Mobile Query Recommendation via Tensor Function Learning

Zhou Zhao[1], Ruihua Song[3], Xing Xie[3], Xiaofei He[2] and Yueting Zhuang[1]
[1]College of Computer Science, Zhejiang University
[2]State Key Lab of CAD&CG, Zhejiang University
[3]Microsoft Research, Beijing, China
{zhaozhou,yzhuang}@zju.edu.cn, xiaofeihe@gmail.com
{rsong,xingx}@microsoft.com

Abstract

With the prevalence of mobile search nowadays, the benefits of mobile query recommendation are well recognized, which provide formulated queries sticking to users' search intent. In this paper, we introduce the problem of query recommendation on mobile devices and model the user-location-query relations with a tensor representation. Unlike previous studies based on tensor decomposition, we study this problem via tensor function learning. That is, we learn the tensor function from the side information of users, locations and queries, and then predict users' search intent. We develop an efficient alternating direction method of multipliers (ADMM) scheme to solve the introduced problem. We empirically evaluate our approach based on the mobile query dataset from Bing search engine in the city of Beijing, China, and show that our method can outperform several state-of-the-art approaches.

1 Introduction

Search marketing is witnessing a dramatic change in the recent years where mobile search has a tremendous growth in the market share. As the sales of smartphones and tablets continue to rise, there's no end in sight to the popularity of mobile search. The benefits of query recommendation in mobile environment are well recognized, which provide formulated queries sticking to users' search intent.

Query recommendation is considered an important component in enhancing keyword-based queries in search engines. The existing approaches [Guo *et al.*, 2010; Feild and Allan, 2013; Li *et al.*, 2008; Anagnostopoulos *et al.*, 2010] mainly focus on desktop query recommendation, which aims to provide alternative queries of the users' issued queries or assist users in refining their queries in desktop search. However, they may not be suitable for mobile query recommendation because of two reasons. First, typing or data entry on mobile devices is more difficult than on desktop computers. It is reported in [Fu *et al.*, 2009] that even for skilled users, they can only reach 21 words per minute on mobile devices; while on desktop computers, the rate is at least 60 words for common users.Second, many mobile devices come with position functions such as Geographical Positioning System (GPS) and sensors. Through these devices, more and more data are being accumulated in the form of current user's location and location annotations. For example, the location-based entity system, *yelp* [1] provides the location annotations such as the reviews for restaurants, shops and etc. Thus, we can identify users' current activities and further improve the performance of mobile query recommendation.

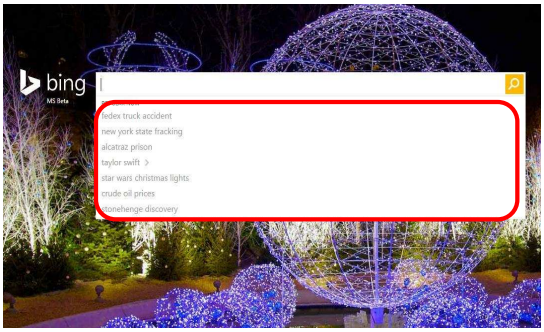


Figure 1: Query Recommendation on Bing Search

In this paper, we introduce a new problem of mobile query recommendation, which is not limited to query refinement, but to find what users need. For example, Figure 1 shows the automatic query recommendation system in Bing search engine. We model the user-location-query relations with a tensor representation. Unlike previous studies on tensor decomposition and completion, we study this problem from the viewpoint of missing value estimation via tensor function learning. Given a tensor indicating the existing user-location-query relations, we want to recommend the right query for users based on their current location. Since the relations between users and queries at some locations are unknown (missing values in the mobile query tensor data), we want to predict the missing values in the tensor first, then recommends users with the queries with high predicted values. We then learn a tensor function based on the mobile query data, location annotations and query information. We obtain the annotations for the location in Beijing, China from a *yelp*-like

[1]http://www.yelp.com/

location-based entity reviewing system, *dianping* [2]. We extract the users' clicked webpages to enrich the information of their issued queries. The main contributions of this paper are as follows:

1. We introduce a new problem of query recommendation on mobile devices, and formulate a tensor function learning model based on the side information of user, location and query.
2. We introduce a tractable relaxation of the tensor function learning, and then obtain a convex problem of functional coefficient nuclear norm minimization. We present an efficient alternating direction method of multipliers (ADMM) scheme to solve the introduced problem.
3. We evaluate the performance of our method on Bing search engine on mobile devices for three months in the city of Beijing, China, and show that our method can outperform several state-of-the-art solutions to the problem.

The rest of this paper is organized as follows. In Section 2, we introduce the notions of the problem and provide a brief review of the related work about tensor completion and query recommendation. We introduce the problem of mobile query recommendation and provide the optimization algorithm in Section 3. A variety of experimental results are presented in Section 4. Finally we provide some concluding remarks in Section 5.

