

MetaFac: Community Discovery via Relational Hypergraph Factorization

Yu-Ru Lin¹ Jimeng Sun² Paul Castro² Ravi Konuru² Hari Sundaram¹ Aisling Kelliher¹

¹Arts Media and Engineering
Program, Arizona State University,
Tempe, AZ 85281 USA

²IBM T.J. Watson Research Center,
Hawthorne, NY 10532 USA

{yu-ru.lin, hari.sundaram, aisling.kelliher}@asu.edu, {jimeng, castrop,rkonuru}@us.ibm.com

ABSTRACT

This paper aims at discovering community structure in rich media social networks, through analysis of time-varying, multi-relational data. Community structure represents the *latent* social context of user actions. It has important applications in information tasks such as search and recommendation. Social media has several unique challenges. (a) In social media, the context of user actions is constantly changing and co-evolving; hence the social context contains time-evolving multi-dimensional relations. (b) The social context is determined by the available system features and is unique in each social media website. In this paper we propose MetaFac (MetaGraph Factorization), a framework that extracts community structures from various social contexts and interactions. Our work has three key contributions: (1) metagraph, a novel relational hypergraph representation for modeling multi-relational and multi-dimensional social data; (2) an efficient factorization method for community extraction on a given metagraph; (3) an on-line method to handle time-varying relations through incremental metagraph factorization. Extensive experiments on real-world social data collected from the Digg social media website suggest that our technique is scalable and is able to extract meaningful communities based on the social media contexts. We illustrate the usefulness of our framework through prediction tasks. We outperform baseline methods (including aspect model and tensor analysis) by an order of magnitude.

Categories and Subject Descriptors

H.2.8 [Database Management]: Database Applications—*Data mining*; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—*Information filtering*; I.5.3 [Pattern Recognition]: Clustering; J.4 [Computer Applications]: Social and Behavioral Sciences—*Economics*

General Terms

Algorithms, Experimentation, Measurement, Theory

Keywords

MetaFac, metagraph factorization, relational hypergraph, non-negative tensor factorization, community discovery, dynamic social network analysis

1. INTRODUCTION

This paper aims at discovering community structure in rich media social networks, through analysis of the time-varying multi-relational data from social media websites. Social media websites

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
KDD '09, June 28– July 1, 2009, Paris, France.
Copyright 2009 ACM 978-1-60558-495-9/09/06...\$5.00.

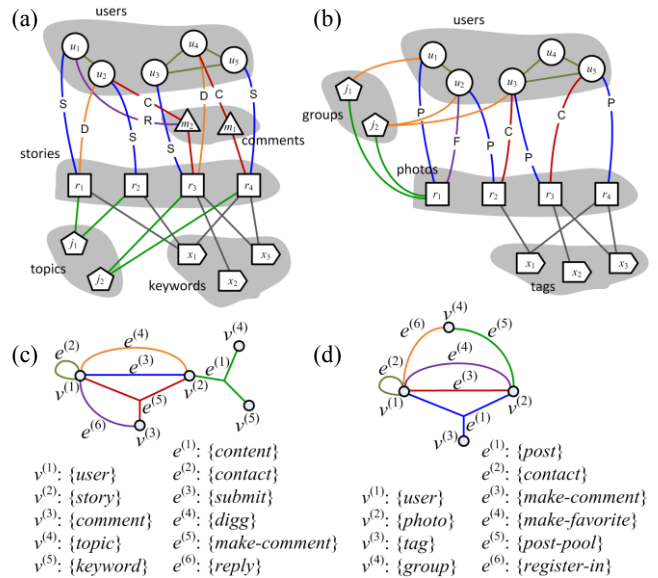


Figure 1: The social context of user actions vary across social media websites – we propose a metagraph representation to model various social context; (a) primary actions and related media objects in Digg; (b) primary actions and related objects in Flickr; (c) a metagraph representation for Digg; (d) a metagraph for Flickr.

such as Flickr, Digg and Facebook allow a wide array of actions on media objects – e.g. uploading photos, submitting and commenting on news stories, bookmarking and tagging, posting documents, creating web-links, as well as actions with respect to other users (e.g. sharing media and links with a friend). The key to social media information tasks such as media recommendation relies in understanding the context of these actions – how they relate to other actions, users and media objects. For example, a user might be motivated to search a story after viewing her friend’s bookmarks.

The problem has two challenges: (1) in social media, the context of user actions is constantly changing and co-evolving, e.g. with respect to other users’ actions, emergent concepts and users’ historic preferences. Hence the social context contains time-evolving multi-dimensional relations; (2) the social context is determined by the available system features that allow interactions on media objects and among people. Hence the social context is unique in each social media website. For example, Figure 1 shows the main actions available in Digg and Flickr, as well as related media objects. In Digg, users might submit / vote (digg) / comment on a story, reply to a comment, reply to a reply, etc. Flickr users might post, tag and comment on a photo, make friend

contacts, label a photo as a favorite, join a photo sharing group (pool), etc. There are some common actions, but more site-specific actions cater to the purpose of each site. The discovery of social context needs to deal with the diverse and dynamic nature of actions in social media.

In this paper we propose MetaGraph Factorization (MetaFac), a framework that extracts community structures (i.e. the latent social context) from various social interactions. Our work has three key contributions: (1) *metagraph*, a novel relational hypergraph representation for modeling multi-relational and multi-dimensional social data; (2) an efficient factorization method for community extraction on a given metagraph; (3) an on-line method to handle time-varying relations through incremental metagraph factorization.

Extensive experiments on real-world social media data suggest that our technique is scalable and is able to extract meaningful communities based on social media context. We illustrate the usefulness of our framework through prediction tasks – to predict users’ future interests on voting or commenting on Digg stories. Our prediction significantly outperforms baseline methods (frequency counts, tensor analysis, etc.), suggesting the utility of leveraging metagraphs to handle time-varying social relational contexts.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 introduces preliminaries and section 4 formalizes the problem. Section 5 and 6 presents our community extraction method on both static and dynamic multi-relational data. Section 7 presents experiments and section 8 concludes.

2. RELATED WORK

Community discovery in rich media social networks deals with a constantly changing “mishmash” of interrelated users and media objects. The problem has three aspects: (1) evolutionary characterization of communities in time-varying social networks, (2) analysis of multi-dimensional data, and (3) relational learning adaptable to different social contexts. To the best of our knowledge, our work is the first unified attempt to address all three aspects within a single problem.

Evolutionary community characterization. Social interactions among people have been studied through a unipartite or bipartite graph, in which the community structure can be characterized by clustering methods [14], and the evolution of community structure is captured in terms of various criteria. Kumar et al. [9] study the evolution of the blogosphere in terms of the change of graph statistics and the burstiness of extracted communities. Sun et al. [14] use the Minimum Description Length principle to extract communities and to detect their changes. Lin et al. [11] use an evolutionary clustering criterion [4] to extract community structures based on both observed networked data and historic community structure. All these works restrict themselves to pairwise relations between entities (e.g. user-user or user-paper). In rich online social media, networked data consists of multiple *co-evolving* dimensions, e.g. users, tags, feeds, comments, etc. Collapsing such multi-way networks into pairwise networks results in the loss of valuable information, and the analysis of temporal correlation among multi-dimensions is difficult.

Multi-dimensional mining. In multi-dimensional network analysis, networks have more than two types of entities. Existing techniques include tensor based analysis [5,14] or multi-graph mining. Tensor factorization is a generalized approach for analyzing multi-way interactions among entities. Note that a

tensor represents complete interactions among all involved entities, which is a very strong assumption in social media since there might be events involving some but not all dimensions. Multi-graph mining considers joint factorization over two or more matrices. The combination of such matrices is domain-specific, e.g. in text mining, Zhu et al. [15] propose a joint matrix factorization combining both linkage and document-term matrices to improve the hypertext classification. In social media, relations depend on the system features, which might vary across websites. Moreover, the system features may change over time in a social media website, which requires flexible relational learning.

