J Supercomput (2018) 74:592–614
https://doi.org/10.1007/s11227-017-2163-y

TDRM: tensor-based data representation and mining for healthcare data in cloud computing environments

Rajinder Sandhu[1,3] · **Navroop Kaur**[2] · **Sandeep K. Sood**[2] · **Rajkumar Buyya**[1]

Published online: 24 October 2017

Abstract Big data analytics proved to be one of the most influential forces in today's competitive business environments due to its ability to generate new insights by processing a large volume and variety of data. Storing as well as mining these datasets is one of the primary challenges of the big data era. If data are stored in a well-defined pattern, then its updation mining and deletion processes become easy. In this paper, granular computing concept is used to store heterogeneous data in the format of tensor. A multi-dimensional matrix, also known as tensor, stores data in the raw format, and then, raw tensor is replicated to multiple tensors of different abstraction levels based on concept hierarchy of each attribute. Mathematical foundation of tensor formation and query processing are developed. The proposed method is successful in creating tensors of a diabetes dataset proving its applicability. The proposed system provides faster computation, low response time, better privacy and high relevancy as compared to baseline PARAFAC2 and CANDELINC tensor analysis method when

✉ Rajinder Sandhu
rajsandhu1989@gmail.com

Navroop Kaur
navonline89@gmail.com

Sandeep K. Sood
san1198@gmail.com

Rajkumar Buyya
rbuyya@unimelb.edu.au

[1] Cloud Computing and Distributed Systems (CLOUDS) Laboratory, School of Computing and Information Systems, The University of Melbourne, Parkville, Australia

[2] Department of Computer Science and Engineering, Guru Nanak Dev University, Gurdaspur, Punjab, India

[3] Department of Computer Science and Engineering, Chitkara University Institute of Engineering and Technology, Chitkara University, Patiala, Punjab, India

run on Microsoft Azure cloud infrastructure. Different levels of information granules in the form of tensors make data storage and its query processing effective.

Keywords Cloud computing · Big data · Granular computing · Tensor · Healthcare data · PARAFAC2 · Data mining

1 Introduction

Data generated by today's Internet-based applications in business environments is voluminous, complex and contains a large variety of formats. Big Data Analytics (BDA) is useful in taming such heterogeneous data so that effective dependencies can be recognized [1]. BDA has proved to be the most influential force in today's competitive business environment even though it is still in its maturing phase. It helps business organizations in understanding customers' needs in a more effective manner and generate large revenue. Mining healthcare data is no different. Many useful predictions and relationships can be derived from raw healthcare data [2]. For example, it was assumed that the success rate of 'tamoxifen,' a cancer treatment medicine, is around 80%. But, the big healthcare data analysis revealed that 'tamoxifen' is effective 100% in 80% of cancer patients and completely ineffective in rest 20% [3]. Such important results can be easily derived using BDA on healthcare data. Hence, effective medical data mining is an important research direction.

In this direction, healthcare data mining poses three major challenges: heterogeneity, privacy and abstraction. Heterogeneity of medical data emerges from the fact that the data are generated from various sources such as health sensors, scan images, ECG, medical reports and prescribed medications. Such heterogeneous data must be stored in a manageable form such that it can be effectively processed. Moreover, since medical data contain crucial personal information, it must be protected from privacy violations in order to meet the second challenge. The third challenge is abstraction. Each stakeholder in any business/healthcare organization requires simplified and effective information that is relevant only to their respective thinking space so that they can derive efficient decisions. For example, government healthcare agencies are concerned with high-level information; hospitals require mining about their required infrastructure, while individual users need predictions for their personal health. Therefore, different healthcare agencies require different levels of abstraction from the available healthcare data. In other words, results required from healthcare data mining are of multiple granularities.

Granularity is 'scale or level of detail in a set of data' [4], and it provides a level of abstraction to the underlying details so that highlighted focus can be applied to important or required attributes. Granular computing (GrC) [5] is an art of extracting information at different, yet finite level of abstractions such that each computing problem is solved at different level of granularity by removing irrelevant details from the available datasets. It uses the concepts of many existing theories such as fuzzy set, set theory, rough sets, and divides & conquer to solve such problems. Furthermore, Yao stated that GrC is inspired from three basic practical requirements [5]: simplicity, low cost clarity and high tolerance for uncertainty; and serves four main objectives

[6]: (a) effective representation of real-world problems, (b) concurrency with human problem-solving techniques, (c) simplification of input as well as output, and (d) cost-effectiveness of the whole process. By careful observation of healthcare data, it is observed that all the above practical requirements and objectives are applicable in healthcare data mining too. Consequently, GrC is found to be an effective technique for medical data mining. In addition, it is found that tensors can be effectively used to store and process big data [7,8].

In order to address all the three challenges for big medical data mining, a tensor- and GrC-based big data mining approach for healthcare data is proposed. The aim of the study conducted in this paper is to design a collaborative framework. The proposed framework contains mathematical foundation of GrC and BDA in the above-stated three tasks. A comparison with PARAFAC2 tensor decomposition technique is also provided using Microsoft Azure cloud computing infrastructure.

The rest of the paper is organized as follows: Section 2 discusses the related work on use of granular computing in healthcare and big data. Section 3 provides preliminaries for tensors. Section 4 evaluates the proposed framework on its theoretical aspect. Section 5 provides experimental evaluation and comparison of proposed method with PARAFAC2 and CANDELINC tensor analysis method. Section 6 concludes the paper with discussion on future work.

2 Related work

Health BDA is gaining momentum due to increased amount of data available in healthcare sector. Research projects and researchers are turning their attention toward extraction of useful information and knowledge from big data in order to furnish better medical facilities. The authors in [9] proposed a method for improving the innovation in medical devices. The proposed method used BDA and regulatory sciences to foster new innovations. On the other hand, the authors in [10] proposed a framework which allows users to avail home-diagnosis services by using big data mining. In addition, BDA finds other interesting applications in health care as discussed by authors in [11–15].

Big data mining operation can be performed using various techniques. The authors in [16] conducted a survey on state-of-art tools and technologies for big data processing. They found that the techniques such as cloud computing, granular computing and quantum computing could be used effectively for harnessing big data capabilities. Cloud computing and quantum computing provide powerful platforms for big data processing, while granular computing captures the essence of big data processing by taming the size of big data with the use of granules such as clusters, intervals and subsets. The details of big data and its mathematical models can be studied in [4,17–22].

Granular computing (GrC) has been extensively used by various authors in different research domains such as soft computing [23], concept learning [24–26], signal processing [27], clustering [28,29], regression [30,31], surrogate modeling [32], ontology [33], and in similarity algorithms [34]. Other than these, GrC has also been used in data mining and information processing by few authors. Specifically, the author in

[35] listed the importance of granular computing in data mining by discussing various issues related to data mining using GrC and concluded that GrC can be very efficiently used for extracting useful information from the given data set(s). Later, the authors in [36] used GrC for data mining and knowledge reduction for larger data sets. In 2006, the authors in [37] provided a detailed theory for application of GrC for information processing. They also provided a practical example of traffic delays using the mentioned theory.

As far as big data processing using GrC is concerned, a few works have explored it. The work in [38] used GrC for Social Network Analysis (SNA) and found that there is a natural connection between GrC and graph theory which makes GrC suitable for SNA. In 2015, Qian et al. [39] turned their attention toward attribute reduction in big data using GrC. The results showed that GrC can be efficiently used for big data processing.

In addition to the above-mentioned work, the work in [40] explored privacy protection of medical data using GrC and emphasized that different levels of granularity can be used to efficiently process medical data without violating the privacy requirements. Furthermore, the successful application of GrC on tensors has been discussed in [41]. But, to the best of our knowledge, none of the existing works have addressed the problem of mining big medical data using various abstraction levels and query processing based on these abstractions. The inherent property of GrC to provide different abstraction levels using granulation avoids extra efforts for abstraction management in healthcare data mining for various healthcare agencies. This paper focuses on theoretical foundations of an effective and novel framework for storing big healthcare data using tensors and mining it using GrC.

3 Background-tensor preliminaries

A tensor is an N-way array of data or objects. A scalar is a zeroth-order tensor; vector is a first-order tensor; and a matrix is a second-order tensor. Formally, an Nth-order tensor can be defined as:

$$\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N};$$

where the size of ith dimension of $\mathcal{X}$ is represented by I_i. The two-dimensional sections of a higher-order tensor are called slices, while the higher-order counterparts of matrix columns and rows are called fibers of a tensor [42].