2 Notions and Related Work

Before reviewing previous work, we first introduce basic tensor notions and terminologies. An n-mode tensor is defined as $\mathbf{Y} \in R^{I_1 \times I_2 \times \ldots \times I_n}$, and its elements are denoted as $y_{i_1,\ldots,i_n}$ where $1 \leq i_k \leq I_k$ and $1 \leq k \leq n$. For example, a matrix is a 2-mode tensor. The k-th mode unfolding, also known as matricization, of an n-mode tensor $\mathbf{Y}$ is denoted as $unfold_k(\mathbf{Y}) = \mathbf{Y}_{(k)} \in R^{I_k \times \prod_{j \neq k} I_j}$ where the k-th mode becomes the row index and all other $(n-1)$ modes become the column indices. The tensor element $(i_1, i_2, \ldots, i_N)$ is mapped to the matrix element (i_n, j), where

$$j = 1 + \sum_{k=1, k \neq n}^{N} (i_k - 1) J_k \text{ with } J_k = \prod_{m=1, m \neq n}^{k-1} I_m.$$

The opposite operation is defined as $fold_k(\mathbf{Y}_{(k)}) = \mathbf{Y}$. The Frobenius norm of the tensor $\mathbf{Y}$ is defined as $\|\mathbf{Y}\|_F = \sqrt{\sum_{i_1,i_2,\ldots,i_n} y^2_{i_1,\ldots,i_n}}$. It is clear that $\|\mathbf{Y}\|_F = \|\mathbf{Y}_{(k)}\|_F$ for any $1 \leq k \leq n$.

Mode-n product is the product of a tensor $\mathbf{W} \in \mathcal{R}^{p_1 \times \ldots \times p_N}$ with a matrix $A \in \mathcal{R}^{J \times p_n}$, denoted by $\mathbf{W} \times_n A$. The result is a new tensor of size $p_1 \times \ldots \times p_{n-1} \times J \times p_{n+1} \times \ldots \times p_N$, where each mode-$n$ fiber is multiplied by A, that is

$$(\mathbf{W} \times_n A)_{i_1,\ldots,i_{n-1},j,i_{n+1},\ldots,i_N} = \sum_{i_n=1}^{p_n} \mathbf{W}_{i_1,\ldots,i_N} A_{j,i_n}.$$

[2] http://www.dianping.com/

Vectorization is a linear transformation which converts a matrix into a column vector. Specially, the vectorization of a matrix $\mathbf{X} \in \mathcal{R}^{m \times n}$, denoted by $vec(\mathbf{X})$, is the $mn \times 1$ column vector obtained by stacking the columns of the matrix $\mathbf{X}$ on top of one another, that is,

$$vec(\mathbf{X}) = [x_{11}, \ldots, x_{m1}, \ldots, x_{1n}, \ldots, x_{mn}]^T,$$

where x_{ij} represents the (i,j)-th element of matrix $\mathbf{X}$. We consider that vectorization is an instance of tensor unfold, which unfolds a matrix to a vector (i.e., $\mathbf{X}_{(1)}$). Similarly, we define the $unvec$ that folds a vector to a matrix (i.e., $unvec(\mathbf{X}_{(1)}) = \mathbf{X}$).

Let $\mathbf{A} \in \mathcal{R}^{m \times n}$ and $\mathbf{B} \in \mathcal{R}^{p \times q}$ be two matrices, respectively. The Kronecker product of two matrices $\mathbf{A}$ and $\mathbf{B}$, denoted by $\mathbf{A} \otimes \mathbf{B}$, is an $mp \times nq$ matrix given by $\mathbf{A} \otimes \mathbf{B} = [a_{ij}\mathbf{B}]_{mp \times nq}$.

Tensor Decomposition and Completion has witnessed the applications in machine learning and data mining. The two popular tensor decomposition methods are CP decomposition [Acar *et al.*, 2010] and Turk decomposition [Kolda and Bader, 2009]. The tensor decomposition method aims to factorize an input tensor into a number of low-rank factors, which are prone to local optimal because they are solving essentially non-convex optimization problems [Liu *et al.*, 2013; 2009]. In order to address this problem, Liu et.al. [Liu *et al.*, 2013] extends the trace norm of matrices to tensors, and generalize matrix completion to convex tensor completion. Specially, given an incomplete n-mode tensor matrix $\mathbf{Y} \in \mathcal{R}^{I_1 \times I_2 \times \ldots \times I_n}$ with low rank, the tensor completion problem is given by

$$\begin{aligned} \min_{\mathbf{X}} \quad & rank(\mathbf{X}) \\ s.t. \quad & \mathcal{P}_\Omega(\mathbf{Y}) = \mathcal{P}_\Omega(\mathbf{X}) \end{aligned} \tag{1}$$

where $\mathbf{Y}, \mathbf{X} \in \mathcal{R}^{I_1 \times I_2 \times \ldots \times I_n}$, and $\mathcal{P}_\Omega$ keeps the entries in Ω and zeros out others. The missing elements of $\mathbf{X}$ are determined such that the rank of the tensor $\mathbf{X}$ is as small as possible. Unfortunately, the above rank minimization problem is NP-hard in general due to the nonconvexity and discontinuous nature of the rank function. Theoretical studies [Recht *et al.*, 2010] show that the nuclear norm is the tightest convex lower bound of the rank function. The trace norm of a tensor is a convex combination of the trace norms of all matrices unfolded along each mode. Therefore, a widely used approach is to apply the nuclear norm as a convex surrogate of the nonconvex tensor rank function

