A Tensor-Based Approach for Big Data Representation and Dimensionality Reduction

Liwei Kuang, Fei Hao, Laurence T. Yang, Man Lin, Changqing Luo, and Geyong Min

***Abstract*—Variety and veracity as two distinct characteristics of big data, are foundations to improve our understanding of underlying information in large scale and heterogeneous data. However, efficient representing and processing the massive heterogeneous data with a unified scheme are becoming challenges in big data area. Existing data representation methods and dimensionality reduction techniques are not suitable for streaming heterogeneous big data, and new solutions are urgently needed to address the challenges.**

In this paper, a unified tensor model is presented to represent the unstructured data (e.g. video clip), semi-structured data (e.g. XML document), and structured data (e.g. GPS data). With tensor extension operator, various types of data are represented as low order sub-tensors and then merged to a high order unified tensor. In order to extract core data sets of the unified tensor, we propose a novel Incremental High Order Singular Value Decomposition (IHOSVD) method. Through recursively calling the incremental matrix decomposition algorithm, IHOSVD can dynamically update the orthogonal bases of tensor unfolding and compute out the new core tensor. Time complexity of the proposed method is $O(k^2n)$ as well as the space complexity is $O(n)$. A case study of intelligent transportation illustrates that approximate data reconstructed from 18% core sets can guarantee 93% accuracy in general which is measured by tensor Frobenius Norm. Theoretical analyses and experiment results demonstrate that the proposed unified tensor model and IHOSVD method are efficient for big data representation and dimensionality reduction.

***Index Terms*—Tensor, HOSVD, Dimensionality Reduction, Data Representation**

I. Introduction

BIG data are a collection of datasets consist of massive unstructured, semi-structured, and structured data. The four main characteristics of big data are volume (amount of data), variety (range of data types and sources), veracity (data quality), and velocity (speed of incoming data). Although many studies have been done on big data processing, very little are known about following two issues: (1) how to represent the various types of data with a simple model; (2) how to extract core data sets which are smaller but contain more valuable information, especially for streaming data. The purpose of this paper is to explore above raised issues which are closely related to the variety and veracity characteristics of big data.

Logic and ontology [1], two knowledge representation methodologies, have been investigated by many researchers. Composed of syntax, semantics and proof theory, logic system is used for making statements about the world. Although the logic is concise, unambiguous and expressive, it only works with the statements that are true or false and is hard to be used for reasoning with unstructured data. Ontology is the set of concepts and relationships that can help people communicate and share knowledge, it is definitive and exhaustive, but it also causes incompatibilities among different application domains, and is not suitable for representing and integrating heterogeneous big data.

The study of data dimensionality reduction has been reported in the literature. Previous approaches include Principal Component Analysis (PCA) [2], Incremental Singular Value Decomposition (SVD) [3], and Dynamic Tensor Analysis (DTA) [4, 5]. The mentioned methods are available for low dimension reduction. However, these methods suffer from some limitations, they are time-consuming when performed on high-dimension data and fail to extract core data sets for streaming big data.

This paper presents a unified tensor model for big data representation and an incremental dimensionality reduction method for high-quality coreset extraction . Data with different formats are employed to illustrate the representation approach, and equivalent theorems are proved to support the proposed reduction method. The major contributions are summarized as follows.

- **Unified Data Representation Model:** We develop a novel tensor model that can unify unstructured, semi-structured, and structured data as a compact model. The tensor model has extensible orders to which new orders can be dynamically appended through the proposed tensor extension operator.
- **Core Tensor Equivalence Theorem:** To solve the recalculation and order inconsistency problems in big data processing with tensor model, we prove a core tensor equivalence theorem which can serve as the theoretical foundation for designing incremental decomposition algorithms.
- **Recursive Incremental HOSVD Method:** We propose a recursive incremental high order singular value decomposition method for streaming data dimensionality reduction. Detailed analysis including time complexity, space complexity and approximation accuracy are also investigated.

The remainder of the paper is organized as follows. Section II recalls the preliminaries of tensor decomposition. We present a unified order tensor model for big data representation in Section III. Section IV proposes an novel incremental dimensionality reduction method. A case study of intelligent transportation is investigated in Section V. After reviewing the related works in Section VI, we conclude the paper in Section VII.

II. Preliminaries

This section reviews the preliminaries on singular value decomposition and tensor decomposition. The core tensor and truncated bases described in the preliminaries can be employed to make big data smaller.

Definition 1: Singular Value Decomposition (SVD). Let $M \in R^{m \times n}$ denotes a matrix, the factorization

$$M = U \Sigma V^{\mathrm{T}} \tag{1}$$

is called the SVD of M. The matrices U and V refer to the left singular vector space and the right singular vector space of matrix M respectively. Both U and V are unitary orthogonal matrices. Matrix $\Sigma = diag(\sigma_1, \sigma_2, ..., \sigma_r, ..., \sigma_k)$, $k = min\{m, n\}$ is a diagonal matrix that contains the singular values of M. In particular,

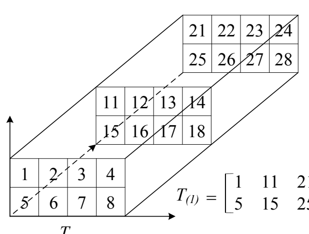
$$M_r = U_r \Sigma_r V_r^{\mathrm{T}} \tag{2}$$

is called the rank-r truncated SVD of M, where $U_r = [u_1, .., u_r]$, $V_r = [v_1, .., v_r]$, $\Sigma_r = diag(\sigma_1, ..., \sigma_r)$, $r < k$. The truncated SVD of M is much smaller to store and faster to compute. Among all rank-r matrices, M_r is the unique minimizer of $\|M - M_r\|_F$.

Definition 2: Tensor Unfolding. Given an N-order tensor $\mathrm{T} \in R^{I_1 \times I_2 \times ... \times I_N}$, the tensor unfolding $T_{(n)} \in R^{I_n \times (I_{n+1} I_{n+2} ... I_N I_1 I_2 ... I_{n-1})}$ contains the element $t_{i_1 i_2 ... i_n i_{n+1} ... i_N}$ at the position with row number i_n and column number that is equal to

$$\begin{aligned}&(i_{n+1} - 1) I_{n+2} ... I_N I_1 ... I_{n-1} + (i_{n+2} - 1)\\ &I_{n+3} ... I_N I_1 ... I_{n-1} + ... + (i_2 - 1) I_3 I_4 ... I_{n-1}\\ &+ \cdots + i_{n-1}.\end{aligned}$$

Example 1. Consider a 3-order tensor $T \in R^{2 \times 4 \times 3}$, Fig. 1 shows the three unfolded matrices $T_{(1)}$, $T_{(2)}$, $T_{(3)}$.



$$T_{(1)} = \begin{bmatrix} 1 & 11 & 21 & 2 & 12 & 22 & 3 & 13 & 23 & 4 & 14 & 24 \\ 5 & 15 & 25 & 6 & 16 & 26 & 7 & 17 & 27 & 8 & 18 & 28 \end{bmatrix}$$

$$T_{(2)} = \begin{bmatrix} 1 & 5 & 11 & 15 & 21 & 25 \\ 2 & 6 & 12 & 16 & 22 & 26 \\ 3 & 7 & 13 & 17 & 23 & 27 \\ 4 & 8 & 14 & 18 & 24 & 28 \end{bmatrix} \quad T_{(3)} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 \\ 21 & 22 & 23 & 24 & 25 & 26 & 27 & 28 \end{bmatrix}$$

Fig. 1. 3-order tensor unfolding, tensor T is unfolded to 3 matrices.

