Secure Tensor Decomposition Using Fully Homomorphic Encryption Scheme

Secure Tensor Decomposition for Big Data Using Fully Homomorphic Encryption Scheme

Liwei Kuang, Laurence T. Yang, Jun Feng, and Mianxiong Dong

Abstract—As the rapidly growing volume of data are beyond the capabilities of many computing infrastructures, to securely process them on cloud has become a preferred solution which can both utilize the powerful capabilities provided by cloud and protect data privacy. This paper presents an approach to securely decompose the high-order tensor, a mathematical model widely used in big data applications, to a core tensor and a certain number of truncated orthogonal bases. The unstructured, semi-structured, and structured data are represented as low-order sub-tensors which are then encrypted to cipher counterparts using the RLWE-based fully homomorphic encryption scheme. A unified high-order cipher tensor model is constructed on cloud by collecting all the cipher sub-tensors and embedding them to a base tensor space. The cipher tensor is decomposed through a proposed Lanczos-based algorithm, in which the non-homomorphic square root operation is eliminated. Theoretical analyses of the algorithm in terms of time complexity, memory usage, decomposition accuracy, and data security are provided. Experimental results demonstrate that the proposed approach is feasible and secure to perform high-order tensor decomposition on cloud for big data applications.

Index Terms—Tensor Decomposition, Fully Homomorphic Encryption, Big Data, Lanczos Method.

1 Introduction

The size of data in many fields is rapidly increasing towards Terabyte level or even Petabyte level, as well as the data structures are becoming more varied. The large scale heterogeneous data have posed great challenges on current computing infrastructures, and new approaches are in urgent need to tackle the problems caused by big data. Cloud Computing [1] is a model that can enable ubiquitous and convenient network access to a shared pool of configurable computing resources such as platform, software and service. A cloud infrastructure is the collection of hardware and software which can provide capabilities to the consumers on a pay-per-use or charge-per-use basis. It is a quite feasible approach to upload big data to the cloud for deeply processing and mining such as dimensionality reduction, classification, and prediction [2]. However, carrying out such types of analysis tasks on cloud may cause a series of safety problems including loss of privacy, disclosure of business information, data tamper, etc. Therefore, the study of secure mining and analyzing of big data on cloud is of great necessity as it is an efficient method to extract valuable information from the large scale heterogeneous data.

- *L. Kuang and J. Feng are with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China. E-mail: liweikuang@gmail.com, fengjunhust@qq.com.*
- *L.T. Yang is with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China, and Department of Computer Science, St. Francis Xavier University, Antigonish, NS, Canada. Email: ltyang@gmail.com.*
- *M. Dong is with the Department of Information and Electronic Engineering, Muroran Institute of Technology, 27 − 1 Mizumoto-cho, Muroran, Hokkaido, 050 − 8585, Japan. Email: mx.dong@ieee.org.*

The fully homomorphic encryption scheme allows specific types of computations to be performed on the cyphertext to generate an encrypted result, of which the decryption is identical to the result obtained by directly carrying out operations on the plaintext. The ideal lattice based scheme [3] proposed by Gentry in 2009 solves the problem of limited number of operations of fully homomorphic encryption, which paves the way for non-trusted computing on cloud. The Learn With Errors (LWE) scheme reported in Ref. [4] is more practical to be employed in data-intensive applications. Although the mentioned schemes provide both additive and multiplicative homomorphisms, they can cause decryption errors when be used by algorithms including non-homomorphic operation such as square root which is a fundamental operation for big data processing.

Many heterogenous data are modeled as tensor [5, 6], a type of high dimension matrix widely used in many applications. Tensor decomposition is a powerful tool to extract valuable information from large scale data. The decomposition is computationally expensive and is strongly suggested to be performed on cloud. Therefore, it is necessary to investigate approaches of secure tensor decomposition on cloud and address the challenges caused by non-homomorphic operations. However, little work has been done on such types of studies.

This paper presents a new computing approach which can securely decompose on cloud the high-order tensor generated from large scale heterogeneous data. The major contributions are summarized as follows.

- We present a holistic framework to address the problem of secure tensor decomposition on cloud. The framework not only allows us to utilize the powerful computational capabilities of the cloud, but also can ensure data security during the process

of tensor decomposition.

- We introduce a new Unified Cipher Tensor (*UCT*) representation model. The detailed procedures of how to encrypt the low-order sub-tensors constructed from heterogeneous data as cipher counterparts using the RLWE-based fully encryption scheme, as well as how to embed them to a base tensor space to generate a unified cipher tensor model are illustrated in this paper.
- We propose to employ the Lanczos method to decompose the obtained cipher tensor model to a core tensor and a certain number of truncated orthogonal bases. A secure tensor decomposition algorithm is designed in which the non-homomorphic square root operation is removed. Theoretical analyses of the algorithm in terms of time complexity, memory usage, decomposition accuracy, and data security are provided.

The remainder of this paper is organized as follows. Section 2 recalls the preliminaries. In Section 3 the problem of secure tensor decomposition is formalized and the corresponding solution framework is illustrated. Section 4 explores the method to represent the heterogeneous data as a unified cipher tensor model. A Lanczos based secure tensor decomposition algorithm is proposed in Section 5. Section 6 evaluates the performance of the proposed approach. After reviewing the related works in Section 7, we conclude the paper in Section 8.

2 PRELIMINARIES

In this section, the preliminaries on tensor decomposition, fully homomorphic encryption, and Lanczos method are reviewed. Additionally, the symbols frequently used in this paper are listed in Table 1.

2.1 Tensor Decomposition

Tensor is a type of high dimension matrix widely used in many applications [5] such as computer vision, data mining, graph analysis and signal processing. High-Order Singular Value Decomposition (HO-SVD) is a type of approach that can factorize the tensor to a core tensor multiplied with truncated matrices. Let $\mathrm{T} \in R^{I_1 \times I_2 \times \ldots \times I_N}$ denote an N-th order tensor, S, $\hat{T}$ refer to the core tensor and approximate tensor respectively, then the HO-SVD method is defined as

$$\begin{aligned} S &= T \times_1 {U_1}^{\mathrm{T}} \times_2 {U_2}^{\mathrm{T}} \ldots \times_N {U_N}^{\mathrm{T}}, \\ \hat{T} &= S \times_1 U_1 \times_2 U_2 \ldots \times_N U_N. \end{aligned} \tag{1}$$

The i-mode product $T \times_i U,\ 1 \leq i \leq N$, of a tensor by a matrix in Eq. (1) is defined as

$$\begin{aligned} &(T \times_i U)_{j_1 j_2 \ldots j_{i-1} k_i j_{i+1} \ldots j_N} \\ &= \sum_{j_i=1}^{I_i} (t_{j_1 j_2 \ldots j_{i-1} j_i j_{i+1} \ldots j_N} \times u_{k_i j_i}), \end{aligned} \tag{2}$$

where $t_{j_1 j_2 \ldots j_{i-1} j_i j_{i+1} \ldots j_i}$ and $u_{k_i j_i}$ refer to the elements of tensor T and matrix U respectively.

TABLE 1
Table of symbols.

Symbol	Definition
T	initial tensor
S	core tensor
$\hat{T}$	approximate tensor
$T_{(i)}$	i-mode unfolded matrix
$Sym(T_{(i)})$	symmetric matrix generated with $T_{(i)}$
D	diagonal matrix
L	tridiagonal matrix
$\alpha,\ \beta$	elements of the tridiagonal matrix
$U\ (V)$	left (right) singular vector matrix
$\Sigma\ (\Lambda)$	singular (eigen) value
$\times_i$	i-mode product of a tensor by a matrix
$\mathcal{R}$	set of real numbers
$\mathcal{Z}$	set of integers
$R\ (R[x])$	ring (polynomial ring)
m	plaintext
$c = \{\dot{c},\ \ddot{c}\}$	ciphertext
χ	discrete gauss distribution
e	randomly selected error from χ
$q,\ p$	big prime integer
$Ec\ (Dc)$	encryption (decryption) function
Ψ^E	cipher data of Ψ, namely $\Psi^E = Ec(\Psi)$

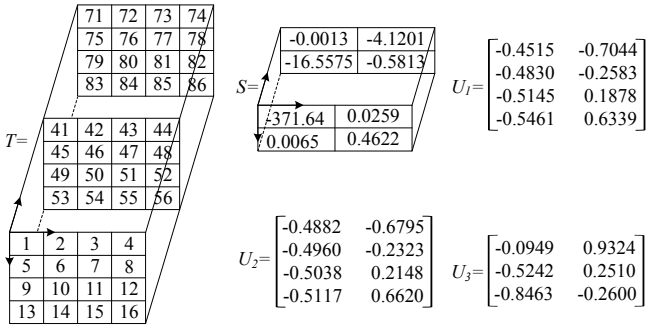


Fig. 1. Decomposing a three-order tensor to a core tensor and three truncated orthogonal bases.