Figure 1a shows a third-order tensor $\mathcal{X}$. When $\mathcal{X}$ is sliced along its 1st dimension, horizontal slices of $\mathcal{X}$ are formed. Similarly, slicing $\mathcal{X}$ along 2nd and 3rd dimension forms lateral and frontal slices, respectively. Figure 1b shows the frontal slices of X. The number of frontal slices of $\mathcal{X}$ is equal to I_3. If X_i denotes the ith frontal slice of $\mathcal{X}$, then $\mathcal{X}$ can be written in terms of its frontal slices as:

$$\mathcal{X} = [X_1 X_2 \ldots\ldots X_{I_3}]; \tag{1}$$

If each X_i is further sliced laterally, then column fibers of $\mathcal{X}$ are obtained as shown in Fig. 1c. The number of lateral slices of each X_i is equal to I_2. Let x_{ij} denotes the jth lateral slice of X_i, then

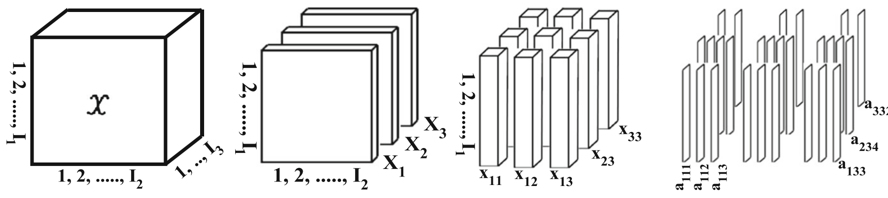


Fig. 1 **a** Third-order tensor $\mathcal{X}$, **b** frontal slices of $\mathcal{X}$, **c** lateral slices of each frontal slice (column fibers of $\mathcal{X}$), **d** lateral slices of each column fiber of $\mathcal{X}$

$$X_i = [x_{i1}x_{i2}\ldots\ldots x_{iI_2}]; \quad \text{where } 1 \le i \le I_3; \tag{2}$$

In addition, if x_{ij} is further sliced laterally, then each attribute inside the fiber is separated as shown in Fig. 1d. The number of lateral slices formed from each x_{ij} depends upon the number of attributes in it. Let n denotes the number of attributes in x_{ij} and a_{ijk} denote the kth slice of x_{ij}, then

$$x_{ij} = [a_{ij1}a_{ij2}\ldots a_{ijn}]; \quad \text{where } 1 \le i \le I_3;\ 1 \le j \le I_2; \tag{3}$$

In particular, $\mathcal{X}$ can be expressed using Eq. (4)

$$\begin{aligned}\mathcal{X} &= [X_1X_2\ldots\ldots.X_{I_3}]\\ &= [[x_{11}x_{12}\ldots\ldots.x_{1I_2}]\,[x_{21}x_{22}\ldots\ldots.x_{2I_2}]\ldots\ldots\ldots\ldots\ldots[x_{I_31}x_{I_32}\ldots\ldots x_{I_3I_2}]]\\ &= \begin{bmatrix} [[a_{111}a_{112}\ldots\ldots a_{11n_1}]\,[a_{121}a_{122}\ldots\ldots a_{12n_1}]\ldots\ldots[a_{1I_21}a_{1I_22}\ldots\ldots a_{1I_2n_1}]]\\ [[a_{211}a_{212}\ldots\ldots a_{21n_2}]\,[a_{221}a_{222}\ldots\ldots a_{22n_2}]\ldots\ldots[a_{2I_21}a_{2I_22}\ldots\ldots a_{2I_2n_2}]]\ldots\\ \left[\left[a_{I_311}a_{I_312}\ldots\ldots a_{I_31n_{I_3}}\right]\left[a_{I_321}a_{I_322}\ldots\ldots a_{I_32n_{I_3}}\right]\ldots\ldots\left[a_{I_3I_21}a_{I_3I_22}\ldots\ldots a_{I_3I_2n_{I_3}}\right]\right]\end{bmatrix}\end{aligned} \tag{4}$$

4 Tensor-based Data Representation and Mining (TDRM)

The proposed tensor-based data representation and mining (TDRM) framework is divided into three phases for effective collection and analysis of medical data as shown in Fig. 2. In the first phase, raw medical data containing text, medical reports, scan images, video and audio files are stored effectively in a tensor called Health Data Tensor (HdTr). The second phase processes the data in HdTr with the help of concept hierarchies resulting in creation of various processed tensors, which are spanned over different levels of granularity. The third phase uses the tensors generated in second phase for effective query processing and result formation by automatically identifying granularity level of each attribute required for the final result formation. Each of the three phases is discussed in detail ahead.

To explain the proposed framework clearly, we used Health Fact Database (HFDB) [43] as an example. HFDB is a part of national health data warehouse of clinical records collected from various hospitals in the USA, which use electronic health service system. It consists of medical data collected for 10 years (1999–2008) and

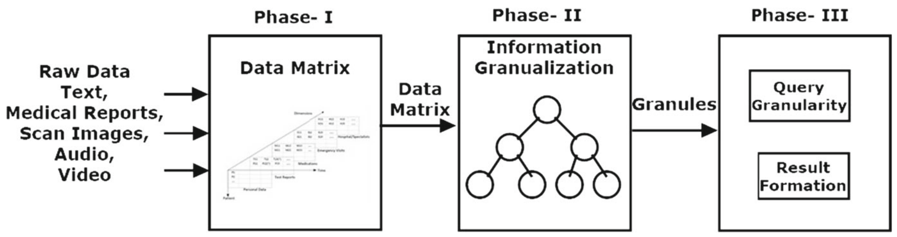


Fig. 2 Three phases of proposed framework for effectively mining healthcare data

contains more than one hundred thousand instances having 55 health attributes each. Complete description and whole database are available at [44].

4.1 Phase I: creating Health data Tensor (HdTr)

The healthcare data come from various sources in different formats such as text data, medical reports, scan images, ECG, audio data and video data. The major challenge of this phase is how to store such heterogeneous data in a manageable form so that effective processing can be done later. To achieve this goal, a tensor, called Health Data Tensor (HdTr), has been proposed to store such data.

Definition 1 (*Health Data Tensor*) Given an unequally spaced temporal medical data of 'n' patients consisting of 'm' health metrics, a health data tensor can be defined by a third-order tensor HdTr$\in \mathbb{R}^{I_P \times I_T \times I_M}$, where the orders I_P, I_T and I_M correspond to the dimensions 'patients,' 'time' and 'health metrics,' respectively, such that $I_P = n$, $I_M = m$ and $I_T = \max(\{t_i\}_{i=1}^{n})$ where t_i denotes the number of distinct timestamps for ith patient.

In the above definition,

- The medical data are termed as temporal data since each instance of data for every patient is associated with a timestamp. In addition, it is called unequally spaced temporal medical data because the patient's visit to doctor and data entry is generally at irregular time intervals.
- Various health metrics are personal data, test reports, medications, etc., as shown in Fig. 3.
- The timestamp differs from one patient to another, even when they are stored at the same column fiber of HdTr; hence, the time axis does not have absolute values.

Mathematically, HdTr can be expressed in terms of frontal slices by using Eq. (5).

$$\text{HdTr} = [H_1, H_2, H_3, H_4, \ldots\ldots\ldots., H_m]; \tag{5}$$

where each H_i is a frontal slice corresponding to 'm' health metrics. Further, each health metric can consist of various attributes. For example, personal data metric includes attributes such as age, weight and SSN. Therefore, each cell of the frontal slice itself forms a matrix. Hence, HdTr can be further written in terms of column fibers

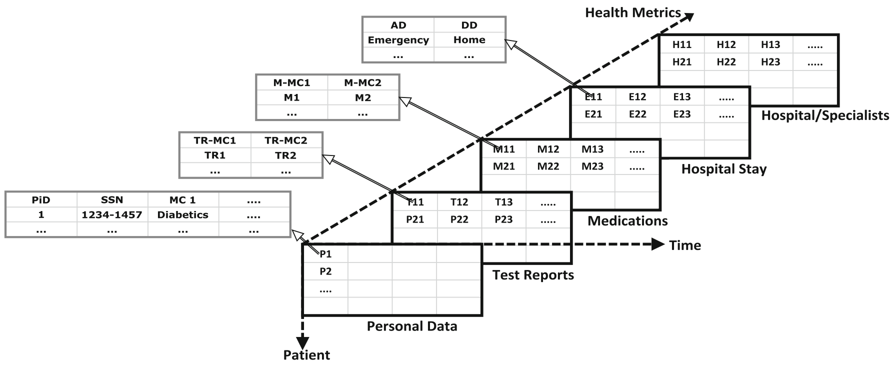


Fig. 3 Tensor representation of medical data. MC Medical condition, TR-MC1 test reports for medical condition 1, M-MC1 medication for medical condition 1, AD admission description, DD discharge description

(h_{ij}) and attributes (a_{ijk}) as shown in Eq. (6). [Equation (6) is written in a similar way as Eq. (4).]

$$\text{HdTr} = \left[\left[h_{11}h_{12}\ldots\ldots h_{1I_T}\right], \left[h_{21}h_{22}\ldots\ldots h_{2I_T}\right], \ldots\ldots\ldots\ldots\ldots, \left[h_{I_M1}h_{I_M2}\ldots\ldots h_{I_MI_T}\right]\right]$$
$$= \left[\begin{array}{l} \left[\left[a_{111}a_{112}\ldots\ldots a_{11n_1}\right], \left[a_{121}a_{122}\ldots\ldots a_{12n_1}\right], \ldots\ldots, \left[a_{1I_T1}a_{1I_T2}\ldots\ldots a_{1I_Tn_1}\right]\right], \\ \left[\left[a_{211}a_{212}\ldots\ldots a_{21n_2}\right], \left[a_{221}a_{222}\ldots\ldots a_{22n_2}\right], \ldots\ldots, \left[a_{2I_T1}a_{2I_T2}\ldots\ldots a_{2I_Tn_2}\right]\right], \ldots, \\ \left[\left[a_{I_M11}a_{I_M12}\ldots\ldots a_{I_M1n_{I_M}}\right], \left[a_{I_M21}a_{I_M22}\ldots\ldots a_{I_M2n_{I_M}}\right], \ldots\ldots, \left[a_{I_MI_T1}a_{I_MI_T2}\ldots\ldots a_{I_MI_Tn_{I_M}}\right]\right] \end{array}\right] \tag{6}$$

