TENSOR LEARNING FOR RECOVERING MISSING INFORMATION: ALGORITHMS AND APPLICATIONS ON SOCIAL MEDIA

A Dissertation

by

HANCHENG GE

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	James Caverlee
Committee Members,	Frank M. Shipman
	Xia Hu
	Jianhua Huang
Head of Department,	Dilma Da Silva

December 2017

Major Subject: Computer Engineering

ABSTRACT

Real-time social systems like Facebook, Twitter, and Snapchat have been growing rapidly, producing exabytes of data in different views or aspects. Coupled with more and more GPS-enabled sharing of videos, images, blogs, and tweets that provide valuable information regarding "who", "where", "when" and "what", these real-time human sensor data promise new research opportunities to uncover models of user behavior, mobility, and information sharing. These real-time dynamics in social systems usually come in multiple aspects, which are able to help better understand the social interactions of the underlying network. However, these multi-aspect datasets are often raw and incomplete owing to various unpredictable or unavoidable reasons; for instance, API limitations and data sampling policies can lead to an incomplete (and often biased) perspective on these multi-aspect datasets. This missing data could raise serious concerns such as biased estimations on structural properties of the network and properties of information cascades in social networks. In order to recover missing values or information in social systems, we identify "**4S**" challenges: extreme **sparsity** of the observed multi-aspect datasets, adoption of rich side information that is able to describe the **similarities** of entities, generation of robust models rather than limiting them on **specific** applications, and **scalability** of models to handle real large-scale datasets (billions of observed entries). With these challenges in mind, this dissertation aims to develop scalable and interpretable tensor-based frameworks, algorithms and methods for recovering missing information on social media. In particular, this dissertation research makes four unique contributions:

- The first research contribution of this dissertation research is to propose a scalable framework based on low-rank tensor learning in the presence of incomplete information. Concretely, we formally define the problem of recovering the spatio-temporal

dynamics of online memes and tackle this problem by proposing a novel tensor-based factorization approach based on the alternative direction method of multipliers (ADMM) with the integration of the latent relationships derived from contextual information among locations, memes, and times.

- The second research contribution of this dissertation research is to evaluate the generalization of the proposed tensor learning framework and extend it to the recommendation problem. In particular, we develop a novel tensor-based approach to solve the personalized expert recommendation by integrating both the latent relationships between homogeneous entities (e.g., users and users, experts and experts) and the relationships between heterogeneous entities (e.g., users and experts, topics and experts) from the geo-spatial, topical, and social contexts.

- The third research contribution of this dissertation research is to extend the proposed tensor learning framework to the user topical profiling problem. Specifically, we propose a tensor-based contextual regularization model embedded into a matrix factorization framework, which leverages the social, textual, and behavioral contexts across users, in order to overcome identified challenges.

- The fourth research contribution of this dissertation research is to scale up the proposed tensor learning framework to be capable of handling real large-scale datasets that are too big to fit in the main memory of a single machine. Particularly, we propose a novel distributed tensor completion algorithm with the trace-based regularization of the auxiliary information based on ADMM under the proposed tensor learning framework, which is designed to scale up to real large-scale tensors (e.g., billions of entries) by efficiently computing auxiliary variables, minimizing intermediate data, and reducing the workload of updating new tensors.

DEDICATION

To my father and mother.

ACKNOWLEDGMENTS

This dissertation would have never been done without a few great people who always supported, inspired and influenced me as well as my work. First and foremost, I would like to express all my gratitude to my advisor James Caverlee who is the best mentor, advisor and friend with his mentorship, guidance, a steady stream of supports and encouragement. One of my best choice I have ever made in my life is to choose him as my advisor for my PhD studies. I learned what the research is, how to do research and how to become a better man from him. His profession, kindness, and generousness are deeply engraved in my mind. I could spend many pages in expressing my gratitudes to Cav, but due to the limited space, I will just say that he is my role model and tirelessly shaped me into a better researcher and man.

I would also like to thank the rest of my dissertation committee, Frank Shipman, Xia Hu and Jianhuang Huang for their continuous advice and support during my PhD. Xia's advice help and motivated me to successfully complete my dissertation. The discussion with Jianhua also inspired me to develop the tensor learning model. I would especially like to thank Frank for his amazing feedback for my dissertation. Besides of activities at Texas A&M University, I was fortunate to have opportunities to work with great people in the industry. I am very grateful to Guru Shivaswamy who offered me my first industry internship at eBay, and is a tremendous mentor giving me absolute freedom and thought-provoking hours of discussions, Lili Yuan who always supported and help me during my internship at eBay, and Jianwu Xu who is my mentor during the internship at NEC Labs America and made my summer of 2016 extremely wonderful and enjoyable. In addition, I also would like to thank Kai Zhang, Bo Zong, Zhengzhang Chen, Wei Cheng, Biplob Debnath, and Hui Zhang for their unselfish help and advice on my internship at NEC Labs

America. I also very enjoyed the discussions with my fellow interns, Ying Zou, Hengtong Zhang, and Fei Wu. Furthermore, I am very thankful to my former roommates of the past years, Chaoyi Gu, Fan Yin and Yi Wang who are my truly friends and people who I can always bounce ideas off, and my friends Nan Zhang, Gang Yang, Jianchao Ge, Gang Wang, Naibao Zhao who all made my journey full of unforgettable moments.

Many thanks to my awesome lab-mates at Infolab for their tremendous support and helpful discussions. I enjoyed all office discussions, jokes, and exchange of ideas. We share the wonderful moments of paper acceptance as well as sad moments of being rejected. Former lab-mates Zhiyuan Cheng, Kyumin Lee, Krishna Kamath and Jeff McGee welcomed me to the lab and help me understand how to be a good graduate student and what I should prepare for the research. My buddies Cheng Cao, Wei Niu and Haokai Lu had a tremendous impact on my PhD. We worked together through these years and had wonderful discussions and collaboration on interesting projects. I would also like to thank my other lab-mates Vandana Bachini, Himanshu Barthwal, Zhijiao Liu, Yuan Liang, Amir Fayazi, Prithivi Tamilarasan, Majid Alfifi, Parisa Kaghazgaran, Xing Zhao, Yin Zhang and Habeeb Hooshmand.

I am deeply indebted to my parents, Yutong Ge, and Qiang Huang, for their love and endless support. They are my role models who were always encouraging me and having faith in me no matter I am in ups and downs in my life. My parents valued good education more than anything else and made sure we had the best environment for it growing up. Things I saw and learned from them encouraged me to be an honest man, an generous person and an excellent researchers. My family is my strongest backing on whatever I confronted and made, which guided me to keep moving forward. The rest of my family are equally amazing. I love you all. Without the endless love and support from them, none of this work would be possible.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professor James Caverlee, Professor Frank Shipman and Professor Xia Hu of the Department of Computer Science and Engineering and Professor Jianhua Huang of the Department of Statistics. All work conducted for the dissertation was completed by the student independently.

Funding Sources

Graduate study was supported by NSF grants IIS-1149383, IIS-1657196 and AFOSR grant FA9550-15-1-0149 as well as a Google Research Award.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

1.1 Motivation

The Web and social media are data rich. With the prolific sharing of videos, images, blogs, and tweets, and the enormous amount of transactional behaviors like retweets, reviews, ratings, and purchase behavior, we increasingly have fine-grained access to the "who", "where", "when" and "what" of real-time human dynamics. This unprecedented data explosion promises new opportunities to uncover models and theories of user behavior, mobility, and information sharing, as well as lead to new algorithms and frameworks.

1.1.1 Multi-Aspect Data

These dynamics usually come in *multiple aspects*. For instance, purchase behaviors of users at online stores can be expressed in millions of *user-item-date* triplets, like *Adam* bought *headphones* on *Monday*. In this case, the user, item and date are the three aspects of the data. Together, these multiple aspects can help uncover important underlying phenomena. For example, the triplets in this case can uncover which items may be popular at different times, and which users are more likely to purchase which items. Similar multiple aspect data can be found in social networks such as Twitter, Snapchat and Facebook: for instance, we can model user-user-topic triplets to capture the kinds of posts that two users tend to share with one another. Alternatively, two friends can chat on both Snapchat and Facebook, and retweet each other on Twitter. All these multi-aspect datasets are able to help better understand the social interactions of the underlying network.

Indeed, discovery, modeling and analyzing multi-aspect data can benefit many domains beyond social media, including analysis of knowledge bases [1], chemometrics [2], signal processing [3, 4], computer vision [5, 6], climate dynamics [7, 8], neurology [9, 10], network traffic [11], Web graphs [12] and more. By gleaning hidden or latent features, the

multi-aspect data in these various domains can lead to new insights into applications such as anomaly detection [13, 14], recommender systems [15, 16] and correlation analysis [17], among many others.

1.1.2 The Missing Data Problem

Nevertheless, the raw multi-aspect datasets are often incomplete owing to various unpredictable or unavoidable reasons. The raw data revealing dynamics on social media are often restricted to proprietary data warehouses (e.g., requiring privileged access to Instagram's backend photo serving services), and so researchers and practitioners typically must rely on sampling-based methods to build models and conduct analysis. Of course, this sampling faces its own challenges – including API limitations and data sampling policies that can lead to an incomplete (and often biased) perspective on the underlying dynamics. Moreover, changes to data access policies can lead to additional challenges – as demonstrated by Twitter's closing of their Firehose API in April 2015. Additionally, even a robust data sampling approach can still face errors due to missing data and errors in the data collection process. This missing data raises serious concerns such as significant bias on the estimation of structural properties of the network [18] and the properties of information cascades in a social network [19].

Already, we have witnessed considerable compelling studies on filling the missing or unobserved entries based upon partially observed data by adopting various techniques including multivariate interpolation [20], spectrum analysis [21], fuzzy neural network [22], and matrix factorization [23, 24]. For example, Candès at al. [25] recover a data matrix from a sampling of its entries via the convex optimization and find the matrix with minimum nuclear norm that fits the data. These methods have shown good success, but typically assume a simple inter-dependence among variables of interest (e.g., user-item, post-location), resulting in a challenge to handling correlations and complex inter-

dependencies among these different factors.

1.1.3 Tensors and Tensor Learning

In contrast, tensors, as a generalization of multi-way arrays, provide a compelling way to capture inter-dependencies among entities (i.e. more than two dimensions). Therefore, it is natural to represent the multi-aspect datasets as tensors. Indeed, in recent years, tensor factorization models have been widely studied and applied in many fields due to its essential nature well-suited for multi-aspect data analysis. Tensor completion has been a popular technique for estimating missing values or information in multi-aspect data with the assumption that the data of interest follows a *low-rank* model, which is accomplished by finding a low-rank tensor model for the observed data and leveraging such low-rank model to estimate the unobserved data. As we can see, it has been extensively studied and employed in applications such as tag recommendation [15], user group detection [26], link prediction [27] and more. For instance, computational phenotyping in electronic health records is further improved by tensor completion for discovering more meaningful and distinct phenotypes [28]. The TFMAP [29] has shown how to leverage tensor completion to boost the mobile app recommendation by treating the problem of predicting how a user will rate an app under some context as the estimation of missing values in a tensor. Researchers from Sandia National Laboratories [30] have tackled the missing data arisen in computer network traffic via tensor completion in order to avoid situation that important datasets are discarded or improperly analyzed due to missing data. Furthermore, some researchers [31, 32] incorporate contextual information into the completion problem for achieving better recovery accuracy instead of only utilizing the observed data. Some others [33, 34] scale up the tensor completion algorithms to handle real large-scale datasets.

1.2 Challenges

In the previous section, we described the problem of missing values or information in the multi-aspect datasets on social media and how to tackle this problem with the state-of-the-art methods. To satisfy the rapid growth of data in both its velocity and volume in the real-world, there are significant research gaps towards effectively and efficiently handling missing information in the large-scale multi-aspect datasets with rich contextual information. We now identify "**4S**" research challenges including **sparsity**, **similarity**, **specificity** and **scalability**, associated with recovering the missing information on social media under the background of large-scale data analytics as follows:

- **Sparsity.** With the consistently increasing volume of the multi-aspect data, it may have very large dimensions (e.g., over 1 million users on Netflix). As a result, problems with high dimensionality lead to the fact that limited observed data points become extremely *sparse* as the dimensionality increase. In particular, users purchase limited items and provide few feedback on them. The collected data would be very sparse comparing with a large number of items online (over millions). The sparsity is able to degrade the performance of models with considerable bias and misleading inference [35]. Given a very sparse multi-aspect data, how can we recover its missing information while perpetuating the systematical performance of models?

- **Similarity.** Massive datasets on social media usually have multiple dimensions (aspects) with the high sparsity that leads to degrading performance of completion algorithms, especially for those with the low-rank assumption. Besides of the relational information among objects in the multi-aspect datasets, we usually have rich contextual information for each dimension, e.g., profile information of customers at Amazon and attributes of products purchased by customers, which are usually complex. All these contextual information are able to describe the similarity of entities among

dimensions, and potentially exploited to improve the accuracy of recovering missing information in the sparse large-scale datasets. How can we model these contextual information? How can we incorporate them into the completion algorithms?

- **Specificity.** Most existing completion algorithms focus on recovering the missing information from one specific dataset or with specified contextual information, which lack of generalization towards a variety of applications or different types of side information, and limit their applicability to real-world big data problems. How can we develop a generalized framework for the recovery of missing information with the integration of all potential pairwise relations, which can be utilized in various applications?

- **Scalability.** The scale of modern multi-aspect data poses the challenge for existing completion algorithms, most of which become entirely infeasible for the Gigabyte and Terabyte data sets that are fast becoming common and too large to fit in the main memory of a single machine. The need for scalable completion algorithms is over increasing, and there is a huge gap that needs to be filled. Can we handle the issue of scalability in order to appropriately recover missing information in large-scale datasets? How can we scale the completion algorithms to massive datasets while pertaining their performance with respect to estimate missing information?

1.3 Contributions

This dissertation seeks to integrate and leverage rich contextual information from social networks to develop new scalable and interpretable algorithms for recovering missing values in big multi-aspect data by bridging Tensor Completion and Data Science for real-world applications. To combat identified challenges, we propose scalable tensor learning algorithms for recovering missing spatio-temporal dynamics of online memes, expanding

them to applications that are personalized expert recommendation and learning user topical profiles, and improving the scalability of tensor learning algorithms in social systems. Concretely, this dissertation makes the following four unique contributions:

- **Overcoming Missing Spatio-Temporal Dynamics**: First, in order to tackle the problem of missing spatio-temporal dynamics of online memes, this dissertation proposes a scalable low-rank tensor learning framework in the presence of incomplete information. Concretely, we formally define the problem of recovering the spatio-temporal dynamics of online memes and solve this problem by proposing a novel tensor-based factorization approach based on the alternative direction method of multipliers (ADMM) [36]. The core insight of the proposed method is to carefully take into account the latent relationships derived from contextual information among locations, memes, and times. We evaluate the performance of our tensor-based framework over both synthetic and Twitter datasets. Through extensive experimental study, we find that our proposed framework achieves an significant improvement on recovering missing spatio-temporal dynamics compared to state-of-the-art alternatives, while achieving significantly greater efficiency.

- **Recommending Personalized Experts**: Second, this dissertation evaluates the generalization of the proposed tensor learning framework and applies it to the recommendation problem that is one of typical missing information problems on social media. Specifically, we present a novel tensor-based approach via the extension of the proposed tensor learning framework to personalized expert recommendation with the integration of both the latent relationships between homogeneous entities (e.g., users and users, experts and experts) and the relationships between heterogeneous entities (e.g., users and experts, topics and experts) from the geo-spatial, topical, and social contexts. Through an experimental study, we find that the pro-

posed model can significantly improve the quality of the recommendation compared to several state-of-the-art baseline methods.

- **Learning User Topical Profiles**: Third, this dissertation extends the proposed tensor learning framework to the user topical profiling problem. Concretely, we propose a tensor-based contextual regularization model embedded into a matrix factorization framework, which leverages the social, textual, and behavioral context across users, in order to overcome identified challenges. We model all contextual signals into one tensor via calculating the user similarity in each of social, textual and behavioral contexts. After applying the tensor decomposition, the latent representation of users can be learned, which is further embedded into a matrix factorization framework in order to learn unknown user topical profiles.

- **A Distributed Scalable Approach to Tensor Learning**: Fourth, this dissertation enables the proposed tensor learning framework to handle real large-scale datasets that are too big to fit in the main memory of a single machine. We propose a novel distributed tensor completion algorithm with regularized trace of the auxiliary information based on ADMM under the proposed tensor learning framework, which is designed to scale up to real large-scale tensors by efficiently computing auxiliary variables, minimizing intermediate data, and reducing the workload of updating new tensors. We successfully tackles the high computational costs and minimizes the intermediate data and find that the proposed distributed scalable algorithm is capable of handling up to 100x larger tensors than existing methods with much faster convergence rate, shows better linearity on the machine scalability, and achieves better accuracy in the applications such as recommender systems and link prediction.

1.4 Dissertation Overview

The remainder of this dissertation is organized as follows:

- **Section 2: Background.** In this chapter, we provide a brief background on tensors containing necessary tensor operations, basic definitions, and notation followed by the tensor decomposition and completion problems throughout this dissertation.

- **Section 3: Related Work.** In this chapter, we discuss related work, specifically tensor factorization, recovering missing information on social media over scenarios such as estimating spatio-temporal dynamics of online memes, recommending personal experts and learning user topical profiles, and scalable tensor algorithms.

- **Section 4: Recovering Missing Dynamics: A Tensor-based Approach with Auxiliary Information.** In this chapter, we propose AIRCP, a novel tensor learning algorithm that leverages auxiliary information among locations, hashtags for recovering the spatio-temporal dynamics of online memes in the presence of incomplete information. Through experimental evaluation on both synthetic and real-world Twitter hashtag data, we see that the proposed framework outperforms alternative stateof-the-art methods with an average improvement of over 27%.

- **Section 5: TAPER: A Contextual Tensor-Based Approach for Personalized Expert Recommendation.** In this chapter, we extend the proposed tensor learning algorithm to propose a tensor-based personalized expert recommendation framework that integrates these factors for revealing latent connections between homogeneous entities (e.g., users and users) and between heterogeneous entities (e.g., users and experts). Through extensive experiments over geo-tagged Twitter data, we find that the proposed framework can improve the quality of recommendation by over 30% in both precision and recall compared to the state-of-the-art.

- **Section 6: T-CRM: Learning User Topical Profiles.** In this chapter, we develop a tensor-based unified model for learning user topical profiles by simultaneously tak-

ing into account pairwise relations among multiple contextual information. Through extensive experiments, we find that the proposed model is capable of learning high-quality user topical profiles, and leads to a 10$\sim$15% improvement in precision and mean average error versus a cross- triadic factorization state-of-the-art baseline.

- **Section 7: DISTENC: A Distributed Scalable Tensor Completion Algorithm.** In this chapter, we propose DISTENC, a distributed large-scale tensor completion algorithm with regularized trace of the auxiliary information based on ADMM running on the Spark framework. Through evaluating for both synthetic and real-world large-scale datasets, experimental results demonstrate the superiority of DISTENC with up to 10$\sim$1000$\times$ larger scalability than existing methods, much faster convergence rates, and better linearity on the machine scalability. Moreover, it achieves up to an average improvement of 23.5% in the accuracy in terms of the applications such as recommender systems and link prediction.

- **Section 8: Conclusions and Future Directions** We conclude our dissertation with a summary of contributions, and discuss potential research extensions to the results presented here.

2. BACKGROUND

2.1 Introduction

In this chapter, we provide a brief background on tensors including tensor operations, basic definitions, and notation. We follow this background with an introduction to the tensor decomposition and completion problems at the core of this dissertation.[*] Recall that tensors are generalizations of multi-array arrays and have proven to be a very powerful tool in a variety of applications that inherently produce multi-dimensional (multi-aspect) datasets. Table 2.1 lists the definition of symbols applied throughout the entire dissertation. Note that we also introduce some specialized notation in some of the following chapters.

2.2 Tensor

Definition 2.1.1 (Tensor). A tensor is a multi-way array, whose dimension is called *mode* or *order*. An Nth-order tensor is an N-mode array, denoted as $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. The number of non-zero elements of a tensor $\mathcal{X}$ is denoted as $nnz(\mathcal{X})$.

Definition 2.1.2 (Kronecker Product). Given two matrices $\mathbf{A} \in \mathbb{R}^{I \times J}$ and $\mathbf{B} \in \mathbb{R}^{K \times L}$, their Kronecker product $\mathbf{A} \otimes \mathbf{B}$ generates a matrix of size $IK \times JL$ defined as:

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{11}\mathbf{B} & \cdots & a_{1J}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}\mathbf{B} & a_{11}\mathbf{B} & \cdots & a_{IJ}\mathbf{B} \end{bmatrix}, \tag{2.1}$$

Definition 2.1.3 (Khatri-Rao Product). It is a column-wise Kronecker product, denoted as $\mathbf{A} \odot \mathbf{B}$ where both $\mathbf{A} \in \mathbb{R}^{I \times R}$ and $\mathbf{B} \in \mathbb{R}^{K \times R}$ have the same number of columns. Their

[*]Note that tensors introduced here are not the same as tensors in physics and engineering; those are generally referred to as tensor fields in mathematics.

Table 2.1: Symbols and Operations.

Symbols	Definitions
$\mathbb{R}$	the set of real numbers
$\mathcal{X}$	tensor (Euler script letter)
$\mathbf{X}_{(n)}$	n-mode matricization of a tensor $\mathcal{X}$
$\mathbf{X}$	matrix (uppercase bold letter)
$\mathbf{x}$	column vector (lowercase bold letter)
x	scalar (lowercase letter)
N	order of a tensor (number of modes)
$[\![\cdot]\!]$	Kruskal operator
$\otimes$	Kronecker product
$\odot$	Khatri-Rao product
$*$	Hadamard product
$\circ$	outer product
$< *, * >$	inner product of matrices
$\times_n$	n-mode tensor-matrix product
$\mathbf{X}^{-1}$	inverse of a matrix $\mathbf{X}$
$\mathbf{X}^{\dagger}$	Moore-Penrose pseudo-inverse of a matrix $\mathbf{X}$
$\mathbf{x}(i)$	ith entry of a vector $\mathbf{x}$
$\mathbf{X}(:, i)$	the entire ith column of a matrix $\mathbf{X}$ (same as tensor)
$\mathbf{X}(i, :)$	the entire ith row of a matrix $\mathbf{X}$ (same as tensor)
$\|\mathbf{X}\|_F^2$	Frobenius norm of $\mathbf{X}$
$nnz(\mathcal{X})$	number of non-zero elements in $\mathcal{X}$

Khatri-Rao product produces a matrix of size $IK \times R$ defined as:

$$\mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1, \cdots, \mathbf{a}_R \otimes \mathbf{b}_R]. \tag{2.2}$$

Definition 2.1.4 (Hadamard Product). Given two matrices $\mathbf{A}$ and $\mathbf{B}$ with the same size

$I \times J$, their Hadamard product $\mathbf{A} \times \mathbf{B}$ is the element-wise matrix product, defined as:

$$\mathbf{A} \times \mathbf{B} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} & \cdots & a_{1J}b_{1J} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}b_{I1} & a_{I2}b_{I2} & \cdots & a_{IJ}b_{IJ} \end{bmatrix}. \tag{2.3}$$

Definition 2.1.5 (Tensor Matricization). Tensor matricization is to unfold a tensor into a matrix format with a predefined sequence of mode order. The *n*-mode matricization of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ is denoted as $\mathcal{X}_{(n)} \in \mathbb{R}^{I_n \times (\prod_{k \neq n} I_k)}$. The order of the other modes except mode n can be arranged randomly to construct the column of $\mathcal{X}_{(n)}$. For instance, we matricize a 3-order tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ in the following three ways that are $\mathbf{X}_{(1)}$ of size $(I \times JK)$, $\mathbf{X}_{(2)}$ of size $(J \times IK)$ and $\mathbf{X}_{(3)}$ of size $(K \times IJ)$. These tensor matricizations are mapped in the following way:

$$\mathcal{X}(i, j, k) \rightarrow \mathbf{X}_{(1)}(i, j + (k-1)J) \tag{2.4}$$

$$\mathcal{X}(i, j, k) \rightarrow \mathbf{X}_{(2)}(j, i + (k-1)I) \tag{2.5}$$

$$\mathcal{X}(i, j, k) \rightarrow \mathbf{X}_{(3)}(k, i + (j-1)I) \tag{2.6}$$

Definition 2.1.5 (n-mode Tensor-Matrix Product). Given an Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ and a matrix $\mathbf{A} \in \mathbb{R}^{I_n \times J}$, their multiplication on its nth-mode is represented as $\mathcal{Y} = \mathcal{X} \times_n \mathbf{A}$ and is of size $I_1 \times \ldots I_{n-1} \times J \times I_{n+1} \ldots \times I_N$. The element-wise result is demonstrated as:

$$\mathcal{Y}(i_1, \ldots, i_{n-1}, j, i_{n+1}, \ldots, i_N) = \sum_{k=1}^{I_n} \mathcal{X}(i_1, \ldots, i_{n-1}, k, i_{n+1}, \ldots, i_N)\mathbf{A}(k, j). \tag{2.7}$$

2.3 Tensor Decomposition

Tensor decomposition has been widely used in tensor learning to discover latent relations in the multi-aspect data. There are two widely used low-rank decompositions of tensors, the CANDECOMP/PARAFAC (CP) and the Tucker decompositions. The CP tensor decomposition is the most popular tensor decomposition method, independently proposed by [37, 38, 39]. Given an N-mode tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_N}$, the CP tensor decomposition method decomposes a tensor $\mathcal{X}$ into a sum of $R \ll min(I_1, \ldots, I_N)$ rank-one tensors as:

$$\mathcal{X} \approx [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!] = \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \ldots \circ \mathbf{a}_r^{(N)}, \tag{2.8}$$

where $\mathbf{a}_r^{(n)}$ is the rth column of the factor matrix $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R}$ at mode n, and R is the rank of a tensor $\mathcal{X}$. Each rank-one component in the CP tensor decomposition can be interpreted as a latent "concept" (cluster) in the multi-aspect data, which can be treated as an explanatory model. In addition, the CP tensor decomposition also holds very good theoretical properties including uniqueness of solvers under very mild conditions. The definition of CP tensor decomposition is given below.

Definition 2.1.5 (CP Tensor Decomposition). Given an N-mode tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_N}$ and rank R, the CP tensor decomposition solves the optimization problem:

$$\underset{\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}, \mathcal{X}}{\text{minimize}} \quad \frac{1}{2} \|\mathcal{X} - \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(N)}\|_F^2,$$

where $\mathbf{A}^{(n)} \in \mathbb{R}^{(I_n \times R)}$ is the factor matrix at mode-n of a tensor $\mathcal{X}$.

2.4 Tensor Completion

Tensor completion is extensively applied in tensor mining to fill the missing elements with partially observed tensors. *Low rank* is often a necessary hypothesis to re-

strict the degree of freedoms of the missing entries being intractable. Hence, we only focus on low-rank tensor completion (LRTC) problem in this dissertation. We introduce CANDECOMP/PARAFAC(CP)-based tensor completion. The CP tensor decomposition proposed by Hitchcock [37] is one of the most used tensor factorization models, which decomposes a tensor into a sum of rank-one tensors. It can be employed for the tensor factorization to the tensor completion problem. Before being actively researched in recent years, the LRTC problem is usually considered as a byproduct of the tensor decomposition problem with missing values. Given an N^{th}-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with the rank $R \ll min(I_1, \ldots, I_N)$, the CP-based tensor completion solves:

$$\begin{aligned} \underset{\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)},\mathcal{X}}{\text{minimize}} \quad & \frac{1}{2}\|\mathcal{X} - [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!]\|_F^2 + \frac{\lambda}{2}\sum_{n=1}^{N}\|\mathbf{A}^{(n)}\|_F^2 \\ \text{subject to} \quad & \boldsymbol{\Omega} * \mathcal{X} = \mathcal{T}, \mathbf{A}^{(n)} \geq 0, n = 1, 2, 3., \end{aligned}$$

where $\mathcal{T}$ denotes the partial observations, $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R}$ are the factor matrices, and $\boldsymbol{\Omega}$ is a non-negative weight tensor with the same size as $\mathcal{X}$:

$$\boldsymbol{\Omega}(i_1, \ldots, i_n, \ldots, i_N) = \begin{cases} 1 & \text{if } \mathcal{X}(i_1, \ldots, i_n, \ldots, i_N) \text{ is observed,} \\ 0 & \text{if } \mathcal{X}(i_1, \ldots, i_n, \ldots, i_N) \text{ is unobserved.} \end{cases}$$

In practice, tensor completion is not only capable of recovering the missing information occurred in the original multi-aspect data well, but also providing the interpretation of latent factors. The CP-based tensor completion guarantees the uniqueness of latent factors that generated the variations in the data.

3. RELATED WORK

In this chapter, we summarize the state-of-the-art methods with respect to recovering missing values on social media from tensor factorization, a variety of applications and scalable tensor factorization algorithms.

3.1 Tensor Factorization

We begin in his section with a review of related work on tensor decomposition and tensor completion, respectively.

3.1.1 Tensor Decomposition

Compared to matrix factorization methods – which focus on two-way data, not multi-dimensional data – tensors, as a generalization of matrices, can naturally model higher-order relationships among entities (i.e. more than two dimensions). Tensor factorization models have been studied and applied in many fields due to their strong power on multi-dimensional data analysis. There are two widely used low-rank decompositions of tensors, the CANDECOMP/PARAFAC (CP) and the Tucker decompositions [40]. The most common methods used to factorize tensors include alternating least square (ALS) [6, 41, 42, 43], stochastic gradient descent (SGD) [44, 45] and coordinate descent (CDD) [33, 46].

3.1.2 Tensor Completion

Tensor completion is used to estimate missing values in tensors based on their low-rank approximations, which has been extensively studied and employed in applications such as recommendations [16, 15], user group detection [26], and link prediction [27]. Most these methods only focus on the sampled data when performing tensor completion without considering auxiliary information with which tensors usually come. These aux-

iliary information help us have a better performance in tensor completion [32]. Though several researchers incorporate contextual (external) information into the matrix factorization problem [31], few studies explore the tensor completion problem with auxiliary information. Technically, it is challenging to embed auxiliary information into a factorization model, especially with many heterogeneous contexts. Narita et al. [32] integrated auxiliary information into tensor decomposition methods, resulting in better performance compared with ordinary tensor decomposition methods. Nevertheless, they primarily focus on general tensor decomposition rather than tensor completion. However, these models usually face some efficiency challenges since [32] requires solving the Sylvester equation with a high cost several times in each of iterations, making them infeasible for large-scale applications. Though several researchers incorporate auxiliary (external) information into the matrix factorization problem [31, 47], few studies explore the tensor completion problem with auxiliary information. Technically, it is challenging to embed auxiliary information into a factorization model, especially with many heterogeneous contexts. Narita et al. [32] integrated side information into tensor decomposition methods, resulting in better performance compared with ordinary tensor decomposition methods. Nevertheless, they primarily focus on general tensor decomposition with auxiliary information, but not tensor completion. In this dissertation, we integrate rich contextual information among objects into a tensor completion framework and scale it up to be capable of handling real large-scale datasets in a variety of applications.

3.2 Estimating Missing Spatio-Temporal Dynamics of Online Memes

In this section, we review related work on spatio-temporal dynamics of online memes and corresponding methods on estimating their missing spatio-temporal dynamics.

3.2.1 Spatio-Temporal Dynamics of Online Memes

The increasingly mobile aspects of social media services like Instagram, Facebook, and Twitter have led to a number of studies on geo-spatial characteristics of users and information sharing. For example, researchers have built models of geo-spatial properties to infer geographic information from tweets, such as spatial modeling to geolocate objects [48] and predicting user locations [49]. Other researchers have analyzed the geo-spatial properties of online memes on Facebook [50] and on YouTube based on propagation patterns [51]. On the other hand, much effort has focused on the temporal properties of online memes. Yang et al. [52] studied temporal patterns of online content including Twitter hashtags and online phrases. Matsubara et al. [53] explored temporal patterns of online information diffusion. Other researchers have focused on both spatial and temporal properties of online memes, like [54].

3.2.2 Estimating Missing Spatio-Temporal Data

Toward recovering missing spatio-temporal data, there have been many proposed methods adopting techniques like multivariate interpolation [20], spectrum analysis [21], and matrix factorization [23, 55]. These methods have shown good success, but typically assume a simple inter-dependence among variables of interest (e.g., memes), space, and time, resulting in a challenge to handling correlations (and complex inter-dependencies) among these different factors. Recently, researchers employ tensor-based approaches to tackle the problem of recovering missing spatio-temporal data. Bahadori et al. [56] proposed a unified low-rank tensor learning framework on spatio-temporal data, under which either spatial or temporal information can be modeled, respectively. Yet, how to leverage both spatial and temporal information simultaneously was not investigated in their study. Zhou et al. [57] developed a Tucker-based tensor model called the spatio-temporal tensor completion to infer missing Internet traffic data by integrating spatio-temporal constraint

information as within-mode regularization. However, these models usually face some efficiency challenges since [32] requires solving the Sylvester equation with a high cost several times in each of iterations, and [57] strongly relies on solving large-scale least square problems, making them infeasible for large-scale applications. In contrast, scalable completion algorithms proposed in this dissertation seek to overcome these challenges by developing an efficient tensor-based method that integrates rich contextual information among objects simultaneously. In addition, the proposed approach not only inherits advantages of efficiency based on ADMM [58] and uniqueness of solutions enhancing the robustness, but also leads to better recovery by incorporating this contextual information.

3.3 Recommendation on Experts

In this section, we review related work on recommending experts for users which can be treated as estimating unobserved ratings of experts who have not been checked by users.

3.3.1 Finding Experts

Expert finding has been widely studied for many years. For instance, Weng et al. [59] proposed a PageRank-based approach to identify topic experts by applying both topical similarity between users and social link structure. Ghosh et al. [60] proposed the Cognos system to find topic experts by relying on Twitter lists. Zhang et al. [61] identified top experts in a Java forum by applying link analysis approaches such as PageRank and HITS. Hu et al. [62] proposed a more personalized recommendation by considering network semantic information, in addition to network topological measures for expert recommendation. Of course, there are many other efforts, including [63, 64, 65].

3.3.2 Systems towards Recommending Experts

One promising approach for recommenders is to cast the problem as a matrix factorization problem, in which user preferences may be projected into a lower dimensional

embedding space [66, 67, 68, 69]. In recent years, tensor factorization models are becoming popular and successfully applied in the recommendation. Karatzoglou et al. [70] studied multi-dimensional recommendations by leveraging contextual information to build a User-Item-Context tensor model. Hidasi and Tikk [71] developed an Alternating Least Squares (ALS) based tensor factorization approach for context-aware recommendations. Bhargava et al. [72] applied a tensor factorization-based approach to provide collaborative recommendations for points of interest (POI) involving multi-dimensions such as locations, activities and time. Similarly, Lu et al. [67] introduced a matrix-factorization approach for personalized expert recommendation, but restricted to two-dimensional data (e.g., a user-expert matrix) and only considering user's geo-spatial preferences on experts. In contrast, the proposed method described in this dissertation is the first to investigate tensor factorization for personalized expert recommendation by integrating user's preferences on experts from geo-topical-social contexts as well as valuable relationships between users, experts and topics, and to explore the impact of these contexts on such an approach.

3.4 Learning User Topical Profiles

In this section, we review related work on finding user interests and expertise, leveraging contexts and factorization models.

3.4.1 Finding User Interests and Expertise

Finding user interests and expertise has numerous applications, and one of the most popular tasks is personalized recommendation. Considerable research [73, 74, 75, 76, 77, 78, 79, 80] has assumed a user's interests or expertise as *implicit* personal preferences for building recommender systems in different domains, such as web search [77, 78], web content [80], rating systems [81, 76], and social media [74, 82, 75, 79].

For social media research, the latent factor model is a state-of-the-art method for user recommendation. Interpreting the latent factors as topics, approaches based on such a

model usually avoid explicitly identifying user interests but instead integrate the factors into a recommendation task. For example, Hong et al. applied matrix factorization on both users and tweets and focused on recommending user's retweeting behavior [82]. Similarly, Jiang et al. presented a probabilistic matrix factorization method to recommend whether a user adopts an item on a social network [75]. Zhong et al. collected user's webpage views to build a matrix factorization profile for web content recommendation [80].

3.4.2 Leveraging Contexts

A sequence of research has focused on using contexts to learn user interests. One of the most traditional information is textual context, upon which many works have built generative models like PLSA and LDA [83, 84, 59, 85]. Another popular context is social information (often via friendships)[75, 86, 87], with the natural assumption that friends tend to have similar profiles. In addition, behavioral context has become a newer factor; for example, Guy et al. used user's tagging behavior as evidence for content recommendation in social media [73]. Lappas et al. considered *user endorsement* as the behavioral context [88]. In [79], Zhao et al. focused on the behaviors of commenting, "+1", and "like" on Google+. Typically, these contexts have been treated separately. URL sharing behavior for topical profiles has received some attention in social media research. Previous work looked into why and what content people share via URLs on social media [89, 90]. Some work has also mentioned the role of URL sharing in social spamming [91].

3.4.3 Factorization Models

Technically, it is challenging to embed context into a factorization model, especially with many heterogeneous contexts. A handful of studies have adopted a regularization model [92, 93, 76] for personalized recommendation, though typically focusing on only one context. In [81], assuming contexts are not independent, latent spaces are learned separately for each context. Tensor-based factorization methods [94, 95] have been used in

many applications such as behavior modeling, healthcare, and urban planning [96, 28, 97]. A more comprehensive survey of tensor factorization and its applications can be found in [40].

Several studies have focused on heterogeneous domains or entities, instead of contexts. Yu et al. put multiple types of entities into a heterogeneous network and used a Bayesian ranking process to estimate user preferences [98]. Similarly, Hu et al. looked into a traditional user-item recommendation problem, presenting a factorization model across heterogeneous items. However, the network will quickly grow when users and items increase. Singh and Gordon proposed a framework to learn different types of relations, where they iteratively do matrix factorization between all pairs of domains [99]. Hu et al. [100] adopted the existing PARAFAC2 factorization algorithm on a tensor model, which is obtained by combining user ratings of different merchandises like book, music, and movie. Zhong et al. [80] directly applies a matrix factorization model on Web users and their clicked content items.

In contrast, we focus on learning user topical profiles rather than rating recommendation, and we are interested in leveraging heterogeneous contexts. We propose a generalized factorization model in which we simultaneously consider contexts via regularization. We further take care of context-wise relations by integrating both tensor decomposition and regularization together.

3.5 Scalable Tensor Algorithms

In this section, we review related works on scalable tensor algorithms, and distributed computing frameworks.

3.5.1 Scalable Tensor Factorization

We witness considerable efforts on developing scalable algorithms for tensor factorization, most of which focus on solving the intermediate data explosion problem. Concretely,

pioneers Bader and Kolda [42] develop efficient algorithms for sparse tensor decomposition by avoiding the materialization of very large, unnecessary intermediate Khatri-Rao products. Kolda and Sun [14] continuously work on the specific tensor decomposition method Tucker for sparse data and solve the intermediate explosion problem by calculating the tensor-matrix multiplication one slice or fiber at a time. An alternative approach, DBN, is introduced in [101] where the authors use Relational Algebra to break down the tensor into smaller tensors, using relational decomposition, and thus achieving scalability. Kang et al. [102] first propose a scalable distributed algorithm GigaTensor under the MAPREDUCE framework for the specific tensor decomposition method PARAFAC by decoupling the Khatri-Rao product and calculate it distributively column by column. Jeon et al. [103] improves on GigaTensor and propose HaTen2 that is a general, unified framework for both Tucker and CP tensor decomposition. There has been other alternatives on solving the intermediate data explosion problem of the pure tensor composition [33, 104, 105].

On the other hand, some researchers put their focus on developing scalable, distributed algorithms for the tensor decomposition with additional side information that is usually represented in a matrix, e.g., a similarity matrix between experts. Papalexakis et al. [45] propose an efficient scalable framework to solve the coupled matrix-tensor factorization problem by leveraging the biased sampling to split the original large data into samples, running the common solver to samples and merging the results based on the common parts in each sample. Beutel et al. [44] propose FLEXIFACT, a MAPREDUCE algorithm to decompose matrix, tensor, and coupled matrix-tensor based on stochastic gradient descent. Jeon et al. [106] propose SCouT for scalable coupled matrix-tensor factorization. Shin et al. [46] propose two scalable tensor factorization algorithms SALS and CDTF based on subset alternating least square and coordinate descent, respectively. Livas et al. [107] develop a constrained tensor factorization framework based on ADMM. Smith et al. [34] optimize and evaluate three distributed tensor factorization algorithms based on ALS, SGD

and CDD, respectively, by extending SPLATT [108] that optimizes the memory usage.

3.5.2 Distributed Computing Frameworks

MAPREDUCE [109] is a distributed computing model for processing large-scale datasets that cannot be handled in a single machine, running in a massively parallel manner. MAPREDUCE has been the most popular distributed computing framework due to its advantages including automatic data distribution, fault tolerance, replication, massive scalability, and functional programming by which users only define two functions `map` and `reduce`. HADOOP [110] is an open-source version of MAPREDUCE. Because of its excellent scalability and ease of use, it has been successfully applied in many data mining applications [102, 111, 112, 113]. However, HADOOP is inefficient to execute iterative algorithms due to its intensive disk accesses [109]. Apache Spark [114] is an in-memory MAPREDUCE, which provides a high-level interface for users to build applications with respect to large-scale data computation. Spark allows to store intermediate data in memory and performs efficient memory-based operations without requiring data to be spilled to disk (effectively reducing the number of disk Input/Output operations). Therefore, Spark is capable of performing iterative algorithms very efficiently. Due to these advantages, Spark has been used in applications [104, 115, 116].

4. AIRCP: A TENSOR-BASED APPROACH FOR RECOVERING MISSING SPATIO-TEMPORAL DYNAMICS OF ONLINE MEMES*

In this chapter, we tackle the challenges of **sparsity** and **similarity** for the problem of missing spatio-temporal dynamics of online memes. The raw data revealing the spatio-temporal dynamics of online memes are often incomplete and error-prone due to many reasons such as API limitations, data sampling policies, and necessary privilege accesses. Thus, we propose and evaluate a scalable framework based on low-rank tensor learning to recover these unobserved spatio-temporal dynamics of online memes. Our framework can effectively and efficiently recover missing dynamics of online memes in the presence of incomplete information. The core insight of the proposed method is to carefully take into account the latent relationships among locations, memes, and times; these relationships can then be embedded into a tensor completion framework for uncovering the approximate complete data based only on partial sparse observations.

4.1 Introduction

Many large-scale mobile social media services have been growing rapidly, enabling millions of users to generate and share location-associated content on a massive scale. For instance, many mobile image sharing services such as Instagram allow users to attach their latitude-longitude coordinates to shared photographs; location sharing services such as Foursquare and Glimpse enable billions of "check-ins"; and Twitter users generate millions of geo-tagged tweets per day. In turn, these fine-grained spatio-temporal logs of user activities promise new research opportunities to uncover models of user behavior,

*Reprinted with permission from "Uncovering the Spatio-Temporal Dynamics of Memes in the Presence of Incomplete Information" by Hancheng Ge, James Caverlee, Nan Zhang and Anna Squicciarini, 2016. Proceedings of the 25th ACM International Conference on Information and Knowledge Management. Copyright 2016 by ACM. DOI: http://dx.doi.org/10.1145/2983323.2983782

mobility, and information sharing. Already, there have been efforts to improve location-based recommendations, targeted advertising, social media search, and event detection [48, 49, 54, 117, 118].

However, the data revealing these dynamics are often raw and incomplete due to various unpredictable or unavoidable reasons such as restriction to proprietary data warehouses, mal-operations, missing at random and API limitations. Since researchers and practitioners typically must rely on sampling-based methods to build spatio-temporal models of user behavior, these data sampling policies can lead to an biased perspective on the underlying dynamics. For instance, Morstatter et al. [119] found significant differences in the quality and composition of sampled Twitter data by comparing different sampling policies over the streaming API and Twitter's Firehose. Moreover, changes to data access policies can lead to additional challenges – as demonstrated by Twitter's closing of their Firehose API in April 2015. Additionally, even a robust data sampling approach can still face errors due to missing data and errors in the data collection process. This missing data raises serious concerns. For example, Kossinets [18] found that missing data in a social network can significantly impact the estimation of structural properties of the network. Similarly, Sadikov et al. [19] pointed out that incomplete data may lead to critically different properties of information cascades in a social network. As a result, models based on mobile social media traces may be of limited usefulness and generalizability in the presence of incomplete data traces.

Hence, in this chapter we explore new scalable methods for recovering the spatio-temporal dynamics of online memes – like shared images, hyperlinks, videos, or hashtags – in the presence of incomplete information. Concretely, we propose a novel tensor-based factorization approach to recover the spatio-temporal dynamics of memes. The core insight of the proposed method is to carefully take into account the latent relationships among locations, memes, and times; these relationships can then be embedded into a ten-

sor completion framework for uncovering the approximate complete data based only on partial observations. We explore how to model and integrate this auxiliary information – here, in the form of relationships among locations, memes, and times – and show how the underlying tensor completion can be efficiently solved compared to existing methods.

Table 4.1: Comparison between AirCP and the state-of-the-art.