Definition 3: *n*-mode product of a tensor by a matrix. Suppose a tensor $T \in R^{I_1 \times I_2 \times ... \times I_{n-1} \times I_n \times I_{n+1} \times ... \times I_N}$ and a matrix $U \in R^{J_n \times I_n}$, the n-mode product $(T \times_n U) \in R^{I_1 \times I_2 \times ... \times I_{n-1} \times J_n \times I_{n+1} \times ... \times I_N}$ is defined as

$$\begin{aligned}&(T \times_n U)_{i_1 i_2 ... i_{n-1} j_n i_{n+1} ... i_N}\\ &= \sum_{i_n = 1}^{I_n} (a_{i_1 i_2 ... i_{n-1} i_n i_{n+1} ... i_N} \times u_{j_n i_n}).\end{aligned} \tag{3}$$

The n-mode product is a key linear operation for dimensionality reduction, the truncated left singular vector matrix $U_{J_n \times I_n}$ $(J_n < I_n)$ is used to reduce the dimensionality of order I_n from I_n to J_n.

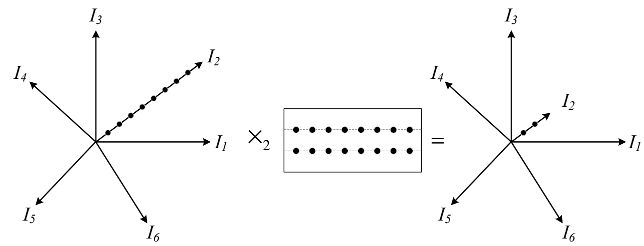


Fig. 2. Tensor dimensionality reduction with n-mode product, dimensionality of the 2nd order is reduced from 8 to 2 by a 2×8 matrix.

Definition 4: Core Tensor and Approximate Tensor. For initial tensor T, the core tensor S and approximate tensor $\hat{T}$ are defined as

$$S = T \times_1 {U_1}^{\mathrm{T}} \times_2 {U_2}^{\mathrm{T}} ... \times_N {U_N}^{\mathrm{T}}, \tag{4}$$

and

$$\hat{T} = S \times_1 U_1 \times_2 U_2 ... \times_N U_N. \tag{5}$$

Core tensor S is viewed as a compressed version of initial tensor T. Big data applications can simply keep the core tensor S and truncated bases $U_1, ..., U_N$. When needed, data can be reconstructed by generating the approximation tensor with Eq. (5). In general cases, the reconstructed data are more efficient than original data because noise, inconsistency and redundancy are removed.

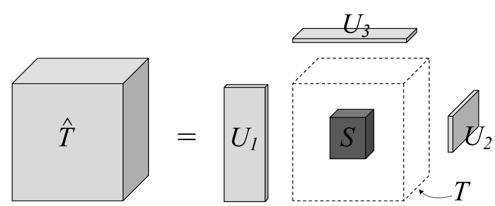


Fig. 3. Illustration of core tensor and approximate tensor. The core tensor and truncated orthogonal unitary bases (U_1, U_2, U_3) are called core data sets that can be used to make big data smaller, while the reconstructed approximate tensor is a substitute for initial tensor.

III. A Unified Data Representation Model

This section proposes a tensor-based data representation model and tensorization approach for transforming heterogeneous data to a unified model. Firstly, a extensible order tensor model and tensor extension operator are presented. Secondly, we illustrate how to tensorize the unstructured, semi-structured, and structured data as sub-tensors. Thirdly, the integration of sub-tensors as a unified tensor is studied. Tensor order and tensor dimension, two confusing concepts, are discussed in the end.

A. Extensible Order Tensor

In general, time and space are two basic characteristics of data collected from different areas, while users are major recipients of data service. Therefore, a general tensor-based data model is defined as

$$T \in R^{I_t \times I_s \times I_u \times I_1 \times \ldots \times I_N}. \tag{6}$$

Eq. (6) shows a $(N+3)$-order tensor which contains two parts, namely the fixed part $R^{I_t \times I_s \times I_u}$ and the extensible part $R^{I_1 \times \ldots \times I_N}$.

In the tensor model, data characteristics are represented as tensor orders. For example, the color space characteristic of unstructured video data can be modeled as I_c. For heterogeneous data, various characteristics are represented as tensor orders and attached to the fix part using the proposed tensor extension operator.

Definition 5: Tensor Extension Operator. Let $A \in R^{I_t \times I_s \times I_u \times I_1}$, and $B \in R^{I_t \times I_s \times I_u \times I_2}$, the tensor extension operator is given by following function

$$f: \; A \vec{\times} B \rightarrow C, \; C \in R^{I_t \times I_s \times I_u \times I_1 \times I_2}. \tag{7}$$

Operator $\vec{\times}$ satisfies the associative law, in other words, $(A \vec{\times} B) \vec{\times} C = A \vec{\times} (B \vec{\times} C)$. With Eq. (7), heterogeneous data can first be tensorized as low order sub-tensors and then extended to a high order unified tensor. The operator merges the identical orders while keep the diverse orders. Elements of the identical order are accumulated together. For instance, sub-tensor T_{sub1} and sub-tensor T_{sub2} have time order denoted as I_{t-1}, I_{t-2}, where $I_{t-1} \in \{i_1, \; i_2\}$, $I_{t-2} \in \{i_1, \; i_3\}$. After extension, time order of the new tensor $T = T_{sub1} \vec{\times} T_{sub2}$ becomes $I_t \in \{i_1, \; i_2, \; i_3\}$.

B. Tensorization Method

Unstructured data include video data and audio data, while semi-structured data are composed of XML documents, ontology data, etc. Representatives of structured data are numbers and character strings stored in relational database. In this paper, video clip, XML document, and GPS data are employed to illustrate the tensorization process.

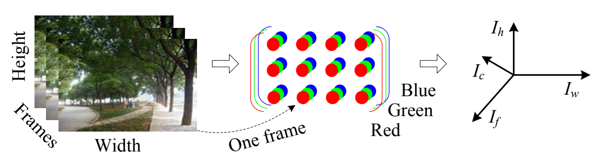


Fig. 4. Represent video clip as 4-order tensor.

Video data can be represented as 4-order tensor or 3-order tensor. To represent a video clip of MPEG-4 format, 25 frames per second, 768×576 resolution and RGB color space, a 4-order tensor $R^{I_f \times I_w \times I_h \times I_c}$ is adopt with I_f, I_w, I_h, I_c indicating frame, width, height and color space. For instance, a 750-frame MPEG-4 video clip with resolution of 768×576 and RGB color can be tensorized as $R^{750 \times 768 \times 576 \times 3}$. In some applications, RGB color is usually transformed to gray level using equation $Gray = 0.299R + 0.587G + 0.114B$, and the representation is replaced by a 3-order tensor $R^{750 \times 768 \times 576}$. Fig. 4 shows the process of transforming a video clip to a 4-order tensor.