For example, Fig. 1 demonstrates the obtained core tensor S and the truncated bases U_1, U_2, U_3 by decomposing the initial tensor T. The 4 by 4 by 3 tensor is decomposed to a 2 by 2 by 2 core tensor, two matrices of 4 by 2 and a matrix of 3 by 2. Generally, the core tensor and the truncated bases are considered as a compressed version of the initial tensor T, the reconstructed data in the approximate tensor $\hat{T}$ are more efficient as the noise data and inessential data are removed.

2.2 Fully Homomorphic Encryption

Homomorphic encryption is a new type of scheme that allows specific types of operations to be performed on the cyphertext to obtain the encrypted result, of which the decryption is identical to the result directly computed by performing operations on the plaintext. Two fully homomorphic encryption schemes [3, 7] are proposed

using ideal lattice and polynomial ring respectively. A new approach without bootstrapping [8] is presented by Brakerski and Gentry. The fully homomorphic encryption scheme utilized in this paper is introduced by Kristin Lauter etc. [4]. The four fundamental steps of this scheme are as follows:

1) ***Setup***$(1^\lambda, 1^\mu)$***:*** Select a μ-bit modulus q to construct a polynomial ring R_q and a discrete gauss distribution χ, set d as the degree of polynomial x^d+1, where $d = d(\lambda, \mu)$. Select $R = \mathcal{Z}[x]/(x^d+1)$, $R_q = R/qR$, then obtain the public parameters $params = \{q,\ d,\ \chi\}$.
2) ***KeyGen***$(params)$***:*** Randomly select elements $s \leftarrow \chi$ and $\ddot{a} \leftarrow R_q$, as well as an error $e \leftarrow \chi$. Let $\dot{a} = -\ddot{a}s + 2e$, then obtain the public key $pk = \{\dot{a},\ \ddot{a}\}$, as well as the private key $sk = s$.
3) ***Ec***$(pk\ m)$***:*** To encrypt a message m, set $u \leftarrow \chi$, $g \leftarrow \chi$, $r \leftarrow \chi$, compute $\dot{c} = \dot{a}u + yg + m,\ \ddot{c} = \ddot{a}u + yr$, and output the ciphertext $c = \{\dot{c},\ \ddot{c}\}$, where y is a modulus parameter.
4) ***Dc***$(sk\ c)$***:*** Obtain the plaintext $m = ((\dot{c} + \ddot{c} \times sk) \bmod q) \bmod y$ using the private key sk.

The encryption scheme supports the homomorphism of addition and multiplication, which can be described as follows

$$\begin{aligned} &Ec(m_1) + Ec(m_2) = Ec(m_1 + m_2), \\ &Ec(m_1 \times m_2) = Ec(m_1) \times Ec(m_2). \end{aligned} \tag{3}$$

Let $c_1 = (\dot{c}_1,\ \ddot{c}_1)$ and $c_2 = (\dot{c}_2,\ \ddot{c}_2)$ be two encrypted messages, according to Eq. (3), the homomorphic addition and multiplication operations are as follows

$$\begin{aligned} c_1 + c_2 &= \dot{c}_1 + \ddot{c}_1 sk + \dot{c}_2 + \ddot{c}_2 sk \\ &= (\dot{c}_1 + \dot{c}_2) + (\ddot{c}_1 + \ddot{c}_2)sk, \\ c_1 \times c_2 &= (\dot{c}_1 + \ddot{c}_1 sk)(\dot{c}_2 + \ddot{c}_2 sk) \\ &= \dot{c}_1\dot{c}_2 + (\dot{c}_1\ddot{c}_2 + \ddot{c}_1\dot{c}_2)sk + \ddot{c}_1\ddot{c}_2 sk^2. \end{aligned} \tag{4}$$

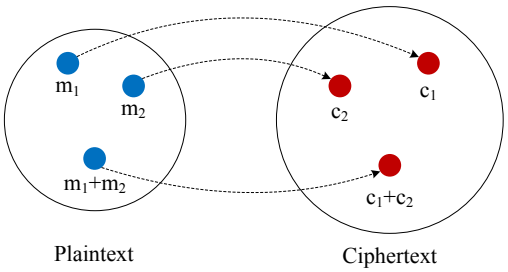


Fig. 2. Illustration of the homomorphic encryption.

Fig. 2 demonstrates the homomorphic encryption of an addition operation. Let m_1, m_2 be two elements in the plaintext, c_1, c_2 in the ciphertext, and $c_1 = Ec(m_1)$, $c_2 = Ec(m_2)$, then $m_1 + m_2 = Dc(Ec(m_1) + Ec(m_2))$.

2.3 Lanczos Method.

The Lanczos method [9] is efficient for computing the eigenvalues and eigenvectors of a sparse symmetric matrix A. It transforms the matrix A with an orthogonal matrix W, where $W = [w_1,\ \ldots,\ w_k]$ and $W^{\mathrm{T}}W = I$, to a tridiagonal matrix as follows

$$L = \begin{bmatrix} \alpha_1 & \beta_2 & & \\ \beta_2 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_k \\ & & \beta_k & \alpha_k \end{bmatrix}. \tag{5}$$

Equating columns in the expression $AW = WL$, the tridiagonal matrix L can be obtained by carrying out the iteration procedures

$$\begin{aligned} &\alpha_j = w_j^{\mathrm{T}} A w_j, \\ &r_j = Aw_j - \alpha_j w_j - \beta_j w_{j-1}, \\ &\beta_{j+1} = \|r_j\|_2,\ w_{j+1} = r_j/\beta_{j+1}. \end{aligned} \tag{6}$$

The components of $\alpha,\ \beta,\ r$ can be progressively calculated. Let the eigenvalue decomposition of matrix L be defined as $L = Q\Lambda Q^{\mathrm{T}}$, then the eigenvalues and eigenvectors of matrix A are Λ and WQ. In Eq. (6), the matrix-vector product is the frequently called linear transformation during the Lanczos iteration.

3 Problem Definition and Solution Framework

This section formalizes the problem of secure tensor decomposition on the bases of the fully homomorphic encryption scheme, and provides an overview of the proposed solution framework.

3.1 Problem Definition

Big data consist of unstructured data D_u, semi structured data D_{semi} and structured data D_s. Let $core$ denote the core data including the core tensor S and the truncated orthogonal bases $U_1,\ U_2,\ \ldots,\ U_N$, then the secure tensor decomposition problem can be formalized as

$$\begin{aligned} &f_r : \{Ec(D_u), Ec(D_{semi}), Ec(D_s)\} \rightarrow Ec(T), \\ &f_d : Ec(T) \rightarrow \{Ec(S), Ec(U_1),\ \ldots,\ Ec(U_N)\}. \end{aligned} \tag{7}$$

In Eq. (7), the data representation function f_r integrates all the encrypted data as a unified cipher tensor model (*UCT*), on which the decomposition function f_d is performed to obtain the encrypted core tensor as well as the encrypted truncated orthogonal bases.

As the decomposition operations are carried out on the encrypted data, the user's privacy are protected. In order to guarantee the correctness of the decomposition result, Eq. (7) satisfies $S = T \times_1 U_1^{\mathrm{T}} \times_2 U_2^{\mathrm{T}} \ldots \times_N U_N^{\mathrm{T}}$. According to the fully homomorphic encryption scheme, the secure decomposition process satisfies the following equation

$$Dc(sk,\ Ev(pk,\ C_{f_d},\ Ec(T))) = C_{f_d}(T), \tag{8}$$

where Ev, Ec, Dc refer to the evaluation, encryption, and decryption function, pk and sk denote the public key and private key, C_{f_d} refers to the boolean circuits according to the decomposition function f_d defined in Eq. (7).

The homomorphism can be guaranteed by simply adding, subtracting, or multiplying the cipher data during the tensor decomposition process. However, new challenges arise as the non-homomorphic operations such as division, square root are adopted in some types of decomposition methods, e.g. Lanczos-based algorithm. A secure tensor decomposition algorithm is proposed in this paper to address these challenges.