	AIRCP	TFAI[32]	TNCP[26]	LRCO[120]
Model				
Tensor	✓	✓	✓	✓
Coupled Tensor-Matrix	✓	✓		
Obj. Function				
Tensor Completion	✓		✓	✓
CP	✓	✓	✓	
Tucker				✓
Auxiliary Info.				
Heterogeneous Info.	✓	✓		
Regularization				
Laplacian	✓	✓		
Tikhonov	✓			
Trace Norm			✓	
Opt. Method				
ADMM	✓		✓	✓
Alternating Least Square		✓		

Through this proposed spatio-temporal dynamics framework – called **AIRCP** that stands for **A**uxiliary **I**nformation **R**egularized **C**ANDECOMP/**P**ARAFAC completion. In Table 4.1, we provide an overview of the state of the art. In short, AIRCP reigns, combining capability of leveraging heterogenous information as well as time efficiency, which is more feasible towards "big data". We explore research questions like: Based on an inherently limited sample, can we recover the underlying distribution of memes at a particular location? And at a particular time? What impact does the amount of sampled data have

on the quality of this recovery? For example, can we build a high-quality model of meme spread with access to only 20% of actual data? Towards tackling these and related questions, in this manuscript we formally define the problem of recovering the spatio-temporal dynamics of online memes by leveraging the latent relationships among memes, locations, and times, and develop approaches for modeling these latent relationships. Furthermore, we propose a novel framework for recovering spatio-temporal dynamics by a CP-based tensor completion model with regularized trace of the auxiliary information from memes, locations, and times, as well as Tikhonov regularization, which is efficiently solved by an efficient algorithm based on ADMM using less computation time than existing methods. We empirically evaluate the proposed framework on both synthetic and real-world Twitter hashtag datasets and find that the proposed method achieves an average over 27% improvement in recovering missing hashtags versus state-of-the-art alternatives, while achieving significantly greater efficiency.

4.2 Problem Statement

We assume that there exists a set of geo-temporal tagged online memes H. A meme in this case could correspond to a shared image, a hyperlink, video, or hashtag, among many other possibilities. Each meme $h \in H$ can be expressed as a tuple (h, l, t) where l is the location in which the meme is posted and t is the time at which the meme was posted. Suppose we have N unique geo-temporal tagged memes, L locations, M different timestamps and occurrences of memes O. Let $o_{lt}^{h} \in O$ be the number of occurrences of a meme h in a location l at a timestamp t. We view the spatio-temporal dynamics of geo-temporal tagged memes as a tensor $\mathcal{X} \in \mathbb{R}^{N \times L \times M}$, in which $\mathcal{X}(i, j, k)$ represents the count of a meme h_i in a location l_j at a timestamp t_k. Due to sampling errors, corrupted data, or other external factors, we further assume that we only can observe parts of the complete dynamics $\mathcal{X}$; we denote this partially observed tensor as $\mathcal{T} \in \mathbb{R}^{N \times L \times M}$, in which some

elements are missing or unobservable.

4.2.1 Spatio-Temporal Dynamics Recovery Problem

Given a set of geo-temporal tagged memes H with only partial knowledge of their dynamics – denoted as the tensor $\mathcal{T}$ – our goal is to learn a model to recover the missing spatio-temporal dynamics of the unobserved memes. But what entries in the partial tensor $\mathcal{T}$ are actually missing? We investigate three common situations:

- *Scenario 1: Random Missing Observations.* This first scenario captures the straightforward case of random corruption or random data sampling errors in the dataset. We assume that some fraction of memes – that is, some of the meme, location, time counts (h_i, l_i, t_i) – are missing, and so our task is to estimate these missing counts based on the observations we do have.

- *Scenario 2: Missing Entire Memes at Some Locations.* The second scenario models the case when the data collected has some systematic errors; specifically, we assume that rather than random missing observations (as the scenario 1), there are some memes that are completely missing for some locations. For example, a data sampling strategy may target the top-k online wmemes at a location, so some locations will be missing memes outside of this top-k. Can we recover the missing counts for these lost memes?

- *Scenario 3: Missing Entire Locations.* The final scenario corresponds to the case of total data loss for some locations. For example, a data sampling strategy may target some locations exclusively, but miss others entirely. Can we recover what memes did occur in those missing locations?

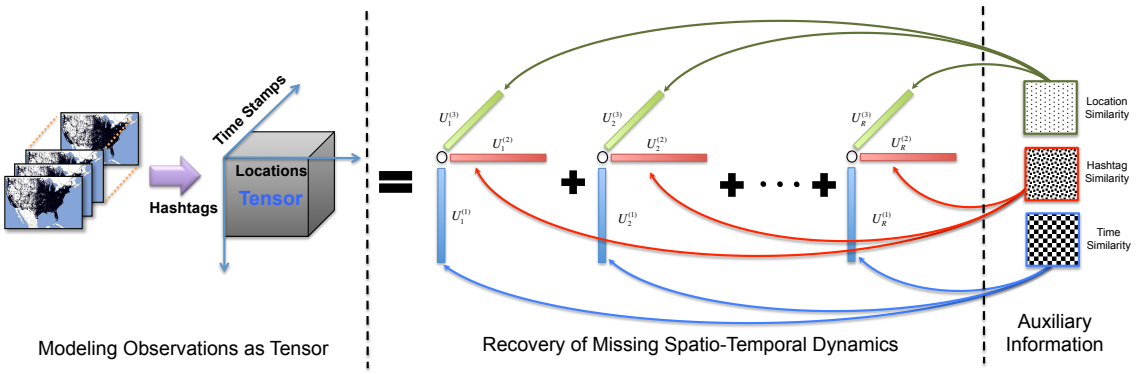


Figure 4.1: The proposed spatio-temporal dynamics recovery framework.

4.2.2 Twitter Hashtags

We ground our discussion in the rest of the chapter with respect to Twitter hashtags. A Twitter hashtag is a popular type of online meme that arises on Twitter, spreads from person to person (and from place to place), resulting in a fine-grained spatio-temporal log of information sharing dynamics. Note that the methods presented here may be applied to any other dataset with meme, location, time characteristics.

4.3 AIRCP: Auxiliary Information Regularized CP Model

In this section, we propose new scalable methods for recovering the spatio-temporal dynamics of online memes. Concretely, we propose to (i) model and exploit the latent relationships among locations, memes and times; (ii) embed these latent relationships into a tensor completion framework for uncovering the approximate complete data based only on partial observations; and (iii) show how the underlying tensor factorization can be efficiently solved compared to many existing methods. The high-level outline of the proposed solution is presented in Figure 4.1. We model the observed data as a tensor (left), and seek to recover the missing spatio-temporal dynamics (center) by integrating auxiliary information like the relationships between locations, memes and times (right). A key aspect of the

proposed approach is an iterative method to overcome the problem of incomplete auxiliary information. In the following, we introduce each part of the proposed solution in detail.

4.3.1 Modeling Recovery of Missing Data

We propose to model the recovery of missing hashtag data based on tensor models. Since hashtags are usually adopted in a few locations within limited life-spans [54], resulting in $\mathcal{X}$ being super sparse and low-rank, this model is built on a CP tensor completion model which can be represented by the following optimization problem:

$$\begin{aligned} \underset{U^{(n)},\mathcal{X}}{\text{minimize}} \quad & \frac{1}{2}\|\mathcal{X} - [\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]\|_F^2 + \frac{\lambda}{2}\sum_{n=1}^{3}\|\boldsymbol{U}^{(n)}\|_F^2 \\ \text{subject to} \quad & \boldsymbol{\Omega} * \mathcal{X} = \mathcal{T}, \boldsymbol{U}^{(n)} \geq 0, n = 1, 2, 3., \end{aligned}$$

where recall that $\mathcal{X}$ denotes the complete spatio-temporal dynamics of hashtags, $\mathcal{T}$ denotes the observations we do have, $\boldsymbol{U}^{(1)} \in \mathbb{R}^{N\times R}$, $\boldsymbol{U}^{(2)} \in \mathbb{R}^{L\times R}$, and $\boldsymbol{U}^{(3)} \in \mathbb{R}^{M\times R}$ are latent factor matrices for location, hashtag, and time dimensions, respectively, $R \ll min(N, L, M)$ is the number of latent factors as the rank of a tensor, $\frac{\lambda}{2}\sum_{n=1}^{N}\|\boldsymbol{U}^{(n)}\|_F^2$ is a Tikhonov regularization term used to avoid overfitting and provide a unique solution, and $\boldsymbol{\Omega}$ is a non-negative weight tensor with the same size as $\mathcal{X}$:

$$\boldsymbol{\Omega}(i,j,k) = \begin{cases} 1 & \text{if } \mathcal{X}(i,j,k) \text{ is observed,} \\ 0 & \text{if } \mathcal{X}(i,j,k) \text{ is unobserved.} \end{cases}$$

Our goal is to seek an estimated $\mathcal{X}$ for recovering the missing spatio-temporal dynamics of hashtags based upon the partial data we do observe. However, as it is the case in many linear-inverse problems, there may not be sufficient information to recover $\mathcal{X}$ only depending on the observed data. We call these *deficient linear-inverse* problems. Apparently, the case of recovering the spatio-temporal dynamics of hashtags with only observed

data is a deficient linear-inverse problem, e.g., it is very difficult to estimate occurrences for the hashtag *#iphone* in San Francisco even if we know the complete dynamics of hashtags in other cities such as New York, Austin, and Los Angeles. Hence, our intuition is to leverage the spatio-hashtag-temporal relationships inherent in the observed data in order to successfully recover the missing information. For instance, if knowing that people in San Francisco tend to adopt similar hashtags to people in Austin, then perhaps we can estimate the dynamics of the hashtag *#iphone* in San Francisco. Hence, we turn in the following discussion to how we can model these latent relationships for integration into the overall framework. We denote spatio-hashtag-temporal relationships as $\mathbf{\Theta}$ in the chapter.

4.3.1.1 Modeling Spatial Relationships

We begin by considering the spatial relationships that connect different locations. Our hope is that we may be able to use location similarity with respect to adopting hashtags to infer propagations to unobserved locations. Concretely, we consider two approaches to model the spatial relationships of hashtags:

Geographical Distance. A natural first step is to treat locations that are near each other as similar in terms of the hashtags that will be adopted. Previous studies such as [54] have shown that the closer two locations are, the most likely they are to adopt the same hashtags due to factors like common language and shared culture, customs, and interests. Hence, we can encode this intuition in a measure of location similarity. Motivated by radial basis function (RBF) kernel [121] widely used as a similarity measure, we propose a unified geographic distance similarity score $\mathbf{\Theta_{GD}}$ that captures the straightforward notion of geo-similarity, approaching 1 when two locations are physically proximate. The geographical similarity score $\mathbf{\Theta_{GD}}$ is defined as:

$$\mathbf{\Theta_{GD}}(l_i, l_j) = exp\left(-\frac{Dist(l_i, l_j)^2}{2\alpha^2}\right),$$

where α is a dispersion constant setting as 25 miles in this study. The score $\boldsymbol{\Theta_{GD}}$ considers the *Haversine* formula to calculate the geographic distance $Dist(l_i, l_j)$ between location l_i and location l_j (based on their GPS coordinates). Compared to a straightline distance, the Haversine formula accounts for Earth's spherical shape.

Adoption Similarity. An alternative approach is to measure the "idea" similarity between two locations. That is, there may be locations that are not necessarily close in terms of geographical distance, but that are close in terms of the hashtags they do adopt. In this way, we can measure the *adoption similarity* between any two locations by considering two factors (i) shared hashtags and (ii) deviation of their occurrences under certain probabilities. We first apply the Jaccard coefficient to measure the degree of shared hashtags $\boldsymbol{\Theta_{SH}}$ between two locations l_i and l_j:

$$\boldsymbol{\Theta_{SH}}(l_i, l_j) = \frac{|H_{l_i} \bigcap H_{l_j}|}{|H_{l_i} \bigcup H_{l_j}|},$$

where recall that H_l is the set of unique hashtags adopted in a location l, and $|H_l|$ is the number of unique hashtags adopted in a location l. Two locations sharing all hashtags in common have a score of 1.0; those sharing no hashtags in common have a score of 0.0. Then, inspired by the work [54], we define the probability of observing a hashtag h as:

$$P_h = \frac{\sum_{l_i \in L} o_{l_i}^h}{\sum_{h' \in H} \sum_{l_i \in L} o_{l_i}^{h'}},$$

where o_l^h is the number of occurrences for a hashtag h in a location l. P_h measures how likely a hashtag h occurs. Locations that adopt a hashtag with similar probabilities are considered more similar than locations that observe a hashtag with a very different adoption probabilities [54]. We continuously define the deviation of hashtag occurrences between

two locations as:

$$\boldsymbol{\Theta_{DL}}(l_i, l_j) = exp\left(-\sum_{h \in H'}(\frac{o^h_{l_i} - o^h_{l_j}}{o^h_{l_{max}}})^2 P_h\right),$$

where $H' = (H_{l_i} \bigcap H_{l_j})$ is denoted as the common hashtags for locations l_i and l_j, $o^h_{l_{max}}$ represents the maximum number of occurrence for hashtag h across all locations, which is used for normalization, and P_h yields the weighted average on the normalized squared difference of hashtag occurrences between two locations. $\boldsymbol{\Theta_{DL}}$, as a modified version of RBF kernel, indicates that two locations should be considered as similar while they have close distributions of occurrences as well as their real counts. Taking into account both of these two factors, we finally define the adoption similarity $\boldsymbol{\Theta_{AS}}$ between two locations by multiplying them together:

$$\boldsymbol{\Theta_{AS}}(l_i, l_j) = \boldsymbol{\Theta_{SH}}(l_i, l_j)\boldsymbol{\Theta_{DL}}(l_i, l_j),$$

where we assume that these two factors are independent and the values of $\boldsymbol{\Theta_{AS}}$ are in the range $[0, 1]$.

Fusion of Two Properties. Naturally, we can integrate both geographical distance similarity and adoption similarity between two locations into a unified model. The intuition is that we can take advantage of both geographical and "idea" similarities between locations. We adopt a simple linear model to fuse these two properties:

$$\boldsymbol{\Theta_{FS}}(l_i, l_j) = \tau\boldsymbol{\Theta_{GD}}(l_i, l_j) + (1 - \tau)\boldsymbol{\Theta_{AS}}(l_i, l_j),$$

where τ is a parameter used to control the contribution from the unified geographical similarity score $\boldsymbol{\Theta_{GD}}$ and the adoption similarity $\boldsymbol{\Theta_{AS}}$. In this study, τ is set to 0.3 via cross-validation.

4.3.1.2 *Modeling Hashtag Relationships*

Complementary to location relationships, we can also directly model the relationships among different hashtags. Some hashtags are mainly local phenomena while others have a global footprint. Hence, we can measure the spatial footprint of different hashtags and compare them toward finding "similar" footprints by considering two factors (i) spatial spread of hashtags and (ii) deviation of their occurrences across all locations. Inspired by Tobler's hypothesis [122], we first define the similarity of spatial spreading for a pair of hashtags as:

$$\boldsymbol{\Theta_{SP}}(h_i, h_j) = exp\left(-|\frac{d_{h_i} - d_{h_j}}{d_{max} - d_{min}}|\right),$$

where d_h is the average distance between all locations in which this hashtag h has been adopted, $|d_{h_i} - d_{h_j}|$ is used to measure the absolute difference of spatial spreading of two hashtags, $d_{max} = max(\{d_h, h \in H\})$, $d_{min} = min(\{d_h, h \in H\})$, and the term $d_{max} - d_{min}$ is a weight factor used for normalization. We then define the probability that a hashtag occurs in a location l as:

$$P_l = \frac{\sum_{h \in H} o_l^h}{\sum_{l_i \in L} \sum_{h \in H} o_{l_i}^h},$$

which represents how likely a hashtag is to be adopted in location l. Following a similar fashion on modeling $\boldsymbol{\Theta_{DL}}$, we define the deviation of occurrences for a pair of hashtags across all locations adopting them as:

$$\boldsymbol{\Theta_{DH}}(h_i, h_j) = exp\left(-\sum_{l \in L}(\frac{o_l^{h_i} - o_l^{h_j}}{o_l^{h_{max}}})^2 P_l\right),$$

where $o_l^{h_{max}}$ denotes the maximum number of occurrence for all hashtags in location l, which is used for normalization, and P_l yields the weighted average on the normalized

squared difference of real counts between two hashtags across locations where they have been adopted. Assuming that these two factors are independent, we finally model the hashtag similarity $\boldsymbol{\Theta_{HS}}$ by multiplying them together as:

$$\boldsymbol{\Theta_{HS}}(h_i, h_j) = \boldsymbol{\Theta_{SP}}(h_i, h_j)\boldsymbol{\Theta_{DH}}(h_i, h_j),$$

The values of $\boldsymbol{\Theta_{HS}}$ are in the range $[0, 1]$, implying that two hashtags adopted in the same locations with the same occurrences have a similarity score of 1; otherwise, they have a similarity score approaching 0.

4.3.1.3 Modeling Temporal Relationships

Finally, we consider enhancing the tensor completion by considering temporal relationships across memes. For the temporal properties of hashtags, we posit that adoptions of hashtags in consecutive timestamps may be similar. Hence, we can define the temporal similarity matrix $\boldsymbol{\Theta_T}$, capturing the smoothness of the spatio-temporal dynamics of hashtags by using the tri-diagonal matrix:

$$\boldsymbol{\Theta_T} = \begin{bmatrix} 0 & 1 & 0 & \dots \\ 1 & 0 & 1 & \dots \\ 0 & 1 & 0 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

where $\boldsymbol{\Theta_T}$ intuitively expresses the fact that $\mathcal{X}$ in consecutive timestamps are often similar, which has been a common assumption in related efforts to recover missing data [57, 123].

4.3.1.4 Integrating Auxiliary Information

So far, we have proposed several models to capture the relationships between locations, hashtags, and times. In this section, we investigate how to take advantage of this

auxiliary information into the basic CP tensor completion model. The basic idea is if two objects are similar, e.g., two cities have similar behaviors on adopting hashtags, the latent representations of these two cities should be similar. Therefore, we want to make the latent representations of two similar objects (i.e. locations, hashtags, or timestamps) as close as possible. We denote $\boldsymbol{\Theta}$ as a similarity matrix encoding relationships between entities like locations, hashtags, or times. The intuition above can be formulated as minimizing the following:

$$\begin{aligned}\mathcal{F} &= \frac{1}{2}\sum_{i,j}\boldsymbol{\Theta}(\boldsymbol{i},\boldsymbol{j})\|\boldsymbol{U}_i^{(n)} - \boldsymbol{U}_j^{(n)}\|^2 \\ &= \sum_{i,j}\boldsymbol{U}_i^{(n)}\boldsymbol{\Theta}(\boldsymbol{i},\boldsymbol{j})\boldsymbol{U}_i^{(n)^T} - \sum_{i,j}\boldsymbol{U}_i^{(n)}\boldsymbol{\Theta}(\boldsymbol{i},\boldsymbol{j})\boldsymbol{U}_j^{(n)^T} \\ &= \text{tr}(\boldsymbol{U}^{(n)^T}(\boldsymbol{D}-\boldsymbol{\Theta})\boldsymbol{U}^{(n)}) \\ &= \text{tr}(\boldsymbol{U}^{(n)^T}\boldsymbol{\mathcal{L}}\boldsymbol{U}^{(n)}),\end{aligned}$$

where $\boldsymbol{U}_i^{(n)}$ is the ith row of the factor matrix $\boldsymbol{U}^{(n)}$ for the nth-mode of a tensor $\boldsymbol{\mathcal{X}}, n \in \{1,2,3\}$, $tr(\cdot)$ is the matrix trace, $\boldsymbol{D}$ is a diagonal matrix with $\boldsymbol{D}(i,i) = \sum_j \boldsymbol{\Theta}(\boldsymbol{i},\boldsymbol{j})$, and $\boldsymbol{\mathcal{L}} = \boldsymbol{D} - \boldsymbol{\Theta}$ is the graph Laplacian of the similarity matrix $\boldsymbol{\Theta}$ which could be any of $\boldsymbol{\Theta_{GD}}$, $\boldsymbol{\Theta_{AS}}$, $\boldsymbol{\Theta_{FS}}$, $\boldsymbol{\Theta_{HS}}$ and $\boldsymbol{\Theta_T}$ introduced previously.

A straightforward way to integrate relationships between locations, hashtags, and times into the basic tensor completion model is as regularization terms such that we are able to regulate latent representations of two similar objects to make them as close as possible. Hence, by integrating these auxiliary information among locations, hashtags and times,

we can formulate the recovery of spatio-temporal dynamics as the objective function:

$$\begin{aligned} \underset{\boldsymbol{U}^{(n)},\mathcal{X}}{\text{minimize}} \quad & \frac{1}{2}\|\mathcal{X} - [\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]\|_F^2 + \frac{\lambda}{2}\sum_{n=1}^{3}\|\boldsymbol{U}^{(n)}\|_F^2 \\ & + \sum_{n=1}^{3}\alpha_n \mathrm{tr}(\boldsymbol{Z}^{(n)^T}\boldsymbol{\mathcal{L}}_n\boldsymbol{Z}^{(n)}) \\ \text{subject to} \quad & \boldsymbol{\Omega} * \mathcal{X} = \mathcal{T}, \boldsymbol{U}^{(n)} = \boldsymbol{Z}^{(n)} \geq 0, n = 1, 2, 3, \end{aligned} \tag{4.1}$$

where α is a parameter used to control the weight of auxiliary information between locations, hashtags, and times.

4.3.2 Optimization Algorithm

Since the objective function in Eq.(4.1) is not convex with respect to variables $\boldsymbol{Z}^{(n)}$ and $\boldsymbol{U}^{(n)}$ together, there is no closed-form solution for this optimization problem. Motivated by methods [26, 32], we now develop an efficient algorithm to find optimal solutions for the objective function above under the framework of ADMM that can be considered as an approximation of the method of multipliers. The objective function can be firstly written in the partial augmented Lagrangian function as follows:

$$\begin{aligned} L_\eta(\boldsymbol{U}^{(n)}, \boldsymbol{Z}^{(n)}, \boldsymbol{Y}^{(n)})_{n=1,2,3} = & \frac{1}{2}\|\mathcal{X} - [\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]\|_F^2 \\ & + \frac{\lambda}{2}\sum_{n=1}^{3}\|\boldsymbol{U}^{(n)}\|_F^2 + \sum_{n=1}^{3}\frac{\alpha_n}{2}\mathrm{tr}(\boldsymbol{Z}^{(n)^T}\mathcal{L}_n\boldsymbol{Z}^{(n)}) \\ & + \sum_{n=1}^{3} < \boldsymbol{Y}^{(n)}, \boldsymbol{Z}^{(n)} - \boldsymbol{U}^{(n)} > + \sum_{i=1}^{3}\frac{\eta}{2}\|\boldsymbol{Z}^{(n)} - \boldsymbol{U}^{(n)}\|_F^2, \end{aligned} \tag{4.2}$$

where $\boldsymbol{Y}^{(n)}$ is the matrix of Lagrange multipliers for $n = 1, 2, 3$, η is a penalty parameter.

Updating $\boldsymbol{Z}^{(1)}, \boldsymbol{Z}^{(2)}, \boldsymbol{Z}^{(3)}$. To update $\boldsymbol{Z}^{(1)}, \boldsymbol{Z}^{(2)}, \boldsymbol{Z}^{(3)}$, we can re-write objective function

in Eq.(7.2) as follows:

$$\underset{\boldsymbol{Z}^{(n)}}{\text{minimize}} \quad \frac{\alpha_n}{2}\text{tr}(\boldsymbol{Z}^{(n)^T}\boldsymbol{\mathcal{L}}_n\boldsymbol{Z}^{(n)}) + \frac{\eta_t}{2}\|\boldsymbol{Z}^{(n)} - \boldsymbol{U}_t^{(n)} + \frac{\boldsymbol{Y}_t^{(n)}}{\eta_t}\|_F^2. \tag{4.3}$$

Thus, $\boldsymbol{Z}^{(n)}$ can be efficiently updated by solving the optimization problem in Eq. (4.3) via:

$$\boldsymbol{Z}_{t+1}^{(n)} = (\eta_t\boldsymbol{I} + \alpha_n\boldsymbol{\mathcal{L}}_n)^{-1}(\eta_t\boldsymbol{U}_t^{(n)} - \boldsymbol{Y}_t^{(n)}),$$

where $\boldsymbol{I}$ is the identity matrix with the same size of $\boldsymbol{\mathcal{L}}_n$. By applying the eigen-decomposition to $\boldsymbol{\mathcal{L}}_n = \boldsymbol{V}_n\boldsymbol{\Lambda}_n\boldsymbol{V}_n^T$, we can re-write the equation above as:

$$\boldsymbol{Z}_{t+1}^{(n)} = \boldsymbol{V}_n(\eta_t + \alpha_n\boldsymbol{\Lambda}_n)^{-1}\boldsymbol{V}_n^T(\eta_t\boldsymbol{U}_t^{(n)} - \boldsymbol{Y}_t^{(n)}), \tag{4.4}$$

where $\eta_t + \alpha_n\Lambda_n$ is a diagonal matrix. Since $\boldsymbol{\mathcal{L}}_n$ is eigen-decomposed at the beginning of the optimization, $(\eta_t\boldsymbol{I} + \alpha_n\boldsymbol{\mathcal{L}}_n)^{-1}$ can be efficiently computed by only reversing entries on the diagonal of $\eta_t + \alpha_n\boldsymbol{\Lambda}_n$ instead of calculating the inverse of the whole matrix.

Updating $\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}$. To update $\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}$, the objective function in Eq.(7.2) can be re-written as follows:

$$\begin{aligned}\underset{\boldsymbol{U}^{(n)}}{\text{minimize}} \quad & \frac{1}{2}\|\boldsymbol{X}_{(n)}^t - \boldsymbol{U}^{(n)}\boldsymbol{B}^{(n)}\|_F^2 + \frac{\lambda}{2}\|\boldsymbol{U}^{(n)}\|_F^2 \\ & + \frac{\eta_t}{2}\|\boldsymbol{Z}_t^{(i)} - \boldsymbol{U}_t^{(i)} + \frac{\boldsymbol{Y}_t^{(i)}}{\eta_t}\|_F^2,\end{aligned} \tag{4.5}$$

where $\boldsymbol{B}^{(n)} = (\boldsymbol{U}^{(N)} \odot \cdots \boldsymbol{U}^{(n+1)} \odot \boldsymbol{U}^{(n-1)} \odot \cdots \boldsymbol{U}^{(1)})^T|_{N=3}$, $\odot$ is Khatri-Rao product, and $\boldsymbol{X}_{(n)}$ is the mode-n unfolding of the tensor $\boldsymbol{\mathcal{X}}$. Then this subproblem in terms of $\boldsymbol{U}^{(n)}$ is solved as follows:

$$\boldsymbol{U}_{t+1}^{(n)} = (\boldsymbol{X}_{(n)}^t\boldsymbol{B}^{(n)^T} + \eta_t\boldsymbol{Z}_{t+1}^{(n)} + \boldsymbol{Y}_t^{(n)})(\boldsymbol{B}^{(n)}\boldsymbol{B}^{(n)^T} + \lambda\boldsymbol{I} + \eta_t\boldsymbol{I})^{-1}. \tag{4.6}$$

Updating $\mathcal{X}$. To update $\mathcal{X}$, we can have that:

$$\boldsymbol{\mathcal{X}}_{t+1} = \boldsymbol{\mathcal{T}} + \boldsymbol{\Omega}^c * [\![\boldsymbol{U}_{t+1}^{(1)}, \boldsymbol{U}_{t+1}^{(2)}, \boldsymbol{U}_{t+1}^{(3)}]\!],$$

where $\boldsymbol{\Omega}^c$ is the complement of $\boldsymbol{\Omega}$ that is equal to $\mathbf{1} - \boldsymbol{\Omega}$.

Updating $\boldsymbol{Y}^{(n)}$. To update $\boldsymbol{Y}^{(n)}$, we can have that:

$$\boldsymbol{Y}_{t+1}^{(n)} = \boldsymbol{Y}_{t}^{(n)} + \eta_t(\boldsymbol{Z}_{t+1}^{(n)} - \boldsymbol{U}_{t+1}^{(n)}).$$

Updating η. We can accelerate the optimization algorithm by adaptively updating η. To update η, we can have that:

$$\eta_{t+1} = min(\rho\eta_t, \eta_{max}),$$

where ρ is a constant that we empirically set to 1.1 via cross-validation.

4.3.3 Recovery with Auxiliary Information

So far we have successfully solved the equation (4.1) by the proposed optimization algorithm based upon ADMM with leveraging auxiliary information. However, we are not able to obtain complete auxiliary information which encode similarities between locations, hashtags, and timestamps based on the sampled data. It is not reasonable to estimate missing spatio-temporal dynamics of hashtags by using auxiliary information derived from the complete data, which becomes a "Chicken-and-Egg" problem. In order to address this problem, we employ an iterative method. The initial similarity matrices derived from auxiliary information are computed based on the sampled data. And then similarity matrices will be re-calculated based on recovered spatio-temporal dynamics of hashtags. This procedure will iteratively proceed until there is no significant difference between the current and previous similarity matrices. This proposed auxiliary information regularized CP de-

composition method (AIRCP) is summarized in **Algorithm 1**.

4.4 Experiments

In this section, we conduct experiments to evaluate the effectiveness of the proposed AIRCP framework for recovering the spatio-temporal dynamics of hashtags. Concretely, we aim to answer the following questions:

- How effective is the proposed method compared with alternative state-of-the-art methods on recovering the missing spatio-temporal dynamics of hashtags?
- What are the effects of auxiliary information – here, in the form of relationships among locations, memes, and times – on recovering the spatio-temporal dynamics of hashtags? Are some relationships more informative than others?
- How dependent on the regularization parameters is the proposed method? That is, do we need to give special care for tuning the approach, or is there a wide choice of parameters that leads to robust recovery?

We begin by introducing the Twitter dataset and the evaluation and experimental setup. Then, we compare the performance of different tensor completion methods on both synthetic and real-word hashtag data sets. At last, the effects of the different auxiliary information sources and their corresponding regularization parameters for the proposed method are investigated.

4.4.1 Data

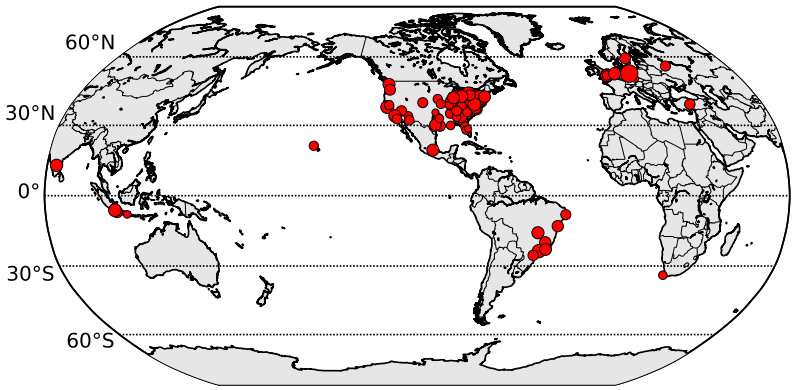
Our work here focuses on an initial sample of over 55 million geo-tagged tweets via the Twitter Streaming API between February 1st and October 1st in 2013. Each tweet is tagged with a latitude and longitude indicating a location where the user posted this tweet. In this study, we first convert the GPS locations associated with tweets to corresponding cities via reverse geo-coding, and then transfer the original timestamps accurate

Algorithm 1: Solving AirCP via ADMM

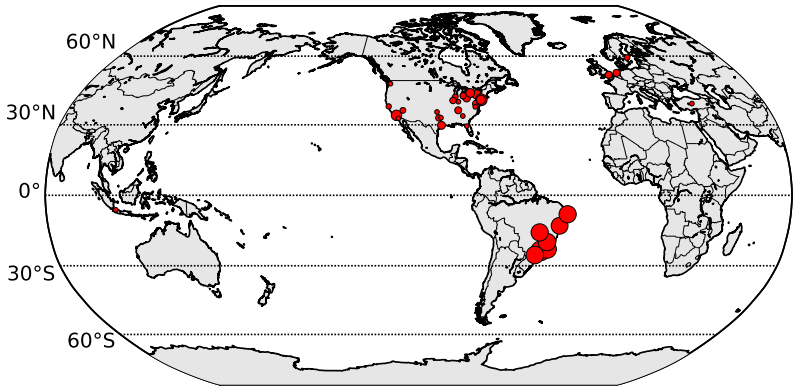
Input: $\mathcal{T}, \mathbf{\Omega}, \mathbf{\Theta}_0^{(n)}, \gamma, \lambda, \alpha_n, \rho, \eta, \eta_{max}, N$
Output: $\mathcal{X}, \boldsymbol{U}^{(n)}$

Algorithm *AirCP()*
  Initialize $\boldsymbol{U}_{iter}^{(n)}$, γ, λ, α_n, ρ, η_0, η_{max}, ϵ, N, $iter = 0$
  while Not Converged and $iter \leq I_{max}$ **do**
    Construct Laplacian matrices $\mathcal{L}_n$ for matrices $\mathbf{\Theta}_{iter}^{(n)}$
    $Optimization(\mathcal{T}, \mathbf{\Omega}, U_{iter}^{(n)}, \mathcal{L}_n, \gamma, \lambda, \alpha_n, \rho, \eta, \eta_{max}, N)$
    Re-calculate similarity matrices $\mathbf{\Theta}_{iter+1}^{(n)}$ based on $\mathcal{X}_{iter}$
    Check the convergence: $\max\{\|\mathbf{\Theta}_{iter+1}^{(n)} - \mathbf{\Theta}_{iter}^{(n)}\|_F, n = 1, 2, \ldots, N\} < \epsilon$
    $iter = iter + 1$
  return $\mathcal{X}_{iter}$, $\boldsymbol{U}_{iter}^{(n)}, n = 1, 2, \ldots, N$

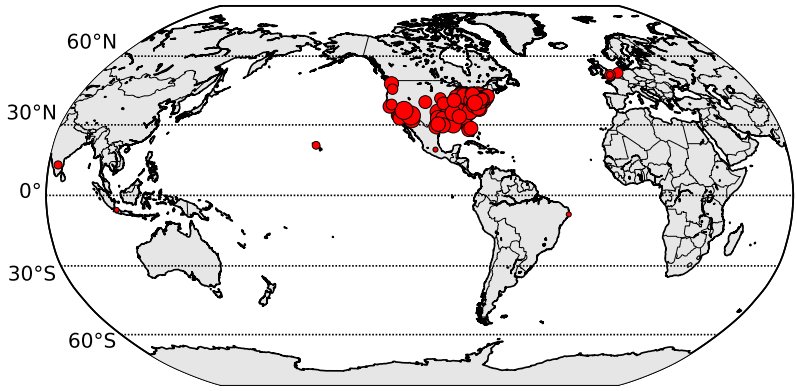
Procedure *Optimization($\mathcal{T}, \mathbf{\Omega}, \boldsymbol{U}_0^{(n)}, \mathcal{L}_n, \gamma, \lambda, \alpha_n, \rho, \eta_t, \eta_{max}, N$)*
  Initialize $\boldsymbol{Z}_0^{(n)} = \boldsymbol{Y}_0^{(n)} = 0$, $t = 0$, tol
  while Not Converged **do**
    for $n \leftarrow 1$ ***to*** N **do**
      Update $\boldsymbol{Z}_{t+1}^{(n)} \leftarrow (\eta_t \boldsymbol{I} + \alpha_n \mathcal{L}_n)^{-1}(\eta_t \boldsymbol{U}_t^{(n)} - \boldsymbol{Y}_t^{(n)})$
      Calculate $\boldsymbol{B}^{(n)} \leftarrow (\boldsymbol{U}^{(N)} \odot \cdots \boldsymbol{U}^{(n+1)} \odot \boldsymbol{U}^{(n-1)} \odot \cdots \boldsymbol{U}^{(1)})^T$
      Update
      $\boldsymbol{U}_{t+1}^{(n)} \leftarrow (\boldsymbol{X}_{(n)}^t \boldsymbol{B}^{(n)^T} + \eta_t \boldsymbol{Z}_{t+1}^{(n)} + \boldsymbol{Y}_t^{(n)})(\boldsymbol{B}^{(n)} \boldsymbol{B}^{(n)^T} + \lambda \boldsymbol{I} + \eta_t \boldsymbol{I})^{-1}$
    Update $\mathcal{X}_{t+1} = \mathcal{T} + \mathbf{\Omega}^c * [\![\boldsymbol{U}_{t+1}^{(1)}, \boldsymbol{U}_{t+1}^{(2)}, \ldots, \boldsymbol{U}_{t+1}^{(N)}]\!]$
    for $n \leftarrow 1$ ***to*** N **do**
      Update $\boldsymbol{Y}_{t+1}^{(n)} = \boldsymbol{Y}_t^{(n)} + \eta_t(\boldsymbol{Z}_{t+1}^{(n)} - \boldsymbol{U}_{t+1}^{(n)})$
    Update $\eta_{t+1} = \min(\rho \eta_t, \eta_{max})$
    Check the convergence: $\max\{\|\boldsymbol{U}_{t+1}^{(n)} - \boldsymbol{Z}_{t+1}^{(n)}\|_F, n = 1, 2, \ldots, N\} < tol$
    $t = t + 1$
  return $\mathcal{X}$, $\boldsymbol{U}^{(n)}, n = 1, 2, \ldots, N$



(a) #iphone



(b) #fato



(c) #healthcare

Figure 4.2: Distributions of three hashtags.

to the second to corresponding dates. Each geo-tagged tweet can be represented by a tuple $< hashtag, city, date >$. To avoid very sparsely represented hashtags, we only consider hashtags having at least 1,000 occurrences across all cities where at least 20 unique hashtags have been adopted during the period of our data collection. Since some hashtags have appeared before the first day of the sample, we only keep those hashtags that first appear after February 1st, 2013, resulting in 4,723 unique hashtags occurring in 2,415 cities. After randomly selecting 2,000 of 4,723 hashtags, the data set consists of 2,000 hashtags occurring in 1,278 cities in the world over a span of 242 days, which we model as a tensor $\mathcal{X} \in \mathbb{R}^{2000 \times 1278 \times 242}$. For the experiments, we view this sample as if it were the true (complete) spatio-temporal dynamics of the 2,000 hashtags across these 242 days. To illustrate, Figure 4.2 shows the global footprint of three different hashtags (#*iphone*, #*fato*, and #*healthcare*) in the dataset.

4.4.2 Experimental Setup and Metrics

We evaluate the effectiveness of the proposed framework compared with alternative methods by evaluating them over the three scenarios introduced in Section 4.2: Scenario 1, in which we have random missing observations; Scenario 2, in which some locations are missing entire hashtags; and Scenario 3, in which we are missing entire locations. We set the parameters in Eq.(4.2) through cross-validation with a separate validation dataset. We empirically set $\lambda = 0.1$ and $\alpha_1 = \alpha_2 = \alpha_3 = 0.1$ for all following experiments.

To investigate the quality of the proposed framework, we adopt *Relative Error*, and *Accuracy@k* as evaluation metrics. Relative Error is defined as:

$$RelativeError = \frac{\|X - Y\|_F}{\|Y\|_F},$$

where X is the recovered tensor and Y is the ground-truth tensor. Accuracy@k represents the percentage of correctly predicted popular hashtags out of the top-k popular hashtags.

Formally, if we denote S_l as the real top-k popular hashtags at a location l and $\hat{S}_l$ as the set of popular hashtags selected by a recovery method at a city l, we have:

$$Accuracy@k = \frac{\hat{S}_l^t \cap S_l^t}{k},$$

which is in the range [0,1]. In the following experiments, we evaluate k at 1, 5, and 10.

4.4.3 Baseline Methods

Previous research has shown that tensor-based completion methods outperform matrix-based ones [6, 57, 120]. Hence, we focus our evaluation here on tensor-based state-of-the-art baseline methods:

- *Tensor Factorization with Auxiliary Information (TFAI)*: The first baseline is a tensor analysis method introduced in [32] that integrates auxiliary information. We adopt the within-mode auxiliary information method that performs better than the cross-mode method according to the results.

- *Trace Norm-based CP Decomposition (TNCP)*: The second baseline method regularizes the trace norm in the CP tensor decomposition method based upon ADMM [26]. We choose parameters $\lambda = 10$, and $\alpha = 0.33$ that result in the best performance for this method.

- *Low-Rank Tensor via Convex Optimization (LRCO)*:The third baseline is a low-rank tensor factorization method with a trace norm regularization [120]. We adopt the "mixture" version, which models the tensor as a mixture of K sub-tensors. We set the initial step-size $\eta_0 = 0.1$ and $\lambda = 0$.

- *Weighted Tucker Decomposition (WTucker)*: The fourth baseline we adopted is a weighted Tucker decomposition [124] that is similar in spirit to PARAFAC [40]

method fort recovering missing data.

- *Fast Low-Rank Tensor Completion (FaLRTC)*: Finally, we consider the fast low-rank tensor completion method [6] as one of our baselines, which estimates missing data based on the smoothed trace norm.

4.4.4 Evaluating AIRCP over Synthetic Data

As a first step toward evaluating the effectiveness and efficiency of the AirCP method, we first test over a synthetic dataset before moving on to the real hashtag data. We generate a low-rank (10,10,10) tensor $\mathcal{M} \in \mathbb{R}^{100\times100\times100}$ with correlated objects as the ground truth data. The factor matrices $\boldsymbol{U}^{(1)} \in 100 \times 10$, $\boldsymbol{U}^{(2)} \in 100 \times 10$, and $\boldsymbol{U}^{(3)} \in 100 \times 10$ are generated by the following linear formulae [32]:

$$\begin{aligned}
\boldsymbol{U}^{(1)}(i,r) &= i\varepsilon_r + \varepsilon', \quad i = 1,2,\ldots,100, r = 1,2,\ldots,10 \\
\boldsymbol{U}^{(2)}(j,r) &= j\zeta_r + \zeta', \quad j = 1,2,\ldots,100, r = 1,2,\ldots,10 \\
\boldsymbol{U}^{(3)}(k,r) &= k\eta_r + \eta', \quad k = 1,2,\ldots,100, r = 1,2,\ldots,10
\end{aligned}$$

where $\{\varepsilon_r, \varepsilon'_r, \zeta_r, \zeta'_r, \eta_r, \eta'_r\}_{r=1,2,\ldots,10}$ are constants generated by the standard Gaussian distribution $N(0,1)$. Then the synthetic tensor $\mathcal{M}$ is calculated as $\mathcal{M} = \mathcal{J} \times_1 \boldsymbol{U}^{(1)} \times_2 \boldsymbol{U}^{(2)} \times_3 \boldsymbol{U}^{(3)}$ where $\mathcal{J} \in \mathbb{R}^{10\times10\times10}$ is a unit tensor with all of its super-diagonal elements being 1 and the other elements being 0 and $\times_i$ means the tensor-matrix operation for the ith dimension of tensor. Since each factor matrix is generated by linear functions mentioned above column by column, the consecutive rows are similar to each other. Therefore, we

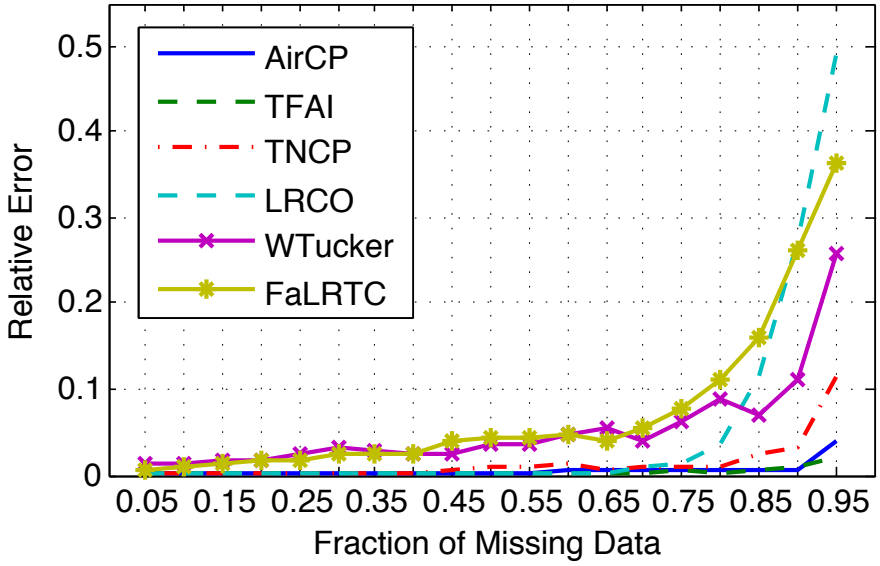


Figure 4.3: Comparison of recovery results over synthetic data.

generate the similar matrix for the ith mode as the following tri-diagonal matrix:

$$\mathbf{\Theta}_i = \begin{bmatrix} 0 & 1 & 0 & \dots \\ 1 & 0 & 1 & \dots \\ 0 & 1 & 0 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{4.7}$$

We randomly sample entries from $\mathcal{M}$ and recover the complete tensor by varying the fraction of unobserved entries from 5% to 95%. We set the tolerance of error as 10^{-5} and the maximal number of iteration as 1,000 for all methods we tested here.

