Why We Go Where We Go: Profiling User Decisions on Choosing POIs

Renjun Hu,[1]* Xinjiang Lu,[2]* Chuanren Liu,[3] Yanyan Li,[2] Hao Liu,[2] Shuai Ma,[1] Hui Xiong[2]
[1]SKLSDE Lab & BDBC, Beihang University, Beijing, China, {hurenjun, mashuai}@buaa.edu.cn
[2]The Business Intelligence Lab, Baidu Research, National Engineering Laboratory of Deep Learning Technology and Application, Beijing, China, {luxinjiang, liyanyanliyanyan, liuhao30}@baidu.com xionghui@gmail.com
[3]Department of Business Analytics and Statistics, University of Tennessee, Knoxville, TN, cliu89@utk.edu

Abstract

While Point-of-Interest (POI) recommendation has been a popular topic of study for some time, little progress has been made for understanding why and how people make their decisions for the selection of POIs. To this end, in this paper, we propose a user decision profiling framework, named PROUD, which can identify the key factors in people's decisions on choosing POIs. Specifically, we treat each user decision as a set of factors and provide a method for learning factor embeddings. A unique perspective of our approach is to identify key factors, while preserving decision structures seamlessly, by maximizing the sum of scalar projection of all related factor embeddings on the aggregated embedding of key factors. In addition, we show that this objective involves nonconvex quadratically constrained quadratic programming (QCQP), which remains NP-hard in general. To address this, our PROUD adopts a self projection attention and an L2 regularized sparse activation to directly estimate the likelihood of each factor to be a key factor. Finally, extensive experiments on real-world data validate the advantage of PROUD in preserving user decision structures. Also, our case study indicates that the identified key decision factors can help us to provide more interpretable recommendations and analysis.

Introduction

Decision-making is an inevitable part of our life. As estimated by various sources, an adult makes about 35,000 remotely conscious decisions each day (Hoomans 2015). With this number, understanding the reasons behind people's decisions is of great importance and benefit. Along this line, in this paper, we study user decision profiling which aims to identify the key factors of people's decisions. An example of user decision profiling is illustrated in Fig. 1, in which a user has made three decisions on choosing Point-of-Interests (POIs) for food. More specifically, POI A is chosen mainly due to distance reasons. On the other hand, POI B locates in downtown and its popularity and food quality may be attracting factors. Finally, the user chooses POI C for better user experience, *e.g.,* low waiting time and quiet environment. These reasons/factors provide a deeper understanding of user behaviors, which can in turn facilitate a broad range of applications, such as user profiling, product recommendation, business intelligence, etc.

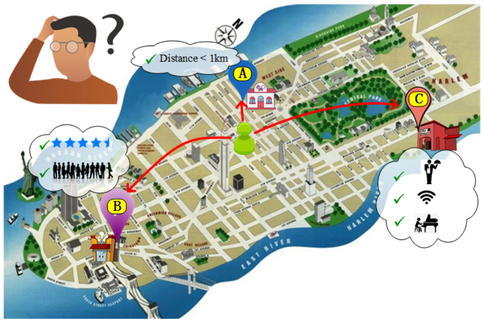


Figure 1: An illustrative example of user decision profiling

The advances of location-based services enable to investigate large-scale human mobile behaviors. We thus focus on profiling POI-choosing decisions. In literature, most studies on POI-choosing consider POI recommendation (Feng et al. 2015; Yang et al. 2017a; Massimo and Ricci 2018; Zhao et al. 2019). Despite of the effectiveness, these approaches usually have troubles in explaining their recommendations. Recently, efforts have also been made on interpretable POI recommendation (Wu and Ester 2015; Wang et al. 2018b). However, the interpretability comes from external data (Wu and Ester 2015) and the effectiveness might be sacrificed owing to feature selection (Wang et al. 2018b). Therefore, they are not suitable for our problem.

The challenge of our problem also arises from the unique characteristics of user decisions. To start with, decision profiling needs to unify heterogeneous factors. For instance, people consider both the basic spatiotemporal influence and the hidden preference and functionality impacts for choosing POIs. Second, the contributions of factors can differ greatly from one decision to another, which is hard to pre-define. Alternatively, it is more promising to determine the various factor contributions automatically. Finally, although our goal is to identify key factors, the complex decision structures also need to be preserved at the same time to ensure the goodness

[*] Both authors contribute equally to this work.

Notation	Description
$\mathcal{D} = \{f_1, \ldots, f_n\}$	a set of factors of decision $\mathcal{D}$
$\boldsymbol{f}_1, \ldots, \boldsymbol{f}_n \in \mathbb{R}^d$	low-dimensional factor embeddings
$\hat{\boldsymbol{f}} = \sum_{i=1}^{n} \boldsymbol{f}_i \in \mathbb{R}^d$	sum of factor embeddings of a decision
$\mathbf{F}, \hat{\mathbf{F}} \in \mathbb{R}^{n \times d}$	factor embedding matrices
$\mathbf{P}, \hat{\mathbf{P}} \in \mathbb{R}^{n \times n}$	pairwise scalar projection matrices
$\mathbf{P}_{ij} = \boldsymbol{f}_i^\intercal \boldsymbol{f}_j / \lvert \boldsymbol{f}_i \rvert$	scalar projection of $\boldsymbol{f}_j$ on $\boldsymbol{f}_i$
$\boldsymbol{l}, \hat{\boldsymbol{l}} \in \mathbb{R}^n$	likelihood vectors
$\boldsymbol{d} = \hat{\boldsymbol{l}}^\intercal \mathbf{F} \in \mathbb{R}^d$	aggregated key embedding
$\mathsf{VR}(\mathcal{D}), \hat{\mathsf{VR}}(\mathcal{D})$	predictive and empirical visit rates

Table 1: Summary of mathematical notations

of the identified key factors.

To this end, in this paper, we propose a novel framework, named ProUD, for PROfiling User Decisions. We collect multifaceted decision factors from different aspects and model factors via representation learning. In this way, all factors are organically integrated in ProUD. To cope with the diverse factor contributions, *i.e.,* each decision is mainly contributed by a few key factors, we propose a novel objective: We maximize the sum of scalar projection of all factor embeddings of a decision on an aggregated embedding of key factors. By projection, the impacts of non-key factors are automatically reduced. However, exactly solving the objective is non-trivial. We show that, with fixed factor embeddings, finding the desired aggregated embedding can be formulated as an NP-hard nonconvex quadratically constrained quadratic programming (QCQP). Worse still, our task requires to solve a great number of nonconvex QCQP instances. Thus, we turn to find the desired aggregated embedding in a purely data-driven manner and directly estimate the likelihood of each factor to be a key factor. As a side effect, our approach can identify key factors and preserve decision structures at the same time by maximizing scalar projection. Finally, using two real-world data sets and four metrics, we conduct extensive experiments to evaluate ProUD. We find that ProUD outperforms baselines by at least 30% for preserving decision structures. Also, our case study demonstrates that the identified key factors are reasonable and insightful for business intelligence.