Extensible Markup Language (XML) is semi-structured. Fig. 5 shows a simple XML document with seven elements and one attribute. The elements contain tags and contents

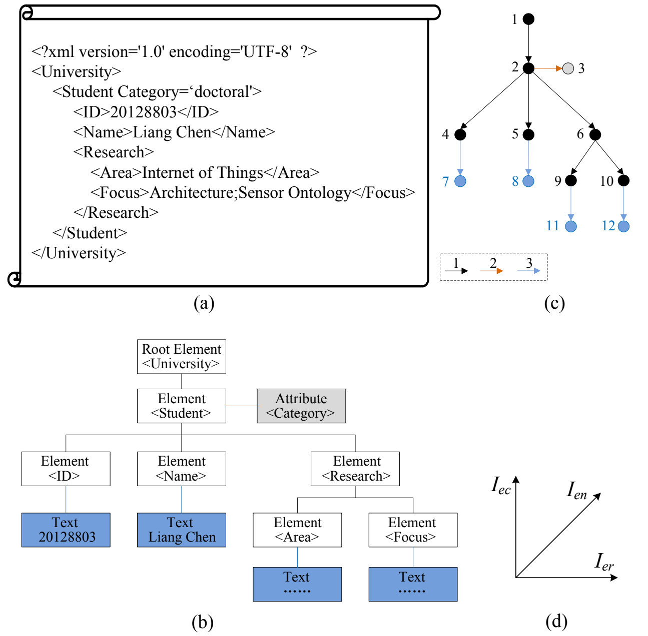


Fig. 5. Represent XML document data as a 3-order tensor. (a) gives an initial XML document, (b) is the parsed tree, (c) shows the relationships between elements, and the 3-order tensor is illustrated in (d).

which consist of characters from unicode repertoire. An XML document has a hierarchical structure and can be parsed as a tree. Fig. 5(b) is the parsed tree of Fig. 5(a). XML Document can be tensorized as a 3-Order Tensor, where I_{er}, I_{ec} indicate the row and column orders of the markup matrix, and I_{en} denotes the content vector order. For example, the XML document in Fig. 5(a) is tensorized as $T \in R^{12 \times 12 \times 28}$, where 28 is the length of Element 'Focus'. Relationships among element, attribute, and text are represented as numbers. In Fig. 5(c), number 1 is used to indicate the parent-child relationship.

Relational database is widely used to manage structured data. In database table, simple field with number or character string type can be modeled as a matrix. For complex field as BLOB, new orders are needed for representation. In Fig. 6, the structured GPS data and student data are unified as a 5-order tensor.

Record	StudentID	Longitude	Latitude	Time
1	D20128803	114.41225837	30.51989529	07-28 10:36:15
2	D20128803	114.41209096	30.51987968	07-28 10:36:25
3	D20128803	114.41194219	30.51992848	07-28 10:36:35
⋮	⋮	⋮	⋮	⋮

Record	StudentID	StudentName
1	D20128803	Liang Chen
⋮	⋮	⋮

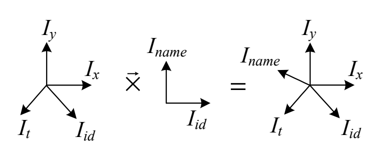


Fig. 6. The upper table is modeled as 4-order sub-tensor, the lower table is modeled as 2-order sub-tensor, the two sub-tensors are unified as 5-order tensor.

C. Unified Tensor Representation Model

Big data are composed of unstructured data d_u, semi-structured data d_{semi} and structured data d_s. Due to the requirement of processing all types of heterogenous data, a unified data tensorization operation is performed using the following equation

$$f : (d_u \cup d_{semi} \cup d_s) \rightarrow \underbrace{T_u \cup T_{semi} \cup T_s}_{T}. \tag{8}$$

With Eq. (7) and Eq. (8), the d_u, d_{semi}, d_s are transformed to subtensors T_u, T_{semi}, T_s which will later be integrated as a unified tensor T. For example, on the basis of transformed video clip, XML document and structured tables as described in Figs. 4, 5, 6, the final tensor is consequently obtained as follows,

$$T \in R^{I_t \times I_s \times I_u \times I_w \times I_h \times I_{er} \times I_{ec} \times I_{en} \times I_{id} \times I_{na}}. \tag{9}$$

In Eq. (9), order I_f is identical to order I_t, order I_x, I_y are combined to order I_s, and order I_c is unnecessary because gray level is adopted. Since too many orders may increase the decomposition complexity, less orders are preferable at the data representation stage.

An element of the 10-order tensor in Eq. (9) is described as an 11-tuple

$$t = (T, S, U, W, H, ER, EC, EN, ID, NA, V), \tag{10}$$

where T, S, U refer to the fixed order time, space and user, W, H denote orders from video data, ER, EC, EN are XML document characteristics, ID, NA are for GPS data, and V is the value of element t. Such type of tuples generated from heterogeneous tensor are usually sparse, and only the nonzero elements are essential for storage and computation. The generalized tuple formate accord with Eq. (6) is defined as

$$t = (T, S, U, i_1, ..., i_N, V). \tag{11}$$

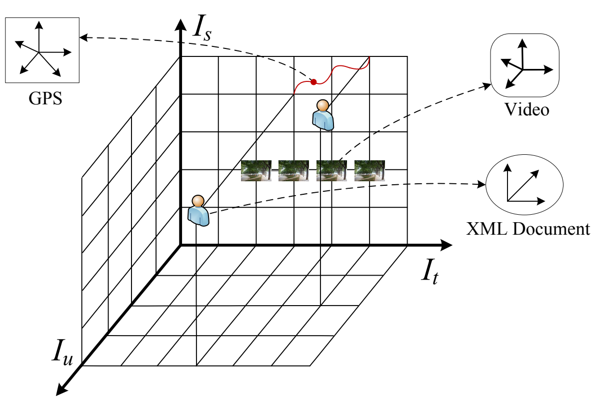


Fig. 7. Visualization of the 2-layer space for data representation

Fig. 7 illustrates the extensible order tensor model from another point of view. The fixed part contains T, S, U is seen as an overall layer, while the extensible part is deemed as an inner layer. Tensor T is simplified as a two layer spaces where the inner space is embedded to the 3-order $(I_t \times I_s \times I_u)$ overall space.

D. Order and Dimension

Tensor order and tensor dimension are two key concepts for data representation, we give a brief comparison between them. Tensor $T \in R^{I_1 \times I_2 \times ... \times I_N}$ has N orders, and order i $(1 \leq i \leq N)$ has I_i dimensions. A N-order tensor can be unfolded to N matrices. For the mode-i unfolded matrix $T_{(i)}$, the number of rows is equal to I_i, while the number of columns is equal to $\prod_{1 \leq k \leq N,\, k \neq i} I_k$. Because of data redundancy and duplication, there exists serious linear dependence which makes the dimensionality very big. In many big data applications, it is impossible to store all dimensions, and only valuable core data sets are essential. During core data sets extraction, the number of tensor orders remains the same while the dimensionality are reduced.