$$\begin{aligned} \min_{\mathbf{X}} \quad & \sum_{i=1}^{n} \|\mathbf{X}_{(i)}\|_* \\ s.t. \quad & \mathcal{P}_\Omega(\mathbf{Y}) = \mathcal{P}_\Omega(\mathbf{X}) \end{aligned} \tag{2}$$

where $\|\mathbf{X}_{(i)}\|_* = \sum_{j=1}^{\min(m,n)} \delta_j(\mathbf{X}_{(i)})$ is the nuclear norm of the i-th mode $\mathbf{X}_{(i)}$ and $\delta_j(\mathbf{X}_{(i)})$ is the j-th largest singular value of $\mathbf{X}_{(i)}$. The existing approaches [Zhang *et al.*, 2014; Liu *et al.*, 2013; Wang *et al.*, 2014; Liu *et al.*, 2014; Zhao *et al.*, 2015] based on the combination of the trace norms of all matrices have achieved excellent empirical performance.

However, the objective of our approach is to learn the tensor function from the data tensor instead of completing the

data tensor directly. Therefore, these methods are not applicable to our problem.

Query Recommendation is an important functionality in the mobile search engine. In existing work [Guo *et al.*, 2010; Feild and Allan, 2013; Li *et al.*, 2008; Anagnostopoulos *et al.*, 2010] for query recommendation aims to assist users to refine the issued queries. In this paper, we introduce a new problem of query recommendation on mobile devices, which is not limited to query refinement, but to find what users need.

3 Query Recommendation via Tensor Function Learning

In this section, we first introduce some notions for mobile query recommendation, which are the observed data tensor of user-location-query relations $\mathbf{Y}$, the side information matrix of users $\mathbf{U}$, annotation of locations matrix $\mathbf{R}$ and query intent matrix $\mathbf{Q}$. We then formally present the problem of mobile query recommendation via tensor function learning and provide the optimization algorithm to solve the problem.

We denote the observed data tensor of user-location-query relations $\mathbf{Y} \in \mathcal{R}^{m\times l\times n}$ where m is the number of queries, l is the number of locations and n is the number of users. The observed user-location-query entries in $\mathbf{Y}$ are the query issued by users at some location. However, we observe that the tensor $\mathbf{Y}$ of mobile query data is very sparse and there are a number of missing relations. We let $\mathbf{\Omega}$ be the set of observed user-location-query relations in $\mathbf{Y}$. Entry Y_{ijk} is said to be observed if the k-th user issue the i-th query at the j-th location (i.e. $(i, j, k) \in \mathbf{\Omega}$).

We consider that the query intent is the side information for the existing queries, denoted by $\mathbf{Q} \in \mathcal{R}^{I_1\times n}$. We extract the clicked webpages of the mobile queries and collect their classification labels such as "enterprise" and "business" . We note that the webpage can have multiple labels. We then denote the side information of locations by $\mathbf{R} \in \mathcal{R}^{I_2\times l}$, which is collected from the location-based entity reviewing systems. We extract the annotated tags for the locations such as "food" and "hotpot" . We represent the feature of locations and query intent using the *bag-of-words model*. We denote the side information of users by $\mathbf{U} \in \mathcal{R}^{I_3\times m}$, which is obtained from the mobile devices such as its brand and operating system. The I_1, I_2 and I_3 are the feature dimension of users, locations and query intent, respectively. Using the notions above, we introduce the problem of tensor function learning below.

3.1 The Problem

We now describe the learning problem for mobile query recommendation. Given query intent $\mathbf{Q} \in \mathcal{R}^{I_1\times n}$, annotated locations $\mathbf{R} \in \mathcal{R}^{I_2\times l}$ and user information $\mathbf{U} \in \mathcal{R}^{I_3\times m}$, we denote the parameter tensor by $\mathbf{W} \in \mathcal{R}^{I_1\times I_2\times I_3}$. Therefore, we present the tensor function by

$$f_{\mathbf{W}}(\mathbf{Q}, \mathbf{R}, \mathbf{U}) = \mathbf{W} \times_1 \mathbf{Q} \times_2 \mathbf{R} \times_3 \mathbf{U} \tag{3}$$

where $\times_1$ is the mode-1 product for parameter tensor $\mathbf{W}$ and query intent $\mathbf{Q}$. For example, the prediction for entry Y_{ijk}

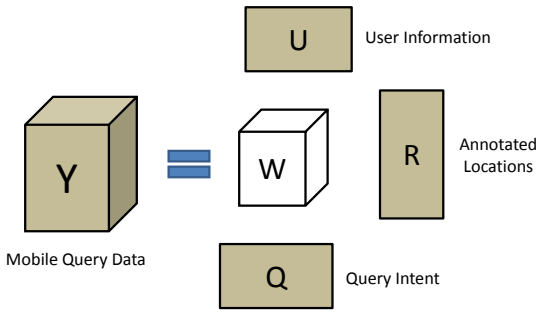


Figure 2: The Framework of Tensor Function Learning for Mobile Query Recommendation

based on parameter tensor $\mathbf{W}$ is given by

$$\hat{Y}_{ijk} = \sum_{j_1=1}^{I_1}\sum_{j_2=1}^{I_2}\sum_{j_3=1}^{I_3} W_{j_1,j_2,j_3} Q_{i,j_1} R_{j,j_2} U_{k,j_3}$$

where $Q_{i,:}$, $R_{j,:}$ and $U_{k,:}$ are the side information of the i-th query, j-th location and k-th user, respectively. We illustrate the framework of tensor function learning for mobile query recommendation in Figure 2. The tensor $\mathbf{Y}$, matrices $\mathbf{U}$, $\mathbf{R}$ and $\mathbf{Q}$ with the gray color are known and the parameter tensor $\mathbf{W}$ with the white color is unknown.

