Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2015) 1–8

Procedia Computer Science

Scalable Tensor Mining

Lee Sael[a], Inah Jeon[b], U Kang[b,*]

[a]*Department of Computer Science, State University of New York Korea, Republic of Korea*
[b]*Department of Computer Science, KAIST, Republic of Korea*

Abstract

Tensors, or multi dimensional arrays, are receiving significant attentions due to the various types of data that can be modeled by them; examples include call graphs (sender, receiver, time), knowledge bases (subject, verb, object), 3-dimensional web graphs augmented with anchor texts, to name a few. Scalable tensor mining aims to extract important patterns and anomalies from a large amount of tensor data. In this paper, we provide an overview of scalable tensor mining. We first present main algorithms for tensor mining, and their scalable versions. Next, we describe success stories of using tensors for interesting data mining problems including higher order web analysis, knowledge base mining, network traffic analysis, citation analysis, and sensor data analysis. Finally, we discuss interesting future research directions for scalable tensor mining.

Keywords: Tensor, Distributed Computing, Big Data

1. Introduction

Given a call graph containing who-calls-whom information, how can we find patterns and anomalies? How can we find groups of closely related (subject, object, verb) triples in a multi-dimensional knowledge base? How can we find anomalous packets from network traffic data? Tensors, or multi-dimensional arrays, appear in numerous applications: network traffic [1], knowledge bases [2, 3, 4, 5], hyperlinks and anchor texts in the Web graphs [6], sensor streams (time, location, and type) [7], and DBLP conference-author-keyword relations [8], to name a few. Analysis of large tensors using scalable tensor mining algorithms is a basis for many interesting applications including clustering, trend detection, anomaly detection [8], correlation analysis [7], network forensic [9], and latent concept discovery [6].

In this paper, we give an overview of scalable mining. We first describe algorithms for tensor analysis, and their scalable versions using distributed platforms, in Section 2. Next, we introduce success stories of using scalable tensor mining for various applications including higher order web link analysis, knowledge base analysis, network traffic mining, citation (DBLP) analysis, and sensor data analysis, in Section 3. The area of scalable tensor mining research is fastly growing, and thus there are many research opportunities waiting for us. We discuss several research directions of scalable tensor mining in Section 4, and then conclude at Section 5.

*Corresponding author.
Email addresses: sael@sunykorea.ac.kr (Lee Sael), inahjeon@kaist.ac.kr (Inah Jeon), ukang@cs.kaist.ac.kr (U Kang)

2. Algorithms

In this section, we give an overview of tensor mining algorithms and their scalable versions.

2.1. Preliminaries

Vectors are denoted by boldface lowercases (e.g. $\mathbf{a}$), matrices are denoted by boldface capitals (e.g. $\mathbf{A}$), and the rth row of the matrix $\mathbf{A}$ is denoted by $\mathbf{a}_r$. Tensor is defined as a multi-dimensional array. A vector and a matrix are special cases of tensor with 1 and 2 dimensions, respectively. Each dimension of a tensor is called a *mode*, and an N-mode tensor is denoted by $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. A mode-$n$ *fiber* of a tensor is a vector determined by fixing all indexes other than n of the tensor. Matricization is a process of transforming a tensor into a matrix; a mode-n matricization of a tensor $\mathcal{X}$ is denoted by $\mathbf{X}_{(n)}$, and contains the mode-n fibers as columns. An n-mode matrix product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with a matrix $\mathbf{A}$ is denoted by $\mathcal{X} \times_n \mathbf{A}$ and is of size $I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$. We refer the reader to [10] for further backgrounds on tensor.

2.2. Tensor Decomposition

Tensor decomposition aims to decompose a tensor into latent factors. We describe two main tensor decomposition methods: PARAFAC and Tucker.

2.2.1. PARAFAC Decomposition

PARAFAC (parallel factors) decomposition [11], or CANDECOMP (canonical decomposition), decomposes a tensor into a sum of rank-one tensors.