We show in Figure 4.3 the relative error for all methods, averaged over 10 independent runs. At a moderate fraction of missing data, most of the methods perform comparably, with only WTucker and FaLRTC performing clearly worse. But in cases when there is a large fraction of missing data (i.e. greater than 75%), we see that AIRCP and TFAI

Table 4.2: Computation time (in seconds) over synthetic data as the fraction of missing data (FM) varies.

FM	AIRCP	TFAI	TNCP	LRCO	WTucker	FaLRTC
20%	**3.97**	46.36	4.51	58.64	195.22	5.47
40%	4.27	49.87	**3.94**	49.61	186.13	5.62
60%	**3.89**	35.98	4.08	51.30	202.58	4.78
80%	4.76	43.75	**4.16**	60.04	183.98	4.46

achieve the lowest relative error in all cases and that this error is objectively low. This result is encouraging since it indicates that the proposed framework for spatio-temporal dynamics recovery can achieve robust recovery in realistic scenarios where only a small fraction of data is available.

While AIRCP and TFAI achieve relatively lower error rates, what about their comparative efficiency? We present the average computation time (in seconds) of all tested approaches in Table 4.2. We can observe that AIRCP is an order of magnitude faster than TFAI and that it is on par with both TNCP and FaLRTC, which both demonstrate higher relative errors (as shown in Figure 4.3). Hence, these experiments over synthetic data show the potential of the proposed AIRCP method to achieve low error rates while also being more appropriate for large-scale data.

4.4.5 Evaluating AIRCP over Hashtag Data

Given these encouraging results, we now turn to an examination of AIRCP over the real hashtag data. We consider the three missing data scenarios introduced previously. For all cases, we set the rank of the tensor to 10. For Scenario 1 (*Random Missing Observations*), we randomly select a fraction of all hashtag-location-time counts and assume that these are unobservable (that is, missing). We report results by varying the fraction from 25% to 55% to 85%. For Scenario 2 (*Missing Entire Memes at Some Locations*), we randomly select for each location some fraction of hashtags that are unobservable. Again, we report results

Table 4.3: Relative errors for recovering missing hashtags as the fraction of missing data varies from 25% to 55% to 85%. We observe that AIRCP is an order of magnitude faster than TFAI.

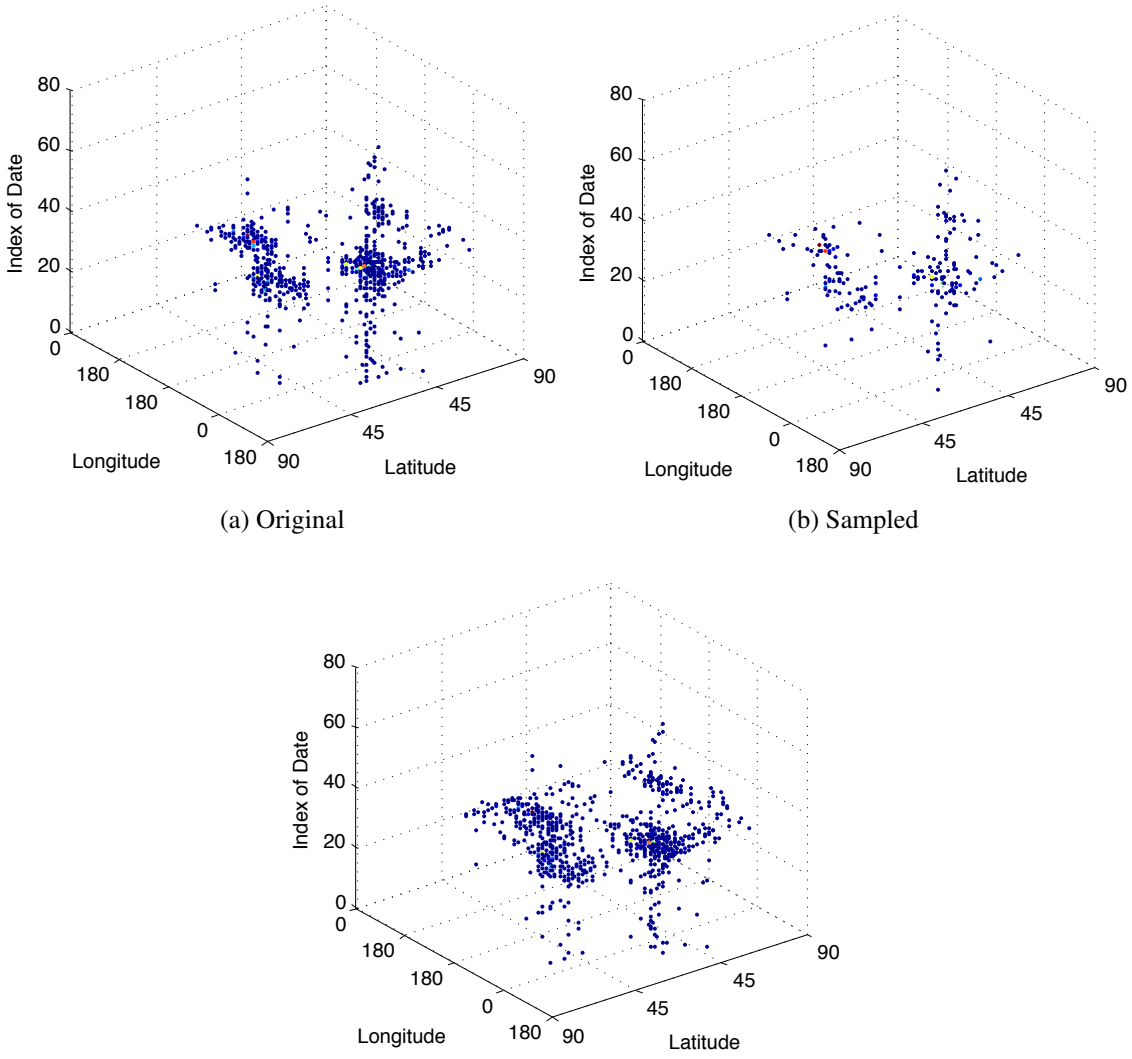
Method	**Scenario 1**			**Scenario 2**			**Scenario 3**			**A.I.***
	25%	55%	85%	25%	55%	85%	25%	55%	85%	
AirCP(FS+HS+T)	0.1059	**0.3330**	**0.5079**	**0.2032**	**0.4615**	**0.6017**	0.2830	**0.5772**	**0.7680**	N/A
TFAI	**0.0999**	0.3565	0.5314	0.2282	0.4870	0.6336	**0.2828**	0.5863	0.7861	3.34%
TNCP	0.1829	0.4198	0.5762	0.2846	0.5937	0.7378	0.3674	0.5919	0.8272	19.62%
LRCO	0.2307	0.4753	0.6851	0.3245	0.6261	0.798	0.4097	0.6288	0.9026	28.02%
WTucker	0.4859	0.778	1.1602	0.7427	1.0155	1.3256	1.1852	1.3914	1.7383	62.65%
FaLRTC	0.2417	0.4543	0.6548	0.3156	0.5621	0.8069	0.3592	0.6243	0.8802	25.09%

A.I.*: Average improvement comparing with AIRCP.

for 25%, 55%, and 85%. Finally, for Scenario 3 (*Missing Entire Locations*), we randomly select a fraction of the 1,278 locations and assume that these locations are completely unobservable (missing) across the whole collection period. We evaluate all methods using a fraction of missing locations of 25%, 55%, and 85%.

We present in Table 4.3 the relative error for all three scenarios across all approaches for three levels of missing data (25%, 55%, and 85%). Reinforcing our observations from the synthetic data experiment, we witness that AIRCP achieves better performance than TFAI with an average improvement of 3.34% over real hashtag data. In practice, again, TFAI still takes around an order of magnitude longer to calculate than the proposed AIRCP method. We see that AIRCP gives an average improvement of 27.8% in terms of relative error over other alternative methods. And as the sparser the observed tensor is (that is, the smaller the number of actual observed hashtags), we see that AIRCP gives an even greater improvement versus the alternatives. To illustrate, Figure 4.4 shows an example recovery for the hashtag *#mtvema* for Scenario 1 when 85% of the data is missing. We can see that the proposed method can successfully recover the sampled data based upon limited sample data (15% of the complete data).

Returning to Table 4.3, for both Scenarios 2 and 3 in which either a portion of all



(a) Original

(b) Sampled

(c) Recovered

Figure 4.4: An example recovery for the hashtag #mtvema when 85% of the data is missing (Scenario 1).

hashtags for a location are missing or the entire location is unobserved (which places great burden on the recovery framework, since there are not even partial observations for those hashtags in those locations as in Scenario 1), we see that integrating the latent relationships among locations, hashtags, and time (e.g., distances between two cities, similarities between hashtags, same hashtags adopted by two cities) can lead to a significant improvement through the tensor factorization. These relationships can alleviate the problem of sparsity to some extent and provide valuable information for the tensor factorization to obtain more interpretable low-rank representations.

Moreover, after varying the rank of tensor from 5 to 20, we found that the proposed method has a better consistency than other alternative state-of-the-art methods as the rank of tensors increases, implying that the proposed method is more robust on predicting missing diffusion dynamics of hashtags. The details are omitted here due to the limited space.

4.4.6 Recovery Under Constraints

While the previous experiment examined whether we could recover the count of the number of hashtags in a location at a particular time, we now turn to two more constrained situations that could arise in practice.

Appearance of Hashtags. In the first situation, we consider the task of determining whether or not a hashtag has appeared at a location at a particular timestamp. By considering only this binary information (rather than count information), we can explore the quality of the proposed approach at identifying rare hashtags, rather than emphasizing on hashtag counts as in the previous experiments. In this way, we can determine how well the approaches recover the appearance information of hashtags. For this experiment, for a hashtag h, the corresponding cell $\mathcal{X}(h, l, t)$ will be set as 1 if that hashtag appears in a city l at a date t. Otherwise, it will be assigned to 0. For the recovered tensor $\mathcal{X}$, the cell $\mathcal{X}(h, l, t)$ will be set as 1 if the value of that cell is larger or equal to a threshold; otherwise, it will be set to

Table 4.4: Relative errors for recovering appearances of hashtags as the fraction of missing data varies from 25% to 55% to 85%. We witness that AIRCP is an order of magnitude faster than TFAI.

Method	**Scenario 1**			**Scenario 2**			**Scenario 3**			**A.I.***
	25%	55%	85%	25%	55%	85%	25%	55%	85%	
AirCP(FS+HS+T)	**0.2032**	0.4675	**0.7017**	**0.2405**	**0.3653**	**0.4719**	**0.3083**	**0.5965**	**0.7451**	N/A
TFAI	0.2082	**0.4470**	0.7236	0.2668	0.3976	0.4970	0.3246	0.6281	0.7900	4.40%
TNCP	0.2846	0.5937	0.7878	0.3148	0.4644	0.5799	0.4071	0.6599	0.8541	18.98%
LRCO	0.3245	0.6261	0.7980	0.6303	0.6142	0.6428	0.4380	0.6605	0.9315	29.23%
WTucker	0.7427	1.0155	1.3256	0.8484	0.8651	0.7403	1.2191	1.4297	1.7844	61.66%
FaLRTC	0.3156	0.5621	0.8069	0.4767	0.599	0.6313	0.4154	0.6660	0.9089	26.03%

A.I.*: Average improvement comparing with AIRCP.

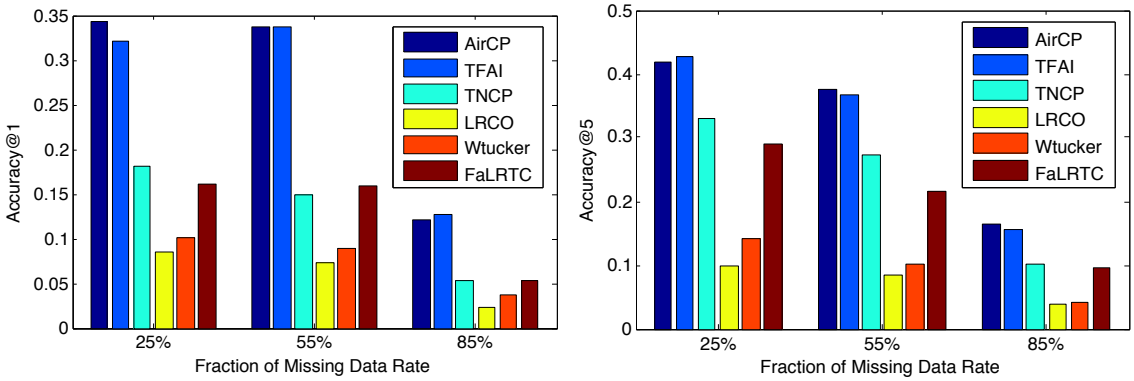
0. The threshold is empirically set to 0.3 via cross-validation. In this experiment, we vary the fraction of missing data in [25%, 55%, 85%] for all three scenarios, meaning that the observations we randomly selected based on fractions of missing data can be considered as training data, and the remainder as the test data. We empirically set the rank of tensor to 10. The experimental results in terms of relative errors are illustrated in Table 4.4.

We can see that in general, the proposed AIRCP consistently outperforms other alternative methods including TFAI, TNCP, LRCO, WTucker, and FaLRTC in all three scenarios. Specifically, it gives an average improvement of 28.01% on the relative error over other alternative methods. This indicates that auxiliary information among locations, hashtags, and times can help predict whether a hashtag has appeared in a location at a time or not. Similar to the previous experiment, TFAI has comparable performance with the proposed method, which performs worse in scenarios 1 and 2 and slightly better in scenario 3. However, as TFAI requires longer computation time, the proposed AIRCP method is more efficient at this task.

Popularity of Hashtags. In the second situation, we consider the task of determining the top-k hashtags at a location at a particular timestamp. In this way, we can explore the quality of the proposed approach at identifying the popularity of hashtags. We evaluate

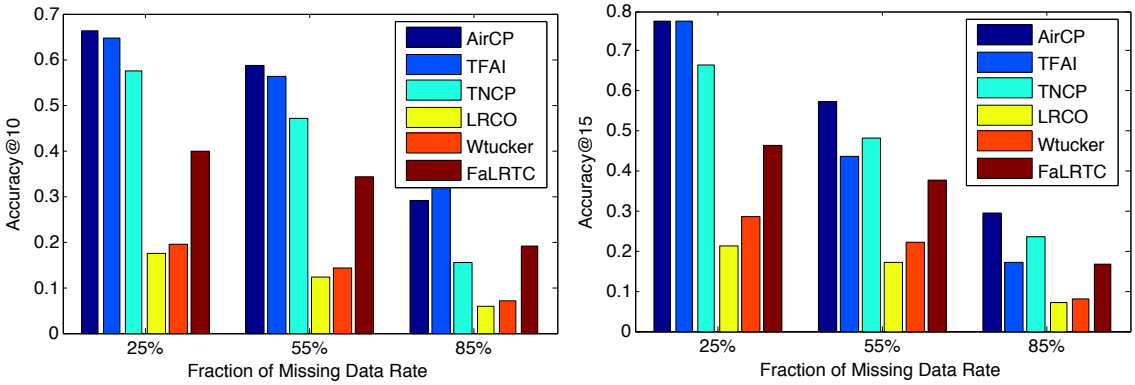
the performance of the AIRCP method in the recovery of top-k popular hashtags in 1,278 cities by varying the value of k as the fraction of missing values varies. For a hashtag h, the corresponding cell $\mathcal{X}(h, l, t)$ will be set as 1 if the total number of occurrences of that hashtag is one of the top-k largest among all hashtags occurred in that city l after the date t; otherwise, it will be assigned to 0. For this problem, only Scenario 3 is reasonable since for Scenarios 1 and 2, we are not likely to accurately mark the top-k popular hashtags in a city only depending on observed sampled data. It indicates that we cannot know the top-k popular hashtags in a location l after a date t while only observing partial data. For instance, we mark hashtags *#cjbbq* and *#subway* as two of top-3 popular hashtags in Houston after the date t only based on observed sample data in the period of the data collection. Nevertheless, once we can retrieve the whole diffusion data of hashtags in Houston, the top-3 popular hashtags are *#northgate*, *#rockets* and *#texian* as we do not observe or partially observe their diffusion data. Therefore, Scenarios 1 and 2 would never happen on this problem. In the implementation of the proposed method, we empirically set the rank of tensor to 10.

The performance comparison is presented in Figure 4.5. As we can see, overall, the proposed AIRCP generally gives the best performance in terms of accuracy for different k with a maximal accuracy of 77.7%. Since the distribution of hashtags at a location usually follows a Zipfian distribution, it could be harder for the problem of predicting top-k hashtags at a location at a date as the value of k decreases. This hypothesis is confirmed in the performance of our method. We observe that all these compared methods as well as the AIRCP method perform better with higher values of k. In addition, we can see that the AIRCP method performs consistently when the fraction of missing data is less then 85%. All these results illustrate that the proposed method can successfully predict top-k popular hashtags for a city after a date with limited observations.



(a) Top-1 popular hashtag

(b) Top-5 popular hashtags



(c) Top-10 popular hashtags

(d) Top-15 popular hashtags

Figure 4.5: Recovering hashtag popularity: Accuracy@1, 5, 10, and 15 as the fraction of missing data varies from 25%, 55% to 85%. Though AIRCP only achieves slightly better performance than TFAI, it has much better time efficiency for the computation with around an order of magnitude faster speed.

4.4.7 Effects of Auxiliary Information

We next turn to the relative impact of the different sources of auxiliary information. Recall that we consider different relationships among locations, hashtags, and time. Here, we are interested to explore whether some of these relationships are more informative than others. To that end, we narrow our focus on the problem of recovering hashtag counts under Scenario 1 (where a random fraction of hashtags are missing), and empirically fix the rank of tensor to 10. We evaluate the proposed AIRCP method with different combinations of auxiliary information. In Figure 4.6, we show how auxiliary information affects the performance of the proposed method in terms of the relative error by varying the fractions of missing data. We can see that, in general, the proposed AIRCP method integrating all three types of auxiliary information (modeled in Section 4.3) achieves the best performance than those only integrating part of the auxiliary information, indicating that the proposed method successfully makes use of all useful information sources to perform effective recovery for the spatio-temporal diffusion dynamics of hashtags. For the spatial information, we find that AIRCP with the fusion of geographical distance similarity (GD) and adoption similarity (AS) performs better than ones with either GD or AS solely. This result implies that the integration of these two types of information yields complementary evidence of hashtag adoption. We also observe that AIRCP with only the hashtag similarity (HS) has comparable performance with one integrating GD and AS (denoted as FS). In summary, the use of all three types of auxiliary information can help enhance the performance of the proposed AIRCP.

4.4.8 Effects of Regularization Parameters

Finally, we explore the impact of the regularization parameters on the quality of hashtag recovery. Recall that these parameters control the contributions of the relationships between locations, hashtags and time on the recovery framework. In order to better un-

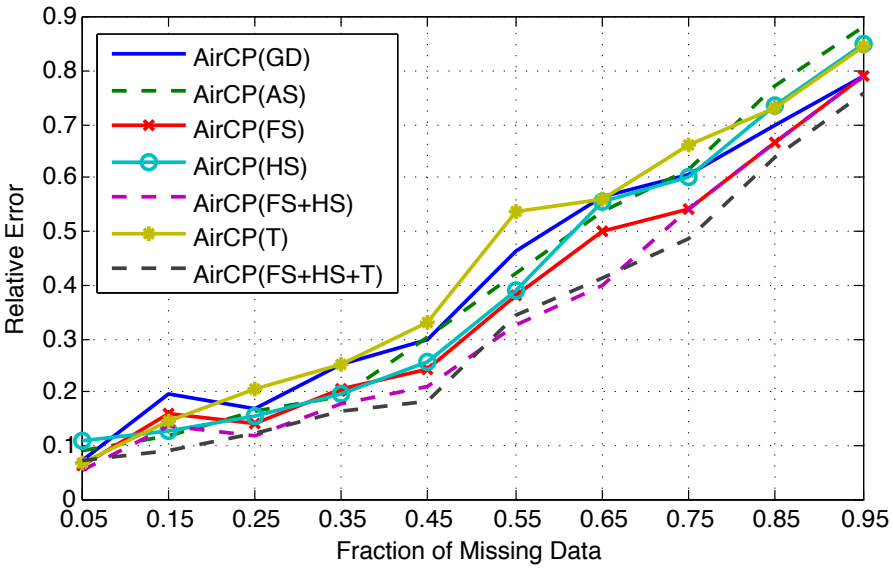


Figure 4.6: Relative errors for different combinations of auxiliary information.

derstand the effect of different choices of these parameters, we vary their values in the range [0.001 0.01, 0.1, 1, 10], and then evaluate the AIRCP method for the scenario in which some hashtags are missing (Scenario 1) with a fraction of missing data of 55%. The rank of the tensor is set to 10 and other settings are the same as we set in Section 4.4.5. We observe in Figure 4.7 that the proposed AIRCP method achieves relatively stable performance when the parameters are in the range [0.01, 0.1, 1]. This result indicates that the proposed framework is fairly robust to reasonable choices of these parameters. Specifically, comparing all results for parameters, we find that setting $\alpha_1(city) = 0.1$, $\alpha_2(hashtag) = 0.1$ and $\alpha_3(date) = 0.1$ achieves the best performance with a relative error of 0.3330, and that parameter settings of $\alpha_1(city) > 0.1$, $\alpha_2(hashtag) > 0.1$ and $\alpha_3(date) > 0.1$ lead to fairly stable relative errors. These results indicate the stability of the proposed AIRCP to these regularization parameters. Similar results can be found when we set the fraction of missing data to 25% and 85%.

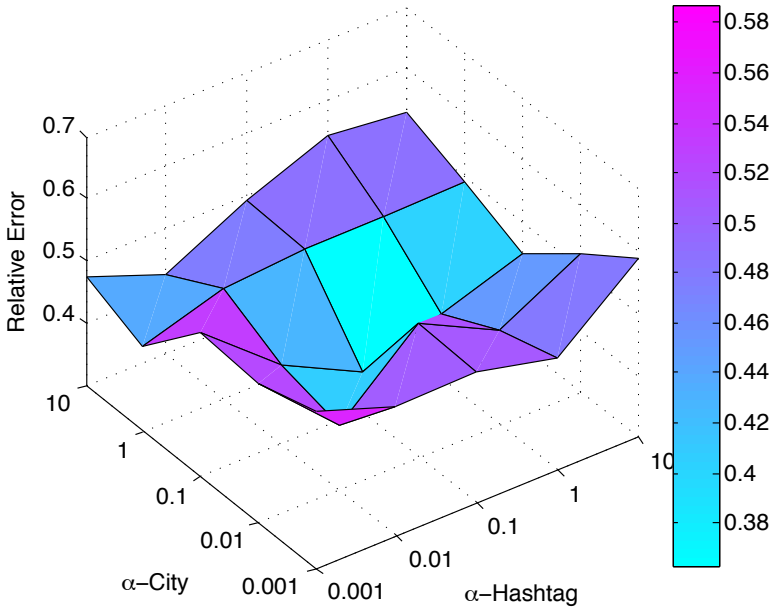


Figure 4.7: Relative errors for different parameter settings: $\alpha_3(date) = 0.1$ with 55% of the fraction of missing data.

4.5 Conclusion

In this paper, we have tackled the critical problem of recovering spatio-temporal dynamics of memes. Concretely, we have proposed a tensor-based spatio-temporal dynamics recovery framework that leverages auxiliary information among locations, hashtags, and times with better time efficiency. Through experimental evaluation on both synthetic and real-world Twitter hashtag data, we see that the proposed framework outperforms alternative state-of-the-art methods with an average improvement of over 27%, and find that the integration of auxiliary information among locations, hashtags, and times are crucial factors in the performance of the proposed framework.

5. TAPER: A CONTEXTUAL TENOSR-BASED APPROACH FOR PERSONALIZED EXPERT RECOMMENDATION*

In this chapter, we tackle the challenges of **sparsity**, **similarity** and **specificity** by extending the proposed tensor learning framework to recommendation. Recommendation is a typical missing information problem on social media, where user feedback is often missing and so latent preferences must be inferred. Specifically, we focus on recommending personalized experts to users, which can be treated as a tensor completion problem in order to estimate unobserved expert preferences of individuals. Through careful modeling of contextual factors like the geo-spatial, topical, and social preferences of users, we propose a tensor-based personalized expert recommendation framework that integrates these factors for revealing latent connections between homogeneous entities (e.g., users and users) and between heterogeneous entities (e.g., users and experts). Through extensive experiments over geo-tagged Twitter data, we find that the proposed framework can improve the quality of recommendation by over 30% in both precision and recall compared to the state-of-the-art baseline methods.

5.1 Introduction

Recommender systems are a cornerstone of how we engage online – by impacting the media we consume, the friends we connect with, and the products we purchase. A typical assumption in many recommender systems is to focus on *specific items* like movies, songs, or books as the basis of recommendation. In a separate direction, there are efforts to focus on *high-quality content producers* rather than specific items [125, 126]. These content

*Reprinted with permission from "TAPER: A Contextual Tensor-Based Approach for Personalized Expert Recommendation" by Hancheng Ge, James Caverlee, and Haokai Lu, 2016. Proceedings of the 10th ACM Conference on Recommender Systems. Copyright 2016 by ACM. DOI: http://dx.doi.org/10.1145/2959100.2959151

producers – like creators of highly-rated Spotify playlists, Amazon's top reviewers, or media curators on platforms like Tumblr, Facebook, and Twitter – can potentially serve as conduits to high-quality curated items. Indeed, previous research has shown that in some cases item-based recommenders can be improved by biasing the underlying models toward the opinions of these "experts" [127].

While some high-quality content producers are easily identifiable (say, by being "favorited" or starred many times), there is a long-tail of potential candidates for whom we have little evidence. Hence, a natural question is whether we can identify these high-quality content producers – whom we shall refer to as *experts* in the rest of this chapter – and recommend them to the right people. Such *personalized expert recommendation* faces a number of key challenges, though. First, many existing works have aimed at uncovering expert users in online systems – e.g., [60, 59, 61, 128, 129] – but typically without an emphasis on personalized recommendation. That is, these efforts have often attempted to explore general topic experts with broad appeal, e.g., the best doctors in Seattle or the top engineers in a certain field, rather than connecting users with personal experts. Second, personalized expert recommendation faces extreme sparsity since few users provide feedback on the quality of content producers. Third, there are typically complex relationships between users, candidate experts, and topics of interest.

Hence, we aim in this manuscript to tackle these challenges via a personalized expert recommendation framework called **TAPER** – for **T**ensor-based **A**pproach for **P**ersonalized **E**xpert **R**ecommendation. This proposed approach inherits the advantages of traditional recommender systems by making personalized recommendations based on the history of actions by similar users. In this way, specific personal experts can be recommended to individuals, rather than relying on globally-recognized (and less personalized) ones. While matrix factorization approaches have shown success in mitigating sparsity [66, 130], ultimately they are restricted to two-dimensional data (e.g., a user-expert matrix). In contrast,

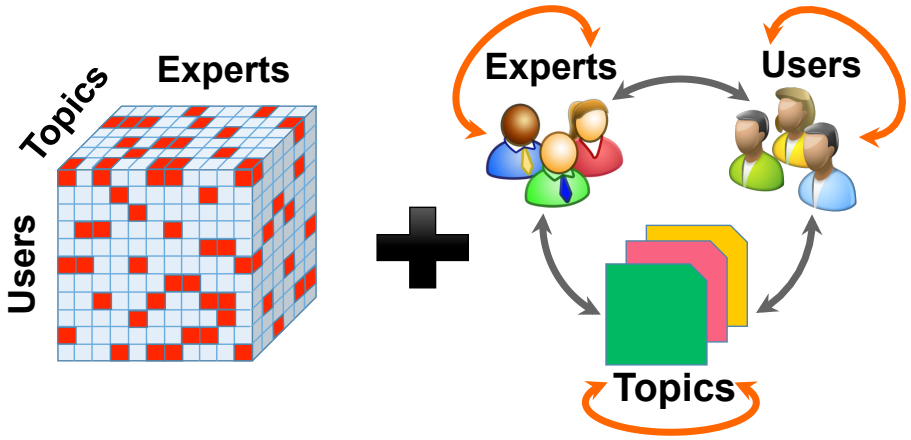


Figure 5.1: Personalized expert recommendation with contextual factors: augmenting the sparse tensor (left) with contextual factors (right) like the geo-spatial, topical, and social relationships between homogeneous entities (e.g., users and users) and between heterogeneous entities (e.g., users and experts).

user preferences for experts may be impacted by many contextual factors including the topic of interest, the location of the user (and possibly of the expert), as well as social connections among users and experts, among many others. As illustrated in Figure 5.1, these user-expert-topic preferences naturally suggest a tensor-based approach where these multiple and varied relationships may augment the sparse tensor (on the left) by considering the relationships (on the right) between both homogeneous entities (e.g., users and users, experts and experts) and the relationships between heterogeneous entities (e.g., users and experts, topics and experts).

Through the TAPER framework, we explore questions like: What kinds of contextual factors impact preferences for personalized experts? How can these contextual factors be integrated into a tensor-based personalized expert recommender? Do these factors result in higher quality recommendations than state-of-the-art methods? And are some contextual factors more important than others?

5.2 Problem Statement

Let $\boldsymbol{L} = \{u_1, u_2, \ldots, u_N\}$ be a set of users where N is the total number of users, and $\boldsymbol{E} = \{e_1, e_2, \ldots, e_M\}$ be a set of experts where M is the total number of experts and $\boldsymbol{E} \in \boldsymbol{L}$. An expert e_i may have expertise in multiple topics expressed as $\boldsymbol{T} = \{t_1, t_2, \ldots, t_K\}$ where K is the total number of topics. A user may personally prefer some experts rather than others based upon the user's personal interest in their expertise. For example, Andy may prefer Bella in the topic of "Python programming", but have no opinion on Chris who may be a better Python developer. We denote the personal preferences of users towards experts in topics as a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{N \times M \times K}$ where element $\boldsymbol{\mathcal{X}}(i, j, k)$ is binary, representing whether a user u_i prefers an expert e_j in a topic t_k.

We define the *personalized expert recommendation problem* as: Given a set of users $\boldsymbol{L}$ with partially observed preferences denoted as a tensor $\boldsymbol{\mathcal{T}}$ on experts $\boldsymbol{E}$ over topics $\boldsymbol{T}$, our goal is to recommend the top-k relevant experts to a user u_i.

5.2.1 Basic Recommendation by Tensor Factorization

As a natural first step, we can tackle this problem with a basic recommendation framework using tensor factorization. Let $\boldsymbol{U}^{(1)} \in \mathbb{R}^{N \times R}$, $\boldsymbol{U}^{(2)} \in \mathbb{R}^{M \times R}$ and $\boldsymbol{U}^{(3)} \in \mathbb{R}^{K \times R}$ be latent factor matrices for users, experts, and topics, respectively, where $R \ll min(N, M, K)$ is the number of latent factors as the rank of a tensor. The basic tensor-based expert recommendation model can be defined as:

$$\begin{aligned} \underset{\boldsymbol{U}^{(n)}, \boldsymbol{\mathcal{X}}}{\text{minimize}} \quad & \frac{1}{2}\|\boldsymbol{\mathcal{X}} - [\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]\|_F^2 + \frac{\lambda}{2}\sum_{n=1}^{3}\|\boldsymbol{U}^{(n)}\|_F^2, \\ \text{subject to} \quad & \boldsymbol{\Psi} * \boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{T}}, \boldsymbol{U}^{(n)} \geq 0, n = 1, 2, 3 \end{aligned} \tag{5.1}$$

where $\boldsymbol{\mathcal{X}}$ denotes the complete preferences of users towards experts across topics, $\boldsymbol{\mathcal{T}}$ denotes the observed user's preferences on experts, $\|\boldsymbol{U}^{(n)}\|_F^2$ is a Tikhonov regularization

term used to avoid overfitting and provide a unique solution, $\circ$ is a Hadamard product operator and $\mathbf{\Psi}$ is a non-negative weight tensor with the same size as $\mathcal{X}$ with $\mathbf{\Psi}(i,j,k) = 1$ indicating that we observe the selection of user u_i on expert e_j in a topic t_k, $\mathbf{\Psi}(i,j,k) = 0$ otherwise. This basic model estimates $\hat{\boldsymbol{\mathcal{X}}}$ that approximates the original (unknown) $\mathcal{X}$ via learning optimal latent factor matrices $\{\boldsymbol{U}^{(n)}, n = 1,2,3\}$. For each user and topic of interest, this model can recommend a ranked list of personalized experts.

5.2.2 Research Challenges

While this basic tensor factorization framework provides a first step toward personalized expert recommendation, it leaves open many critical questions:

- First, since the user-expert-topic tensor is necessarily sparse (meaning that even with factorization the underlying latent factors may be of poor quality), can we augment this basic approach with additional contextual preferences for experts by considering the relationships among both homogeneous entities (e.g., users and users, experts and experts) and heterogeneous entities (e.g., users and experts, topics and experts)?

- How can we integrate these contextual preferences into a tensor-based personalized expert recommendation framework?

- How effective is the proposed tensor-based framework in comparison with other state-of-the-art baselines? And which contextual preferences have the most significant impact on the quality of personalized expert recommendations?

5.3 The TAPER Framework

We turn in this section toward constructing the contextual tensor factorization framework, as illustrated in Figure 5.2.

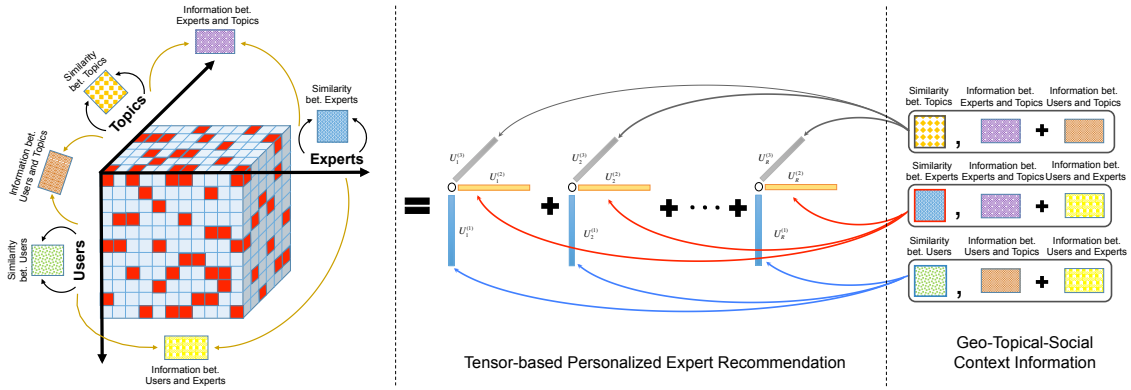


Figure 5.2: Overview of the proposed tensor-based personalized expert recommendation framework.

5.3.1 Evidence of Geo-Topical-Social Impact

We begin our investigation by examining the impact of three factors – geo-spatial, topical, and social context – on the observed preferences towards experts. Our goals in this section are to assess whether and to what degree these factors do affect how users select experts. Informed by these observations, we turn in the following to integrate them into the tensor-based factorization framework.

5.3.1.1 Geo-Tagged Twitter Lists

We adopt geo-tagged Twitter lists as evidence of the revealed preferences of users for other users. A Twitter list allows a user u_i to label another user u_j with an annotation (e.g., news, food, technology). In isolation these lists support the curation of an individual user's information stream, but in the aggregate the list labels can encode what a target user is "known-for". Many efforts have demonstrated that these labels can provide a crowdsourced expertise profile of the target user [60, 129, 131, 132]. Concretely, we use a geo-tagged Twitter list dataset containing over 12 million crowd-generated lists and 14 million geo-tagged list relationships between list creators and members. We filter the

Table 5.1: Dataset Summary

Data Type	Total Number of Records
Twitter Lists	11,322
Users (list creators)	10,559
Experts (list members)	8,417
List Relationships	117,187
Sparsity	0.13%

lists to only keep US-based users in topics: news, music, technology, celebrities, sports, business, politics, food, fashion, art, science, education, marketing, movie, photography, and health. The dataset is summarized in Table 5.1. Both list creators and list members are associated with GPS coordinates. For those without GPS coordinates, their locations can be estimated with their tweets by an approach previously used for check-ins and geo-tagged images [133]. In order to simplify our study, we only focus on users with GPS coordinates. We shall refer to list creators as *users* and members in the lists as *experts*.

5.3.1.2 Geo-Spatial Context

We begin by investigating the impact of distance on the experts selected by users. Figure 5.3(a) shows the cumulative distribution of the average distance between a user and the experts they have labeled, aggregated for eight different cities. In general, we see that users from different cities have different levels of locality. For example, users in San Francisco are more likely to select experts from a wider geographical scope than users based in Chicago. Specifically, almost 40% of users in Chicago have an average distance to their experts within 100 miles. However, only 14% of users in SF have an average distance within 100 miles. In a similar fashion, Figure 5.3(b) shows the cumulative distribution of the average distance between users and the experts they have labeled for seven different topics. For a fixed distance (e.g., 100 miles), the topic *food* has the largest probability. This implies that users interested in *food* are closer to their chosen experts while users

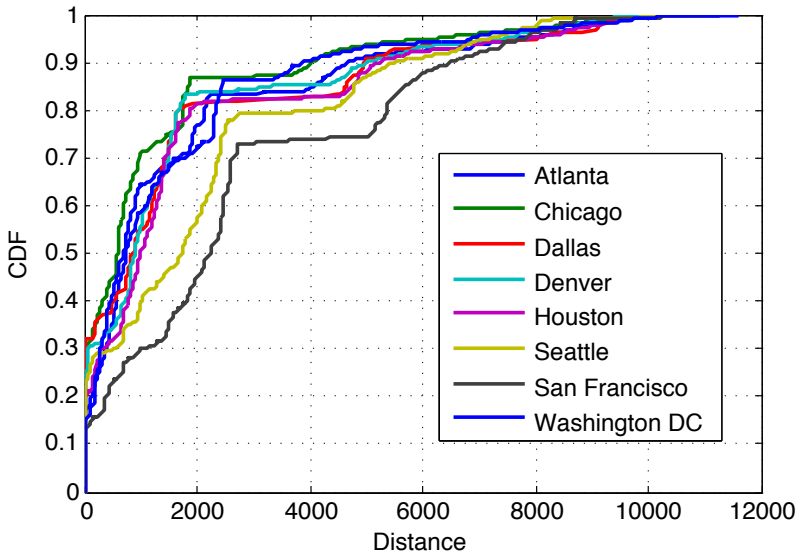
interested in a broad topic like *celebrity* instead select experts with a wider geographical scope. Hence, we can conclude that *the geo-spatial context of users, experts, and topics does indeed affect the preference for experts, and that these factors are impacted to varying degrees based on topic and on the particular locations of both users and experts*.

5.3.1.3 Topical Context

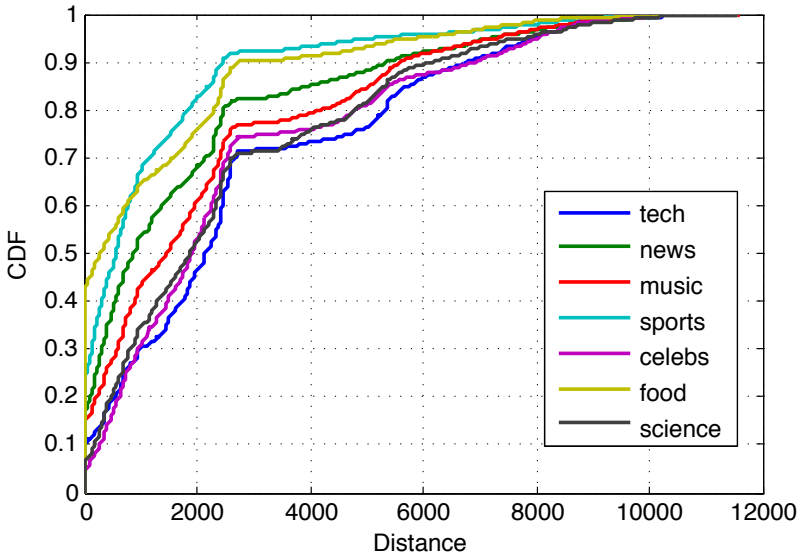
As we observed in Figure 5.3(b), different topics have different levels of locality. Here, we further investigate the impact of topical context on personalized preferences for experts. We begin by viewing each user as a vector of the experts they have selected, in which the element is 1 when the expert is listed by the user, otherwise 0. We then measure the impact of the number of shared topics between users (e.g., if two users have used three of the same topics in their lists, then we consider the number of shared topics between them to be three) on how similar are the users with respect to the experts they have selected. As we can see in Figure 5.4(a), users are more likely to select similar experts when they share more common topics. In other words, common topical interest impacts the choice of experts. In a similar fashion, we can view each expert as a vector of all users, in which the element is 1 when the expert is listed by the user, otherwise 0. We then measure the impact of the number of shared topics between experts (e.g., if two experts have been labeled by four of the same topics, then we consider the number of shared topics between them to be four) on how similar are the experts with respect to the users who have selected them. As we can see in Figure 5.4(b), experts who share more topics are more likely to be preferred by similar users. *We conclude that topical context has a strong impact on user preferences for experts.*

5.3.1.4 Social Context

Finally, the social connections among users and experts can be a strong indicator of shared topical interests, as well as an implicit signal that two users are more likely to be

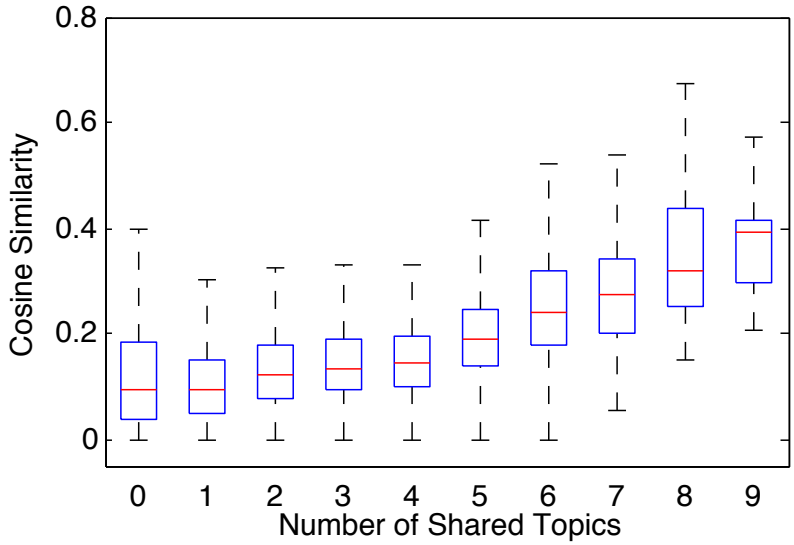


(a) Different Cities

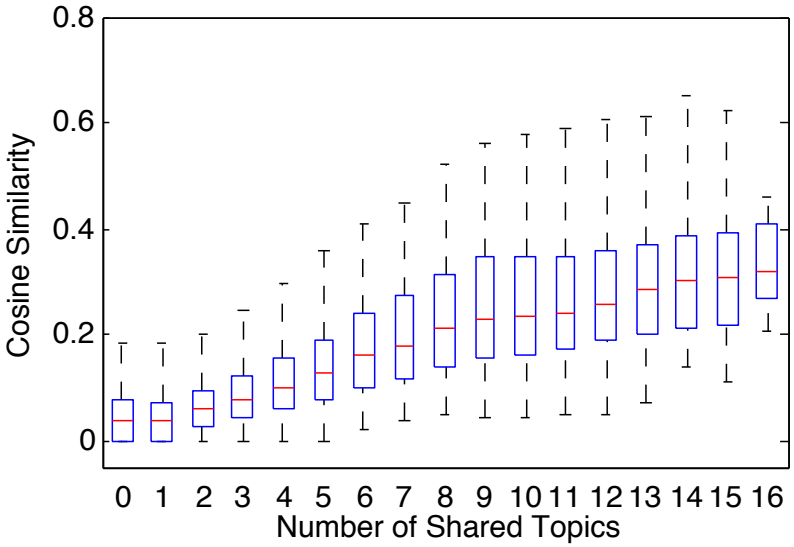


(b) Different Topics

Figure 5.3: The impact of distance on expert preferences by location (a) and by topic (b).