We observe that there is a strong correlation between queries, locations and users in the mobile query data $\mathbf{Y}$. For example, both the query "having an injection" and the query "where is the zoo" often co-exist in the mobile queries of some users. Thus, to avoid the overfitting problem of function learning and exploit the correlation between queries, locations and users, we encourage the parameter tensor $\mathbf{W}$ to have a simple structure in the sense that they involves a small number of "degree of freedoms". Thus, it is natural to assume that the tensor function learning are of low-rank constraints. Consequently, we cast the problem of tensor function learning into the optimization problem of tensor completion, given by

$$\begin{aligned} \min_{\mathbf{W}} \quad & rank(\mathbf{W}) \\ s.t. \quad & \mathcal{P}_\Omega(\boldsymbol{Y}) = \mathcal{P}_\Omega(f_{\mathbf{W}}(\mathbf{Q}, \mathbf{R}, \mathbf{U})) \end{aligned} \tag{4}$$

where the minimization of $rank(\mathbf{W})$ is considered as the regularizer for function learning. By requiring $\mathcal{P}_\Omega(\boldsymbol{Y}) = \mathcal{P}_\Omega(f_{\mathbf{W}}(\mathbf{Q}, \mathbf{R}, \mathbf{U}))$, we expect that the learned tensor function can accurately estimate the query-location-user relations in $\mathbf{Y}$.

Unlike the standard algorithm for tensor completion that requires solving an optimization problem involved the data matrix $\mathbf{Y}$ of $m \times l \times n$, the optimization problem given in Problem (4) only deals with the parameter tensor $\mathbf{W}$ with $I_1 \times I_2 \times I_3$. The size of parameter tensor $\mathbf{W}$ is much smaller than the data tensor $\mathbf{Y}$. Thus, the computation burden of our model is significantly more efficient.

Following the convex relaxation approach for tensor trace norm in [Liu *et al.*, 2013; Tomioka and Suzuki, 2013], we introduce the auxiliary matrix variables $\mathbf{Z}_1$, $\mathbf{Z}_2$ and $\mathbf{Z}_3$ for the unfolding of parameter tensor $\mathbf{W}$ on the mode of query

Algorithm 1 Solving Problem (5) via TFL

Input: data tensor $\mathbf{Y}$, user matrix $\mathbf{U}$, query matrix $\mathbf{Q}$ and location matrix $\mathbf{R}$
Initialize: $\mathbf{W} = \mathbf{0}$, $\mathbf{Z}_n = \mathbf{0}$, $\mathbf{\Gamma}_n = \mathbf{0}$, $\mu = 10^{-6}$, $max_\mu = 10^{10}$, $\rho = 1.05$, $\varepsilon = 10^{-8}$

1: **while** not converge **do**
2: Update $\mathbf{Z}_n$, $\mathbf{W}$ and $\mathbf{X}$ by Equation (8), (14) and (16), respectively.
3: Update the multiplier $\mathbf{\Gamma}_n$ by $\mathbf{\Gamma}_n = \mathbf{\Gamma}_n + \mu(\mathbf{W}_{(n)} - \mathbf{Z}_n)$.
4: Update the parameter μ by $\mu = \min(\rho\mu, max_\mu)$.
5: Check the convergence condition, $\max(\|\mathbf{W}_{(n)} - Z_n\|_F^2, n = 1, 2, 3) < \epsilon$.
6: **return** $\mathbf{X}$ and $\mathbf{W}$.

$\mathbf{W}_{(1)}$, location $\mathbf{W}_{(2)}$ and user $\mathbf{W}_{(3)}$, respectively. Therefore, we convert Problem (4) to the following equivalent problem:

$$\begin{aligned} \min_{\mathbf{W},\{\mathbf{Z}_n\},\mathbf{X}} \quad & \sum_{n=1}^{3} \|\mathbf{Z}_n\|_* + \frac{\lambda}{2}\|\mathbf{X} - f_{\mathbf{W}}(\mathbf{Q}, \mathbf{R}, \mathbf{U})\|_F^2 \\ s.t. \quad & \mathbf{Z}_{(n)} = \mathbf{W}_{(n)}, n = 1, 2, 3. \\ & \mathcal{P}_\Omega(\mathbf{X}) = \mathcal{P}_\Omega(\mathbf{Y}) \end{aligned} \tag{5}$$

where the summation of the auxiliary matrix variables $\sum_{n=1}^{3} \|\mathbf{Z}_n\|_*$ is the convex relaxation of $rank(\mathbf{W})$. The trade-off parameter λ balances the weight between the data penalty $\|\mathbf{X} - f_{\mathbf{W}}(\mathbf{Q}, \mathbf{R}, \mathbf{U})\|_F^2$ and the regularization term $\sum_{n=1}^{3} \|\mathbf{Z}_n\|_*$, which is usually set empirically.

