iOLAP: A Framework for Analyzing the Internet, Social Networks, and Other Networked Data

Yun Chi, Shenghuo Zhu, Koji Hino, Yihong Gong, *Member, IEEE*, and Yi Zhang

***Abstract*—As the amount of noisy, unorganized, linked data on the Internet increases dramatically, how to efficiently analyze such data becomes a challenging research problem. In this paper, we propose a framework, iOLAP, that offers functionalities for analyzing networked data from Internet, social networks, scientific paper citations, etc. We first identify four main data dimensions that are common in most of networked data, namely people, relation, content, and time. Motivated by the fact that various dimensions of data jointly affect each other, we propose a polyadic factorization approach to directly model all the dimensions simultaneously in a unified framework. We provide detailed theoretical analysis of the new modeling framework. In addition to the theoretical framework, we also present an efficient implementation of the algorithm that takes advantage of the sparseness of data and has time complexity linear in the number of data records in a dataset. We then apply the proposed models to analyzing the blogosphere and personalizing recommendation in paper citations. Extensive experimental studies showed that our framework is able to provide deep insights jointed obtained from various dimensions of networked data.**

***Index Terms*—Information filtering, knowledge management applications, modeling structured, personalization, textual and multimedia data, web mining.**

I. Introduction

A. Internet OLAP

THE advent of online publishing has dramatically lowered the threshold for ordinary people to publish their own experiences and opinions. This has resulted in the explosion of information that comprises huge amounts of noisy, unorganized, linked data on the Internet in the form of blogs, discussion forums, and other social networks. Among these huge amounts of data, only few of them are of high quality, and are maintained by professional organizations. If we use internet search engines to perform information retrieval, most likely we will get a ranking distribution that follows a power-law distribution where only a handful of dominant websites receive high ranks, and the majority of data are in the long tail. In other words, internet search engines only take good care of a small number of dominant websites, and mostly ignore the data in the long tail. However, long tail data also contain a lot of valuable information, such as grassroots opinions, sentiments, wisdom of crowds, etc. Such information is more valuable for the purpose of business intelligence, better decision making, and market strategies. We need new technologies to dig out such valuable information from the long tail data.

In terms of technology development path, we see a lot of analogies between the database area and the Internet area. During the 1960s and 1970s, researchers focused on the development of relational database technologies to address the query and retrieval needs on structured data. However, people gradually recognized that query and retrieval do not cover data access needs from all kinds of users. Quite often, instead of retrieving a small subset of data, users want to know the overall picture, or want to visualize the distribution of the entire data set. This kind of abilities is more important for business intelligence, and decision making supports. Because of this, during the 1980s and 1990s, researchers developed database OLAP technologies to provide analysis, summarization and visualization tools on structured data.

We envision a similar technology evolution path in the Internet area. During the 1990s and 2000s, researchers focused on the development of internet search engines to address the query and retrieval needs on unorganized, linked data. Nowadays, there is an increasing need for OLAP-like technologies on unorganized data for business intelligence and decision making supports. However, there is no matured technology for this purpose in the market. Although in recent years we see many researchers and start-up companies developing various technologies for data analysis, summarizations, visualization, etc., in general all these research activities provide one or two specific analytic abilities using ad-hoc approaches. So far nobody is taking a systematic approach to accomplishing Internet OLAP.

B. Four Dimensions of Networked Data

Although networked data on the Web are usually unorganized, they are not totally unstructured. Most of these networked data share certain essential features. In our iOLAP framework, we have identified four such essential features that can serve as good descriptors of networked data and can answer commonly asked queries about networked data. These four features, for which we call the four *dimensions*, of networked data are

- **People**: the generators of social media, e.g., bloggers in the blogosphere and authors in a paper citation network;
- **Relation**: the interrelationship among people, e.g., friendship in social networks and coauthorship in research communities;

Manuscript received June 06, 2008; revised November 13, 2008. Current version published March 18, 2009. The work of Y. Zhang was supported in part by National Science Foundation IIS-0713111 and in part by AFOSR. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Lexing Xie.

Y. Chi, S. Zhu, K. Hino, and Y. Gong are with NEC Laboratories America, Cupertino, CA 95014 USA (e-mail: ychi@sv.nec-labs.com; zsh@sv.nec-labs.com; hino@sv.nec-labs.com; ygong@sv.nec-labs.com).

Y. Zhang is with the School of Engineering, University of California Santa Cruz, Santa Cruz, CA 95064 USA (e-mail: yiz@soe.ucsc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2009.2012912

- **Content**: the media generated by people, e.g., blog and paper contents;
- **Time**: the timestamps for the generation of contents and relations.

Of course, one can argue that more dimensions could be used to describe networked data. For example, in many social networks, additional metadata such as tags (e.g., flickr.com) and trust scores (e.g., epinions.com) are available. However, on the one hand, we believe that these four dimensions have already captured most interesting information in networked data, and on the other hand, as will be shown later in this paper, our iOLAP framework can handle data with arbitrary number of dimensions and therefore can incorporate these additional metadata easily.

C. Polyadic Factor Model

With the four dimensions defined, we are able to put networked data into a multidimensional networked data cube and develop OLAP-type operations on the cube. However, some of the dimensions (such as content and relation) do not have counterparts in traditional relational databases, and therefore traditional OLAP techniques cannot be directly used to analyzing networked data cubes. To overcome this difficulty, as a core technique in our iOLAP framework, we develop a polyadic factorization method that decomposes a networked data cube into meaningful factors, where these factors together with their correlations are used as the basic units for the OLAP-type operations, such as roll-up and drill-down, in the iOLAP framework.

To analyze the networked data cube, an important and challenging issue is how to combine heterogeneous information from different dimensions. Various approaches have been proposed to tackle the challenge of analyzing multiple-dimension data. However, most existing research work adopts a two-step approach: first, only relationships between pairs of dimensions are analyzed; second, the obtained pairwise relationships are combined in certain ways. A main weak point of such a two-step approach is that it considers the relationships among different pairs of dimensions *independently* while in reality, different dimensions of polyadic data affect each other in a *joint* way.

There are some recent studies, such as [18], [24], and [25], that combine the two steps mentioned above into a single-step process. The key idea of these studies is to use *the same* set of concepts to simultaneously capture all the pairwise relationships among different dimensions. These approaches are more accurate in modeling polyadic data, however they have two major weaknesses. First, they usually use a linear combination to fuse all pairwise relationships among different dimensions. Such a linear combination is somewhat ad hoc, which makes it difficult to find a good intuition behind the coefficients as well as a principled way to set the values of the coefficients. Second, they ignore valuable information on the higher-order (higher than pairwise) correlation among various dimensions of data. In this study, we propose a probabilistic factorization model for analyzing polyadic data in a coherent way.

D. Efficient Implementation

A key for the successful application of our framework is that the implementation of the framework should be scalable enough to handle real data, where data in real Internet and Blogosphere applications usually have extremely large size in each of the four dimensions. For example, in a blog dataset, the size of the content dimension is the size of the vocabulary used in the dataset and the size of the people dimension is the number of bloggers in the dataset. Fortunately, very often data in real applications are very sparse. For example, the keywords that a blogger uses are very limited and the links generated by a blogger usually only point to those pointing to the blogger's close friends. Based on this observation, we develop an efficient implementation of our polyadic factorization algorithm that takes advantage of the sparseness of data. This implementation has a provable time complexity that is linear (per iteration) in the number of data records in a dataset.

E. Paper Organization

The rest of the paper is organized as the following. In Section II, we describe the core of our iOLAP framework—the polyadic factor models—in detail. In Section III, we provide some details on efficient computation of the model parameters. In Section IV, we demonstrate the applications of the iOLAP framework in analyzing the blogosphere and in personalized recommendation. In Section V, we survey related work. Finally we give conclusions in Section VI.

