Liu, "POI recommendation of location-based social networks using tensor factorization," in *Proc. 19th IEEE Int. Conf. Mobile Data Manage. (MDM)*, Jun. 2018, pp. 116–124.
[31] S. Zhao, I. King, and M. R. Lyu, "Aggregated temporal tensor factorization model for Point-of-Interest recommendation," *Neural Process. Lett.*, vol. 47, no. 3, pp. 975–992, Jun. 2018.
[32] W. Luan, G. Liu, and C. Jiang, "Collaborative tensor factorization and its application in POI recommendation," in *Proc. IEEE 13th Int. Conf. Netw., Sens., Control (ICNSC)*, Apr. 2016, pp. 1–6.
[33] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu, "Modeling user activity preference by leveraging user spatial temporal characteristics in LBSNs," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 45, no. 1, pp. 129–142, Jan. 2015.
[34] R. Ding, Z. Chen, and X. Li, "Spatial-temporal distance metric embedding for time-specific POI recommendation," *IEEE Access*, vol. 6, pp. 67035–67045, 2018.
[35] C. Trattner, A. Oberegger, L. Marinho, and D. Parra, "Investigating the utility of the weather context for point of interest recommendations," *Inf. Technol. Tourism*, vol. 19, nos. 1–4, pp. 117–150, Jun. 2018.
[36] P. Bhargava, T. Phan, J. Zhou, and J. Lee, "Who, what, when, and where: Multi-dimensional collaborative recommendations using tensor factorization on sparse user-generated data," in *Proc. 24th Int. Conf. World Wide Web (WWW)*, 2015, pp. 130–140.
[37] J. Zeng, Y. Li, F. Li, J. Wen, and S. Hirokawa, "A Point-of-Interest recommendation method using location similarity," in *Proc. 6th IIAI Int. Congr. Adv. Appl. Informat. (IIAI-AAI)*, Jul. 2017, pp. 436–440.
[38] J. He, X. Li, and L. Liao, "Next point-of-interest recommendation via a category-aware listwise Bayesian personalized ranking," *J. Comput. Sci.*, vol. 28, pp. 206–216, Sep. 2018.
[39] Q. Song, H. Ge, J. Caverlee, and X. Hu, "Tensor completion algorithms in big data analytics," *ACM Trans. Knowl. Discovery Data (TKDD)*, vol. 13, no. 1, pp. 1–48, Jan. 2019.
[40] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, Aug. 2009.
[41] S. Ruder, "An overview of gradient descent optimization algorithms," *CoRR*, vol. abs/1609.04747, pp. 1–14, Aug. 2016. [Online]. Available: http://arxiv.org/abs/1609.04747
[42] W. X. Zhao, N. Zhou, A. Sun, J.-R. Wen, J. Han, and E. Y. Chang, "A time-aware trajectory embedding model for next-location recommendation," *Knowl. Inf. Syst.*, vol. 56, no. 3, pp. 559–579, Sep. 2018.

**JING LU** received the Ph.D. degree from the University of Stuttgart, Germany, in 2009. She is currently a Teaching Fellow with the University of Shanghai for Science and Technology. Her current research interests include location recommendation, entity resolution, and truth discovery.

**MARTIN A. INDECHE** received the degree from the University of Shanghai for Science and Technology, Shanghai, China, in 2017, where he is currently pursuing the M.S. degree in computer science and technology with the Department of Optical-Electrical and Computer Engineering. His current research interests include location recommendation and data integration.