For convenience, in the following sections this paper adopts the symbol Ψ^E to denote the cipher data according to the plain data Ψ, namely $\Psi^E = Ec(\Psi)$. Therefore, the encrypted tensor $Ec(T)$ is denoted as T^E.

3.2 Overview of the Solution Framework

To address the problem defined above, this paper proposes a secure tensor decomposition approach based on the fully homomorphic encryption scheme. Fig. 3 provides an overview of the framework where the unstructured, semi-structured, and structured data are encrypted and represented as a unified tensor model which are then securely factorized to a core tensor and a certain number of truncated orthogonal bases. All the processes are carried out on the encrypted data without decryption. The four representative steps of the framework are summarized as follows.

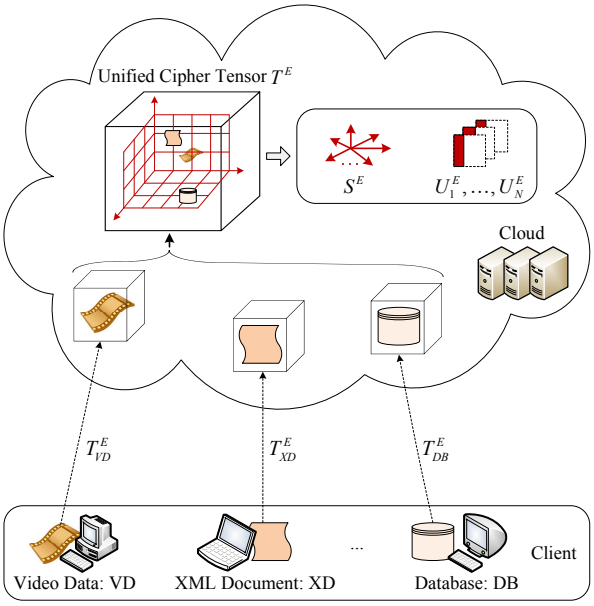


Fig. 3. Framework overview of the secure tensor decomposition approach.

1) ***Data Representation, Encryption and Submission:*** The heterogeneous data collected in the clients are represented as low-order sub-tensors using the method proposed in our previous work [6]. Then the sub-tensors are encrypted using the RLWE-based fully homomorphic encryption scheme and the obtained cipher results are submitted to the cloud for unification and decomposition. In Fig. 3, the unstructured video data VD, semi-structured XML document XD and structured database DB are transformed to cipher sub-tensors T_{VD}^E, T_{XD}^E, T_{DB}^E respectively.
2) ***Construction of Cipher Tensor:*** The obtained sub-tensors T_{VD}^E, T_{XD}^E, and T_{DB}^E are then embedded to a base tensor model $T_{base} \in R^{I_{tim} \times I_{spa} \times I_{clt}}$ to generate a unified cipher tensor model T^E using the tensor extension operation $T^E = T_{base} \vec{\times} T_{VD}^E \vec{\times} T_{XD}^E \vec{\times} T_{DB}^E$. The three orders I_{tim}, I_{spa}, I_{clt} of the base tensor denote the time, space and client characteristic.
3) ***Secure Tensor Decomposition:*** After unfolding the unified cipher tensor T^E to matrices $T_{(1)}^E, \ldots, T_{(N)}^E$, where N is the number of orders of tensor T^E, the symmetrization transformation is performed on each tensor unfolding to obtain the symmetric matrix $sym(T_{(i)}^E) = T_{(i)}^E (T_{(i)}^E)^{\mathrm{T}}$, $1 \leq i \leq N$. The eigen vectors of the symmetric matrix $sym(T_{(i)}^E)$ are corresponding to the left singular vectors of matrix $T_{(i)}^E$. The Lanczos method is employed to perform the eigen value decomposition, namely, $sym(T_{(i)}^E) = U_i^E \Lambda^E (U_i^E)^{\mathrm{T}}$. The cipher core tensor S^E can be computed by applying Eq. (1) to the truncated bases $U_1^E, \ldots, U_N^E$ and the unified cipher tensor T^E.
4) ***Obtain the Plain Core Tensor and Bases:*** By decrypting the cipher core tensor and cipher truncated bases obtained in Step 3, the plain core tensor S and plain truncated bases $U_1, \ldots, U_N$ can be computed. As the homomorphism are supported during the secure tensor decomposition, the obtained results are correct and are identical to that computed over the plain data.

This paper focuses on Step 2 and Step 3, which correspond to the secure representation function f_r and secure tensor decomposition function f_d.

4 Construction of Cipher Tensor via Fully Homomorphic Encryption

This section illustrates the process of representing the heterogeneous data as a unified cipher tensor model via the fully homomorphic encryption scheme. New concepts and operations closely related to the cipher tensor are introduced.

4.1 Cipher Tensor and Nil Element

In order to clearly describe the process of representing the unstructured, semi-structured, and structured data as a unified cipher tensor model, this paper introduces some definitions as follows.

Definition 1: Cipher Tensor. A cipher tensor T^E is obtained by encrypting the elements in the plain tensor T using the fully homomorphic encryption scheme. The number of orders of tensor T^E is equal to that of tensor T. The construction process is defined as $T^E = \{Ec(t) | t \in T\}$.

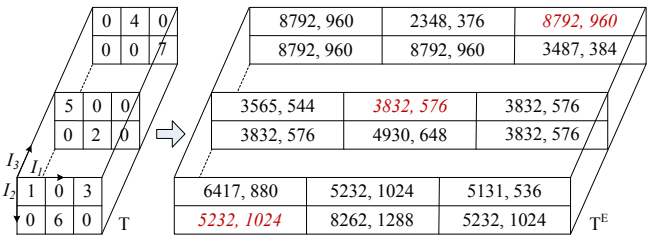


Fig. 4. A plain tensor and the corresponding cipher tensor.

Fig. 4 demonstrates a plain tensor and the corresponding cipher tensor. In this instance, the public key is $pk = (141,\ 13)$ as well as the private key is $sk = 31$. The parameter q is set to 78791, and the messages are of three bit, namely $m \in \{0,\ 1,\ \ldots,\ 7\}$. The encryption function is formalized as $\dot{c} = \dot{a}u{+}8g{+}m,\ \ddot{c} = \ddot{a}u{+}8r$, as well as the decryption function is $m = ((\dot{c} + \ddot{c}s)) \bmod q) \bmod 8$. The parameters u, g, r are randomly selected from a discrete gauss distribution. The two tensors in Fig. 4 satisfy the equation $T^E = Ec(pk,\ T) = Ec(\{141,\ 13\},\ T)$. This is a simple example to illustrate the construction of a cipher tensor. In practice, the fully homomorphic encryption scheme chooses large integers for these parameters.

Definition 2: Nil Cipher Element. The element obtained by encrypting the plain element 0 is called the Nil cipher element. This paper adopts symbol 0^E to denote the Nil cipher element, namely $0^E = Ec(0)$.

As three randomly selected parameters, i.e. u, g, r, are employed during tensor encryption, the plain element 0 may be encrypted to different Nil cipher elements. For example, in Fig. 4, the plain element $t_{121} = t_{212} = t_{313} = 0$, however the cipher element $t^E_{121} = (5232,\ 1024)$, $t^E_{212} = (3832,\ 576)$, and $t^E_{313} = (8792,\ 960)$.

Definition 3: Sparse Cipher Tensor. A cipher tensor containing a large portion of Nil elements is called a sparse one. In this paper, a sparse tensor is assumed to contain more than 60% Nil elements.

In Fig. 4, the cipher tensor T^E consists of 18 elements, of which 11 are Nil elements. Therefore, T^E is deemed as a sparse cipher tensor.

Definition 4: Reduced Cipher Tensor. A reduced cipher tensor is obtained by removing all the Nil elements from the cipher tensor model.

As the zero element in the plain data may be encrypted to different Nil elements in the cipher tensor, special methods are needed to remove the Nil elements to obtain the reduced cipher tensor. In the proposed solution framework demonstrated in Fig. 3, the clients are responsible for removing the zero elements from the plain tensor models before encryption. This method can reduce the communication traffic.

4.2 Constructing a Unified Cipher Tensor Model

In this paper, the heterogenous data are first represented and encrypted as cipher sub-tensors in the clients, then they are submitted to the cloud for unification. To integrate all the cipher tensors, a base tensor model is proposed, which is defined as $T_{base} \in R^{I_{tim} \times I_{spa} \times I_{clt}}$, where I_{tim}, I_{spa}, I_{clt} refer to the time, space and client characteristic. The three orders serve as a basis to which various types of encrypted sub-tensors can be appended to generate a unified cipher tensor model.