II. Core for the iOLAP Framework: Polyadic Factor Models

In this section, we describe the theoretical framework of the core technique in our iOLAP framework—the polyadic factor model. First, the factor model based on a probabilistic polyadic factorization is presented. Then an interpretation using a nonnegative tensor factorization is given. Finally, the connection between the two is proved. For ease of discussion we show the cases of 3-D data in the rest of the paper, although our factor model can be defined on data with an arbitrary number of dimensions.

A. Notations

First, we introduce some mathematical notations that will be used in later sections. We denote scalars by lowercase letters (e.g., a, b, α, β), vectors by lowercase letters in vector forms (e.g., $\vec{p}$, $\vec{q}$), matrices by capital letters (e.g., X, Y, Z), and tensors by calligraphic letters (e.g., $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$). We reserve i, j, k, i', j', k' to indicate the indices of tensors and matrices. We say a tensor $\mathcal{A}$ is obtained from another tensor $\mathcal{C}$ by a *base transform* if $\mathcal{A}_{i'j'k'} = \sum_{ijk} X_{i'i} Y_{j'j} Z_{k'k} \mathcal{C}_{ijk}$, which is denoted by

$$\mathcal{A} = [\mathcal{C}, X, Y, Z].$$

In such a transform, we refer to $\mathcal{C}$ as a *core tensor*. The same notation applies to matrices where $[S, U, V] = USV^T$. The *dot product* between two tensors is denoted by

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{ijk} \mathcal{A}_{ijk} \mathcal{B}_{ijk}.$$

Similar dot products are defined on matrices and vectors. For example, $\langle U, V\rangle = tr(U^T V)$, where tr is the trace operator. The KL-divergence between $\mathcal{A}$ and $\mathcal{B}$ is defined as

$$KL(\mathcal{A}||\mathcal{B}) = \sum_{ijk} \mathcal{A}_{ijk} \log \mathcal{A}_{ijk}/\mathcal{B}_{ijk} - \sum_{ijk} \mathcal{A}_{ijk} + \sum_{ijk} \mathcal{B}_{ijk}.$$

KL-divergence is defined on matrices and vectors in a similar fashion.

The *element-wise product* between two matrices X and Y is denoted by

$$Z = X \circ Y$$

with $Z_{ij} = X_{ij} Y_{ij}$. Similarly, the *element-wise division* is denoted by

$$Z = X/Y$$

with $Z_{ij} = X_{ij}/Y_{ij}$. We easily extend element-wise product and division operations to tensors.

As the partial derivative of $\langle \mathcal{A}, [\mathcal{C}, X, Y, Z]\rangle$ with respect to X is independent to X, we denote it by $\langle \mathcal{A}, [\mathcal{C}, \bullet, Y, Z]\rangle$. That is, with $T_{i'i} = \sum_{jk,j'k'} Y_{j'j} Z_{k'k} \mathcal{C}_{ijk} \mathcal{A}_{i'j'k'}$, we have

$$T = \langle \mathcal{A}, [\mathcal{C}, \bullet, Y, Z]\rangle \doteq \frac{\partial}{\partial X} \langle \mathcal{A}, [\mathcal{C}, X, Y, Z]\rangle .$$

Similarly, we use $\langle \mathcal{A}, [\mathcal{C}, X, \bullet, Z]\rangle$, $\langle \mathcal{A}, [\mathcal{C}, X, Y, \bullet]\rangle$, and $\langle \mathcal{A}, [\bullet, X, Y, Z]\rangle$ to indicate the partial derivatives of $\langle \mathcal{A}, [\mathcal{C}, X, Y, Z]\rangle$ with respect to Y, Z, and $\mathcal{C}$, respectively. Note that

$$\frac{\partial}{\partial \mathcal{C}} \langle \mathcal{A}, [\mathcal{C}, X, Y, Z]\rangle = [\mathcal{A}, X^T, Y^T, Z^T].$$

B. Probabilistic Polyadic Factor Model

We describe the probabilistic framework of our factor model by using a paper citation dataset, CiteSeer, as a running example where in the CiteSeer data, each data record is an author-keyword-reference triple. That is, each data record corresponds to an observation that an *author*, in an article on a specific *keyword*, cites a *reference*. Assume that the dataset contains I distinct authors, J distinct keywords, and K distinct references. Furthermore, assume that there are L latent factors (groups) of authors where each author belongs to these L groups with different probabilities. Note that these L latent factors are not explicit, i.e., they are hidden and need to be learned from training data. A random variable C' is used to represent the prior distribution for the L latent factors of authors. Similarly, it is assumed that there are M latent factors of keywords and N latent factors of references, and random variables C'' and C''' are used to represent the prior distributions of these two sets of latent factors. It is worth noting that the cardinalities of C', C'', and C''' do not have to be identical. In the following discussion, i, j, and k are used to represent the indices for author, keyword, and reference, respectively; l, m, and n are used to represent the indices for the factors of author, keyword, and reference, respectively.

With these latent factors defined, the following generative model is used to model how each data record is generated. To generate a data record (i.e., an author-keyword-reference triple $\langle i, j, k\rangle$), a factor c'_l for author, a factor c''_m for keyword, and a factor c'''_n for reference are selected with probability $P(c'_l, c''_m, c'''_n)$. Then depending on the selected factors, an author i is picked with probability $P(i|c'_l)$, a keyword j with probability $P(j|c''_m)$, and a reference k with probability $P(k|c'''_n)$. The assumption of this generative model is that the selections of author, keyword and reference are independent with each other, given the latent factors c'_l, c''_m, and c'''_n have been chosen. In the following discussion, to avoid notational clutter, we write $P(c'_l, c''_m, c'''_n)$ as $P(c_{lmn})$, $P(i|c'_l)$ as $P(i|l)$, $P(j|c''_m)$ as $P(j|m)$, and $P(k|c'''_n)$ as $P(k|n)$, respectively.

Furthermore, it is assumed that the data records are generated following an identical independent distribution. By using a_{ijk} to denoted the number of occurrences of triple $\langle i, j, k\rangle$ in the data, the likelihood of observing the data is

$$\prod_{i,j,k} \left[\sum_{l,m,n} P(c_{lmn}) P(i|l) P(j|m) P(k|n) \right]^{a_{ijk}}. \quad (1)$$

Then to complete the generative model, the parameters Θ are to be learned in order to maximize the above data likelihood, or equivalently, the logarithm of the data likelihood

$$\arg\max_{\Theta} \sum_{i,j,k} a_{ijk} \log \left[\sum_{l,m,n} P(c_{lmn}) P(i|l) P(j|m) P(k|n) \right] \quad (2)$$

where
$\Theta = \{P(c_{lmn}), P(i|l), P(j|m), P(k|n)\} \forall l, m, n, i, j, k.$

It can be shown that the following EM algorithm can be used to compute the MLE parameters Θ in our generative model (due to the space limit, we skip all proofs in this paper and refer interested readers to [4] for details):

$$E - Step:$$

$$\hat{p}_{lmn|ijk} = \frac{P(c_{lmn}) P(i|l) P(j|m) P(k|n)}{\sum_{l',m',n'} P(c_{l'm'n'}) P(i|l') P(j|m') P(k|n')} \quad (3)$$

$$M - Step:$$

$$P(c_{lmn}) = \sum_{i,j,k} a_{ijk} \hat{p}_{lmn|ijk} \Big/ \sum_{ijk} a_{ijk} \quad (4)$$

$$P(i|l) = \sum_{j,k,m,n} a_{ijk} \hat{p}_{lmn|ijk} \Big/ \sum_{i,j,k,m,n} a_{ijk} \hat{p}_{lmn|ijk} \quad (5)$$

$$P(j|m) = \sum_{i,k,l,n} a_{ijk} \hat{p}_{lmn|ijk} \Big/ \sum_{i,j,k,l,n} a_{ijk} \hat{p}_{lmn|ijk} \quad (6)$$

$$P(k|n) = \sum_{i,j,l,m} a_{ijk} \hat{p}_{lmn|ijk} \Big/ \sum_{i,j,k,l,m} a_{ijk} \hat{p}_{lmn|ijk}. \quad (7)$$