For the formation of HdTr from HFDB, various metrics are screened and stored. Some of the screened metrics of HFDB are listed in Table 1. Though Table 1 defines five health metrics and only two medical conditions, but, without loss of generality, the health metrics as well as the medical conditions can be expanded or reduced depending upon the dataset requirements. Each health metric further consists of some attributes as listed in Table 1. The attribute PiD is assigned to each distinct patient by the proposed framework in order to uniquely identify each patient. Thereafter, raw HFDB data are stored in HdTr such that the first order of HdTr corresponds to the patients while the second and third orders correspond to time and health metrics, respectively. Figure 3 shows frontal slice representation of HdTr where each frontal slice corresponds to one of the health metric. Here, personal data slice represents the personal information about the patient and his/her medical conditions. It serves the purpose of storing multiple medical conditions in one tensor slice effectively. With respect to multiple medical conditions, different test reports, medications, hospital stay and doctors assigned are stored in the corresponding cells of other tensor slices.

When a new patient registered with the system, he/she is added to HdTr and this operation is called 'Data Addition.' On the other hand, adding a new data entry about an existing patient in HdTr is called 'Data Appending.' These two operations help to automatically update HdTr with time.

Table 1 Identified health metrics of HFDB

Health metrics	Attributes	Remarks
Personal Data (H_1)	Patient Identification number (PiD), SSN, Name, Gender, Age, Weight, Address, Contact number, Medical Condition 1 (MC1), Medical Condition 2 (MC2), Timestamp (T)	PiD is a unique number assigned by the proposed system to each distinct patient in the medical data
		Medical conditions consist of values such as diabetics, cardiac arrest and cancer. For simplicity, only two medical conditions are indicated here
Test Reports (H_2)	Test report for MC1 (TR-MC1), Test report for MC2 (TR-MC2)	Test reports include blood test reports, scan images, ECG and MRI
Medications (H_3)	Medications for MC1 (M-MC1), Medications for MC2 (M-MC2)	—
Hospital stay (H_4)	Admission description, Discharge description	Admission description includes values such as emergency, elective, urgent, and transfer
		Discharge description includes values such as discharged to home, transferred and expired
Hospital/Specialist (H_5)	—	Gives details of hospital and specialist where the patient is treated

Definition 2 (*Data Addition*) Given a health data tensor HdTr$\in \mathbb{R}^{I_P \times I_T \times I_M}$ and data '*d*' of a new patient, the data addition operation adds '*d*' in HdTr such that the order of HdTr remains same but the dimension 'patients' expands by one, i.e., $I_P = I_P + 1$, $I_M = I_M$ and $I_T = I_T$.

Definition 3 (*Data Appending*) Given a health data tensor HdTr$\in \mathbb{R}^{I_P \times I_T \times I_M}$ and new data '*d*' of an existing patient '*j*,' the data appending operation adds '*d*' in HdTr such that the order of HdTr remains same but the dimension 'time' expands by one if $I_T = t_j$, where t_j denotes the number of distinct timestamps for patient '*j*,' i.e., $I_P = I_P$, $I_M = I_M$ and $I_T = \begin{cases} I_T + 1, & I_T = t_j \\ I_T, & \text{otherwise} \end{cases}$.

In HdTr, all information is stored in its raw format, i.e., images as images, sound files as audios and video files as video. These files are stored in their respective database and linked in the tensor slices. Though HdTr provides an effective way to store new as well as old information about every patients' health attributes with time, it is not efficient to analyze data in such raw formats. To make tensor efficient for information granularization step, HdTr is converted to processed tensor with the help of BDA tools. These tools analyze different formats of data, extract information in text format and store it in the tensor slices as shown in Fig. 4. This step of filtering is well known and discussed by many researchers [45]. Therefore, we concentrated on information representation and granularization step. The processed tensor, hence formed, is termed as Processed HdTr (PHdTr).

4.2 Phase II: information granularization

Once the medical data are effectively stored in PHdTr, it must be converted to different levels of abstraction, hence forming various granular levels. The second phase of proposed framework maps processed tensor and granular computing concepts to create different abstraction levels of tensors. In order to achieve this goal, concept hierarchy is created for each attribute present in the tensor, as shown in Fig. 5. Then, each attribute is mapped one to one with the concept hierarchy levels and different tensors are created. Therefore, each attribute is divided into multi-level concept hierarchies as shown by the concept hierarchy tree in Fig. 5 for the Age, Address and Patient Leaving the hospital. Similarly, concept hierarchies are created for all attributes present in the dataset. Each concept hierarchy has multiple levels, which are different for all attributes based on level of classification required. For example, there are three levels for attribute 'Age' and 'Leaving' while five levels for attribute 'Address.'

Definition 4 (*Concept Hierarchy*) Given any medical attribute '*a*' and its specific-to-general ordering such that the most-specific order consists of precise values for '*a*,' while the most-general order has value 'ANY,' then the concept hierarchy of '*a*' is defined by a partial ordered relation from the most-specific to the most-general ordering of '*a*.'

The concept hierarchy of each medical attribute is created by forming the specific-to-general ordering of the attribute. For example, for attribute 'age' having value

2 years, the specific-to-general ordering is given by <2, infant, child, ANY$>$. The most-specific order consists of value '2,' while the most-general order has value 'ANY.' When such ordering is created for every value in the domain of the attribute, then concept level of the values can be determined by one-to-one mapping with the concept hierarchy tree.

Definition 5 (*Concept Level* (l_a)) Given concept hierarchy of any medical attribute 'a,' then the concept level of any value of 'a' in concept hierarchy is defined by (a) $l_a \epsilon Z(b) l_a = 0$ for the domain of values at the most-specific order of 'a,' (b) l_a increases by one while moving from most-specific to most-general ordering. In other words, a medical attribute 'a' is said to be at ith concept level for each value in the domain of 'a' which is at ith level in its concept hierarchy.

Corollary 5.1 Coarser concept: *Given two concept levels 'l_a' and 'l'_a' of any medical attribute 'a' such that $l_a = i$ and $l'_a = j$, then l_a is coarser than l'_a iff $i > j$ and is denoted by $l_a \succcurlyeq l'_a$.*

Corollary 5.2 Finer concept: *Given two concept levels 'l_a' and 'l'_a' of any medical attribute 'a' such that $l_a = i$ and $l'_a = j$, then l_a is finer than l'_a iff $i < j$ and is denoted by $l_a \preccurlyeq l'_a$.*

Definition 6 (*Concept Hierarchy Tree*) A concept hierarchy tree for any medical attribute 'a' is defined by a finite set of nodes such that: (a) the tree contains a distinct

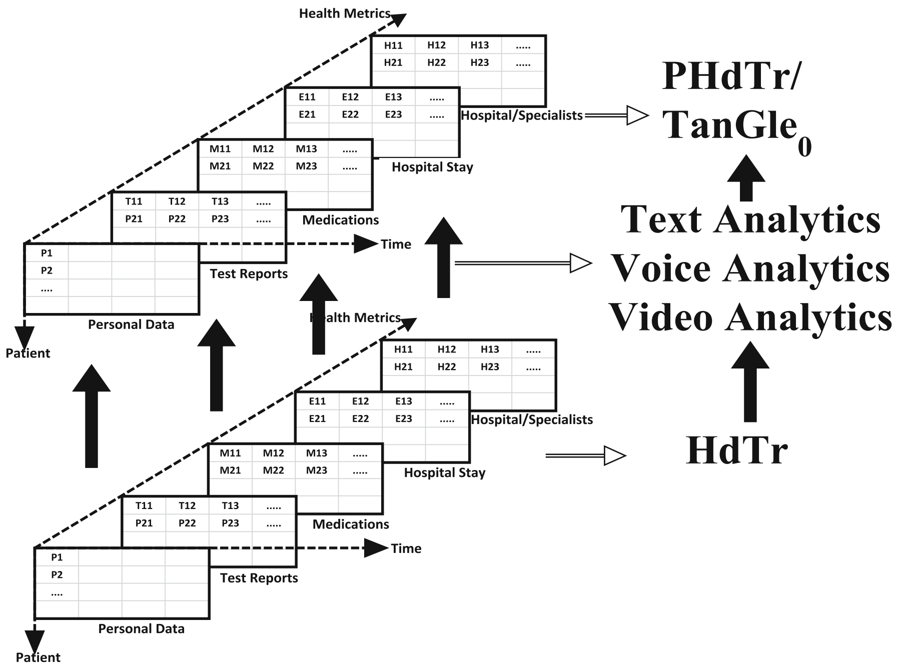


Fig. 4 Creation of processed HdTr (PHdTr/TanGle$_0$) from HdTr using analytics tools