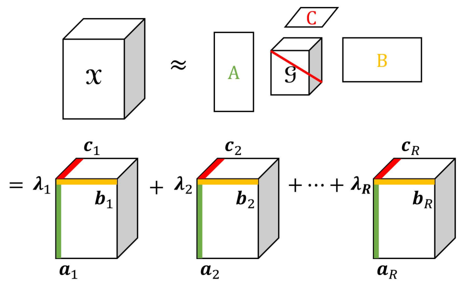

Figure 1: Rank-R PARAFAC decomposition of a three-way tensor $\mathcal{X}$ into three factor matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$.

PARAFAC decomposition for 3-way tensor. Let $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ be a three-way tensor. The rank-R PARAFAC decomposition factorizes the tensor $\mathcal{X}$ into 3 factor matrices, $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$, as follows:

$$\mathcal{X} \approx [\mathbf{A}, \mathbf{B}, \mathbf{C}] = \sum_{r=1}^{R} \lambda_r (\mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r),$$

where $\circ$ denotes an outer product, λ_r is a constant to make a_r, b_r, and c_r unit vectors, and $\mathbf{A} \in \mathbb{R}^{I \times R}$, $\mathbf{B} \in \mathbb{R}^{J \times R}$, and $\mathbf{C} \in \mathbb{R}^{K \times R}$ are the factor matrices. Figure 1 shows the 3-way PARAFAC tensor decomposition.

PARAFAC decomposition for N-way tensor. Generalizing the PARAFAC decomposition for N-way tensor is straightforward. An N-way tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ is decomposed into R factor matrices using PARAFAC as follows:

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

where $\mathbf{A}^{(1)} \in \mathbb{R}^{I_1 \times R}$, $\mathbf{A}^{(2)} \in \mathbb{R}^{I_2 \times R}$, ... , $\mathbf{A}^{(N)} \in \mathbb{R}^{I_N \times R}$ are the factor matrices.

Algorithm for PARAFAC. A well known algorithm for computing the PARAFAC decomposition is Alternating Least Square (ALS), whose 3-way version is shown in Algorithm 1 [10]. Note that $\odot$, $*$, and $\dagger$ denote Kronecker product, Hadamard product, and pseudo-inverse, respectively. The main idea is to alternately minimize each factor matrix after fixing all other factors.

Algorithm 1: ALS for 3-way PARAFAC Decomposition

```
Input: Tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, rank $R$, maximum iterations $T$
Output: PARAFAC decomposition $\lambda \in \mathbb{R}^{R \times 1}, \mathbf{A} \in \mathbb{R}^{I \times R}$, $\mathbf{B} \in \mathbb{R}^{J \times R}$, $\mathbf{C} \in \mathbb{R}^{K \times R}$
 1: Initialize $\mathbf{A}, \mathbf{B}, \mathbf{C}$;
 2: for $t = 1, ..., T$ do
 3:     $\mathbf{A} \leftarrow \mathbf{X_{(1)}} (\mathbf{C} \odot \mathbf{B}) (\mathbf{C}^T\mathbf{C} * \mathbf{B}^T\mathbf{B})^{\dagger}$;
 4:     Normalize columns of $\mathbf{A}$ (storing norms in vector $\lambda$);
 5:     $\mathbf{B} \leftarrow \mathbf{X_{(2)}} (\mathbf{C} \odot \mathbf{A}) (\mathbf{C}^T\mathbf{C} * \mathbf{A}^T\mathbf{A})^{\dagger}$;
 6:     Normalize columns of $\mathbf{B}$ (storing norms in vector $\lambda$);
 7:     $\mathbf{C} \leftarrow \mathbf{X_{(3)}} (\mathbf{B} \odot \mathbf{A}) (\mathbf{B}^T\mathbf{B} * \mathbf{A}^T\mathbf{A})^{\dagger}$;
 8:     Normalize columns of $\mathbf{C}$ (storing norms in vector $\lambda$);
 9:     if convergence criterion is met then
10:         break for loop;
11:     end if
12: end for
13: return $\lambda, \mathbf{A}, \mathbf{B}, \mathbf{C}$;
```

2.2.2. *Tucker Decomposition*

In Tucker decomposition [12], which is also called N-mode Principal Component Analysis (PCA) or N-mode Singular Value Decomposition (SVD), a tensor is decomposed into a non-diagonal core tensor and factor matrices of each mode. While the factor matrices represent the principal components of each mode, the core tensor represents the interactions between the different components. Compared to PARAFAC, Tucker decomposition can express more complex interactions between factors. PARAFAC decomposition can be interpreted as a special case of Tucker decomposition where the core tensor is diagonal.