Relational learning. Relational techniques such as PRMs [6] extend generative models to deal with various combinations of probabilistic dependency among entities. Such techniques can be computationally expensive, and may not scale to the large amount of data typically collected by social media websites. There have been relational learning techniques through pairwise relationships among entities, e.g. [3,12], which involve loss of information when data has higher-order interactions. Our work shares the same advantages as Kemp et al. [8] and Banerjee et al. [2], which deal with multiple tensors, but their static settings are different from our problem.

In sum, social media analysis requires a flexible and scalable framework that exploits relational context defined by the system features of individual social media sites. Such relational context is multi-dimensional, sparse (not all dimensions are involved in an event), specific, and evolving over time. We propose a unified approach to analyze the dynamics of rich media social networks.

3. PRELIMINARIES ON TENSOR

This section provides notations and minimal background on tensors and some basic operations used in this work. We refer readers to [1] for a more comprehensive review on tensors.

3.1 Tensors

A tensor is a mathematical representation of a multi-way array. The *order* of a tensor is the number of modes (or ways). A first-order tensor is a vector, a second-order tensor is a matrix, and a higher-order tensor has three or more modes. We use \mathbf{x} as a vector, \mathbf{X} as a matrix, and \mathcal{X} as a tensor. The *dimensionality* of a mode is the number of elements in that mode. We use I_q to denote the dimensionality of mode q . E.g., the tensor $\mathcal{X} \in \mathfrak{R}_+^{I_1 \times I_2 \times I_3}$ has 3 modes with dimensionalities of I_1 , I_2 and I_3 , respectively. \mathfrak{R}_+ indicates all elements of the tensor \mathcal{X} have nonnegative values, which is usually the case for a data tensor. The (i_1, i_2, i_3) -element of a third-order tensor is denoted by $x_{i_1 i_2 i_3}$. Indices typically range from 1 to their capital version, e.g. $i_1 = 1, \dots, I_1$.

3.2 Basic Operations

Mode- d matricization or unfolding: Matricization is the process of reordering the elements of an M -way array into a matrix. The mode- d matricization of a tensor $\mathcal{X} \in \mathfrak{R}^{I_1 \times \dots \times I_M}$ is denoted by $\mathbf{X}_{(d)}$,

i.e. $\text{unfold}(\mathcal{X}, d) = \mathbf{X}_{(d)} \in \mathfrak{R}^{I_d \times \prod_{q \in \{1, \dots, M\}, q \neq d} I_q}$. Unfolding a tensor on mode d results in a matrix with height I_d and its width is the product of dimensionalities of all other modes.

The inverse operation is denoted as $\mathcal{X} = \text{fold}(\mathbf{X}_{(d)}) \in \mathfrak{R}^{I_1 \times \dots \times I_M}$.

In general the unfolding operation can be defined on multiple modes. For example, we can define mode- (c, d) unfolding as

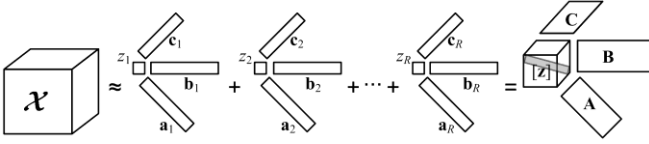


Figure 2: CP decomposition of a three-way tensor.

$unfold(\mathcal{X}, (c, d)) = \mathcal{X}_{(c,d)} \in \mathfrak{R}^{(I_c \times I_d) \times (\prod_{q=1, \dots, M, q \neq c, d} I_q)}$. Unfolding a tensor on two modes c and d results in a cube (three-way tensor). Similarly, we can define a *vectorization* operation $\mathbf{x} = vec(\mathcal{X})$, which linearizes the tensor into a vector.

Mode- d product: The mode- d matrix product of a tensor $\mathcal{X} \in \mathfrak{R}^{I_1 \times \dots \times I_M}$ with a matrix $\mathbf{U} \in \mathfrak{R}^{J \times I_d}$ is denoted by $\mathcal{X} \times_d \mathbf{U}$ and results in a tensor of size $I_1 \times \dots \times I_{d-1} \times J \times I_{d+1} \times \dots \times I_M$. Elementwise, we have $(\mathcal{X} \times_d \mathbf{U})_{i_1 \dots i_{d-1} j i_{d+1} \dots i_M} = \sum_{i_d=1}^{I_d} x_{i_1 i_2 \dots i_M} u_{j i_d}$.

Mode- d accumulation: A mode- d accumulation or summation is defined as $acc(\mathcal{X}, d) = \mathbf{X}_{(d)} \mathbf{1} \in \mathfrak{R}^{I_d}$. The operation sums up all entries across all modes except for mode d , which results in a vector of length I_d . Accumulating a tensor on mode d can be obtained by unfolding the tensor on mode d into a matrix and then multiplying the matrix with an all-one vector. Like unfolding operation, accumulation can be defined on multiple modes, e.g. a mode- (c, d) accumulation $acc(\mathcal{X}, (c, d)) = \mathcal{X}_{(c,d)} \times_3 \mathbf{1} \in \mathfrak{R}^{I_c \times I_d}$. This will result in a matrix of size $I_c \times I_d$.

Tensor decomposition or factorization is a form of higher-order principal component analysis. It decomposes a tensor into a core tensor multiplied by a matrix along each mode. Thus, in the three-way case where $\mathcal{X} \in \mathfrak{R}^{I_1 \times I_2 \times I_3}$, we have $\mathcal{X} \approx \mathbf{Z} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$, which means each element of the tensor \mathcal{X} is the product of the corresponding matrix elements multiplied by weight z_{pqr} , i.e.

$x_{ijk} \approx \sum_{p=1}^P \sum_{q=1}^Q \sum_{r=1}^R z_{pqr} a_{ip} b_{jq} c_{kr}$. Here, $\mathbf{A} \in \mathfrak{R}^{I_1 \times P}$, $\mathbf{B} \in \mathfrak{R}^{I_2 \times Q}$ and $\mathbf{C} \in \mathfrak{R}^{I_3 \times R}$ are called *factor matrices* or *factors* and can be thought of as the principal components of the original tensor along each mode. The tensor $\mathbf{Z} \in \mathfrak{R}^{P \times Q \times R}$ is called the *core tensor* and its elements show the level of interaction between different components. A special case of tensor decomposition is referred as CP or PARAFAC decomposition [1], where the core tensor is superdiagonal and $P=Q=R$. (A tensor $\mathcal{X} \in \mathfrak{R}^{I_1 \times \dots \times I_M}$ is diagonal if $x_{i_1 \dots i_M} \neq 0$ only if $i_1 = \dots = i_M$.) The CP decomposition of a third-order tensor is then simplified as $x_{ijk} \approx \sum_{r=1}^R z_r a_{ir} b_{jr} c_{kr}$, as

illustrated in Figure 2. We use $[\mathbf{z}]$ to denote a superdiagonal tensor, where the operation $[\cdot]$ transforms a vector \mathbf{z} to a superdiagonal tensor by setting tensor element $z_{k \dots k} = z_k$ and other elements as 0. Thus the CP decomposition of a three-way tensor can be written as $\mathcal{X} \approx [\mathbf{z}] \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$, where $[\mathbf{z}]$ denotes a corresponding superdiagonal core tensor.

4. PROBLEM FORMULATION

This section defines the problem of discovering latent community structure that represents the context of user actions in social networks. The problem has three parts: (1) how to represent multi-relational social data (section 4.1), (2) how to reveal the latent communities consistently across multiple relations, and (3) how to track the communities over time (section 4.2).

4.1 Metagraph Representation

We introduce *metagraph*, a relational hypergraph for representing multi-relational and multi-dimensional social data. We use a metagraph to configure the relational context specific to the system features – this is the key to making our community analysis adaptable to various social media contexts, e.g. Digg and Flickr (Figure 1). We shall use the Digg example to illustrate three concepts: *facet*, *relation*, and *relational hypergraph*.