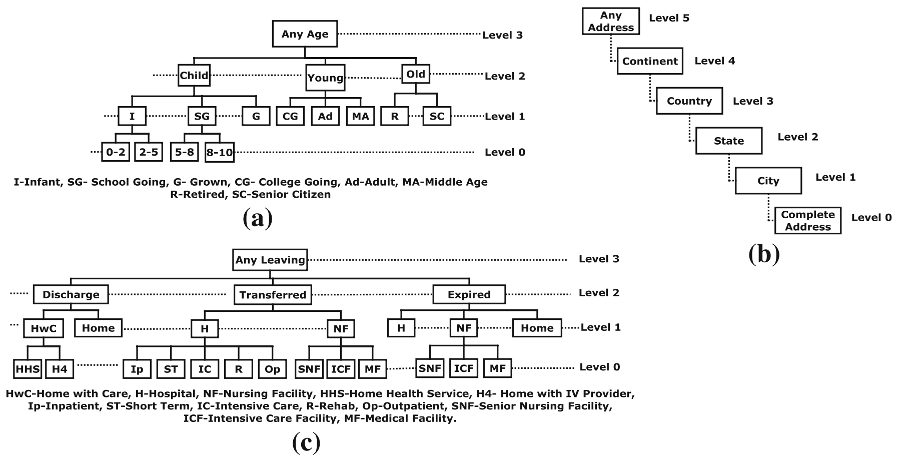

Any Age
Level 3
Child
Young
Old
Level 2
I
SG
G
CG
Ad
MA
R
SC
Level 1
0-2
2-5
5-8
8-10
Level 0
I-Infant, SG- School Going, G- Grown, CG- College Going, Ad-Adult, MA-Middle Age
R-Retired, SC-Senior Citizen
(a)
Any Address
Level 5
Continent
Level 4
Country
Level 3
State
Level 2
City
Level 1
Complete Address
Level 0
(b)
Any Leaving
Level 3
Discharge
Transferred
Expired
Level 2
HwC
Home
H
NF
H
NF
Home
Level 1
HHS
H4
Ip
ST
IC
R
Op
SNF
ICF
MF
SNF
ICF
MF
Level 0
HwC-Home with Care, H-Hospital, NF-Nursing Facility, HHS-Home Health Service, H4- Home with IV Provider,
Ip-Inpatient, ST-Short Term, IC-Intensive Care, R-Rehab, Op-Outpatient, SNF-Senior Nursing Facility,
ICF-Intensive Care Facility, MF-Medical Facility.
(c)

Fig. 5 Concept hierarchy tree of medical attributes stored in the database. **a** Age, **b** Address, **c** Leaving the hospital

root node such that its concept level $l_r \succcurlyeq l_c; \forall c$, where '$c$' is any node other than the root node, (b) the remaining nodes of the tree form an ordered collection of zero or more disjoint trees such that concept level of each node 'd,' $l_d = l_p - 1$, where l_p is the concept level of parent node of 'd,' (c) the leaf nodes are at concept level zero.

The concept hierarchies, thus formed, are then used to create tensor granules at different levels of granularity. Each of the resultant tensor is called $TanGLe_i$ (Tensor at Granular Level 'i').

Definition 7 TanGLe$_\mathbf{i}$ (*Tensor at Granular Level i*): Given processed medical tensor 'PHdTr.' If l_a and l'_a denote, respectively, the jth and finest concept level of any medical attribute 'a,' then medical tensor at granular level 'i' is defined by: (a) $TanGle_0 = PHdTr$, (b) $TanGle_i$ is formed from $TanGle_{i-1}$ such that $\forall a in TanGle_i$,

$$l_a = \begin{cases} i; & if\, l_a \neq l'_a \\ l'_a; & \text{otherwise} \end{cases}.$$

The maximum value of 'i' is given by $\max(\{l_a\}_{a=1}^{x})$, where 'x' is the total number of medical attributes.

Each $TanGLe_i$ is formed by one-to-one mapping with the concept hierarchy, as shown in Fig. 6. One-to-one mapping of each concept hierarchy and dimension of lower tensor will produce a higher level of tensor with more refined information. The lower- and higher-level tensors are called finer and coarser tensors, respectively.

Corollary 7.1 Coarser tensor : *Given two tensors $TanGLe_i$ and $TanGLe_j$, then $TanGLe_i$ is said to coarser than $TanGLe_j$ iff $i > j$ and is denoted by $TanGLe_i \succcurlyeq TanGLe_j$.*

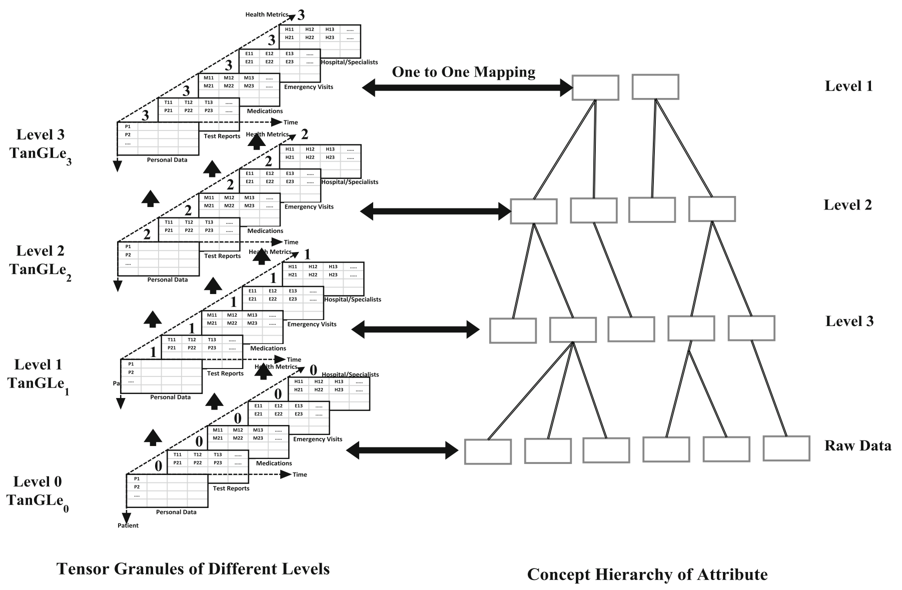


Fig. 6 One-to-one mapping of concept hierarchy of attributes and different level of tensors

Corollary 7.2 Finer tensor: *Given two tensors $TanGLe_i$ and $TanGLe_j$, then $TanGLe_i$ is said to finer than $TanGLe_j$ iff $i < j$ and is denoted by $TanGLe_i \preccurlyeq TanGLe_j$.*

Hence, a hierarchy of tensors from the finest tensor to the coarsest tensor is formed such that the finest tensor is at granular level 0, while coarsest tensor is at highest granular level as shown in Fig. 6.

Mathematically, $TanGLe_i$ can be written in terms of frontal slices using Eq. (7).

$$TanGLe_i = [H_1^i H_2^i H_3^i H_4^i \ldots\ldots. H_m^i]; \tag{7}$$

where the superscript 'i' is added to denote the concept levels of each frontal slice. Furthermore, if an attribute a_{xyz} at concept level 'i' is denoted by a^i_{xyz}, then $TanGLe_i$ can be expressed in terms of column fibers and attributes by using Eq. (8).

Therefore, it can be observed that by using this proposed combination of tensor information representation and concept hierarchy concept, large amount of information can be stored in an effective way. Information can be mined at different levels depending on type of query arrived at the system.

4.3 Phase III: query processing and result formation

After efficient storage of tensors at different granular levels in phase II, the most important step is to determine how the incoming query is to be processed in order to generate a correct result. To achieve this goal, initially the query is analyzed to recognize the attributes required in the query result. A tensor, called 'Resultant Tensor

(RTr)' consisting of all the required attributes and their values is formed. The order and dimensions of RTr can be different from the HdTr/$TanGLe_i$ and depends upon the number of required attributes in the query. The result of the query can then be easily retrieved from RTr. The benefit creating RTr rather than directly solving the query is twofold. Firstly, the complexity of data is reduced to only the required attributes and elimination of non-required information. Secondly, similar to the concept of locality of reference [46], the probability of relevance of a new query to the previous one is generally high. Therefore, the next query can be solved using the same RTr resulting in reduced time for query resolving.

In the light of the above, it can be noted that there are three steps for query processing. The first step identifies the required attributes, while the second step creates RTr. The third step simply applies functions such as aggregate and sum on RTr to obtain the result. These steps are explained in detail below.

Step 1: Identifying required attributes In order to do so, keywords are extracted from the query using keyword extraction algorithm [45]. The keywords are selected such that they correspond to either a health metric, an attribute or value of attribute.
For example, let the following query arrives:

Q1: 'How many hospitals in New York have been treating Diabetics patients in the age group 0–2.'