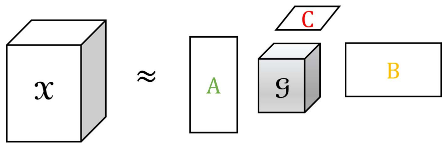

Figure 2: Tucker decomposition of a three-way tensor $\mathcal{X}$ into a core tensor $\mathcal{G}$, and three factor matrices **A**, **B**, and **C**.

Tucker decomposition for 3-way tensor. The 3-way Tucker decomposition of a tensor is as follows.

$$\mathcal{X} \approx [\mathcal{G}; \mathbf{A}, \mathbf{B}, \mathbf{C}] = \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$$
$$= \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} \mathbf{a}_p \circ \mathbf{b}_q \circ \mathbf{c}_r,$$

where $\mathcal{G} \in \mathbb{R}^{P \times Q \times R}$ is the core tensor, and $\mathbf{A} \in \mathbb{R}^{I \times P}, \mathbf{B} \in \mathbb{R}^{J \times Q}$, and $\mathbf{C} \in \mathbb{R}^{K \times R}$ are the factor matrices. Figure 2 shows the Tucker decomposition of a 3-way tensor $\mathcal{X}$.

Tucker decomposition for N-way tensor. The N-way Tucker decomposition of a tensor is as follows.

$$\mathcal{X} \approx [\mathcal{G}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, ..., \mathbf{A}^{(N)}] = \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} ... \times_N \mathbf{A}^{(N)},$$

where $\mathcal{G} \in \mathbb{R}^{J_1 \times J_2 \ldots \times J_N}$ is the core tensor, and $\mathbf{A}^{(1)} \in \mathbb{R}^{I_1 \times J_1}$, $\mathbf{A}^{(2)} \in \mathbb{R}^{I_2 \times J_2}$, ..., and $\mathbf{A}^{(N)} \in \mathbb{R}^{I_N \times J_N}$ are the factor matrices.
Tucker-ALS. Similar to PARAFAC, a well-known algorithm for computing to Tucker decomposition is ALS, which alternately minimizes each factor matrix after fixing all other factors. Algorithm 2 shows the standard SVD-based algorithm called Higher-Order Orthogonal Iteration (HOOI) for 3-way Tucker decomposition.

Algorithm 2: 3-way Tucker-ALS

```
Input: Tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, desired core size: $P \times Q \times R$
Output: Core tensor $\mathcal{G} \in \mathbb{R}^{P \times Q \times R}$ and orthogonal factor matrices $\mathbf{A} \in \mathbb{R}^{I \times P}$, $\mathbf{B} \in \mathbb{R}^{J \times Q}$, and $\mathbf{C} \in \mathbb{R}^{K \times R}$
 1: Initialize $\mathbf{B}, \mathbf{C}$;
 2: repeat
 3:     $\mathcal{Y} \leftarrow \mathcal{X} \times_2 \mathbf{B}^T \times_3 \mathbf{C}^T$;
 4:     $\mathbf{A} \leftarrow P$ leading left singular vectors of $\mathbf{Y}_{(1)}$;
 5:     $\mathcal{Y} \leftarrow \mathcal{X} \times_1 \mathbf{A}^T \times_3 \mathbf{C}^T$;
 6:     $\mathbf{B} \leftarrow Q$ leading left singular vectors of $\mathbf{Y}_{(2)}$;
 7:     $\mathcal{Y} \leftarrow \mathcal{X} \times_1 \mathbf{A}^T \times_2 \mathbf{B}^T$;
 8:     $\mathbf{C} \leftarrow R$ leading left singular vectors of $\mathbf{Y}_{(3)}$;
 9:     $\mathcal{G} \leftarrow \mathcal{Y} \times_3 \mathbf{C}$;
10: until $||\mathcal{G}||$ ceases to increase or the maximum number of outer iterations is exceeded.
```