To sum up, our main contributions are as follows:

- We study user decision profiling to provide explanations for people's decisions.
- We propose a novel objective for the problem and connect it with nonconvex QCQP to show the hardness.
- We devise a framework ProUD which is able to directly estimate the likelihood of each factor to be a key factor.
- We demonstrate the effectiveness of ProUD quantitatively and qualitatively through extensive experiments.

User Decision Profiling

In this section, we first formally define user decision profiling. Afterward, we introduce our main idea to attack the problem and show its hardness. The mathematical notations used throughout the paper are listed in Table 1.

Problem Definition

We start by defining several concepts related to the problem.

Definition 1 (Factor) *A factor f is a concrete and explainable item that has impacts on decision-making. For people's decisions on choosing POIs, examples of such factors include distance, time, and POI category.*

In this work, we consider decision factors from three aspects: user, POI, and decision context. User-related factors are user identifier and frequently-visited areas/POIs. POI-related factors contain POI identifier, category, brand, and POI popularity. Finally, those context-related ones are decision time (*i.e.,* hour) and the distance to home, work, and POI at the decision time, respectively. We incorporate user and POI identifiers to model the distinct impacts of individual users and POIs. Besides, according to our statistics, we discretize the continuous popularity into six levels based on the standard scores z of log-scaled popularity: (i) *strongly unpopular* if $z \leq -1$, (ii) *unpopular* if $-1 < z \leq -0.5$, (iii) *weakly unpopular* if $-0.5 < z \leq 0$, (iv) *weakly popular* if $0 < z \leq 0.5$, (v) *popular* if $0.5 < z \leq 1$, and (vi) *strongly popular* if $z > 1$. We also discretize distance into five levels: (i) *1km and less*, (ii) *between 1 and 3 km*, (iii) *between 3 and 7 km*, (iv) *between 7 and 15 km*, and (v) *15 km and more*.

Definition 2 (Decision) *A decision $\mathcal{D}$ is represented as a set of factors,* i.e., $\mathcal{D} = \{f_1, \ldots, f_n\}$ *(we note that n may vary for different $\mathcal{D}$). More specifically, a POI-choosing decision includes all factors of a specific user, a specific POI, and the corresponding decision context.*

To better understand user behaviors, we profile user decisions to discover the reasons behind. Formally, our studied problem is stated as below.

Problem 1 (User decision profiling) *Given each decision $\mathcal{D}$, identify a subset of key factors that actively contribute to $\mathcal{D}$ and determine the contributing weights of key factors.*

In this paper we focus on profiling user decisions on choosing POIs. However, it is worth pointing out that both the above problem definition and our solution are applicable to various domains as long as the basic decision factors are well defined in terms of interpretability and true impacts.

A Scalar Projection Maximization Perspective

We next present our main idea, *i.e.,* scalar projection maximization, to tackle user decision profiling. Recall that we use factors from various aspects to represent user decisions and the principle is the more the better. This is to avoid missing any possible clues. However, in practice, it is unlikely that all factors play a role in the decision-making process. Typically, each decision is mainly contributed by a few key factors that we aim to identify, while the rest are supporting factors of minor importance.

Inspired by this, we propose to distinguish key factors from supporting ones for user decision profiling. More specifically, we learn hidden representations for factors and, for each decision, compute an aggregated key embedding as a weighted combination of the embeddings of its (key) factors. We then maximize the sum of scalar projection of each

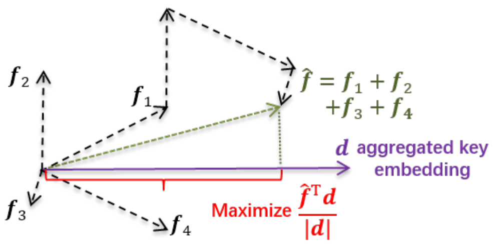

f_2
f_1
$\hat{f} = f_1 + f_2 + f_3 + f_4$
d aggregated key embedding
f_3
f_4
Maximize $\frac{\hat{f}^\intercal d}{|d|}$

Figure 2: Decision profiling by scalar projection maximization. By choosing f_1 and f_4 as key factors, the scalar projection of $\boldsymbol{f}_1 + \boldsymbol{f}_2 + \boldsymbol{f}_3 + \boldsymbol{f}_4$ on the aggregated key embedding $\boldsymbol{d}$ is maximized. (We have scaled $|\boldsymbol{d}|$ for visual purpose.)

factor embedding on the aggregated key embedding to preserve decision structures. By scalar projection, the impacts of supporting factors, *i.e.,* those not contributing much in the aggregated key embedding, are reduced. Finally, the optimal aggregated embeddings reveal key decision factors.

The above idea is further illustrated by an example in Fig. 2. Suppose we aim to profile a decision with four factors $f_1, \dots, f_4$. We learn factor embeddings $\boldsymbol{f}_1, \dots, \boldsymbol{f}_4 \in \mathbb{R}^d$, and the core is to compute an aggregated key embedding $\boldsymbol{d}$, which is the mean of $\boldsymbol{f}_1$ and $\boldsymbol{f}_4$ in our case, such that the sum of scalar projection $\hat{\boldsymbol{f}}^\intercal \boldsymbol{d}/|\boldsymbol{d}|$ is maximized, where $\hat{\boldsymbol{f}} = \sum_{i=1}^4 \boldsymbol{f}_i$ and $|\cdot|$ is the Euclidean norm. Finally, f_1 and f_4 are selected as key factors with equal contributions, while supporting factors f_2 and f_3 only have limited contributions to the decision, as evaluated by scalar projection.