Here, the keywords include <hospitals, New York, Diabetics (Age, 0–2)>. It can be noted here that the keyword 'hospital' is one of the health metric, keywords 'New York' and 'diabetics' are the values for attributes 'address' and 'medical condition,' respectively, while the keyword 'age' is an attribute itself.

$$TanGLe_i = \left[\left[h^i_{11} h^i_{12} \ldots\ldots h^i_{1I_T}\right], \left[h^i_{21} h^i_{22} \ldots\ldots h^i_{2I_T}\right], \ldots\ldots\ldots\ldots\ldots\ldots, \left[h^i_{I_M 1} h^i_{I_M 2} \ldots\ldots h^i_{I_M I_T}\right]\right]$$
$$= \begin{bmatrix} \left[\left[a^i_{111} a^i_{112} \ldots\ldots a^i_{11n_1}\right], \left[a^i_{121} a^i_{122} \ldots\ldots a^i_{12n_1}\right], \ldots\ldots, \left[a^i_{1I_T 1} a^i_{1I_T 2} \ldots\ldots a^i_{1I_T n_1}\right]\right], \\ \left[\left[a^i_{211} a^i_{212} \ldots\ldots a^i_{21n_2}\right], \left[a^i_{221} a^i_{222} \ldots\ldots a^i_{22n_2}\right], \ldots\ldots, \left[a^i_{2I_T 1} a^i_{2I_T 2} \ldots\ldots a^i_{2I_T n_2}\right]\right], \ldots, \\ \left[\left[a^i_{I_M 11} a^i_{I_M 12} \ldots\ldots a^i_{I_M 1n_{I_M}}\right], \left[a^i_{I_M 21} a^i_{I_M 22} \ldots\ldots a^i_{I_M 2n_{I_M}}\right], \ldots\ldots, \left[a^i_{I_M I_T 1} a^i_{I_M I_T 2} \ldots\ldots a^i_{I_M I_T n_{I_M}}\right]\right] \end{bmatrix} \tag{8}$$

When the extracted keyword corresponds to the value of some attribute, then the appropriate attribute is identified using concept hierarchy tree. For example, in case of keyword 'New York,' the attribute 'address' is identified.

The above process results in a list 'L' of attributes and/or health metrics required in the query. In addition, the concept level of each attribute and metric in 'L' is also identified using concept hierarchy tree. For example, in Q1, the concept level of 'age' is 0, while the concept level of 'address' is 1 (as observed from Fig. 5). Hence, the list 'L' consists of all the required attributes and/or health metrics along with the concept level of each.
Mathematically, 'L' is given by the following equations.

$$\text{If } H^i = \left\{H^i_1, H^i_2, \ldots, H^i_m\right\},$$
$$A^i = \left\{a^i_{xyz} | 1 \leq x \leq I_M, 1 \leq y \leq I_T, 1 \leq z \leq n_{I_M}\right\},$$

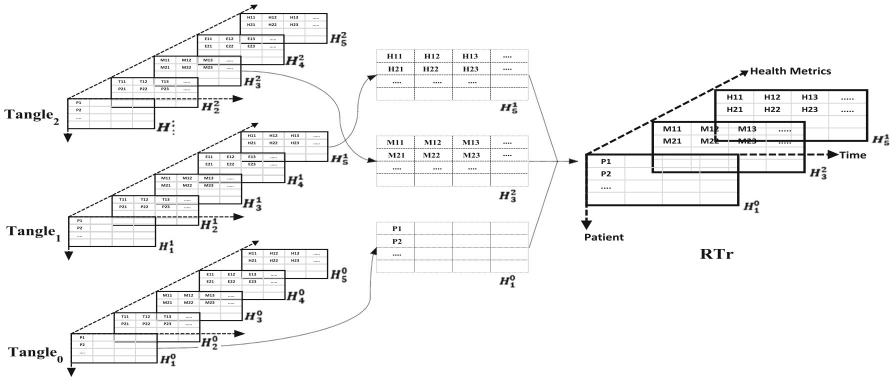


Fig. 7 Creating resultant tensor (RTr)

$$H = \bigcup_i H^i \text{ and } A = \bigcup_i A^i$$
$$\text{then } L = L_1 \cup L_2; \text{ where } L_1 \subseteq H \text{ and } L_2 \subseteq A.$$

Step 2: Creation of RTr: Using ‘L,’ RTr is created by:

(a) If $L_2 = \emptyset \Rightarrow L = L_1$, i.e., ‘$L$’ consists of only health metric(s), then RTr is formed by selecting the frontal slices corresponding to health metric(s) in ‘L.’ For example, if $L = \{H_1^0, H_3^2, H_5^1$, then $RTr = [H_1^0, H_3^2, H_5^1]$ as shown in Fig. 7.

(b) If $L_1 = \emptyset \Rightarrow L = L_2$, i.e., ‘$L$’ consists of only attributes, then two cases arise:

Case 1: All the attributes in ‘L’ belong to different frontal slices Let $a_{pqr}^i \epsilon L$ and $a_{xyz}^j \epsilon L$ be two attributes such that $p \neq x$, then $RTr = [H_p^i, H_x^j]$. Without loss of generality, it can be extended to any number of attributes. In other words, RTr is formed by selecting the frontal slices corresponding to health metric(s) which consists of attributes in ‘L,’ i.e., for every $a_{pqr}^i \epsilon L$, $RTr = [H_p^i]$. For example, let $L = \{a_{114}^2, a_{323}^4$, then $RTr = [H_1^2, H_3^4]$.

Case 2: Two or more attributes in ‘L’ belong to same frontal slice Let $a_{pqr}^i \epsilon L$ and $a_{xyz}^j \epsilon L$ be two attributes such that $p = x$, then RTr is created by first constructing each column fiber and a frontal slice from these fibers such that

$$h_{ps} = \left[a_{ps1}^k a_{ps2}^k \ldots a_{pqr}^i \ldots a_{xyz}^j \ldots a_{pqs}^k\right];$$
$$\text{where } k = \begin{cases} i; \; if\, i > j \\ j; \text{ otherwise} \end{cases} \text{ and } s = \{1, 2, \ldots, I_T\};$$
$$H_p = \left[h_{p1} h_{p2} \ldots\ldots h_{pI_T}\right];$$

It can be noted that here superscript of h_{ps} *and* H_p has been omitted since the fiber and slice consist of attributes from different concept levels ‘i’ and ‘j.’ In addition, for all the other attributes (belonging to different frontal slices) the corresponding frontal slices are selected. Once all the frontal slices are obtained, RTr is constructed in a similar way as in (a).

(c) If $L_1 \neq \emptyset$ and $L_2 \neq \emptyset$, i.e., 'L' consists of health metric(s) as well as attributes, then RTr is created by using steps of both (a) and (b).

Step 3: Final Result Formation

Once RTr is created, the phase III is just to apply functions such as aggregate and sum on RTr to obtain the result. Hence, it can be observed that RTr provides a highly flexible and efficient method to resolve any type of query arriving at the system.

5 Performance evaluation

The proposed method for representation of biomedical data using tensor analysis and information granularization is experimentally evaluated and compared with baseline tensor tools and algorithms. Many tensor decomposition tools exist such as CANDECOMP/PARAFAC Decomposition [47], Tucker Decomposition [48], INDSCAL [49], PARAFAC2 [50] and CANDELINC [51]. In all traditional decomposition methods, PARAFAC2 decomposition method is chosen for the comparison with the proposed TDRM because PARAFAC2 supports the tensor with variable length of rows and constant columns.

Dataset used in the performance evaluation of the proposed TDRM system was extracted from the information of 130 clinical care hospitals from 1999 to 2008 (10 years) with average number of beds equal to 300. This dataset can be download from the VCU DMB lab (website available at [44]). This dataset has 117 different feature values and contains 74,036,643 hospital visits of approximately 17,880,231 unique patients and 2,889,571 facilities providers. This dataset includes all the admissions and discharge from the hospital for inpatients, outpatients and emergency department. Some adjustments have been made in the dataset so that it can be used in performance evaluation in this paper. Dataset is converted into the tensor format using the Tensor Toolbox [52] available in MATLAB, and we are calling this tensor as processed tensor.

The proposed TDRM has been compared with CANDELINC also for which we have chosen student performance dataset [53]. CANDELINC requires rows and columns of tensors to be fixed in complete dataset, and student performance dataset has been converted to fixed rows and columns so that TDRM can also be compared with CANDELINC. Student dataset contains 30 distinct attributes with more than 649 instances. Few attributes are classified as personal attributes to check the privacy feature of proposed TDRM.

PARAFAC2 is applied for decomposition of processed tensor created from HFDB [43,44] using the Tensor Toolbox [52]. On the parallel hand for TDRM, processed tensor is divided into multiple tensors based on the information granularity levels of nine features which are age, address, race, doctor specialty, admission department, admission description, discharge description, diabetic type and diabetic values.