(a) Users-Users



(b) Experts-Experts

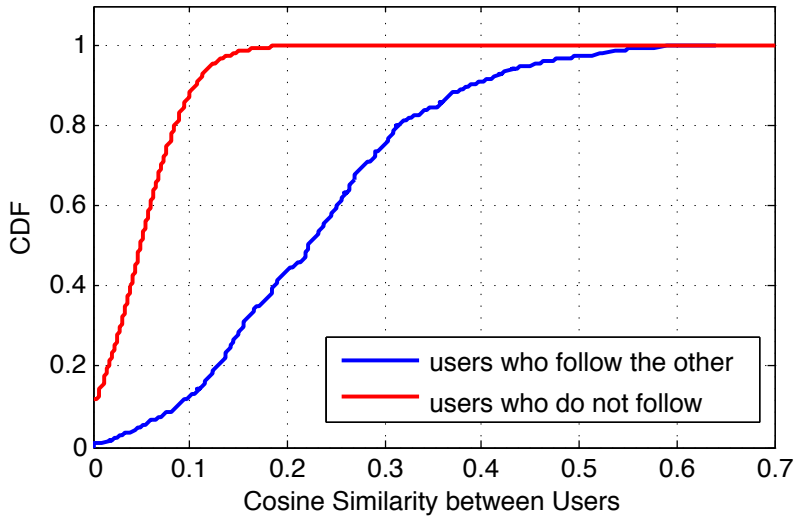
Figure 5.4: The relationship between the number of shared topics and the similarity between users (a) and experts (b).

near each other. Researchers have found that social ties increase the likelihood of two users being near each other [50]. Figure 5.5 shows the cumulative distributions of the similarity of users and experts, respectively. Using the same approach introduced above, each user is represented as a vector of all experts; each expert is represented as a vector of all users. The cosine similarity is employed to calculate the similarity of users and experts. We observe that users who follow the other generally have a larger similarity on selecting experts. For experts, we can see a similar pattern that those who follow the other are more likely to be selected by the same users. Through these observations, we can draw the conclusion that *social context strongly affects a user's preferences for experts.*

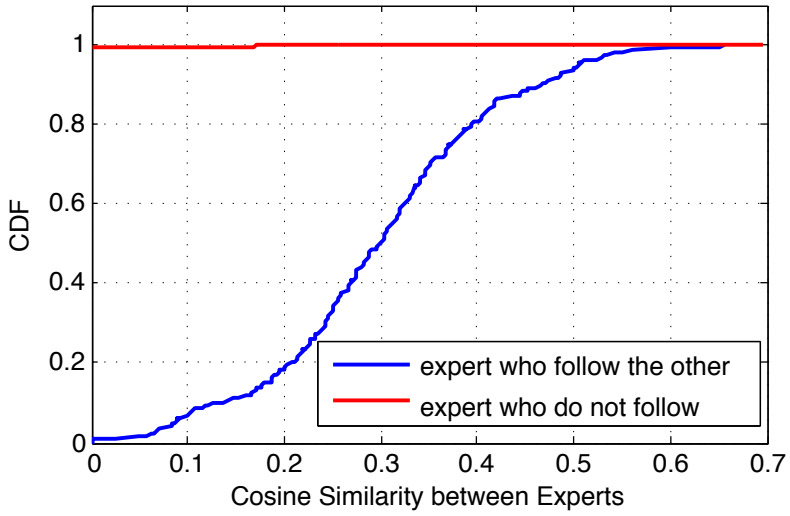
5.3.2 Integrating Contextual Preferences

Given this evidence of the significance of geo-topical-social context with respect to preferences on experts, a natural approach is to integrate them into the basic tensor-based expert recommendation framework as regularization terms. Intuitively, if two entities are similar, e.g., two users have similar preferences in recognizing experts across topics, the latent representations of these two entities should be similar. Hence, we can regulate latent representations of two similar entities to make them as close as possible. We denote $\boldsymbol{S}$ as a symmetric similarity matrix encoding contextual information between homogeneous entities (e.g., users and users, experts and experts), and then formally minimize the following loss function:

$$\begin{aligned}
\Theta &= \frac{1}{2}\sum_{i,j}\boldsymbol{S}(i,j)\|\boldsymbol{U}_i^{(n)} - \boldsymbol{U}_j^{(n)}\|^2 \\
&= \sum_i \boldsymbol{U}_i^{(n)}\boldsymbol{D}(i,i)\boldsymbol{U}_i^{(n)^T} - \sum_{i,j}\boldsymbol{U}_i^{(n)}\boldsymbol{S}(i,j)\boldsymbol{U}_j^{(n)^T} \\
&= \text{tr}(\boldsymbol{U}^{(n)^T}(\boldsymbol{D}-\boldsymbol{S})\boldsymbol{U}^{(n)}) \\
&= \text{tr}(\boldsymbol{U}^{(n)^T}\boldsymbol{\mathcal{L}}\boldsymbol{U}^{(n)})
\end{aligned}$$



(a) Users-Users



(b) Experts-Experts

Figure 5.5: The impact of social connections on selecting experts for users (a) and being selected for experts (b)

where $\boldsymbol{U}_i^{(n)}$ is the ith row of the factor matrix $\boldsymbol{U}^{(n)}$ for the nth-mode of a tensor $\boldsymbol{\mathcal{X}}, n \in \{1,2,3\}$, $tr(\cdot)$ is the matrix trace, $\boldsymbol{D}$ is a diagonal matrix with the element $\boldsymbol{D}(i,i) = \sum_j \boldsymbol{S}(i,j)$, and $\boldsymbol{\mathcal{L}} = \boldsymbol{D} - \boldsymbol{S}$ is the graph Laplacian of the similarity matrix $\boldsymbol{S}$. For the contextual information between heterogeneous entities (e.g., users and experts, topics and experts), we denote $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ as matrices encoding the contextual information between users and experts, users and topics and experts and topics, respectively. We then regulate them by directly adding regularization terms $\|\boldsymbol{A} - \boldsymbol{U}^{(1)}\boldsymbol{U}^{(2)^T}\|_F^2$, $\|\boldsymbol{B} - \boldsymbol{U}^{(1)}\boldsymbol{U}^{(3)^T}\|_F^2$, and $\|\boldsymbol{C} - \boldsymbol{U}^{(2)}\boldsymbol{U}^{(3)^T}\|_F^2$ into the basic framework. In order to simplify the parameter tuning, $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ are normalized to the same scale.

After integrating all regularization terms into Eq.(5.1), an open question is how can we efficiently estimate the solution to this optimization problem since there is no closed form solution? TAPER leverages ADMM to solve the following optimization problem:

$$\begin{aligned} \underset{\boldsymbol{U}^{(n)}, \boldsymbol{\mathcal{X}}}{\text{minimize}} \quad & \frac{1}{2}\|\boldsymbol{\mathcal{X}} - [\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]\|_F^2 + \frac{\lambda}{2}\sum_{n=1}^{3}\|\boldsymbol{U}^{(n)}\|_F^2 \\ & + \frac{\gamma}{2}\sum_{n=1}^{3}\mathrm{tr}(\boldsymbol{Z}^{(n)^T}\boldsymbol{\mathcal{L}}_n\boldsymbol{Z}^{(n)}) + \frac{\beta}{2}(\|\boldsymbol{A} - \boldsymbol{U}^{(1)}\boldsymbol{U}^{(2)^T}\|_F^2 \\ & + \|\boldsymbol{B} - \boldsymbol{U}^{(1)}\boldsymbol{U}^{(3)^T}\|_F^2 + \|\boldsymbol{C} - \boldsymbol{U}^{(2)}\boldsymbol{U}^{(3)^T}\|_F^2), \\ \text{subject to} \quad & \boldsymbol{\Psi} * \boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{T}}, \boldsymbol{U}^{(n)} = \boldsymbol{Z}^{(n)} \geq 0, n = 1,2,3 \end{aligned} \tag{5.2}$$

where γ controls the weight of the contextual information between homogeneous entities (e.g., users and users, experts and experts), and β controls the weight of the contextual information between heterogeneous entities (e.g., users and experts, topics and experts). But how specifically should we model these contextual preferences for integration into the tensor-based framework? In the following, we consider geo-spatial, topical, and social preferences in turn.

5.3.3 Modeling Geo-Spatial Preferences

We first aim to model geo-spatial context for integration into the tensor-based factorization approach.

5.3.3.1 Relationships between Homogeneous Entities

Supported by the data-driven observations of the previous section and following Tobler's First Law of Geography [122] – which asserts that near things are more related than distant things – we propose to model user preferences as a function of distance. Concretely, we consider three graphs over homogeneous entities $\boldsymbol{\mathcal{G}}_G^{uu}$, $\boldsymbol{\mathcal{G}}_G^{ee}$ and $\boldsymbol{\mathcal{G}}_G^{tt}$ for users, experts, and topics, respectively. For each graph, we treat "nearby" nodes as more alike if they are geographically closer. In this way, preferences on experts may be propagated to these nearby nodes in the tensor-based factorization approach.

To begin, consider the user-user graph $\boldsymbol{\mathcal{G}}_G^{uu}$, where nodes are users and edges capture the affinity between users. We can define the adjacency matrix $\boldsymbol{H}_G \in \mathbb{R}^{N \times N}$ as:

$$\boldsymbol{H}_G(u_i, u_j) = exp(-\frac{Dist(u_i, u_j)^2}{2\alpha^2}),$$

where α is a decay constant (which we experimentally set to 20 miles) and $Dist$ is to measure the geographic distance between two users by using the Haversine formula. The affinity $\boldsymbol{H}_G(u_i, u_j)$ approaches one as two users are nearer each other.

Similarly, the expert-expert graph $\boldsymbol{\mathcal{G}}_G^{ee}$ can be constructed, where nodes are experts and here edges represent the spatial similarity of pairs of experts. Rather than purely measuring the geographic nearness of two experts (which does not take into account the recognizability of experts in various locations), we consider how the users who have labeled those experts are distributed. The intuition is that experts who are preferred have higher popularity in the location of a user. Let $l(u_i) \in \mathbb{L}$ be a location of user u_i and $\phi_{e_i}^{l(u_i)}$

as the spatial popularity be the number of users selecting expert e_i in the location of user u_i across topics. The adjacency matrix $\boldsymbol{V}_G \in \mathbb{R}^{M \times M}$ can be defined as:

$$\boldsymbol{V}_G(e_i, e_j) = exp(-\frac{Dist(l(e_i), l(e_j))^2}{2\alpha^2} * \sum_{u_i \in U} (\phi_{e_i}^{l(u_i)} - \phi_{e_j}^{l(u_i)})^2),$$

where the first part in the exponential is to calculate the deviation of geo-spatial preferences between experts (so that experts who are nearby are considered "closer"), the second part is to consider the difference of their spatial popularities over all locations of users (to capture the nearness in terms of who is interested in them). In this case, we discretize the continental U.S. surface with a 1° by 1° geodesic grid, so we can map the location of a user (GPS coordinate) to a discrete region.

The third graph – the topic-topic graph $\boldsymbol{\mathcal{G}}_G^{tt}$ – has topics as nodes and edges that represent the geo-spatial correlation between topics. Intuitively, selecting experts could be impacted by the choice of topic. The spatial preference in the topic *food* is much more local than the topic *technology*. We aim to capture the similarity of spatial preference between topics. Let $d_{u_i}^{t_i}$ be the average distance between user u_i and a set of experts $\boldsymbol{E}_{u_i}^{t_i}$ he/she recognizes in a topic t_i. The empirical distribution of spatial preference for a topic t_i can be obtained from calculations $\{d_{u_i}^{t_i}, u_i \in \boldsymbol{U}\}$, which is denoted as Υ_{t_i}. We then apply the Kullback-Leibler divergence $D_{KL}(\cdot\|\cdot)$ to measure the closeness of empirical distributions between topics. Hence, we can define the adjacency matrix $\boldsymbol{W}_G \in \mathbb{R}^{K \times K}$ as:

$$\boldsymbol{W}_G(t_i, t_j) = 1 - D_{KL}(\Upsilon_{t_i} \| \Upsilon_{t_j}),$$

where $\boldsymbol{W}_G$ will approach to 1 if two topics have similar empirical distributions of spatial preferences.

5.3.3.2 *Relationships between Heterogeneous Entities*

In addition to the geo-spatial correlations between users, experts, and topics themselves, we can additionally consider the relationships across heterogeneous entities. Specifically, we consider the relationships between users and experts, as well as between users and topics.

First, we propose to leverage the geo-spatial preferences between users and experts so that users are more likely to select experts who have a high popularity in the location of this user. The intuition is that the local popularity of an expert can be considered as a prior on what a user would prefer. For example, given a new user in Seattle with no expert preferences, we can default to a locally popular expert like Jeff Bezos. Concretely, we propose to improve the learning of latent matrices of users and experts as:

$$\mathcal{F}_{\boldsymbol{A}_G} = \|\boldsymbol{A}_G - \boldsymbol{U}^{(1)}\boldsymbol{U}^{(2)^T}\|_F^2,$$

where $\boldsymbol{A}_G$ is the adjacency matrix in which an element indicates the spatial popularity of an expert in the location of a user. By minimizing $\mathcal{F}_{\boldsymbol{A}_G}$, the recommender will prefer locally popular experts.

Similarly, the geo-spatial preferences between users and topics can also be leveraged so that a user is more likely to select experts on topics that have high popularity in the location of this user. Formally, we have the latent matrices of users and topics as:

$$\mathcal{F}_{\boldsymbol{B}_G} = \|\boldsymbol{B}_G - \boldsymbol{U}^{(1)}\boldsymbol{U}^{(3)^T}\|_F^2,$$

where $\boldsymbol{B}_G$ is the adjacency matrix in which an element indicates the spatial popularity of a topic in the location of a user. Again, by minimizing $\mathcal{F}_{\boldsymbol{B}_G}$, the recommender will prefer experts on topics that are more popular locally.

5.3.4 Modeling Topical Preferences

Beyond geo-spatial preferences, we next turn to models of topical preference. As we have observed in Section 5.3.1.3, the topical context can influence a user's preference for experts. Intuitively, users who have similar interests on topics tend to have similar preferences on experts.

5.3.4.1 Relationships between Homogeneous Entities

We begin by considering two graphs – one for the user-user graph $\boldsymbol{\mathcal{G}}_T^{uu}$ and the other for the expert-expert graph $\boldsymbol{\mathcal{G}}_T^{ee}$ in terms of topical preference.

In this case, the user-user graph $\boldsymbol{\mathcal{G}}_T^{uu}$ has nodes that represent users but now edges represent the similarity between users in terms of topical preferences (rather than in terms of distance as in the geo-spatial case in the previous section). Inspired by the work [72], we define the adjacency matrix $\boldsymbol{H}_T$ as:

$$\boldsymbol{H}_T(u_i, u_j) = \frac{|\boldsymbol{T}_{u_i} \bigcap \boldsymbol{T}_{u_j}|}{|\boldsymbol{T}_{u_i} \bigcup \boldsymbol{T}_{u_j}|} * exp(\sum_{t \in T}(o_{u_i}^t - o_{u_j}^t)P_t),$$

where $\boldsymbol{T}_{u_i}$ is the set of topics a user u_i is interested in by labeling the lists with keywords related to these topics, $o_{u_i}^t$ denotes the number of experts a user u_i labeled in a topic t, and P_t denotes the probability of being interested in a topic t, which is formally defined as:

$$P_t = \frac{\sum_{u \in U} o_u^t}{\sum_{t \in T} \sum_{u \in U} o_u^t}. \tag{5.3}$$

The first part in $\boldsymbol{H}_T$ is used to measure how common two users share topics; the second part is applied to quantitively evaluate the similarity of their interests across all topics.

For the second graph – the expert-expert graph $\boldsymbol{\mathcal{G}}_T^{ee}$ – we have nodes as experts and edges representing the similarity of experts towards their expertise. The adjacency matrix

$\boldsymbol{V}_T$ is defined as:

$$\boldsymbol{V}_T(e_i, e_j) = \frac{|\boldsymbol{T}_{e_i} \bigcap \boldsymbol{T}_{e_j}|}{|\boldsymbol{T}_{e_i} \bigcup \boldsymbol{T}_{e_j}|} * exp(\sum_{t \in T} (\delta^t_{e_i} - \delta^t_{e_j}) P_t),$$

where $\boldsymbol{T}_{e_i}$ is the set of topics in which an expert e_i have expertise labeled by users, and $\delta^t_{e_i}$ is denoted as the number of times an expert e_i labeled by users in a topic t. The first part in $\boldsymbol{V}_T$ is used to measure how common two experts have the same expertise; the second part is applied to quantitively evaluate the similarity of their expertise across all topics.

5.3.4.2 Relationships between Heterogeneous Entities

In addition to the topical correlations between users and experts, themselves, we additionally consider the relationships across users and topics, as well as across experts and topics. Since each Twitter list is labeled with certain labels, the topic preferences of a user can be revealed by aggregating all of the labels he/she applies in the lists. As a result, we propose to leverage a user's topical preferences to improve the learning of latent matrices of users and topics as follows:

$$\mathcal{F}_{\boldsymbol{B}_T} = \|\boldsymbol{B}_T - \boldsymbol{U}^{(1)} \boldsymbol{U}^{(3)^T}\|^2_F,$$

where $\boldsymbol{B}_T$ is the affinity matrix where an element indicates whether a user is interested in certain topic by applying keywords related to this topic in her lists. Our goal is to minimize $\mathcal{F}_{\boldsymbol{B}_T}$ so that a user is more likely to select experts who have expertise on topics of interest to this user.

Meanwhile, an expert's expertise can be also found by aggregating all of the labels in all the lists this expert appears. Therefore, we leverage the expert's topic preferences to

improve the learning of latent matrices of experts and topics as illustrated below:

$$\mathcal{F}_{\boldsymbol{C}_T} = \|\boldsymbol{C}_T - \boldsymbol{U}^{(2)}\boldsymbol{U}^{(3)^T}\|_F^2,$$

where $\boldsymbol{C}_T$ is the affinity matrix in which the element indicates the number of times that an expert has been recognized by users with respect to certain topics. Through minimizing $\mathcal{F}_{\boldsymbol{C}_T}$, an expert who has higher recognition in a topic is more likely to be selected by users who are interested in this topic.

5.3.5 Modeling Social Preferences

Finally, we consider how to model the social preferences of users with respect to personalized expert preferences. Concretely, we consider the social connections among users and experts:

5.3.5.1 Relationships between Homogeneous Entities

Similar to our previous efforts, we construct a graph $\boldsymbol{\mathcal{G}}_S^{uu}$ in which nodes represent users and edges represent the pairwise similarity of users in terms of their social preferences, and a graph $\boldsymbol{\mathcal{G}}_S^{ee}$ in which nodes represent experts and edges represent the pairwise similarity of experts with respect to their social connections. By applying the Jaccard coefficient, the adjacency matrix $\boldsymbol{H}_S$ for users is defined as:

$$\boldsymbol{H}_S(u_i, u_j) = \frac{|\boldsymbol{F}_{u_i} \bigcap \boldsymbol{F}_{u_j}|}{|\boldsymbol{F}_{u_i} \bigcup \boldsymbol{F}_{u_j}|},$$

where $\boldsymbol{F}_{u_i}$ represents the set of users u_i follows. As a similar fashion, we define the adjacency matrix $\boldsymbol{V}_S$ for experts as:

$$\boldsymbol{V}_S(e_i, e_j) = \frac{|\boldsymbol{F}_{e_i} \bigcap \boldsymbol{F}_{e_j}|}{|\boldsymbol{F}_{e_i} \bigcup \boldsymbol{F}_{e_j}|},$$

where $\boldsymbol{F}_{e_i}$ represents the set of users e_i follows. The intuition behind $\boldsymbol{H}_S$ and $\boldsymbol{V}_S$ is that users/experts who share more friends are more likely have similar behaviors on selecting experts/being selected by users.

5.3.5.2 Relationships between Heterogeneous Entities

In order to take advantage of the social connections between different entities (e.g., user-expert) into the expert recommendation, we leverage the following social relationships from a user to an expert to improve the learning of latent matrices of users and experts, as shown below:

$$\mathcal{F}_{\boldsymbol{A}_S} = \|\boldsymbol{A}_S - \boldsymbol{U}^{(1)}\boldsymbol{U}^{(2)^T}\|_F^2, \tag{5.4}$$

where $\boldsymbol{A}_S$ is a matrix where the element indicates if a user follows an expert. In this way, the recommender can prefer experts who are followed by this user through minimizing $\mathcal{F}_{\boldsymbol{A}_S}$. Note that this is a very strong signal, and unlikely to be present for most users.

5.3.6 Solving TAPER

Given these models of geo-spatial, topical and social preferences above, we now introduce an efficient optimization algorithm to solve Eq. (5.2) under the framework of ADMM that can be considered as an approximation of the method of multipliers with more efficiency. The overall approach is summarized in **Algorithm 2**. Our objective function can

Algorithm 2: Optimization Algorithm of the Proposed Model via ADMM

Input: $\mathcal{T}, \boldsymbol{U}_0^{(n)}, \boldsymbol{\Psi}, \boldsymbol{\Psi}^c, \gamma, \lambda, \rho, \eta, \eta_{max}, N$
Output: $\mathcal{X}, \boldsymbol{U}^{(n)}, \boldsymbol{Z}^{(n)}, \boldsymbol{Y}^{(n)}$

Initialize $\boldsymbol{U}_0^{(n)} \geq 0$, $\boldsymbol{Z}_0^{(n)} = \boldsymbol{Y}_0^{(n)} = 0, t = 0$
while Not Converged **do**
  for $n \leftarrow 1$ ***to*** N **do**
    Update $\boldsymbol{Z}_{t+1}^{(i)} \leftarrow (\boldsymbol{Y}_t^{(i)} - \eta_t \boldsymbol{U}_t^{(i)})(\eta_t \boldsymbol{I} + \gamma \mathcal{L}_i)^{-1}$
    Calculate $\boldsymbol{E}^{(n)} = (\boldsymbol{U}^{(N)} \odot \cdots \boldsymbol{U}^{(n+1)} \odot \boldsymbol{U}^{(n-1)} \odot \cdots \boldsymbol{U}^{(1)})^T$
    Update $\boldsymbol{U}_{t+1}^{(1)} \leftarrow (\boldsymbol{X}_{(1)}^t \boldsymbol{E}^{(1)^T} + \beta \boldsymbol{A}\boldsymbol{U}^{(2)} + \beta \boldsymbol{B}\boldsymbol{U}^{(3)} + \eta_t \boldsymbol{Z}_{t+1}^{(1)} + \boldsymbol{Y}_t^{(1)})(\boldsymbol{E}^{(2)}\boldsymbol{E}^{(2)^T} + \beta \boldsymbol{U}^{(1)^T}\boldsymbol{U}^{(1)} + \beta \boldsymbol{U}^{(3)^T}\boldsymbol{U}^{(3)} + \lambda \boldsymbol{I} + \eta_t \boldsymbol{I})^{-1}$
    Update $\boldsymbol{U}_{t+1}^{(2)} \leftarrow (\boldsymbol{X}_{(2)}^t \boldsymbol{E}^{(2)^T} + \beta \boldsymbol{A}^T\boldsymbol{U}^{(1)} + \beta \boldsymbol{C}\boldsymbol{U}^{(3)} + \eta_t \boldsymbol{Z}_{t+1}^{(2)} + \boldsymbol{Y}_t^{(2)})(\boldsymbol{E}^{(2)}\boldsymbol{E}^{(2)^T} + \beta \boldsymbol{U}^{(1)^T}\boldsymbol{U}^{(1)} + \beta \boldsymbol{U}^{(3)^T}\boldsymbol{U}^{(3)} + \lambda \boldsymbol{I} + \eta_t \boldsymbol{I})^{-1}$
    Update $\boldsymbol{U}_{t+1}^{(3)} \leftarrow (\boldsymbol{X}_{(3)}^t \boldsymbol{E}^{(3)^T} + \beta \boldsymbol{B}_T\boldsymbol{U}^{(1)} + \beta \boldsymbol{C}^T\boldsymbol{U}^{(2)} + \eta_t \boldsymbol{Z}_{t+1}^{(3)} + \boldsymbol{Y}_t^{(3)})(\boldsymbol{E}^{(3)}\boldsymbol{E}^{(3)^T} + \beta \boldsymbol{U}^{(1)^T}\boldsymbol{U}^{(1)} + \beta \boldsymbol{U}^{(2)^T}\boldsymbol{U}^{(2)} + \lambda \boldsymbol{I} + \eta_t \boldsymbol{I})^{-1}$
  Update $\mathcal{X}_{t+1} = \mathcal{T} + \boldsymbol{\Psi}^c * [\![\boldsymbol{U}_{t+1}^{(1)}, \boldsymbol{U}_{t+1}^{(2)}, \ldots, \boldsymbol{U}_{t+1}^{(N)}]\!]$
  for $n \leftarrow 1$ ***to*** N **do**
    Update $\boldsymbol{Y}_{t+1}^{(n)} = \boldsymbol{Y}_t^{(n)} + \eta_t(\boldsymbol{Z}_{t+1}^{(n)} - \boldsymbol{U}_{t+1}^{(n)})$
  Update $\eta_{t+1} = \min(\rho\eta_t, \eta_{max})$
  Check the convergence: $\max\{\|\boldsymbol{U}_{t+1}^{(n)} - \boldsymbol{Z}_{t+1}^{(n)}\|_F, n = 1, 2, \ldots, N\} < tol$
  $t = t + 1$
return $\mathcal{X}$, $\boldsymbol{U}^{(n)}, n = 1, 2, \ldots, N$

be firstly written in the partial augmented Lagrangian function as follows:

$$
\begin{aligned}
L_\eta(\boldsymbol{U}^{(n)}, \boldsymbol{Z}^{(n)}, \boldsymbol{Y}^{(n)})_{n=1,2,3} = & \frac{1}{2}\|\mathcal{X} - [\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]\|_F^2 \\
& + \frac{\lambda}{2}\sum_{i=1}^{3}\|\boldsymbol{U}^{(i)}\|_F^2 + \frac{\gamma}{2}\sum_{i=1}^{3}\mathrm{tr}(\boldsymbol{Z}^{(i)^T}\mathcal{L}_i\boldsymbol{Z}^{(i)}) \\
& + \frac{\beta}{2}(\|\boldsymbol{A} - \boldsymbol{U}^{(1)}\boldsymbol{U}^{(2)^T}\|_F^2 + \|\boldsymbol{B} - \boldsymbol{U}^{(1)}\boldsymbol{U}^{(3)^T}\|_F^2 + \|\boldsymbol{C} - \boldsymbol{U}^{(2)}\boldsymbol{U}^{(3)^T}\|_F^2) \\
& + \sum_{i=1}^{3} < \boldsymbol{Y}^{(i)}, \boldsymbol{Z}^{(i)} - \boldsymbol{U}^{(i)} > + \sum_{i=1}^{3}\frac{\eta}{2}\|\boldsymbol{Z}^{(i)} - \boldsymbol{U}^{(i)}\|_F^2,
\end{aligned}
\tag{5.5}
$$

where $\boldsymbol{Y}^{(n)}$ is the matrix of Lagrange multipliers for $n = 1, 2, \ldots, N$ and η is a penalty parameter.

Updating $\boldsymbol{Z}^{(i)}$. To update $\boldsymbol{Z}^{(i)}$, we can rewrite the Eq. (5.5) as:

$$\min_{\boldsymbol{Z}^{(i)}} \quad \frac{\gamma}{2}\mathrm{tr}(\boldsymbol{Z}^{(i)^T}\boldsymbol{\mathcal{L}}_i\boldsymbol{Z}^{(i)}) + \frac{\eta_t}{2}\|\boldsymbol{Z}^{(i)} - \boldsymbol{U}_t^{(i)} + \frac{\boldsymbol{Y}_t^{(i)}}{\eta_t}\|_F^2. \tag{5.6}$$

Thus, $\boldsymbol{Z}^{(i)}$ can be efficiently updated by solving the optimization problem in Eq. (5.6):

$$\boldsymbol{Z}_{t+1}^{(i)} = (\boldsymbol{Y}_t^{(i)} - \eta_t\boldsymbol{U}_t^{(i)})(\eta_t\boldsymbol{I} + \gamma\boldsymbol{\mathcal{L}}_i)^{-1}, \tag{5.7}$$

where $\boldsymbol{I}$ is the identity matrix with the same size of $\boldsymbol{\mathcal{L}}_i$.

Updating $\boldsymbol{U}^{(i)}$. To update $\boldsymbol{U}^{(i)}$, the Eq. (5.5) can be rewritten as:

$$\begin{aligned}\min_{\boldsymbol{U}^{(i)}} \quad & \frac{1}{2}\|\boldsymbol{X}_{(i)}^t - \boldsymbol{U}^{(i)}\boldsymbol{E}^{(i)}\|_F^2 + \frac{\lambda}{2}\|\boldsymbol{U}^{(i)}\|_F^2 \\ & + \frac{\beta}{2}(\|\boldsymbol{A} - \boldsymbol{U}^{(1)}\boldsymbol{U}^{(2)^T}\|_F^2 + \|\boldsymbol{B} - \boldsymbol{U}^{(1)}\boldsymbol{U}^{(3)^T}\|_F^2 \\ & + \|\boldsymbol{C} - \boldsymbol{U}^{(2)}\boldsymbol{U}^{(3)^T}\|_F^2) + \frac{\eta_t}{2}\|\boldsymbol{Z}_i - \boldsymbol{U}_t^{(i)} + \frac{\boldsymbol{Y}_t^{(i)}}{\eta_t}\|_F^2,\end{aligned} \tag{5.8}$$

where $\boldsymbol{E}^{(i)} = (\boldsymbol{U}^{(N)} \odot \cdots \boldsymbol{U}^{(i+1)} \odot \boldsymbol{U}^{(i-1)} \odot \cdots \boldsymbol{U}^{(1)})^T|_{N=3}$, let $\boldsymbol{X}_{(n)}$ be the mode-n unfolding of the tensor $\boldsymbol{\mathcal{X}}$. Then this subproblem in terms of $\boldsymbol{U}^{(i)}$ is solved as follows:

$$\begin{aligned}\boldsymbol{U}_{t+1}^{(1)} = (\boldsymbol{X}_{(1)}^t\boldsymbol{E}^{(1)^T} + \beta\boldsymbol{A}\boldsymbol{U}^{(2)} + \beta\boldsymbol{B}\boldsymbol{U}^{(3)} + \eta_t\boldsymbol{Z}_{t+1}^{(1)} + \boldsymbol{Y}_t^{(1)}) \\ (\boldsymbol{E}^{(1)}\boldsymbol{E}^{(1)^T} + \beta\boldsymbol{U}^{(2)^T}\boldsymbol{U}^{(2)} + \beta\boldsymbol{U}^{(3)^T}\boldsymbol{U}^{(3)} + \lambda\boldsymbol{I} + \eta_t\boldsymbol{I})^{-1},\end{aligned} \tag{5.9}$$

$$\begin{aligned}\boldsymbol{U}_{t+1}^{(2)} = (\boldsymbol{X}_{(2)}^t\boldsymbol{E}^{(2)^T} + \beta\boldsymbol{A}^T\boldsymbol{U}^{(1)} + \beta\boldsymbol{C}\boldsymbol{U}^{(3)} + \eta_t\boldsymbol{Z}_{t+1}^{(2)} + \boldsymbol{Y}_t^{(2)}) \\ (\boldsymbol{E}^{(2)}\boldsymbol{E}^{(2)^T} + \beta\boldsymbol{U}^{(1)^T}\boldsymbol{U}^{(1)} + \beta\boldsymbol{U}^{(3)^T}\boldsymbol{U}^{(3)} + \lambda\boldsymbol{I} + \eta_t\boldsymbol{I})^{-1},\end{aligned} \tag{5.10}$$

$$\boldsymbol{U}_{t+1}^{(3)} = (\boldsymbol{X}_{(3)}^{t}\boldsymbol{E}^{(3)^T} + \beta\boldsymbol{B}_T\boldsymbol{U}^{(1)} + \beta\boldsymbol{C}^T\boldsymbol{U}^{(2)} + \eta_t\boldsymbol{Z}_{t+1}^{(3)} + \boldsymbol{Y}_t^{(3)}) \\ (\boldsymbol{E}^{(3)}\boldsymbol{E}^{(3)^T} + \beta\boldsymbol{U}^{(1)^T}\boldsymbol{U}^{(1)} + \beta\boldsymbol{U}^{(2)^T}\boldsymbol{U}^{(2)} + \lambda\boldsymbol{I} + \eta_t\boldsymbol{I})^{-1}. \quad (5.11)$$

Updating $\mathcal{X}$. To update $\mathcal{X}$, we can have that:

$$\mathcal{X}_{t+1} = \mathcal{T} + \boldsymbol{\Psi}^c[\![\boldsymbol{U}_{t+1}^{(1)}, \boldsymbol{U}_{t+1}^{(2)}, \boldsymbol{U}_{t+1}^{(3)}]\!],$$

where $\boldsymbol{\Psi}^c$ is the complement of $\boldsymbol{\Psi}$.

Updating $\boldsymbol{Y}^{(i)}$. To update $Y^{(i)}$, we can have that:

$$\boldsymbol{Y}_{t+1}^{(i)} = \boldsymbol{Y}_t^{(i)} + \eta_t(\boldsymbol{Z}_{t+1}^{(i)} - \boldsymbol{U}_{t+1}^{(i)}).$$

Updating η. We can accelerate our optimization algorithm by adaptively updating η. To update η, we can have that:

$$\eta_{t+1} = \min(\rho\eta_t, \eta_{max}),$$

where ρ is a constant to control the step. In this study, we empirically set $\eta_0 = 1.1$ and $\rho = 0.1$ via cross-validation.

5.4 Experiments

In this section, we investigate (i) the effectiveness of TAPER versus alternatives; (ii) the impact of specific contextual factors (e.g., geo-spatial, topical, and social); (iii) how incorporating evidence of experts for whom a user is *not interested* affects recommendation quality; and (iv) the impact of the amount of training data.

5.4.1 Experimental Setup

To evaluate the performance of the proposed framework, we randomly split experts for a user into 50% for training and 50% for testing. For latent factor dimension, we empirically choose 20 for all methods after testing various settings {5, 10, 20, 30, 40, 50, 100} for a tradeoff between accuracy and the computational cost. For the number of experts that a user does not pick up in the Twitter lists, we empirically select 350 through all experiments we conducted. The effects of this number will be further discussed in Section 5.4.3.2. Three positive parameters are involved in the experiments: λ, γ and β in Eq.(4.1). λ is the regularization parameter used to avoid overfitting. γ is to control the contribution of contextual information between homogeneous entities. β is to control the contribution of contextual information between heterogeneous entities. As a common way, we employ the cross-validation to tune these parameters. We find that parameter settings of $\gamma < 1$ and $\beta < 0.1$ lead to fairly stable precision and recall, indicating the stability of TAPER to these regularization parameters. Concretely, we empirically set $\lambda = 0.1$, $\gamma = 0.1$ and $\beta = 0.01$ for general experiments, respectively. Their effects on the performance of the proposed framework is evaluated and discussed in the Section 5.4.4.

We adopt Precision@k and Recall@k as our evaluation metrics. Precision@k represents the percentage of correctly recommended experts out of the top-k recommendations; Recall@k represents the percentage of experts emerging in the top-k recommendations. Both of them have been widely used to evaluate the quality of recommendation. In our experiments, we test for k at 5, 10, and 15.

5.4.2 Baselines

We consider seven baselines in addition to the proposed TAPER approach. The first three baseline are classical recommender system approaches:

- *Most Popular (**MP**)*: This baseline recommends the most listed experts in a topic to

all users.

- *User-based Collaborative Filtering (**UCF**)*: We adopt a user-based recommendation framework [134] to recommend personal experts, which discovers user's implicit preferences towards experts by aggregating similar users.

- *Matrix Factorization (**MF**)*: This baseline computes a user's preferences on experts for each topic by a pair-wise latent matrix factorization model trained by stochastic gradient descent [67].

The next four baselines are simplified variants of the proposed TAPER approach, all building on tensor factorization:

- *Tensor Factorization (**TF**)*: In analogy to matrix factorization, tensor factorization computes each user's preferences for experts by considering users, experts, and topics simultaneously. This basic tensor factorization model corresponds to Eq. (5.1).

- *Geo-based TAPER (**G-TAPER**)*: This model is a variant of the basic tensor factorization model, but only integrates geo-spatial preferences: that is, $\boldsymbol{H}_G$, $\boldsymbol{V}_G$, $\boldsymbol{W}_G$, $\boldsymbol{A}_G$ and $\boldsymbol{B}_G$ in Eq. (4.1).

- *Topical-based TAPER (**T-TAPER**)*: This model only integrates topical preferences: that is, $\boldsymbol{H}_T$, $\boldsymbol{V}_T$, $\boldsymbol{B}_T$ and $\boldsymbol{C}_T$ in Eq. (4.1).

- *Social-based TAPER (**S-TAPER**)*: This variant only considers social preferences: that is, $\boldsymbol{H}_S$, $\boldsymbol{V}_S$ and $\boldsymbol{A}_S$ in Eq. (4.1).

Finally, we consider the proposed TAPER approach:

- *Contextual Personalized Expert Recommendation (**TAPER**)*: This is the proposed framework, incorporating all three types of contextual information among users,

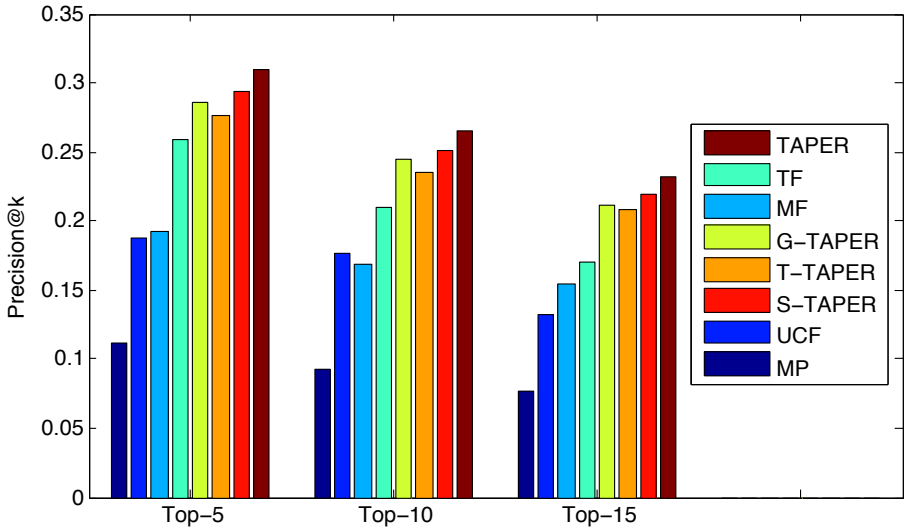


Figure 5.6: Precision@k: Comparing TAPER versus Alternative Methods.

experts, and topics in Eq. (4.1). Specifically, we let $\boldsymbol{H} = \boldsymbol{H}_G * \boldsymbol{H}_T * \boldsymbol{H}_S$, $\boldsymbol{V} = \boldsymbol{V}_G * \boldsymbol{V}_T * \boldsymbol{V}_S$, $\boldsymbol{W} = \boldsymbol{W}_G$, $\boldsymbol{A} = \boldsymbol{A}_G + \boldsymbol{A}_S$, $\boldsymbol{B} = \boldsymbol{B}_G + \boldsymbol{B}_T$ and $\boldsymbol{C} = \boldsymbol{C}_T$.

5.4.3 Results

We begin by investigating the quality of TAPER versus each of the baselines. We adopt 10-fold cross validation and report the average precision and recall over 10 test runs in Figure 5.6 and Figure 5.7. Overall, the proposed personal expert recommendation framework TAPER performs the best among all alternative baseline methods in both precision and recall. From Figures 5.6 and 5.7, we can observe that TAPER consistently outperforms the baseline methods MP, UCF, MF, and TF with an average improvement of 42.2% over the best of these four methods in precision and 33.5% in recall. Concretely, TAPER performs better than TF with an average improvement of 14.1% in precision and 17.8% in recall, indicating the superiority of a tensor factorization model integrating rich contextual preferences. Moreover, TAPER has a better performance than MF with an average improvement of 26.6% in precision and 25.8% in recall, which is significantly higher than a

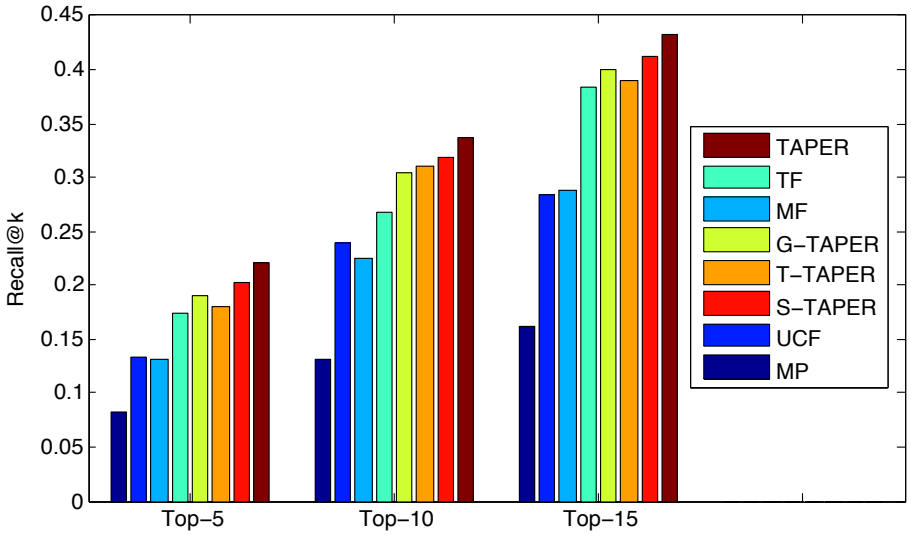


Figure 5.7: Recall@k: Comparing TAPER versus Alternative Methods.

related matrix factorization approach in [67].

We observe that MP and UCF perform the worst among all approaches. We attribute the poor performance of the UCF approach to sparseness – the low density of data can lead to poor recommendations, whereas both MF and TF can leverage the low-rank approximation of user preferences towards experts. Overall, we observe that the best method achieves a precision of around 0.3 and a recall of around 0.4 As Ye et al. [135] have observed, the effectiveness of recommenders with sparse datasets is usually low.

We also observe that adding additional contextual factors improves the basic tensor factorization (TF) approach. Concretely, G-TAPER gives an average improvement of 8.1% in precision and 7.3% in recall over TF. This indicates that the geo-spatial preferences among users, experts, and topics can help identify similar users and distinguish popular experts according to their spatial popularity. T-TAPER performs slightly better than TF with an average improvement of 5.2% in precision and 4.9% in recall, implying that the topical preferences among users and experts can help improve the performance of personalized

Table 5.2: What Impact Does Contextual Preference Have on Each Approach? Here we compare contextual preferences of heterogeneous entities versus homogeneous entities.

Metric	Scenario	G-TAPER-Het	G-TAPER-Hom	T-TAPER-Het	T-TAPER-Hom
Precision	Top-5	0.271	0.282	0.262	0.274
	Top-10	0.234	0.242	0.227	0.232
	Top-15	0.204	0.208	0.192	0.199
	Scenario	S-TAPER-Het	S-TAPER-Hom	TAPER-Het	TAPER-Hom
	Top-5	0.289	0.278	0.296	0.306
	Top-10	0.248	0.241	0.254	0.260
	Top-15	0.214	0.201	0.219	0.229
Recall	Top-5	0.183	0.182	0.177	0.178
	Top-10	0.286	0.297	0.289	0.304
	Top-15	0.391	0.388	0.386	0.381
	Scenario	S-TAPER-Het	S-TAPER-Hom	TAPER-Het	TAPER-Hom
	Top-5	0.187	0.180	0.199	0.201
	Top-10	0.310	0.304	0.313	0.329
	Top-15	0.402	0.398	0.416	0.411

expert recommendation via tensor factorization. Furthermore, S-TAPER gives an average improvement of 13.2% in precision and 10.6% in recall. This indicates that the social ties of users and experts can help find users with similar behaviors on selecting experts, which provide more significant contributions to the personal expert recommendation than the geo-spatial and topical. Recall that social ties implicitly capture latent geo-spatial and topical preferences.

5.4.3.1 The Impact of Contextual Preferences

We have seen that geo-spatial, topical, and social preferences can be integrated into tensor factorization for improved personalized expert recommendation. In this section, we aim to dig deeper into the impact of geo-spatial, topical, and social signals on the quality of personalized recommendation. What impact do heterogeneous and homogeneous contextual preferences have? Are these impacts equal across approaches? For this experiment, we add the suffixes *Het* and *Hom* to indicate which variant of the proposed framework is

at study. For instance, G-TAPER-Hom represents the model G-TAPER by only leveraging the geo-spatial preferences between homogeneous entities including $\boldsymbol{H}_G$, $\boldsymbol{V}_G$, and $\boldsymbol{W}_G$. Similarly, we consider variations of T-TAPER, S-TAPER, as well as the full TAPER.

As it can be seen in Table 5.2, TAPER-Hom outperforms TAPER-Het with an average improvement of 3.4% in precision and 1.6% in recall, indicating that in general, the contextual preferences between homogeneous entities plays a more important role than between heterogeneous entities since such information can help identify similar users and further improve the quality of the recommendation. G-TAPER-Hom has a better performance than G-TAPER-Het with an average improvement of 3.0% in precision and 1.6% in recall. T-TAPER-Hom gives an average improvement of 3.5% in precision and 1.9% in recall. These results indicate that the geo-spatial and topical preferences between homogeneous entities contribute more to the proposed framework. However, it is surprising that S-TAPER-Het performs better than S-TAPER-Hom with an average improvement of 4.3% over S-TAPER-Hom in precision and 2.4% in recall. This implies that the following relationships between users and experts are a strong signal, and confirms that if a user is already following this expert, it is very likely that this user will include this expert on the list [67]. In addition, we also observe that the social preferences are more significant in contributing to the proposed framework than other factors.