As shown in Figure 1(a), Digg allows various actions for news sharing – users might submit (indicated by the line labeled “S”) a news story associated with a particular topic. They might vote (or digg, line “D”) or comment (line “C”) on the submitted story, reply (line “R”) to a comment created by other users, or even reply to a reply (not shown in the figure), etc. To describe the context of actions, we call a set of objects or entities of the same type a *facet*, e.g. a user facet is a set of users, a story facet is a set of stories, etc. We call the interactions among facets a *relation*; a relation can involve two (i.e. binary relation) or more facets, e.g. the “digg” relation involves two facets (user, story), and the “make-comment” is a 3-way relation (user, story, comment). A facet can be implicit, depending on whether the facet entities interact with other facets, e.g. the set of digg objects might be omitted due to no interactions with other facets.

Formally, we denote the q -th facet as $v^{(q)}$ and the set of all facets as V . A set of instantiations of an M -way relation e on facets $v^{(1)}, v^{(2)}, \dots, v^{(M)}$ is a subset of the Cartesian product $v^{(1)} \times \dots \times v^{(M)}$. We denote a particular relation by $e^{(r)}$ where r is the relation index. The observations of an M -way relation $e^{(r)}$ is represented as an M -way data tensor $\mathcal{X}^{(r)}$.

Now we introduce a *multi-relational hypergraph* (denoted as metagraph in this paper) to describe the combination of relations and facets in a social media context. A hypergraph is a graph where edges, called *hyperedges*, connect to any number of vertices. The idea is to use an M -way hyperedge to represent the interactions of M facets: each facet as a vertex and each relation as a hyperedge on a hypergraph. A metagraph defines a particular structure of interactions among facets, not among facet elements.

Formally, for a set of facets $V = \{v^{(q)}\}$ and a set of relations $E = \{e^{(r)}\}$, we construct a metagraph $G = (V, E)$. To reduce notational complexity, V and E also represent the set of all vertex and edge indices respectively. A hyperedge/relation $e^{(r)}$ is said to be *incident* to a facet/vertex $v^{(q)}$ if $v^{(q)} \in e^{(r)}$, which is represented by $v^{(q)} \sim e^{(r)}$ or $e^{(r)} \sim v^{(q)}$. E.g., in Figure 1(c) $v^{(1)}$ represents the user facet, $e^{(5)} = \{v^{(1)}, v^{(2)}, v^{(3)}\}$ represents the “make-comment” relation. We summarize our notations in Table 1.

Symbol	Description
\mathbf{x}	a vector (boldface lower-case letter)
\mathbf{X}	a matrix (boldface capital letter)
\mathcal{X}	a tensor (boldface Euler script letter)
I_1, \dots, I_M	the dimensionality of mode 1, ..., M
$v^{(q)}$	a vertex $v^{(q)} \in V$ represents the facet $v^{(q)}$
$e^{(r)}$	a hyperedge $e^{(r)} \subseteq V$ represents the relation $e^{(r)}$
V	the set of all facets $V = \{v^{(q)}\}$, or the set of all vertex indices
E	the set of all relations $E = \{e^{(r)}\}$ or all hyperedge indices
G	a metagraph $G = (V, E)$, where V is a set of facets/vertices and E is a set of relations/hyperedges
K, L	constants

Table 1: Description of notations.

4.2 Community Discovery on Metagraph

We formalize the community discovery problem as latent space extraction from multi-relational social data represented by a metagraph. Our goal is to discover latent community structures that represent the context of user actions in social media networks. We are interested in clusters of people who interact with each other in a coherent manner. Some of the interaction can be implicit, e.g. two users may comment on the same stories, and the interactions can be further enhanced by other interactions. Hence we consider a community as a latent space of consistent interactions or relations among users and objects.

By assuming consistent interactions in a community, the interaction between any two entities (users or media objects) i and j in a community k , written as x_{ij} , can be viewed as a function of the relationships between community k with entity i , and k with j . If we consider the function to be stochastic, i.e. let $p_{k \rightarrow i}$ indicate how likely an interaction in the k -th community involves the i -th entity and p_k is the probability of an interaction in the k -th community, we can express x_{ij} by $x_{ij} \approx \sum_k p_{k \rightarrow i} p_{k \rightarrow j} p_k$. Likewise a 3-way interaction among entity i_1 , i_2 and i_3 is $x_{i_1 i_2 i_3} \approx \sum_k p_k \cdot p_{k \rightarrow i_1} \cdot p_{k \rightarrow i_2} \cdot p_{k \rightarrow i_3}$. A set of such interactions among entities in facet $v^{(1)}$, $v^{(2)}$ and $v^{(3)}$ can be written by:

$$\mathcal{X} \approx \sum_{k=1}^K p_k \mathbf{u}_k^{(1)} \circ \mathbf{u}_k^{(2)} \circ \mathbf{u}_k^{(3)} = [\mathbf{z}] \prod_{m=1}^3 \times_m \mathbf{U}^{(m)}, \quad \langle 1 \rangle$$

where $\mathcal{X} \in \mathfrak{R}_+^{I_1 \times I_2 \times I_3}$ is the data tensor representing the observed three-way interactions among facet $v^{(1)}$, $v^{(2)}$ and $v^{(3)}$. $p_{k \rightarrow i_q}$ is written as an (i_q, k) -element of $\mathbf{U}^{(q)}$ for $q=1,2,3$. $\mathbf{U}^{(q)}$ is a $I_q \times K$ matrix, where I_q is the size of $v^{(q)}$. The probabilities of communities are elements of \mathbf{z} , i.e. $p_k = \mathbf{z}_k$. This is similar to the CP decomposition of a tensor (section 3.2), except that the core tensor $[\mathbf{z}]$ and the factor matrices $\{\mathbf{U}^{(q)}\}$ are constrained to contain nonnegative probability values. Under the nonnegative constraints, the 3-way tensor factorization is equivalent to the three-way aspect model in a three-dimensional co-occurrence data [13].

The nonnegative tensor decomposition can be viewed as community discovery in a single relation. The interactions in social media networks are more complex – usually involving multiple two- or multi-way relations. By using metagraphs, we represent a diverse set of relational contexts in the same form and define the community discovery problem on a metagraph, with the following two technical issues.

The first issue is how to extract community structure as coherent interaction latent spaces from observed social data defined on a metagraph, which is formally stated as follows.

Problem (Metagraph Factorization, or MF): given a metagraph $G=(V,E)$ and a set of observed data tensors $\{\mathcal{X}^{(r)}\}_{r \in E}$ defined on G , find a nonnegative core tensor $[\mathbf{z}]$ and factors $\{\mathbf{U}^{(q)}\}_{q \in V}$ for corresponding facets $V=\{v^{(q)}\}$. (Since E also represents the set of all edge indices, the notations $r \in E$ and $e^{(r)} \in E$ are exchangeable. Likewise, $q \in V$ and $v^{(q)} \in V$ are exchangeable.)

The second issue concerns the dynamic nature of human activities – those interactions might be consistent during a short time period but are unlikely to be consistent all the time. The problem, how to extract community structure as coherent interaction latent spaces from time evolving data given a metagraph, is defined as follows.

Problem (Metagraph Factorization for Time evolving data, or MFT): given a metagraph $G=(V,E)$ and a sequential set of

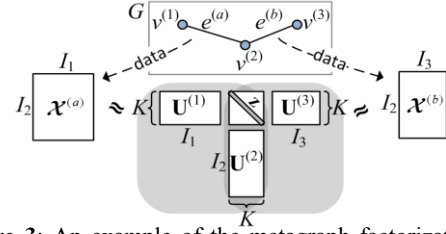


Figure 3: An example of the metagraph factorization (MF). Given observed data tensors $\{\mathcal{X}^{(a)}, \mathcal{X}^{(b)}\}$ and a metagraph G that describes the interaction among facets $\{v^{(1)}, v^{(2)}, v^{(3)}\}$, find a consistent community structure expressed by core tensor $[\mathbf{z}]$ and facet factors $\{\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}\}$.

observed data tensors $\{\mathcal{X}^{(r)}\}_{r \in E}$ defined on G for time $t=1,2,\dots$, find a nonnegative core tensor $[\mathbf{z}]$ and factors $\{\mathbf{U}_t^{(q)}\}_{q \in V}$ corresponding to facets $V=\{v^{(q)}\}$ for each time t .