This probabilistic polyadic factor model is related to the aspect model proposed by Hofmann *et al.* [15]. However, it is

different from the aspect model in two ways. First, the aspect model is defined on dyadic data (matrices) while our factor model can handle polyadic data with arbitrary number of dimensions. Second, in the dyadic aspect model, there is a one-one mapping between the factors in the row space and the factors in the column space while in our model, such a one-one mapping is not enforced. Instead, in our factor model a joint probability, $P(c_{lmn})$, is used to capture the correlation among factors in different dimensions. As a result, our factor model is more flexible and it allows many-to-many mapping between factors in different dimensions.

C. Model Based on Nonnegative Tensor Factorization

Now we connect the probabilistic polyadic factor model in the previous section with a nonnegative version of the Tucker tensor factorization. For this purpose, we reinterpret the generative model in a different way. Assume that each author i is encoded with an L-dimensional vector $\vec{x}_i \in \mathcal{R}_+^L$, where $\mathcal{R}_+$ represents all nonnegative real numbers. Similarly, each keyword j is encoded by $\vec{y}_j \in \mathcal{R}_+^M$ and each reference k by $\vec{z}_k \in \mathcal{R}_+^N$. In addition, a_{ijk} is approximated by a function $f(\vec{x}_i \otimes \vec{y}_j \otimes \vec{z}_k)$ where f belongs to a certain family of functions $\mathcal{F}$ and $\otimes$ denotes the Kronecker product. The objective is to find the best encoding, in terms of $\vec{x}_i$, $\vec{y}_j$, $\vec{z}_k$, and f, that minimizes the following loss:

$$loss = \sum_{i,j,k} dist\left(a_{ijk}, f(\vec{x}_i \otimes \vec{y}_j \otimes \vec{z}_k)\right). \tag{8}$$

Now two questions arise: which family of functions $\mathcal{F}$ to choose and what the distance function to use. For the first question, we simply choose the family of linear functions, i.e., $f_{\vec{c}}(\vec{x}_i \otimes \vec{y}_j \otimes \vec{z}_k) = \langle \vec{c}, \vec{x}_i \otimes \vec{y}_j \otimes \vec{z}_k \rangle$, where $\langle \cdot, \cdot \rangle$ represents the vector inner product. For the second question, we choose the KL-divergence between a_{ijk} and $f(\vec{x}_i \otimes \vec{y}_j \otimes \vec{z}_k)$, where

$$KL(\vec{p}||\vec{q}) = \sum_i p_i \log \frac{p_i}{q_i} - \sum_i p_i + \sum_i q_i. \tag{9}$$

Then, we can rewrite the objective function (8) in terms of tensor factorization. Data are first put into a *data tensor* $\mathcal{A} \in \mathcal{R}_+^{I \times J \times K}$ where $(\mathcal{A})_{ijk} = a_{ijk}$. Next, the vectors $\vec{x}_i$'s are concatenated to a matrix $X \in \mathcal{R}_+^{I \times L}$ where the ith row of X is the transpose of $\vec{x}_i$. Similarly $\vec{y}_j$'s and $\vec{z}_k$'s are concatenated into matrices $Y \in \mathcal{R}_+^{J \times M}$ and $Z \in \mathcal{R}_+^{K \times N}$. Furthermore, $\vec{c}$ is *folded* into a tensor $\mathcal{C} \in \mathcal{R}_+^{L \times M \times N}$ with $(\mathcal{C})_{lmn} = \vec{c}_{(l-1)MN+(m-1)N+n}$, and we call $\mathcal{C}$ the *core tensor*. After the concatenation and the folding, we have $f_{\vec{c}}(\vec{x}_i \otimes \vec{y}_j \otimes \vec{z}_k) = [\mathcal{C}, X, Y, Z]_{ijk}$ where $\times_1$, $\times_2$, and $\times_3$ are the mode-1, mode-2, and mode-3 multiplications of the tensor $\mathcal{C}$ by the matrices X, Y, and Z, respectively. (Please refer to [5] for details about the terminologies.) Then the problem of best encoding becomes to find the best approximation, in terms of X, Y, Z, and $\mathcal{C}$, that minimizes the following loss:

$$loss_{KL} = KL\left(\mathcal{A}||[\mathcal{C}, X, Y, Z]\right). \tag{10}$$

In addition, because of the scale-free issue (e.g., $[\mathcal{C}, X, Y, Z] = [\alpha\mathcal{C}, (1/\alpha)X, Y, Z]$), we add additional constraints that each column of X, Y, and Z must sum to one.

Equation (10) turns out to be a nonnegative version of the Tucker tensor decomposition [23]. By directly extending the algorithm given by Lee *et al.* [16] for solving the nonnegative matrix factorization problem, we have the following iterative update algorithm.

Theorem 1: For a given $\mathcal{A} \in \mathcal{R}_+^{I \times J \times K}$, we first define another tensor $\mathcal{B}$ as

$$\mathcal{B} = \mathcal{A}/[\mathcal{C}, X, Y, Z] \tag{11}$$

Then the following update rules are guaranteed to converge to an optimal solutions to the objective function defined in (10):

$$\mathcal{C} \leftarrow \mathcal{C} \circ \langle \mathcal{B}, [\bullet, X, Y, Z] \rangle \tag{12}$$

$$X \leftarrow {}_1X \circ \langle \mathcal{B}, [\mathcal{C}, \bullet, Y, Z] \rangle \tag{13}$$

$$Y \leftarrow {}_1Y \circ \langle \mathcal{B}, [\mathcal{C}, X, \bullet, Z] \rangle \tag{14}$$

$$Z \leftarrow {}_1Z \circ \langle \mathcal{B}, [\mathcal{C}, X, Y, \bullet] \rangle \tag{15}$$

where $\leftarrow_1$ denotes the operation of after all updates are completed, normalizing so that all columns sum to ones.

D. Equivalence Between the Two Models

It turns out that the model we defined using the NTF framework is *equivalent* to the probabilistic polyadic factor model and therefore, it is just a different interpretation of the probabilistic factor model. We state this claim in the following theorem.

Theorem 2: The problem of minimizing the loss in (10) is equivalent to the MLE problem described by (2).

As can be expected, there is a close connection between the EM algorithm in (3)–(7) and the NTF factorization algorithm in (11)–(15). With some simple derivation we can show that the two algorithms share almost the identical form. However, there is a subtle difference between the two. In the EM algorithm, all parameters are updated *together* in the M-step. In comparison, in the NTF factorization algorithm, X, Y, Z, and $\mathcal{C}$ must be updated *alternatively*, which makes it a block descent algorithm.

III. Implementation Details and Time Complexity

In this section, we describe some practical details in the implementation of our algorithm and provide time complexity analysis.

A. Implementation Details

In the following discussion, without loss of generality, it is assumed that $L \geq M \geq N$ and $I \geq J \geq K$. We assume that there are n_z data records in the dataset, which implies that there are n_z nonzero entries in the data tensor $\mathcal{A}$. In addition, we assume that there are n_p distinct (i, j) pairs in the dataset. For ease of discussion, we use the NTF framework in the following discussion, although the same ideas apply to the EM algorithm.