2.3. Streaming Variants

Sun et al. [13] proposed Dynamic Tensor Analysis (DTA) method to analyze general high dimensional tensor data coming in a stream. DTA decomposes a tensor sequence (a subset of tensor stream) efficiently by updating variance matrices incrementally. They also proposed Streaming Tensor Analysis (STA), which approximates DTA, to further improve the speed. Another variant is a Window-based Tensor Analysis (WTA), by Sun et al. [7], whose purpose is to summarize a tensor stream. WTA is a window-based tensor decomposition method which uses Frobenious norm as a divergence function, and an alternating least square approach. WTA has two variations, independent-window tensor analysis (IW) and moving-window tensor analysis (MW). IW simply treats each tensor window in a tensor stream independently, while MW has an efficient update scheme that utilizes the overlapping information of two consecutive tensor windows.

2.4. Scaling Up

Tensor decomposition algorithms need to be carefully implemented, since they require several multiplications of tensors and factors which could be prohibitive. For example, line 3 of Algorithm 1, if naively implemented, would generate too much intermediate data; this is known as the 'intermediate data explosion' problem. Bader et al. [14] proposed an efficient method for the computation by exploiting the characteristic of Kronecker multiplication, Kang et al. [15, 16] proposed distributed ALS algorithms for the computation using HADOOP, and Beutel et al. [17] proposed a distributed stochastic gradient descent algorithm using HADOOP.

3. Success Stories

In this section, we describe success stories of using scalable tensor mining for analyzing various real world data.

3.1. Higher Order Web Links

Kolda et al. [18] proposed a method called Topical Hypertext Induced Topic Selection (TOPHITS) that extends the Kleinberg's HITS algorithm [19]. The main contribution of the TOPHITS in the determination of important web pages given a query is in the application of anchor text information in addition to the hyperlink structure of the web. In HITS, every node has a hub score and an authority score. The two scores, in convergence, are equal to the singular vectors of the adjacency matrix computed using singular vector decomposition (SVD). The hub score and the authority score are then used to identify hubs and authoritative nodes of the given graph. One limitation of HITS is that the graph analyzed need to be pre-filtered based on the query term. Moreover, even with the pre-filtering, some of the identified hubs and authoritative nodes may not be relevant to the original query due to "topic drift", i.e. the highly ranked pages are not relevant to the original query topic [19]. Addition of the anchor text reduces both problems. However, the addition of anchor text (third dimension) in to the hyperlink structure (first and second dimensions) requires a three-way tensor called the adjacency tensor for storing information. The adjacency tensor forms an adjacency matrix for each anchor text, only linking nodes that are both linked through web structure and have the common anchor text. Similar to SVD in HITS, TOPHIS applies PARAFAC decomposition [20] on the adjacency tensor to extract topic, hub and authority scores of the nodes. The triple vectors $< topic^{(i)}, hub^{(i)}, authority^{(i)} >$ of the i^{th} principle factors of PARAFAC identify important nodes of a topic. In terms of scalability, TOPHITS is not originally designed to run on "big data", but with greedy PARAFAC algorithm and careful selection of MATLAB storage format, it is able to analyze adjacency tensors as large as $50,000$ by $50,000$ by $50,000$ with $500,000$ non-zero entries.

3.2. *Knowledge Base*

Kang et al. [15] applied a large-scale PARAFAC algorithm to a knowledge base data called NELL [2], a real world knowledge base tensor containing (subject, object, verb) triples (e.g., 'Beatles' 'Yesterday', 'sing'). The data contain 15 thousands of subjects/objects, 29 thousands of verbs, and 77 millions of non-zeros. They performed two tasks: discovering important concepts, and detecting contextual synonyms.