Table 2 Different types of queries to HFDB tensor

Type of query	Probability	Description
Normal query	.40	Normal queries require data from one dimension of tensor with maximum two attributes
Personal queries	.10	Personal queries ask about personal information of user
Advance queries	.30	Advance queries require data from two dimensions of tensor with maximum four attributes
Complex queries	.20	Complex queries require data from all dimensions of tensor with any number of attributes

5.1 Experimental setup

The main objective of experiment is to check the accuracy achieved in results of queries by dividing tensor into different information granularity levels. PARAFAC2 is applied for decomposition of processed tensor created from HFDB [43,44] using the Tensor Toolbox [52]. Similarly, CANDELINC is applied for the decomposition of processed tensor created from student performance dataset [53] using the Tensor Toolbox [52]. On the parallel hand for TDRM, processed tensor is divided into multiple tensors based on the information granularity levels of nine features which are age, address, race, doctor specialty, admission department, admission description, discharge description, diabetic type and diabetic values. For student performance dataset, information granularity levels are created for school, age, address, family size, study time, travel time, failures and health. In the experiment of 60 min, 10 queries per minute are sent to PARAFAC2, CANDELINC and proposed TDRM for respective datasets. MATLAB Tensor Toolbox [52] is used in both the cases to get the result of the queries. A set of 400 different types of queries are created for HDFB dataset and 100 different types of queries are created for student performance dataset. Table 2 shows different types of queries and their probability of occurrence for both datasets. Normal queries are simple select queries from the tensor based on some simple condition, personal queries are select queries which retrieve personal information of the user, advance queries use join operate to select information based on more than two dimensions of tensor and complex queries retrieve data using join operation in more than five dimensions of tensor. MATLAB code has been deployed as Java package, which selects query randomly from these set of queries following the probability distribution stated in Table 2. Complete experimental setup has been done on Microsoft Azure DS1$_{\mathrm{V2}}$ instance with 1 cores and 3.5 GB of memory, 2 data disks, 3200 max IOPS using Windows Server 2012 R2, MATLAB R2016a and Java 7. Figure 8 provides experimental results of TDRM as compared with the PARAFAC2, and Fig. 9 shows experiment results of TDRM as compared with CANDELINC. Discussion of these results is given in the next subsection.

5.2 TDRM on cloud computing infrastructure

The proposed TDRM has been deployed with different cloud configurations to test its scalability across public cloud. Table 3 shows different cluster configurations used

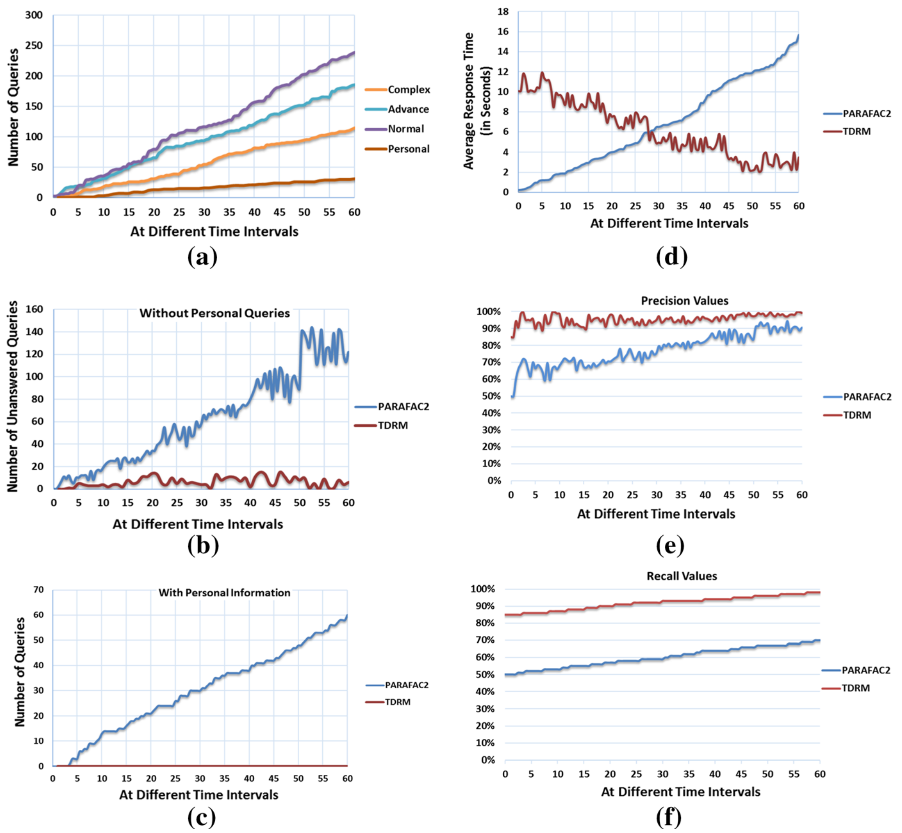


Fig. 8 Different metrics from simulation of TDRM and PARAFAC2. **a** Total number of queries to both systems, **b** total unanswered queries of both systems, **c** total answered queries for personal information of both systems, **d** average response to answered queries of both systems, **e** precision value of PARAFAC2 as compared to proposed method and **f** recall values of PARAFAC2 as compared to proposed method

for Microsoft Azure cloud [54]. Figure 10 shows different parameters recorded for PARAFAC2 and TDRM on Microsoft Azure cloud infrastructure. Figure 11 shows different parameters recorded for CANDELINC and TDRM on Microsoft Azure cloud infrastructure.

5.3 Discussion

For evaluation of the proposed framework, an experiment of 60 min is conducted where both PARAFAC2 and TDRM received same number and type of queries from the query pool created. Figure 8a shows the total number of queries submitted to both PARAFAC2 and TDRM systems using which multiple important results are generated. We conducted similar experiments using queries with CANDELINC and TDRM using student performance dataset as shown in Fig. 9a.

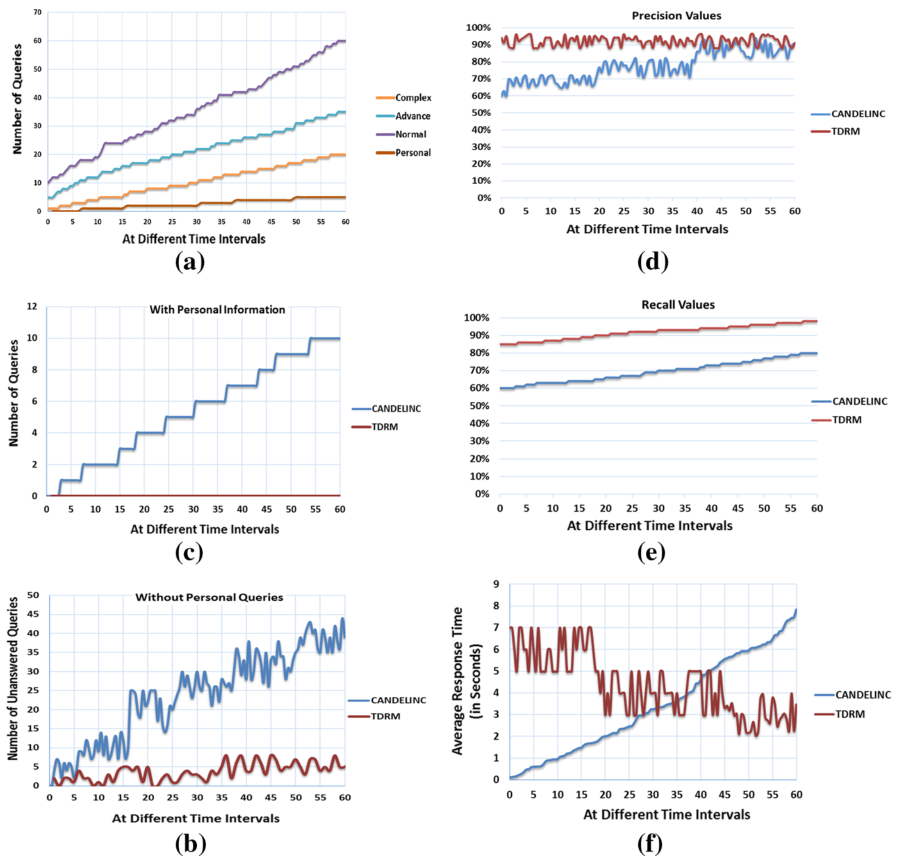


Fig. 9 Different metrics from simulation of TDRM and CANDELINC. **a** Total number of queries to both systems, **b** total unanswered queries of both systems, **c** total answered queries for personal information of both systems, **d** precision value of CANDELINC as compared to TDRM, **e** recall values of CANDELINC as compared to TDRM and **f** average response to answered queries of both systems

Unanswered queries are the queries for which there is no result. Due to granularization and concept levels created, almost all queries are answered by the TDRM effectively, whereas PARAFAC2 and CANDELINC systems are only able to answer very few queries and very few complex queries are answered by PARAFAC2 and CANDELINC in given time frame. Figures 8b and 9b show the number of unanswered queries by both the systems. Total unanswered queries by PARAFAC2 and CANDELINC methods increased linearly with time, whereas TDRM unanswered queries are at same level throughout the experiment.