It turns out the most components in (11)–(15), especially those time-consuming parts, can be computed in the same computational framework. We design an implementation that takes advantage of the sparseness of the dataset and has a time

complexity that is linear (per iteration) in n_z, the number of nonzero entries in the raw data tensor $\mathcal{A}$.

We describe this computation framework by showing how to compute $\mathcal{B}$ in (11), while other computations can be conducted in a similar fashion. Assuming the data are stored in the format of $\langle key1, key2, key3, v\rangle$, where $key1$, $key2$, and $key3$ are the indices of the data record in the first, second, and third dimensions, respectively; v is the value of the data record, which can be, for example, the number of times that an author cite a reference on a keyword. In the first step of the algorithm, the data records are sorted according to $key1$ as the major key and then $key2$ and then $key3$ as the minor keys. This sorting takes time linear in n_z because the keys are integers with known upper-bounds and therefore a linear sorting algorithm, such as bucket sort, can be applied. In addition, to simplify the discussion, it is assumed that $\langle key1, key2, key3\rangle$ is a primary (unique) key for the data records.

There are two key ideas in our implementation. First, our implementation is a *lazy* one where only the absolute necessary computations are performed. In such a lazy fashion, the implementation takes advantage of the sparseness of the data and only performs computations on entries corresponding to nonzero values in the dataset. In comparison, a naive implementation for computing $\mathcal{B}$ will expand $[\mathcal{C}, X, Y, Z]$, which turns out to be a dense tensor in $\mathcal{R}_+^{I\times J\times K}$. Because such a dense tensor contains IJK entries, it is impractical to make such an explicit expansion in many real applications (e.g., consider a dataset with 10 K authors, 10 K keywords, and 10 K references, where $IJK = 1$ trillion).

The second key idea of our implementation is that the computations are carefully ordered in a way such that repeated computations are minimized. Algorithm 1 illustrates these two key ideas using pseudo-code.

Algorithm 1 Algorithm for computing $\mathcal{B}$

input: data records $\langle key1, key2, key3, v\rangle$

output: $\mathcal{B}$

1) $i \leftarrow -1, j \leftarrow -1$;
2) **for each** data record $\langle key1, key2, key3, v\rangle$ **do**
3) **if** $i \neq key1$
4) $i \leftarrow key1, j \leftarrow -1$;
5) $D \leftarrow [\mathcal{C}, X_{row_i}, I_M, I_N]$;
6) **if** $j \neq key2$
7) $j \leftarrow key2$;
8) $\vec{d} \leftarrow Y_{row_j} \cdot D$;
9) $k \leftarrow key3$;
10) $b_{ijk} \leftarrow v/(Z_{row_k} \cdot \vec{d})$;
11) **return** $\mathcal{B}$;

In the algorithm, D is an $M \times N$ matrix and $\vec{d}$ is a vector of length N. In $[\mathcal{C}, X_{row_i}, I_M, I_N]$, I_M, and I_N are M-by-M and N-by-N identity matrices, respectively. $[\mathcal{C}, X_{row_i}, I_M, I_N]$ has size 1 in the first dimension and so can be written as a matrix. The idea of lazy computation is reflected by the fact that the *for* loop enumerates only nonzero entries in the dataset (or equivalently, in $\mathcal{A}$). To see the idea of ordered computation, we note that the D matrix computed at line 5 for a given i is usually reused several times at line 8 for different j's, and the $\vec{d}$ computed at line 8 is usually reused several times at line 10 for different k's.

B. Time Complexity

It can be easily seen that the algorithm in Algorithm 1 has a time complexity of $O(n_z \cdot L + n_p \cdot L^2 + I \cdot L^3)$, where n_z is the number of data records and n_p is the number of distinct (i, j) pairs in the dataset. (Recall that we assume $L \geq M \geq N$ and so L^2 is an upper bound for LM and L^3 is an upper bound for LMN.) Here are some observations on the time complexity.

- If we consider L, the number of factors, as a constant, then the time is linear in n_z. This characteristic is good because datasets in real applications often have high cardinality in each dimension but are sparse overall and therefore, an algorithm linear in n_z is practically appealing.
- If we do not consider L as a constant, then some terms in the time complexity are proportional to L^3, L^2, and L, respectively. However, we have $n_z > n_p > I$. When the difference between n_z, n_p, and I is large, which is very common in real applications where data distribution is skewed, the L^2 term or even the L term dominates the running time instead of the L^3 term.
- An additional good property of our algorithm is that since the data records are accessed in a sequential way, the algorithm can be easily parallelized.

In addition, our algorithm is an iterative updating one and therefore the loss in (10) is needed in each iteration. Again, to explicitly expand $[\mathcal{C}, X, Y, Z]$ is impractical. However, it turns out that the loss in KL-divergence can be easily computed while computing

$$\mathcal{B} = \mathcal{A}/[\mathcal{C}, X, Y, Z]$$

at no extra cost. As a result, computing the losses does not change the total time complexity of the algorithms.

IV. Applications in the Blogosphere Analysis and Personalized Recommendation

In this section, we apply the iOLAP framework to two applications, to demonstrate the functionalities that the iOLAP framework can provide for analyzing networked data. First, we study a social network dataset obtained by an in-house blog crawler. We use this blog dataset to demonstrate certain operations, such as roll-up and drill-down, offered by the iOLAP framework. Second, we study a paper citation dataset, to show that the iOLAP framework provides a nice solution to the application of personalized recommendation.

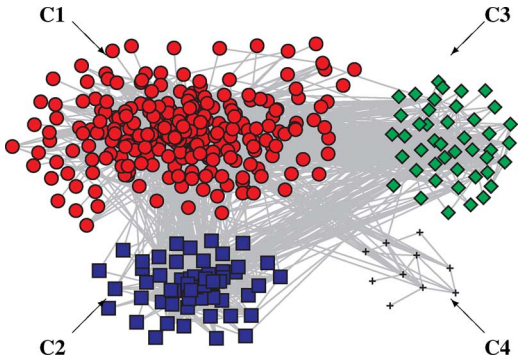


Fig. 1. Four major communities in the NEC dataset.

A. Analyzing the Blogosphere

1) Problem and Dataset Description: At NEC Laboratories America, we have built a focused crawler on blogs. Given a manually preselected set of technology-related seed blogs, the crawler followed the embedded hyperlinks in these seed blogs to discover additional blogs that are densely connected with the original seed blogs. More specifically, we only added blogs containing at least certain number of hyperlinks pointing to and pointed by the seed blogs. This resulted in an expanded set of blogs that densely connected with each other. The crawler then monitored new entries generated in this expanded set of blogs over a long time period. The final dataset contains 407 English blogs with 274 679 entries in 441 days (63 weeks) between July 10, 2005 and September 23, 2006. These entries are connected with 148 681 links.

We start with analyzing the overall picture of the dataset. By manually investigating the contents of the posts, we found that the blogs in our dataset fall into four main communities. We draw the aggregated graph in Fig. 1, according to the main community each blog most likely belongs to. In the figure, nodes of the same shape belong to the same community and their relative distance is determined by Kamada–Kamai layout algorithm in the Pajek package [6]. The links in the figure represent the entry-to-entry links in the dataset. In addition, in Table I we list the top keywords, measured by the *tf-idf* score, that occur in the posts of these four communities. We find that community $C1$ focuses on technology, $C2$ on politics, $C3$ on international entertainment, and $C4$ on digital libraries.