5.4.3.2 The Impact of Negative Experts

We next turn to the impact of "negative experts", that is, to incorporate evidence of experts for whom a user is *not interested*. We seek to understand if these negative experts can be used as evidence in addition to the positive relationships investigated so far (e.g., by exploiting the geo-spatial preferences of users for experts). For example, knowing that a user is interested in a California Politics expert, but not interested in an expert on US National Politics may convey strong information about the preferences for that user

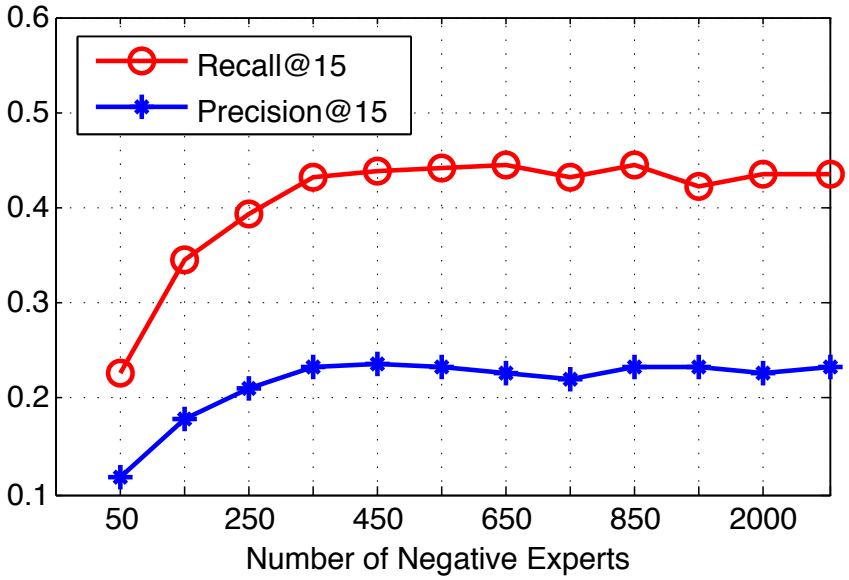


Figure 5.8: Effect of Number of Negative Experts.

on topics of regional (but not necessarily national) issues. To test the impact of these negative experts, we run the following experiment: first, we take the task of recommending the top-15 experts to users, and then we vary the number of negative experts. We vary the number of negative experts from 50 to 3,000 for each user by randomly selecting experts whom this user do not put in the Twitter list. Figure 5.8 demonstrates the impact of an increasing number of negative experts on the precision and recall of personalized expert recommendation. First, we observe that both precision and recall increase as the number of negative experts increases. This indicates that this signal of not being interested can provide some additional information beyond the positive relationships exploited so far. Second, we observe that the precision and recall curves flatten once the number of negative experts is larger than 350. Since there are nearly 9,000 experts in the dataset, the probability of false negative samples is small when only selecting a tiny part of them. However, this probability will increase as the number of negative experts grows, further affecting the quality of recommendations.

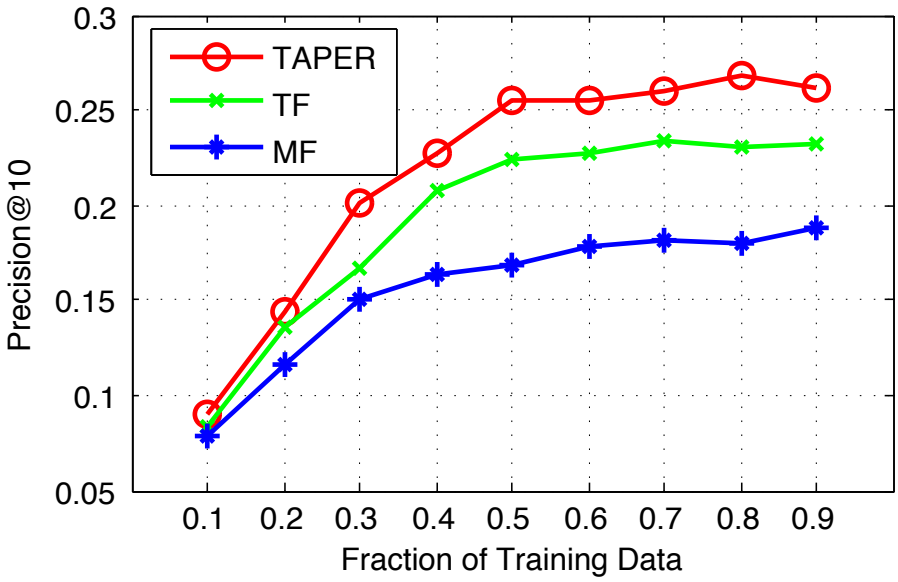


Figure 5.9: Precison@10 by varying the amount of training data.

5.4.3.3 Varying the Amount of Training Data

For the experiments so far, we have relied on cross-validation over a random split of experts into a training half and a testing half. Here, we explore the impact of varying the amount of training data on the task of personalized expert recommendation. Does the proposed approach still perform better than alternatives even with low amounts of training data? Do precision and recall plateau at some point? We vary the fraction of training data from 10% to 90% and evaluate TAPER versus a baseline tensor factorization method (TF) and a non-negative matrix factorization (MF) method on the task of personalized recommendation of the top-10 experts to each user. As we can see in Figures 5.9 and 5.10, the proposed framework TAPER consistently outperforms both MF and TF in precision and recall, across all fractions of training data. We also observe that the precision curves for all methods plateaus around 30%, indicating that good results may be achieved with even less training data. Naturally, the recall of all methods consistently increases as the

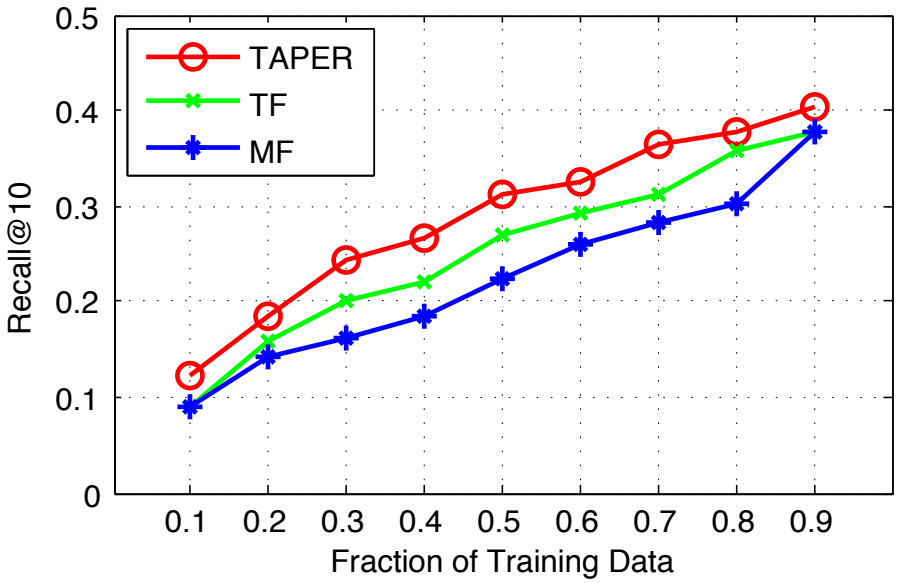


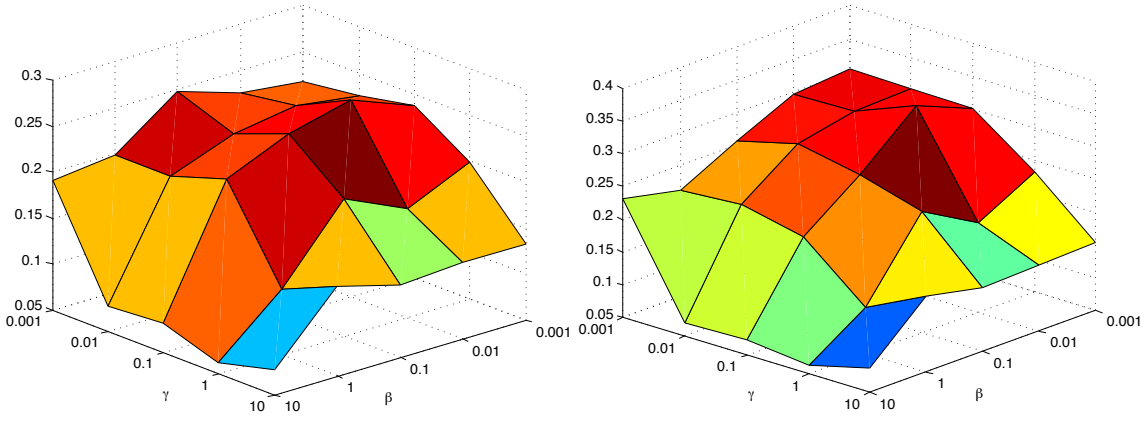
Figure 5.10: Recall@10 by varying the amount of training data.

training data increases, since recall is more sensitive to the number of testing samples.

5.4.4 Parameter Analysis

Two of the critical parameters for the proposed TAPER are γ and β. Recall that γ controls the contribution of contextual information between homogeneous entities (e.g., users and users, experts and experts, and topics and topics), whereas β controls the contribution of contextual information between heterogenous entities (e.g., users and experts, users and topics, and experts and topics). In order to better understand the impacts of these two parameters, we next conduct experiments to compare TAPER across different parameter settings. We vary values of these parameters in [0.001 0.01, 0.1, 1, 10] and present the experimental results of precision and recall in Figure 5.11 for the task of recommending the top-10 experts for each user. From the figures, we observe that our proposed framework TAPER achieves relatively consistent performance when choosing these regularization parameters across a wide range. Specifically, we find that the setting $\gamma = 0.1$, $\beta = 0.01$ gives

the best performance, and that parameter settings of $\gamma < 1$ and $\beta < 0.1$ lead to fairly stable precision and recall. These results indicate the stability of TAPER to these regularization parameters.



(a) Precison@10 (b) Recall@10

Figure 5.11: Impact of β (for heterogeneous entities) and γ (for homogeneous entities)

5.5 Conclusion

We have studied the problem of personalized expert recommendation through a tensor-based exploration of geo-spatial, topical, and social context across users, experts and topics. Through a Twitter dataset, we have seen that the proposed framework can improve the quality of the recommendation by over 30% in both precision and recall compared to state-of-the-art baselines.

6. T-CRM: LEARNING USER TOPICAL PROFILES*

In this chapter, we address the challenges of **sparsity**, **similarity** and **specificity** on the recovery of missing information on social media. Concretely, we extend the proposed tensor learning algorithm to another missing information problem on social media — *learning user topical profiles*. User interests and expertise are valuable but often hidden resources on social media even though social networks (e.g., Twitter Lists and Linkedin's Skill Tags) may provide a partial perspective on what users are known for. Specifically, we tackle the problem by developing a tensor-based contextual regularization model that simultaneously considers multiple (possibly conflicting) contexts based on a user's friends, interests, and behaviors. We show how these contexts can be embedded in a generalized tensor-based optimization framework that takes into account pairwise relations among all contexts for robustly learning user profiles. Extensive experiments demonstrate the superiority of the proposed model in terms of learning high-quality user topical profiles, and outperforms a cross-triadic factorization state-of-the-art baseline.

6.1 Introduction

In social media systems, *demographic profiles* — often including name, age, gender, and location — provide an important first step toward creating rich user models for information personalization. For example, a user's location can be a signal to surface local content in the Facebook newsfeed. Of course, these demographic profiles typically reveal very little about a user's topical interests (what she likes) or expertise (what she is known for). Hence, there is great effort toward building high-quality *user topical profiles*,

*Reprinted with permission from "What Are You Known For? Learning User Topical Profiles with Implicit and Explicit Footprints" by Cheng Cao, Hancheng Ge, James Caverlee, Haokai Lu, and Xia Hu, 2017. Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. Copyright 2017 by ACM. DOI: hŁp://dx.doi.org/10.1145/3077136.3080820

toward improving user experience and powering important applications like personalized recommendation [78], expert mining [60], and community detection [85].

Indeed, there are two major approaches to build the topical profiles for social media users. One thread of methods seeks to uncover latent factors that may be descriptive of a user. For example, running Latent Dirichlet Allocation (LDA) over a user's posts in social media can reveal the topics of interest of the user [59, 77, 136]; similarly, matrix factorization approaches have proven popular at capturing user factors, often for personalization purposes [74, 75, 76, 78, 80, 82, 137]. Aside from such recommendation applications, latent factor models have also been used to find influential users, mine communities, and predict review quality [59, 85, 93]. Another thread of methods seeks to encourage social media users to directly assess each other's interests and expertise, providing a partial perspective on user topical profiles. For example, LinkedIn users can choose *skill tags* for their own profiles and can endorse these tags on the profiles of others. *Twitter Lists* allow users to organize others according to user-selected keywords, e.g., placing a group of popular chefs on the list "Top Chefs". In this way, some list names can be viewed as a topical tag for list members. In the aggregate, this crowd-contributed tagging knowledge can be viewed as explicit evidence for capturing user interests and expertise [60, 131, 132].

Both approaches, however, face great challenges. Approaches that identify latent topics (often, as a distribution over features in some lower dimensional space) are typically trained only over content (ignoring other important footprints) and are difficult to directly interpret. Methods that only use crowdsourced tags typically suffer from limited coverage; that is, while the hand-curated tags may be of high-quality, very few users actually have descriptive topical tags associated with them. For example, in a random sample of 3.5 million Twitter users, we find that only 2% have been labeled with a topical tag (more details in Section 6.4). Moreover, to better understand user topical interests and expertise, a more comprehensive profiling framework is necessary. For instance, it is unclear what kind of

evidence is useful for user topical profiling. And how can such potentially heterogeneous evidence be modeled for user topical profiling?

In this chapter, we tackle the problem of learning user topical profiles on Twitter by exploiting heterogeneous footprints (e.g., tags, friends, interests, behavior). Based on a small set of explicit user tags, our goal is to extend this known set to the wider space of users who have no explicit tags. Toward bridging these two approaches, we propose to exploit heterogeneous contextual information for intelligently learning user topical profiles. The key intuition is to identify "similar" users in terms of their topical profiles by exploiting their similarity in a *contextual space*. For instance, Twitter users who post similar hashtags may have similar interests, and YouTube users who upvote the same videos may have similar preferences. Such evidence of *homophily* has been widely studied in the sociological literature [138] and repeatedly observed in online social media, e.g., [139, 140, 141, 142, 143]. But what contextual spaces are appropriate for finding this homophily? What impact do they have on the discovery of user topical profiles? Which contexts are more effective? Toward answering these questions, we formulate the problem of learning user topical profiles in social media, with a focus on leveraging heterogeneous contexts and demonstrate how to model textual, social, and behavioral contexts under this framework, and we present a unified factorization model in which we simultaneously consider all of these contexts (called CRM). In addition, we extend this initial approach through a generalized model that integrates the pairwise relations across all potential contexts via a tensor-based contextual model (called T-CRM), which provides a more robust framework for user profile learning. Through extensive experiments, we find the proposed T-CRM model is capable of learning high-quality user topical profiles with better performance than state-of-the-art baselines, and find that behavioral context is the single strongest factor, but that intelligent combination of all three contexts leads to the best overall performance.

6.2 Learning User Topical Profiles

Let $\boldsymbol{\mathcal{U}} = \{u_1, u_2, \ldots, u_N\}$ be a set of users where N is the number of users, and $\boldsymbol{T} = \{t_1, t_2, \ldots, t_M\}$ is a set of M tags each of which is associated with a particular topic. Suppose we have a subset of users $\boldsymbol{\mathcal{S}} \subset \boldsymbol{\mathcal{U}}$ where each user $u_i \in \boldsymbol{\mathcal{S}}$ has been labeled with a subset of $\boldsymbol{T}$, typically based on the collective efforts of the crowd. Practical examples of such tagging include LinkedIn Skill Tags and Twitter Lists, wherein users can provide a crowdsourced summary of a user's interests and expertise [131, 60, 132]. We denote these crowdsourced user topical profiles as the user-tag matrix $\boldsymbol{P} \in \mathbb{R}^{|\mathcal{S}| \times M}$ in which element $\boldsymbol{P}(i, j)$ represents the number of times u_i is labeled by t_j.

Learning User Topical Profiles. Given a set of users $\boldsymbol{\mathcal{U}}$, a set of tags $\boldsymbol{T}$, and a subset of users $\boldsymbol{\mathcal{S}} \subset \boldsymbol{\mathcal{U}}$ for whom we know their user topical profiles $\boldsymbol{P}$, the problem of *Learning User Topical Profiles* is the task of inferring the unknown tags from $\boldsymbol{\mathcal{T}}$ for users in $\boldsymbol{\mathcal{U}} - \boldsymbol{\mathcal{S}}$.

An Initial Attempt. A natural choice for attacking the challenge of learning user topical profiles is the matrix completion approach, which has been adopted in many related works [82, 99, 98, 80]. Under a matrix completion approach, we can extend $\boldsymbol{P}$ to a larger matrix $\boldsymbol{X} \in \mathbb{R}^{N \times M}$ by including all users of $\boldsymbol{\mathcal{U}}$. Then, we can formulate the learning user topical profiles problem as a matrix completion problem:

$$\begin{aligned} \min_{\boldsymbol{U}, \boldsymbol{V}} \quad & \frac{1}{2} \|\boldsymbol{\Omega} \odot (\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T)\|_F^2, \\ \text{s. t.} \quad & \boldsymbol{U} \geq 0, \boldsymbol{V} \geq 0, \end{aligned} \tag{6.1}$$

where $\boldsymbol{X}$ is a user-tag matrix, and $\boldsymbol{U} \in \mathbb{R}^{N \times K}$ and $\boldsymbol{V} \in \mathbb{R}^{M \times K}$ are latent representations of users and tags, respectively. $K \ll min(N, M)$ is the number of latent dimensions. Since the given $\boldsymbol{X}$ is naturally non-negative, we add the same constraints for $\boldsymbol{U}$ and $\boldsymbol{V}$ so that we can better interpret the values in them. $\boldsymbol{\Omega}$ is a non-negative matrix with the same

size of $\boldsymbol{X}$:

$$\boldsymbol{\Omega}(i,j) = \begin{cases} 1 & \text{if } \boldsymbol{X}(i,j) \text{ is observed,} \\ 0 & \text{if } \boldsymbol{X}(i,j) \text{ is unobserved.} \end{cases}$$

The basic matrix completion model above learns an optimal set of $\{\boldsymbol{U}, \boldsymbol{V}\}$ to approximate the original matrix $\boldsymbol{X}$, estimating for unobserved users through observed user-tag pairs. However, as in many linear-inverse problems, there may not be sufficient information to estimate the original matrix $\boldsymbol{X}$ based only on the partially observed data. The problem of learning user topical profiles is one such case, since most of our target users do not have any partially known topical profiles. Hence, with this challenge in mind, we propose to exploit multiple heterogeneous contexts — social, textual, and behavioral — to build a robust contextual factorization approach for learning unknown user topical profiles. The key intuition is to identify "similar" users in terms of their topical profiles by exploiting their similarity in these *contextual spaces*.

6.3 A Generalized Contextual Regularization Model

We turn in this section to proposing a generalized contextual regularization model for learning user topical profiles. We first identify multiple contexts and show how to model them. We then introduce a matrix factorization based approach — called the Contextual Regularization Model (CRM) — for learning user topical profiles, before extending this version to a more general tensor-based approach — called the Tensor-based Contextual Regularization Model (T-CRM).

6.3.1 Modeling Contexts

As discussed in Section 6.1, we aim to model three kinds of behavioral contexts for learning user topic profiles: social, textual, and behavioral. The intuition is that these contextual spaces can connect related users, such that user topical profiles can be propagated

from user to user. But how should we model these contexts? And how can we integrate them into a matrix completion model?

6.3.1.1 Social Context

Social context — directly suggested by homophily — naturally indicates that connected users may share common interests, and hence can be used for inferring user topical profiles [73, 75, 86, 87]. For example, if Carol and David are friends on Twitter, then the social context can assert that they will share common interests.

These social network connections between users can be naturally modeled as a matrix. We denote the matrix as $\boldsymbol{E} \in \mathbb{R}^{N \times N}$ in which the binary element $E(i, j)$ represents if user u_i and user u_j have a connection on a social network. We can model this social context as a regularization term:

$$\mathcal{L}_1 = \frac{1}{2} \| \boldsymbol{E} - \boldsymbol{U}\boldsymbol{U}^T \|_F^2.$$

Our goal is to optimize the user latent matrix $\boldsymbol{U}$ in order to minimize $\mathcal{L}_1$, with the intuition that friends are likely to have similar profiles. Of course, users may form relationships in social media for many diverse reasons, and so these relationships may not be appropriate for inferring similar topical profiles. As one example, family members may be "friends" in a social network but can have distinct topical profiles (e.g., sister vs brother, grandson vs grandfather). Hence, we next consider additional contexts that may serve to mitigate these challenges.

6.3.1.2 Textual Context

The second context we consider is textual context. Text posted by users can semantically reflect related subjects associated with their interests or expertise. Thus, many studies have directly applied LDA on posted texts, assuming the (latent) topics in user's posts are their topical profiles [59, 77, 136]. In Figure 6.1, Alice is a basketball fan and

Figure 6.1: Examples of Textual Context on Learning User Topical Profiles.

she has posted many tweets talking about the upcoming NBA all-star game. We find that Bob's tweets share many of the same words as Alice's. Hence, their posted texts demonstrate their shared interests in basketball, suggesting that Alice's user topical profile may be similar to Bob's user topical profile.

We can model this context like so: let $\boldsymbol{w} = \{w_1, w_2, \ldots, w_L\}$ be the set of words, where L denotes the number of words. $\boldsymbol{A} \in \mathbb{R}^{N \times L}$ is a user-word matrix in which $\boldsymbol{A}(i,j)$ is the frequency of word w_j appearing in user u_i's posts. Similarly, $\boldsymbol{B} \in \mathbb{R}^{M \times L}$ is a tag-word matrix where $\boldsymbol{B}(i,j)$ represents the frequency of word w_j posted by all users who have tag t_i. We propose to leverage a user's textual context as the following regularization term:

$$\mathcal{L}_2 = \frac{1}{2}\|\boldsymbol{A} - \boldsymbol{U}\boldsymbol{W}^T\|_F^2 + \frac{1}{2}\|\boldsymbol{B} - \boldsymbol{V}\boldsymbol{W}^T\|_F^2,$$

where $\boldsymbol{W} \in \mathbb{R}^{L \times K}$ represents word's latent topics. Our goal is to minimize $\mathcal{L}_2$ so that two users who are "nearby" in the textual space tend to have similar topical profiles. Of course,

Figure 6.2: Examples of Behavioral Context on Learning User Topical Profiles.

a user's posts are often short (like on Twitter) and may contain many nonsense characters or off-topic posts, which can interfere with clearly revealing user topical profiles. Hence, we next turn to a third context for overcoming these challenges.

6.3.1.3 Behavioral Context

Finally, we propose to augment the social and textual contexts with behavioral context [73, 79, 88]. According to the homophily evidence in the behavior dimension [138], for instance, two YouTube users may have close tastes if they usually "like" or "dislike" the same videos. A retweet on Twitter is a strong indication of personal endorsement, so two users can have similar preferences if they often retweet the same tweets. Hence, these behaviors may provide strong evidence beyond who users are connected to (social) and what they post (textual).

Here we adopt *URL sharing* as a public, observable behavior that may serve as a first step toward improving the learning of user topical profiles. Other behavioral contexts are possible, and we anticipate revisiting these in our future work. Through URL sharing, users can concisely express their viewpoints, interests, and professional expertise.

For instance, a person who works in the IT industry may usually post URLs linking to `engadget.com`. A user who likes sports may often share URLs of `espn.com`. In Figure 6.2, Carol is a political journalist so she regularly posts some URLs linking to `huffingtonpost.com`, and we see David also usually shares the same URLs. In this case we may infer politics-relevant tags for David.

Concretely, let $\boldsymbol{Z} = \{z_1, z_2, \ldots, z_P\}$ be the set of URLs posted by users (in this case on Twitter). Similar to textual context, we define $\boldsymbol{C} \in \mathbb{R}^{N \times P}$ as a user-URL matrix where $\boldsymbol{C}(i, j)$ is the frequency of URL z_j posted by user u_i. Also, $\boldsymbol{D} \in \mathbb{R}^{M \times P}$ is a tag-URL matrix with $\boldsymbol{D}(i, j)$ as the frequency of URL z_j appearing in all posts from users having tag t_i. As a result, we leverage URL sharing via the following loss function:

$$\mathcal{L}_3 = \frac{1}{2}\|\boldsymbol{C} - \boldsymbol{U}\boldsymbol{G}^T\|_F^2 + \frac{1}{2}\|\boldsymbol{D} - \boldsymbol{V}\boldsymbol{G}^T\|_F^2,$$

where $\boldsymbol{G} \in \mathbb{R}^{P \times K}$ represents URL's latent topical spaces. Our goal is to minimize $\mathcal{L}_3$, with the idea that users may have similar topical profiles if they behave similarly when posting URLs.

6.3.2 CRM: A Contextual Regularization Model

Since evidence from these heterogeneous contexts may provide conflicting evidence, potentially leading to lower quality user profiles than considering contexts in isolation, we turn in this section to developing a unified model that can integrate all these heterogeneous contexts together into a matrix completion model. Since all contexts are modeled as regularization terms in Section 6.3.1, intuitively we can linearly incorporate them into the proposed *Contextual Regularization Model (CRM)*.

Figure 6.3 gives an overview of CRM. We factorize each of the social, textual, and behavioral matrices, and assume the objective user-tag matrix shares the same latent user dimensions with them. This is the fundamental assumption in most factorization-based

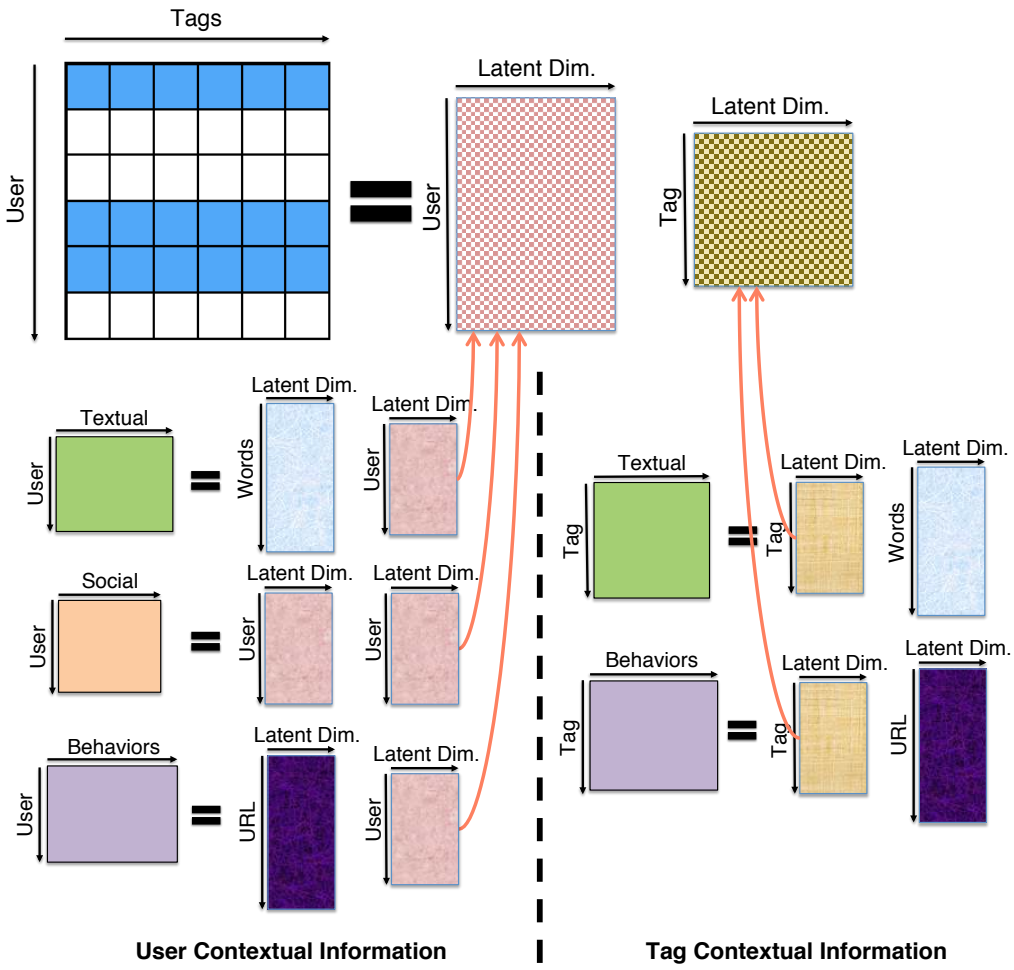


Figure 6.3: An Overview of CRM

methods for solving matrix completion problems. Similarly, we collect each tag's latent representations, and multiply them with each user's latent factors for estimating the objective matrix.

Specifically, we can formulated the following optimization problem:

$$\begin{aligned}\min_{\boldsymbol{U},\boldsymbol{V},\boldsymbol{W},\boldsymbol{G}} \quad & \mathcal{F} = \frac{1}{2}\|\boldsymbol{\Omega} \odot (\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T)\|_F^2 \\ & + \frac{\lambda}{2}(\|\boldsymbol{A} - \boldsymbol{U}\boldsymbol{W}^T\|_F^2 + \|\boldsymbol{B} - \boldsymbol{V}\boldsymbol{W}^T\|_F^2) \\ & + \frac{\gamma}{2}(\|\boldsymbol{C} - \boldsymbol{U}\boldsymbol{G}^T\|_F^2 + \|\boldsymbol{D} - \boldsymbol{V}\boldsymbol{G}^T\|_F^2) \\ & + \frac{\delta}{2}\|\boldsymbol{E} - \boldsymbol{U}\boldsymbol{U}^T\|_F^2 \\ & + \frac{\alpha}{2}(\|\boldsymbol{U}\|_F^2 + \|\boldsymbol{V}\|_F^2 + \|\boldsymbol{W}\|_F^2 + \|\boldsymbol{G}\|_F^2) \\ \text{s. t.} \quad & \boldsymbol{U} \geq 0, \boldsymbol{V} \geq 0, \boldsymbol{W} \geq 0, \boldsymbol{G} \geq 0,\end{aligned} \tag{6.2}$$

where λ, γ, δ and α are positive regularization parameters controlling the contributions of different contexts. λ is introduced to control the weights of text associated with users and tags, γ is a parameter to control the weights of behaviors from users as well as tags, and δ is to control the weight of social contexts between users. $\|\boldsymbol{U}\|_F^2$, $\|\boldsymbol{V}\|_F^2$, $\|\boldsymbol{W}\|_F^2$ and $\|\boldsymbol{G}\|_F^2$ are deployed to avoid overfitting. Similar to Equation 6.1, we insert the non-negative constraints for $\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{W}$, and $\boldsymbol{G}$.

Next, we introduce an algorithm to solve the optimization problem in Eq.(6.2). Since there are multiple variables in the objective function, we propose an algorithm to alternatively learn optimal solutions for four variables $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{W}$ and $\boldsymbol{G}$. The key idea is to optimize the objective function with respect to one variable, while fixing others. The algorithm will keep updating the variables until convergence or reaching the number of maximum iterations.

The derivation of the objective function in Eq.(6.2) regarding four variables $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{W}$

and $\boldsymbol{G}$ are demonstrated as:

$$\begin{aligned}
\frac{\partial \mathcal{F}}{\partial \boldsymbol{U}} =& -\boldsymbol{\Omega} \odot \boldsymbol{\Omega} \odot (\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T)\boldsymbol{V} - \lambda(\boldsymbol{A} - \boldsymbol{U}\boldsymbol{W}^T) \\
& - \gamma(\boldsymbol{C} - \boldsymbol{U}\boldsymbol{G}^T) - 2\delta(\boldsymbol{E} - \boldsymbol{U}\boldsymbol{U}^T) + \alpha\boldsymbol{U}, \\
\frac{\partial \mathcal{F}}{\partial \boldsymbol{V}} =& -\boldsymbol{\Omega}^{\boldsymbol{T}} \odot \boldsymbol{\Omega}^{\boldsymbol{T}} \odot (\boldsymbol{X}^T - \boldsymbol{V}\boldsymbol{U}^T)\boldsymbol{U} - \lambda(\boldsymbol{B} - \boldsymbol{V}\boldsymbol{W}^T) \\
& - \gamma(\boldsymbol{D} - \boldsymbol{V}\boldsymbol{G}^T) + \alpha\boldsymbol{V}, \\
\frac{\partial \mathcal{F}}{\partial \boldsymbol{W}} =& -\lambda(\boldsymbol{A}^T - \boldsymbol{W}\boldsymbol{U}^T)\boldsymbol{U} - \lambda(\boldsymbol{B}^T - \boldsymbol{W}\boldsymbol{V}^T)\boldsymbol{V} + \alpha\boldsymbol{W}, \\
\frac{\partial \mathcal{F}}{\partial \boldsymbol{G}} =& -\gamma(\boldsymbol{C}^T - \boldsymbol{G}\boldsymbol{U}^T)\boldsymbol{U} - \gamma(\boldsymbol{D}^T - \boldsymbol{G}\boldsymbol{V}^T)\boldsymbol{V} + \alpha\boldsymbol{G}.
\end{aligned} \tag{6.3}$$

Based upon the derivations illustrated above, we then apply stochastic gradient descent to iteratively update each variable by taking a step η along its gradient ascending. The algorithm details are presented in **Algorithm 3** in which learning steps η_u, η_v, η_w and η_g are chosen based upon the Goldstein Conditions [144]. We implement the non-negative constraints on $\boldsymbol{U}$ and $\boldsymbol{V}$ through forcing their negative values to 0 in each iteration. As shown, this algorithm considers all three contexts together to estimate the topical profiles for each user.

Though unifying all three heterogeneous contexts, this initial CRM approach has two main drawbacks. First, it will become complex if we introduce additional contexts, as we bring in more controlling parameters of new contexts to be tuned. In addition, CRM does not take into account the relations between those heterogeneous contexts which could be jointly explored in the latent space. Given these concerns, can we find a generalized model that can jointly leverage all potential heterogeneous contexts? We turn in the following section to answering this question.

Algorithm 3: CRM Solver

Input: user-tag matrix $\boldsymbol{X}$, user-word matrix $\boldsymbol{A}$, tag-word matrix $\boldsymbol{B}$, user-url matrix $\boldsymbol{C}$, tag-url matrix $\boldsymbol{D}$, user friendship matrix $\boldsymbol{E}$, observation indication matrix $\boldsymbol{\Omega}$ and parameters $\{\lambda, \gamma, \delta, \rho, \eta\}$

Output: $\boldsymbol{U}, \boldsymbol{V}$

Initialize $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{W}$ and $\boldsymbol{G}$ randomly, $t = 0$

while Not Converged **do**

  Compute $\frac{\partial \mathcal{F}}{\partial \boldsymbol{U}}$, $\frac{\partial \mathcal{F}}{\partial \boldsymbol{V}}$, $\frac{\partial \mathcal{F}}{\partial \boldsymbol{W}}$ and $\frac{\partial \mathcal{F}}{\partial \boldsymbol{G}}$ in Eq.(6.3)

  Update $\boldsymbol{U}_{t+1} \leftarrow \max(\boldsymbol{U}_t - \eta_u \frac{\partial \mathcal{F}}{\partial \boldsymbol{U}}, 0)$

  Update $\boldsymbol{V}_{t+1} \leftarrow \max(\boldsymbol{V}_t - \eta_v \frac{\partial \mathcal{F}}{\partial \boldsymbol{U}}, 0)$

  Update $\boldsymbol{W}_{t+1} \leftarrow \max(\boldsymbol{W}_t - \eta_w \frac{\partial \mathcal{F}}{\partial \boldsymbol{U}}, 0)$

  Update $\boldsymbol{G}_{t+1} \leftarrow \max(\boldsymbol{G}_t - \eta_g \frac{\partial \mathcal{F}}{\partial \boldsymbol{U}}, 0)$

  $t = t + 1$

return $\boldsymbol{U}$ and $\boldsymbol{V}$

6.3.3 T-CRM: A Tensor-based Contextual Regularization Model

In this section, we augment CRM with a *Tensor-based Contextual Regularization Model (T-CRM)* toward jointly exploring the relationships across contexts for more robust user topical profile learning. First, to relieve the dramatic increase of parameters when introducing more regularization terms, we need to replace the linear combination model in CRM by a more compact factorization model. Second, such a compact factorization model should consider all possible pairwise relations between all contexts. Therefore, we adopt a *tensor factorization* model which explicitly takes into account the data's multi-way structure. Moreover, the factorization will only happen once even if we introduce more heterogeneous contexts.

Figure 6.4 shows an overview of T-CRM. In general, we model all contexts in one tensor via calculating the user similarity in each context. There can be many options for measuring the user similarity in every context. We test many of them and report the one providing the best performance in Section 6.4. Then, we factorize the tensor and obtain

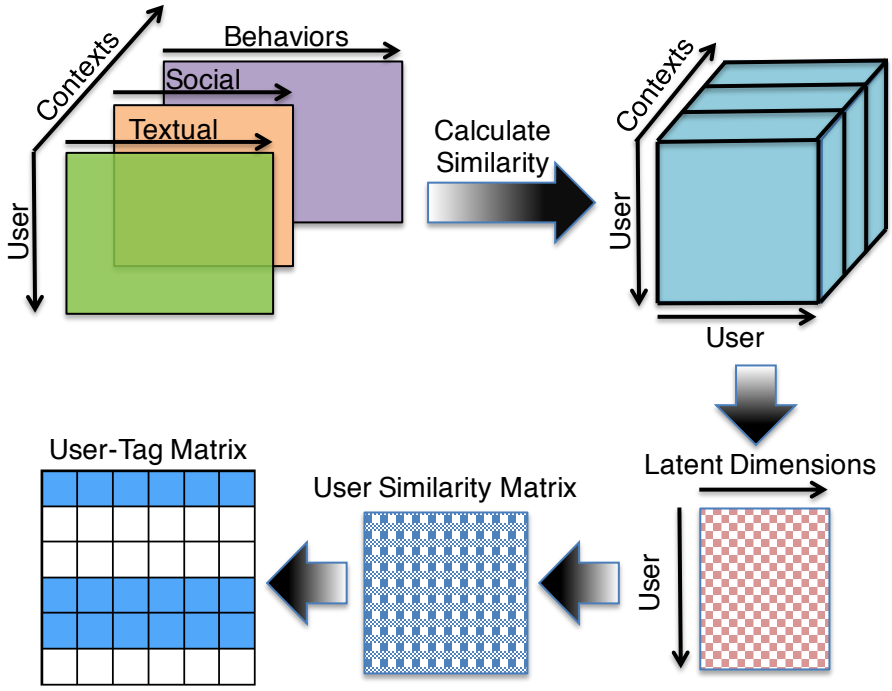


Figure 6.4: An Overview of T-CRM

a matrix of latent representations for all users, upon which we extract a user similarity matrix to estimate the original user-tag matrix.

Concretely, we denote the tensor as $\boldsymbol{\mathcal{C}} \in \mathbb{R}^{N \times N \times R}$ which is a multidimensional array where R is the number of contexts and N is the size of the user set. We can factorize the tensor $\boldsymbol{\mathcal{C}}$ to two latent user matrices $\boldsymbol{Q} \in \mathbb{R}^{N \times K}$ and $\boldsymbol{S} \in \mathbb{R}^{N \times K}$, and one latent context matrix $\boldsymbol{Y} \in \mathbb{R}^{R \times K}$, where K is the number of latent dimensions. The tensor factorization is to solve the optimization problem defined below:

$$\min_{\boldsymbol{Q},\boldsymbol{S},\boldsymbol{Y}} \quad \frac{1}{2}\|\boldsymbol{\mathcal{C}} - [\![\boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{Y}]\!]\|_F^2 + \frac{\alpha}{2}(\|\boldsymbol{Q}\|_F^2 + \|\boldsymbol{S}\|_F^2 + \|\boldsymbol{Y}\|_F^2), \tag{6.4}$$

where $[\![\boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{Y}]\!] \in \mathbb{R}^{N \times N \times R}$ is given by

$$[\![\boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{Y}]\!] = \sum_{k=1}^{K} \boldsymbol{q}_k \circ \boldsymbol{s}_k \circ \boldsymbol{y}_k.$$

Here $\boldsymbol{q}_k$, $\boldsymbol{s}_k$ and $\boldsymbol{y}_k$ are the k^{th} column vectors of $\boldsymbol{Q}$, $\boldsymbol{S}$ and $\boldsymbol{Y}$ respectively. We adopt the existing CPOPTR method [30] to solve Eq.(6.4). The latent context matrix $\boldsymbol{Y}$ represents the contribution of each type of context to different latent dimensions. The latent matrices $\boldsymbol{Q}$ and $\boldsymbol{S}$ are jointly calculated across the different contexts. Hence, each user can be represented by latent features after combining $\boldsymbol{Q}$ and $\boldsymbol{S}$. In this study, we employ $\boldsymbol{\Lambda} = \boldsymbol{Q} + \boldsymbol{S}$ as the latent features after column normalization.

The next natural question is how to leverage the new latent space $\boldsymbol{\Lambda}$ of all users. The basic idea is that two users tend to have similar topical profiles if they have similar latent representations derived by jointly considering all their contexts. Thus, we can see $\boldsymbol{\Lambda}$ as a "new context" and formulate it as the new loss function:

$$\begin{aligned}
\boldsymbol{\Theta} &= \frac{1}{2} \sum_{i,j} \Psi(i,j) \|\boldsymbol{U}_i - \boldsymbol{U}_j\|^2 \\
&= \sum_{i,j} \boldsymbol{U}_i \Psi(i,j) \boldsymbol{U}_i{}^T - \sum_{i,j} \boldsymbol{U}_i \Psi(i,j) \boldsymbol{U}_j{}^T \\
&= \sum_{i} \boldsymbol{U}_i \boldsymbol{D}(i,i) \boldsymbol{U}_i{}^T - \sum_{i,j} \boldsymbol{U}_i \Psi(i,j) \boldsymbol{U}_j{}^T \\
&= \text{tr}(\boldsymbol{U}^T (\boldsymbol{D} - \boldsymbol{\Psi}) \boldsymbol{U}) \\
&= \text{tr}(\boldsymbol{U}^T \boldsymbol{\mathcal{L}} \boldsymbol{U}),
\end{aligned} \tag{6.5}$$

where $\boldsymbol{U}_i$ is the ith row of the user latent matrix $\boldsymbol{U}$, $tr(\cdot)$ denotes the matrix trace, and $\boldsymbol{\Psi}$ is a user similarity matrix computed from latent features of users $\boldsymbol{\Lambda}$ by the cosine similarity. $\boldsymbol{D}$ is a diagonal matrix in which $\boldsymbol{D}(i,i) = \sum_j \boldsymbol{\Psi}(i,j)$, and $\boldsymbol{\mathcal{L}} = \boldsymbol{D} - \boldsymbol{\Psi}$ is the graph Laplacian of the user similarity matrix $\boldsymbol{\Psi}$.

Algorithm 4: T-CRM Solver

Input: user-tag matrix $\boldsymbol{X}$, user-word matrix $\boldsymbol{A}$, user-url matrix $\boldsymbol{C}$, user friendship matrix $\boldsymbol{E}$, observation indication matrix $\boldsymbol{\Omega}$ and parameters $\{\alpha, \beta, \eta_u, \eta_v\}$

Output: $\boldsymbol{U}, \boldsymbol{V}$

Calculate the tensor $\boldsymbol{\mathcal{C}}$ from $\boldsymbol{A}$, $\boldsymbol{C}$ and $\boldsymbol{E}$
Calculate $[\boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{Y}] \leftarrow \text{CPOPTR}(\boldsymbol{\mathcal{C}})$
Calculate the latent features of users $\boldsymbol{\Lambda} = \boldsymbol{Q} + \boldsymbol{S}$
Calculate the user similarity matrix $\boldsymbol{\Psi}$ based on $\boldsymbol{\Lambda}$
Construct the graph Laplacian matrix $\boldsymbol{\mathcal{L}}$ for $\boldsymbol{\Psi}$
Initialize $\boldsymbol{U}$ and $\boldsymbol{V}$, randomly, $t = 0$
while Not Converged **do**
    Compute $\frac{\partial \mathcal{F}}{\partial \boldsymbol{U}} = -(\boldsymbol{\Omega} \odot \boldsymbol{\Omega})(\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T))\boldsymbol{V} + \beta \boldsymbol{\mathcal{L}}\boldsymbol{U}$
    Compute $\frac{\partial \mathcal{F}}{\partial \boldsymbol{V}} = -(\boldsymbol{\Omega}^T \odot \boldsymbol{\Omega}^T)(\boldsymbol{X}^T - \boldsymbol{V}\boldsymbol{U}^T))\boldsymbol{U}$
    Update $\boldsymbol{U}_{t+1} \leftarrow \max(\boldsymbol{U}_t - \eta_u \frac{\partial \mathcal{F}}{\partial \boldsymbol{U}}, 0)$
    Update $\boldsymbol{V}_{t+1} \leftarrow \max(\boldsymbol{V}_t - \eta_v \frac{\partial \mathcal{F}}{\partial \boldsymbol{U}}, 0)$
    $t = t + 1$
return $\boldsymbol{U}$ and $\boldsymbol{V}$

How can we utilize the new context $\boldsymbol{\Theta}$ to learn user topical profiles? Similarly, we are able to use $\boldsymbol{\Theta}$ to regulate latent representations of two similar users to make them as close as possible. Hence, we can build the T-CRM by solving the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{U},\boldsymbol{V}} \quad & \frac{1}{2}\|\boldsymbol{\Omega} \odot (\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T)\|_F^2 + \frac{\beta}{2}\text{tr}(\boldsymbol{U}^T\boldsymbol{\mathcal{L}}\boldsymbol{U}) \\ & + \frac{\alpha}{2}(\|\boldsymbol{U}\|_F^2 + \|\boldsymbol{V}\|_F^2), \\ \text{s. t.} \quad & \boldsymbol{U} \geq 0, \boldsymbol{V} \geq 0, \end{aligned} \tag{6.6}$$

where β is the controlling parameter. This optimization problem can be solved similarly as introduced in Section 6.3.2. The detailed solver is presented in **Algorithm 4**.