3.2 The Optimization

In this section, we propose an efficient Tensor Function Learning algorithm (TFL) based on alternating direction method of multipliers (ADMM) to solve the Problem (5). ADMM [Boyd and Vandenberghe, 2004] decomposes a large problem into a series of smaller subproblems, and coordinates the solutions of subproblems to compute the optimal solution.

We rewrite Problem (5) via augmented Lagrangian function, given by

$$\begin{aligned} \mathcal{L} \quad = \quad & \sum_{n=1}^{3}(< \Gamma_n, \mathbf{W}_{(n)} - \mathbf{Z}_n > + \frac{\mu}{2}\|\mathbf{W}_{(n)} - \mathbf{Z}_{(n)}\|_F^2 \\ + \quad & \|\mathbf{Z}_n\|_*) + \frac{\lambda}{2}\|\mathbf{X} - \mathbf{W} \times_1 \mathbf{Q} \times_2 \mathbf{R} \times_3 \mathbf{U}\|_F^2 \end{aligned} \tag{6}$$

where $\mathbf{\Gamma}_n, n = 1, 2, 3$, are the matrices of Lagrange multipliers, and $\mu > 0$ is a penalty parameter. ADMM solves the proposed problem by minimizing the Lagrange function $\mathcal{L}$ over $\mathbf{W}$, $\mathbf{X}$, $\{\mathbf{Z}_n\}$, and then updating the multipliers $\{\mathbf{\Gamma_n}\}$.

We first introduce a useful tool: the singular value shrinkage operator [Cai *et al.*, 2010] to estimate the matrix $\mathbf{Z}_n$ in tensor function learning.

Definition 1 *(Singular Value Shrinkage Operator) Consider the singular value decomposition (SVD) of a matrix $\mathbf{Z} \in \mathcal{R}^{m\times n}$ of rank r,*

$$\mathbf{Z} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \mathbf{\Sigma} = diag(\{\delta_i\}_{1\le i\le r}).$$

Define the singular value shrinkage operator D_τ as follows:

$$D_\lambda(\mathbf{Z}) = \mathbf{U}D_\lambda(\mathbf{\Sigma})\mathbf{V}^T$$

and

$$D_\lambda(\mathbf{\Sigma}) = diag(\{max\{0, \delta_i - \lambda\}\}).$$

Using the singular value shrinkage operator above, we have the following useful theorems for the composite objective functions below:

Theorem 1 *([Cai* et al.*, 2010]) For each $\lambda \ge 0$ and $\mathbf{W} \in \mathcal{R}^{m\times n}$, we have*

$$D_\lambda(\mathbf{W}) = \arg\min_\alpha \frac{1}{2}\|\mathbf{Z} - \mathbf{W}\|_F^2 + \lambda\|\mathbf{Z}\|_*,$$

where the matrix $D_\lambda(\mathbf{W})$ is the solution to this optimization problem with nuclear norm regularizer.

Updating $\mathbf{Z}_n$: Ignoring constant terms, the minimization with respect to $\mathbf{Z}_n$ is given by

$$\mathbf{Z}_n = \arg\min_{\mathbf{Z}_n} \frac{1}{\mu}\|\mathbf{Z}_n\|_* + \frac{1}{2}\|\mathbf{W}_{(n)} - \mathbf{Z}_n + \frac{\mathbf{\Gamma}_n}{\mu}\|_F^2. \tag{7}$$

Thus, we can obtain the closed form solution of $\mathbf{Z}_n$ by Theorem 1 as follows:

$$\mathbf{Z}_n = D_{\frac{1}{\mu}}\left(\mathbf{W}_{(n)} + \frac{\mathbf{\Gamma}_n}{\mu}\right). \tag{8}$$

We then introduce the following Theorem to estimate the parameter tensor $\mathbf{W}$ given the side information of queries $\mathbf{Q}$, locations $\mathbf{R}$ and users $\mathbf{U}$.

Theorem 2 *(Block Matrices and Kronecker Products) Suppose $vec(\mathbf{X})$ denotes the vectorization of the matrix $\mathbf{X}$ formed by stacking the columns of $\mathbf{X}$ into a single column vector. Consider for instance the equation $\mathbf{AXB} = \mathbf{C}$, where $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are given matrices and the matrix $\mathbf{X}$ is the unknown. We can rewrite this equation as*

$$(\mathbf{B}^T \otimes \mathbf{A})vec(\mathbf{X}) = vec(\mathbf{AXB}) = vec(\mathbf{C}), \tag{9}$$

where $\otimes$ is the Kronecker product.