For example, a unstructured video data VD can be transformed to a four order tensor model $T_{VD} \in R^{I_f \times I_h \times I_w \times I_{cs}}$, where the orders I_f, I_h, I_w, I_{cs} refer to the frame, image height, image width, and color space respectively. A semi-structured data XD can be represented as a three order tensor model $T_{XD} \in R^{I_{ia} \times I_{ib} \times I_r}$, where the orders I_{ia}, I_{ib}, I_r denote the XML elements and relationships [6]. The two tensors can be encrypted to T^E_{VD}, T^E_{XD} which then be embedded to the base tensor model T_{base}. The constructed unified cipher tensor model can be defined as

$$T^E \in R^{I_{tim} \times I_{spa} \times I_{clt} \times I_h \times I_w \times I_{cs} \times I_{ia} \times I_{ib} \times I_r}. \tag{9}$$

The frame order of the video data is integrated to tensor order I_{tim}. This nine-order tensor includes all data characteristics from the video, XML document and base tensor. All elements in the cipher tensor T^E are involved in secure decomposition.

4.3 Tensor Unfolding and Memory Storage Scheme

When the unified cipher tensor is generated, the next critical step is to obtain the tensor unfolding, which are then transformed to symmetric matrices. For sparse tensor, the Compressed Row Storage (CRS) [10] method is employed to store the unfolded matrices. The CRS scheme is efficient for matrix-vector product and can reduce memory usage during tensor decomposition. Additionally, to decrease time consumption of the secure tensor decomposition algorithm, this paper employs $T^E_{(i)}((T^E_{(i)})^T v)$ to perform the matrix-vector operation on the symmetric matrix of the i-mode tensor unfolding, namely, the vector v is first left multiplied with matrix $(T^E_{(i)})^T$ to obtain a temp vector, which are then left multiplied with matrix $T^E_{(i)}$.

In order to unfold a cipher unified tensor, the non-Nil elements in the cipher tensor are rearranged along the rows of the corresponding unfolded matrices. Given an N-order tensor $\mathrm{T^E} \in R^{I_1 \times I_2 \times \ldots \times I_N}$, the tensor unfolding [5] $T^E_{(i)} \in R^{I_p \times (I_{i+1} I_{i+2} \ldots I_N I_1 I_2 \ldots I_{i-1})}$ contains the element $t_{j_1 j_2 \ldots j_i j_{i+1} \ldots j_N}$ at the position with the row number j_i and the column number that is equal to $(j_{i+1} - 1) I_{j+2} \ldots I_N I_1 \ldots I_{j-1} + (j_{i+2} - 1) I_{j+3} \ldots I_N I_1 \ldots I_{i-1} + \ldots + (j_2 - 1) I_3 I_4 \ldots I_{j-1} + \ldots + i_{i-1}$. For the i-mode unfolded matrix $T^E_{(i)}$ of the cipher tensor T^E, the number of rows is equal to I_i, as well as the number of columns is equal to $\prod_{j=1, j \neq i}^{N} I_i$.

Fig. 5 demonstrates the 1-mode unfolded matrix of a three order cipher tensor. The five none-Nil elements in tensor T^E are kept and rearranged to the unfolded matrix $T^E_{(1)}$. This unfolded cipher matrix is employed for singular value decomposition. The Compressed Row

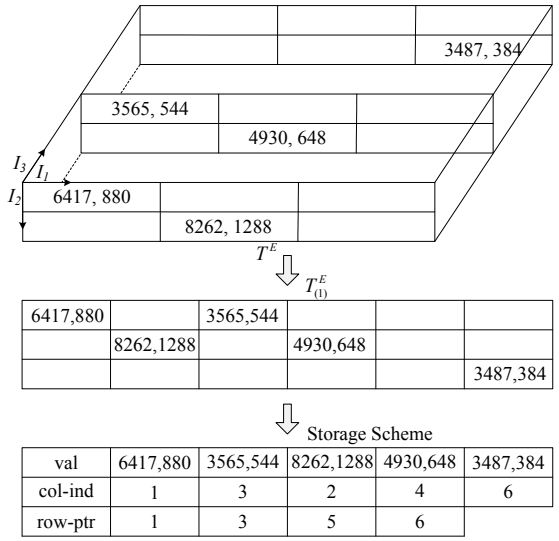


Fig. 5. The 1-mode unfolded matrix of a 3-order cipher tensor and the corresponding storage scheme.

Storage scheme [10] of the unfolded matrix is demonstrated at the bottom table in Fig. 5, where the *val* array consists of the five non-Nil elements, *col-ind* contains the column indices, *row-ptr* stores the four locations that start new rows. For example, the element 3 in array *row-ptr* indicates that $(8262,\ 1288)$ starts a new row.

4.4 Cipher Tensor Representation Algorithm

Based on the above mentioned methods, this paper proposes Algorithm 1 to represent the heterogeneous data as a unified cipher tensor (*UCT*) model.

Algorithm 1 Cipher Tensor Representation. $T^E = f_r(D_u,\ D_{semi},\ D_s)$

Input:
Heterogeneous data $D_u,\ D_{semi},\ D_s$.
Output:
The unified cipher tensor model T^E.
1: Represent local data as low-order sub-tensors, and encrypt them to cipher sub-tensors in clients.
2: Upload the obtained cipher sub-tensors to cloud.
3: Embed the uploaded cipher sub-tensors to the base tensor model T_{base}, and obtain the unified cipher tensor model T^E.
4: Unfold the cipher tensor to matrices and obtain the symmetric matrices for decomposition.

In Line 1 of the proposed Algorithm, the unstructured, semi-structured, and structured data are transformed to low-order sub-tensors, which are then encrypted using the fully homomorphic encryption scheme. In this paper, the zero elements of the plain data are removed during the encryption procedure. The cloud embeds all the cipher sub-tensors to the base tensor model in Line 3 to obtain the unified cipher tensor model T^E. Line 4 obtains the symmetric matrices of each tensor unfolding for secure decomposition.

5 Secure Tensor Decomposition

This section presents a secure tensor decomposition algorithm to obtain the core tensor and the truncated orthogonal bases. Theoretical analyses of the algorithm in terms of time complexity, memory usage, decomposition accuracy and data security are provided in this section.

5.1 Non-Homomorphic Operations During Lanczos-based Decomposition

TABLE 2
Utilized Operations in Lanczos Method.

Operation	Homomorphic	Step
$+$	yes	$\alpha_j = w_j^{\mathrm{T}} A w_j$
$-$	yes	$r_j = Aw_j - \alpha_j w_j - \beta_j w_{j-1}$
$\times$	yes	$\alpha_j = w_j^{\mathrm{T}} A w_j$
$\div$	no	$\omega_{j+1} = {}^{r_j}/_{\beta_{j+1}}$
$\sqrt{x}$	no	$\beta_{j+1} = \|r_j\|_2$

Table 2 shows the five types of operations utilized in the Lanczos iteration, namely, addition $+$, subtraction $-$, multiplication $\times$, division $\div$, and square root $\sqrt{x}$. The first three operations are homomorphic, while the last two are non-homomorphic. To guarantee the correctness of the decomposition results of the cipher tensor, new methods need to be developed to address the challenges of non-homomorphic operations on cipher tensor, which can be depicted as follows:

Challenge 1: Non-Homomorphic Square Root Operation on Cipher Data. This challenge is to perform the operation $\beta_{j+1} = \|r_j\|_2$, which is responsible for obtaining the second norm of the vector r_j.

Challenge 2: Non-Homomorphic Division Operation on Cipher Data. The division operation is utilized to obtain the normalized orthogonal vectors using the following equation $\omega_{j+1} = {}^{r_j}/_{\beta_{j+1}}$.

5.2 Removing the Non-homomorphic Operations

Theorem 1: Lanczos-based Cipher Tensor Decomposition without Square Root Operation. Let T^E denote an N-order cipher tensor, S^E refer to cipher core tensor, $U_1^E,\ \ldots,\ U_N^E$ be the truncated orthogonal bases, then with the Lanczos method, the core data $core^E = \{S^E,\ U_i^E\}$ can be obtained without performing the non-homomorphic square root operation on the cipher data.