In summary, we first present a context-based factorization model (called CRM) in which each of three heterogeneous contexts is modeled as regularization terms. We provide Algorithm 3 to solve the optimization problem in Equation 6.2. Then we extend CRM

to a compact tensor-based contextual model (called T-CRM). Based on a tensor decomposition method, T-CRM can jointly handle relationships across multiple contexts without introducing new parameters. The complete overview of T-CRM is shown in Figure 6.4, and we propose Algorithm 4 to solve Equation 6.6.

6.4 Experiments

In this section, we conduct a series of experiments to answer the following questions: (i) How well do the proposed CRM and T-CRM models work? (ii) Which contexts are most effective? (iii) How does T-CRM compare with other alternatives? Does it really improve upon the simpler CRM approach? (iv) How do the proposed approaches compare to other variants?; and (v) What impact do the model parameters have on the ultimate performance? We begin by introducing the experimental setup including dataset collection and evaluation method.

6.4.1 Experiment Setup

6.4.1.1 Twitter Lists

We adopt Twitter Lists, a large publicly-accessible collection of crowd-contributed tagging knowledge for social media users. Recall that these lists allow one user to annotate another with a list name (or tag), e.g., politics, music, art. We randomly sample a set of 3.468 million Twitter users, and crawl the list membership information for each of them via the public Twitter API. We identify 977,000 users who have ever been included in some list, but we find a huge amount of noise. For instance, nonsense tags (like numbers, unicode characters, single letters) take up a major proportion. Many tags (e.g., “friend”, “love”, and “amigo”) are not reflective of topical profiles. Also, there exist many near-synonyms and variants such as “writer-author” and “news-noticia”. To obtain a set of candidate tags for our problem, we rank all tags by the number of labeled users, and manually curate the top-500 tags through merging variants and filtering noise.

6.4.1.2 Context

For the textual context, we aggregate all terms each user has posted and adopt the standard LDA topic model after filtering stopwords and stemming. We further measure user similarity in the textual context by calculating the pairwise Jensen-Shannon divergence. For the social context, we crawl the friendship connection information for each user. Following a user can be quite casual on Twitter, so we focus on mutual followings as the basis of user similarity in the social context. For the behavioral context, we aggregate all URLs a user has posted in her tweets and obtain the posting counts. We resolve all crawled URLs (most are shortened) to take care of URL variants, and focus on the URL domain name which conceptually represents a website. For quantifying similar URL sharing patterns, we test a set of measurements (e.g., intersection, cosine, jaccard) and find the one in [145] works best. We exclude those users who just occasionally share URLs, i.e., less than 10 postings.

6.4.1.3 Users

We collect a set of 72,096 users who have all three types of contextual information and have been labeled by at least one of the candidate tags. Since many of them have sparse tagging information, we rank all users by the number of tags they have and focus on the top 10,000.

In our proposed models, we end up with scores of all candidate tags for each user. Since we should take those most associated tags as user topical profiles, we rank them in descending order and focus on the top-k ranked tags. Our evaluation is based on ten-fold cross validation.

6.4.1.4 Metrics

We pick several metrics which can cover different evaluation aspects. On the one side, we would like to see the ratio of correct inferences for learning user topical profiles. And on the other side, we want to measure the prediction error. Thus, we adopt *precision@k* which measures the percentage of correctly estimated top-k tags, and *Mean Absolute Error (MAE)* which quantifies the prediction quality in terms of errors. Note that a lower MAE means a better performance.

Furthermore, besides the absolute measurement in accuracy, the relative ranking order is another important perspective, especially in some recommendation scenarios. The rank correlation coefficients of both *Kendall's* τ and *Spearman's* ρ are two prevalent metrics for measuring rank-based agreement across two lists. We use them both to measure the number of pairs of tags that are correctly ordered from our results. Their values both range from -1 to 1, with the higher the more relevant.

6.4.1.5 Baselines

We select three baselines as alternatives to the proposed T-CRM approach. To be fair, we incorporate all three proposed contexts and maintain the same experimental setup for all the following approaches:

- **Nearest Neighborhood (NN).** An intuitive solution is based on the traditional nearest neighborhood model. For each target user, we separately find a set of closest seed users in each context, and pick the intersected neighbors from whom we propagate their tags and scores and take the average for each tag.

- **Cross-domain Triadic Factorization (CTF)** [100]. This state-of-the-art approach directly combines user ratings of different merchandise (e.g., book, music, movie) into one tensor model, in which all the values are user ratings. Then, it adopts the exist-

ing PARAFAC2 algorithm to factorize the tensor and use the resulting latent factors for predicting user ratings.

- **Contextual Regularization Model (CRM).** Introduced in Section 6.3.2, this model is a basic version that considers each context as a regularization term and linearly adds them together.

6.4.1.6 Parameter Settings

To determine the number of latent dimensions in both CRM and T-CRM, we experiment with a sequence of settings {5, 10, 20, 30, 40, 50, 100} and empirically select 20 for both CRM and T-CRM, as a trade-off between accuracy and efficiency. In Algorithm 3, there are five parameters λ, γ, δ, α, and η. The first four parameters are used to control the contributions of various contexts. The last one is a step along its gradient ascending. As is commonly done, we iteratively employ cross-validation to tune these parameters. Specifically, we empirically set $\lambda = 0.02$, $\gamma = 0.7$, $\beta = 0.1$, $\alpha = 0.4$ and $\eta = 0.05$ for general experiments, respectively. In T-CRM, we choose 10 for the number of latent dimension in tensor factorization. In addition, two positive parameters α and β in Eq. (6.6) are involved in the experiments. Concretely, we empirically set $\alpha = 0.3$ and $\beta = 0.02$ via the cross-validation. The step size η in T-CRM is set to 0.05.

6.4.2 The Impact of Contexts

In general, as mentioned in Section 6.1, textual, social, and behavioral contexts have different emphases on user topical profiles. Hence, which contexts work better (or best) is one of the most compelling questions to answer. Hence, we compare different combinations of all contexts in both NN and CRM. The reason we do not test them in T-CRM is that the multi-way manner of T-CRM may not clearly tell which context contributes more. We show the results in Table 6.1 in which T is for textual, S is for social, and B is for

Table 6.1: The Impact of Different Contexts for Learning User Topical Profiles

Method	Precision			MAE		
	Top 5	**Top 10**	**Top 15**	**Top 5**	**Top 10**	**Top 15**
NN-T	0.2113	0.2356	0.2673	0.2914	0.2692	0.2432
NN-S	0.1920	0.2153	0.2330	0.3048	0.2791	0.2642
NN-B	0.2423	0.2629	0.3155	0.2650	0.2342	0.2110
CRM-T	0.3438	0.3791	0.4668	0.2264	0.2069	0.1897
CRM-S	0.3390	0.3837	0.4561	0.2298	0.2093	0.1887
CRM-B	0.3556	0.3980	0.4733	0.2275	0.1982	0.1699
CRM-T+S	0.3494	0.3847	0.4657	0.2300	0.2107	0.1872
CRM-T+B	0.3587	0.4132	0.4758	0.2193	0.1894	0.1909
CRM-S+B	0.3544	0.4069	0.4729	0.2238	0.1930	0.1852
CRM-T+S+B	**0.3616**	**0.4189**	**0.4931**	**0.2137**	**0.1861**	**0.1772**
Method	**Kendall's τ**			**Spearman's ρ**		
	Top 5	**Top 10**	**Top 15**	**Top 5**	**Top 10**	**Top 15**
NN-T	0.2460	0.1687	0.1531	0.3054	0.2262	0.1784
NN-S	0.2110	0.1420	0.1289	0.2670	0.1852	0.1682
NN-B	0.2826	0.2044	0.1834	0.3314	0.2429	0.2106
CRM-T	0.3221	0.2464	0.2031	0.4163	0.2987	0.2409
CRM-S	0.3172	0.2421	0.2003	0.4135	0.2916	0.2341
CRM-B	0.3286	0.2557	0.2067	0.4302	0.3015	0.2426
CRM-T+S	0.3205	0.2516	0.2085	0.4189	0.2970	0.2378
CRM-T+B	0.3329	0.2606	0.2197	0.4348	0.3071	0.2535
CRM-S+B	0.3272	0.2588	0.2185	0.4322	0.3054	0.2561
CRM-T+S+B	**0.3403**	**0.2746**	**0.2267**	**0.4414**	**0.3104**	**0.2682**

behavioral.

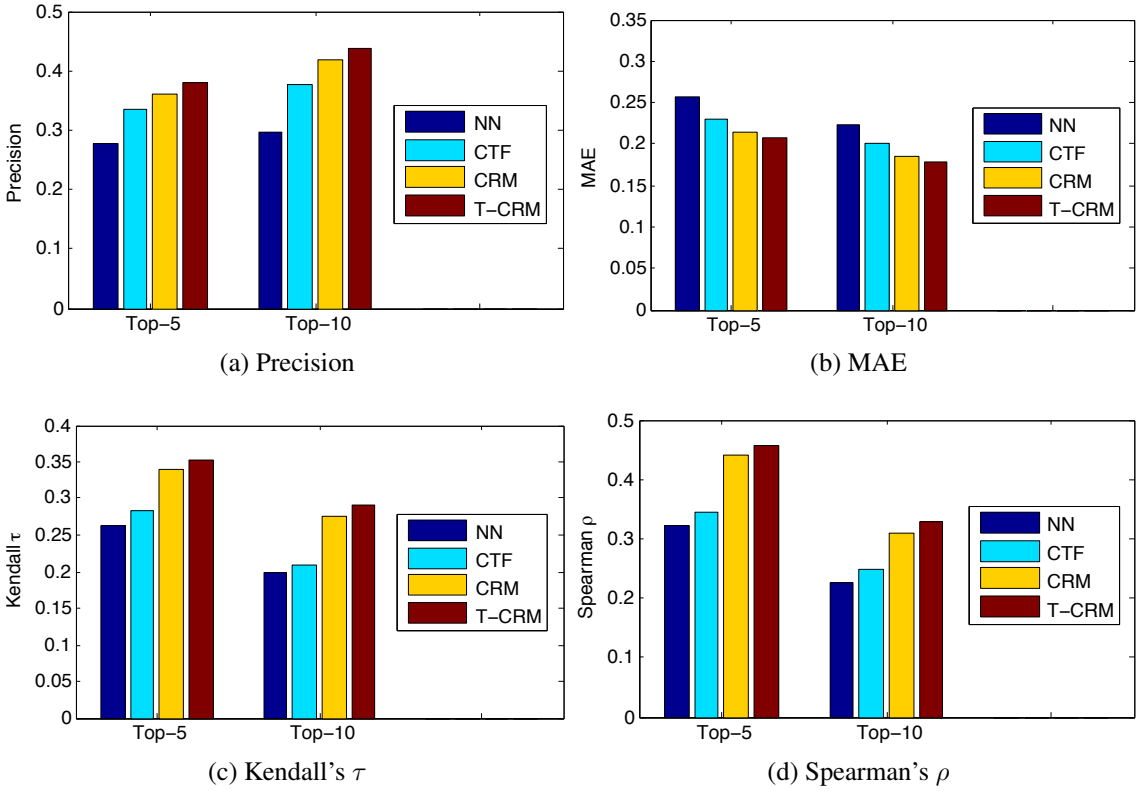
When individually using each context, we find that the behavioral context (URL sharing) always performs the best in any setting. Moreover, combining it with other contexts can always bring the biggest improvement. For instance, within the NN method, URL sharing context has up to 24% larger Spearman correlation than social context. In CRM, the MAE@10 decreases by 8% when URL sharing context is added with textual context. These results indicate the importance of capturing actual user behaviors as a critical step for identifying user topical profiles (in contrast, to relying purely on social connections

or on the content of what users post). These results support the intuition that social context may capture spurious user similarities (e.g., linking two very different users) and that textual context may insert noise into learning user topical profiles. In contrast, behavioral cues provide a clearer perspective on user's interests and expertise.

What if behavioral data is scarce? URL sharing is one of the few publicly-available sources of behavioral information, but sometimes it can still be a scarce resource because not all users will share many URLs on social media. In contrast, social and textual contexts are typically more universally available. We see in Table 6.1 that textual and social contexts can still work well even without behavioral context. For example, in CRM, textual context is only 5% behind behavioral context in precision@10, and social context has just 1% larger MAE@5 than behavioral context. These observations show that our model can still get a good performance even when we have scarce behavioral evidence. But that together, the three contexts can complement each other, leading to even better user topical profiles.

6.4.3 Evaluating CRM and T-CRM

Given the evidence of the importance of different contexts, we now turn to evaluating the two proposed models — CRM and T-CRM — versus alternatives. As we can see in Figure 6.5, both CRM and T-CRM perform better than the Nearest Neighbor (NN) and the Cross-domain Triadic Factorization (CTF) across all four evaluation metrics. For precision@5, T-CRM is 36% and 13% better than NN and CTF, respectively. For MAE@10, T-CRM outperforms NN by 20% and CTF by 11.8%. The gaps become even larger for the two ranking correlation coefficients, as we can see in Figure 6.5 (c) and 6.5 (d). These results suggest that the proposed contextual regularization models can better leverage all contexts together than either the neighborhood-based propagation or the immediate tensor decomposition. Note that the CTF method is fundamentally different from our problem



(a) Precision (b) MAE

(c) Kendall's τ (d) Spearman's ρ

Figure 6.5: Comparisons Between Proposed Models and Alternative Baselines.

setting where we cannot simply put together all heterogeneous contextual information. In contrast, we exploit latent factors to build a user similar matrix and find its graph Laplacian as a new regularization term. We show the effectiveness of this step in Section 6.4.4.

Recall that we introduced T-CRM as an extension to CRM to provide a more compact factorization and to jointly handle relationships across multiple contexts. In Figure 6.5 we find T-CRM surpasses CRM in all settings. T-CRM has an improvement of 4.2% in precision@10, 3% in MAE@5, 5.9% in Kendall correlation@10, and 3.8% in Spearman correlation@5. These findings indicate that T-CRM can better exploit the joint correlations between all heterogeneous contexts for improved learning of user topical profiles.

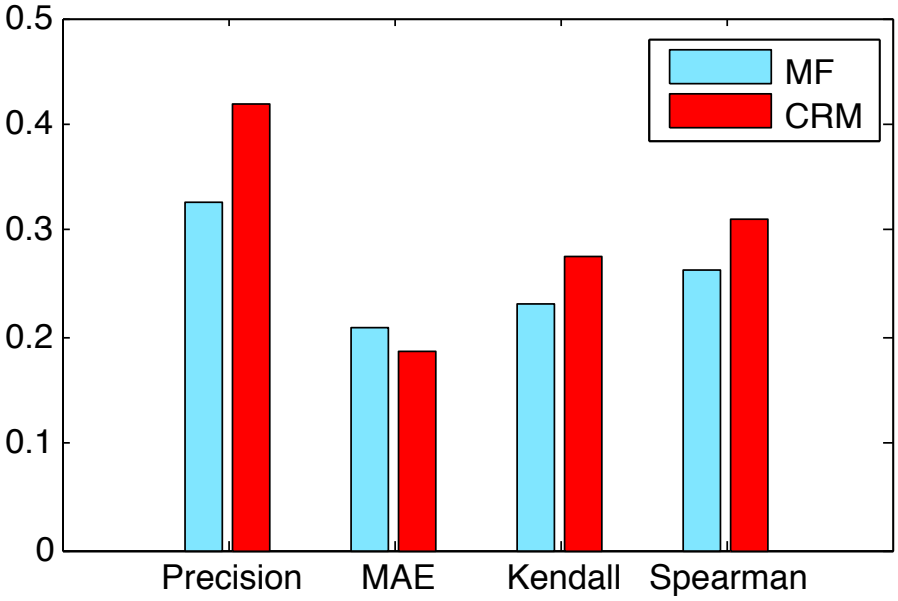


Figure 6.6: Comparisons Between CRM and Standard MF

6.4.4 Considering Other Variants

Why We Need Regularization? A natural question is why we need a regularization model. Why not just put all contexts into one large matrix and directly apply state-of-the-art matrix factorization methods? To investigate this question, we put them into one matrix upon which we adopt the standard factorization technique, and we call such a method MF. We do normalization for the data of each context since their values can have distinct scales. We follow the same evaluation methodology and show the comparisons in Figure 6.6. All results are measured at the top 10. We clearly see our CRM results in better performances than the MF in every metric. These results tell that we should take better care of the heterogeneous contexts and the regularization-based CRM is a good solution in comparison.

Why We Do Regularization After Tensor Factorization? In T-CRM, after having the latent factors of users from tensor factorization, we build a user similar matrix and find its graph Laplacian as the new regularization term. Why not just directly replace the

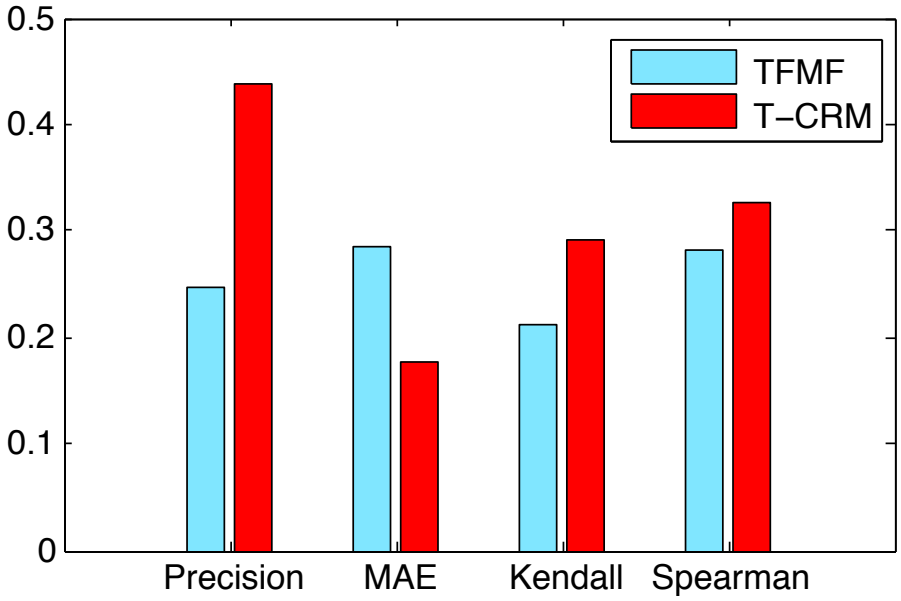
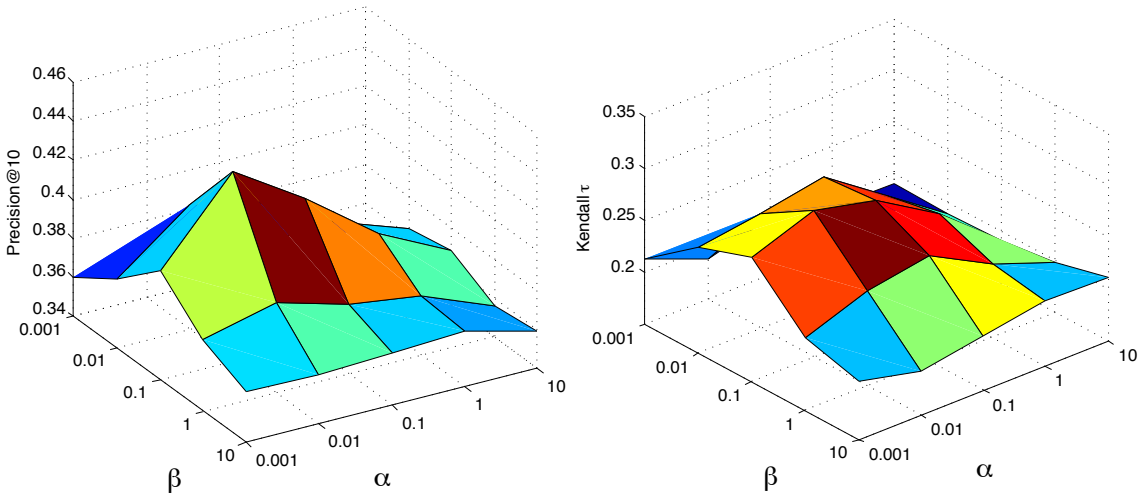


Figure 6.7: Comparisons between T-CRM and TFMF

user's latent matrix $\boldsymbol{U}$ in $\boldsymbol{X}$ after factorizing the tensor? We call such a scheme Tensor Factorization-based Matrix Factorization (TFMF), and we show the comparison results in Figure 6.7 for all metrics at top 10. Our T-CRM outperforms TFMF in all settings (e.g., 68% precison, 38% MAE, 45% Kendall correlation). These outcomes show that regularization after tensor factorization can significantly improve the performance.

Impact of Parameters. Finally, two critical parameters in T-CRM are α and β. Recall that α is used to avoid overfitting; the parameter β is used to control the contribution of similarity information between users derived from three types of contextual evidence. In order to better understand the impacts of these two parameters, we evaluate the performance of T-CRM across various parameter settings. We vary values of these parameters in [0.001 0.01, 0.1, 1, 10] and present the experimental results of precision and Kendall's τ in Figure 6.8 for learning top-10 topical profile tags of users. As we can see, T-CRM achieves relatively consistent performance when choosing these regularization parameters



(a) Precision@10 (b) Kendall's τ

Figure 6.8: Comparisons Between Proposed Models and Alternative Baselines.

across a wide range. Particularly, we find that the setting $\alpha = 0.1$ and $\beta = 0.01$ gives the best performance. These results indicate the stability of T-CRM to these parameters.

6.5 Conclusion

Mining user's topical profiles (e.g., user interests and expertise) has important applications in diverse domains such as personalized recommendation and expert detection. In this manuscript, we tackled the problem of learning user topical profiles on Twitter. In particular, we investigated how to leverage user-generated information in heterogeneous contexts. Concretely, we proposed T-CRM — a tensor-based contextual regularization model that integrates textual, social, and behavioral contexts and can be generalized to other potential contexts. By takeing into account pairwise relations among all contexts, the proposed T-CRM intelligently combines all potential benefits of each context to find the best evidence across contexts for learning high-quality user profiles. And indeed, extensive experiments demonstrate the effectiveness of T-CRM. For instance, it surpasses other alternatives up to 36% in precision@5 and 20% in MAE@10. URL sharing, as

one type of publicly-accessible user behavior, brings better results than other contexts in every evaluation setting. Moreover, compared with other variants in terms of modeling, our model also has the best performances, e.g., up to 68% for precision@10 and Kendall correlation@10.

7. DISTENC: A DISTRIBUTED SCALABLE TENSOR COMPLETION ALGORITHM

In this chapter, we tackle the challenges of **sparsity** and **scalability** in order to enable the proposed tensor learning framework to handle very large-scale datasets. In order to efficiently recover missing values in large-scale datasets that cannot fit into the memory of a single machine, we propose DISTENC, a distributed large-scale tensor completion algorithm with regularized trace of the auxiliary information based on ADMM running on the Spark framework. By efficiently handling trace-based regularization terms, updating factor matrices with caching, and optimizing the update of new tensor, DISTENC successfully tackles the high computational costs and minimizes the intermediate data. We find that DISTENC outperforms the state-of-the-art methods in many aspects including data scalability, convergence rate, machine scalability and recovering accuracy in the applications such as recommender systems and link prediction.

7.1 Introduction

With the rapid growth of data in both its velocity and volume in the real world, we are stepping into the era of *Big Data*. Extremely large and sparse multi-dimensional data arise in numerous important applications such as location-based recommendations, targeted advertising, social media search, and event detection [8, 15, 79]. Tensors – or multi-dimensional arrays – are commonly used to capture this multi-dimensionality. For instance, a movie rating from a user can be modeled as a tensor where each element is an interaction between a movie, a user, and the context in which this user rates the movie (e.g., genre, date of the rating, etc.). A multi-dimensional social network such as the DBLP network can be represented as a tensor with 4-tuples, e.g., *author-paper-term-venue*. Analytics over such large, diverse, and multi-dimensional datasets can provide valuable insights

with respect to the underlying relationships between different entities.

However, in practice, many types of multidimensional data may be *noisy* or *incomplete*, limiting the effectiveness of such analytics. For example, data may be restricted due to data sampling policies, partial access to legacy data warehouses, sparse feedback from users (e.g., ratings in a recommender system), data missing at random, and so on [119, 146]. Traditional methods like matrix completion methods have shown good success in recovering two-dimensional data, but may not be suitable for handling missing data in these large multi-dimensional cases. Analogous to matrix completion, *tensor completion* aims to recover a low-rank tensor that best approximates partially observed data and further predict the unobserved data using this low-rank tensor.

While recovering the missing values by tensor completion is attractive, it is challenging to efficiently handle large-scale tensors (e.g., ones containing billions of observations in each mode) due to the high computational costs and space requirements. Tensor completion in these scenarios faces challenges such as: (i) the intermediate data explosion problem where in updating factor matrices, the amount of intermediate data of an operation exceeds the capacity of a single machine or even a cluster [14, 46, 102, 108]; (ii) the large regularization problem where the regularization term can affect the scalability and parallelism of tensor completion [44, 45]; and (iii) since architectures on modern computing facilities have lower ratios of memory bandwidth to compute capabilities, computations on tensors that usually have unstructured access patterns are usually degraded. While there has been research addressing these challenges of scalability separately, most focus on tensor factorization, which are not suitable for tensor completion that needs to estimate all missing values in a tensor at each iteration. There is a need to fill a gap between tensor completion and applications with real large-scale datasets.

With these challenges in mind, we propose to fill this gap through DISTENC (Distributed Tensor Completion), a new distributed large-scale tensor completion algorithm

running on Apache Spark. Our intuition is to tackle the challenges of large-scale tensor completion through three key insights: (i) by designing an efficient algorithm for handling the trace-based regularization term; (ii) by updating factor matrices with caching; and (iii) by optimizing the update of the new tensor at each iteration, while minimizing the generation and shuffling of intermediate data. We analyze the proposed DISTENC that shows up to 10$\sim$1000$\times$ better scalability, performs a better linearity on machine scalability, and converges much faster than other state-of-the-art methods. We also investigate the proposed DISTENC with respect to time complexity, memory requirement and the amount of shuffled data, and find that DISTENC leads to high efficiency compared with state-of-the-art methods, while delivering similar (and in many cases improved) accuracy.

7.2 Preliminaries

In this section, we provide a brief recap on the proposed tensor learning framework based on CP-based tensor completion with auxiliary information.

7.2.1 Tensor Completion with Auxiliary Information

With the increasing ratio of missing entries, the tensor completion may perform unsatisfactory imputation with degrading accuracy due to its assumptions on low-rank and uniformly sampling. In real-world data-driven applications, besides the target tensor object, variety additional side information such as spacial and temporal similarities among objects or auxiliary coupled matrices/tensors may also exist and have potential help for improving completion quality. An example of Twitter List tensor is illustrated in Figure 7.1. Given an Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with the rank $R \ll min(I_1, \ldots, I_N)$ and similarity matrices $\mathbf{B}^{(n)}, n = 1, \ldots, N$ of size $I_1 \times I_1, \ldots$, and $I_N \times I_N$, the tensor

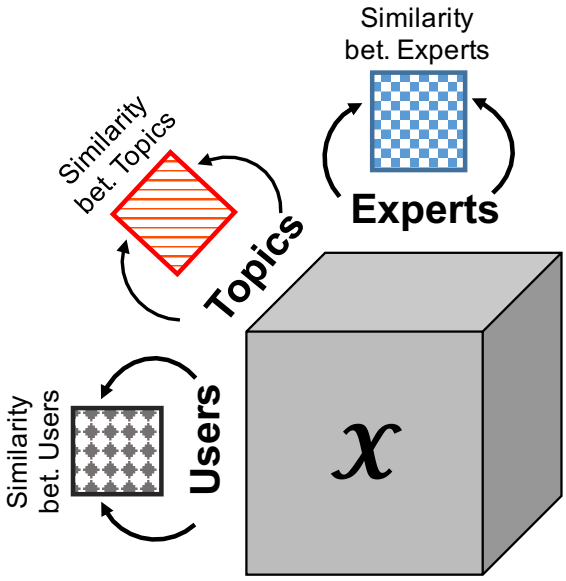


Figure 7.1: A 3-order Twitter List tensor with *user-expert-topic* triples and three similarity matrices generated from auxiliary information of users, experts and topics, respectively.

decomposition with auxiliary information solves:

$$\begin{aligned} \underset{\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)},\mathcal{X}}{\text{minimize}} \quad & \frac{1}{2}\|\mathcal{X} - [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!]\|_F^2 + \frac{\lambda}{2}\sum_{n=1}^{N}\|\mathbf{A}^{(n)}\|_F^2 \\ & + \sum_{n=1}^{N}\alpha_n \text{tr}(\mathbf{B}^{(n)^T}\mathcal{L}_n\mathbf{B}^{(n)}) \\ \text{subject to} \quad & \boldsymbol{\Omega} * \mathcal{X} = \mathcal{T}, \mathbf{A}^{(n)} = \mathbf{B}^{(n)} \geq 0, n = 1, 2, \ldots, N., \end{aligned} \tag{7.1}$$

where $\mathcal{L}_n \in \mathbb{R}^{I_n \times I_n}$ is the graph Laplacian of the similarity matrix $\mathbf{S}_n$ for the mode n, $\mathbf{B}^{(n)}, n = 1, 2, \ldots, N$ are introduced as auxiliary variables, $tr(\cdot)$ is denoted as the matrix trace and α_n is to control the weight of auxiliary information in the mode n. Figure 7.2 shows the rank-R CP tensor completion of a 3-order tensor with auxiliary information. The tensor $\mathcal{X}$ is decomposed into three factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ by taking into account auxiliary information, and recovered based on factor matrices.

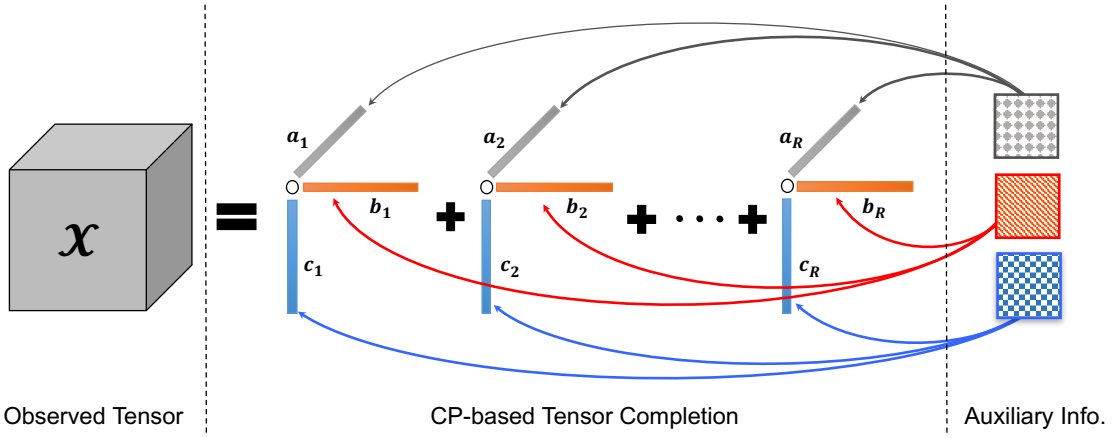


Figure 7.2: Rank-R CP tensor completion of a 3-order tensor with auxiliary information.

7.2.2 Optimization Algorithm

Since the objective function in Eq.(7.1) is not convex with respect to variables $\mathbf{A}^{(n)}$ and $\mathbf{B}^{(n)}$ together, there is no closed-form solution for this optimization problem. Motivated by methods [26, 32], an algorithm under the framework of ADMM is employed to find optimal solutions for the objective function above. ADMM [36] has illustrated its superiority over alternating least square (ALS) in terms of both reconstruction efficiency and accuracy [107]. In order to apply ADMM, the objective function Eq. (7.1) can be firstly written in the partial augmented Lagrangian functions as follow:

$$\begin{aligned} L_\eta(\mathbf{A}^{(n)}, \mathbf{B}^{(n)}, \mathbf{Y}^{(n)})_{n=1,2,\ldots,N} = & \frac{1}{2}\|\mathcal{X} - [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!]\|_F^2 \\ & + \frac{\lambda}{2}\sum_{n=1}^{N}\|\mathbf{A}^{(n)}\|_F^2 + \sum_{n=1}^{N}\frac{\alpha_n}{2}\mathrm{tr}(\mathbf{B}^{(n)^T}\mathcal{L}_n\mathbf{B}^{(n)}) \\ & + \sum_{n=1}^{N} < \mathbf{Y}^{(n)}, \mathbf{B}^{(n)} - \mathbf{A}^{(n)} > + \sum_{i=1}^{N}\frac{\eta}{2}\|\mathbf{B}^{(n)} - \mathbf{A}^{(n)}\|_F^2, \end{aligned} \tag{7.2}$$

where $\mathbf{Y}^{(n)}$ is the matrix of Lagrange multipliers for $n = 1, 2, \ldots, N$, η is a penalty parameter. The variables $\mathbf{A}^{(n)}, \mathbf{B}^{(n)}, \mathbf{Y}^{(n)}, n = 1, 2, \ldots, N$ can be iteratively updated by

Algorithm 5: CP-based Tensor Completion via ADMM

Input: $\mathcal{T}, \mathbf{A}_0^{(n)}, \mathbf{\Omega}, \mathbf{\Omega}^c, \lambda, \rho, \eta, \eta_{max}, N$
Output: $\mathcal{X}, \mathbf{A}^{(n)}, \mathbf{B}^{(n)}, \mathbf{Y}^{(n)}$

Initialize $\mathbf{A}_0^{(n)} \geq 0$, $\mathbf{B}_0^{(n)} = \mathbf{Y}_0^{(n)} = 0$, $t = 0$
while Not Converged **do**
  for $n \leftarrow 1$ ***to*** N **do**
    Update $\mathbf{B}_{t+1}^{(i)} \leftarrow (\eta_t \boldsymbol{I} + \alpha_n \boldsymbol{\mathcal{L}}_n)^{-1}(\eta_t \mathbf{A}_t^{(n)} - \mathbf{Y}_t^{(n)})$
    Calculate $\mathbf{U}_t^{(n)} \leftarrow$
    $(\mathbf{A}_t^{(N)} \odot \cdots \odot \mathbf{A}_t^{(n+1)} \odot \mathbf{A}_t^{(n-1)} \odot \cdots \odot \mathbf{A}_t^{(1)})$
    Update $\mathbf{A}_{t+1}^{(n)} \leftarrow$
    $(\mathbf{X}_{(n)}^t \mathbf{U}_t^{(n)} + \eta_t \mathbf{B}_{t+1}^{(n)} + \mathbf{Y}_t^{(n)})(\mathbf{U}_t^{(n)^T} \mathbf{U}^{(n)_t} + \lambda \boldsymbol{I} + \eta_t \boldsymbol{I})^{-1}$
  Update $\mathcal{X}^{t+1} = \mathcal{T} + \mathbf{\Omega}^c * [\![\mathbf{A}_{t+1}^{(1)}, \mathbf{A}_{t+1}^{(2)}, \ldots, \mathbf{A}_{t+1}^{(N)}]\!]$
  for $n \leftarrow 1$ ***to*** N **do**
    Update $\mathbf{Y}_{t+1}^{(n)} = \mathbf{Y}_t^{(n)} + \eta_t(\mathbf{B}_{t+1}^{(n)} - \mathbf{A}_{t+1}^{(n)})$
  Update $\eta_{t+1} = \min(\rho\eta_t, \eta_{max})$
  Check the convergence: $\max\{\|\mathbf{A}_{t+1}^{(n)} - \mathbf{B}_{t+1}^{(n)}\|_F, n = 1, 2, \ldots, N\} < tol$
  $t = t + 1$
return $\mathcal{X}$, $\mathbf{A}^{(n)}, n = 1, 2, \ldots, N$

calculating the partial derivatives with fixing other variables, as shown in **Algorithm 5** detailed in [146]. There are many ways to check for convergence. The stopping criterion for Algorithm 5 is either one of the following: i) the maximal difference between factor matrices of consecutive iterations is smaller than a threshold; ii) the maximum number of iterations is exceeded.

7.3 Proposed Method

In this section, we present our proposed method DISTENC which is a distributed algorithm for scalable tensor completion on Spark.

7.3.1 Overview

DISTENC provides an efficient distributed algorithm for the CP-based tensor completion with auxiliary information on Spark. The **Algorithm 5** contains three challenging operations: (i) updating auxiliary variables $\mathbf{B}^{(n)}$ (line 4); (ii) updating factor matrices $\mathbf{A}^{(n)}$ (lines 7 and 8); and (iii) updating tensor $\mathcal{X}$ (line 9). In the following subsections, we address the above challenges with the following main ideas that efficiently update auxiliary variables, factor matrices and tensors in distributed systems, while reducing floating point operations (FLOPs).

- (Section 7.3.2) Eigen-decomposing a graph Laplacian matrix and carefully ordering of computation to decrease FLOPs in updating auxiliary variables.
- (Section 7.3.3) Carefully partitioning of the workload and distributing intermediate generation to remove redundant data generation and reducing the amount of intermediate data transfer in updating factor matrices.
- (Section 7.3.4) Utilizing the residual tensor to avoid the explicit computation of the dense tensor and reuse intermediate data to decrease FLOPs in updating tensor.

7.3.2 Calculating Inverse of Graph Laplacian Matrices

Since the update rules for auxiliary variables $\mathbf{B}^{(n)}, n = 1, 2, \ldots, N$ are similar, we focus on updating the variable $\mathbf{B}^{(n)}$ where n could be an arbitrary one from $\{1, 2, \ldots, N\}$. The operation in line 4 of **Algorithm 5** requires us to compute the pseudo-inverse of the summation of a matrix $\alpha_n \mathcal{L}_n$ and a diagonal matrix $\eta_t \boldsymbol{I}$ where $\boldsymbol{I}$ is an identity matrix with the same size of $\mathcal{L}_n$. Since such summation will change with the penalty parameter η_t that will be updated at every iteration, the question is how to efficiently calculate such a pseudo-inverse instead of computing it at every iteration due to its high computational cost with complexity $\mathcal{O}(I_n^3)$.

As a graph Laplacian matrix $\boldsymbol{\mathcal{L}}_n$ derived from the similarity matrix $\mathbf{S}_n$ is symmetric and predefined without any change in **Algorithm 5**, we apply an efficient truncated eigen-decomposition method proposed by Bientinest et al. [147] with the time complexity $\mathcal{O}(KI_n)$ and the space complexity $\mathcal{O}(I_n)$ to it and get its truncated decomposition as $\boldsymbol{\mathcal{L}}_n = \mathbf{V}_n\boldsymbol{\Lambda}_n\mathbf{V}_n^T$ where $\mathbf{V}_n \in \mathbb{R}^{I_n \times K}$ and $\boldsymbol{\Lambda}_n \in \mathbb{R}^{K \times K}$. Hence, line 4 of **Algorithm 5** can be re-written as follow:

$$\mathbf{B}_{t+1}^{(n)} \leftarrow \mathbf{V}_n(\eta_t + \alpha_n\Lambda_n)^{-1}\mathbf{V}_n^T(\eta_t\mathbf{A}_t^{(n)} - \mathbf{Y}_t^{(n)}). \tag{7.3}$$

Since $(\eta_t + \alpha_n\boldsymbol{\Lambda}_n)$ is a diagonal matrix whose inverse, its inverse can be efficiently computed by only computing reciprocal of entries on the diagonal instead of calculating the inverse of the whole matrix $(\eta_t\boldsymbol{I} + \alpha_n\boldsymbol{\mathcal{L}}_n)$. Furthermore, Eq.(7.3) performs the matrix multiplication of the four matrices $\mathbf{V}_n$, $(\eta_t + \alpha_n\Lambda_n)^{-1}$, $\mathbf{V}_n^T$ and $(\eta_t\mathbf{A}_t^{(n)} - \mathbf{Y}_t^{(n)})$. Its computing order may significantly affect the efficiency of calculation for updating $\mathbf{B}^{(n)}$. In order to reduce FLOPs, we compute it by firstly multiplying the last two matrices that result in a relatively small matrix with the size $K \times R$, and broadcasting the result with the second one to the first matrix:

$$\mathbf{B}_{t+1}^{(n)} \leftarrow \mathbf{V}_n(\eta_t + \alpha_n\Lambda_n)^{-1}(\mathbf{V}_n^T(\eta_t\mathbf{A}_t^{(n)} - \mathbf{Y}_t^{(n)})). \tag{7.4}$$

By this way, it is able to perform the update of an auxiliary variable $\mathbf{B}^{(n)}$ in $\mathcal{O}(I_nR + I_nKR + I_nK^2R)$ time.

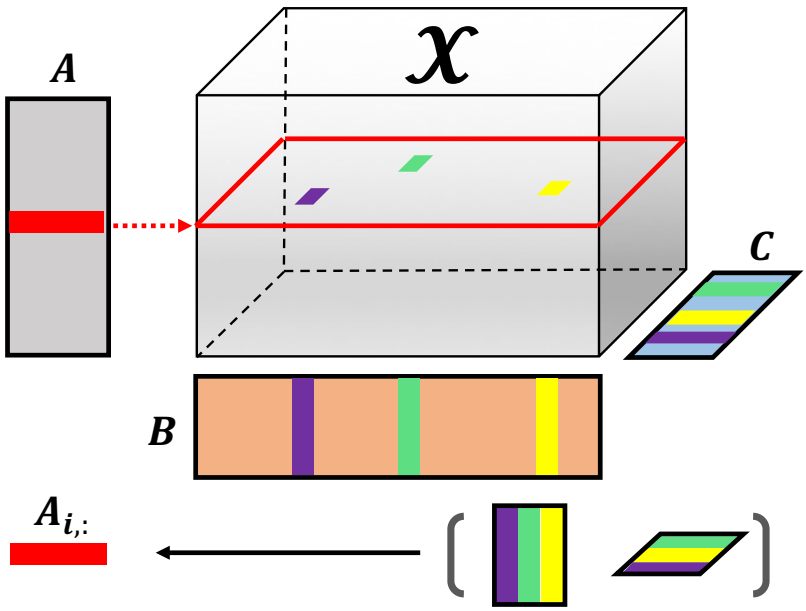


Figure 7.3: Memory access during the update of $\mathbf{A}_{i,:}$ in a 3-order tensor.

7.3.3 Reducing Intermediate Data

As shown in lines 7 and 8 in **Algorithm 5**, we focus on the updating rule for an arbitrary $\mathbf{A}^{(n)}$ as follow:

$$\mathbf{A}^{(n)} \leftarrow (\mathbf{X}_{(n)}\mathbf{U}^{(n)} + \eta\mathbf{B}^{(n)} + \mathbf{Y}^{(n)})(\mathbf{U}^{(n)^T}\mathbf{U}^{(n)} + \lambda\boldsymbol{I} + \eta\boldsymbol{I})^{-1}. \tag{7.5}$$

where $\mathbf{U}^{(n)} = (\mathbf{A}^{(N)} \odot \cdots \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \cdots \odot \mathbf{A}^{(1)})$ with size $(\prod_{k\neq n} I_k) \times R$, which entails three matrix-matrix multiplications as:

$$\begin{aligned} \boldsymbol{\mathcal{H}}_1 &= \mathbf{X}_{(n)}\mathbf{U}^{(n)}, \\ \boldsymbol{\mathcal{H}}_2 &= (\mathbf{U}^{(n)^T}\mathbf{U}^{(n)} + \lambda\boldsymbol{I} + \eta\boldsymbol{I})^{-1}, \\ \boldsymbol{\mathcal{H}}_3 &= (\boldsymbol{\mathcal{H}}_1 + \eta\mathbf{B}^{(n)} + \mathbf{Y}^{(n)})\boldsymbol{\mathcal{H}}_2. \end{aligned} \tag{7.6}$$

We denote $\boldsymbol{\mathcal{H}}_1 = \mathbf{X}_{(n)}\mathbf{U}^{(n)}$ as the *matricized tensor times Khatri-Rao product* (MTTKRP) that will lead to the intermediate data explosion problem in the tensor completion when tensor $\boldsymbol{\mathcal{X}}$ is very large. Explicitly calculating $\mathbf{U}^{(n)}$ and performing the matrix multiplication with $\mathbf{X}_{(n)}$ requires more memory than what a common cluster can afford as computing $\mathbf{U}^{(n)}$ is prohibitively expensive with the size $(\prod_{k\neq n} I_k) \times R$. Though the matricized $\mathbf{X}_{(n)}$ is very sparse, $\mathbf{U}^{(n)}$ is very large and dense. Hence, inspired by the work [108], we perform MTTKRP in place by exploiting the block structure of the Khatri-Rao product. For a better illustration, we assume that $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is a 3-order sparse tensor whose entry $\boldsymbol{\mathcal{H}}_1(i_1, r)$ can be represented as:

$$\boldsymbol{\mathcal{H}}_1(i_1, r) = \sum_{\boldsymbol{\mathcal{X}}_{i_1,:,:}} \boldsymbol{\mathcal{X}}_{i_1,i_2,i_3} \mathbf{A}^{(3)}_{i_3,r} \mathbf{A}^{(2)}_{i_2,r} \tag{7.7}$$

As shown in Eq.(7.7), we observe two important properties of MTTKRP: i) non-zeros in $\boldsymbol{\mathcal{X}}_{i_1,:,:}$ are only associated with the computation of $\boldsymbol{\mathcal{H}}_1(i_1, :)$; ii) the row indices i_2 and i_3 in $\mathbf{A}^{(2)}$ and $\mathbf{A}^{(3)}$, respectively, will be accessed based upon which appear in non-zeros in $\boldsymbol{\mathcal{X}}_{i_1,:,:}$ when calculating $\boldsymbol{\mathcal{H}}_1(i_n, r)$. Thus, our idea is to *compartmentalize* the sparse tensor $\boldsymbol{\mathcal{X}}$ and factor matrices $\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \cdots, \mathbf{A}^{(N)}$ into blocks in order to make the computation of MTTKRP fit into the memory. Taking a 3-order tensor as an example, we divide rows of $\mathbf{A}^{(1)}$, $\mathbf{A}^{(2)}$ and $\mathbf{A}^{(3)}$ into P, Q, and K blocks, respectively. Correspondingly, the tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ can be further divided into $P \times Q \times K$ blocks. A block of tensor is denoted as $\boldsymbol{\mathcal{X}}(p, q, k)$ with corresponding blocks of factor matrices $\mathbf{A}^{(1)}_{(p)}$, $\mathbf{A}^{(2)}_{(q)}$ and $\mathbf{A}^{(3)}_{(k)}$. Each process only works on a block of a factor matrix with entries in the tensor with which this block is associated, and aggregate partial results computed by other processors for this block.