Optimization Hardness

Let $\mathbf{F} = [\boldsymbol{f}_1, \dots, \boldsymbol{f}_n]^\intercal \in \mathbb{R}^{n \times d}$ be the factor embedding matrix of decision $\mathcal{D}$ and $\boldsymbol{a} \in \mathbb{R}^n$ be an n-dimensional vector. Formally, the objective of scalar projection maximization for decision $\mathcal{D}$ is as follows:

$$\max_{\mathbf{F}, \boldsymbol{a}} \frac{\left(\sum_i (\boldsymbol{a}_i \boldsymbol{f}_i)\right)^\intercal \sum_i \boldsymbol{f}_i}{|\sum_i (\boldsymbol{a}_i \boldsymbol{f}_i)|} \qquad \text{s.t. } \boldsymbol{a} \succeq \mathbf{0}. \tag{1}$$

Here, $\boldsymbol{a} \succeq \mathbf{0}$ means that $\boldsymbol{a}_i \geq 0$ for every $i \in \{1, \dots, n\}$ and vector $\boldsymbol{a}$ indicates the contributions of different factors in the aggregated key embedding, *i.e.,* $\boldsymbol{d} = \sum_i (\boldsymbol{a}_i \boldsymbol{f}_i)$. We then show the hardness of the objective, by connecting it with nonconvex quadratically constrained quadratic programming (QCQP) (Park and Boyd 2017).

Proposition 1 *When fixing* $\boldsymbol{F}$*, solving Eq. (1) can be transformed to nonconvex QCQP.*

Proof: By scaling $\boldsymbol{a}$ to ensure that $\left|\sum_i (\boldsymbol{a}_i \boldsymbol{f}_i)\right| = 1$, we derive the QCQP form of Eq. (1) when fixing $\mathbf{F}$ as below:

$$\begin{aligned} \min_{\boldsymbol{a}} \; & -\Big(\sum_i (\boldsymbol{a}_i \boldsymbol{f}_i)\Big)^\intercal \sum_i \boldsymbol{f}_i = \boldsymbol{a}^\intercal \mathbf{0} \boldsymbol{a} - (\mathbf{1}^\intercal \mathbf{F}\mathbf{F}^\intercal)\boldsymbol{a} \\ \text{s.t. } \; & \boldsymbol{a}^\intercal (\mathbf{F}\mathbf{F}^\intercal)\boldsymbol{a} - 1 \leq 0, \boldsymbol{a}^\intercal(-\mathbf{F}\mathbf{F}^\intercal)\boldsymbol{a} + 1 \leq 0, \\ & -\boldsymbol{a} \preceq \mathbf{0}. \end{aligned} \tag{2}$$

Note that for any vector $\boldsymbol{x}$, we have $\boldsymbol{x}^\intercal \mathbf{F}\mathbf{F}^\intercal \boldsymbol{x} = (\mathbf{F}^\intercal \boldsymbol{x})^\intercal \mathbf{F}^\intercal \boldsymbol{x} = |\mathbf{F}^\intercal \boldsymbol{x}|^2 \geq 0$. Thus, $\mathbf{F}\mathbf{F}^\intercal$ is positive semi-definite and, accordingly, $-\mathbf{F}\mathbf{F}^\intercal$ is negative semi-definite. With both positive and negative semi-definite constraint matrices, the QCQP in Eq. (2) is nonconvex and remains NP-hard in general (Park and Boyd 2017). □

Due to the NP-hardness, nonconvex QCQP is usually attacked for approximate solutions in practice. Besides the sub-optimality, our task requires to solve a great number of nonconvex QCQP instances: updating $\mathbf{F}$ of a specific decision once requires to solve a nonconvex QCQP instance. Recently, by mining good heuristics in a data-driven manner, machine learning has been successfully exploited for combinatorial optimization (Vinyals, Fortunato, and Jaitly 2015; Khalil et al. 2017). Following these studies, we propose our framework for profiling user decisions. Instead of solving QCQP, it directly learns the aggregated key embedding given factor embedding matrix $\mathbf{F}$ and updates $\mathbf{F}$ accordingly to maximize the sum of scalar projection.

Proposed Model

In this section, we present our decision profiling framework, named ProUD, which learns to identify key factors and preserve decision structures simultaneously.

Framework Overview

The overview of ProUD is illustrated in Fig. 3, which consists of four components:

- Input & embedding takes a decision as input and assigns a d-dimensional embedding to each of the factors.
- Self projection attention assigns another attention embedding to each factor in which projection information is well encoded. It computes a pairwise scalar projection matrix and uses it as attention weights.
- Sparse likelihood estimator evaluates the likelihood of each factor to be a key factor. It combines the initial and attention factor embeddings via a multilayer perceptron (MLP) and adopts an L2 regularized sparse activation.
- Decision structure learner computes the aggregated key embedding given likelihood and preserves decision structures by maximizing the sum of scalar projection of factor embeddings on that aggregated embedding.

We next introduce the details of these components.

Self Projection Attention

Our ProUD learns to directly evaluate the likelihood of each factor to be a key factor in a purely data-driven manner. Observe that, for each factor, whether it is a key factor or not depends much on the scalar projection of other factors on it. Intuitively, a factor is likely to be a key factor if it is well supported by a number of other factors whose scalar projection on that factor is large in general. Thus, we adopt a self projection attention to compute another embedding for each factor, which encodes projection information for learning likelihood in subsequent steps.

Formally, given factors $f_1, \dots, f_n$ in a decision and the corresponding factor embedding matrix $\mathbf{F} = [\boldsymbol{f}_1, \dots, \boldsymbol{f}_n]^\intercal$, it first computes a pairwise scalar projection matrix $\mathbf{P} \in$