To study the relevancy of TDRM in biomedical data storage, it is tested with personal information queries as listed in Table 2. TDRM did not answer any query related to personal information of user that is lowest level of concept hierarchies, as shown in Figs. 8c and 9c. However, traditional tensor decomposition methods PARAFAC2 and CANDELINC do not distinguish the personal queries from all other queries, which

Table 3 Different cluster configurations used for experiment

S. no.	Name	Cores	Memory	Data disks	Max IOPS	Cost per hour
1	$DS1_{V2}$ standard	1	3.5	2	3200	0.16
2	$DS2_{V2}$ standard	2	7	4	6400	0.312
3	$DS3_{V2}$ standard	4	14	8	12,800	0.624
4	$DS4_{V2}$ standard	8	28	16	25,600	1.248

make them unsuitable for medical data mining. Hence, TDRM method is able to provide privacy and security to medical data of individuals.

Figures 8d and 9f depict the average response time of answered queries for both the systems with TDRM. TDRM initially has large response time than PARAFAC2 and CANDELINC because it must create resultant tensor for each query. But after approx. 28 min of experiment for PARAFAC2 and 20 min for CANDELINC, response time of proposed system decreases to 50% due to cache storage of resultant tensors. However, PARAFAC2 and CANDELINC response time increases linearly since the number of queries increased with time throughout the experiment.

Precision and recall values are very important metrics for any query system. Both precision and recall explains the measure of relevance of any system. In case of TDRM, precision is the ratio of retrieved relevant answers to the specific queries, whereas recall is the ratio of relevant answers to the all the queries that are generated by the system. It provides a measure to check the answer provided by the system because irrelevant answer will again confuse the user and decrease the usability of the system. Precision of the TDRM is above 85% from the starting, and it ranges between 94% and 99% after 10 min of the experiment. On the other hand, precision in case of PARAFAC2 is around 50% in the starting and reaches to maximum of 87% till the end of experiment as shown in Fig. 8e. For CANDELINC, precision rate started around 60% and was similar to TDRM at the end of experiment. This is due to the fact that student performance dataset is small and size is also fixed as shown in Fig. 9d. Similarly, recall values are very high for the proposed method as compared to PARAFAC2. The proposed method provides recall values above 85% throughput the experiment, whereas PARAFAC2 reaches maximum to 70% as shown in Fig. 8f. For CANDELINC, recall values reach from 60 to 80% throughout the experiment as shown in Fig. 9e. These values prove that proposed system not only provides answers to more queries but answers provided by the system are relevant also.

Figures 10a and 11a provide average response time of PARAFAC2 and CANDELINC with TDRM, respectively, for 60 min of experiments. TDRM provides approximately 30% improvement from PARAFAC2 and 20% improvement with CANDELINC in all four types of instances types due to granularization of tensors. Figure 11b provides the comparison of execution time on Microsoft Azure for CANDELINC for 500 queries of student performance dataset. Response time for both methods decreased for larger instance type due to parallel running of queries. Experiments are conducted for different sizes of input queries to test the total execution time of both methods in providing results to all the submitted queries. Figure 10b, c pro-

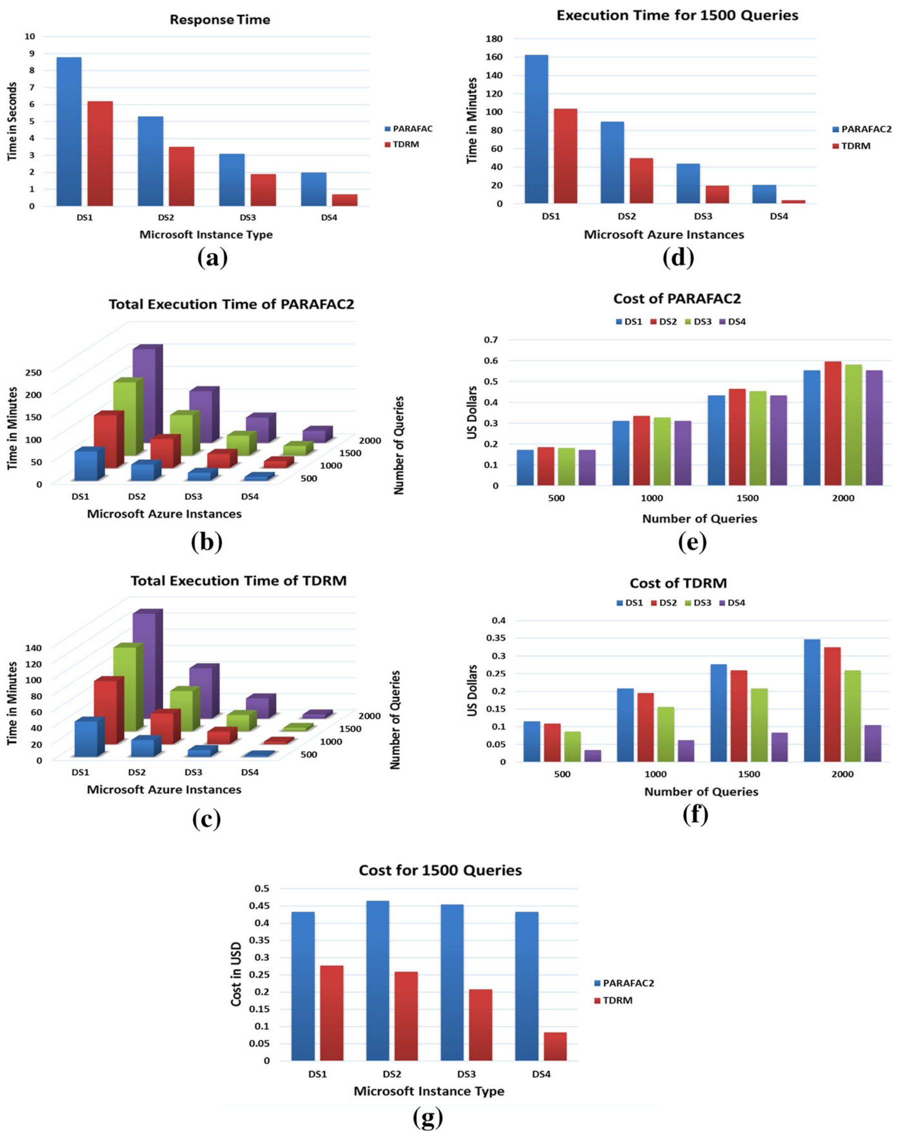
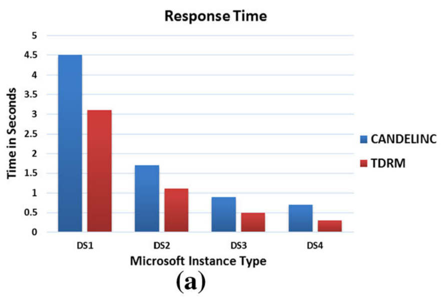


Fig. 10 Different metrics from PARAFAC2 and TDRM experiment on Microsoft Azure cloud instances of different configurations. **a** Average response time of queries for 60 min of experiment on different instance types, **b** total execution time for different sets of queries and instances types for PARAFAC2, **c** total execution time for different sets of queries and instances types for TDRM. **d** Comparison of execution time for PARAFAC2 and TDRM for 1500 queries, **e** cost of running PARAFAC2 on different instance types, **f** cost of running TDRM on different instance types and **g** comparison of cost for PARAFAC2 and TDRM for 1500 queries

vides total execution time for both systems for different query sizes and instance types. Difference in execution time for different query sizes is less in TDRM as compared to PARAFAC2 due to storage of resultant tensors. Figure 10d provides the comparison of execution time for PARAFAC2 and TDRM using 1500 queries. Figure 10e,



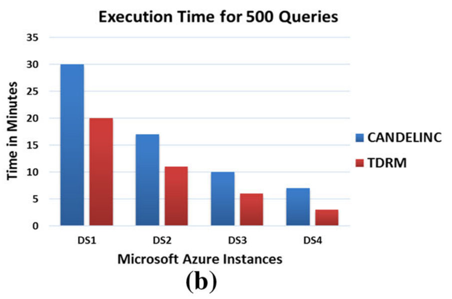


Fig. 11 Different metrics from CANDELINC and TDRM experiment on Microsoft Azure cloud instances of different configurations. **a** Average response time of queries for 60 min of experiment on different instance types and **b** comparison of execution time for CANDELINC and TDRM for 500 queries

f shows the total cost of running experiments for both PARAFAC2 and TDRM. For PARAFAC2, cost of running the experiments is almost same for all instance types. However, TDRM reduces the cost to around 60% from DS1 to DS4 because TDRM stores resultant tensor which makes faster execution of application so less cost for running it. Figure 10g provides the comparison of cost for the execution of PARAFAC2 and TDRM on different Microsoft instances using 1500 queries.

From all the experimental results, it is clear that proposed system provides a unique method to store as well as mine medical data using tensors and information granularization. The proposed system provides faster computation, low response time, more privacy and high relevancy as compared to baseline PARAFAC2 tensor analysis method.

6 Conclusions and future work

Big Data storage and mining research is the latest trend in data mining communities, and we proposed a tensor-based data representation and mining framework for healthcare data. Our work provides an efficient method to store data in the form of information granules. Key element of our work is the storage of raw data in the form of multi-dimensional matrix and replicating this matrix at different abstraction levels using concept hierarchy of each attribute. We provided the theoretical as well as experimental aspects behind the proposed methodology, which is effective and efficient in big data storage. Experimental results conducted on Microsoft Azure cloud show reduction in execution time and cost for TDRM as compared to PARAFAC2 and CANDELINC.