Algorithm 6: DISTENC-Greedy Algorithm for Load Balance

Input: observed tensor $\mathcal{T}$, number of partitions at n-mode P_n, number of modes N
Output: $\boldsymbol{w}_n, n = 1, \cdots, N$

for $n = 1, \cdots, N$ **do**
  $\delta = nnz(\mathcal{T})/P_n \leftarrow$ Calculate the chunk size for n-mode;
  $sum = 0$ and $\epsilon_{pre} = \delta$;
  $\boldsymbol{\theta}^{(n)} \leftarrow$ Calculate nnz for each slice at n-mode;
  for $i = 1, \cdots, I_n$ **do**
    $sum \leftarrow sum + \boldsymbol{\theta}_i^{(n)}$;
    $\epsilon \leftarrow$ Calculate the difference between sum and δ;
    if $sum \geq \delta$ **then**
      $\boldsymbol{w}_n \leftarrow$ add i if $\epsilon < \epsilon_{pre}$; otherwise, add $i - 1$;
    $\epsilon_{pre} = \epsilon$;

7.3.3.1 Load Balancing

Since the tensor $\mathcal{X}$ is very sparse, randomly dividing it into $P \times Q \times K$ blocks could result in *load imbalance* [148]. A block is defined as a unit of workload distributed across machines, which determines the level of parallelism and the amount of shuffled data. The questions is how to identify the block boundaries. In order to fully utilize the computing resources, a greedy algorithm is proposed to generate blocks for balancing the workload. For instance, we split a mode into P partitions. Each partition will be generated by continuously adding indices until the number of non-zeros in this partition is equal to or larger than $nnz(\mathcal{X})/P$ that is considered as the target partition size. Once adding a slice makes a partition over the target size, we compare the number of non-zeros in this partition before and after adding it and pick whichever is closer to the target size. Other modes would follow the same routine to identify boundaries for Q partitions and K partitions. The algorithm for balancing the load for DISTENC is demonstrated in **Algorithm 6**, which will take $\mathcal{O}(Nnnz(\mathcal{X}))$.

7.3.3.2 *Computing MTTKRP*

After compartmentalizing the tensor and factor matrices into blocks, each process holds the tensor non-zeros with necessary blocks of factor matrices (non-local factor matrix rows will be transfered to this process from others), and performs MTTKRP as shown in Eq.(7.7). Specifically, we parallelize such computation based on the efficient fiber-based data structure [108] in local, indicating that we directly calculate the row of $\boldsymbol{\mathcal{H}}_1$ as follow:

$$\boldsymbol{\mathcal{H}}_1(i_1, :) = \sum_{\mathcal{X}_{i_1,:,:}} \mathcal{X}_{i_1,i_2,i_3}(\mathbf{A}^{(3)}_{i_3,:} * \mathbf{A}^{(2)}_{i_2,:}). \tag{7.8}$$

Since such calculation can be done at the granularity of factor matrix rows, it only requires $\mathcal{O}(R)$ intermediate memory per thread in parallelism. By this way, $\boldsymbol{\mathcal{H}}_1$ are row-wise computed and distributed among all processes. We only need to broadcast relatively small factor matrices along with corresponding indices in the non-zero elements of a sparse tensor for each machine instead of having to compute and materialize the entire Khatri-Rao product.

7.3.3.3 *Calculating* $\mathbf{U}^{(n)^T}\mathbf{U}^{(n)}$

Based upon the property of Khatri-Rao product, we can re-write $\mathbf{U}^{(1)^T}\mathbf{U}^{(1)}$ as follow:

$$\mathbf{U}^{(1)^T}\mathbf{U}^{(1)} = \mathbf{A}^{(3)^T}\mathbf{A}^{(3)} * \mathbf{A}^{(2)^T}\mathbf{A}^{(2)}. \tag{7.9}$$

By this way, it avoids explicitly computing the large intermediate matrix $\mathbf{U}^{(1)}$ with the size $I_2I_3 \times R$ by calculating the self-products $\mathbf{A}^{(2)^T}\mathbf{A}^{(2)}$ and $\mathbf{A}^{(3)^T}\mathbf{A}^{(3)}$ of factor matrices $\mathbf{A}^{(2)}$ and $\mathbf{A}^{(3)}$. With applying the block matrix form, the computation of $\mathbf{A}^{(1)^T}\mathbf{A}^{(1)}$ can be

represented in a distributed fashion as:

$$\mathbf{A}^{(1)^T}\mathbf{A}^{(1)} = \sum_{p=1}^{P} \mathbf{A}_{(p)}^{(1)^T}\mathbf{A}_{(p)}^{(1)}. \tag{7.10}$$

Each process calculates a local $\mathbf{A}_{(p)}^{(1)^T}\mathbf{A}_{(p)}^{(1)}$ in the thread-level parallelism. By aggregating all computations across processes, the final matrix $\mathbf{A}^{(1)^T}\mathbf{A}^{(1)}$ will be generated and distributed among all processes. Since it is a matrix of size $R \times R$ that can easily fit into the memory of each process. Thus, it can be seen that $(\boldsymbol{\mathcal{H}}_2 + \lambda \boldsymbol{I} + \eta \boldsymbol{I})^{-1}$ can be efficiently calculated in $\mathcal{O}(R^3)$ time in a single machine.

7.3.4 Computing the Updated Tensor

Unlike the tensor factorization/decomposition in which the input tensor is fixed, tensor completion requires to update the tensor $\boldsymbol{\mathcal{X}}$ by filling out unobserved elements in each iteration as shown in line 9 in **Algorithm 5**. Once completing updates of $\mathbf{A}^{(1)}, \cdots, \mathbf{A}^{(N)}$ in a iteration, unobserved elements in a sparse tensor will be filled out by estimated values. Thus, it turns out to be a dense tensor that leads to a significant increase in the computation of updating factor matrices in lines 7 and 8 in **Algorithm 5**. The question is how to avoid such problem and keep the computation in the level of $\mathcal{O}(nnz(\boldsymbol{\mathcal{X}}))$ time. First of all, we define the residual tensor as:

$$\boldsymbol{\mathcal{E}} = \boldsymbol{\Omega} * (\boldsymbol{\mathcal{T}} - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]), \tag{7.11}$$

which is sparse with the same size of the observed sparse tensor $\boldsymbol{\mathcal{T}}$. Based upon the definition of *tensor matricizied*, its *n*-mode matricized form can be expressed as:

$$\boldsymbol{\mathcal{X}}_{(n)} \approx \mathbf{A}^{(n)}(\mathbf{A}^{(N)} \odot \cdots \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n+1)} \odot \cdots \mathbf{A}^{(1)})^T. \tag{7.12}$$

For the brevity, we take a 3-order tensor as an example as demonstrated in the previous section. By leveraging the residual tensor and the 1-mode matricized form of a tensor, we can re-write $\boldsymbol{\mathcal{H}}_1^{t+1}$ shown in the Eq.(7.6) as:

$$\begin{aligned}
\boldsymbol{\mathcal{H}}_1^{t+1} &= \mathbf{X}_{(1)}^t \mathbf{U}_t^{(1)} \\
&= (\llbracket \mathbf{A}_t^{(1)}, \mathbf{A}_t^{(2)}, \mathbf{A}_t^{(3)} \rrbracket_{(1)} + \boldsymbol{\mathcal{E}}_{(1)}^t) \mathbf{U}_t^{(1)} \\
&= \llbracket \mathbf{A}_t^{(1)}, \mathbf{A}_t^{(2)}, \mathbf{A}_t^{(3)} \rrbracket_{(1)} \mathbf{U}_t^{(1)} + \boldsymbol{\mathcal{E}}_{(1)}^t \mathbf{U}_t^{(1)} \\
&= \mathbf{A}_t^{(1)} (\mathbf{A}_t^{(3)} \odot \mathbf{A}_t^{(2)})^T (\mathbf{A}_t^{(3)} \odot \mathbf{A}_t^{(2)}) + \boldsymbol{\mathcal{E}}_{(1)}^t \mathbf{U}_t^{(1)} \\
&= \mathbf{A}_t^{(1)} ({\mathbf{U}_t^{(1)}}^T \mathbf{U}_t^{(1)}) + \boldsymbol{\mathcal{E}}_{(1)}^t \mathbf{U}_t^{(1)}
\end{aligned} \tag{7.13}$$

It can be seen that $\boldsymbol{\mathcal{H}}_1^{t+1}$ consists of two parts that are able to reduce the time complexity to $\mathcal{O}(nnz(\boldsymbol{\mathcal{X}}))$. Concretely, the first part $\mathbf{A}_t^{(1)} ({\mathbf{U}_t^{(1)}}^T \mathbf{U}_t^{(1)})$ takes $\mathcal{O}((I_1+I_2+I_3)R^2)$ FLOPs as shown in the computation of $\boldsymbol{\mathcal{H}}_2$ of the section 7.3.3; the second part $\boldsymbol{\mathcal{E}}_{(1)}^t \mathbf{U}_t^{(1)}$ is performed by the method illustrated in the section 7.3.3 with the complexity $\mathcal{O}(nnz(\boldsymbol{\mathcal{X}}))$ since it is only related to the residual tensor $\boldsymbol{\mathcal{E}}$ instead of using a updated dense tensor. Fortunately, each ${\mathbf{U}_t^{(1)}}^T \mathbf{U}_t^{(1)}$ is computed during the iteration and the results can be cached and reused in only $\mathcal{O}(R^2)$ space.

7.3.5 Complexity Analysis

We now analyze the proposed DISTENC algorithm with respect to time complexity, memory requirement and data communication. Its cost is bounded by MTTKRP and its associated communications. For the sake of simplicity, we take a N-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times \cdots \times I}$ as the input tensor. We denote M as the number of machines, p as the number of partitions for one mode, $P = p \times p \times p$ as the number of blocks in a tensor and K as the number of components in the eigen-decomposition of a graph Laplacian matrix $\boldsymbol{\mathcal{L}}$.

Lemma 1. *The time complexity of* DISTENC *is* $\mathcal{O}(nnz(\boldsymbol{\mathcal{X}}) + NI + NIR + Rnnz(\boldsymbol{\mathcal{X}}) +$

Algorithm 7: DisTenC Algorithm

Input: $\mathcal{T}, \mathbf{A}_0^{(n)}, \mathbf{\Omega}, \lambda, \rho, \eta, \eta_{max}, N$
Output: $\mathcal{X}, \mathbf{A}^{(n)}, \mathbf{B}^{(n)}, \mathbf{Y}^{(n)}$

for $n \leftarrow 1$ ***to*** N **do**
  $\boldsymbol{w}_n \leftarrow GreedyAlgorithm(\mathcal{X})$ introduced in Algorithm 6
Partition $\mathcal{X}$ based upon $\boldsymbol{w}_n, n = 1, \cdots, N$
Initialize $\mathbf{A}_0^{(n)} \geq 0$, $\mathbf{B}_0^{(n)} = \mathbf{Y}_0^{(n)} = 0$, $t = 0$
Calculate the residual tensor $\mathcal{E}^0 = \mathbf{\Omega} * (\mathcal{T} - [\![\mathbf{A}_0^{(1)}, \ldots, \mathbf{A}_0^{(N)}]\!])$
for $t \leftarrow 0$ ***to*** T **do**
  for $n \leftarrow 1$ ***to*** N **do**
    Update $\mathbf{B}_{t+1}^{(i)} \leftarrow (\eta_t \boldsymbol{I} + \alpha_n \mathcal{L}_n)^{-1}(\eta_t \mathbf{A}_t^{(n)} - \mathbf{Y}_t^{(n)})$
    Calculate and cache $\mathcal{F}_n^t = \mathbf{U}_t^{(n)^T} \mathbf{U}_t^{(n)}$
    Calculate $\mathbf{H}_n^t = MTTKRP(\mathcal{E}_{(n)}^t \mathbf{U}_t^{(n)})$
    Update and cache $\mathbf{A}_{t+1}^{(n)} \leftarrow$
     $(\mathbf{A}_t^{(n)} \mathcal{F}_n^t + \mathbf{H}_n^t + \eta_t \mathbf{B}_{t+1}^{(n)} + \mathbf{Y}_t^{(n)})(\mathcal{F}_n^t + \lambda \boldsymbol{I} + \eta_t \boldsymbol{I})^{-1}$
    Update and cache $\mathbf{Y}_{t+1}^{(n)} = \mathbf{Y}_t^{(n)} + \eta_t(\mathbf{B}_{t+1}^{(n)} - \mathbf{A}_{t+1}^{(n)})$
  Calculate and cache the residual tensor $\mathcal{E}^{t+1} = \mathbf{\Omega} * (\mathcal{T} - [\![\mathbf{A}_t^{(1)}, \ldots, \mathbf{A}_t^{(N)}]\!])$
  Update $\eta_{t+1} = \min(\rho \eta_t, \eta_{max})$
  Check the convergence: $\mathbf{max}\{\|\mathbf{A}_{t+1}^{(n)} - \mathbf{A}_t^{(n)}\|_F^2\} < tol$
  if *converged* **then**
    break out of **for** loop
return $\mathcal{X}$, $\mathbf{A}^{(n)}, n = 1, 2, \ldots, N$

$N(IR + IKR + IK^2R) + N(IR^2 + \lceil nnz(\mathcal{X})/I \rceil R + 3IR + R^3) + NIR)$.

Proof. In the beginning, the tensor $\mathcal{X}$ is split into P blocks by applying Algorithm 6. For each mode, computing the number of non-zero elements in slices takes $\mathcal{O}(nnz(\mathcal{X}))$ time via incremental computations that employ prior summation results. Identifying the partition boundaries for each mode takes $\mathcal{O}(I)$ time. Since a non-zero element is determined and mapped to a machine in a constant time based on identified boundaries, decentralizing all non-zero elements in $\mathcal{X}$ to blocks in machines takes $\mathcal{O}(nnz(\mathcal{X}))$. In total, partitioning the sparse tensor takes $\mathcal{O}(nnz(\mathcal{X}) + NI)$. After splitting the tensor

into blocks and mapping all non-zero elements to blocks, the factor matrices are randomly initialized and distributed among machines based upon block boundaries identified during the split of the tensor. This process takes $\mathcal{O}(NIR)$ time. The residual tensor $\mathcal{E}$ is sparse with the same number of non-zero elements as the input sparse tensor $\mathcal{X}$. Bounded by the non-negative weight tensor $\mathbf{\Omega}$, calculating the residual tensor takes $\mathcal{O}(Rnnz(\mathcal{X}))$ time as an entry of $[\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$ can be obtained in $\mathcal{O}(R)$ time. Pre-computing the truncated eigen-decomposition of a graph Laplacian matrix $\mathcal{L}_n$ for n-mode takes $\mathcal{O}(KI)$ time. The error between a factor matrix $\mathbf{A}^{(n)}$ and a matrix of Lagrange multipliers $\mathbf{Y}^{(n)}$ can be computed in $\mathcal{O}(RI)$ time. Based upon the order of updating $\mathbf{B}^{(n)}$ introduced in the section 7.3.2, computing the multiplication of last two matrices $\mathbf{V}_n^T(\eta_t \mathbf{A}_t^{(n)} - \mathbf{Y}_t^{(n)})$ takes $\mathcal{O}(IKR)$. Due to the relatively small size ($K \times R$) of the result, we broadcast it with $(\eta_t + \alpha_n \Lambda_n)^{-1}$ of size $K \times K$ to the first matrix $\mathbf{V}_n$ and compute their multiplication in $\mathcal{O}(IK^2R)$ time. In total, updating an auxiliary variable takes $\mathcal{O}(IR+IKR+IK^2R)$ time, and $\mathcal{O}(N(IR+IKR+IK^2R))$ time for all modes. The update of a factor matrix contains three steps. The self-product $\mathbf{A}^{(n)^T}\mathbf{A}^{(n)}$ for n-mode requires to be performed in $\mathcal{O}(IR^2)$ time. Through all N modes in the tensor, it takes $\mathcal{O}(NIR^2)$ time. In each mode, computing MTTKRP of the residual tensor and factor matrices as shown in line 7 of Algorithm 7 by the proposed row-wise method takes $\mathcal{O}(\lceil nnz(\mathcal{X})/I \rceil R)$. As illustrated in line 8 of Algorithm 7, updating a factor matrix performs a multiplication of two matrices. The first one can be obtained in $\mathcal{O}(IR^2+3IR)$ time. For the second one, it takes $\mathcal{O}(R^3)$ to calculate the inverse of the matrix $(\mathcal{F}_n^t + \lambda \boldsymbol{I} + \eta_t \boldsymbol{I})$. The multiplication of these two matrices takes $\mathcal{O}(IR^2)$. Thus, updating a factor matrix takes $\mathcal{O}(IR^2 + \lceil nnz(\mathcal{X})/I \rceil R + 3IR + R^3)$. In total, it takes $\mathcal{O}(N(IR^2 + \lceil nnz(\mathcal{X})/I \rceil R + 3IR + R^3))$. Updating the matrix of Lagrange Multiplier $\mathbf{Y}^{(n)}$ takes $\mathcal{O}(IR)$ time. Checking the convergence criterion requires $\mathcal{O}(NIR)$ to be performed. □

Lemma 2. *The amount of memory required by* DISTENC *is* $\mathcal{O}(nnz(\mathcal{X})+3NIR+NIK+NK+MNR^2)$.

Proof. During the computation, DISTENC needs to store data in memory at each iteration as follows: the observed tensor $\mathcal{X}$, the residual tensor $\mathcal{E}$, factor matrices $\mathbf{A}^{(1)}$, $\mathbf{A}^{(2)}$, and $\mathbf{A}^{(3)}$, auxiliary variables $\mathbf{B}^{(1)}$, $\mathbf{B}^{(2)}$, and $\mathbf{B}^{(3)}$, Lagrange multiplier matrices $\mathbf{Y}^{(1)}$, $\mathbf{Y}^{(2)}$, and $\mathbf{Y}^{(3)}$, eigen-decomposed graph Laplacian matrix $\mathcal{L}_n = \mathbf{V}_n\mathbf{\Lambda}_n\mathbf{V}_n^T$, and the self-product $\mathbf{A}^{(n)^T}\mathbf{A}^{(n)}$ for n-mode. Since the residual tensor $\mathcal{E}$ is calculated only for those non-zero elements in the observed tensor $\mathcal{X}$, both of them are kept in the memory at each iteration with a distributed fashion, which require $\mathcal{O}(nnz(\mathcal{X}))$ memory. For each mode, its factor matrix $\mathbf{A}^{(n)}$ has the same size as its auxiliary variable $\mathbf{B}^{(n)}$ and Lagrange multiplier matrix $\mathbf{Y}^{(n)}$. Thus, the total amount of memory used for storing them for all modes is $\mathcal{O}(3NIR)$. The Laplacian matrix $\mathcal{L}_n$ for n-mode is eigen-decomposed into an eigenvector matrix $\mathbf{V}_n$ and a diagonal matrix $\mathbf{\Lambda}_n$ that is stored as a vector in the machine. By considering all modes in a tensor, the memory is required to hold $\mathcal{O}(NIK + NK)$ space. The self-product $\mathbf{A}^{(n)^T}\mathbf{A}^{(n)}$ for n-mode only takes $\mathcal{O}(R^2)$ memory. Since we will broadcast it to all M machines, the amount of memory for storing these self-products for all modes is $\mathcal{O}(MNR^2)$. □

Lemma 3. *The amount of shuffled data caused by* DISTENC *is* $\mathcal{O}(nnz(\mathcal{X}) + TNMIR + TNMR^2)$

Proof. DISTENC initially employed the greedy algorithm to identify the partition boundaries for each mode, and partitions the observed tensor $\mathcal{X}$ into defined groups. In this process, the whole input tensor is shuffled and cached across all machines. Thus, the amount of shuffled data in the partition of the observed tensor is $\mathcal{O}(nnz(\mathcal{X}))$. In each iteration, DISTENC requires to send rows of factor matrices, auxiliary variables and Lagrange multiplier matrices to corresponding partitions in which each tensor entry is updated by rows

associated with its index in all modes, which takes $\mathcal{O}(NMIR)$ space in total. Moreover, the self-product $\mathbf{A}^{(n)^T}\mathbf{A}^{(n)}$ for n-mode is calculated by aggregating all matrices of size $R \times R$ across machines, and then broadcast back to all machines. In this process, it takes $\mathcal{O}(NMR^2)$ by considering all modes in the tensor. Similarly, updating the residual tensor $\mathcal{E}$ needs to copy associated rows of factor matrices into machines, which takes $\mathcal{O}(NMIR)$ space in sum. Therefore, by considering all cases above, the amount of shuffled data for DISTENC after T iterations is $\mathcal{O}(nnz(\mathcal{X}) + TNMIR + TNMR^2)$. □

7.3.6 Implementation on Spark

In this section, we explore practical issues in terms of implementations of DisTenC on Spark. Our implementation is carefully designed to obtain best speed-up and scalability. Since the input tensor is sparse, all entries are stored in a list with the coordinate format (COO). The input sparse tensor is loaded as RDDS. First of all, we apply functions functions `map` and `reduceByKey` to calculate the number of non-zero elements for all indices in a mode with the key that is the index in that mode. These count results are then used to generate partition boundaries for that mode and `persist`ed in memory. After that, we apply functions `map` and `aggregateByKey` to partition the tensor into blocks: `map` transforms an entry of the sparse tensor into an element in the RDD whose key is a block ID; `aggregateByKey` groups these non-zero elements by block IDs. Partitioned tensor RDDs are then `persist`ed in memory. In order to speed-up the following computation, for each mode, we transform a tensor to a pair RDD whose key is an index in that mode and value is all block IDs with which entries associated with this index appear in blocks by employing RDD's functions `flatMap` and `reduceByKey`, and `persist` them in memory. Factor matrices are initialized with random numbers, which are stored as RDDs and distributed based upon partition boundaries identified previously. Following the same fashion, matrices of Lagrange multipliers are initialized with

zeros as RDDs. After applying the efficient truncated eigen-decomposition method, a graph Laplacian matrix is decomposed into eigenvalues and eigenvectors. As shown in the section 7.3.2, a diagonal matrix of eigenvalues is stored as a `Array` and broadcast to all machines in the cluster; eigenvectors are stored as RDDs where the key is the index and the value is the associated eigenvector with the same partition as factor matrices. The self-product of a factor matrix is transformed from a factor matrix RDD by utilizing functions `flatMap` and `reduceByKey` and broadcast to all machines. We update factor matrices as well as auxiliary variables and by using RDD's functions `flatMap`, `join` and `reductByKey`. Since the operation `join` will shuffle the data and exponentially increase the computational time, we keep the same partitions when applying `join` to two RDDs. In the implementation, we also replace operations `groupByKey` by `reduceByKey` and `combineByKey` that combines pairs with the same key on the same machine for efficiency. We also limit the number of `combineByKey` operations so that edges of the same element are available at the same physical location, minimizing data shuffling. As it can be seen, we cache reused RDD in memory in order to minimize disk accesses between consecutive iterations, which would not be possible if using a system like Hadoop to distribute the computation.

7.4 Experiments

To evaluate the proposed DISTENC, we experimentally perform experiments to answer the following questions:

Q1: Data Scalability. How well do DISTENC and other baseline methods scale up with the input tensor in terms of factors such as the number of non-zeros, dimension, mode length, and rank?

Q2: Machine Scalability. How well does DISTENC scale up in terms of th number of machines?

Q3: Discovery. How accurately do DISTENC and other baseline methods perform given the real-world tensor data?

7.4.1 Experimental Setup

7.4.1.1 Cluster/Machines

DISTENC is implemented on a 10-node Spark cluster in which each node has a quad-core Intel Xeon E5410 2.33GHz CPU, 16GB RAM and 4 Terabytes disk. The cluster runs Spark v2.0.0 and consists of one driver node and 9 worker nodes. In the experiments, we employ 9 executors, each of which uses 8 cores. The amount of memory for the driver and each executor process is set to 8GB and 12GB, respectively.

7.4.1.2 Datasets

Both synthetic and real-world data are used to evaluate the proposed method. In order to evaluate the proposed method, we generate two synthetic datasets, one for testing the scalability and the other for testing the reconstruction error. For the scalability tests, we generate random tensors of size $I \times J \times K$ by randomly setting a data point at (i, j, k). For the sake of simplification, we assume that their similarity matrices are identity matrices for all modes. For the reconstruction error tests, we first generate randomly factor matrices $\mathbf{A}^{(1)}$, $\mathbf{A}^{(2)}$ and $\mathbf{A}^{(3)}$ with the specific rank $R = 20$ by the following linear formula [32]:

$$\mathbf{A}^{(1)}(i, r) = i\varepsilon_r + \varepsilon', \quad i = 1, 2, \ldots, I_1, r = 1, 2, \cdots, R$$
$$\mathbf{A}^{(2)}(j, r) = j\zeta_r + \zeta', \quad j = 1, 2, \ldots, I_2, r = 1, 2, \cdots, R$$
$$\mathbf{A}^{(3)}(k, r) = k\eta_r + \eta', \quad k = 1, 2, \ldots, I_3, r = 1, 2, \cdots, R$$

where $\{\varepsilon_r, \varepsilon'_r, \zeta_r, \zeta'_r, \eta_r, \eta'_r\}_{r=1,2,\ldots,R}$ are constants generated by the standard Gaussian distribution $N(0, 1)$. Since each factor matrix is generated by linear functions mentioned above column by column, the consecutive rows are similar to each other. Therefore, we

Table 7.1: Summary of the real-world and synthetic datasets used. **K**: thousand, **M**: million, **B**: billion.

Datasets	**I**	**J**	**K**	**Non-zeros**
Netflix	480K	18K	2K	100M
Facebook	60K	60K	5	1.55M
DBLP	317K	317K	629K	1.04M
Twitter	640K	640K	16	1133M
Synthetic-scalability	1K$\sim$1B	1K$\sim$1B	1K$\sim$1B	10K $\sim$10B
Synthetic-error	10K	10K	10K	10M

generate the similar matrix for the ith mode as the following tri-diagonal matrix:

$$\mathbf{S}_i = \begin{bmatrix} 0 & 1 & 0 & \dots \\ 1 & 0 & 1 & \dots \\ 0 & 1 & 0 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{7.14}$$

We then randomly select tensor data points (i, j, k) as our observation and calculate its value via $\mathbf{A}^{(1)}(i, :) \circ \mathbf{A}^{(2)}(j, :) \circ \mathbf{A}^{(3)}(k, :)$. This process is performed until we have the desired number of observed data points. We vary the dimensionality of the synthetic data as well as the rank in order to test the scalability and the reconstruction error, respectively.

For real-world datasets, we use Netflix, Facebook, DBLP, and Twitter summarized in Table 7.1 with the following details:

- **Netflix:** Movie rating data employed in Netflix prize. It forms a *user-movie-time* tensor data by considering the time at which a user rated a movie. The rating is scaled from 1 to 5.

- **Facebook:** Temporal relationships between users from the Facebook New Orleans networks [149]. We convert them into a 3-order tensor in which the third mode

correspond to the date when one anonymized user adds the other user in the first user's friend list.

- **DBLP:** A record of DBLP (a computer science bibliography) publications including authors, papers, conferences, etc. We convert the dataset into a a co-authorship network with *author-author-paper* elements, and conduct a *author-author* similarity based on whether they come from the same affiliation.

- **Twitter:** Geo-tagged Twitter lists data. A Twitter list allows a user (creator) to label another user (expert) with an annotation (e.g., news, food, technology). Since there are a large number of annotations, we transfer them into 16 general topics like news, music, technology, sports, etc. We convert relationships between list creators and experts into a 3-dimensional tensor by adding the topics of lists as the third mode, and produce a *creator-expert* similarity matrix based on their following relationships.

7.4.1.3 Baseline Methods

We compare DISTENC with three tensor completion methods, ALS [34] that is a distributed tensor completion method based upon the alternating least square (ALS) with MPI and OpenMP, and TFAI [32] that is a single-machine tensor completion method with the integration of auxiliary information. We use the original implementation of ALS. Since the other completion method CCD++ based upon the circle coordinate descent [34] has a similar performance with ALS, we only consider ALS as one of our baseline methods here. In addition, we also compare with the state-of-the-art distributed matrix-tensor factorization methods SCOUT [106] and FLEXIFACT [44] implemented on MAPREDUCE. We integrate the similarity matrices of all modes as coupled matrices into SCOUT and FLEXIFACT, respectively.

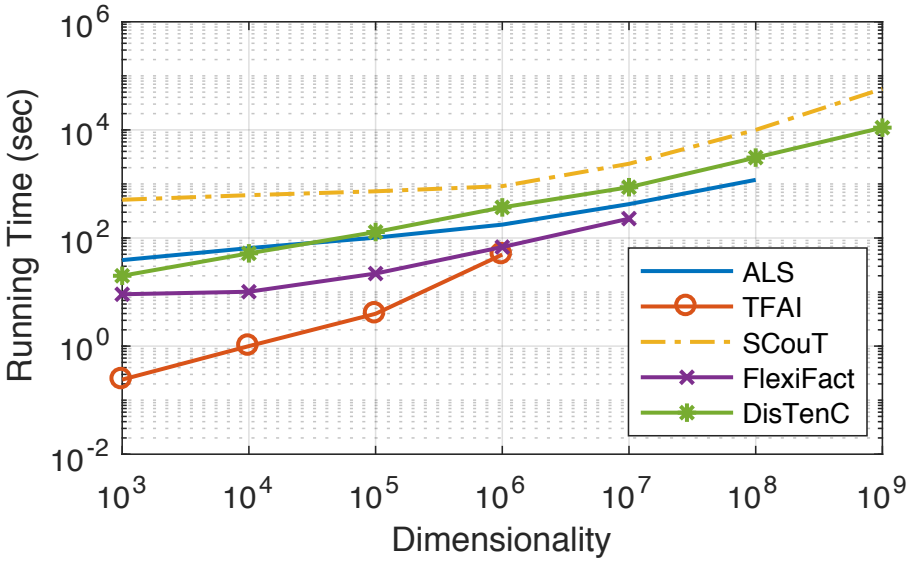


Figure 7.4: Data Scalability: Dimensionality

7.4.2 Data Scalability

We employ synthetic tensors to evaluate the scalability of DISTENC comparing with other baseline methods in terms of three aspects: dimensionality, the number of non-zeros and rank. For the sake of simplification, we set the similarity matrices of all modes to the identity matrices in the scalability tests. All experiments are allowed to run 8 hours. If methods cannot conduct any result within 8 hours, they will be marked as Out-Of-Time.

7.4.2.1 Dimensionality

We increase the tensor size $I = J = K$ from 10^3 to 10^9 while fixing the rank to 20 and the number of non-zero elements to 10^7. As shown in Figure 7.4, DISTENC and SCOUT outperform other baseline methods by successfully performing tensor completion on tensors of size $I = J = K = 10^9$. On the other hand, both ALS and FLEXIFACT run with the out-of-memory (O.O.M.) error when $I = J = K \geq 10^7$; TFAI causes the O.O.M. error when $I = J = K \geq 10^6$. FLEXIFACT does not scale up for very large

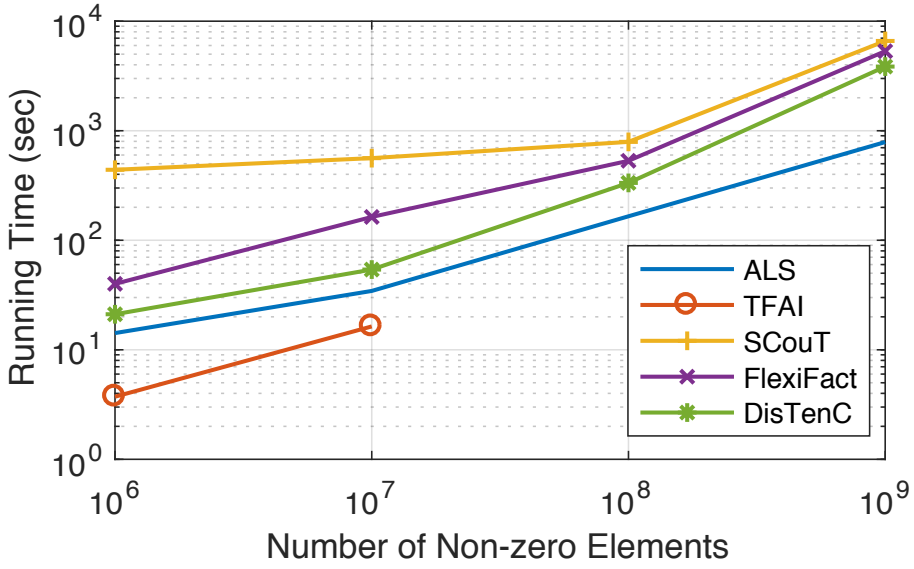


Figure 7.5: Data Scalability: Number of Non-zeros

datasets due to its high communication cost with an exponential increase. ALS requires each communication of entire factor matrices per epoch in the worst case as a coarse-grained decomposition. TFAI is bounded by the memory of a single machine.

7.4.2.2 Number of Non-Zeros

We increase the number of non-zero elements (density) from 10^6 to 10^9 while fixing the dimensionality of the input tensor to $I = J = K = 10^5$ and the rank to 10. As demonstrated in Figure 7.5, only TFAI runs out of memory due to the bound of a single machine while other methods including the proposed DISTENC, ALS, SCOUT and FLEXIFACT are able to scale up to 10^9 non-zero elements. DISTENC takes more running time than ALS with shrinked differences as the number of non-zero elements increases. But DISTENC outperforms both SCOUT and FLEXIFACT due to the advantages of Spark that is more fit for running the iterative algorithms with less disk accesses.

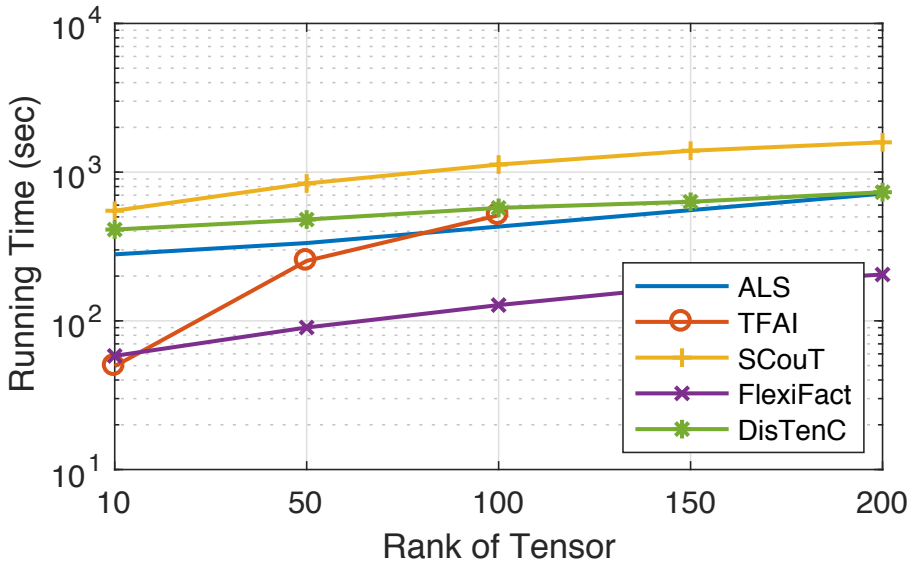


Figure 7.6: Data Scalability: Rank

7.4.2.3 Rank

We increase the rank of a tensor from 10 to 500 while fixing the dimensionality to $I = J = K = 10^6$ and the number of non-zero elements to 10^7. As shown in Figure 7.6, all methods except of TFAI are capable of scaling up to rank 200. The running time of ALS rapidly increases when the rank becomes large due to its cubically increasing computational cost. DISTENC has a relatively flat curve as the increase of rank due to its optimization on calculating the inverse of a symmetric matrix.

7.4.3 Machine Scalability

We measure the machine scalability of the proposed DISTENC by increasing the number of machines from 1 to 8. The synthetic dataset of size $I = J = K = 10^5$ with 10^7 non-zero elements is applied and its rank is set to 10. In Figure 7.7, we report the ratio T_1/T_M where T_M is the running time using M machines. Since TFAI is a single-machine tensor completion method and FLEXIFACT has a worse scalability on machines

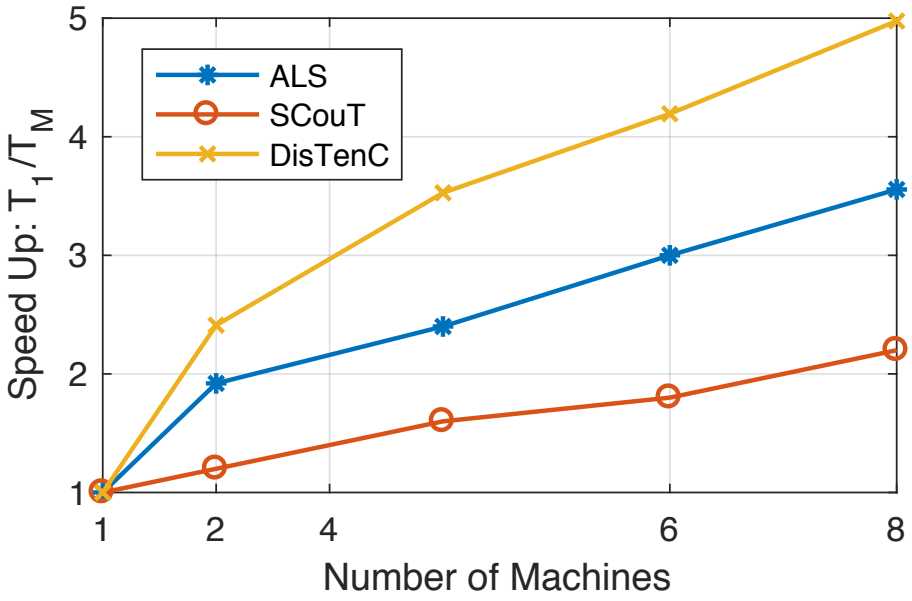


Figure 7.7: Machine scalability of DISTENC compared with ALS and SCOUT. The proposed DISTENC has the best performance in terms of machine scalability with 4.9× speed-up, which also achieves a better linearity on the scalability with respect to the number of machines.

than SCOUT [106], we only compare ALS, SCOUT and the proposed DISTENC. It can be seen that DISTENC obtains 4.9× speed-up as increasing the number of machines from 1 to 8 and achieves a better linearity in terms of machine scalability, which SCOUT slows down due to the intensive hard disk accesses and high communication cost.

7.4.4 Reconstruction Error

In order to evaluate the accuracy of the proposed DISTENC with respect to the reconstruction error, we use the synthetic dataset of size $I = J = K = 10^4$ with 10^7 non-zero elements and set its rank to 10, and adopt adopt *Relative Error* as our evaluation metric. Relative Error is defined as

$$RelativeError = \frac{\|\mathcal{X} - \mathcal{Y}\|_F}{\|\mathcal{Y}\|_F}, \tag{7.15}$$

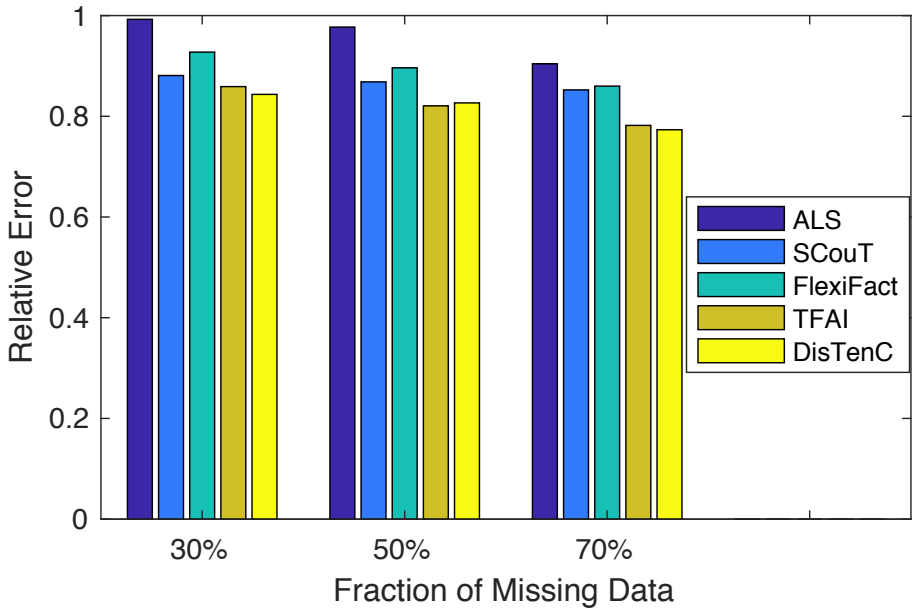


Figure 7.8: Reconstruction error on the synthetic data.

where $\mathcal{X}$ is the recovered tensor and $\mathcal{Y}$ is the ground-truth tensor. We randomly sample the non-zero elements based upon the missing rate as the testing data to calculate the relative error; the rest is used as the training data. We report results in Figure 7.8 by varying the missing rate from 30%, 50% and 70%. All results are averaged by running 5 times in order to reduce the dependency of randomness. Overall, we witness that DISTENC achieves comparable performance with TFAI, but better than ALS, and SCOUT. We see that the integrated auxiliary information (similarity matrix) lead to such significant improvement through the tensor completion. These relationships can alleviate the problem of sparsity to some extent and provide valuable information for the tensor factorization to obtain more interpretable low-rank representations.

7.4.5 Recommender System

In this section, we apply DISTENC to perform recommendation on large scale real-world datasets and present our findings. We are mostly interested in illustrating the power

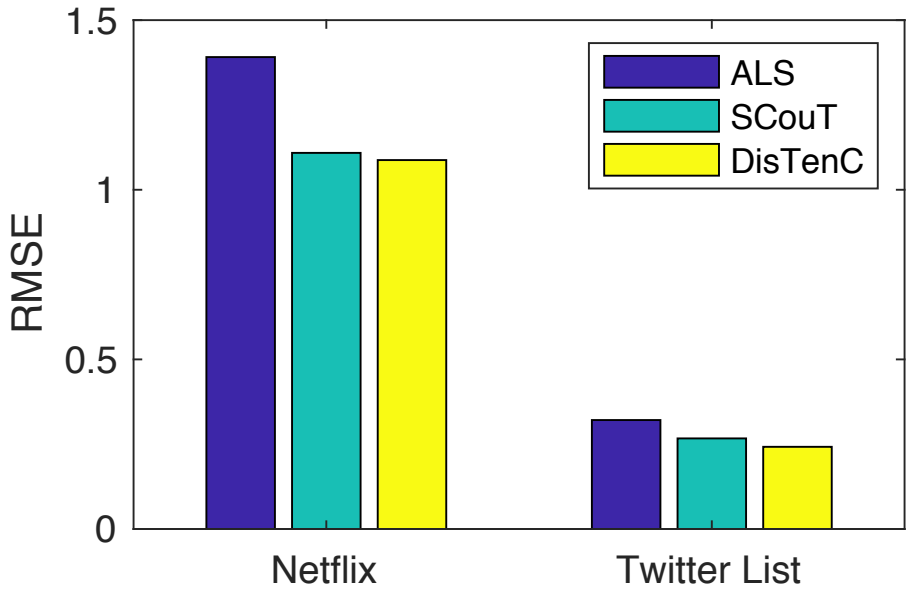


Figure 7.9: Results on recommender system with respect to RMSE.

of our approach rather than systemically comparing with all other state-of-the-art methods. Since SCOUT has a better scalability than FLEXIFACT and TFAI cannot handle such large-scale datasets, we only compare out proposed DISTENC with other two baseline methods ALS and SCOUT. The root-mean-square error (RMSE) is adopted as our evaluation metric, which represents the sample standard deviation of the differences between observed tensor $\mathcal{T}$ and predicted tensor $\mathcal{X}$ as follows.

$$RMSE = \sqrt{\frac{\|\Omega * (\mathcal{T} - \mathcal{X})\|_F^2}{nnz(\mathcal{T})}}. \tag{7.16}$$

It has been commonly used in the evaluation of recommender systems. We randomly use 50% of the observation for training, and the rest for testing. All results are reported by running 5 times and computing the average performance.