A concept in a knowledge base is a group of related subjects, objects, and verbs. Some examples of the discovered concepts from the NELL data include: 1) the "web protocol" concept which consists of a set of subjects (internet, file, data), a set of objects (protocol, software, suite), and a set of verbs (stream, marketing), and 2) the "health system" concept which consists of a set of subjects (health, child, home), a set of objects (provider, system), and a set of verbs (care, insurance, service).

The analyzed factors from the PARAFAC decomposition can be used to detect contextual synonyms, which are noun phrases occurring in similar contexts. Each row of the factor matrix for the "subject" mode from the PARAFAC decomposition can be regarded as a lower-dimensional embedding of each subject, and thus a cosine similarity of two rows of the factor matrix can be used to detect contextual synonyms. Representative examples of synonyms include: 1) "pollutants" is similar to "dioxin", "sulfur dioxide", "greenhouse gases", "nitrogen oxide", etc., and 2) "disabilities" is similar to "infections", "dizziness", "injuries", etc.

3.3. *Network Traffic*

Anomaly detection is especially important in analyzing network traffic data to detect malicious attacks. The following two works effectively find anomalies by spotting abnormal points, and their approaches can be applied in general anomaly detection problems, too.

Maruhashi et al. [9] proposed MultiAspectForensics that detects subgraph patterns in a heterogenous network and visualizes them. They did not examine the overall network but cleverly figured out distinctive points and extracted subgraph patterns from those points. They first drew attribute-eigenscore histogram from factor vectors, and found out spikes that indicate scores shared by many elements. A spike can be regarded as a trend or a pattern in a subgraph. To clearly characterize those subgraph patterns, they defined two generalized patterns: a generalized star pattern and a generalized bipartite pattern. A generalized star pattern is composed of conterminous edges that differ only in one mode, and a generalized bipartite pattern is a dense bipartite structure where edges share exactly the same set of attribute. Using the defined patterns, they characterized abnormal patterns such as port-scanning or distributed denial of service(DDoS) attack, and successfully pointed out those patterns on real-world network traffic dataset. They used LBNL network traffic log dataset which has 4 modes (source IP, destination IP, port number, and a time tick in second), and 281K non-zero elements. They built a 3-mode tensor including source IP, destination IP, and port number from the LBNL dataset and applied MultiAspectForensics on it. The port-scanning attack belongs to a generalized star pattern because a suspicious source machine sends packets to a destination machine through a number of ports. The DDoS attack belongs to generalized bipartite pattern because many malicious hosts send huge amount of packets to a victim through many ports.

Another anomaly detection example with the network traffic log dataset is presented by Sun et al. [13]. With the proposed DTA method (Section 2.3), they narrowed down suspicious points by a multi-level screening process. Using reconstruction errors from DTA, they first detected abnormal tensors, and then found abnormal modes among the abnormal tensors. Lastly, they found the abnormal dimensions in the abnormal modes. They successfully detected the onset of worm-like hierarchical scanning activities.

3.4. DBLP

Analyzing DBLP bibliography dataset reveals interesting patterns that can be useful for recommending related papers or researchers for cooperation, and for tracking changes in research trend.

Kolda et al. [21] presented three data mining case-studies: clustering, trend detection and anomaly detection. They showed a clustering example with DBLP dataset that has author-conference-keyword (5K-1K-1K) triples. They applied a clustering algorithm on decomposed result factor matrices, and they successfully clustered conferences and authors based on their research areas: e.g., KDD, ICDM, and PAKDD are conferences related to groups for data mining. In addition, related keywords are grouped: e.g., database, data, and query are relevant for the area of database.

Sun et al. [13] performed multi-way latent semantic indexing on DBLP dataset to find highly correlated modes. They constructed author-keyword-time (4K-3K-11) 3-way tensor from the DBLP dataset and performed latent semantic indexing. In latent semantic indexing, they found correlation within a mode and correlations across modes. Correlation within a mode is found in a projection matrix where the entries with high values in a column represent important dimensions for the concept group. The core tensor indicates correlations across modes: the entry with high value in the core tensor means a high correlation of corresponding concept groups. They especially focused on correlations across time to find changes in the research trend of a research group: they found that a research group changed its topics from 'object-oriented' (1995) to stream (2004).