Proof. During the decomposition process, the square root operation only occurs during computing the second norm of a vector, namely, $\beta_{j+1} = \|r_j\|_2$. We loosen the orthogonal unitary matrix W to an orthogonal but non-unitary matrix to remove the square root operation. Let W be orthogonal vectors, $W^{\mathrm{T}}W = D$, where $D = diag(\delta_1,\ \delta_2,\ \ldots)$. Then multiplying the symmetric

matrix of the i-mode tensor unfolding with matrices W^{T} and W, we obtain

$$W^{\mathrm{T}}T^E_{(i)}(T^E_{(i)})^T W = \begin{bmatrix} \delta_1 & & & \\ & \delta_2 & & \\ & & \delta_3 & \\ & & & \ddots \end{bmatrix} \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ 1 & \alpha_2 & \beta_3 & & \\ & 1 & \alpha_3 & \ddots & \\ & & \ddots & \ddots & \end{bmatrix}. \tag{10}$$

Let $\beta_j = {}^{\delta_j}/_{\delta_{j-1}}$, Eq. (10) can be transformed to

$$W^{\mathrm{T}}T^E_{(i)}(T^E_{(i)})^T W = \begin{bmatrix} \alpha_1\delta_1 & \delta_2 & & & \\ \delta_2 & \alpha_2\delta_2 & \delta_3 & & \\ & \delta_3 & \alpha_3\delta_3 & \ddots & \\ & & \ddots & \ddots & \end{bmatrix}. \tag{11}$$

Left-multiplying the two parts of Eq. (11) with matrix W and selecting the j-th vector of the result matrix, we obtain the following equation

$$T^E_{(i)}(T^E_{(i)})^T w_j = \delta_j w_{j-1} + \alpha_j\delta_j w_j + \delta_{j+1}w_{j+1}. \tag{12}$$

As the vectors of matrix W are orthogonal, according to Eq. (12), all the parameters in the tridaigonal matrix can be computed as

$$\begin{aligned} &\alpha_j = w_j^{\mathrm{T}}T^E_{(i)}(T^E_{(i)})^T w_j \Big/ \delta_j, \\ &w_{j+1} = v_j - \alpha_j w_j, \\ &\delta_{j+1} = w_{j+1}^{\mathrm{T}} w_{j+1}, \\ &\beta_{j+1} = {}^{\delta_{j+1}}/_{\delta_j}, \\ &v_{j+1} = T^E_{(i)}(T^E_{(i)})^T w_j - \beta_{j+1}w_j. \end{aligned} \tag{13}$$

In the above procedures, parameter i denotes the i-mode unfolded matrix of the cipher tensor T^E, j refers to the j-th iteration of the Lanczos method. δ_j is a non-zero element prematurely. Based on the tridiagonalized matrix, we can compute the left singular matrix U_i^E of the cipher tensor unfolding $T^E_{(i)}$. Therefore, the core tensor S^E can be obtained with equation $S^E = T^E \times_1 (U_1^E)^{\mathrm{T}} \times_2 (U_2^E)^{\mathrm{T}} \ldots \times_N (U_N^E)^{\mathrm{T}}$. In Eq. (13), the non-homomorphic square root operation is removed.

Note that the parameters $\alpha,\ \beta,\ \delta,\ w,\ v$ are all in ciphertext format. For convenance, the superscripts are omitted during the proof procedure in this paper. The division operation is transferred to client. In each Lanczos iteration, the cloud send $w_j^{\mathrm{T}}T^E_{(i)}(T^E_{(i)})^T w_j$, δ_{j+1}, δ_j to client where the results of the divisions are computed and passed back in ciphertext format.

5.3 Secure Tensor Decomposition Algorithm

In this paper, Algorithm 2 is presented for secure tensor decomposition. The numbers defined in real fields in raw data are multiplied with 10^k to obtain the corresponding integers. Hence, all the operations are defined in the integer field. The non-homomorphic operations, square root and division, are replaced.

In Line 1, the unified cipher tensor T^E is unfolded to N matrices which are transformed to symmetric matrices $T^E_{(i)}(T^E_{(i)})^{\mathrm{T}}$. The left singular vector matrix of $T^E_{(i)}$ is equal to the eigen vector matrix of $T^E_{(i)}(T^E_{(i)})^{\mathrm{T}}$. From Line 3 to Line 14, Algorithm 2 computes the truncated orthogonal bases using the Lanczos method. The non-homomorphic operation, square root, is removed during the iterations, and the division challenge is addressed by transferring the operations to client in Line 5 and Line 9. The tridiagonal matrix L is obtained in Line 12, which is utilized for eigen value decomposition. The truncated orthogonal bases are computed in Line 13 as well as the cipher core tensor is obtained in Line 14.

Algorithm 2 Secure Tensor Decomposition on Cloud $\{S^E,\ U_1^E,\ \ldots,\ U_N^E\} = f_d(T^E)$.

Input:
The reduced cipher tensor T^E.
Output:
The cipher core tensor S^E and cipher truncated orthogonal bases $U_1^E,\ \ldots,\ U_N^E$.

1: Unfold the cipher tensor to matrices and obtain the corresponding symmetric matrices $T^E_{(i)}(T^E_{(i)})^{\mathrm{T}}$.
2: **for** each matrix $T^E_{(i)}(T^E_{(i)})^{\mathrm{T}},\ 1 \le i \le N$ **do**
3: Initialize the parameters by setting $j = 1$, $w_j = random\ vector$, $\delta_j = w_j^{\mathrm{T}}w_j$, $\beta_1 = 1$, $v_j = T^E_{(i)}(T^E_{(i)})^{\mathrm{T}}w_j$.
4: **while** $\delta_j \neq 0$ **do**
5: Compute $w_j^{\mathrm{T}}T^E_{(i)}(T^E_{(i)})^{\mathrm{T}}w_j$ and obtain the parameter α_j by receiving the division result computed in client.
6: Compute vector $w_{j+1} = v_j - \alpha_j w_j$.
7: Increase j by 1, $j+ = 1$.
8: Replace δ_j with $w_j^{\mathrm{T}}w_j$.
9: Send δ_{j+1} and δ_j to client and receive the division result β_j.
10: Compute vector $v_j = T^E_{(i)}(T^E_{(i)})^{\mathrm{T}}w_j - \beta_j w_{j-1}$.
11: **end while**
12: Construct the tridiagonal matrix L using the obtained $\alpha_j,\ \beta_j$, and compute the eigen pair.
13: Compute the left singular vector matrix and obtain the truncated orthogonal basis U_i^E.
14: Obtain the cipher core tensor S^E using the equation $S^E = T^E \times_1 (U_1^E)^{\mathrm{T}} \times_2 (U_2^E)^{\mathrm{T}} \ldots \times_N (U_N^E)^{\mathrm{T}}$
15: Return tensor S^E and the bases $U_1^E,\ \ldots,\ U_N^E$.
16: **end for**

5.4 Algorithm Analysis

The performance of the proposed secure tensor decomposition algorithm is theoretically analyzed in this paper in terms of time complexity, memory usage, decomposition accuracy and data security.

Time Complexity: Execution time of the proposed secure decomposition algorithm consists of matrix unfolding, Lanczos-based singular value decomposition of each unfolded matrices, and product of a tensor by the truncated bases. Let $Time_{unf}$, $Time_{lan}$ and $Time_{prod}$ denote the time used by the above processes respectively,

then the total time consumption $Time$ can be computed with the following equation

$$Time = Time_{unf} + Time_{lan} + Time_{prod}. \quad (14)$$

Tensor unfolding is a simple transformation with $O(1)$ time complexity. $Time_{lan}$ is equal to $Time_1 + Time_2 + ... + Time_N = \sum_{i=1}^{N} Time_i$, where $Time_i$ refers to the decomposition time consumed by unfolded matrix $T^E_{(i)}$. The time complexity of decomposing one cipher symmetric matrix generated from a tensor unfolding is $O(n^2)$. For a truncated orthogonal basis U with k column vectors, time complexity of the product of a tensor by a matrix is $O(kn)$, where k is the number of vectors in the truncated basis U^E_i. Hence, to decompose a N-order cipher tensor T^E with N unfolded matrices, the time complexity of the proposed secure tensor decomposition algorithm is $O(1) + O(Nkn) + O(Nn^2)$, namely $O(Nn^2)$ in general, where N refers to the number of tensor orders, n denotes the dimensionality of the tensor unfolding.

Memory Usage. The memory usage of the secure decomposition algorithm is related to the number of non-Nil elements in the ciper tensor T^E. Let m^i_{nz} denote the number of rows that contain nonzero elements in the symmetric matrix $Sym(T^E_{(i)})$, then the total memory usage is equal to $2N \times nnz(T^E) + \sum_{i=1}^{N} m^i_{nz}$. Based on the Compressed Row Storage (CRS) scheme described in Section 4.3, the proposed secure tensor decomposition algorithm can significantly save computer memory.