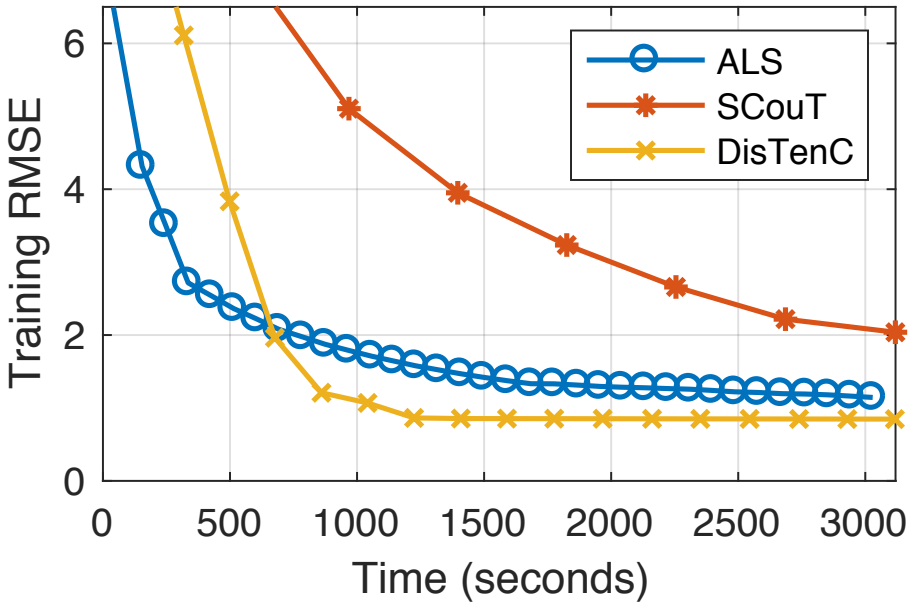


Figure 7.10: Convergence rate for all methods on the Netflix data.

7.4.5.1 Netflix

We conduct the recommendation on Netflix dataset which contains a *user-movie-time* tensor and a movie-movie similarity matrix generated based on the movie title. As shown in Figure 7.9, we observe that the proposed DISTENC obtains the best performance in the precision of recommendation an average improvement of 14.9% over other baseline methods. In addition, by introducing the auxiliary information, both DISTENC and SCOUT outperform ALS. On the other hand, Figure 7.10 shows that the proposed DISTENC converges the fastest to the best solution by taking advantage of ADMM [36, 107]. SCOUT takes much longer time on the convergence by employing the MAPREDUCE framework that requires intensive disk I/Os.

7.4.5.2 Twitter

Using DISTENC, we also perform the expert recommendation on Twitter List dataset which consists of a *creator-expert-topic* tensor as well as *creator-creator* and *expert-expert* similarity matrices calculated based on whether they are located in the same location (cities). As demonstrated in Figure 7.9, DISTENC performs the best among all alternative baseline methods with an average improvement of 21.4% in the precision. Concretely, DISTENC outperforms ALS with an improvement of 32.6% in precision, indicating the the superiority of a tensor completion model integrating auxiliary information. With respect to the convergence, DISTENC has similar performance as one on the Netflix dataset shown in Figure 7.10. Due to the limited space, we omit its details.

7.4.6 Link Prediction

As one of the most applications for tensor completion, link prediction aims to recover unobserved links between nodes in a low-rank tensor (the matrix is a special case). Using DISTENC, we perform link prediction on Facebook dataset that includes a *user-user-time* tensor and a similarity matrix *user-user* generated based on the similarities between their wall posts. As a similar fashion in the previous section, we randomly select 50% of observations for training, and the rest for testing. We also adopt RMSE as the evaluation metric in this experiment. To reduce statistical variability, experimental results are averaged by running 5 times. Figure 7.11 illustrates the testing accuracy and the training convergence. As we can see, both DISTENC and SCOUT have comparable performance and are better than ALS in precision. Specifically, DISTENC outperforms ALS with an average improvement of 27.4%; SCOUT has a better performance than ALS with an average improvement of 19.5%. In terms of convergence, DISTENC converges faster to the best solution.

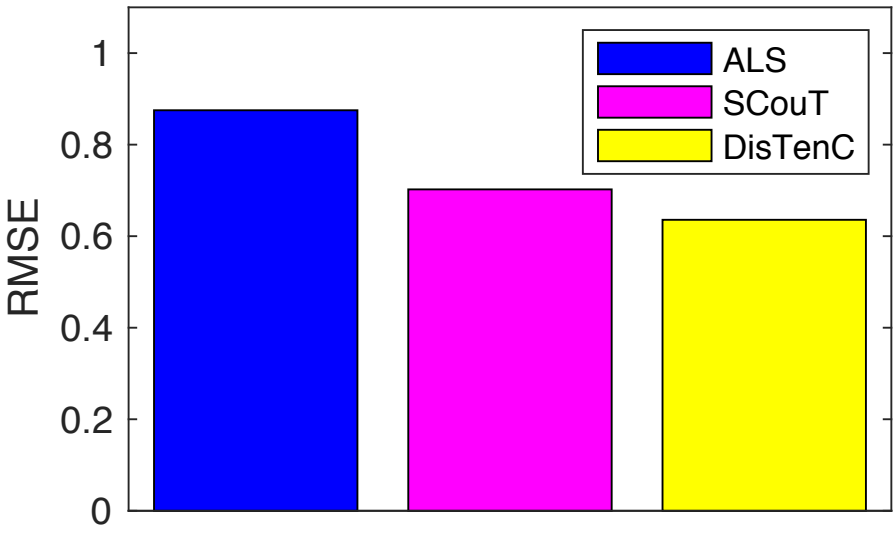


Figure 7.11: Results on link prediction in terms of RMSE.

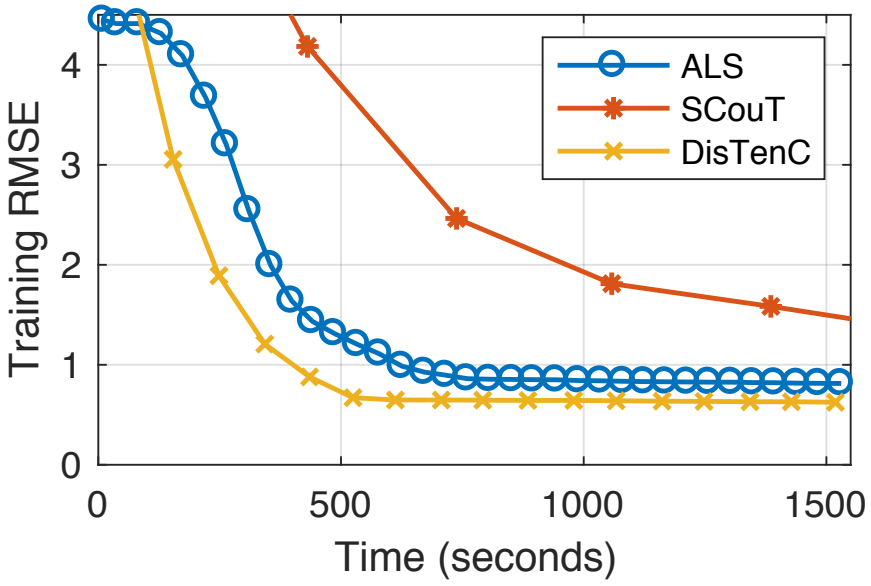


Figure 7.12: Convergence rate for all methods on link prediction.

Table 7.2: Example of concept discovery results on DBLP dataset.

Concept	Authors	Conferences
Datebase	Surajit Chaudhuri, Michael J. Carey, David J. DeWitt, Rajeev Rastogi, Dan Suciu, Ming-Syan Chen,	SIGMOD VLDB ICDE
Data Mining	Jiawei Han, Philip S. Yu, George Karypis, Christos Faloutsos, Shusaku Tsumoto, Rakesh Agrawal.	KDD ICDM PKDD
Info. Retrieval	W. Bruce Croft, Mark Sanderson, Iadh Ounis, ChengXiang Zhai, Gerard Salton, Clement T. Yu.	SIGIR ECIR WWW

7.4.7 Discovery

Since tensor completion perform both imputation and factorization meanwhile, we apply DISTENC on DBLP dataset that contains a *author-paper-venue* tensor with a similarity matrix *author-author*. We randomly select 50% of observations for training the model. After that, we pick top-k highest valued elements from each factor after filtering too general elements. We show 3 notable concepts we found in Table 7.2. It can be seen that all conferences within a concept are correlated and all famous researchers in each concept are discovered.

7.5 Conclusion

In this paper, we propose DISTENC, a distributed algorithm for tensor completion with the integration of auxiliary information based on ADMM, which is capable of scaling up to billion size tensors and achieving good performance across many applications. By efficiently handling trace-based regularization term, updating factor matrices with caching, and optimizing the update of new tensor, DISTENC successfully addresses the high computational cost, and minimizes the generation and shuffling of the intermediate data. Through extensive experiments, DISTENC shows up to 10$\sim$1000$\times$ larger scalabil-

ity than existing methods, and converges much faster than state-of-the-art methods. In addition, DISTENC obtains an average improvement of 18.2% on a recommender system scenario and 23.5% on link prediction.

8. CONCLUSIONS AND FUTURE RESEARCH OPPORTUNITIES

With the increasing ubiquity of large-scale data rapidly growing in both volume and velocity in the real-world, missing information spanning a wide variety of real world applications need to be effectively and efficiently handled by leveraging rich side information. Accomplishing this will require new methods and new systems that are capable of handling real-large datasets as well as their dynamics over time. We have outlined some of the challenges facing this opportunity and highlighted four of our related efforts toward informing this emerging research area.

8.1 Conclusions

Concretely, in this dissertation, we focus on recovering missing information in large-scale multi-aspect datasets by exploiting rich side information on a variety of applications including estimation of missing spatio-temporal dynamics of online memes, personalized expert recommendation and learning user topical profiles. The primary tool used to recover missing information in this dissertation is tensor completion. While tensor completion for large-scale datasets could benefit many domains, there is a significant gap towards handling missing information with rich side information for both effectiveness and efficiency. Therefore, this dissertation made three unique contributions:

First, to tackle the challenges **sparsity** and **similarity**, we present the design of a novel tensor learning framework to recover the spatio-temporal dynamics of memes by exploiting auxiliary information among locations, hashtags, and times. Concretely, we formally define the problem of recovering the spatial-temporal dynamics of online memes and propose a novel tensor-based factorization approach. The core insight of the proposed method is to carefully take into account the latent relationships derived from contextual information among locations, memes, and times; these relationships can then be embedded into

a tensor completion framework for uncovering the incomplete dynamics based on partial observations. We explore how to model and integrate these side information – here, in the form of relationships among locations, memes, and times – and show how the underlying tensor factorization can be efficiently solved with an efficient algorithm based on ADMM. We evaluate the performance of our tensor-based framework over Twitter datasets and through extensive experimental study, we find that our framework achieves a significant improvement in recovery compared to state-of-the-art alternatives, while achieving significantly greater efficiency.

Second, to address the challenge **specificity**, we investigate the proposed tensor learning framework over two key scenarios in social media that are characterized by extreme sparsity and complex relationships: (i) recommending high-quality content producers to users; and (ii) learning user topical profiles. For the first scenario, our goal is to recommend personalized experts to users. Specifically, we extend the proposed tensor learning algorithm to develop a tensor-based completion model TAPER by incorporating both the latent relationships between homogeneous entities (e.g., users and users, experts and experts) and the relationships between heterogeneous entities (e.g., users and experts, topics and experts) from the geo-spatial, topical, and social contexts into the tensor completion framework. Our empirical evaluation over Twitter Lists has demonstrated the superiority of the proposed TAPER to improve the quality of the recommendation in both precision and recall over state-of-the-art baselines. For the second scenario, we tackle the problem of learning user topical problem and also investigate how to leverage user-generated information in heterogeneous and diverse footprints. In particular, we propose a generalized tensor-based contextual regularization model embedded into a matrix factorization framework with integrating pairwise relations among social, textual, and behavioral contexts. We model all contextual signals into one tensor via calculating the user similarity in each of social, textual and behavioral contexts. After applying the tensor decomposition,

the latent representation of users can be learned, which is further embedded into a matrix factorization framework in order to learn unknown user topical profiles. We.evaluate the performance of our proposed model over Twitter datasets and find it to be significantly effective while pertaining high accuracy on learning user topical profiles.

Third, to handle the challenge **scalability**, we enables the proposed tensor learning framework to handle real large-scale datasets that are too big to fit in the main memory of a single machine. Specifically, we propose DISTENC, a novel distributed scalable tensor completion algorithm with incorporating auxiliary information, which is running on the modern parallel computing architecture Spark. By efficiently computing auxiliary variables, minimizing intermediate data, and reducing the workload of updating new tensors. We successfully tackles the high computational costs and minimizes the intermediate data and find that the proposed distributed scalable algorithm is capable of handling up to $10\sim1000\times$ larger tensors than existing methods with much faster convergence rate, shows better linearity on the machine scalability, and achieves up to an average improvement of 23.5% in the applications such as recommender systems and link prediction.

8.2 Future Research Opportunities

Though we have seen the success that tensor completion has experienced in data mining in recent years, there are still many challenges and open problems that have not been appropriately addressed. With respect to future work, we are quite interesting in the following directions:

- **Tackling Rapidly Evolving Multi-aspect Datasets**. In this dissertation, we propose scalable tensor learning algorithms to recover the missing information across a variety of applications in which the datasets are mostly static. With the rapid growth of datasets in both volume and velocity, real-world applications are often imbued with high velocity streaming data beyond traditional static setting. For in-

stance, Facebook users update almost 700,000 messages per minute; Twitter users generate over 100,000 tweets every single minute. Motivated by such fast-evloving data, dynamic tensor completion algorithms need to be developed to fill missing information timely and incorporate newly emerging patterns. Recently, Qingquan et al. [150] propose a Multi-Aspect Streaming Tensor completion framework (MAST) to track the subspace of general incremental tensors for completion by effectively capturing its low-rank structure. A nature follow up for both ours and Qingquan's work is to scale up dynamic tensor completion algorithms to be capable of handling real large-scale datasets that usually have billions of entries. Interesting questions to study include: How can we develop a scalable dynamic tensor completion algorithm that can be deployed in the modern distributed computing architecture like Hadoop and Spark? Can we incorporate side information into the dynamic tensor completion algorithms to improve the complete model reconstruction in a distributed fashion?

- **Exploiting Heterogeneous Information Networks**. As we can see, there are considerable existing work on Heterogeneous Information Networks (HINs) that are graphs between different types of nodes with different types of edges. In this dissertation, we integrate rich side information into tensor completion algorithms to improve performance of recovering missing information based upon the assumption that there exist some statistical regularities under the intertwined facts stored in the multi-aspect datasets. From a different point of view, HINs can be represented as tensors (e.g., multi-aspect social network). Similar nodes or communities can be explored by techniques like Meta-Path [151] that traverses multiple types of nodes with a similar fashion of random walk. We are interested in connecting between HINs and tensor completion towards unifying different data mining approaches as well as better exploiting rich side information.

- **Connecting with Neural Network**. Deep learning is becoming popular due to the expressive power of neural networks through depth, which has been successfully applied in many fields such as computer vision, data mining, nature language processing, bioinformatics, etc. In this dissertation, we mainly focus on tensor completion methods through factorizing the tensor data and recovering missing information based upon factor matrices. Recently, Socher et al. [152] propose Neural Tensor Networks (NTNs) for knowledge base completion, which allows mediated interaction of entity vectors via a tensor. We are interested in connecting between deep learning and tensor completion with adaptability of scaling slices and applying to applications such as recommendation, attribute embedding, and link prediction.

REFERENCES

[1] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr, and T. M. Mitchell, "Toward an architecture for never-ending language learning," in *AAAI*, vol. 5, p. 3, 2010.

[2] C. J. Appellof and E. R. Davidson, "Strategies for analyzing data from video fluorometric monitoring of liquid chromatographic effluents," *Analytical Chemistry*, vol. 53, no. 13, pp. 2053–2056, 1981.

[3] A. Cichocki, D. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and H. A. Phan, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 145–163, 2015.

[4] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3551–3582, 2017.

[5] M. A. O. Vasilescu and D. Terzopoulos, "Multilinear analysis of image ensembles: Tensorfaces," in *European Conference on Computer Vision*, pp. 447–460, Springer, 2002.

[6] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 208–220, 2013.

[7] K. Hasselmann, "Multi-pattern fingerprint method for detection and attribution of climate change," *Climate Dynamics*, vol. 13, no. 9, pp. 601–611, 1997.

[8] R. Yu and Y. Liu, "Learning from multiway data: Simple and efficient tensor regression," in *International Conference on Machine Learning*, pp. 373–381, 2016.

[9] E. Acar, C. Aykut-Bingol, H. Bingol, R. Bro, and B. Yener, "Multiway analysis of epilepsy tensors," *Bioinformatics*, vol. 23, no. 13, pp. i10–i18, 2007.

[10] E. E. Papalexakis, A. Fyshe, N. D. Sidiropoulos, P. P. Talukdar, T. M. Mitchell, and C. Faloutsos, "Good-enough brain model: Challenges, algorithms, and discoveries in multisubject experiments," *Big Data*, vol. 2, no. 4, pp. 216–229, 2014.

[11] K. Maruhashi, F. Guo, and C. Faloutsos, "Multiaspectforensics: Pattern mining on large-scale heterogeneous networks with tensor analysis," in *IEEE International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 203–210, IEEE, 2011.

[12] T. Kolda and B. Bader, "The tophits model for higher-order web link analysis," in *Workshop on Link Analysis, Counterterrorism and Security*, vol. 7, pp. 26–29, 2006.

[13] H. Fanaee-T and J. Gama, "Tensor-based anomaly detection: An interdisciplinary survey," *Knowledge-Based Systems*, vol. 98, pp. 130–147, 2016.

[14] T. G. Kolda and J. Sun, "Scalable tensor decompositions for multi-aspect data mining," in *Proceedings of the 8th IEEE International Conference on Data Mining*, pp. 363–372, IEEE, 2008.

[15] S. Rendle and L. Schmidt-Thieme, "Pairwise interaction tensor factorization for personalized tag recommendation," in *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*, pp. 81–90, ACM, 2010.

[16] H. Ge, J. Caverlee, and H. Lu, "Taper: A contextual tensor-based approach for personalized expert recommendation.," in *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 261–268, 2016.

[17] J. Sun, D. Tao, S. Papadimitriou, P. S. Yu, and C. Faloutsos, "Incremental tensor analysis: Theory and applications," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 2, no. 3, p. 11, 2008.

[18] G. Kossinets, "Effects of missing data in social networks," *Social Networks*, vol. 28, no. 3, pp. 247–268, 2006.

[19] E. Sadikov, M. Medina, J. Leskovec, and H. Garcia-Molina, "Correcting for missing data in information cascades," in *Proceedings of the 4th ACM international Conference on Web Search and Data Mining*, pp. 55–64, ACM, 2011.

[20] M. Scheuerer, R. Schaback, and M. Schlather, "Interpolation of spatial data–a stochastic or a deterministic problem?," *European Journal of Applied Mathematics*, vol. 24, no. 4, pp. 601–629, 2013.

[21] D. Kondrashov and M. Ghil, "Spatio-temporal filling of missing points in geophysical data sets," *Nonlinear Processes in Geophysics*, vol. 13, no. 2, pp. 151–159, 2006.

[22] Y. Li and L. E. Parker, "A spatial-temporal imputation technique for classification with missing data in a wireless sensor network," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3272–3279, IEEE, 2008.

[23] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[24] Y. Zhang, M. Roughan, W. Willinger, and L. Qiu, "Spatio-temporal compressive sensing and internet traffic matrices," in *ACM SIGCOMM Computer Communication Review*, vol. 39, pp. 267–278, ACM, 2009.

[25] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational Mathematics*, vol. 9, no. 6, p. 717, 2009.

[26] Y. Liu, F. Shang, L. Jiao, J. Cheng, and H. Cheng, "Trace norm regularized candecomp/parafac decomposition with missing data," *IEEE Transactions on Cybernetics*, vol. 45, no. 11, pp. 2437–2448, 2015.

[27] D. M. Dunlavy, T. G. Kolda, and E. Acar, "Temporal link prediction using matrix and tensor factorizations," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 5, no. 2, p. 10, 2011.

[28] Y. Wang, R. Chen, J. Ghosh, J. C. Denny, A. Kho, Y. Chen, B. A. Malin, and J. Sun, "Rubik: Knowledge guided tensor factorization and completion for health data analytics," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1265–1274, ACM, 2015.

[29] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, A. Hanjalic, and N. Oliver, "Tfmap: optimizing map for top-n context-aware recommendation," in *Proceedings of the 35th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 155–164, ACM, 2012.

[30] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Mørup, "Scalable tensor factorizations with missing data," in *Proceedings of the 2010 SIAM International Conference on Data Mining*, pp. 701–712, SIAM, 2010.

[31] R. Forsati, M. Mahdavi, M. Shamsfard, and M. Sarwat, "Matrix factorization with explicit trust and distrust side information for improved social recommendation," *ACM Transactions on Information Systems (TOIS)*, vol. 32, no. 4, p. 17, 2014.

[32] A. Narita, K. Hayashi, R. Tomioka, and H. Kashima, "Tensor factorization using auxiliary information," *Data Mining and Knowledge Discovery*, vol. 25, no. 2,

pp. 298–324, 2012.

[33] L. Karlsson, D. Kressner, and A. Uschmajew, "Parallel algorithms for tensor completion in the cp format," *Parallel Computing*, vol. 57, pp. 222–234, 2016.

[34] S. Smith, J. Park, and G. Karypis, "An exploration of optimization algorithms for high performance tensor completion," in *International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 359–371, IEEE, 2016.

[35] S. Greenland, J. A. Schwartzbaum, and W. D. Finkle, "Problems due to small samples and sparse data in conditional logistic regression analysis," *American Journal of Epidemiology*, vol. 151, no. 5, pp. 531–539, 2000.

[36] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[37] F. L. Hitchcock, "The expression of a tensor or a polyadic as a sum of products," *Studies in Applied Mathematics*, vol. 6, no. 1-4, pp. 164–189, 1927.

[38] J. D. Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of âĂIJeckart-youngâĂİ decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[39] R. A. Harshman, "Foundations of the parafac procedure: models and conditions for an" explanatory" multimodal factor analysis," 1970.

[40] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[41] J. H. Choi and S. Vishwanathan, "Dfacto: Distributed factorization of tensors," in *Advances in Neural Information Processing Systems*, pp. 1296–1304, 2014.

[42] B. W. Bader and T. G. Kolda, "Efficient matlab computations with sparse and factored tensors," *SIAM Journal on Scientific Computing*, vol. 30, no. 1, pp. 205–231, 2007.

[43] G. Tomasi and R. Bro, "Parafac and missing values," *Chemometrics and Intelligent Laboratory Systems*, vol. 75, no. 2, pp. 163–180, 2005.

[44] A. Beutel, P. P. Talukdar, A. Kumar, C. Faloutsos, E. E. Papalexakis, and E. P. Xing, "Flexifact: Scalable flexible factorization of coupled tensors on hadoop," in *Proceedings of the 2014 SIAM International Conference on Data Mining*, pp. 109–117, SIAM, 2014.

[45] E. E. Papalexakis, C. Faloutsos, T. M. Mitchell, P. P. Talukdar, N. D. Sidiropoulos, and B. Murphy, "Turbo-smt: Accelerating coupled sparse matrix-tensor factorizations by 200x," in *Proceedings of the 2014 SIAM International Conference on Data Mining*, pp. 118–126, SIAM, 2014.

[46] K. Shin and U. Kang, "Distributed methods for high-dimensional and large-scale tensor factorization," in *Proceedings of the IEEE 30th International Conference on Data Mining*, pp. 989–994, IEEE, 2014.

[47] X. Hu, J. Tang, H. Gao, and H. Liu, "Social spammer detection with sentiment information," in *Proceedings of the 30th IEEE International Conference on Data Mining*, pp. 180–189, IEEE, 2014.

[48] N. Dalvi, R. Kumar, and B. Pang, "Object matching in tweets with spatial models," in *Proceedings of the 5th ACM International Conference on Web Search and Data Mining*, pp. 43–52, ACM, 2012.

[49] M. Clements, P. Serdyukov, A. P. De Vries, and M. J. Reinders, "Using flickr geotags to predict user travel behaviour," in *Proceedings of the 33rd international ACM*

SIGIR Conference on Research and Development in Information Retrieval, pp. 851–852, ACM, 2010.

[50] L. Backstrom, E. Sun, and C. Marlow, "Find me if you can: improving geographical prediction with social and spatial proximity," in *Proceedings of the 19th International Conference on World Wide Web*, pp. 61–70, ACM, 2010.

[51] A. Brodersen, S. Scellato, and M. Wattenhofer, "Youtube around the world: geographic popularity of videos," in *Proceedings of the 21st International Conference on World Wide Web*, pp. 241–250, ACM, 2012.

[52] J. Yang and J. Leskovec, "Patterns of temporal variation in online media," in *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*, pp. 177–186, ACM, 2011.

[53] Y. Matsubara, Y. Sakurai, B. A. Prakash, L. Li, and C. Faloutsos, "Rise and fall patterns of information diffusion: model and implications," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 6–14, ACM, 2012.

[54] K. Y. Kamath, J. Caverlee, K. Lee, and Z. Cheng, "Spatio-temporal dynamics of online memes: a study of geo-tagged tweets," in *Proceedings of the 22nd International Conference on World Wide Web*, pp. 667–678, ACM, 2013.

[55] Y. Hu, D. Zhang, J. Ye, X. Li, and X. He, "Fast and accurate matrix completion via truncated nuclear norm regularization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 9, pp. 2117–2130, 2013.

[56] M. T. Bahadori, Q. R. Yu, and Y. Liu, "Fast multivariate spatio-temporal analysis via low rank tensor learning," in *Advances in Neural Information Processing Systems*, pp. 3491–3499, 2014.

[57] H. Zhou, D. Zhang, K. Xie, and Y. Chen, "Spatio-temporal tensor completion for imputing missing internet traffic data," in *IEEE 34th International Performance Computing and Communications Conference*, pp. 1–7, IEEE, 2015.

[58] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Computers & Mathematics with Applications*, vol. 2, no. 1, pp. 17–40, 1976.

[59] J. Weng, E.-P. Lim, J. Jiang, and Q. He, "Twitterrank: finding topic-sensitive influential twitterers," in *Proceedings of the third ACM International Conference on Web Search and Data Mining*, pp. 261–270, ACM, 2010.

[60] S. Ghosh, N. Sharma, F. Benevenuto, N. Ganguly, and K. Gummadi, "Cognos: crowdsourcing search for topic experts in microblogs," in *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 575–590, ACM, 2012.

[61] J. Zhang, M. S. Ackerman, and L. Adamic, "Expertise networks in online communities: structure and algorithms," in *Proceedings of the 16th International Conference on World Wide Web*, pp. 221–230, ACM, 2007.

[62] D. Hu and J. L. Zhao, "Expert recommendation via semantic social networks," *Proceedings of the 2008 International Conference on Information Systems*, p. 196, 2008.

[63] I. Guy, U. Avraham, D. Carmel, S. Ur, M. Jacovi, and I. Ronen, "Mining expertise and interests from social media," in *Proceedings of the 22nd International Conference on World Wide Web*, pp. 515–526, ACM, 2013.

[64] D. W. McDonald and M. S. Ackerman, "Expertise recommender: a flexible recommendation system and architecture," in *Proceedings of the 3rd ACM conference on*

Computer Supported Cooperative Work, pp. 231–240, ACM, 2000.

[65] P. Serdyukov, H. Rode, and D. Hiemstra, "Modeling multi-step relevance propagation for expert finding," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pp. 1133–1142, ACM, 2008.

[66] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, 2009.

[67] H. Lu and J. Caverlee, "Exploiting geo-spatial preference for personalized expert recommendation," in *Proceedings of the 9th ACM Conference on Recommender Systems*, pp. 67–74, ACM, 2015.

[68] B. Yang and S. Manandhar, "Tag-based expert recommendation in community question answering," in *Proceedings of the 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 960–963, IEEE, 2014.

[69] H.-F. Yu, C.-J. Hsieh, S. Si, and I. Dhillon, "Scalable coordinate descent approaches to parallel matrix factorization for recommender systems," in *Proceedings of the 12th IEEE International Conference on Data Mining*, pp. 765–774, IEEE, 2012.

[70] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering," in *Proceedings of the 4th ACM Conference on Recommender Systems*, pp. 79–86, ACM, 2010.

[71] B. Hidasi and D. Tikk, "Fast als-based tensor factorization for context-aware recommendation from implicit feedback," *Machine Learning and Knowledge Discovery in Databases*, pp. 67–82, 2012.

[72] 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 *Proceedings of the 24th International Conference on World Wide Web*, pp. 130–140, International World Wide Web Conferences Steering Committee, 2015.

[73] I. Guy, N. Zwerdling, I. Ronen, D. Carmel, and E. Uziel, "Social media recommendation based on people and tags," in *Proceedings of the 33rd international ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 194–201, ACM, 2010.

[74] J. Hannon, M. Bennett, and B. Smyth, "Recommending twitter users to follow using content and collaborative filtering approaches," in *Proceedings of the 4th ACM Conference on Recommender Systems*, pp. 199–206, ACM, 2010.

[75] M. Jiang, P. Cui, R. Liu, Q. Yang, F. Wang, W. Zhu, and S. Yang, "Social contextual recommendation," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, pp. 45–54, ACM, 2012.

[76] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*, pp. 287–296, ACM, 2011.

[77] A. Majumder and N. Shrivastava, "Know your personalization: learning topic level personalization in online services," in *Proceedings of the 22nd International Conference on World Wide Web*, pp. 873–884, ACM, 2013.

[78] A. Sieg, B. Mobasher, and R. Burke, "Web search personalization with ontological user profiles," in *Proceedings of the 16th ACM conference on Conference on Information and Knowledge Management*, pp. 525–534, ACM, 2007.

[79] Z. Zhao, Z. Cheng, L. Hong, and E. H. Chi, "Improving user topic interest profiles by behavior factorization," in *Proceedings of the 24th International Conference on*

World Wide Web, pp. 1406–1416, International World Wide Web Conferences Steering Committee, 2015.

[80] E. Zhong, N. Liu, Y. Shi, and S. Rajan, "Building discriminative user profiles for large-scale content recommendation," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2277–2286, ACM, 2015.

[81] M. Jamali and L. Lakshmanan, "Heteromf: recommendation in heterogeneous information networks using context dependent factor models," in *Proceedings of the 22nd International Conference on World Wide Web*, pp. 643–654, ACM, 2013.

[82] L. Hong, A. S. Doumith, and B. D. Davison, "Co-factorization machines: modeling user interests and predicting individual decisions in twitter," in *Proceedings of the 6th ACM International Conference on Web Search and Data Mining*, pp. 557–566, ACM, 2013.

[83] R. Ottoni, D. B. Las Casas, J. P. Pesce, W. Meira Jr, C. Wilson, A. Mislove, and V. A. Almeida, "Of pins and tweets: Investigating how users behave across image-and text-based social networks.," in *The 8th International AAAI Conference on Weblogs and Social Media*, 2014.

[84] M. Qiu, F. Zhu, and J. Jiang, "It is not just what we say, but how we say them: Lda-based behavior-topic model," in *Proceedings of the 2013 SIAM International Conference on Data Mining*, pp. 794–802, SIAM, 2013.

[85] X. Zhang, J. Cheng, T. Yuan, B. Niu, and H. Lu, "Toprec: domain-specific recommendation through community topic mining in social network," in *Proceedings of the 22nd International Conference on World Wide Web*, pp. 1501–1510, ACM, 2013.

[86] H. Ma, "On measuring social friend interest similarities in recommender systems," in *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 465–474, ACM, 2014.

[87] A. Mislove, B. Viswanath, K. P. Gummadi, and P. Druschel, "You are who you know: inferring user profiles in online social networks," in *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*, pp. 251–260, ACM, 2010.

[88] T. Lappas, K. Punera, and T. Sarlos, "Mining tags using social endorsement networks," in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 195–204, ACM, 2011.

[89] E. Bakshy, I. Rosenn, C. Marlow, and L. Adamic, "The role of social networks in information diffusion," in *Proceedings of the 21st International Conference on World Wide Web*, pp. 519–528, ACM, 2012.

[90] B. Suh, L. Hong, P. Pirolli, and E. H. Chi, "Want to be retweeted? large scale analytics on factors impacting retweet in twitter network," in *Proceedings of the IEEE Second International Conference on Social Computing*, pp. 177–184, IEEE, 2010.

[91] H. Gao, J. Hu, C. Wilson, Z. Li, Y. Chen, and B. Y. Zhao, "Detecting and characterizing social spam campaigns," in *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, pp. 35–47, ACM, 2010.

[92] X. Liu and K. Aberer, "Soco: a social network aided context-aware recommender system," in *Proceedings of the 22nd International Conference on World Wide Web*, pp. 781–802, ACM, 2013.

[93] Y. Lu, P. Tsaparas, A. Ntoulas, and L. Polanyi, "Exploiting social context for review quality prediction," in *Proceedings of the 19th International Conference on World wide web*, pp. 691–700, ACM, 2010.

[94] A. Anandkumar, R. Ge, D. J. Hsu, S. M. Kakade, and M. Telgarsky, "Tensor decompositions for learning latent variable models.," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 2773–2832, 2014.

[95] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000.

[96] M. Jiang, P. Cui, F. Wang, X. Xu, W. Zhu, and S. Yang, "Fema: flexible evolutionary multi-faceted analysis for dynamic behavioral pattern discovery," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1186–1195, ACM, 2014.

[97] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: concepts, methodologies, and applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 5, no. 3, p. 38, 2014.

[98] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: A heterogeneous information network approach," in *Proceedings of the 7th ACM international Conference on Web Search and Data Mining*, pp. 283–292, ACM, 2014.

[99] A. P. Singh and G. J. Gordon, "Relational learning via collective matrix factorization," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 650–658, ACM, 2008.

[100] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and C. Zhu, "Personalized recommendation via cross-domain triadic factorization," in *Proceedings of the 22nd International Conference on World Wide Web*, pp. 595–606, ACM, 2013.

[101] M. Kim and K. S. Candan, "Decomposition-by-normalization (dbn): leveraging approximate functional dependencies for efficient cp and tucker decompositions," *Data Mining and Knowledge Discovery*, vol. 30, no. 1, pp. 1–46, 2016.

[102] U. Kang, E. Papalexakis, A. Harpale, and C. Faloutsos, "Gigatensor: scaling tensor analysis up by 100 times-algorithms and discoveries," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 316–324, ACM, 2012.

[103] I. Jeon, E. E. Papalexakis, U. Kang, and C. Faloutsos, "Haten2: Billion-scale tensor decompositions," in *Proceedings of the IEEE 31st International Conference on Data Engineering*, pp. 1047–1058, IEEE, 2015.

[104] N. Park, S. Oh, and U. Kang, "Fast and scalable distributed boolean tensor factorization," in *Proceedings of the IEEE 33rd International Conference on Data Engineering*, pp. 1071–1082, IEEE, 2017.

[105] O. Kaya and B. Uçar, "Scalable sparse tensor decompositions in distributed memory systems," in *International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–11, IEEE, 2015.

[106] B. Jeon, I. Jeon, L. Sael, and U. Kang, "Scout: Scalable coupled matrix-tensor factorization-algorithm and discoveries," in *Proceedings of the IEEE 32nd International Conference on Data Engineering*, pp. 811–822, IEEE, 2016.

[107] A. P. Liavas and N. D. Sidiropoulos, "Parallel algorithms for constrained tensor factorization via alternating direction method of multipliers," *IEEE Transactions*

on Signal Processing, vol. 63, no. 20, pp. 5450–5463, 2015.

[108] S. Smith, N. Ravindran, N. D. Sidiropoulos, and G. Karypis, "Splatt: Efficient and parallel sparse tensor-matrix multiplication," in *IEEE International Parallel and Distributed Processing Symposium*, pp. 61–70, IEEE, 2015.

[109] V. Kalavri and V. Vlassov, "Mapreduce: Limitations, optimizations and open issues," in *Proceedings of the 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 1031–1038, IEEE, 2013.

[110] S. Wadkar, M. Siddalingaiah, and J. Venner, *Pro Apache Hadoop*. Apress, 2014.

[111] Y. Low, D. Bickson, J. Gonzalez, C. Guestrin, A. Kyrola, and J. M. Hellerstein, "Distributed graphlab: a framework for machine learning and data mining in the cloud," *Proceedings of the VLDB Endowment*, vol. 5, no. 8, pp. 716–727, 2012.

[112] S. Papadimitriou and J. Sun, "Disco: Distributed co-clustering with map-reduce: A case study towards petabyte-scale end-to-end mining," in *Proceedings of the 8th IEEE International Conference on Data Mining*, pp. 512–521, IEEE, 2008.

[113] G. D. F. Morales and A. Bifet, "Samoa: scalable advanced massive online analysis.," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 149–153, 2015.

[114] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, pp. 2–2, USENIX Association, 2012.

[115] R. Bosagh Zadeh, X. Meng, A. Ulanov, B. Yavuz, L. Pu, S. Venkataraman, E. Sparks, A. Staple, and M. Zaharia, "Matrix computations and optimization in

apache spark," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 31–38, ACM, 2016.

[116] M. S. Wiewiórka, A. Messina, A. Pacholewska, S. Maffioletti, P. Gawrysiak, and M. J. Okoniewski, "Sparkseq: fast, scalable and cloud-ready tool for the interactive genomic data analysis with nucleotide precision," *Bioinformatics*, vol. 30, no. 18, pp. 2652–2653, 2014.

[117] Z. Lu, D. Agarwal, and I. S. Dhillon, "A spatio-temporal approach to collaborative filtering," in *Proceedings of the 3rd ACM Conference on Recommender Systems*, pp. 13–20, ACM, 2009.

[118] T. Sakaki, M. Okazaki, and Y. Matsuo, "Tweet analysis for real-time event detection and earthquake reporting system development," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 4, pp. 919–931, 2013.

[119] F. Morstatter, J. Pfeffer, H. Liu, and K. M. Carley, "Is the sample good enough? comparing data from twitter's streaming api with twitter's firehose.," in *The 7th International AAAI Conference on Weblogs and Social Media*, 2013.

[120] R. Tomioka, K. Hayashi, and H. Kashima, "Estimation of low-rank tensors via convex optimization," *arXiv preprint arXiv:1010.0789*, 2010.

[121] C. E. Rasmussen, "Gaussian processes for machine learning," 2006.

[122] W. R. Tobler, "A computer movie simulating urban growth in the detroit region," *Economic Geography*, 1970.

[123] H. Lamba, V. Nagarajan, K. Shin, and N. Shajarisales, "Incorporating side information in tensor completion," in *Proceedings of the 25th International Conference Companion on World Wide Web*, pp. 65–66, International World Wide Web Conferences Steering Committee, 2016.

[124] M. Filipović and A. Jukić, "Tucker factorization with missing data with application to low-n-rank tensor completion," *Multidimensional Systems and Signal Processing*, vol. 26, no. 3, pp. 677–692, 2015.

[125] A. Pal, A. Herdagdelen, S. Chatterji, S. Taank, and D. Chakrabarti, "Discovery of topical authorities in instagram," in *Proceedings of the 25th International Conference on World Wide Web*, pp. 1203–1213, International World Wide Web Conferences Steering Committee, 2016.

[126] S. A. Paul, L. Hong, and E. H. Chi, "Who is authoritative? understanding reputation mechanisms in quora," *arXiv preprint arXiv:1204.3724*, 2012.

[127] X. Amatriain, N. Lathia, J. M. Pujol, H. Kwak, and N. Oliver, "The wisdom of the few: a collaborative filtering approach based on expert opinions from the web," in *Proceedings of the 32nd international ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 532–539, ACM, 2009.

[128] K. Balog, L. Azzopardi, and M. De Rijke, "Formal models for expert finding in enterprise corpora," in *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 43–50, ACM, 2006.

[129] Z. Cheng, J. Caverlee, H. Barthwal, and V. Bachani, "Who is the barbecue king of texas?: a geo-spatial approach to finding local experts on twitter," in *Proceedings of the 37th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 335–344, ACM, 2014.

[130] H. Ma, H. Yang, M. R. Lyu, and I. King, "Sorec: social recommendation using probabilistic matrix factorization," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pp. 931–940, ACM, 2008.

[131] P. Bhattacharya, S. Ghosh, J. Kulshrestha, M. Mondal, M. B. Zafar, N. Ganguly, and K. P. Gummadi, "Deep twitter diving: Exploring topical groups in microblogs at scale," in *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work and Social Computing*, pp. 197–210, ACM, 2014.

[132] V. Rakesh, D. Singh, B. Vinzamuri, and C. K. Reddy, "Personalized recommendation of twitter lists using content and network information.," in *The 8th International AAAI Conference on Weblogs and Social Media*, 2014.

[133] M. Naaman, Y. J. Song, A. Paepcke, and H. Garcia-Molina, "Automatic organization for digital photographs with geographic coordinates," in *Proceedings of the 2004 Joint ACM/IEEE Conference on Digital Libraries*, pp. 53–62, IEEE, 2004.

[134] D. Zhou, B. Wang, S. M. Rahimi, and X. Wang, "A study of recommending locations on location-based social network by collaborative filtering.," in *Canadian Conference on AI*, pp. 255–266, Springer, 2012.

[135] M. Ye, X. Liu, and W.-C. Lee, "Exploring social influence for recommendation: a generative model approach," in *Proceedings of the 35th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 671–680, ACM, 2012.

[136] Q. Liu, E. Chen, H. Xiong, C. H. Ding, and J. Chen, "Enhancing collaborative filtering by user interest expansion via personalized ranking," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 1, pp. 218–233, 2012.

[137] H. Yin, B. Cui, L. Chen, Z. Hu, and Z. Huang, "A temporal context-aware model for user behavior modeling in social media systems," in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, pp. 1543–1554, ACM, 2014.

[138] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual Review of Sociology*, vol. 27, no. 1, pp. 415–444, 2001.

[139] J. Bollen, B. Gonçalves, G. Ruan, and H. Mao, "Happiness is assortative in online social networks," *Artificial Life*, vol. 17, no. 3, pp. 237–251, 2011.

[140] D. Centola, "An experimental study of homophily in the adoption of health behavior," *Science*, vol. 334, no. 6060, pp. 1269–1272, 2011.

[141] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?," in *Proceedings of the 19th International Conference on World wide web*, pp. 591–600, ACM, 2010.

[142] J. Tang, Y. Chang, and H. Liu, "Mining social media with social theories: a survey," *ACM SIGKDD Explorations Newsletter*, vol. 15, no. 2, pp. 20–29, 2014.

[143] R. Xiang, J. Neville, and M. Rogati, "Modeling relationship strength in online social networks," in *Proceedings of the 19th International Conference on World wide web*, pp. 981–990, ACM, 2010.

[144] A. A. Goldstein, "Constructive real analysis," tech. rep., WASHINGTON UNIV SEATTLE DEPT OF MATHEMATICS, 1967.

[145] C. Cao, J. Caverlee, K. Lee, H. Ge, and J. Chung, "Organic or organized?: Exploring url sharing behavior," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 513–522, ACM, 2015.

[146] H. Ge, J. Caverlee, N. Zhang, and A. Squicciarini, "Uncovering the spatio-temporal dynamics of memes in the presence of incomplete information," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pp. 1493–1502, ACM, 2016.

[147] P. Bientinesi, I. S. Dhillon, and R. A. Van De Geijn, "A parallel eigensolver for dense symmetric matrices based on multiple relatively robust representations," *SIAM Journal on Scientific Computing*, vol. 27, no. 1, pp. 43–66, 2005.

[148] S. Smith and G. Karypis, "A medium-grained algorithm for distributed sparse tensor factorization," in *Proceedings of the 30th IEEE International Parallel & Distributed Processing Symposium*, 2016.

[149] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in facebook," in *Proceedings of the 2nd ACM Workshop on Online Social Networks*, pp. 37–42, ACM, 2009.

[150] Q. Song, X. Huang, H. Ge, J. Caverlee, and X. Hu, "Multi-aspect streaming tensor completion," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 435–443, ACM, 2017.

[151] C. Meng, R. Cheng, S. Maniu, P. Senellart, and W. Zhang, "Discovering meta-paths in large heterogeneous information networks," in *Proceedings of the 24th International Conference on World Wide Web*, pp. 754–764, International World Wide Web Conferences Steering Committee, 2015.

[152] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Advances in Neural Information Processing Systems*, pp. 926–934, 2013.