The tasks we defined on this dataset are exploratory. That is, by using the iOLAP framework we want to answer certain exploratory queries from the user such as "What are the main communities in the blog dataset?", "What are their main topics during a given period of time?", "How do things change over time?", and "What are people with different backgrounds saying about a given product, e.g., 'iPod'?".

2) Experimental Results: From the blog dataset, we build a networked data cube where each entry is a sid-did-keyword-time quadrable, which corresponds to "A source blog with id *sid* links to a destination blog with id *did* on a *keyword* at a given *time*." In this data cube, the relation corresponds to the notion of "citation" and consists of pairs of source and destination blogs. Of course, in many applications there exist other types of pairwise relations (e.g., friendship, trust) and relations that are not necessarily pairwise (e.g., a set of authors that coauthored a paper).

TABLE I
TOP KEYWORDS AMONG THE FOUR MAJOR COMMUNITIES IN THE NEC DATASET, SORTED BY THE *tf-idf* SCORE

C1	adsense, beta, skype, firefox, msn, rss, aol, yahoo, google, ebay, desktop, wordpress, voip, feeds, myspace, podcasting, technorati, search, engine, browser, ads, gmail, windows, os, developer, venture, marketing, apple, podcasts, developers, engines, mac, publishers, linux, itunes, feed, podcast
C2	gop, uranium, hezbollah, democrats, rove, cia, republicans, saddam, qaeda, tax, republican, iraqi, roberts, bush, clinton, iraq, senate, troops, terrorists, administration, terrorist, wilson, conservative, taxes, liberal, intelligence, israel, terror, iran, weapons, war, soldiers, congress, americans
C3	shanghai, robots, installation, japan, japanese, architecture, art, chinese, china, saudi, phones, filed, mobile, games, korea, rfid, sex, green, camera, sound, cell, body, africa, phone, entertainment, film, gay, india, fuel, archive, design, elections, flash, device, water, wireless, virtual, image
C4	library, learning, digital, resources, collection, conference, staff, communities, students, session, books, database, access, survey, university, science, canada, myspace, articles, education, technologies, knowledge, filed, virtual, tools, research, learn, services, computers, professional, journal

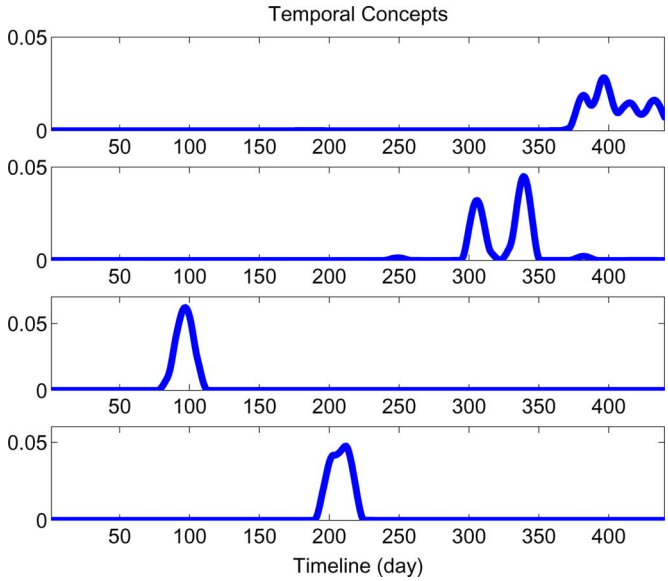


Fig. 2. Some factors in the time dimension.

We first apply the polyadic factor model on the 4-D data cube, with 20 factors in each dimension. Some sample factors in the time dimension and in the relation dimension are shown in Figs. 2 and 3, respectively. As can be seen from the figures, the factors in the time dimension show some interesting patterns that may correspond to certain events while the factors in the relation dimension form some coherent communities. Note that in Fig. 2, for the ease of visualization, we have applied a moving average with a window size of seven days; in Fig. 3 we have kept the color, shape, and position of the nodes as they are in Fig. 1, in order to easily visualize the major community to which a blog belongs.

Next, we show some iOLAP operations that can be applied to the factors and their correlations which have been computed from the polyadic factorization. In the first example, the query is "What are the main groups of bloggers and what are the main topics in each group?". To answer this query, we *roll-up* the core

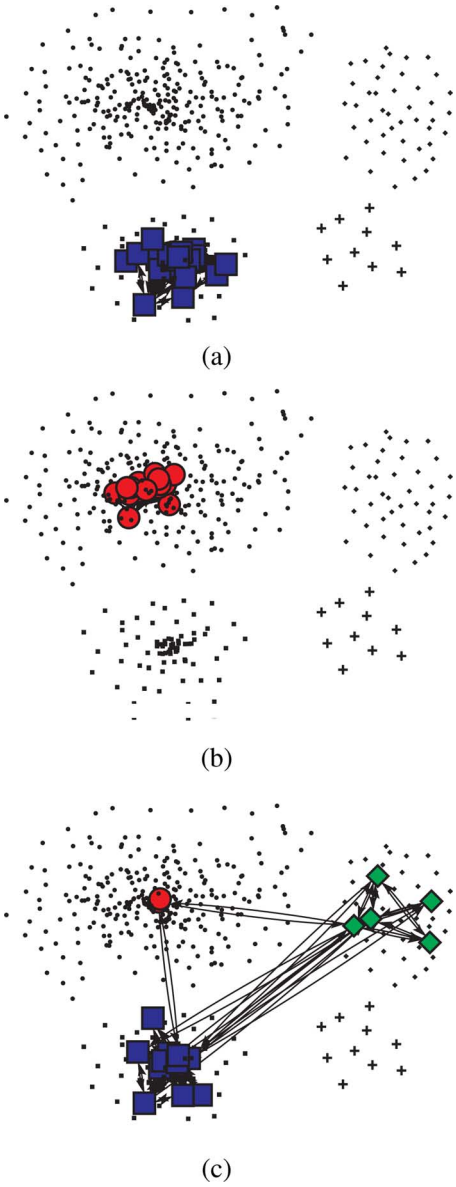


Fig. 3. Some factors in the relation dimension.

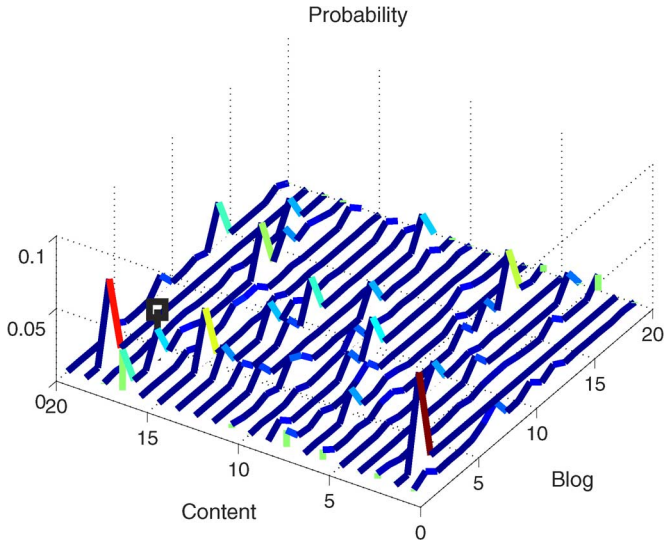


Fig. 4. Core tensor, rolled-up on the time and the relation dimensions.

tensor by aggregating the relation dimension into blogs and aggregating out the time dimension. Fig. 4 shows the result. From the figure we can clearly see distinct blog groups and content groups (topics). By using the factors in the corresponding dimensions, we can easily recover the top bloggers and top keywords in the each group. An interesting observation from Fig. 4 is that multiple blog groups can share similar content factors (topics) while a single blog group may contain several different topics. For example, content factors 6 and 17 both correspond to political issues, but with slightly different focuses. The top keywords for factor 6 are *washington, bush, election, state, house, american, left, katrina, conservative*, etc., and those for factor 17 are *washington, terrorists, military, american, attacks, officials, war, bush, nation*, etc. As can be seen, factor 17 focuses more on the issue of terrorism. By looking at the blog factors who have high values jointly with these two content factors, we found that the two content factors share very similar blog factors, except that blog factor 4 focuses mainly on content factor 17 but not on content factor 6 (as indicated by a black square in Fig. 4). It turns out that blog factor 4 consists of a small group of blogs whose top member is the blog of Michelle Malkin (http://michellemalkin.com/), who is a political blogger well-known for her insights on the issue of terrorism.