IV. Incremental Tensor Dimensionality reduction

A novel method is proposed for dimensionality reduction on streaming data in this section. Firstly, two problems of tensor decomposition are defined. Then two equivalence theorems are proved and an Incremental High-Order Singular Value Decomposition (IHOSVD) method that can efficiently compute the core data sets on streaming data is presented. Finally, complexity and computation accuracy of the proposed method are disscussed.

A. Problems Definition

Two important problems related to incremental tensor dimensionality reduction are: (1) the recalculation problem; (2) the order inconsistency problem. We formally define them.

Problem 1: Tensor Decomposition Recalculation. Let S_1 denotes the core tensor obtained from previous tensor T_1, χ denotes a new tensor. combining T_1 with χ, we obtain $T_2 = T_1 \cup \chi$. According to Eq. (4), the new core tensor S_2 of new tensor T_2 is computed with

$$S_2 = T_2 \times_1 U_1^{\mathrm{T}} \times_2 U_2^{\mathrm{T}} ... \times_N U_N^{\mathrm{T}}. \tag{12}$$

Decomposition recalculation occurs in Eq. (12) because the previous decomposition results during computing core tensor S_1 are not reused.

Problem 1 can be solved with algorithm 1 and 2 that are designed with the proposed recursive incremental singular value decomposition method.

Problem 2: Tensor Order Inconsistency. Assume T_1, S_2 and T_2 are defined as previous tensor, new core tensor and new combined tensor, to compute S_2 with Eq. (3), the row number of the truncated orthogonal matrix U must be consistent with dimensionality of the tensor order I_n. However, one order dimensionality of the combined tensor T_2 is not equal to the row number of truncated orthogonal matrix U.

For instance, let $T_1 \in R^{2 \times 2 \times 2}$ be a 3-order tensor, and $T_{1(1)} \in R^{2 \times 4}$, $T_{1(2)} \in R^{2 \times 4}$, $T_{1(3)} \in R^{2 \times 4}$ are three unfolded matrices of T_1. Given a new tensor $\chi \in R^{2 \times 2 \times 2}$, combine it with previous tensor T_1 along the third order I_3, we obtain $T_2 \in R^{2 \times 2 \times 4}$. The third order dimensionality of T_2 is 4, while the row number of the truncated orthogonal basis computed from matrix $T_{1(3)}$ is 2. This leads to order inconsistency.

In the following sections, theorem 1, 2 and algorithm 3 are introduced to solve problem 2.

B. Basis and Core Tensor Equivalence Theorems

The left singular vector matrix U plays a key role on dimensionality reduction and data reconstruction. Similarly, the truncated r-rank orthogonal unitary bases $U_1, U_2, ..., U_N$ of the unfolded matrices construct the most basic coordinate axes of a N-order tensor. For heterogeneous big data dimensionality reduction, the major difficulty lies in computing the bases on variable dimension. Our approach extends dimension to fixed length and find out equivalent basis. In this paper, two theorems are presented and proved to support our approach.

Theorem 1: Basis Equivalence of SVD. Define M_1 as a m_1 by n matrix, and M_2 as a m_2 by n matrix whose left m_1 columns contain matrix M_1 and right $m_2 - m_1$ columns are zero. Namely, $M_2 = [M_1 \ 0]$, $M_1 \in R^{m_1 \times n}$, $M_2 \in R^{m_2 \times n}$, $m_1 < m_2$. If the singular value decompositions of matrix M_1 and matrix M_2 are expressed as

$$M_1 = U_1 \Sigma_1 V_1^{\mathrm{T}}, \ M_2 = U_2 \Sigma_2 V_2^{\mathrm{T}}, \tag{13}$$

Then, the unitary orthogonal basis U_1 is equivalent to U_2.

Proof. From Eq. (13), we obtain

$$M_2 M_2^{\mathrm{T}} = [M_1 \ \ 0] \times \left[\begin{array}{c} M_1^{\mathrm{T}} \\ 0 \end{array} \right] = M_1 M_1^{\mathrm{T}}. \tag{14}$$

Consider

$$M_2 \times M_2^{\mathrm{T}} = U_2 \Sigma_2 V_2^{\mathrm{T}} \times V_2 \Sigma_2^{\mathrm{T}} U_2^{\mathrm{T}} = U_2 (\Sigma_2 \Sigma_2^{\mathrm{T}}) U_2^{\mathrm{T}}, \tag{15}$$

and

$$M_1 \times M_1^{\mathrm{T}} = U_1 \Sigma_1 V_1^{\mathrm{T}} \times V_1 \Sigma_1^{\mathrm{T}} U_1^{\mathrm{T}} = U_1 (\Sigma_1 \Sigma_1^{\mathrm{T}}) U_1^{\mathrm{T}}, \tag{16}$$

we obtain

$$U_1 (\Sigma_1 \Sigma_1^{\mathrm{T}}) U_1^{\mathrm{T}} = U_2 (\Sigma_2 \Sigma_2^{\mathrm{T}}) U_2^{\mathrm{T}}. \tag{17}$$

Note that both sides of Eq. (17) are spectral decompositions of two equal symmetric matrix. Additionally, the diagonal matrices $\Sigma_1 \Sigma_1^T$ and $\Sigma_2 \Sigma_2^T$ consist of the eigenvalues of the equal matrix. According to the uniqueness characteristic of eigenvalues, $\Sigma_1 \Sigma_1^T$ and $\Sigma_2 \Sigma_2^T$ are equal. It can be concluded that U_1 is equivalent to U_2. The equivalence implies that U_1 can be calculated by multiply U_2 with a series of Elementary Matrices [6].

On the basis of theorem 1, the following two corollaries can be derived.

Corollary 1: Let $M_1 = [v_1, v_2, ..., v_n]$, $M_2 = [v_1, v_2, ..., 0, ..., 0, ..., v_n]$, where v_i is column vector, then the two matrices have equivalent left singular vector bases.

Corollary 2: Suppose $M_2 = \left[\begin{array}{c} M_1 \\ 0 \end{array} \right]$, then matrix M_1 and matrix M_2 have equivalent left singular vector bases. With corollary 2, the orthogonal basis U_1 can be obtained by trimming the bottom zeros of the orthogonal basis U_2.

Theorem 1 and corollaries 1, 2 are employed to proved Theorem 2 defined as follows. Before the proof, we introduce a special matrix which will be used in Theorem 2.

Definition 6: Extension Matrix. An extension matrix is defined as

$$M = \left[\begin{array}{c} I \\ 0 \end{array} \right], \ M \in R^{J_n \times I_n}, \ J_n > I_n.$$

Multiply the N-order tensor $T \in R^{I_1 \times I_2 \times ... \times I_n \times ... \times I_N}$ with extension matrix M along order n, the dimensionality of this order is extended from I_n to J_n.