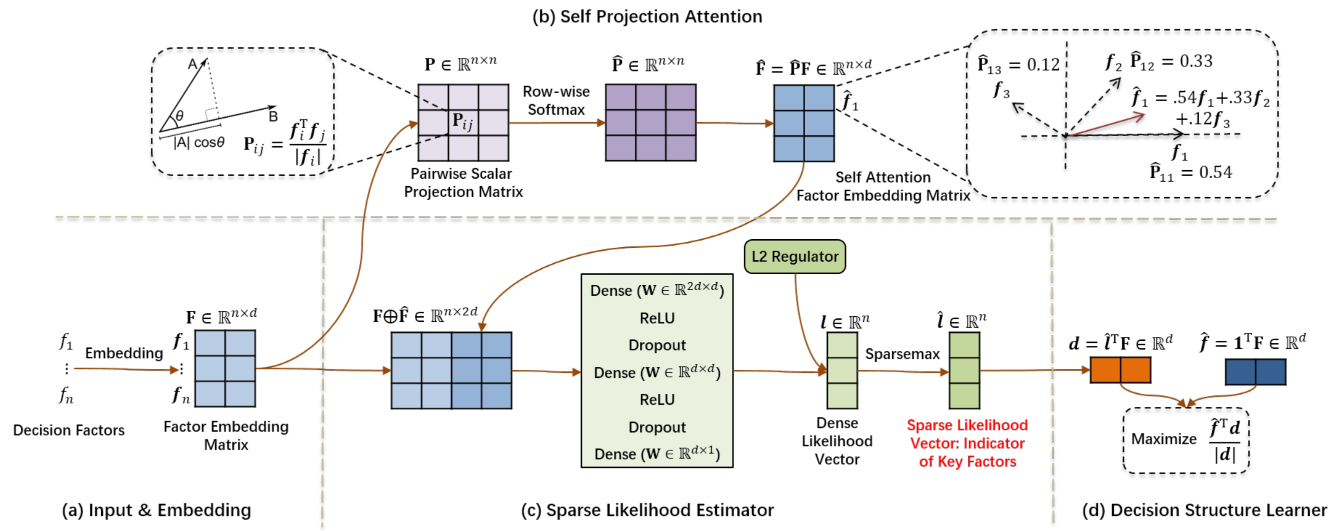

(b) Self Projection Attention
A
B
θ
\|A\| cosθ
$\mathbf{P}_{ij} = \frac{f_i^\mathsf{T} f_j}{|f_i|}$
$\mathbf{P} \in \mathbb{R}^{n\times n}$
$\mathbf{P}_{ij}$
Pairwise Scalar Projection Matrix
Row-wise Softmax
$\hat{\mathbf{P}} \in \mathbb{R}^{n\times n}$
$\hat{\mathbf{F}} = \hat{\mathbf{P}}\mathbf{F} \in \mathbb{R}^{n\times d}$
$\hat{f}_1$
Self Attention Factor Embedding Matrix
$\hat{\mathbf{P}}_{13} = 0.12$
f_3
f_2 $\hat{\mathbf{P}}_{12} = 0.33$
$\hat{f}_1 = .54f_1 + .33f_2 + .12f_3$
f_1
$\hat{\mathbf{P}}_{11} = 0.54$
f_1
⋮
f_n
Embedding
Decision Factors
$\mathbf{F} \in \mathbb{R}^{n\times d}$
f_1
f_n
Factor Embedding Matrix
(a) Input & Embedding
$\mathbf{F}\oplus\hat{\mathbf{F}} \in \mathbb{R}^{n\times 2d}$
Dense ($\mathbf{W} \in \mathbb{R}^{2d\times d}$)
ReLU
Dropout
Dense ($\mathbf{W} \in \mathbb{R}^{d\times d}$)
ReLU
Dropout
Dense ($\mathbf{W} \in \mathbb{R}^{d\times 1}$)
L2 Regulator
$\boldsymbol{l} \in \mathbb{R}^n$
Sparsemax
$\hat{\boldsymbol{l}} \in \mathbb{R}^n$
Dense Likelihood Vector
Sparse Likelihood Vector: Indicator of Key Factors
(c) Sparse Likelihood Estimator
$\boldsymbol{d} = \hat{\boldsymbol{l}}^\mathsf{T}\mathbf{F} \in \mathbb{R}^d$
$\hat{\boldsymbol{f}} = \mathbf{1}^\mathsf{T}\mathbf{F} \in \mathbb{R}^d$
Maximize $\frac{\hat{\boldsymbol{f}}^\mathsf{T}\boldsymbol{d}}{|\boldsymbol{d}|}$
(d) Decision Structure Learner

Figure 3: Framework overview of PROUD

$\mathbb{R}^{n\times n}$, in which $\mathbf{P}_{ij} = \boldsymbol{f}_i^\mathsf{T}\boldsymbol{f}_j/|\boldsymbol{f}_i|$ is the scalar projection of $\boldsymbol{f}_j$ on $\boldsymbol{f}_i$. It then normalizes $\mathbf{P}$ with row-wise softmax:

$$\hat{\mathbf{P}}_{i:} = \mathsf{softmax}(\mathbf{P}_{i:}),\ i \in \{1,\ldots,n\}. \tag{3}$$

Afterward, it computes an attention embedding $\hat{\boldsymbol{f}}_i = \sum_{j=1}^n \hat{\mathbf{P}}_{ij}\boldsymbol{f}_j \in \mathbb{R}^d$ for each factor i, which is the sum of all factor embeddings $\boldsymbol{f}_j$ weighted by $\hat{\mathbf{P}}_{ij}$. An example of computing attention factor embeddings is illustrated at the upper-right corner of Fig. 3. As can be seen, projection information is well encoded in $\hat{\boldsymbol{f}}_1$ such that factors with higher scalar projection, *i.e.,* f_1 and f_2, contribute more to $\hat{\boldsymbol{f}}_1$. Finally, the matrix form of those attention factor embeddings is given by $\hat{\mathbf{F}} = \hat{\mathbf{P}}\mathbf{F} \in \mathbb{R}^{n\times d}$.

Sparse Likelihood Estimator

With both initial and attention factor embeddings, the sparse likelihood estimator component evaluates the likelihood of each factor to be a key factor. We first concatenate the two factor embedding matrices into $\mathbf{F}\oplus\hat{\mathbf{F}} \in \mathbb{R}^{n\times 2d}$ and feed it to a three-layer MLP to derive an unnormalized likelihood vector $\boldsymbol{l} \in \mathbb{R}^n$ for the n factors:

$$\begin{aligned} \mathbf{L} &= \mathsf{Dropout}(\mathsf{ReLU}((\mathbf{F}\oplus\hat{\mathbf{F}})\mathbf{W}_1 + \boldsymbol{b}_1)), \\ \boldsymbol{l} &= \mathsf{Dropout}(\mathsf{ReLU}(\mathbf{L}\mathbf{W}_2 + \boldsymbol{b}_2))\mathbf{W}_3. \end{aligned} \tag{4}$$

Here $\mathbf{W}_1 \in \mathbb{R}^{2d\times d}$, $\mathbf{W}_2 \in \mathbb{R}^{d\times d}$, $\mathbf{W}_3 \in \mathbb{R}^{d\times 1}$, $\boldsymbol{b}_1 \in \mathbb{R}^d$, and $\boldsymbol{b}_2 \in \mathbb{R}^d$ are trainable MLP parameters. The unnormalized likelihood $\boldsymbol{l}_i$ of factor f_i is determined by both $\boldsymbol{f}_i$ and $\hat{\boldsymbol{f}}_i$. In other words, the likelihood of each factor has considered both the factor itself and other factors, with an emphasis on those having high scalar projection on $\boldsymbol{f}_i$.