In the second example, with the assistance of the factors and their correlation provided by the iOLAP framework, we answer an exploratory query "What are people with different backgrounds saying about 'iPod'?". To answer this query, we first select all the sentences that contain the word "iPod", and the *drill-down* the result according to the factors in the relation dimension (by projecting the blogger of each sentence into the relation factors) and the factors in the content dimension (by projecting the content of each sentence into the content factors). Such a drill-down operation naturally categorizes the sentences related to "iPod" into separated groups. Table II shows the top sentences in several resulting groups (where to facilitate reader understanding, we named each group by a title and listed the top keywords occurred within each group). As can be seen, sentences in different groups clearly focus on different aspects of the same query word "iPod". Also, by comparing the top keywords with those in Table I, we can see that each of these groups clearly belongs to one of the manually detected communities $C1 - C4$.

These functionalities provided by the iOLAP framework, as demonstrated by the above examples, are very valuable for the purpose of business intelligence, better decision making, and market strategies.

B. Personalized Recommendation

1) Problem and Dataset Description: The dataset we used in this application is obtained from the CiteSeer website.[1] A set of articles from various areas are selected first. The author(s), abstract, and references of each article is extracted and the abstracts are split into keywords. Finally the *author-keyword-reference* triples are extracted, which result in a dataset containing

[1]http://citeseer.ist.psu.edu/

TABLE II
SOME TOP SENTENCES RESULTS ON THE QUERY "IPOD", DRILLED-DOWN ON THE RELATION AND THE CONTENT DIMENSIONS

international affairs	
1.	Full-face stickers that let you do the full Hello Kitty to your **iPod** nano.
2.	South Korea bravely resists the evil imperialist forces... of **iPod**.
3.	The **iPod** is also the top seller in Canada, Australia, the United Kingdom, France, Spain, Italy and Japan (50 percent).
4.	**iPod** fails to dent South Korea.
5.	Sakito has been credited with leading a successful marketing push for the **iPod** in Japan.
Top keywords: japan, chinese, japanese, china, north, asia, english, south, east, countries, french, europe	

usage and user interfaces	
1.	Install vPod in the same folder you installed CDex, again to your **iPod**.
2.	**iPod** users can now use iTunes to automatically sync calendars with Outlook or contacts with Outlook or Outlook Express.
3.	Customize the menu instead of burrowing through the **iPod**'s default menus, put your favorite menu items on top.
4.	It searches your **iPod** and then makes it easy for you.
5.	Unzip the files and place the folder (s) in the notes folder on your **iPod**.
Top keywords: developer, beta, browser, jeremy, wordpress, publishers, feed, rss, searches, firefox, desktop	

politics	
1.	So be it, until there is no enemy, but **iPods**.
2.	IPaction is doing some lobbying of their own, promising to send everyone in the senate an **iPod**.
3.	I'm not sure I want "Suicide Bombers" displayed on my **iPod**.
4.	We've reported that President Bush is an **iPod** lover, but has there been any proof of this?
5.	So they're buying a video **iPod** for the campaigns of senators who work on legislation affecting technology.
Top keywords: muslim, qaeda, defense, terrorists, military, liberal, republican, senate, iraqi, troops, attacks	

16 466 authors, 920 keywords, 29 309 references, and 3 636 020 data records. On this CiteSeer dataset, we define the task as follows: for a given author and on a given keyword, recommend relevant references. This task is useful for an author to collect important references when writing an article on a particular topic.

A straightforward solution to this problem is to simply count for each reference the number of articles on the given keyword that cite the reference. Such an approach, however, makes global recommendations to all the authors and therefore ignores any personal information about the author to whom the recommendation is made. Personal information can be very useful for recommendations. For example, different authors have different areas of expertise and research focuses. So for a given keyword, the most popular reference on the keyword may not be the most relevant reference for a particular author to write a paper on that keyword.

Our factor model provides a natural way to incorporate an author's background while making personalized recommendation. We can compute the following conditional probability for all the reference k's to be selected given the keyword is i and the author is j:

$$P(k|ij) = \frac{P(ijk)}{P(ij)} \sim P(ijk) = \left[\mathcal{C}, X_{row_i}, Y_{row_j}, Z_{row_k}\right] \tag{16}$$

and then recommend the references with the highest conditional probabilities.

2) Experimental Results: For the performance studies, data are randomly split into training data and testing data following an 80-20 ratio. Factorizations are applied to the training data to get the parameters for our factor models, i.e., the factors in each dimensions and their correlation. Then the learned factor models are applied to the testing data for top-K personalized recommendation with various K's. The recommendation performance is measured by the average normalized discounted cumulative gain (NDCG), where NDCG is defined as

$$NDCG = \sum_j \frac{2^{rel(j)} - 1}{\log(1+j)}. \tag{17}$$

NDCG is a relatively new measure used extensively in Web search. It allows different levels of relevance and weighs the recommendations according to their ranks in the ranked list. In our study, we use a binary relevance score for $rel(j)$ where $rel(j) = 1$ if item j occurs in the testing data and 0 otherwise. More specifically, we use top-3 tag recommendation as an example to illustrate how the average NDCG is computed. Assume that for a user i on a url k, our model recommends an ordered list of three tags $\{j_1, j_2, j_3\}$. Furthermore, assume that $\langle i, j_1, k\rangle$ and $\langle i, j_3, k\rangle$ show up in the testing data but not $\langle i, j_2, k\rangle$. Then the NDCG for the (i, k) pair is $1/\log(1+1) + 1/\log(3+1) = 1.5$. The average NDCG is the NDCG scores average over all distinct (i, k) pairs in the testing data.

We compare the performance of our factor model with that of two baseline algorithms. These baseline algorithms offer *global* rather than *personalized* recommendations. We explain these baseline algorithms using the task of reference recommendation. First, the *author-keyword-reference* triples are aggregated into *keyword-reference* pairs together with the count of each pair. The aggregated results are put into a matrix M where the rows and columns of M correspond to keywords and references, respectively, and the entry M_{jk} of M represents the count of occurrences of keyword j associated with reference k. For the first baseline algorithm, the recommendation is only based on the count of occurrences. That is, for keyword j, this baseline recommends the references with highest magnitudes in the jth row of M. This baseline algorithm corresponds to recommending by popularity and such a straightforward method usually shows very good performance when enough observations are sampled from data. In the second baseline algorithm, a nonnegative matrix factorization (NMF) is applied to M to compute an approximation $\tilde{M}$ to M. After $\tilde{M}$ is obtained, the recommendations are made in a similar way to the first baseline, except that $\tilde{M}$ is used instead of M. The intuition behind this baseline is similar to the intuition behind the latent semantic indexing (LSI [7])—by capturing the major components from the raw data, we may be able to reduce the noise in the raw data and therefore discover the hidden relations. Note that both baseline algorithms make recommendations *globally* while our factor model offers *personalized* recommendations.