Theorem 2: Core Tensor Equivalence of HOSVD. Let T and G be N-order tensors, where $T \in R^{I_1 \times I_2 \times ... \times I_N}$ and $G \in R^{I_1 \times I_2 \times ... \times (kI_n) \times ... \times I_N}$. Define M as an extension matrix, $M \in R^{I_n \times (kI_n)}$. Tensor T and G satisfy

$$T = G \times_n M = G \times_n \left[\begin{array}{c} I_n \\ 0_{kn} \end{array} \right].$$

Proof. Unfold tensor T and tensor G to N matrices $T_{(1)}, \ T_{(2)}, ..., \ T_{(N)}$, and $G_{(1)}, \ G_{(2)}, ..., \ G_{(N)}$. According to Theorem 1 and Corollaries 1, 2, the corresponding unfolded matrices of tensor T and G have equivalent left singular vector bases. Besides, n-mode product of tensor T and matrices A, B posses following properties

$$T \times_m A \times_n B = T \times_n B \times_m A, \tag{18}$$

and

$$T \times_n A \times_n B = T \times_n (BA). \tag{19}$$

Employing Eq. (4), core tensors S_T, S_G are calculated with following equations

$$S_T = T \times_1 U_1^{\mathrm{T}} \times_2 U_2^{\mathrm{T}} \times_3 ... \times_N U_N^{\mathrm{T}}, \tag{20}$$

and

$$S_G = G \times_1 U_1^{\mathrm{T}} \times_2 U_2^{\mathrm{T}} \times_3 ... \times_N U_N^{\mathrm{T}}. \tag{21}$$

With Eqs. (18), (19), (20) and (21), we obtain

$$\begin{aligned} S_T &= T \times_1 U_1^{\mathrm{T}} \times_2 U_2^{\mathrm{T}} \times_3 ... \times_N U_N^{\mathrm{T}} \\ &= (G \times_n M) \times_1 U_1^{\mathrm{T}} \times_2 U_2^{\mathrm{T}} \times_3 ... \times_N U_N^{\mathrm{T}} \\ &= G \times_1 U_1^{\mathrm{T}} \times_2 U_2^{\mathrm{T}} \times_3 ... \times_N U_N^{\mathrm{T}} \times_n M \\ &= S_G \times_n M. \end{aligned} \tag{22}$$

Theorem 2 reveals that extending a tensor by padding zero elements will not transform the core tensor. After unified representation of big data, order number of the incremental tensor and the initial tensor are equal, but the dimensionality are different. Theorem 2 can be used to solve this problem by resizing dimensionality.

C. Incremental High Order Singular Value Decomposition

We propose an IHOSVD method for incremental dimensionality reduction on streaming data. IHOSVD method consists of three algorithms that are used for recursive matrix singular value decomposition and incremental tensor decomposition. The three algorithms are separately described in detail.

Algorithm 1 is a recursive algorithm with recursive function given in Eq. (23). During running process, function f will call itself (Step 4) over and over again to decompose matrices M_n and C_n. Each successive call reduces the size of matrix and moves closer to a solution until finally matrix M_1 is reached, the recursion stops, and the function can exit.

$$f(M_n, C_n) = \begin{cases} svd(M_1), & n = 1 \\ mix(f(M_{n-1}, C_{n-1}), C_n), & n > 1 \end{cases} \tag{23}$$

Algorithm 1 calls function *mix* (Step 5) to merge column vectors of the incremental matrix with the decomposed components of initial matrix. Detailed procedures of function *mix* is described in algorithm 2.

Algorithm 1 Recursive matrix singular value decomposition, $(U,\ \Sigma,\ V) = R - MSvd(M_n,\ C_n)$.

Input:

Initial matrix M_n.

Incremental matrix C_n.

Output:

Decomposition results $U,\ S,\ V$ of matrix $[M_n\ C_n]$.

1: **if** $(n == 1)$ **then**
2: $\quad [U,\ \Sigma,\ V] = svd(M_1)$.
3: **else**
4: $\quad [U_m,\ \Sigma_m,\ V_m] = R - MSvd(M_{n-1},\ C_{n-1})$.
5: $\quad [U,\ \Sigma,\ V] = mix(M_{n-1},\ C_{n-1},\ U_m,\ \Sigma_m,\ V_m)$.
6: **end if**
7: **return** $U,\ S,\ V$.

For most tensor unfolding, the number of rows is less than the number of columns. For such type of matrices, Algorithm 1 can efficiently compute the singular values and singular vectors by splitting the columns for recursive decomposition.

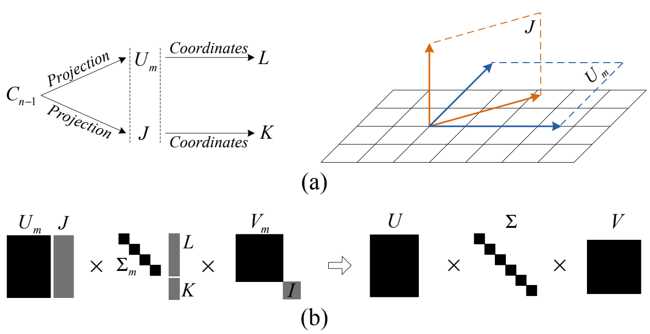


Fig. 8. (a): Incrementally incoming column vectors are projected on unitary orthogonal bases; (b): The middle quasi-diagonal matrix is diagonalized and previous singular vector matrices are updated.

Algorithm 2 applies SVD updating [3] technique for incrementally matrix factorization. Included in matric C_{n-1} are the additional columns which will be projected on the unitary orthogonal bases of previous matrix M_{n-1} (Step 1). Some column vectors are linear combination of orthogonal unitary bases U_m, others are components orthogonal to the space spanned by U_m. As illustrated in Fig. 8, these two types of vectors are separated to obtain the bases U_m, J and coordinates L, K. The operations are implemented as Steps $2 \sim 4$. The column space of singular vector matrix U are spanned by the direct sum of above two unitary orthogonal bases as follows

$$CS(U) = span(U_m \oplus J). \tag{24}$$

Combining the coordinates with previous singular values, we obtain a quasi-diagonal sparse matrix which is easy for decomposition. The new equation consists of above orthogonal bases and coordinates is defined as

$$[M_{n-1},\ C_{n-1}] = [U_m,\ J] \begin{bmatrix} \Sigma_m & L \\ 0 & K \end{bmatrix} \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix}^{\mathrm{T}}. \tag{25}$$

Let U', V' denote the unitary orthogonal bases of the quasi-diagonal matrix in Eq. (25), the updated singular vector matrices are

$$U = [U_m\ J] \times U',\ V = \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix} V'. \tag{26}$$

Eq. (3) suggests only the left singular vector matrix U is essential for tensor decomposition, therefore, computation of matrix V can be omitted in Step 6 of algorithm 2.

Algorithm 2 Merge incremental matrix with decomposition results, $(U,\ \Sigma,\ V) = mix(M_{n-1},\ C_{n-1},\ U_m,\ \Sigma_m,\ V_m)$.

Input:

Initial matrix M_{n-1} and incremental matrix C_{n-1}.

Decomposition results $U_m,\ \Sigma_m,\ V_m$ of matrix M.

Output:

New decomposition results $U,\ \Sigma,\ V$.