We then adopt a sparse activation function to normalize $\boldsymbol{l}$ to be a valid likelihood vector. In the meanwhile, the sparsity constraint automatically retains key factors while ignores those supporting ones. We consider sparsemax (Martins and Astudillo 2016) which is similar to softmax except for outputting sparse probabilities. Formally, sparsemax ensures sparsity by returning the Euclidean projection of the input vector onto the probability simplex:

$$\mathsf{sparsemax}(\boldsymbol{l}) = \operatorname*{argmin}_{\boldsymbol{p}\in\Delta^{n-1}} |\boldsymbol{p} - \boldsymbol{l}|, \tag{5}$$

where $\Delta^{n-1} = \{\boldsymbol{p} \in \mathbb{R}^n \mid \mathbf{1}^\mathsf{T}\boldsymbol{p} = 1, \boldsymbol{p} \succeq \mathbf{0}\}$ is the probability simplex for n-dimensional vectors. In practice, $\mathsf{sparsemax}(\boldsymbol{l})$ can be easily computed in $O(n\log n)$ time by sorting and linearly scanning the entries in $\boldsymbol{l}$.

Finally, $\hat{\boldsymbol{l}} = \mathsf{sparsemax}(\boldsymbol{l}) \in \mathbb{R}^n$ is the valid likelihood vector indicating which factors are key factors and their contributing weights. The number of non-zero entries in $\hat{\boldsymbol{l}}$ is not controllable given its definition. However, there is usually a need to provide flexibility for the number of activated entries, *i.e.,* key factors. Thus, we further equip an L2 regulator on the unnormalized $\boldsymbol{l}$ before sparsemax. Note that the larger the L2 weight is, the more key factors $\hat{\boldsymbol{l}}$ identifies.

Decision Structure Learner

Given likelihood vector $\hat{\boldsymbol{l}}$, the aggregated key embedding $\boldsymbol{d} \in \mathbb{R}^d$ of decision $\mathcal{D}$ is computed as a weighted sum of key factor embeddings: $\boldsymbol{d} = \sum_{i=1}^n \hat{\boldsymbol{l}}_i\boldsymbol{f}_i = \hat{\boldsymbol{l}}^\mathsf{T}\mathbf{F}$. We then preserve decision structures by maximizing the sum of scalar projection of all related factor embeddings on $\boldsymbol{d}$:

$$\max_{\mathbf{F}} \hat{\boldsymbol{f}}^\mathsf{T}\boldsymbol{d}/|\boldsymbol{d}|. \tag{6}$$

Recall that $\hat{\boldsymbol{f}} = \sum_{i=1}^n \boldsymbol{f}_i$ and, compared with Eq. (1), the objective does not need to search for vector $\boldsymbol{a}$ with QCQP.

To train factor embeddings, we need both positive and negative decision instances (Wang et al. 2018a). The decisions $\mathcal{D}$ we refer to by far are all positive: The user chooses the POI to visit under certain context. We then denote the empirical visit rate $\hat{\mathsf{VR}}(\mathcal{D}) = 1$. For each positive decision $\mathcal{D}$, we can generate several negative instances $\mathcal{D}^-$ by replacing the POI-related factors in $\mathcal{D}$ with factors of other

Description	BEIJING	NYC
time spanning	3/20/18∼8/30/18	4/12/12∼2/16/13
# of users	90,090	1,083
# of POIs	169,528	109,018
# of factors	285,099	122,343
# of positive $\mathcal{D}$	199,106	146,325
# of negative $\mathcal{D}^-$	1,694,365	1,282,302
# of factors per $\mathcal{D}$/$\mathcal{D}^-$	20.5	45

Table 2: Data set statistics

POIs that the user does not decide to visit. Typical alternative POIs for negative instances can be those near the visited one or those of the same category as the visited POI. Similarly, we denote the empirical visit rate $\hat{\mathsf{VR}}(\mathcal{D}^-) = 0$. The predictive visit rates $\mathsf{VR}(\mathcal{D})/\mathsf{VR}(\mathcal{D}^-)$ of both positive and negative decisions are determined by scalar projection, *i.e.,* $\sigma(\hat{\boldsymbol{f}}^\intercal \boldsymbol{d}/|\boldsymbol{d}|)$, where $\sigma(x) = 1/(1+e^{-x})$ is the sigmoid function that maps arbitrary scalar projection into a probability within $[0, 1]$.

Note that $\mathsf{VR}(\cdot)$ and $\hat{\mathsf{VR}}(\cdot)$ can be regarded as the predictive and empirical distributions of visit rates of decision instances. We then learn to preserve decision structures via minimizing the following objective:

$$O = \mathsf{dist}\big(\mathsf{VR}(\cdot), \hat{\mathsf{VR}}(\cdot)\big), \tag{7}$$

where $\mathsf{dist}(\cdot, \cdot)$ refers to a metric that evaluates the distance between two distributions. Replacing $\mathsf{dist}(\cdot, \cdot)$ with KL-divergence and ignoring the constant terms, we have the following to minimize (Tang et al. 2015):

$$O = -\sum_{\mathcal{D}} \log \mathsf{VR}(\mathcal{D}) - \sum_{\mathcal{D}^-} \log(1 - \mathsf{VR}(\mathcal{D}^-)). \tag{8}$$

That is, the objective sums over all decision instances and maximizes $\mathsf{VR}(\mathcal{D})$ for positive ones while minimizes $\mathsf{VR}(\mathcal{D}^-)$ for negative ones. Note that maximizing $\mathsf{VR}(\mathcal{D})$ (or, minimizing $\mathsf{VR}(\mathcal{D}^-)$) is indeed maximizing (or, minimizing) the corresponding scalar projection $\hat{\boldsymbol{f}}^\intercal \boldsymbol{d}/|\boldsymbol{d}|$ as σ keeps monotonically increasing on $\mathbb{R}$.