We will present our method in two steps: (1) present a solution to MF (section 5); (2) extend the solution to solve MFT (section 6).

5. METAGRAPH FACTORIZATION

This section presents our solution to the metagraph factorization problem (MF). Our method relies on formulating MF as an optimization problem (section 5.1). We then provide an algorithm to solve the optimization objective (section 5.2) and discuss its computational complexity (section 5.3).

5.1 Optimization Objective

The MF problem can be stated in terms of optimization. Let us first consider a simple metagraph case. Assume we are given a metagraph $G=(V,E)$ with three vertices $V=\{v^{(1)}, v^{(2)}, v^{(3)}\}$ and two 2-way hyperedges $E=\{e^{(a)}, e^{(b)}\}$ that describe the interactions among these three facets, as shown in Figure 3. The observed data corresponding to the hyperedges are two second-order data tensors (i.e. matrices) $\{\mathcal{X}^{(a)}, \mathcal{X}^{(b)}\}$ with facets $\{v^{(1)}, v^{(2)}\}$ and $\{v^{(2)}, v^{(3)}\}$ respectively. The facet $v^{(2)}$ is shared by both tensors.

The goal is to extract community structure from data tensors, through finding a nonnegative core tensor $[\mathbf{z}]$ and factors $\{\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}\}$ corresponding to the three facets. The core tensor and factors need to consistently explain the data, i.e. we can approximately express the data by $\mathcal{X}^{(a)} \approx [\mathbf{z}] \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)}$ and $\mathcal{X}^{(b)} \approx [\mathbf{z}] \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)}$, as in eq.<1>. The core tensor $[\mathbf{z}]$ and facet $\mathbf{U}^{(2)}$ are shared by the two approximations, and the length of \mathbf{z} is determined by the number of latent spaces (communities) to be extracted. Since both the left- and the right-hand side of the approximation are probability distributions, it is natural to use the KL-divergence (denoted as $D(\cdot \parallel \cdot)$) as a measure of approximation cost. To simultaneously reduce two approximation costs we can define a cost function as:

$$D(\mathcal{X}^{(a)} \parallel [\mathbf{z}] \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)}) + D(\mathcal{X}^{(b)} \parallel [\mathbf{z}] \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)}), \quad \langle 2 \rangle$$

where $D(\mathcal{A} \parallel \mathcal{B}) = \sum_i (\mathbf{a}_i \log \mathbf{a}_i / \mathbf{b}_i - \mathbf{a}_i + \mathbf{b}_i)$ is the KL-divergence between tensor \mathcal{A} and \mathcal{B} and $\mathbf{a} = \text{vec}(\mathcal{A})$, $\mathbf{b} = \text{vec}(\mathcal{B})$.

The solution to eq.<2> will be an MF solution for the metagraph in Figure 3. We observe three things in this example: In eq.<2>, each $D(\cdot \parallel \cdot)$ correspond to a hyperedge, each tensor product corresponds to how facets are incident to an hyperedge and the summation corresponds to all hyperedges on the graph. We can generalize eq.<2> to any metagraph G , as follows.

Given a metagraph $G=(V,E)$, the objective is to factorize all data tensors such that all tensors can be approximated by a common nonnegative core tensor $[\mathbf{z}]$ and a shared set of nonnegative factors $\{\mathbf{U}^{(q)}\}$, i.e. to minimize the following cost function:

$$J(G) = \min_{\mathbf{z}, \{\mathbf{U}^{(q)}\}} \sum_{r \in E} D(\mathcal{X}^{(r)} \parallel [\mathbf{z}] \prod_{m: \mathcal{V}^{(m)} \sim e^{(r)}} \times_m \mathbf{U}^{(m)}) \quad <3>$$

$$s.t. \mathbf{z} \in \mathfrak{R}_+^{1 \times K}, \mathbf{U}^{(q)} \in \mathfrak{R}_+^{I_q \times K} \quad \forall q, \sum_i \mathbf{U}_{ik}^{(q)} = 1 \quad \forall q \forall k$$

where K is the number of communities, and $D(\cdot \parallel \cdot)$ is the KL-divergence as described above. The constraint that each column of $\{\mathbf{U}^{(q)}\}$ must sum to one is added due to the modeling assumption that the probability of an occurrence of a relation on an entity is independent of other entities in a community. Eq.<3> can be easily extended to incorporate weights on relations.

5.2 Algorithm

We provide a solution to the objective function defined in eq.<3>. It is difficult to guarantee a global minima solution, as eq.<3> is not convex in all variables. By employing the concavity of the log function (in the KL-divergence), we derive a local minima solution to eq.<3>. The solution can be found by the following updating algorithm:

$$\mathbf{z}_k \leftarrow \frac{1}{L} \sum_{r \in E} \sum_{i_1 \dots i_{M_r}} \mathcal{X}_{i_1 \dots i_{M_r}}^{(r)} \mu_{i_1 \dots i_{M_r}, k}^{(r)}, \quad <4>$$

$$\mathbf{U}_{i_q k}^{(q)} \leftarrow \frac{1}{L_q} \sum_{l: e^{(l)} \sim \mathcal{V}^{(q)}} \sum_{i_1 \dots i_{q-1} i_{q+1} \dots i_{M_q}} \mathcal{X}_{i_1 \dots i_{M_q}}^{(l)} \mu_{i_1 \dots i_{M_q}, k}^{(l)}, \quad <5>$$

where \mathbf{z} is a length K vector, $L=|E|$ denotes the total number of hyperedges on G , $L_q=|\{l: e^{(l)} \sim \mathcal{V}^{(q)}\}|$ denotes the number of hyperedges incident to $\mathcal{V}^{(q)}$, and

$$\mu_{i_1 \dots i_{M_r}, k}^{(r)} \leftarrow \frac{\mathbf{z}_k \prod_{m: \mathcal{V}^{(m)} \sim e^{(r)}} \mathbf{U}_{i_m k}^{(m)}}{([\mathbf{z}] \prod_{m: \mathcal{V}^{(m)} \sim e^{(r)}} \times_m \mathbf{U}^{(m)})_{i_1 \dots i_{M_r}}}. \quad <6>$$

After updates, each column of $\mathbf{U}^{(q)}$ are normalized to sum to one. Because of this normalization step, we can omit dividing by L_q in eq.<5>. This iterative update algorithm is a generalization of the algorithm proposed by Lee et al. [10] for solving the single nonnegative matrix factorization problem. In metagraph factorization, the update for core tensor $[\mathbf{z}]$ depends on all hyperedges on the metagraph, and the update for each facet factor $\mathbf{U}^{(q)}$ depends on the hyperedges incident to the facet. The proof for the convergence of our algorithm is omitted due to space limit.

The computation in eq.<4>–<6> can be time-consuming due to the high dimensionalities of tensors. We now discuss an efficient implementation of the update rules. In eq.<4>–<6>, $\mu_{i_1 \dots i_{M_r}, k}^{(r)}$ is an element of a $I_1 \times \dots \times I_{M_r} \times K$ tensor. Let $\mathcal{M}^{(r)} \in \mathfrak{R}_+^{\psi}$ denote this tensor, where ψ denotes the dimensionalities $I_1 \times \dots \times I_{M_r} \times K$ in short. Because $\mathcal{M}^{(r)}$ is expensive to compute and operate, we want to reduce computation that involves $\mathcal{M}^{(r)}$. By observing the shared part for updating the core tensor and all facet factors in eq.<4> and <5>, we can use the following strategy to achieve efficient computation: Instead of computing $\mathcal{M}^{(r)}$ explicitly, we compute an intermediate tensor $\mathcal{S}^{(r)}$ of the same dimensionalities as $\mathcal{M}^{(r)}$. $\mathcal{S}^{(r)}$ will save the repeating part of multiplication of $\mathcal{M}^{(r)}$ with $\{\mathbf{U}^{(q)}\}$ and \mathbf{z} in eq.<4> and <5>. Thus, the above update rules

can be rewritten as follows. First, for each $e^{(r)}$, compute a tensor $\mathcal{S}^{(r)} \in \mathfrak{R}_+^{\psi}$ by:

$$\mu^{(r)} \leftarrow \text{vec}(\mathcal{X}^{(r)} \oslash ([\mathbf{z}] \prod_{m: \mathcal{V}^{(m)} \sim e^{(r)}} \times_m \mathbf{U}^{(m)})) \quad <7>$$

$$\mathcal{S}^{(r)} = \text{fold}(\mu^{(r)} * (\mathbf{z} * \mathbf{U}^{(M_r)} * \dots * \mathbf{U}^{(1)})^T) \quad <8>$$

where \oslash denotes the element-wise division, and $*$ denotes the Khatri-Rao product. The Khatri-Rao product of two matrices \mathbf{A} and \mathbf{B} , denoted by $\mathbf{A} * \mathbf{B}$, is defined by $\mathbf{A} * \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1 \quad \mathbf{a}_2 \otimes \mathbf{b}_2 \quad \dots \quad \mathbf{a}_K \otimes \mathbf{b}_K]$, where \mathbf{a}_k and \mathbf{b}_k are the k^{th} column vectors of \mathbf{A} and \mathbf{B} respectively, and where $\mathbf{a} \otimes \mathbf{b}$ is the Kronecker product of \mathbf{a} and \mathbf{b} .