1: Project C_{n-1} on the orthogonal space spanned by U_m, $L = U_m^{\mathrm{T}} \times C_{n-1}$.
2: Compute H which is orthogonal to U_m, $H = C_{n-1} - U_m \times L$.
3: Obtain the unitary orthogonal basis J from matrix H.
4: Compute the coordinates of matrix H, $K = J^{\mathrm{T}} \times H$.
5: Execute SVD on the new matrix $[U\ J]$, $[U',\ \Sigma',\ V'] = svd([U\ J])$.
6: Obtain new decomposition results, $([U\ \ J],\ \ U') \rightarrow U,\ \Sigma' \rightarrow \Sigma,\ V' \rightarrow V$.
7: **return** $U,\ S,\ V$.

Employing above two algorithms, we propose Algorithm 3 for incrementally computing core tensor. In the algorithm, extension matrix is used to ensure order consistency (Step 1). Unitary orthogonal bases $U_{(1)}, ..., U_{(N)}$ are updated from Step 2 to Step 4, as well as the new core tensor S is obtained in Step 6. For demonstration of the running process, Fig. 9 shows a simple example with a 3-order tensor.

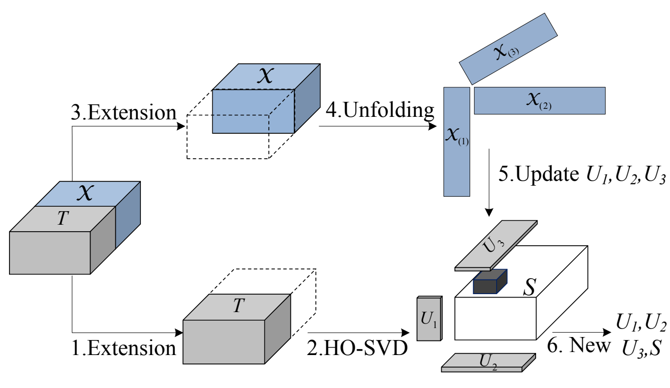


Fig. 9. Example of incremental tensor decomposition, truncated orthogonal bases $U_1,\ U_2,\ U_3$ of new tensor χ are updated incrementally.

Algorithm 3 Incremental tensor singular value decomposition, $(S,\ [U,\ \Sigma,\ V]_{new}) = I - TSvd(\chi,\ T,\ [U,\ \Sigma,\ V]_{initial})$.

Input:

New tensor $\chi \in R^{I_1 \times I_2 \times ... \times I_N}$.

Previous tensor $T \in R^{I_1 \times I_2 \times ... \times I_N}$.

Previous unfolded matrices SVD results $[U,\ \Sigma,\ V]_{initial}$.

Output:

New truncated SVD results $[U,\ \Sigma,\ V]_{new}$.

New core tensor S.

1: Extend tensor χ and tensor T to identical dimensionality.
2: Unfold new tensor χ to matrices $\chi_{(1)}, ..., \chi_{(N)}$.
3: Call algorithm $R - MSvd$ to update above unfolded matrices.
4: Truncate the new orthogonal bases.
5: Combine new tensor χ with initial tensor T.
6: Obtain new core tensor S with n-mode product.
7: **return** S, and $[U,\ \Sigma,\ V]_{new}$.

D. Complexity and Computation Accuracy

1) Time Complexity: Time consumption processes of the proposed IHOSVD method consist of matrix unfolding, incrementally singular value decomposition of each unfolded matrices, and n-mode product of tensor by the truncated bases $U_1, ..., U_N$. Let $Time_{unf}$, $Time_{isvd}$, $Time_{prod}$ denote the time consumed by above process respectively, the total time consumption T satisfies

$$Time = Time_{unf} + Time_{isvd} + Time_{prod}. \qquad (27)$$

Tensor unfolding is a simple transformation with $O(1)$ time complexity. $Time_{isvd}$ is equal to $Time_1 + Time_2 + ... + Time_N = \sum_{i=1}^{N} Time_i$, where $Time_i$ refers to the time consumed by unfolded matrix $T_{(i)}$. According to Eq. (23), time $Time_{isvd}$ are

$$Time(n) = \begin{cases} C_1, & n = 1 \\ Time(n-1) + C_2, & n > 1 \end{cases}, \qquad (28)$$

C_1, and C_2 are constants. The recursive calling process first add columns and then update them with previous decomposition results, the time complexity of one unfolded matrix is $O(k^2 n)$ and the total time complexity is $\sum_{i=1}^{N} k_i^2 I_i$, where I_i denotes the length of order i. The time complexity of T_{isvd} is $O(k^2 n)$ in general.

Computation cost of n-mode product of tensor by matrix lies in the matrix-matrix computation. For a truncated orthogonal basis U with k column vectors, the time complexity is $O(k^2 n)$. Conclusion can be drawn that time complexity of the proposed IHOSVD method is $O(1) + O(k^2 n) + O(k^2 n)$, namely $O(k^2 n)$.

2) Space Complexity: Let $Space_u$ denotes the space used to store all truncated orthogonal bases of unfolded matrices, and $Space_{r-msvd}$, $Space_{mix}$ refer to the memory space for recursive process in algorithm 1. The total space consumed by IHOSVD method is defined as

$$Space = Space_u + Space_{r-msvd} + Space_{mix}. \qquad (29)$$

Complexity of $Space_u$ is equal to $O(n)$. To incrementally compute the core tensor, IHOSVD method needs to keep all the truncated orthogonal bases, the consumed space are $\sum_{i=1}^{N} k_i I_i$. During the recursive process, the temporary spaces needed are

$$M_n + C_n + M_{n-1} + C_{n-1} + ... + M_1 + C_1. \qquad (30)$$

Complexity of above temporary space is $O(n)$. It can be concluded that space complexity of IHOSVD method is $O(n)$.

3) Computation Accuracy: Reconstruction error between initial tensor and approximate tensor can be exactly measured with Frobenius Norm [7] as

$$\left\| T - \hat{T} \right\|_F = \left(\sum_{i_1=1}^{I_1}, ..., \sum_{i_n=1}^{I_N} (a_{i_1,...,i_n} - \hat{a}_{i_1,...,i_n})^2 \right)^{\frac{1}{2}}. \qquad (31)$$

For the unfolded matrix $T_{(i)}$ of initial tensor T, the approximate matrix is $\hat{T}_{(i)} = U_i \Sigma_i V_i^{\mathrm{T}}$. The reconstruction error is caused by approximation of all unfolded matrices. To clearly analyze tensor dimensionality reduction degree and tensor approximation degree, we present two ratios.

Definition 7: The Dimensionality Reduction Ratio of tensor T is defined as

$$\rho = \frac{nnz(S) + \sum_{i=1}^{N} nnz(U_i)}{nnz(T)}, \qquad (32)$$

where S denotes the core tensor, and U_i is the mode-i truncated orthogonal basis. The core data sets of tensor T are composed of S (core tensor) and $U_1, U_2, ..., U_N$. Because only nonzero elements of the core data sets are stored, ratio ρ can accurately reflect dimensionality reduction degree.