Experiments

In this section we conduct extensive experiments to evaluate our PROUD framework for profiling user decisions. Due to the unavailability of decision records with ground-truth key factors, we evaluate the effectiveness of PROUD for preserving user decision structures, *i.e.,* how PROUD distinguishes positive decision instances from negative ones. Besides, we also present a case study for qualitatively assessing the key factors identified by PROUD.

Experimental Setups

We first present the setting of our experiments.

Data sets. We chose two data sets to test our approach.
(1) BEIJING was produced using the map query and mobility data provided by Baidu Map.[1] Each query was associated with an anonymous user identifier, a location, a time stamp, and a list of related POIs. For each query, we constructed a positive decision instance if the user visited one POI in the list in following two days, and constructed negative instances with those unvisited POIs. We said a user visited a POI if (i) the user connected the Wi-Fi of the POI, or (ii) the user generated a mobility record, *i.e.,* a pair of location and time stamp, which was no farther than 100 meters from the POI. We discarded a map query if no POIs in the list were visited. The encrypted user home, work, and frequently-visited area information and the POI popularity data were also provided by the platform.
(2) NYC was produced based on a public Foursquare check-in data set (Yang et al. 2015). We collected POIs of NYC and the numbers of likes to POIs (for evaluating POI popularity) with Foursquare developers APIs.[2] We treated each check-in as a positive decision and generated negative instances by replacing the POI with (i) those nearby and (ii) those of the same category. We mined frequently-visited POIs of users from check-in histories while did not consider distance factors since decision locations were unknown.

For each data set, we randomly split the data into 70% for training, 10% for validation, and 20% for testing. The statistics of our data are illustrated in Table 2.

Metrics. We adopted Prec (Precision), Recall, $\mathsf{F1}$, and AUC (Area Under the ROC Curve) to evaluate the performance. We used the optimal threshold on validation set to compute Prec, Recall, and $\mathsf{F1}$ on test data. Note that for all metrics, a higher score indicates a better performance.

Algorithms. We compared our approach with various baselines that could be used to preserve decision structures.

- LINE (Tang et al. 2015) is a network embedding method that preserves first- and second-order proximities. We constructed a POI graph based on co-visiting relationships and learned POI embeddings. Users were then represented as the sum of embeddings of POIs visited by users. Finally, we used the inner products of user and POI embeddings for preserving user decision structures.
- GE (Xie et al. 2016) is a graph embedding approach to location recommendation. It constructs four graphs (*i.e.,* POI-POI, POI-region, POI-time, and POI-word) and learns embeddings of these entities via preserving graph structures. We treated POI categories as words and users were represented the same to LINE. Finally, we adopted a logistic regression to combine the proximities between these entities for the task.
- MP2VEC (Dong, Chawla, and Swami 2017) is a heterogeneous network embedding method that learns node representations from metapath-based random walks. We constructed a heterogeneous network from decision data and considered the following metapath: POI-X-user-X-POI-$\cdots$, where X could be any types except for user and POI. Predictions were done the same to GE.
- LEARNSUC (Wang et al. 2018a) denotes behavior records as multi-type itemsets and learns behavior success by preserving itemset structures. It differs from our approach that the success rate is modeled as the length of the sum

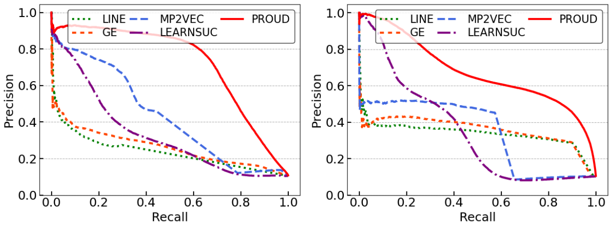
[1] https://map.baidu.com/

[2] https://developer.foursquare.com/

Algorithm	BEIJING				NYC			
	Prec	Recall	F1	AUC	Prec	Recall	F1	AUC
LINE	0.2684	0.2433	0.2552	0.5827	0.3528	0.3835	0.3261	0.6459
GE	0.3005	0.3499	0.3234	0.6271	0.3767	0.5262	0.4391	0.7134
MP2VEC	0.4594	0.4180	0.4377	0.6801	0.4665	0.5247	0.4939	0.7281
LEARNSUC	0.2222	0.5849	0.3221	0.6791	0.3984	0.4011	0.3993	0.5600
PROUD	**0.7637**	**0.6375**	**0.6949**	**0.9248**	**0.5487**	**0.7743**	**0.6422**	**0.9439**

PROUD significantly outperforms all approaches at the 0.01 level, paired t-test.

Table 3: Accuracy evaluation on preserving decision structures



(a) BEIJING

(b) NYC

Figure 4: Precision-recall curves

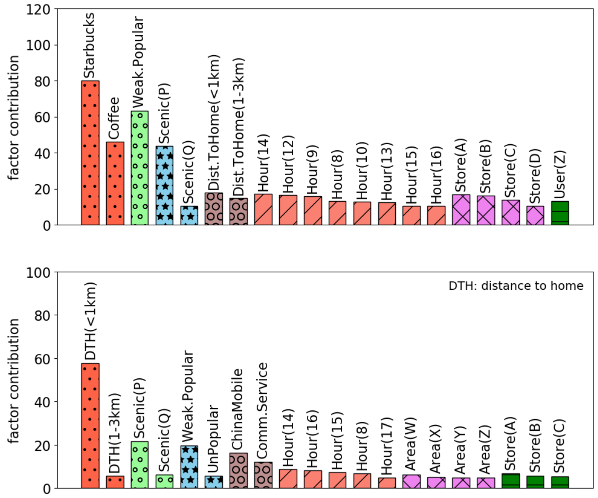


Figure 5: Top-20 key factors of Starbucks (up) and ChinaMobile (bottom) on BEIJING

of item embeddings included in a behavior and the item contributions remain fixed.

Implementation. We used the Adam optimizer with default parameters and a batch size of 512 to train our PROUD. The learning rate γ was set to 0.01 at first and decayed to 0.7γ after each epoch. We employed three types of regularization during training: (i) an L2 regularization with weight 10^{-5} on all trainable parameters and the unnormalized likelihood vector $\boldsymbol{l}$, (ii) a dropout with $P_{drop} = 0.2$ in Eq. (4), and (iii) an early stopping if the F1 on validation set did not increase in successive 5 epochs. The number d of dimensions was fixed to 64. Note that these hyperparameters were tuned on validation set. When quantity measures were evaluated, the test was repeated over 5 times using different data splits and the average was reported.