The second step is to update \mathbf{z} and $\{\mathbf{U}^{(q)}\}$ by:

$$\mathbf{z} \leftarrow \frac{1}{L} \sum_{r \in E} \text{acc}(\mathcal{S}^{(r)}, M_r + 1) \quad <9>$$

$$\mathbf{U}^{(q)} \leftarrow \sum_{l: e^{(l)} \sim \mathcal{V}^{(q)}} \text{acc}(\mathcal{S}^{(l)}, (M_l + 1, q)) \quad <10>$$

where $M_r + 1$ is the last mode of $\mathcal{S}^{(r)}$. The multiplication of $\mathcal{M}^{(r)}$ and $\mathcal{X}^{(r)}$ in eq.<4> and <5> is now pre-computed in eq.<7> and <8> by utilizing the Khatri-Rao product. To obtain \mathbf{z} and $\{\mathbf{U}^{(q)}\}$, we only need to accumulate $\mathcal{S}^{(r)}$ on the corresponding modes. $\{\mathbf{U}^{(q)}\}$ obtained from eq.<10> will be equivalent to those from eq.<5> after normalization. Eq.<7>–<10> yield exactly the same results as eq.<4>–<6>. The algorithm shares the same form of the expectation-maximization algorithm, where eq.<7> and <8> correspond to the E-step and eq.<9> and <10> correspond to the M-step. Note that the information contained in each data tensor with respect to a hyperedge is aggregated through the E-step and is shared by the core tensor and all facet factors in the M-step, thus the extracted communities will be coherent latent spaces. Table 2 summarizes the whole process to solve an MF problem.

Algorithm 1: MF

Input: metagraph $G = (V, E)$ and data tensors $\{\mathcal{X}^{(r)}\}$ on G

Output: \mathbf{z} and $\{\mathbf{U}^{(q)}\}$

Method: Initialize \mathbf{z} , $\{\mathbf{U}^{(q)}\}$

Repeat until *convergence*

For each $r \in E$, compute $\mathcal{S}^{(r)}$ by eq.<7> and <8>

update \mathbf{z} by eq. <9>

For each $q \in V$, update $\mathbf{U}^{(q)}$ by eq. <10>

Table 2: The MF (metagraph factorization) algorithm.

We refer the solution core tensor and facet matrices as a community model, from which we infer the probabilistic (soft) membership of entities in each facet. As described in section 4.2, each (i, k) -element of a facet matrix \mathbf{U} is $p(i|k)$ (i.e. $p_{k \rightarrow i}$, how likely an interaction in the community k involves entity i), and each element $\mathbf{z}_k = p(k)$ (i.e. p_k , the probability of an interaction in community k). Thus we compute the conditional probability $p(k|i)$ to indicate the soft membership of entity i with respect to community k by $p(i|k)p(k)/p(i)$, where $p(i) = \sum_k p(i|k)p(k)$ is the probability of an interaction involving entity i .

5.3 Computational Complexity

We now discuss the time complexity for the updates. The most time-consuming step in the algorithm is to compute $\mathcal{S}^{(r)}$ for each hyperedge $e^{(r)}$. As can be seen in eq.<7>, we can take advantage of the sparseness of the data tensor $\mathcal{X}^{(r)}$ and compute only the

non-zero elements (total number of tuples) in $\mathcal{X}^{(r)}$. Let n denote the largest number of non-zero elements of the involved data tensors. This step has time complexity $O(nKML)$, where K is the number of clusters, M is the maximal number of incident facets of a relation, and L is the total number of input relations. Usually, K, M, L are much smaller than n . If we consider K, M and L are bounded by some constants, the time complexity per iteration is linear in $O(n)$, the number of non-zero elements in all data tensors.

6. TIME EVOLVING EXTENSION

This section presents our solution to the problem of metagraph factorization with time evolving data (MFT).

6.1 Optimization Objective

In the MFT problem, the relational data is constantly changing as evolving tensor sequences. We propose an online version of MF to handle dynamic data. Since historic information is contained in the community model extracted based on previously observed data, the new community structure to be extracted should be consistent with previous community model and new observations, which is similar to the evolutionary clustering discussed in [11]. To achieve this, we extend the objective in eq.<3> as follows.

A community model for a particular time t is defined uniquely by the factors $\{\mathbf{U}_t^{(q)}\}$ and core tensor $[\mathbf{z}_t]$. (To avoid notation clutter, we omit the time indices for t .) For each time t , the objective is to factorize the observed data into the nonnegative factors $\{\mathbf{U}^{(q)}\}$ and core tensor $[\mathbf{z}]$ which are close to the prior community model, $[\mathbf{z}_{t-1}]$ and $\{\mathbf{U}_{t-1}^{(q)}\}$. We introduce a cost l_{prior} to indicate how the new community structure deviates from the previous structure in terms of the KL-divergence. The new objective is defined as follows:

$$J_2(G) = \min_{\mathbf{z}, \{\mathbf{U}^{(q)}\}} (1-\alpha) \sum_{r \in E} D(\mathcal{X}^{(r)} \parallel [\mathbf{z}]) \prod_{m, \mathcal{Y}^{(m)} \sim \mathcal{C}^{(r)}} \times_m \mathbf{U}^{(m)} + \alpha l_{prior}$$

$$l_{prior} = D(\mathbf{z}_{t-1} \parallel \mathbf{z}) + \sum_q D(\mathbf{U}_{t-1}^{(q)} \parallel \mathbf{U}^{(q)}) \quad <11>$$

$$s.t. \mathbf{z} \in \mathfrak{R}_+^{1 \times K}, \mathbf{U}^{(q)} \in \mathfrak{R}_+^{J_q \times K} \quad \forall q, \sum_i \mathbf{U}_{ik}^{(q)} = 1 \quad \forall q \forall k$$

where α is a real positive number between 0 and 1 to specify how much the prior community model contributes to the new community structure. l_{prior} is a regularizer used to find similar pairs of core tensors and pairs of facet factors for consecutive times. The new community structure will be a solution incrementally updated based on a prior community model.

6.2 Algorithm

Based on a derivation similar to the discussion in section 5, we provide a solution to eq. <11> as follows:

$$\mathbf{z}_k \leftarrow (1-\alpha) \sum_{r \in E} \sum_{i_1 \dots i_{M_r}} \mathcal{X}_{i_1 \dots i_{M_r} k}^{(r)} \mu_{i_1 \dots i_{M_r} k}^{(r)} + \alpha \mathbf{z}_{k:t-1}, \quad <12>$$

$$\mathbf{U}_{i_q k}^{(q)} \leftarrow (1-\alpha) \sum_{l: \mathcal{C}^{(l)} \sim \mathcal{Y}^{(q)}} \sum_{i_1 \dots i_{q-1} i_{q+1} \dots i_{M_l}} \mathcal{X}_{i_1 \dots i_{M_l} k}^{(l)} \mu_{i_1 \dots i_{M_l} k}^{(l)} + \alpha \mathbf{U}_{i_q k:t-1}^{(q)}, \quad <13>$$

where $\mu_{i_1 \dots i_{M_r} k}^{(r)}$ is defined as in eq.<6>. After updates, each column of $\mathbf{U}^{(q)}$ and the vector \mathbf{z} are normalized to sum to one. Because of this normalization step, we have dropped the scaling constant for updating \mathbf{z} and $\mathbf{U}^{(q)}$.