Definition 8: The Reconstruction Error Ratio of tensor T is defined as

$$e = \frac{\left\| T - \hat{T} \right\|_F}{\|T\|_F}. \qquad (33)$$

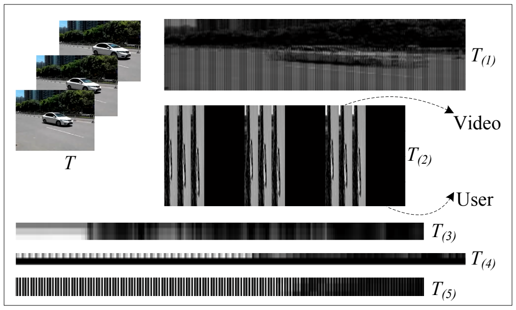
Ratio e reflects the degree of reconstruction error with tensor Frobenius Norm. In this paper, pair (ρ, e) is used to describe the reduction degree and error degree. Obviously, the ratio ρ is inversely proportional to ratio e. Computation accuracy is important for tensor data approximation, and in most applications, HOSVD type algorithms can find a better approximation. To obtain higher accuracy, High-Order Orthogonal Iteration (HOOI) [8] method can be utilized to find the best rank approximation.

V. Case Study

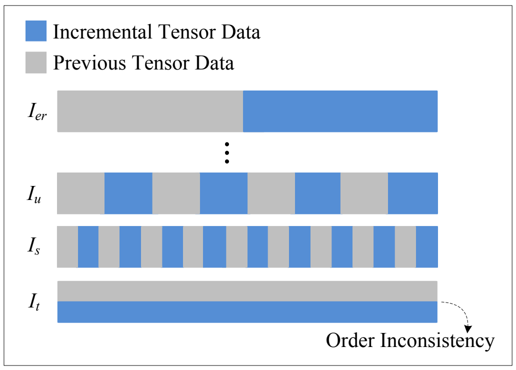
In this section, we illustrate the proposed unified data representation model and incremental dimensionality reduction method with an *Intelligent Transportation* case. Tensor unfolding is demonstrated as well as performance of the proposed method is evaluated.

A. Tensor Unfolding Demonstration

We construct a 5-order tensor $T \in R^{480 \times 640 \times 3 \times 2 \times 3}$ by extracting three frames from unstructured video clip and three users from semi-structured XML document. Fig. 10(a) shows the five unfolded matrices of tensor T. The five orders represent height, width, color space, time and user respectively.



(a) Five unfolded matrices of 5-order tensor.



(b) Incremental data on unfolded matrices of 8-order tensor.

Fig. 10. Heterogenous tensor unfolding and incremental tensor unfolding.

To demonstrate incremental tensor unfolding, an 8-order tensor $T \in R^{I_t \times I_s \times I_u \times I_h \times I_w \times I_c \times I_{ec} \times I_{er}}$ is constructed. Incremental data are appended along the time order I_t. Unfolded matrices of the combined new tensor (initial tensor and incremental tensor) are shown in Fig. 10(b). Order inconsistency of the new tensor occurs in order I_t, because the incremental data are appended as rows on the bottom of the unfolded matrix.

Figs. 10(a), (b) and Fig. 7 in section III illustrate the tensor model from different viewpoints, and demonstrate how the heterogeneous data are stacked together.

B. Dimensionality Reduction and Reconstruction Error

There exists a tradeoff between dimensionality reduction and approximation error. Fig. 11 shows two video frames reconstructed from above 5-order tensor under three different approximation error ratio, namely 0.4%, 7%, and 24%. Fig. 12(a) plots the two ratios together, and illustrates that the reconstruction error ratio increases gradually as the dimensionality reduction ratio decreases. The core data sets are composed of core tensor S and truncated orthogonal bases $U_1, ..., U_5$. Fig. 12(b) shows their proportions to the dimensionality reduction ratio. Generally, the proportion of the core tensor is bigger than the truncated bases.

Diverse data types can result in different dimensionality reduction ratios and approximation error ratios. With repeated experiments on video clips, XML documents and GPS data in this case study, results show that 18% core data sets can guarantee 93% accuracy in general (7% reconstruction error ratio). In practice, the balance between dimensionality reduction and computation accuracy is determined by the application requirement.

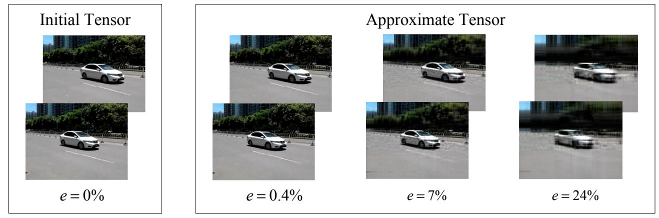
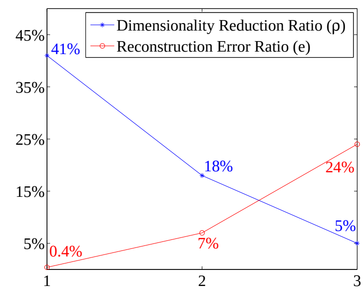
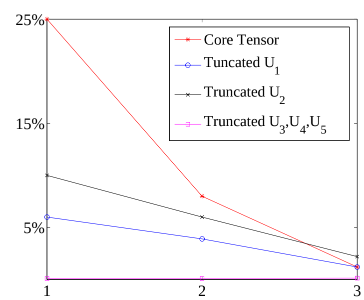


Fig. 11. Video frames reconstructed with different approximation error ratios.



(a)



(b)

Fig. 12. (a) Tradeoff between dimensionality reduction and reconstruction error. (b) Proportion of the core tensor to truncated orthogonal bases. X-Coordinate denotes the serial number of experiment, while Y-Coordinate denotes the experiment result.

C. Consumption of Time and Memory

Compared with the volume of big data, capability of a single computer is limited. This paper mainly focus on the variety and veracity characteristics of big data. To evaluate the proposed incremental dimensionality reduction method, we specially execute the algorithms on a computer with low hardware configurations, i.e., Intel Pentium Dual 1.60 GHz CPU, and 1 GB RAM. Fig. 13 shows that recalculation time of general SVD is 568 seconds which is the sum of all times consumed for decomposing various size of data. Time used in recursive incremental SVD is 428 seconds. The experimental result indicates that the recalculation time is greater than time used in the proposed IHOSVD method. From theoretical point of view, with more orthogonal bases are appended to the left singular vector matrix, the middle quasi-diagonal contains less orthogonal columns, and the time consumption during the diagonalization process decrease. Moreover, IHOSVD method has low space complexity and can decompose tensor more than 30M bytes on the low performance computer, it is because the incremental method projects additional matrices to the previous truncated orthogonal bases rather than directly execute orthogonalization procedure.

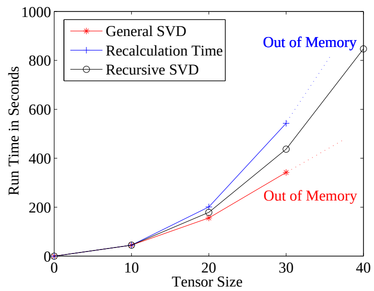


Fig. 13. Comparison between recursive SVD and general SVD, tensor size is measured in Megabyte. Decomposition is conducted with four kinds of size, namely, 10MB, 20MB, 30MB and 40M.