For this task of recommending relevant references to a given author on a given keyword, Table III shows the performance. For our factor model, ten factors are used for each of the three dimensions. As can be seen from the results, in this task our

TABLE III
PERFORMANCE OF DIFFERENT ALGORITHMS FOR THE TASK OF RECOMMENDING REFERENCES ON CITESEER DATA

	top-1	top-3	top-5	top-10
baseline1	0.025	0.045	0.057	0.076
baseline2	0.021	0.037	0.047	0.062
Factor_KL_10	**0.038**	**0.070**	**0.090**	**0.120**

TABLE IV
TOP MEMBERS IN SOME AUTHOR FACTORS

DB & DM	Networks	AI & ML
T Eiter	J L Boudec	S Thrun
G Karypis	D Estrin	W Burgard
G D Giacomo	Z Zhang	H Zhang
W Faber	J A Stankovic	J Malik
N Leone	V Firoiu	M J Black
E Keogh	K G Shin	J A Fessler
J Yang	G B Giannakis	M J Mataric
J Han	J Liebeherr	C Sanderson
V J Tsotras	T Abdelzaher	R Deriche
D Calvanese	C Lu	D Fox
D Zhang	D Towsley	S Belongie
W Wang	I Stojmenovic	J Huang
M Hanus	T He	S Z Li
H Garcia-Molina	G Manimaran	I Cohen
S Mehrotra	E W Knightly	G Sapiro
D Gunopulos	M Reisslein	S Soatto
C Zaniolo	J Heidemann	A M Tekalp
K Chakrabarti	C Li	A S Willsky
L Bertossi	K Lam	M Gales
E A Rundensteiner	A Helmy	T Poggio

factor model is able to outperform the baseline algorithms using very small number of factors.

This good performance suggests that in this CiteSeer dataset, personalization tells us a lot about an author. To validate this point, we show in Tables IV–VI the top members in some factors obtained by our framework. We have chosen some representative factors that correspond to the research areas that we are more familiar with and for each factor, we show the top-ranked members (i.e., those members whose entries have the largest values in the corresponding columns of X, Y, and Z). As can be seen, the factors obviously form clusters of authors focusing on different research areas with different relevant references. It is worth mentioning that these factors for the three dimensions are *simultaneously* obtained by our factor models from the polyadic raw data. That is, our factor models are effective in simultaneously analyzing all dimensions as well as their correlations among polyadic data.

V. RELATED WORK

As we have mentioned in the introduction, traditional OLAP techniques were developed during the 1980s and 1990s to provide analysis, summarization, and visualization tools on structured data [10], [11]. One feature that distinguishes traditional OLAP techniques from our iOLAP framework is that in traditional OLAP data are usually numerical values, which makes data statistics such as min, max, average, etc., well-defined for the data cube and its subsets. In comparison, in the iOLAP framework, we provide novel techniques to analyze data types (e.g., relations and contents) that have no counterparts in traditional databases. Another unique feature of our iOLAP framework is that instead of handling databases with clearly defined relations (e.g., through the foreign keys among tables in a database), the iOLAP framework identifies important dimensions in unorganized data, discovers meaningful factors in each of the dimensions, and provides OLAP-type functionalities to analyze the correlations among the factors in all the dimensions.

TABLE V
TOP MEMBERS IN SOME KEYWORD FACTORS

DB & DM	Networks	AI & ML
database	network	recognition
documents	scheduling	segmentation
information	protocol	bayesian
document	differentiated	wavelet
language	mobility	feature
semantic	location	reconstruction
approach	packets	localization
semi-structured	capacity	approach
disjunctive	channels	statistical
declarative	communication	learning
extraction	schedulers	matching
programs	differentiation	correspondences
representation	multiplexing	estimate
dimensionality	queueing	registration
datasets	support	classification

TABLE VI
TOP MEMBERS IN SOME REFERENCE FACTORS

1. Fast algorithms for mining association rules
2. Mining association rules between sets of items in large databases
3. The anatomy of a large-scale hypertextual Web search engine
4. Authoritative sources in a hyperlinked environment
5. The stable model semantics for logic programming
6. Information retrieval
1. Dynamic source routing in ad hoc wireless networks
2. A performance comparison of multi-hop wireless ad hoc network routing protocols
3. Ad-hoc on-demand distance vector routing
4. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers
5. Next century challenges: scalable coordination in sensor networks
6. A highly adaptive distributed routing algorithm for mobile wireless networks
1. Reinforcement learning I: introduction
2. Bagging predictors
3. Support-vector networks
4. Optimization by simulated annealing
5. A decision-theoretic generalization of on-line learning and an application to boosting
6. Experiments with a new boosting algorithm

Another research area that is closely related to our work is general tensor factorizations where the nonnegative constraint is removed. The Tucker model [23] (which has recently been generalized to the higher order singular value decomposition [5]) and the PARAFAC model [12] are two such tensor decompositions. These tensor factorization techniques show some good properties in terms of minimizing certain losses in the Frobenius norm. However, the factors and core tensors obtained by these general factorizations have negative values. In addition, factors obtained from these tensor factorizations usually have some orthogonality constraints. All of these restrict the applications of these tensor factorizations.

As we have mentioned before our factor model for polyadic data is closely related to the aspect model [15] and the probabilistic semantic analysis (PLSA) [14] proposed by Hofmann *et*

al. and the nonnegative matrix factorization (NMF) algorithms proposed by Lee *et al.* [16]. The MLE interpretation of our factor model is an extension to PLSA and the NTF interpretation is an extension to NMF. Recently, some work, such as [8] and [9], has shown the equivalence between the PLSA and the NMF algorithms. Therefore it comes at no surprise that the MLE and the NTF interpretations of our factor model are equivalent. Also, we are not the first ones to notice that the NMF algorithms developed by Lee *et al.* [16] can be directly extended to solving NTF problems. For example, Mørup *et al.* [19] have explored this connection and developed similar algorithms. In addition, there also exist nonnegative versions of the PARAFAC model such as [13] and [21]. However, a weak point of such algorithms is that they require the number of factors in each dimension to be identical and the core tensor to be diagonal. This limits the usage of the model in many applications. In [3], Banerjee *et al.* proposed a multi-way clustering framework that can handle multidimensional relations. However, the approach by Banerjee *et al.* is derived from the Bregman divergence and it uses an optimization framework. In comparison, our polyadic factor model is derived from the probabilistic generative model and therefore provides an underlying probabilistic interpretation to the results.

In addition, all the implementations in the above papers can only handle small-scale dense tensors, such as images or time series with sizes of hundreds or thousands in each dimension. Although there exist some attempts to take advantage of data sparseness, such as the Matlab tensor toolbox by Badar *et al.* [2], to the best of our knowledge, our implementation is the first one that is capable of conducting decompositions on large-scale sparse tensors.

Papers analyzing communities and their evolutions in social networks are related to our work. Recently, there exists a growing body of literature on analyzing communities and their evolutions in dynamic social networks. Palla *et al.* [20] analyzed a coauthorship network and a mobile phone network, where both networks are dynamic. Toyoda *et al.* [22] studied the evolution of Web communities from a series of Web archives. Asur *et al.* [1] introduced a family of events on both communities and individuals to characterize evolution of communities. Lin *et al.* [17] proposed a generative model for capturing community evolutions. Our framework only captures the change of *intensity* of communities over time and does not support analysis on the change of community structures. To extend our framework to capturing the evolutions of communities is among our future directions.