It can be shown that the parameters in the previous model (\mathbf{z}_{t-1} and $\{\mathbf{U}_{t-1}^{(q)}\}$) act as Dirichlet prior distribution to inform the solution search (ref. [11]), thus the solution is consistent with previous community structure. The update rules can be rewritten as the following operations with $\mathcal{S}^{(r)}$ pre-computed by eq.<7> and <8>:

$$\mathbf{z} \leftarrow (1-\alpha) \sum_{r \in E} acc(\mathcal{S}^{(r)}, M_r + 1) + \alpha \mathbf{z}_{t-1} \quad <14>$$

$$\mathbf{U}^{(q)} \leftarrow (1-\alpha) \sum_{l: \mathcal{C}^{(l)} \sim \mathcal{Y}^{(q)}} acc(\mathcal{S}^{(l)}, (M_l + 1, q)) + \alpha \mathbf{U}_{t-1}^{(q)} \quad <15>$$

where M_r+1 is the last mode of $\mathcal{S}^{(r)}$. The whole process of finding solutions to the MFT problem is summarized in Table 3.

Algorithm 2: MFT

Input: hypergraph $G = (V, E)$, the data tensors $\{\mathcal{X}^{(r)}\}$ on G observed at

Output: new model \mathbf{z} and $\{\mathbf{U}^{(q)}\}$

Method: Initialize $\mathbf{z}, \{\mathbf{U}^{(q)}\}$

Repeat until *convergence*

For each $r \in E$, compute $\mathcal{S}^{(r)}$ by eq.<7> and <8>

update \mathbf{z} by eq. <14>

For each $q \in V$, update $\mathbf{U}^{(q)}$ by eq. <15>

Table 3: The MFT algorithm.

For time evolving social data, changes might happen in interactions among entities, or even in interactions among facets (e.g. due to the evolution of system features) which lead to changes in metagraph. One advantage of our MFT algorithm is it only requires new observed data defined on any given metagraph, so it is straightforward to incorporate the changes of a metagraph.

7. EXPERIMENTS

This section reports our experimental study on a real-world social media dataset collected from Digg. We first describe the dataset (section 7.1) and present the extracted communities (section 7.2). We evaluate our technique through prediction tasks (section 7.3). Finally, we evaluate the scalability of our factorization method on synthetic datasets (section 7.4).

7.1 Digg Dataset

We have collected data from a large set of user actions from Digg. Digg is a popular social news aggregator that allows users to submit, vote (i.e. digg) and comment on news stories. It also allows users to create social networks by designating other users as friends and tracking friends' activities. The dataset used in our experiments include stories, users and their actions (submit, digg, comment and reply) with respect to the stories, as well as the explicit friendship (contact) relation among these users. To analyze users' topical interests, we also retrieve the topics of the stories and extract keywords from the stories' titles.

Relation	Tensor / incident facets	#Tuples
(R1) content	dynamic (story, keyword, topic)	151,779
(R2) contact	static (user, user)	56,440
(R3) submit	dynamic (user, story)	44,005
(R4) digg	dynamic (user, story)	1,157,529
(R5) comment	dynamic (user, story, comment)	241,800
(R6) reply	dynamic (user, comment)	94,551

Table 4: Summary of the relations in Digg dataset.

From this dataset, we select 5 facets (user, story, comment, keyword and topic) and build 6 relations among them. The relations are summarized in Table 4, which correspond to the metagraph shown in Figure 1(c). Except for the contact relation, all relations have timestamps. We assume the contact relation is static and consider the other relations as dynamic. For dynamic relations, we extract tuples with timestamps ranging from August 1 to August 27, 2008. To study the data evolution, we segment the duration into 9 time slots (i.e. every three days), and construct a sequence of data tensors for each dynamic relation. In the

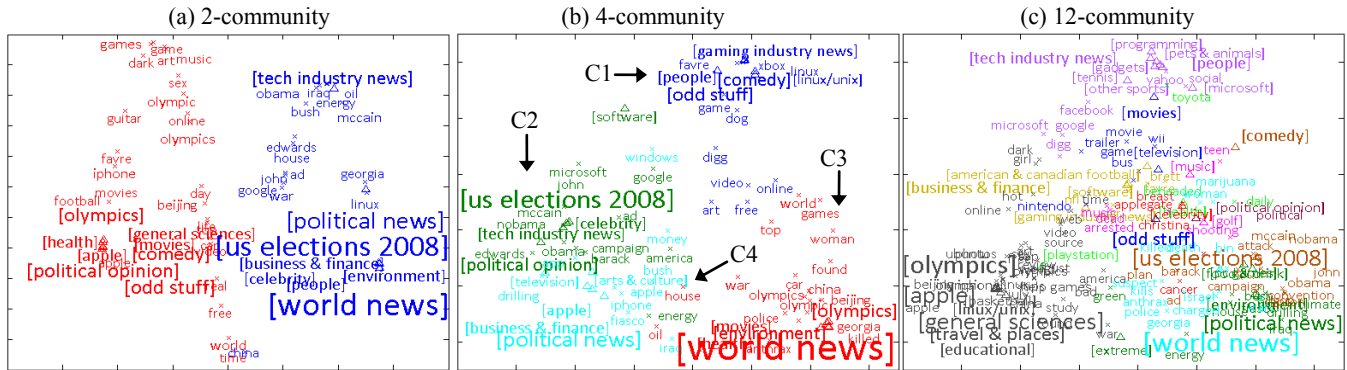


Figure 4: Community extracted based on user digging activities, for time $t=3$ (August 6-9, 2008) and number of communities $K=2, 4$ and 12. The most likely keyword and topic terms (shown within brackets) in each community are projected based on their soft membership. The size of each term indicates its probability and each term is colored based on its most likely community. The results show coherent topical preference in communities, as the terms with the same colors are located closely.

following we shall use $t \in [1, 9]$ to denote a time slot index. The total number of tuples in each tensor sequence per relation is listed in Table 4. Our dataset and code are available online (<http://www.public.asu.edu/~ylin56/kdd09sup.html>).

7.2 Community Analysis

We present a qualitative analysis of the communities extracted by our method, which demonstrates an advantage of probabilistic interpretation given by our method. We first show all communities extracted for a particular time and then examine the community evolution within two of these communities.

To illustrate what kinds of stories are “dugg” by what kind of communities, we track the latent communities based on the digging activities which involve relation R1 and R4. Figure 6(a) and (c) shows the corresponding metagraph and the number of tuples in the two relations. In our factorization algorithm, we assume that the number of communities, K , is given beforehand. Here we show communities extracted given $K=2, 4$ and 12.

Based on relation R1 and R4, four facets are involved: user, story, keyword and topic. We present the keyword and topic facets because they are more informative to the readers than other facets. Figure 4 shows the most likely keywords and topics in each community. We present the results of $t=3$ (August 6-9, 2008). We project those keyword and topic (shown within brackets) terms onto a 2D plane. The location of the i -th keyword or topic term indicates its relative proximity to other terms and is computed based on its soft membership $p(k|i)$. (The position is determined by standard multidimensional scaling with the soft membership as input.) The size of the i -th term indicates how likely the term appears in a story and is determined based on the probability $p(i)$. Each term is colored based on its most likely community, i.e. by choosing k with maximal $p(k|i)$. In the figure we can see the communities based on users’ digging activities have coherent topical preference, as the terms with the same colors are located closely. The 2-, 4- and 12-community results show the communities at different resolution. The 2-community result distinguishes political interests from the Olympics news (Figure 4(a)). The 4-community shows four topical interests in communities: C1: gaming industry news, C2: US election news, C3: world news, and C4: general political news (Figure 4(b)). The two major topics (“olympics” and “georgia”) in C3 are further split in the 12-community result (Figure 4(c)).