Experimental Results

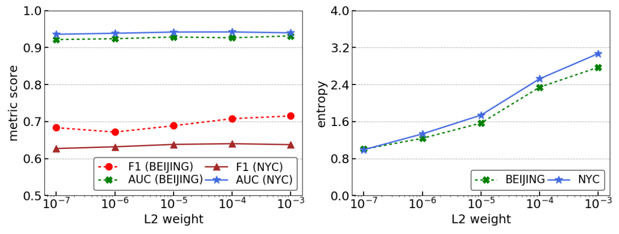
We next present the results of our experiments.

Exp-1: Preserving decision structures. We first evaluate the overall performance of considered approaches for distinguishing positive and negative user decision instances. Recall that PROUD preserves decision structures via identifying key factors and maximizing scalar projection. The effectiveness of PROUD for preserving decision structures directly relies on the goodness of identified key factors. The results are reported in Table 3.

With Prec, Recall, and F1, *i.e.,* predicting each instance as either positive or negative, PROUD consistently outperforms all other baselines. The conclusion is significant at the 0.01 level with paired t-test. Notably, PROUD is the only method whose majority of predicted positive instances are true positive. This is achieved without sacrificing Recall. Indeed, PROUD also recalls the most positive instances among all methods. As a result, the F1 of PROUD is on average (172%, 115%, 59%, 116%) and (97%, 46%, 30%, 61%) higher than (LINE, GE, MP2VEC, LEARNSUC) on BEIJING and NYC, respectively.

With AUC, *i.e.,* ranking instances according to their likelihood of being positive, PROUD also dominates the comparison on both data sets. Indeed, the AUC of PROUD is on average (59%, 47%, 36%, 36%) and (46%, 32%, 30%, 69%) higher than (LINE, GE, MP2VEC, LEARNSUC) on BEIJING and NYC, respectively. According to the meaning of AUC, PROUD can rank a random positive instance higher than a random negative instance with probability 0.92.

To give a comprehensive understanding of how these approaches preserve user decision structures, we present the precision-recall curves of all tested approaches on both data sets in Fig. 4. The Prec of LINE and GE soon drops to between 0.4 and 0.5 when the Recall slightly exceeds 0, and then keeps decreasing with the increment of Recall. The situation of MP2VEC on NYC is similar. On the other hand, the Prec of MP2VEC on BEIJING and of LEARNSUC remains at a relatively high level when Recall is small. In other words, they can identify a fraction of positive instances with a high accuracy. Finally, the Prec of PROUD is consistently higher than others at all levels of Recall. The gap between PROUD and other approaches clearly confirms its superiority for preserving user decision structures.



(a) F1 and AUC (b) Entropy of vector $\hat{l}$

Figure 6: Impacts of L2 regulator for likelihood vectors

Exp-2: Case study. We next present a case study on the identified key factors to further evaluate PROUD. We collected all positive decisions for Starbucks and ChinaMobile (a major communication service provider in China) on BEIJING and listed their top-20 key factors in Fig. 5. As shown, these key factors are reasonable and insightful in general.

For Starbucks, the top-1 key factor is the brand. Note that nowadays Starbucks has become the most popular chain coffee brand in Beijing. Similarly, the category Coffee is another main key factor. Moreover, people also consider the popularity of stores when making decisions. Besides, we identify two scenic spots and, inspired, Starbucks may consider to expand its business in their surrounding areas. We also find that distance and time are two key factors: People make decisions for Starbucks when they are near their homes and usually visit stores in late morning and afternoon hours. Finally, stores themselves can be an importance factor. The identified four stores are located in business areas or popular residence areas of foreigners and, thus, attract a large number of regular customers.

For ChinaMobile, the impacts of brand and service decrease while distance and popularity factors play a more crucial role. Similarly, time such as early morning and afternoon remains influential. Moreover, we identify four resident areas as driving factors for ChinaMobile.

Exp-3: Impacts of L2 regulator. We adopt an L2 regulator on the unnormalized likelihood vector to control the number of key factors. We finally evaluate the impacts of the regulator. The results of F1, AUC, and the entropy of the normalized likelihood vector $\hat{l}$ are reported in Fig. 6.

When increasing the L2 weight from 10^{-7} to 10^{-3}, both F1 and AUC keep stable in general, with an exception that the F1 on BEIJING is slightly better with larger L2 weights. We omit Prec and Recall, and their results are similar. On the other hand, the entropy keeps increasing: More key factors are identified with a stronger L2 regulator. To conclude, the regulator can provide flexibility for key factors without sacrificing the overall effectiveness.

Related Work

Explainable recommender systems aim to yield both recommendations and explanations (Zhang and Chen 2018). As such, user satisfaction as well as system effectiveness and transparency can be improved. The popular matrix factorization methods usually have troubles in interpreting the meanings of representation vectors. To tackle the issue, several models, such as explicit factor models (Zhang et al. 2014) and explainable matrix factorization (Abdollahi and Nasraoui 2017), have been developed. In addition, graph learning is also leveraged, such as graph-based propagation (He et al. 2015) and graph clustering (Heckel et al. 2017). More recently, due to the great success achieved in a number of domains, deep learning is widely exploited for recommendation (Seo et al. 2017; Donkers, Loepp, and Ziegler 2017; Chen et al. 2018; Li et al. 2019).

Although explainable recommendation has been applied in many scenarios, the progress for POIs is limited. In (Wu and Ester 2015), the authors propose a probabilistic model which combines aspect-based opinion mining and collaborative filtering to provide explainable recommendations. Besides, (Wang et al. 2018b) exploits a tree-enhanced embedding model for interpretable tourist and restaurant recommendation. In this paper, we study how to identify the key factors contributing to people's decisions on choosing POIs. We learn representations for interpretable factors to preserve decision structures, which differentiates our work from priors. By this "decision profiling" capability, we are able to make recommendations and, at the same time, provide the key factors behind as explanations.

Contextual representation learning proposes to tackle relationship-centric tasks and combinational problems via learning latent representations, such as factor models (Jamali and Lakshmanan 2013) and matrix/tensor decomposition (Yang et al. 2017b; Zhou et al. 2019). Network embedding is among the most successful for capturing semantics of item interactions. For example, LINE (Tang et al. 2015) provides clear objectives to learn homogeneous relationships. In addition, MetaPath2Vec (Dong, Chawla, and Swami 2017) is further proposed to deal with heterogeneous structural information. Network embedding has also been proven effective for a number of tasks such as anomaly detection (Hu et al. 2016) and multi-modal transportation recommendation (Liu et al. 2019a; 2019b). Recently, (Wang et al. 2018a) represents a behavior as a multi-type itemset and learns the collective interactions of items to preserve the success rate of each behavior. Moreover, by learning good heuristics automatically from data, representation learning approaches have been exploited for combinatorial optimization problems (Vinyals, Fortunato, and Jaitly 2015; Khalil et al. 2017).