3.5. Sensor

Sun et al. [7] presented Multi-Aspect Correlation Analysis (MACA) that simultaneously finds correlations within and across all aspects. In MACA, they suggested meanings of decomposed result: each concept group represents a trend, and a weight from the core tensor corresponds to a global importance of a concept group. A score of an attribute indicates a participation of the attribute in a trend. They analyzed a sensor stream dataset containing 3 modes (time, location and type). They found two important concept groups based on the global importance, one represents the main trend and the other represents the major abnormal trend, and compared the normal main trend and the abnormal trend. In the concept group for main trend, there are daily activation patterns over time which shows high activation during the day and low activation during the night. In the concept group for abnormal trend, there are abnormal daily activation patterns which show high activation during all the time, and abnormal spikes on location pattern.

4. Research Directions

In this section, we discuss promising future research directions in scalable tensor mining.

4.1. Scaling Up

The area of scalable tensor mining is only in its beginning stage, and there are many opportunities in this area. A main opportunity is in developing distributed algorithms for scalable tensor mining; although there are several existing works [15, 22], there are still many tensor algorithms whose distributed version need to be developed: Tucker, DEDICOM, and nonnegative tensor factorizations are some of the examples.

Another opportunity is in sampling tensor data and performing an approximate computation, since the tensor data might be too large to handle even in distributed systems. Papalexakis et al. proposed a sampling based PARAFAC method [23, 24], but there are many tensor algorithms which can benefit the sampling technique.

4.2. Coupled Tensor Analysis

Tensor decomposition can be more accurate if we use an extra information available in addition to the tensor data [25, 17, 26]. For example, decomposing a three-way tensor containing (user, item, time) information can be more accurate if we have an additional matrix of user and their features (age, location, etc.). In another example, a phone call tensor (caller, callee, time) can be more accurately decomposed if we have a social network information between the user. The main challenge is to make a model that effectively exploits both data, and developing efficient and scalable algorithms.

4.3. Applications

Despite the potential to help solve many data mining problems, tensors have not been applied widely. The main reasons are: 1) the lack of scalable algorithms, 2) setting the parameters (e.g., number of ranks) is difficult, 3) pre-processing data (e.g. applying log() to the values) that allow effective tensor mining is not straightforward, and 4) interpretation of the results often require careful post-processing of the output. We expect that the progresses on the above problems will allow many more applications benefiting from the tensor analysis; especially, applications involving time evolving data (e.g. social network over time, phone call network, sensor data, etc.) will benefit significantly.

5. Conclusion

In this paper, we describe algorithms, success stories, and future directions of scalable tensor mining. Tensor mining is a general tool since most multi-dimensional data are modeled as tensors; we discussed many real world tensors as well as success stories (e.g., web analysis, knowledge base mining, network traffic analysis, etc.). However, there still remain challenges and opportunities in scalable tensor mining, including scalability, using side information, and wide applications. We believe scalable tensor mining has a great potential to help solve many data mining problems, and we just started to address the challenges.

Acknowledgments

This work was supported by the Basic Science Research Program through the National Research Foundation of Korea funded by the Ministry of Science, ICT and Future Planning (grants No. 2013005259 and No. 2013R1A1A1064409).

References

[1] K. Maruhashi, F. Guo, C. Faloutsos, Multiaspectforensics: Pattern mining on large-scale heterogeneous networks with tensor analysis, in: International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2011, Kaohsiung, Taiwan, 25-27 July 2011, 2011, pp. 203–210. doi:10.1109/ASONAM.2011.80.
URL `http://dx.doi.org/10.1109/ASONAM.2011.80`

[2] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. H. Jr., T. M. Mitchell, Toward an architecture for never-ending language learning, in: AAAI, 2010.