Community Evolution. We select the 4-community result and examine its evolution. Figure 5(a) shows the probabilities of the

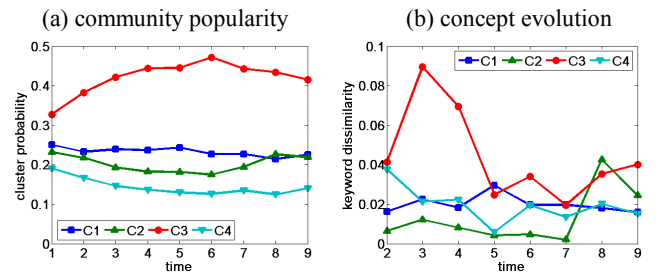


Figure 5: The community evolution characterized based on (a) change in size of communities, and (b) change in keyword distribution in each community.

four communities over time, and Figure 5(b) shows the keyword dissimilarity across time where the dissimilarity is computed based on the cosine similarity of keyword distribution in each community of consecutive timestamps. (We use a cosine similarity measure in order to emphasize the differences at the “head” of the distributions.) We observe two critical times in Figure 5(b): for community C2 and C3, the keywords distribution change drastically at $t=3$ (August 6-9) and $t=8$ (August 21-24). To examine the events occurring during these times, we look at the keyword distributions of the two communities. Table 5 lists the top 10 keywords that are mostly likely to appear in C3 and C2, at $t=2,3$ and $t=7,8$ respectively. At $t=3$, the new popped keywords “olympics” and “georgia” reflect users’ attention to two significant world news items: the 2008 Summer Olympics began on August 8 and the 2008 Russia-Georgia conflict started on August 7. At $t=8$, the new popped keywords “joe”, “biden”, “vp” correspond to the time when presidential candidate Barack Obama announced that Joe Biden would be his running mate (on August 22). Another critical time is captured by the change in community size. In Figure 5(a), we see the community C3 keeps growing until $t=6$, when the Russia-Georgia conflict ended with a ceasefire agreement signed on August 15 and 16.

C3		C2	
t=2	t=3	t=7	t=8
olympic 0.007	olympics 0.01	mccain 0.048	mccain 0.048
china 0.006	olympic 0.01	obama 0.039	obama 0.043
beijing 0.006	beijing 0.009	john 0.009	biden 0.018
anthrax 0.005	china 0.007	google 0.006	john 0.01
police 0.004	top 0.006	barack 0.005	joe 0.008
olympics 0.004	georgia 0.005	campaign 0.005	vp 0.007
found 0.004	war 0.004	nobama 0.005	google 0.006
woman 0.004	world 0.004	war 0.004	barack 0.005
top 0.004	games 0.004	georgia 0.004	ad 0.005
world 0.004	killed 0.004	convention 0.004	news 0.005

Table 5: The keyword distribution of community C2 and C3 during two critical times, $t=3$ and $t=8$.

The characterization of community evolution based on change in the probability of a cluster and on change in the distribution of entities such as keywords (Figure 5 and Table 5) demonstrates the advantage of our soft clustering method. The presented case study suggests that our method is able to generate meaningful mining results from dynamic multi-relational social networks.

7.3 Evaluation via Prediction

We use a prediction task to demonstrate the utility of our techniques. Based on the Digg scenario, we design two prediction tasks – to predict users’ future interests on digging (i.e. voting) and commenting on Digg stories. We study three aspects of our method through the prediction tasks: (1) How does our community discovery framework help predict users’ *future* interests? (2) How much historic information do we need? (3) Which relation is relevant to the prediction?

Prediction setting. There are two tasks: (a) digg prediction – what stories a user will digg, and (b) comment prediction – what stories a user will comment on. Both tasks are evaluated on data from each time slot. We use stories that have digging or commenting events in time slot $t_s \in [2, 9]$ as testing sets and the available relational data (ref. Table 4) in time slot $t_s - 1$ as training sets. The prediction results are compared with the actual diggs and comments occurring in slot t_s . This is a constrained setting because there might be more digging or commenting activities occurring after t_s . In our prediction experiments we only consider diggs and comments in each single slot t_s as ground truth.

Evaluation metrics. We use two metrics adopted in Information Retrieval: (1) “P@10” (the precision of the top 10 results): For each user we compute the precision based on the top 10 stories retrieved for the user. The overall P@10 for the set of users is computed by taking the mean of P@10 per user, per time slot. (2) “NDCG” (Normalized Discount Cumulative Gain [7]): One advantage of the measure is its sensitivity to the prediction order. The NDCG is proportional to $\sum_i \delta(i) / \log(1+i)$, where i is the rank of predicted stories, $\delta(i)=1$ if the prediction of the rank- i story is correct and 0 otherwise.

Our prediction method. We generate predictions based on the community structure extracted by our method, denoted by MF and MFT. The MF algorithm outputs community structure from relational data of each time slot $t_s - 1$. The MFT algorithm uses the same data as MF, with the aid of a community model extracted for time $t_s - 2$ as an informative prior. Hence MFT gives results incrementally. From an extracted community model we obtain the probability of a community k , $p(k)$, and the probability of a user u , a keyword w and a topic j , given community k , i.e. $p(u|k)$, $p(w|k)$ and $p(j|k)$. To predict if a user u will digg or comment on a story r , we first use a folding-in technique (ref. e.g. [13]) to compute $p(r|k)$, the probability of a story given each community k , based on the story topic and keywords. Then a prediction is made based on the condition probability $p(r|u) \propto p(u, r) \approx \sum_k p(k) p(u|k) p(r|k)$.

Baseline methods. Three baseline methods are used: (1) Frequency based heuristics (FREQ) – predicting stories based on the frequencies of story topic and keywords at $t_s - 1$. (2) Standard tensor analysis (PARAFAC) – predicting stories by using the CP/PARAFAC tensor decomposition [1] for data in slot $t_s - 1$. The stories to be predicted are first projected on the latent spaces, and the prediction is made based on the dot product of the user and story projected vectors. (3) Multi-way aspect model (MWA) – predicting stories by using the multi-way aspect model [13], a special case of our model (ref. section 4.2).

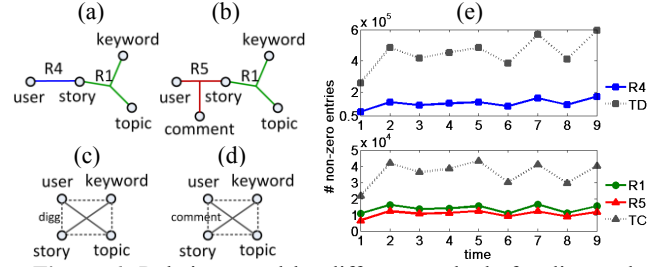


Figure 6: Relations used by different methods for digg and comment prediction: (a) R1 and R4 used in our method for digg prediction; (b) R1 and R5 used in our method for comment prediction; (c) TD tensors used in PARAFAC and MWA for digg prediction; (d) TC tensors used in PARAFAC and MWA for comment prediction; (e) number of tuples in each relation over time.

The ability to handle relational contexts is the key to our comparison. We choose specific relations to illustrate the utility of leveraging a specific context by a metagraph – relation R1 and R4 for digg prediction and R1 and R5 for comment prediction (ref. Figure 6(a) and (b)), and we shall evaluate the effect of other relations later in this section. Since PARAFAC and MWA only deal with a single high dimensional relation, we construct two 4-way tensors per time that contains digg actions and comment actions with respect to stories. The two tensors, denoted by TD and TC are shown in Figure 6(c) and (d). Figure 6(e) shows the number of non-zero entries (tuples) of these data tensors over time. The number of tuples in an R5 tensor corresponds to the number of stories per time.