Updating $\mathbf{W}$: Ignoring constant terms, the minimization with respect to $\mathbf{W}$ is given by

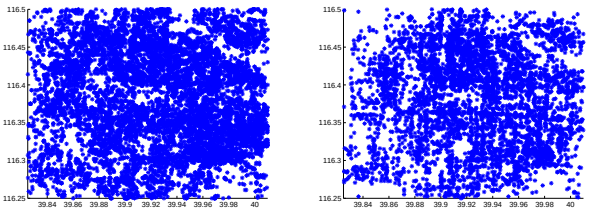
$$\begin{aligned} \mathbf{W} \quad = \quad & \arg\min_{\mathbf{W}} \sum_{n=1}^{3} \frac{\mu}{2}\|\mathbf{W}_{(n)} - \mathbf{Z}_n + \frac{\mathbf{\Gamma}_n}{\mu}\|_F^2 \\ + \quad & \frac{\lambda}{2}\|\mathbf{X} - \mathbf{W} \times_1 \mathbf{Q} \times_2 \mathbf{R} \times_3 \mathbf{U}\|_F^2. \end{aligned} \tag{10}$$

For ease of presentation, we let

$$\mathbf{A} = \sum_{n=1}^{3} refold(\mathbf{Z}_n - \frac{\mathbf{\Gamma}_n}{\mu}), \tag{11}$$

$$\mathbf{B} = \mathbf{X} \times_1 \mathbf{Q}^T \times_2 \mathbf{R}^T \times_3 \mathbf{U}^T, \tag{12}$$

$$\mathbf{C} = \mathbf{Q}\mathbf{Q}^T \otimes \mathbf{R}\mathbf{R}^T \otimes \mathbf{U}\mathbf{U}^T. \tag{13}$$



(a) GPS Distribution of Mobile Queries (b) GPS Distribution of Annotated Locations

Figure 3: Data Distribution in Beijing: The dotted points in the left figure represent the GPS of mobile queries and the dotted points in the right figure represent the GPS of annotated locations. The x-axis and y-axis represent the longitude and latitude of the dotted points, respectively.

Therefore, the optimal solution of $\mathbf{W}$ is given by

$$\mathbf{W} = unvec((3\mu\mathbf{I} + \lambda\mathbf{C})^{-1}vec(\mu\mathbf{A} + \lambda\mathbf{B})). \quad (14)$$

Updating $\mathbf{X}$: The optimization problem with respect to $\mathbf{X}$ is formulated as follows:

$$\begin{aligned} \min_{\mathbf{X}} \quad & \|\mathbf{X} - \mathbf{W} \times_1 \mathbf{Q} \times_2 \mathbf{R} \times_3 \mathbf{U}\|_F^2 \\ s.t. \quad & P_\Omega(\mathbf{X}) = P_\Omega(\mathbf{Y}) \end{aligned} \quad (15)$$

By deriving the KKT conditions for Equation (15), the optimization solution of $\mathbf{X}$ is given by

$$\mathbf{X} = P_\Omega(\mathbf{Y}) + P_{\Omega^c}(\mathbf{W} \times_1 \mathbf{Q} \times_2 \mathbf{R} \times_3 \mathbf{U}) \quad (16)$$

where Ω^c is the complement of Ω, i.e., the set of the unobserved query log entries.

The entire procedure for tensor function learning is summarized in **Algorithm** 1. The main computation cost of the algorithm in each iteration is the cost of SVD in Line 2. The addition cost in Line 3 and Line 4 is much smaller. Note that our function learning model alleviates the computation burden since the size of parameter tensor is much smaller than data tensor. For large scale problems, we can adopt some existing techniques [Li *et al.*, 2010] to accelerate the computation of SVD and make Algorithm 1 more efficient, and apply the parallel ADMM techniques [Shang *et al.*, 2014] for the proposed optimization problem. Suppose the penalty term μ in the final iteration is μ^*, the convergence of Algorithm 1 for Problem (5) is $O(\frac{1}{\mu^*})$, which is guaranteed in [Lin *et al.*, 2011].

4 Experiments

In this section, we conduct several experiments on the mobile queries from Bing search engine and location-based entity reviewing systems dianping, to show the effectiveness of our proposed approach for query recommendation on mobile devices. The experiments are conducted by using Matlab and TensorToolBox [Bader *et al.*, 2015], tested on machines with Windows OS Intel(R) Core(TM) i7-2600 CPU 3.40GHz, and 128 GB RAM.

4.1 Data Preparation

In the experiment, we apply our method on the Bing mobile dataset, which consists of users' location and their mobile queries from Bing search engine during Jan. 2014 to June 2014 in the city of Beijing, China. To protect users' privacy, we remove the GPS points for work places, homes, and users' information, and use the sampled data for doing the experiments. We sample 3,000 users and 1,000 different mobile queries from the dataset where users' id is anonymized during the sampling. Thus, we collect the 114,138 query-location-user relations. We sample 10% of the mobile queries and illustrate the GPS distribution of them in Figures 3(a).

We collect 105,271 entities from the location-based entity reviewing system, *dianping*. We sample 10% of the location-based entities and illustrate the their GPS distribution 3(b). The locations used in the experiments are in the categories of "shopping", "food", "hotel", "education", "automotive" and etc. We then obtain query intent from the clicked webpages and side information of location-based entities from the reviews in *dianping*. To further protect users' location privacy, we represent the users' location by location category. In this work, we use 30 location categories to represent users' location.