VI. Related Work

This section reviews related works on data representation and high order singular value decomposition.

Data Representation: Big data are composed of unstructured, semi-structured, and structured data. In particular, the multimedia as an unstructured data, is mostly encoded as MPEG4 and H.264. MPEG-4 [9] is a method for defining compression of audio and visual digital data. H.264 [10] is a widely used standard for video compression. The semi-structured Extensible Markup Language (XML) [11, 12] is a flexible text format that defines a set of rules for Encoding documents. XML is both for human-readable and machine-readable. The characteristics making up an XML document are divided into markup and content. Kim and Candan [13] proposed an tensor-based relational data model that can process multi-dimensional structured data. Ontology, such as resource description framework (RDF) [14] and web ontology language (OWL) [15], is playing an ever important role in the exchange of a wide variety of data.

Higher Order Singular Value Decomposition: A tensor [16, 17] is the generalisation of a matrix and usually called multidimensional array. Tensor is a more effective data representation model from which valuable information can be extracted using high order singular value decomposition (HOSVD) [18] method. Because HOSVD imposes orthogonal constraints on the truncated column bases, it may be considered as a special case of the commonly used TUCKER [19] decomposition. Although low rank truncation of the HOSVD is not the best approximation of the initial data, it is considered to be sufficiently good for many applications. Analysis and mining of data with HOSVD has been adopted in many applications such as tag recommendations [20, 21], trajectory indexing and retrieval [22], hand-written digit classification [23].

VII. Conclusions and Futurework

This paper aims at representing and processing the large scale heterogeneous data generated from multiple sources. Firstly, we present a unified tensor-based data representation model that can integrate unstructured, semi-structured, and structured data. Secondly, according to the proposed model, an incremental high order singular value decomposition (I-HOSVD) method is proposed for dimensionality reduction on steaming data, besides, we provide and prove two theorems that can solve the problem of decomposition recalculation and order inconsistency. Finally, an intelligent transportation case is investigated for evaluating the method. Theoretical analysis and results of the case study provide evidence that the proposed data representation model and incremental dimensionality reduction method are promising, and they can pave a way for efficiently mining and analyzing in big data applications.

Future work may focus on recursive and incremental method on CANDECOMP/PARAFAC [24] decomposition, as well as parallel and distributed decomposition on unified tensor model.

References

[1] S. Yang, Y. Xu, and Q. He, "Ontology based service discovery method for internet of things," in *Internet of Things (iThings/CPSCom), 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing*. IEEE, 2011, pp. 43–47.

[2] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[3] M. Brand, "Incremental singular value decomposition of uncertain data with missing values," in *Computer VisionIECCV 2002*. Springer, 2002, pp. 707–72

[4] J. Sun, D. Tao, and C. Faloutsos, "Beyond streams and graphs: dynamic tensor analysis," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, pp. 374–383.

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

[6] H. Anton, *Elementary linear algebra*. Wiley. com, 2010.

[7] C. Meyer, *Matrix analysis and applied linear algebra book and solutions manual*. Siam, 2000, vol. 2.

[8] L. De Lathauwer, B. De Moor, and J. Vandewalle, "On the best rank-1 and rank-(r 1, r 2,..., rn) approximation of higher-order tensors," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1324–1342, 2000.

[9] I. E. Richardson, *H. 264 and MPEG-4 video compression: video coding for next-generation multimedia*. Wiley. com, 2004.

[10] D. Marpe, T. Wiegand, and G. J. Sullivan, "The h. 264/mpeg4 advanced video coding standard and its applications," *Communications Magazine, IEEE*, vol. 44, no. 8, pp. 134–143, 2006.

[11] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau, "Extensible markup language (xml)," *World Wide Web Journal*, vol. 2, no. 4, pp. 27–66, 1997.

[12] E. Van der Vlist, *XML Schema: The W3C's Object-*

Oriented Descriptions for XML. O'Reilly Media, Inc., 2011.

[13] M. Kim and K. S. Candan, "Approximate tensor decomposition within a tensor-relational algebraic framework," in *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 2011, pp. 1737–1742.

[14] I. Horrocks, P. F. Patel-Schneider, and F. Van Harmelen, "From shiq and rdf to owl: The making of a web ontology language," *Web semantics: science, services and agents on the World Wide Web*, vol. 1, no. 1, pp. 7–26, 2003.

[15] D. L. McGuinness, F. Van Harmelen *et al.*, "Owl web ontology language overview," *W3C recommendation*, vol. 10, no. 2004-03, p. 10, 2004.

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

[17] C. M. Martin, "Tensor decompositions workshop discussion notes," *American Institute of Mathematics*, 2004.

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

[19] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.

[20] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos, "Tag recommendations based on tensor dimensionality reduction," in *Proceedings of the 2008 ACM conference on Recommender systems*. ACM, 2008, pp. 43–50.

[21] R. Wetzker, C. Zimmermann, C. Bauckhage, and S. Albayrak, "I tag, you tag: translating tags for advanced user models," in *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 2010, pp. 71–80.

[22] Q. Li, X. Shi, and D. Schonfeld, "A general framework for robust hosvd-based indexing and retrieval with high-order tensor data," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 873–876.

[23] B. Savas and L. Eldén, "Handwritten digit classification using higher order singular value decomposition," *Pattern recognition*, vol. 40, no. 3, pp. 993–1003, 2007.

[24] A. Stegeman and J. M. Ten Berge, "Kruskal's condition for uniqueness in candecomp/parafac when ranks and k-ranks coincide," *Computational statistics & data analysis*, vol. 50, no. 1, pp. 210–220, 2006.

Liwei Kuang is a doctoral student in School of Computer Science and Technology at Huazhong University of Science and Technology, China. His main research interests include Pervasive Computing, Cloud Computing and Big Data. Contact him at liweikuang@gmail.com.

Fei Hao is an assistant professor in School of Computer Science and Technology at Huazhong University of Science and Technology, China. His research interests include Social Computing, Data Mining and Cloud Computing. Contact him at feehao@gmail.com.

Laurence T. Yang is a professor in computer science at Huazhong University of Science and Technology, China and St. Francis Xavier University, Canada. His research interests include Parallel and Distributed Computing, Embedded and Ubiquitous Computing. Contact him at ltyang@gmail.com.

Man Lin is an Associate Professor in the Department of Computer Science, St. Francis Xavier University, Antigonish, NS, Canada. Her research interests include real-time and embedded system design and analysis, scheduling, power aware computing, and optimization algorithms. Contact her at mlin@stfx.ca.

Changqing Luo is an assistant professor in School of Computer Science and Technology at Huazhong University of Science and Technology, China. His research interests include mobile/wireless networks, Cloud Computing, and Green Computing/Communication. Contact him at chqluo2013@gmail.com.

Geyong Min is a professor of Computer Science in the Department of Computing at the University of Bradford, UK. His research interests include Next Generation Internet, Wireless Communications, Modeling and Performance Engineering. Contact him at g.min@brad.ac.uk.