VI. Conclusion and Future Directions

In this paper, we present a novel framework, iOLAP, for analyzing social media. In the proposed framework, four important dimensions—people, relation, content, and time—are used to capture the information contained in networked data. The core technique of the iOLAP framework, the polyadic factor model, is used to partition networked data cubes into meaningful factors. The key advantage of the proposed polyadic factor model is that it directly analyzes all the dimensions of a networked data cube simultaneously in a unified framework. As a result, the obtained factors are more accurate and the correlation among the factors in different dimensions is explicitly derived. Both the factors and their relation are used in the iOLAP analysis of the networked data cube. It is worth noting that our iOLAP framework is general enough to handle any type networked data, such as blogs, scientific papers, social bookmarking data, and e-mails, because it can handle data with arbitrary number of dimensions. Experimental results on a social network dataset and a paper citation dataset demonstrated some valuable functionalities for multidimensional data analysis that are offered by the iOLAP framework.

For future studies, we plan to work on the following directions. First, we plan to explore how to extend our framework to efficiently support more OLAP-type functionalities, such as summarizing query results. Second, we plan to conduct more comprehensive studies to compare the performance of our framework with some other approaches (e.g., those in [3] and [25]) under the same number of parameters.

Acknowledgment

The authors would like to thank Professor C. L. Giles for providing us the CiteSeer dataset.

References

[1] S. Asur, S. Parthasarathy, and D. Ucar, "An event-based framework for characterizing the evolutionary behavior of interaction graphs," in *Proc. 13th ACM SIGKDD Conf.*, 2007.

[2] B. W. Bader and T. G. Kolda, "Algorithm 862: Matlab tensor classes for fast algorithm prototyping," *ACM Trans. Math. Softw.*, vol. 32, no. 4, pp. 635–653, 2006.

[3] A. Banerjee, S. Basu, and S. Merugu, "Multi-way clustering on relation graphs," in *Proc. 7th SIAM Int. Conf. Data Mining*, 2007.

[4] Y. Chi, S. Zhu, Y. Gong, and Y. Zhang, "Probabilistic polyadic factorization and its application to personalized recommendation," in *Proc. 17th ACM CIKM Conf.*, 2008.

[5] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 21, no. 4, pp. 1253–1278, 2000.

[6] W. de Nooy, A. Mrvar, and V. Batagelj, *Exploratory Social Network Analysis With Pajek*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[7] S. Deerwester, S. Dumais, T. Landauer, G. Furnas, and R. Harshman, "Indexing by latent semantic analysis," *J. Amer. Soc. Inf. Sci.*, vol. 41, pp. 391–407, 1990.

[8] C. Ding, X. He, and H. D. Simon, "On the equivalence of nonnegative matrix factorization and spectral clustering," in *Proc. SIAM SDM*, 2005.

[9] E. Gaussier and C. Goutte, "Relation between PLSA and NMF and implications," in *Proc. SIGIR*, 2005.

[10] J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh, "Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals," *J. Data Mining Knowl. Discov.*, vol. 1, no. 1, pp. 29–53, 1997.

[11] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed. San Francisco, CA: Morgan Kaufmann, 2005.

[12] R. A. Harshman, "Foundations of the parafac procedure: Models and conditions for an "explanatory" multi-modal factor analysis," *UCLA Working Papers in Phonetics*, vol. 16, 1970.

[13] T. Hazan, S. Polak, and A. Shashua, "Sparse image coding using a 3d non-negative tensor factorization," in *Proc. 10th IEEE Int. Conf. Computer Vision (ICCV '05)*, 2005.

[14] T. Hofmann, "Unsupervised learning by probabilistic latent semantic analysis," *Mach. Learn.*, vol. 42, no. 1–2, pp. 177–196, 2001.

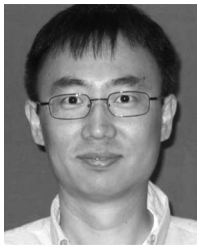
[15] T. Hofmann and J. Puzicha, Unsupervised Learning From Dyadic Data, Int. Comput. Sci. Inst., Berkeley, CA, Tech. Rep. TR-98-042, 1998.

[16] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Proc. NIPS*, 2000.

[17] Y. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. L. Tseng, "FacetNet: A framework for analyzing communities and their evolutions in dynamic networks," in *Proc. 17th WWW Conf.*, 2008.

[18] Q. Mei, D. Cai, D. Zhang, and C. Zhai, "Topic modeling with network regularization," in *Proc. 17th WWW Conf.*, 2008.

[19] M. Mørup, L. K. Hansen, and S. M. Arnfred, Algorithms for Sparse Non-Negative TUCKER (also named HONMF), Dept. Informat. Math. Model., Tech. Univ. Denmark, 2007, Tech. Rep.
[20] G. Palla, A.-L. Barabasi, and T. Vicsek, "Quantifying social group evolution," *Nature*, vol. 446, pp. 664–667, 2007.
[21] A. Shashua and T. Hazan, "Non-negative tensor factorization with applications to statistics and computer vision," in *Proc. 22nd Int. Conf. Machine Learning (ICML '05)*, 2005.
[22] M. Toyoda and M. Kitsuregawa, "Extracting evolution of web communities from a series of web archives," in *Proc. 14th ACM Conf. Hypertext and Hypermedia (HYPERTEXT '03)*, 2003.
[23] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, pp. 279–311, 1966.
[24] D. Zhou, S. Zhu, K. Yu, X. Song, B. L. Tseng, H. Zha, and C. L. Giles, "Learning multiple graphs for document recommendations," in *Proc. WWW*, 2008.
[25] S. Zhu, K. Yu, Y. Chi, and Y. Gong, "Combining content and link for classification using matrix factorization," in *Proc. SIGIR*, 2007.

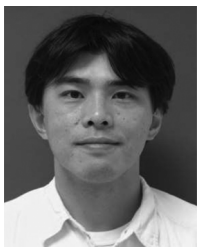


Yun Chi received the Ph.D. degree in computer science from the University of California, Los Angeles, in 2005.

He is currently a research staff member with NEC Laboratories America, Cupertino, CA. His primary research interests include data mining, machine learning, and information retrieval.

Shenghuo Zhu received the Ph.D. degree in computer science from the University of Rochester, Rochester, NY, in 2003.

He is currently a research staff member with NEC Laboratories America, Cupertino, CA. His primary research interests include information retrieval, machine learning, and data mining.



Koji Hino received the M.Eng. degree from Kyushu University, Japan in 1990.

He is currently a research staff member with NEC Laboratories America, Cupertino, CA. Before joining NEC Laboratories America, he worked for NEC Japan as a research staff member and assistant manager in Internet engineering research. His primary research interests include distributed network control application and social network analysis.

Mr. Hino is a member of IEICE and IPSJ.

Yihong Gong (M'92) received the B.S., M.S., and Ph.D. degrees in electronic engineering from the University of Tokyo, Tokyo, Japan, in 1987, 1989, and 1992, respectively.

He then joined the Nanyang Technological University (NTU) of Singapore, where he worked as an Assistant Professor in the School of Electrical and Electronic Engineering. Between June 1996 and December 1998, he worked for the Robotics Institute, Carnegie Mellon University as a project scientist, and was a principal member of both the Informedia Digital Video Library project and the Experience-On-Demand project funded by NSF, DARPA, and other governmental agencies. In 1999, he joined NEC Laboratories America, Cupertino, CA, where he built the multimedia content analysis team from scratch. In 2006, he became the head of the silicon valley branch of the labs. His research interests include computer vision, multimedia content analysis, and machine learning.

Yi Zhang received the Ph.D. degree in language technologies from the School of Computer Science at Carnegie Mellon University, Pittsburgh, PA.

She is an Assistant Professor at Baskin School of Engineering, University of California Santa Cruz. Her research interests include information retrieval, text mining, statistical machine learning, and natural language processing.