4.2 Evaluation Criteria

We evaluate the performance of our method for mobile query recommendation using RMSE (root mean-square error). We randomly sample 90% of the observed query-location-user relations in tensor $\mathbf{Y}$ as training data. We then consider the remaining 10% of the observed relations as testing data. The RMSE metric is given by

$$RMSE = \sqrt{\frac{\|\mathcal{P}_{\Omega_t}(\mathbf{Y} - f_{\mathbf{W}}(\mathbf{Q}, \mathbf{R}, \mathbf{U}))\|_F^2}{|\Omega_t|}},$$

where $\mathbf{Y}$ and $f_{\mathbf{W}}(\mathbf{Q}, \mathbf{R}, \mathbf{U}))$ are the true and predicted query-location-user relations, respectively. The set Ω_t consists the indices of all the relations in the testing data and $\mathcal{P}_{\Omega_t}$ keeps the entries in Ω_t and zeros out others. The metric RMSE has been widely adopted in the evaluation of recommender systems, such as Netflix Prize [Bennett and Lanning, 2007]. In our setting, the RMSE indicates the difference between the true existing query-location-user relations and the predicted ones. The smaller RMSE means better performance of the method. For each setting, we carry out the cross-validation on RMSE for ten times and record the mean value.

4.3 Performance Comparisons

We compare our proposed method with other five popular tensor recommendation algorithms including canonical polyadic decomposition (CP) [Acar *et al.*, 2010], Turk decomposition (Turk) [Kolda and Bader, 2009], generalized higher-order orthogonal decomposition (gHOI) [Liu *et al.*, 2014], and tensor recommendation incorporating side information (UCLAF) [Zheng *et al.*, 2010]. We summarize these algorithm as follows:

- **CP** algorithm: The canonical polyadic decomposition algorithm is a generalization of the matrix singular value

Table 1: Testing RMSE values on mobile query recommendation. Results with best mean values are bolded.

Sampled Training Data	Algorithm				
	CP	Turk	gHOI	UCLAF	TFL
50%	0.2076	0.2092	0.1702	0.1438	**0.14**
55%	0.2066	0.2088	0.1694	0.1417	**0.1288**
60%	0.2063	0.208	0.1682	0.1385	**0.1244**
65%	0.2061	0.2078	0.1682	0.1374	**0.1228**
70%	0.2051	0.2071	0.1667	0.1374	**0.1208**
75%	0.2049	0.2068	0.1667	0.1367	**0.1195**
80%	0.2046	0.2066	0.1664	0.136	**0.1153**
85%	0.2042	0.2061	0.1664	0.1352	**0.1126**
90%	0.2042	0.2056	0.1661	0.1349	**0.1104**

decomposition (SVD) to tensors, which represents a tensor by a sum of the outer products of rank-1 tensors.

- **Turk** algorithm: The Turk decomposition algorithm decomposes a tensor into a set of matrices and one small core tensor, where each matrix is orthogonal.
- **gHOI** algorithm: The generalized higher-order orthogonal decomposition algorithm decomposes a tensor under the framework of Turk decomposition, where the core tensor is regularized by nuclear norm.
- **UCLAF** algorithm: The UCLAF algorithm decomposes a tensor into a set of matrices, where the matrices are regularized by side information.

We illustrate the performance of all the algorithm in Table 1 by sampling 50% to 90% of the training data. The best mean values are bolded. We expect the relative decreasing of RMSE for a more effective recommendation method.

We summarize the experimental results in Table 1 as follows:

- We observe that the performance of the gHOI algorithm is better than both the popular tensor decomposition algorithms CP and Turk. Apart from CP and Turk, gHOI formulates the convex relaxation of the tensor decomposition problem based on the combination of the nuclear norm of all matrices and achieves excellent empirical performance.
- The UCLAF method achieves better performance than other baseline methods. This suggest that the side information can also improve the performance of mobile query recommendation.
- In all the cases, our TFL method achieves the best performance. This shows that leveraging the power of both the side information of locations and queries, and the convex formulation based on nuclear norm minimization on parameter tensor of tensor function learning, the performance of mobile query recommendation can be further improved.

4.4 Convergence Study

The updating rule for minimizing the objective function of TFL is essentially iterative. Here we investigate how TFL method converges.

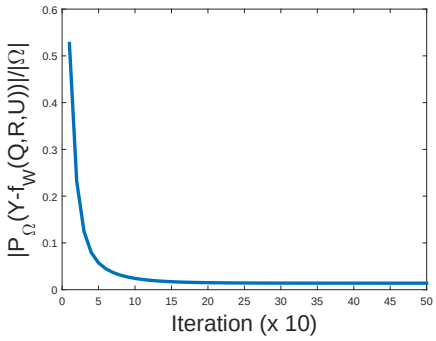


Figure 4: The convergence of TFL

Figure 4 shows the convergence curve of TFL method. The y-axis is the value of error rate on the training set (i.e., $\frac{|\mathcal{P}_\Omega(\mathbf{Y}-f_\mathbf{W}(\mathbf{Q},\mathbf{R},\mathbf{U}))|}{|\Omega|}$ and x-axis denotes the iteration number. We can observe that method TFL converges after the 500-th iteration.