Our work also learns representations for a set of factors. However, we devise a novel scalar projection maximization objective, which has not been considered before. The self projection attention and L2-regularized sparse activation are also deeply coupled with our problem.

Concluding Remarks

In this paper, we studied user decision profiling to provide explanations for people's decisions. We represented each user decision as a set of factors and identified key factors. By learning factor representations, we showed that maximizing the sum of scalar projection of related factor embeddings on the aggregated embedding of key factors is a good objective to tackle the problem. Such an objective involves nonconvex quadratically constrained quadratic programming, which remains NP-hard in general. We proposed to directly learn the likelihood of each factor to be a key factor, with a self projection attention and an L2-regularized sparse activation. Using real-world data, we conducted extensive experiments to demonstrate the advantage of our approach. It achieved the best performance for preserving user decision structures, which indirectly verified the goodness of the identified key factors. We also presented a case study to show the interpretability and usefulness of key factors.

Acknowledgments

This work is supported in part by NSFC 61925203 & U1636210 & 61421003 & 71531001.

References

Abdollahi, B., and Nasraoui, O. 2017. Using explainability for constrained matrix factorization. In *RecSys*.

Chen, C.; Zhang, M.; Liu, Y.; and Ma, S. 2018. Neural attentional rating regression with review-level explanations. In *WWW*.

Dong, Y.; Chawla, N. V.; and Swami, A. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *SIGKDD*.

Donkers, T.; Loepp, B.; and Ziegler, J. 2017. Sequential user-based recurrent neural network recommendations. In *RecSys*.

Feng, S.; Li, X.; Zeng, Y.; Cong, G.; Chee, Y. M.; and Yuan, Q. 2015. Personalized ranking metric embedding for next new poi recommendation. In *IJCAI*.

He, X.; Chen, T.; Kan, M.-Y.; and Chen, X. 2015. Trirank: Review-aware explainable recommendation by modeling aspects. In *CIKM*.

Heckel, R.; Vlachos, M.; Parnell, T.; and Dünner, C. 2017. Scalable and interpretable product recommendations via overlapping co-clustering. In *ICDE*.

Hoomans, J. 2015. 35,000 decisions: The great choices of strategic leaders. https://go.roberts.edu/leadingedge/the-great-choices-of-strategic-leaders. Accessed: 2019-09-04.

Hu, R.; Aggarwal, C. C.; Ma, S.; and Huai, J. 2016. An embedding approach to anomaly detection. In *ICDE*.

Jamali, M., and Lakshmanan, L. 2013. Heteromf: recommendation in heterogeneous information networks using context dependent factor models. In *WWW*.

Khalil, E. B.; Dai, H.; Zhang, Y.; Dilkina, B.; and Song, L. 2017. Learning combinatorial optimization algorithms over graphs. In *NIPS*.

Li, C.; Quan, C.; Peng, L.; Qi, Y.; Deng, Y.; and Wu, L. 2019. A capsule network for recommendation and explaining what you like and dislike. In *SIGIR*.

Liu, H.; Li, T.; Hu, R.; Fu, Y.; Gu, J.; and Xiong, H. 2019a. Joint representation learning for multi-modal transportation recommendation. In *AAAI*.

Liu, H.; Tong, Y.; Zhang, P.; Lu, X.; Duan, J.; and Xiong, H. 2019b. Hydra: A personalized and context-aware multimodal transportation recommendation system. In *SIGKDD*.

Martins, A. F. T., and Astudillo, R. F. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *ICML*.

Massimo, D., and Ricci, F. 2018. Harnessing a generalised user behaviour model for next-poi recommendation. In *RecSys*.

Park, J., and Boyd, S. 2017. General heuristics for nonconvex quadratically constrained quadratic programming. *arXiv preprint arXiv:1703.07870*.

Seo, S.; Huang, J.; Yang, H.; and Liu, Y. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *RecSys*.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. LINE: large-scale information network embedding. In *WWW*.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2015. Pointer networks. In *NIPS*.

Wang, D.; Jiang, M.; Zeng, Q.; Eberhart, Z.; and Chawla, N. V. 2018a. Multi-type itemset embedding for learning behavior success. In *SIGKDD*.

Wang, X.; He, X.; Feng, F.; Nie, L.; and Chua, T.-S. 2018b. Tem: Tree-enhanced embedding model for explainable recommendation. In *WWW*.

Wu, Y., and Ester, M. 2015. Flame: A probabilistic model combining aspect based opinion mining and collaborative filtering. In *WSDM*.

Xie, M.; Yin, H.; Wang, H.; Xu, F.; Chen, W.; and Wang, S. 2016. Learning graph-based poi embedding for location-based recommendation. In *CIKM*.

Yang, D.; Zhang, D.; Zheng, V. W.; and Yu, Z. 2015. Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45(1):129–142.

Yang, C.; Bai, L.; Zhang, C.; Yuan, Q.; and Han, J. 2017a. Bridging collaborative filtering and semi-supervised learning: a neural approach for poi recommendation. In *SIGKDD*.

Yang, K.; Li, X.; Liu, H.; Mei, J.; Xie, G.; Zhao, J.; Xie, B.; and Wang, F. 2017b. Tagited: Predictive task guided tensor decomposition for representation learning from electronic health records. In *AAAI*.

Zhang, Y., and Chen, X. 2018. Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192*.

Zhang, Y.; Lai, G.; Zhang, M.; Zhang, Y.; Liu, Y.; and Ma, S. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *SIGIR*.

Zhao, P.; Zhu, H.; Liu, Y.; Xu, J.; Li, Z.; Zhuang, F.; Sheng, V. S.; and Zhou, X. 2019. Where to go next: a spatio-temporal gated network for next poi recommendation. In *AAAI*.

Zhou, J.; Gou, S.; Hu, R.; Zhang, D.; Xu, J.; Wu, X.; Jiang, A.; and Xiong, H. 2019. A collaborative learning framework to tag refinement for points of interest. In *SIGKDD*.