Decomposition Accuracy. The reconstruction error between the initial tensor and the obtained approximate tensor can be computed using the Frobenius Norm [11] which is defined as

$$\left\| T - \hat{T} \right\|_F = \left(\sum_{i_1=1}^{I_1}, ..., \sum_{i_p=1}^{I_P} (a_{i_1,...,i_p} - \hat{a}_{i_1,...,i_p})^2 \right)^{\frac{1}{2}}. \quad (15)$$

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, this paper presents two ratios. The reduction ratio is defined as $\rho = \frac{nnz(S) + \sum_{i=1}^{N} nnz(U_i)}{nnz(T)}$, where S denotes the core tensor, and U_i is the i-mode truncated orthogonal basis. The core data set of tensor T consists of the core tensor S and truncated bases $U_1, U_2, ..., U_N$. As only nonzero elements of the core data set are stored, ratio ρ can accurately reflect the dimensionality reduction degree. The approximation ratio is $\vartheta = \frac{\left\| T - \hat{T} \right\|_F}{\|T\|_F}$, which reflects the degree of reconstruction error with tensor Frobenius Norm. In this paper, the pair (ρ, ϑ) is employed to describe the dimensionality reduction degree and reconstruction error degree. Obviously, the ratio ρ is inversely proportional to ratio ϑ.

Data Security. The fully homomorphic encryption scheme employed in the proposed secure tensor decomposition algorithm is based on the assumption of Ring Learning with Errors (RLWE) [12]. The assumption is parameterized by the polynomial $f(x) \in \mathcal{Z}[x]$ of degree d, a prime integer $q \in \mathcal{Z}$, as well as an error distribution χ over $\mathcal{Z}$ [4]. According to this assumption, given any polynomial number of samples of the form $a_i,\ c_i = a_i \cdot sk + e_i,\ a_i \in \mathcal{R}_q,\ e_i \in \chi$, it is very difficult to compute c_i in polynomial time. It is equivalent to a variant of which the noise e_i are multiples of some integers that are relatively prime to the modulus q.

6 Performance Evaluation

This section illustrates some evaluation results of the presented secure tensor decomposition approach. We performed the experiments on commodity computers, each of them is of Intel(R) Core(TM) $i5-3470$ CPU @3.20 GHZ, 8 GB RAM, and is running CentOS 6.4 Operating System. We adopted the software library which implements the BGV fully homomorphic encryption scheme. The NTL-6.2.1 mathematical library was compiled and installed in the experimental machines. The experimental data are from our university including the unstructured video data collected with fixed cameras, semi-structured XML documents about staffs and students in our university, and structured trajectory data collected by mobile phones. All the various types of data are encrypted and integrated as a unified cipher tensor model for secure decomposition. We have implemented a number of secure algorithms on cipher data including singular value decomposition, eigen value decomposition, tensor construction, but due to space constraints, we only present some representative results.

6.1 Operation Performance on Cipher and Plain Data

We performed the addition and multiplication operations on the cipher tensor and plain tensor constructed from unstructured video data and semi-structured XML documents. We set value 999983 and 1 to integer p and r respectively for the encryption process. The experiments were carried out many times and the average results are demonstrated in Fig. 6. We also normalized the data size and execution time for convenient comparison. The operation in the experiment was matrix-vector product which is frequently called during the Lanczos iteration. The number of addition operations was about equal to that of multiplication operations.

In Fig. 6, the normalized execution time of the operations performed on plain data increase generally. It increased by 0.12 from the normalized data size 0.4 to 1. For the cipher data, there is a considerable increase occurred from the normalized data size 0.2. The normalized execution time increased by 0.53 from the normalized data size 0.4 to 1, which is about 4.42 times that of the plain data. The experimental results show that the BGV fully homomorphic encryption scheme is lack

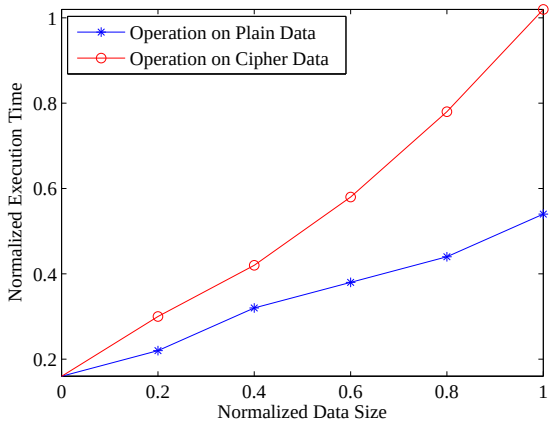


Fig. 6. The operation performance of matrix-vector product carried out on cipher and plain data.

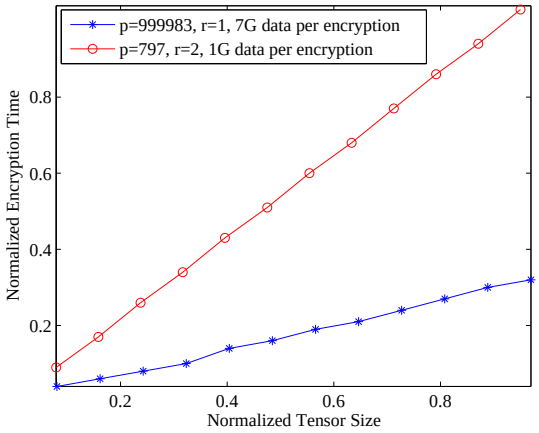


Fig. 7. The time used for construction of cipher tensors.

of performance. This is caused by the noise expansion during the multiplication operations which need more time to tackle them. The problem has been investigated in Refs [3, 7, 8].

As the addition and multiplication operations are carried out on the non-zero elements of the cipher tensor and plain tensor, the sparsity is one of the critical factor that can greatly influence the execution performance. In many fields, the constructed unified tensor model contains large amount of zero elements, this can decrease the execution time. In addition, the multiplication operations may cause more noise than the addition operation and subtraction operation, therefore in practice, the big data applications can try to employ more addition and subtraction than multiplication.

6.2 Performance of Cipher Tensor Construction

We evaluated the performance of constructing high-order cipher tensors using the collected heterogenous data. Two types of experiments are carried out, of which the first employed the parameters $p = 999983$, $r = 1$ as well as the second used $p = 797$, $r = 2$. According to the software library of BGV homomorphic encryption scheme, the parameter r was set with value 1 for ordinary homomorphic computation, while 2 for bootstrapping [13]. The plain space in the implemented *PAlgebraMod* class of the software has the form $A_{p^r} = A/p^r A$. As the plaintext slots are different according to the polynomials degree, we utilized different sizes of data during the encryption experiments. The data are partitioned to $1G$ and $7G$ chunks respectively for cipher tensor construction, and the *EncryptedArry* class was called to perform the encryption process.

The graph presented in Fig. 7 shows that the normalized encryption time increases linearly as the normalized tensor size increase. The time consumption in the experiment with parameters $p = 797$ and $r = 2$ is about threefold of that in experiment with parameters $p = 999983$ and $r = 1$. This reveals that the adjustments of security parameters as well as the conversion to new encryption procedures are time-consuming.

6.3 Effects of Dimensionality Reduction

To evaluate the effects of dimensionality reduction of secure tensor decomposition, we utilized a three-order tensor formed by gray video clips, which is of MPEG4 format, 15 frames per second. The tensor was unfolded to three matrices, which were transformed to symmetric matrices and then factorized using the Lanczos method. We adopted different truncation ratios to preserve the left singular vector spaces which contain the unitary orthogonal vectors of the the tensor unfolding. This section separately demonstrates some experimental results of the singular values, orthogonal bases, core tensor, and decomposition ratio that includes dimensionality reduction ratio and tensor approximation ratio.