[3] M. Nickel, V. Tresp, H. Kriegel, Factorizing YAGO: scalable machine learning for linked data, in: Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012, 2012, pp. 271–280.

[4] T. Franz, A. Schultz, S. Sizov, S. Staab, Triplerank: Ranking semantic web data by tensor decomposition, in: The Semantic Web - ISWC 2009, 8th International Semantic Web Conference, ISWC 2009, Chantilly, VA, USA, October 25-29, 2009. Proceedings, 2009, pp. 213–228.

[5] K. Chang, W. Yih, C. Meek, Multi-relational latent semantic analysis, in: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL, 2013, pp. 1602–1612.

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

[7] J. Sun, S. Papadimitriou, P. S. Yu, Window-based tensor analysis on high-dimensional and multi-aspect streams, in: ICDM, 2006, pp. 1076–1080.

[8] T. Kolda, J. Sun, Scalable tensor decompositions for multi-aspect data mining, in: ICDM, 2008.

[9] K. Maruhashi, F. Guo, C. Faloutsos, Multiaspectforensics: Pattern mining on large-scale heterogeneous networks with tensor analysis, in: ASONAM, 2011.

[10] T. Kolda, B. Bader, Tensor decompositions and applications, SIAM review 51 (3).

[11] R. Harshman, Foundations of the parafac procedure: model and conditions for an explanatory multi-mode factor analysis, UCLA working papers in phonetics 16 (1970) 1–84.

[12] L. R. Tucker, Some mathematical notes on three-mode factor analysis, Psychometrika 31 (1966c) 279–311.

[13] J. Sun, D. Tao, C. Faloutsos, Beyond streams and graphs: Dynamic tensor analysis, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '06, ACM, New York, NY, USA, 2006, pp. 374–383.

[14] B. W. Bader, T. G. Kolda, Efficient matlab computations with sparse and factored tensors, SIAM J. Scientific Computing 30 (1) (2007) 205–231.

[15] U. Kang, E. E. Papalexakis, A. Harpale, C. Faloutsos, Gigatensor: scaling tensor analysis up by 100 times - algorithms and discoveries, in: KDD, 2012, pp. 316–324.

[16] I. Jeon, E. E. Papalexakis, U. Kang, C. Faloutsos, Haten2: Billion-scale tensor decompositions, in: ICDE, 2015.

[17] A. Beutel, A. Kumar, E. E. Papalexakis, P. P. Talukdar, C. Faloutsos, E. P. Xing, Flexifact: Scalable flexible factorization of coupled tensors on hadoop, in: SDM, 2014.

[18] T. G. Kolda, B. W. Bader, J. P. Kenny, Higher-order web link analysis using multilinear algebra, in: ICDM, 2005, pp. 242–249.

[19] J. Kleinberg, Authoritative sources in a hyperlinked environment, Journal of the ACM 46 (5) (1999) 604–632.

[20] R. Harshman, Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis.

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

[22] K. Shin, U. Kang, Distributed methods for high-dimensional and large-scale tensor factorization, in: ICDM, 2014.

[23] E. E. Papalexakis, C. Faloutsos, N. D. Sidiropoulos, Parcube: Sparse parallelizable tensor decompositions, in: Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part I, 2012, pp. 521–536.

[24] E. E. Papalexakis, U. Kang, C. Faloutsos, N. D. Sidiropoulos, A. Harpale, Large scale tensor decompositions: Algorithmic developments and applications, IEEE Data Eng. Bull. 36 (3) (2013) 59–66.
URL `http://sites.computer.org/debull/A13sept/p59.pdf`

[25] E. Acar, T. G. Kolda, D. M. Dunlavy, All-at-once optimization for coupled matrix and tensor factorizations, in: MLG'11: Proceedings of Mining and Learning with Graphs, 2011. arXiv:1105.3422.

[26] E. E. Papalexakis, C. Faloutsos, T. M. Mitchell, P. P. Talukdar, N. D. Sidiropoulos, B. Murphy, Turbo-smt: Accelerating coupled sparse matrix-tensor factorizations by 200x, in: SDM, 2014, pp. 118–126.