5 Conclusions

In this paper, we introduce the problem of mobile query recommendation from the perspective of tensor function learning. Unlike previous studies based on tensor decomposition and completion, we propose the tensor function learning approach for estimating the missing query-location-user relations for mobile query recommendation. We learn the tensor function from the side information of the queries, locations and users with low-rank constraints. We then provide an iterative procedure for Tensor Function Learning, TFL, to solve the proposed optimization problem, using the framework of ADMM. We collect the mobile queries in Bing search engine for three months in the city of Beijing, China. We conduct the experiments on Bing mobile query dataset, and the location-based entity reviewing system, *dianping*. The experimental results demonstrate the effectiveness of our method against several state-of-the-art tensor recommendation algorithms. In the future, we will explore the kernel tensor function for the problem of mobile query recommendation.

Acknowledgements This work was supported by National Basic Research Program of China (973 Program) under Grant

2012CB316400 and Grant 2015CB352300, National Program for Special Support of Top-Notch Young Professionals, and National Natural Science Foundation of China under Grant 61233011 and Grant 61125203, and the Fundamental Research Funds for the Central Universities.

References

[Acar *et al.*, 2010] Evrim Acar, Daniel M Dunlavy, Tamara G Kolda, and Morten Mørup. Scalable tensor factorizations with missing data. In *SDM*, pages 701–712. SIAM, 2010.

[Anagnostopoulos *et al.*, 2010] Aris Anagnostopoulos, Luca Becchetti, Carlos Castillo, and Aristides Gionis. An optimization framework for query recommendation. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 161–170. ACM, 2010.

[Bader *et al.*, 2015] Brett W. Bader, Tamara G. Kolda, et al. Matlab tensor toolbox version 2.6. Available online, February 2015.

[Bennett and Lanning, 2007] James Bennett and Stan Lanning. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35, 2007.

[Boyd and Vandenberghe, 2004] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[Cai *et al.*, 2010] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

[Feild and Allan, 2013] Henry Feild and James Allan. Task-aware query recommendation. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 83–92. ACM, 2013.

[Fu *et al.*, 2009] Shunkai Fu, Bingfeng Pi, Michel Desmarais, Ying Zhou, Weilei Wang, and Song Han. Query recommendation and its usefulness evaluation on mobile search engine. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pages 1292–1297. IEEE, 2009.

[Guo *et al.*, 2010] Jiafeng Guo, Xueqi Cheng, Gu Xu, and Huawei Shen. A structured approach to query recommendation with social annotation data. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 619–628. ACM, 2010.

[Kolda and Bader, 2009] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[Li *et al.*, 2008] Lin Li, Zhenglu Yang, Ling Liu, and Masaru Kitsuregawa. Query-url bipartite based approach to personalized query recommendation. In *AAAI*, volume 8, pages 1189–1194, 2008.

[Li *et al.*, 2010] Mu Li, James T Kwok, and B-L Lu. Making large-scale nyström approximation possible. In *ICML 2010-Proceedings, 27th International Conference on Machine Learning*, page 631, 2010.

[Lin *et al.*, 2011] Zhouchen Lin, Risheng Liu, and Zhixun Su. Linearized alternating direction method with adaptive penalty for low-rank representation. In *NIPS*, pages 612–620, 2011.

[Liu *et al.*, 2009] Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for existing missing values in visual data. *ICCV*, 2009.

[Liu *et al.*, 2013] Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for estimating missing values in visual data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(1):208–220, 2013.

[Liu *et al.*, 2014] Yuanyuan Liu, Fanhua Shang, Wei Fan, James Cheng, and Hong Cheng. Generalized higher-order orthogonal iteration for tensor decomposition and completion. In *Advances in Neural Information Processing Systems*, pages 1763–1771, 2014.

[Recht *et al.*, 2010] Benjamin Recht, Maryam Fazel, and Pablo A Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

[Shang *et al.*, 2014] Fanhua Shang, Yuanyuan Liu, and James Cheng. Generalized higher-order tensor decomposition via parallel admm. *arXiv preprint arXiv:1407.1399*, 2014.

[Tomioka and Suzuki, 2013] Ryota Tomioka and Taiji Suzuki. Convex tensor decomposition via structured schatten norm regularization. In *Advances in Neural Information Processing Systems*, pages 1331–1339, 2013.

[Wang *et al.*, 2014] Hua Wang, Feiping Nie, and Heng Huang. Low-rank tensor completion with spatio-temporal consistency. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[Zhang *et al.*, 2014] Xiaoqin Zhang, Zhengyuan Zhou, Di Wang, and Yi Ma. Hybrid singular value thresholding for tensor completion. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[Zhao *et al.*, 2015] Zhou Zhao, Lijun Zhang, Xiaofei He, and Wilfred Ng. Expert finding for question answering via graph regularized matrix completion. *Knowledge and Data Engineering, IEEE Transactions on*, pages 993–1004, 2015.

[Zheng *et al.*, 2010] Vincent Wenchen Zheng, Bin Cao, Yu Zheng, Xing Xie, and Qiang Yang. Collaborative filtering meets mobile recommendation: A user-centered approach. In *AAAI*, volume 10, pages 236–241, 2010.