6.3.1 *Singular Values of Unfolded matrices*

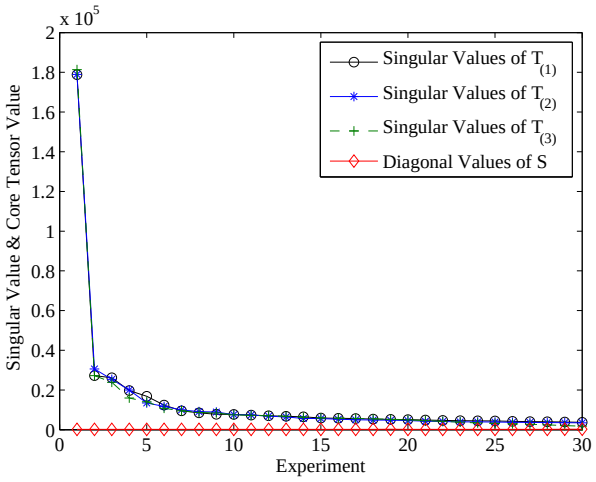
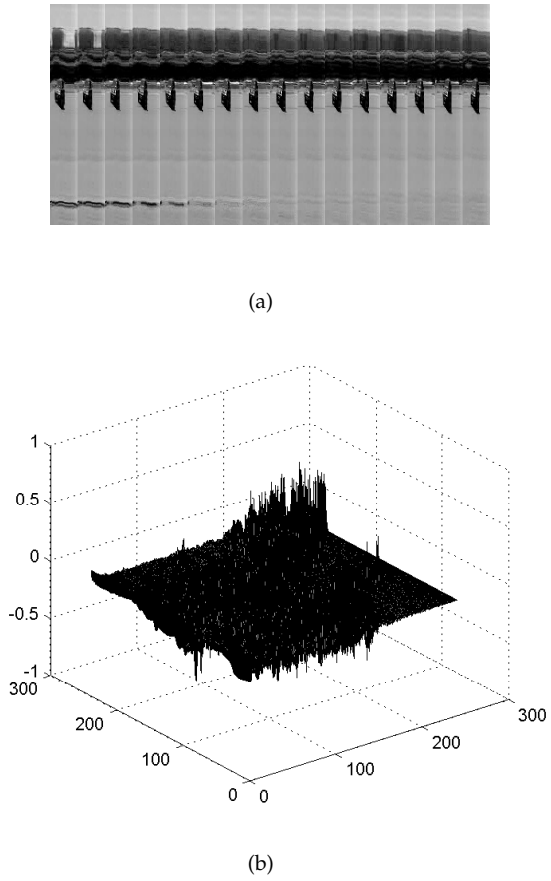


Fig. 8. The singular values of unfolded matrices and the diagonal values of the core tensor.

Fig. 8 demonstrates the singular values of the three unfolded matrices $T_{(1)}$, $T_{(2)}$, and $T_{(3)}$. We also drew the super-diagonal values of the core tensor in the figure for comparison. The graph shows that the first singular values of the three tensor unfolding are generally greater than the others. In our experiments, the first singular values are 6.559, 5.856, and 6.652 times of the second singular values of the tensor unfolding respectively. In addition, there is obvious declining trend from the second singular value to the eight singular value. From the ninth singular value, the rate of decrease slows down. The scaling ratios of the first singular values to the thirtieth singular values are 48.00, 51.73, and 103.35 respectively. Compared to the singular values, the diagonal values of the core tensor are so small that they are located at the bottom of the graph.

6.3.2 Unfolded Matrix and Truncated Orthogonal Basis



(a)

(b)

Fig. 9. (a) The 1-mode unfolded matrix. (b) The left singular matrix.

Fig. 9 shows an example of the 1-mode tensor unfolding $T_{(1)}$ and the truncated left singular vector matrix U_1. The number of rows in matrix $T_{(1)}$ is equal to the dimensionality of order I_1, as well as the number of columns is equal to $I_2 \times I_3$. The singular vector matrices are composed of unitary orthogonal vectors, the elements of the orthogonal vectors are normalized which are between -1 and 1. The maximum elements of matrix U_1 is 0.53, as well as the minimum value is -0.73. The elements that between $(-0.2,\ 0.2)$ are account for 98.54 percent. About 33.96 percent of the elements are from $(-0.01,\ 0.01)$.

6.3.3 Matrices of The Core Tensor

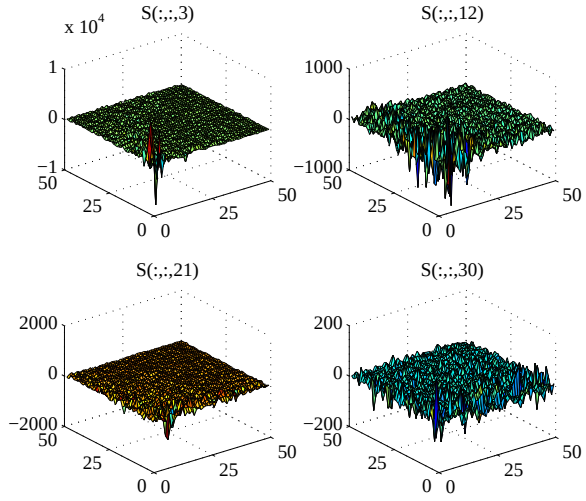


Fig. 10. Illustration of four matrices of the core tensor.

The projection coordinates are contained in the core tensor which has the same number of orders as the initial tensor. We illustrate the structure of the core tensor with some representative matrices. Four matrices of the core tensor are demonstrated in Fig. 10. The matrix $S(:,:,3)$ has more larger elements than matrix $S(:,:,30)$. The maximum element of matrix $S(:,:,3)$ and $S(:,:,30)$ are 6286.07 and 156.27 respectively. The elements in matrix $S(:,:,12)$ are between -936.40 and 893.84, as well elements in matrix $S(:,:,21)$ are between -1085.67 and 539.93. In this experiment, the mean values of the four matrices are -0.87, 0.41, -0.69, and 0.65 respectively.

6.3.4 Reduction Ratio and Reconstruction Ratio

We decomposed the unified tensor model to core tensor and truncated orthogonal bases. The dimensionality reduction ratio and approximation ratio which is equal to the subtraction of the reconstruction error ratio from 100%, are utilized for evaluation. From the first experiment to the thirty-fifth experiment, the dimensionality reduction ratio increases from 0.28% to 78.19% in Fig. 11, as well as the tensor approximation ratio increases slowly from 79.21% to 98.56%. In the fourteenth experiment, 14.72% core data can guarantee 92.66% approximation accuracy. In the eighteenth experiment, 23.41% core data can guarantee 94.20% approximation accuracy. The line graph of dimensionality reduction ratio in Fig. 11 increases sharply than the tensor approximation ratio. Averagely, about 21% core data can guarantee 94% approximation accuracy.

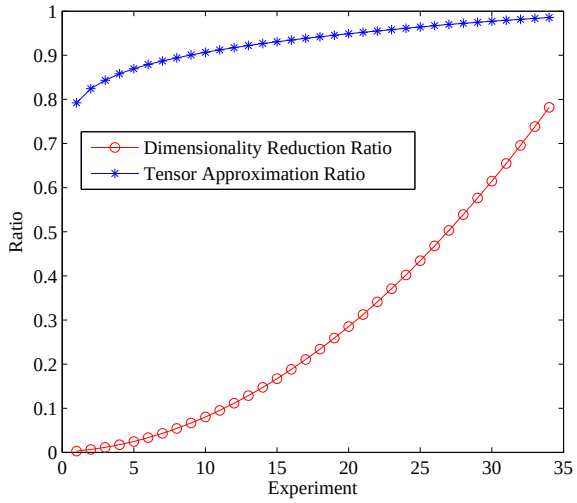


Fig. 11. The relationship between the dimensionality reduction ratio and the tensor reconstruction ratio.

7 Related Work

This section reviews some previous studies on tensor decomposition, fully homomorphic encryption scheme and Lanczos method..

7.1 Tensor and HO-SVD

A tensor model is used to illustrate the linear relations between the scalars, vectors, and other tensors. Tensor [5, 14] is a generalization of a matrix model, which is usually called multidimensional array. It can effectively represent the heterogeneous data as a concise model with which the valuable information can be extracted using the High Order Singular Value Decomposition (HO-SVD) [15] method. As the HO-SVD method imposes orthogonal constraint on the truncated vector bases, it is considered as a special case of the commonly used TUCKER [16] decomposition. HO-SVD has been adopted for data analysis and mining in many fields such as tag recommendations [17] and hand-written digit classification [18].

7.2 Fully Homomorphic Encryption

The concept of fully homomorphic is first introduced in 1978 [19]. The encryption schemes reported in Refs. [20–24] support either addition homomorphism or multiplication homomorphism. However, none of them can support both operations in a single scheme. A new approach is presented in Ref. [25] which constructs a scheme capable of carry out both addition and multiplication operations, it handles an arbitrary number of additions but one multiplication. In 1999, Gentry [3] constructed a fully homomorphic encryption scheme (FHE) which can evaluate an arbitrary number of additions and multiplications on the encrypted data. From then on many works [4, 7, 8, 26, 27] have been carried out in order to present new efficient fully homomorphic encryption schemes.