Except for the FREQ method, all methods are tested with number of clusters or latent spaces $K=4 \dots 20$. For MFT, we use $\alpha=0.2$.

metric \ method	digg prediction		comment prediction	
	P@10	NDCG	P@10	NDCG
FREQ	0.175±0.061	0.035±0.016	0.015±0.007	0.004±0.002
PARAFAC	0.369±0.004	0.145±0.002	0.049±0.002	0.018±0.001
MWA	0.195±0.002	0.069±0.001	0.067±0.001	0.020±0.000
MF	0.529±0.008	0.212±0.002	0.117±0.001	0.038±0.000
MFT	0.543±0.007	0.215±0.004	0.135±0.001	0.043±0.000

Table 6: The average prediction performance for digg and comment prediction, evaluated by P@10 and NDCG metrics.

Results and Discussion. The overall prediction performance is obtained by taking the average of prediction performance on data for each time slot ($t=1 \dots 8$ for training and $t=2 \dots 9$ for testing) over different K values. The results (mean and standard deviation) are given in Table 6. There are several observations. First, our method significantly outperforms all baseline methods. In digg prediction, our MF method outperforms the baselines by 43% to 5X on the average. In comment prediction, the MF method outperforms the baselines by 73% to 10X. Second, the MFT performs the best. It slightly outperforms MF in digg prediction and improves MF by 15% in comment prediction. Next we show how our prediction can be further improved by (a) incorporating a historic model and (b) leveraging other relations through a metagraph.

Effect of historic information. We vary the weight of the prior model in MFT and report the average P@10 over α values (Figure 7(a)). The results suggest that incorporating historic information as prior knowledge works better than no prior ($\alpha=0$). Note for comment prediction, the performance increases until $\alpha \leq 0.8$. This suggests that the comment activities are more consistent with the historic community structure than the digg activities.

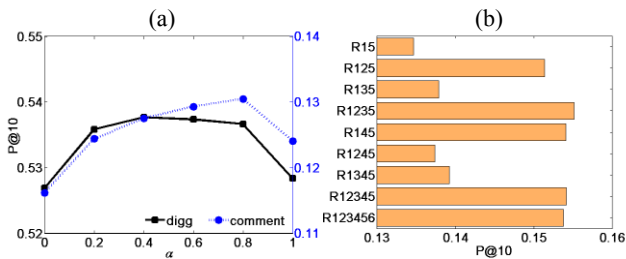


Figure 7: Effect of (a) prior community model (historic information) and (b) different input relations.

Effect of various relational context. For comment prediction, we evaluate the prediction performance over different relational contexts. Figure 7(b) shows the average prediction results. The label R^* indicates which relations are used in the training set, e.g. $R125$ denotes relation $R1$, $R2$ and $R5$. We observe that different combinations of the relations affect the prediction performance. For example, incorporating the contact relation $R2$ with $R1$ and $R5$ significantly helps predict users' comment activities.

7.4 Scalability Evaluation

We use synthetic datasets to illustrate the scalability of our algorithms. We study how the computational time of our algorithm increases with four variables, including different types of data growth – (a) non-zero elements in a data tensor, (b) number of tensor modes (dimensions), (c) number of relations (tensors) on a given metagraph, as well as (d) the algorithm parameter, i.e. number of clusters. In the simulation we randomly generate tensors by varying one of the above variables (e.g. the number of non-zero elements) and fixing all remaining variables.

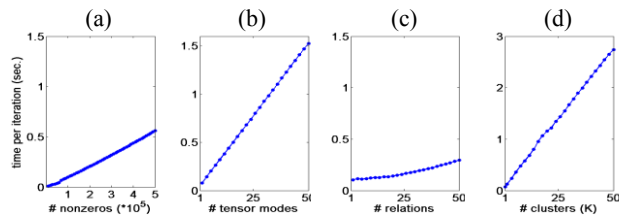


Figure 8: Running time per iteration (sec.) for different types of data growth (let n denote the value on the x -axis of each plot): (a) number of non-zero elements (one 3-way tensor with n non-zero elements), (b) number of tensor modes (one n -way tensor), (c) number of relations (n 3-way tensors) in a metagraph, and (d) for different algorithm parameter, the number of clusters (K).

Figure 8 shows the simulation results, indicating that the running time per iteration scales linearly with the data size, the number of tensor modes, the total number of relations and the number of clusters. Note that the slope for increasing tensor modes is steeper than increasing relations. Empirically the non-zero elements in a higher mode tensor are usually much more than lower mode tensors (as in Figure 6(e)). Therefore by leveraging a metagraph we can efficiently combine multiple low-dimensional relations instead of constructing a high-dimensional tensor. The experimental results on the synthetic datasets correspond to our analysis in section 5.3 and suggest that our algorithm can efficiently deal with large sparse multi-relational data.

8. CONCLUSION

We proposed the MetaFac framework to extract community structures from various social contexts and interactions. There

were three key ideas: (1) metagraph, a relational hypergraph for representing multi-relational social data; (2) MF algorithm, an efficient non-negative multi-tensor factorization method for community extraction on a given metagraph; (3) MFT, an on-line factorization method to handle time-varying relations. We conducted extensive experiments on real-world data collected from Digg. Our case study demonstrated that meaningful mining results can be generated by our method. We evaluated our method by predicting digg / comment actions on stories. We generated the predictions based on the extracted community models and compare results with baselines. Our method outperformed baseline measures up to an order of magnitude. Our method can be further improved by (a) incorporating a historic model and (b) leveraging other relations through a metagraph. As part of our future work, we plan to investigate the following open issues: (1) the resolution (including number of communities) of community structures; (2) kernel based representation of facet relationships.

9. REFERENCES

- [1] B. BADER and T. KOLDA (2006). *Algorithm 862: MATLAB tensor classes for fast algorithm prototyping*. *TOMS* 32(4): 635-653.
- [2] A. BANERJEE, S. BASU and S. MERUGU (2007). *Multi-way Clustering on Relation Graphs*, *SDM*, 2007.
- [3] R. BEKKERMAN, R. EL-YANIV and A. MCCALLUM (2005). *Multi-way distributional clustering via pairwise interactions*, *ACM Intl. Conf. Proc. Series*, 41-48.
- [4] D. CHAKRABARTI, R. KUMAR and A. TOMKINS (2006). *Evolutionary clustering*, *SIGKDD*, 554-560.
- [5] Y. CHI, S. ZHU, Y. GONG and Y. ZHANG (2008). *Probabilistic Polyadic Factorization and Its Application to Personalized Recommendation*, *CIKM*, 2008.
- [6] N. FRIEDMAN, L. GETOOR, D. KOLLER and A. PFEFFER (1999). *Learning probabilistic relational models*, *IJCAI*, 1300-1309, 1999.
- [7] K. JÄRVELIN and J. KEKÄLÄINEN (2000). *IR evaluation methods for retrieving highly relevant documents*, *SIGIR*, 41-48, 2000.
- [8] C. KEMP, J. TENENBAUM, T. GRIFFITHS, T. YAMADA and N. UEDA (2006). *Learning Systems of Concepts with an Infinite Relational Model*, *Proc. of the Natl. Conf. on AI*, 381.
- [9] R. KUMAR, J. NOVAK and A. TOMKINS (2006). *Structure and evolution of online social networks*, *SIGKDD*, 611-617, 2006.
- [10] D. LEE and H. SEUNG (2001). *Algorithms for non-negative matrix factorization*, *NIPS*, 556-562, 2001.
- [11] Y. LIN, Y. CHI, S. ZHU, H. SUNDARAM and B. TSENG (2008). *Facetnet: a framework for analyzing communities and their evolutions in dynamic networks*, *WWW*, 2008.
- [12] B. LONG, Z. ZHANG and P. YU (2007). *A probabilistic framework for relational clustering*, *SIGKDD*, 470-479.
- [13] A. POPESCU, L. H. UNGAR, D. M. PENNOCK and S. LAWRENCE (2001). *Probabilistic Models for Unified Collaborative and Content-Based Recommendation in Sparse-Data Environments*, *UAI 2001*, 437-444.
- [14] J. SUN, C. FALOUTSOS, S. PAPANIMITRIOU and P. YU (2007). *GraphScope: parameter-free mining of large time-evolving graphs*, *SIGKDD*, 687-696, 2007.
- [15] S. ZHU, K. YU, Y. CHI and Y. GONG (2007). *Combining content and link for classification using matrix factorization*, *SIGIR*, 487-494, 2007.