7.3 Lanczos Method

The Lanczos method [28], an adaptation of power methods, is efficient for finding several extreme eigenvectors and eigenvalues of a large scale sparse symmetric matrix. In Ref. [29] a block Lanczos type algorithm is introduced to compute the tridiagonal matrix. Parallel implementation of Lanczos algorithms [30, 31] are studied to improve the efficiency. Those algorithms aim to effectively parallelize the matrix-vector or vector-vector operations. Ref. [32] reports a new algorithm that can remove the square root operation from the Lanczos iteration. An implicitly restarted method is explored in Ref. [33] for obtaining the smallest singular triplets. In Ref. [34], a new error bound for Lanczos method is introduced.

Many studies on tensor decomposition, fully homomorphic encryption and Lanczos method have been performed over the past few decades. However, all the investigations mentioned above are concentrated on special topics, no systematic research has been dedicated to secure tensor decomposition of big data on cloud. The present study was undertaken to propose a holistic approach to process the large scale heterogeneous data on cloud while protect the privacy of user data.

8 Conclusion

Aiming to propose a new computing approach that can securely process big data on cloud, this paper formalizes the secure tensor decomposition problem, and proposes a holistic solution framework to address it. A unified cipher tensor model is presented to integrate all the encrypted sub-tensors as a unified model, concise examples are provided for illustrating the process of cipher tensor construction and unfolding. A Lanczos-based secure tensor decomposition algorithm is introduced, in which the non-homomorphic square root operation is replaced. Theoretical analyses in terms time complexity, memory usage, decomposition accuracy, and data security are provide. Experiments are carried out to evaluate the performance of the presented methods. The results support that the proposed approach is efficient and can pave a way for secure processing of big data on cloud.

References

[1] P. Mell and T. Grance, "The NIST Definition of Cloud Computing (Draft)," *NIST Special Publication*, vol. 800, no. 145, p. 7, 2011.

[2] M. Kantardzic, *Data Mining: Concepts, Models, Methods, and Algorithms*. John Wiley & Sons, 2011.

[3] C. Gentry, "Fully Homomorphic Encryption Using Ideal Lattices," in *STOC*, vol. 9, 2009, pp. 169–178.

[4] K. Lauter, M. Naehrig, and V. Vaikuntanathan, "Can Homomorphic Encryption be Practical?" Cryptology ePrint Archive, Report 405, 2011, http://eprint.iacr.org/2011/.

[5] T. G. Kolda and B. W. Bader, "Tensor Decompositions and Applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.

[6] K. Liwei, H. Fei, L. T. Yang, L. Man, L. Changqing, and M. Geyong, "A Tensor-Based Approach for Big Data Representation and Dimensionality Reduction," *IEEE Transactions on Emerging Topics in Computing, (in Press, DOI 10.1109/TETC.2014.2330516)*.

[7] M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully Homomorphic Encryption Over the Integers," in *Advances in Cryptology–EUROCRYPT 2010*. Springer, 2010, pp. 24–43.

[8] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "Fully Homomorphic Encryption Without Bootstrapping," in *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. ACM, 2012, pp. 309–325.

[9] M. Grüning, A. Marini, and X. Gonze, "Implementation and Testing of Lanczos-based Algorithms for Random-Phase Approximation Eigenproblems," *Computational Materials Science*, vol. 50, no. 7, pp. 2148–2156, 2011.

[10] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. SIAM, 2000, vol. 11.

[11] C. Meyer, *Matrix Analysis and Applied Linear Algebra Book and Solutions Manual*. SIAM, 2000, vol. 2.

[12] V. Lyubashevsky, C. Peikert, and O. Regev, "On Ideal Lattices and Learning with Errors Over Rings," *Journal of the ACM (JACM)*, vol. 60, no. 6, p. 43, 2013.

[13] C. Gentry, S. Halevi, and N. P. Smart, "Better Bootstrapping in Fully Homomorphic Encryption," in *Public Key Cryptography–PKC 2012*. Springer, 2012, pp. 1–16.

[14] C. M. Martin, "Tensor Decompositions Workshop Discussion Notes," *American Institute of Mathematics*, 2004.

[15] 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.

[16] L. R. Tucker, "Some Mathematical Notes on Three-mode Factor Analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.

[17] 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.

[18] B. Savas and L. Eldén, "Handwritten Digit Classification Using Higher Order Singular Value Decomposition," *Pattern Recognition*, vol. 40, no. 3, pp. 993–1003, 2007.

[19] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On Data Banks and Privacy Homomorphisms," *Foundations of Secure Computation*, vol. 4, no. 11, pp. 169–180, 1978.

[20] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-key Cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[21] S. Goldwasser and S. Micali, "Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Information," in *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*. ACM, 1982, pp. 365–377.

[22] T. ElGamal, "A Public Key Cryptosystem and A Signature Scheme Based on Discrete Logarithms," in *Advances in Cryptology*. Springer, 1985, pp. 10–18.

[23] J. D. Cohen and M. J. Fischer, "A Robust and Verifiable Cryptographically Secure Election Scheme," in *FOCS*, vol. 85, 1985, pp. 372–382.

[24] P. Paillier, "Public-key Cryptosystems Based on Composite Degree Residuosity Classes," in *Advances in Cryptology EUROCRYPT99*. Springer, 1999, pp. 223–238.

[25] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-DNF Formulas on Ciphertexts," in *Theory of Cryptography*. Springer, 2005, pp. 325–341.

[26] C. Gentry and S. Halevi, "Fully Homomorphic Encryption Without Squashing Using Depth-3 Arithmetic Circuits," in *IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS), 2011*. IEEE, 2011, pp. 107–109.

[27] N. P. Smart and F. Vercauteren, "Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes," in *Public Key Cryptography–PKC 2010*. Springer, 2010, pp. 420–443.

[28] J. K. Cullum and R. A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations: Vol. 1: Theory*. SIAM, 2002, vol. 41.

[29] I. Popovyan, "Efficient Parallelization of Lanczos Type Algorithms," Tech. Rep., 2011.

[30] I. Flesch and R. Bisseling, "A New Parallel Approach to the Block Lanczos Algorithm for Finding Nullspaces Over GF (2)," *Master's Thesis, Department of Mathematics, Utrecht University, Utrecht, the Netherlands*, 2002.

[31] P. L. Montgomery, "Distributed Linear Algebra," in *Proceedings, 4th Workshop on Elliptic Curve Cryptography*, 2000.

[32] R. J. Lambert, *Computational Aspects of Discrete Logarithms*. Citeseer, 1997.

[33] Z. Jia and D. Niu, "A Refined Harmonic Lanczos Bidiagonalization Method and An Implicitly Restarted Algorithm for Computing the Smallest Singular Triplets of Large Matrices," *SIAM Journal on Scientific Computing*, vol. 32, no. 2, pp. 714–744, 2010.

[34] Q. Ye, "Error Bounds for the Lanczos Methods for Approximating Matrix Exponentials," *SIAM Journal on Numerical Analysis*, vol. 51, no. 1, pp. 68–87, 2013.

Liwei Kuang is currently studying for the PhD degree in School of Computer Science and Technology at Huazhong University of Science and Technology, Wuhan, China. He received the master's degree in School of Computer Science from Hubei University of Technology, Wuhan, China, in 2004. From 2004 to 2012, he was a Research Engineer with FiberHome Technologies Group, Wuhan, China. His research interests include big data, pervasive computing and cloud computing.

Laurence T. Yang received the B.E. degree in Computer Science and Technology from Tsinghua University, China and the PhD degree in Computer Science from University of Victoria, Canada. He is a professor in the School of Computer Science and Technology at Huazhong University of Science and Technology, China, and in the Department of Computer Science, St. Francis Xavier University, Canada. His research interests include parallel and distributed computing, embedded and ubiquitous/pervasive computing, and big data. His research has been supported by the National Sciences and Engineering Research Council, and the Canada Foundation for Innovation.

Jun Feng is currently studying for the PhD degree in School of Computer Science and Technology at Huazhong University of Science and Technology, Wuhan, China. He received the master's degree from the School of Computer Science and Information, Guizhou University, Guiyang, China, in 2013. His research interests include big data, pervasive computing and information security.

Mianxiong Dong received his B.Sc. degree in Computer Science (2009) from the University of Aizu (Japan). His research interests include networking and wireless security.