In the future, we plan to implement our framework using parallel programs of tensor decomposition using frameworks such as Aneka and Hadoop to harness the power of multiple nodes (VMs) across one or more cloud datacenter infrastructure. We also plan to extend this method to different application domains such as agriculture.

References

1. Hashem IAT, Yaqoob I, Badrul Anuar N, Mokhtar S, Gani A, Ullah Khan S (2014) The rise of 'big data' on cloud computing: review and open research issues. Inf Syst 47:98–115
2. Clifton L, Clifton DA, Pimentel MAF, Watkinson PJ, Tarassenko L (2014) Predictive monitoring of mobile patients by combining clinical observations with data from wearable sensors. IEEE J Biomed Health Inform 18(3):722–30
3. Dean J (2014) Big data, data mining, and machine learning. Wiley, London
4. Qian Y, Liang J, Wu WZZ, Dang C (2011) Information granularity in fuzzy binary GrC model. IEEE Trans Fuzzy Syst 19(2):253–264
5. Lin TY (2003) Granular computing. In: Proceedings of 4th Chinese National Conference Rough Sets Soft Computing, vol 2639, pp 16–24
6. Yao Y (2005) Perspectives of granular computing. In: 2005 IEEE International Conference on Granular Computing, pp 85–90
7. Kuang L, Hao F, Yang LT, Lin M, Luo C, Min G (2014) A tensor-based approach for big data representation and dimensionality reduction. IEEE Trans Emerg Top Comput 2(3):280–291
8. Yuan L, Yu Z, Luo W, Hu Y, Feng L, Zhu AX (2015) A hierarchical tensor-based approach to compressing, updating and querying geospatial data. IEEE Trans Knowl Data Eng 27(2):312–325
9. Erdman AG, Keefe DF, Schiestl R (2013) Grand challenge: applying regulatory science and big data to improve medical device innovation. IEEE Trans Biomed Eng 60(3):700–706
10. Lin W, Dou W, Zhou Z, Liu C (2015) A cloud-based framework for Home-diagnosis service over big medical data. J Syst Softw 102:192–206
11. Zhang F, Cao J, Khan SU, Li K, Hwang K (2014) A task-level adaptive MapReduce framework for real-time streaming data in healthcare applications. Future Gener Comput Syst 43–44:149–160
12. Castiglione A, Pizzolante R, De Santis A, Carpentieri B, Castiglione A, Palmieri F (2015) Cloud-based adaptive compression and secure management services for 3D healthcare data. Future Gener Comput Syst 43–44:120–134
13. Saleem M, Kamdar MR, Iqbal A, Sampath S, Deus HF, Ngonga Ngomo AC (2014) Big linked cancer data: integrating linked TCGA and PubMed. J Web Semant 27:34–41
14. Jiang P, Winkley J, Zhao C, Munnoch R, Min G, Yang LT (2014) An intelligent information forwarder for healthcare big data systems with distributed wearable sensors. IEEE Syst J 10(3):1–13
15. Belaud JP, Negny S, Dupros F, Michéa D, Vautrin B (2014) Collaborative simulation and scientific big data analysis: illustration for sustainability in natural hazards management and chemical process engineering. Comput Ind 65(3):521–535
16. Philip Chen CL, Zhang CY (2014) Data-intensive applications, challenges, techniques and technologies: a survey on big data. Inf Sci (Ny) 275:314–347
17. Yao Y (2007) The art of granular computing. Rough Sets Intell Syst Paradig 4585:101–112
18. Skowron A, Stepaniuk J (2001) Information granules: towards foundations of granular computing. Int J Intell Syst 16(1):57–85
19. Qiu GF, Ma JM, Yang HZ, Zhang WX (2010) A mathematical model for concept granular computing systems. Sci China Ser F Inf Sci 53(7):1397–1408
20. Lin TY (2008) Granular computing: common practices and mathematical models.' In: IEEE International Conference on Fuzzy Systems, pp 2405–2411
21. Qian Y, Zhang H, Li F, Hu Q, Liang J (2013) International journal of approximate reasoning set-based granular computing: a lattice model. Int J Approx Reason 1:1–19
22. Yao YY (2002) A generalized decision logic language for granular computing. In: 2002 IEEE World Congress on Computational Intelligence. 2002 IEEE International Conference on Fuzzy Systems. FUZZ-IEEE'02. Proceedings (Cat. No. 02CH37291), vol 1
23. Song M, Wang Y (2015) Human centricity and information granularity in the agenda of theories and applications of soft computing. Appl Soft Comput J 27:610–613
24. Li J, Mei C, Xu W, Qian Y (2015) Concept learning via granular computing: a cognitive viewpoint. Inf Sci (Ny) 298:447–467
25. Xu W, Li W (2014) Granular computing approach to two-way learning based on formal concept analysis in fuzzy datasets. IEEE Trans Cybern 46(2168–2275 (Electronic)):366–379
26. Yao Y (2009) Interpreting concept learning in cognitive informatics and granular computing. IEEE Trans Syst Man Cybern Part B Cybern 39(4):855–866

27. Gacek A (2015) Signal processing and time series description: a perspective of computational intelligence and granular computing. Appl Soft Comput 27:590–601
28. Sanchez M a, Castillo O, Castro JR, Melin P (2014) Fuzzy granular gravitational clustering algorithm for multivariate data. Inf Sci (Ny) 279:498–511
29. Wang X, Liu X, Zhang L (2014) A rapid fuzzy rule clustering method based on granular computing. Appl Soft Comput 24:534–542
30. Peters G (2011) Granular box regression. IEEE Trans Fuzzy Syst 19(6):1141–1152
31. a Cimino MGC, Lazzerini B, Marcelloni F, Pedrycz W (2014) Genetic interval neural networks for granular data regression. Inf Sci (Ny) 257:313–330
32. Cruz-Vega I, Escalante HJ, Reyes CA, Gonzalez JA, Rosales A (2015) Surrogate modeling based on an adaptive network and granular computing. Soft Comput 20(4):1549–1563
33. Liu Y, Jiang Y, Huang L (2010) Modeling complex architectures based on granular computing on ontology. IEEE Trans Fuzzy Syst 18(3):585–598
34. Zhou D, Dai X (2015) A method for discovering typical process sequence using granular computing and similarity algorithm based on part features. Int J Adv Manuf Technol 78(9–12):1781–1793
35. Yao Y (2006) Granular computing for data mining. In: Proceedings of SPIE Conference on Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security (2006). doi:10.1117/12.669023
36. Wu WZ, Leung Y, Mi JS (2009) Granular computing and knowledge reduction in formal contexts. IEEE Trans Knowl Data Eng 21(10):1461–1474
37. Bargiela A, Pedrycz W (2006) Toward a theory of granular computing for human-centred information processing. IEEE Trans Fuzzy Syst 16(2):320–330
38. Yager RR (2008) Intelligent social network analysis using granular computing. Int J Intell Syst 23(11):1197–1219
39. Qian J, Lv P, Yue X, Liu C, Jing Z (2015) Hierarchical attribute reduction algorithms for big data using MapReduce. Knowl Based Syst 73:18–31
40. Wang D-W, Liau C-J, Hsu T-S (2004) Medical privacy protection based on granular computing. Artif Intell Med 32(2):137–149
41. Minaev YN, Filimonova OY, Minaeva JI (2014) Kronecker (tensor) models of fuzzy-set granules. Cybern Syst Anal 50(4):519–528
42. Kolda TG, Bader BW (2008) Tensor decompositions and applications. SIAM Rev 51(3):455–500
43. Strack B et al (2014) Impact of HbA1c measurement on hospital readmission rates: analysis of 70,000 clinical database patient records. Biomed Res Int 2014:1–11
44. VCU DMB Lab. http://cioslab.vcu.edu/. Accessed 21 June 2017
45. Meng S, Dou W, Zhang X, Chen J (2014) KASR: a keyword-aware service recommendation method on mapreduce for big data applications. IEEE Trans Parallel Distrib Syst 25(12):3221–3231
46. Wang D, Yeo CK (2012) Exploring locality of reference in P2P VoD systems. IEEE Trans Multimed 14(4):1309–1323
47. Pouryazdian S, Beheshti S, Krishnan S (2016) CANDECOMP/PARAFAC model order selection based on reconstruction error in the presence of Kronecker structured colored noise. Digit Signal Process 48:12–26
48. Tucker LR (1966) Some mathematical notes on three-mode factor analysis. Psychometrika 31(3):279–311
49. Carroll JD, Chang J-J (1970) Analysis of individual differences in multidimensional scaling via an n-way generalization of 'Eckart-Young' decomposition. Psychometrika 35(3):283–319
50. Kiers HAL, ten Berge JMF, Bro R (1999) PARAFAC2—Part I. A direct fitting algorithm for the PARAFAC2 model. J Chemom 13(3–4):275–294
51. Douglas Carroll J, Pruzansky S, Kruskal JB (1980) Candelinc: a general approach to multidimensional analysis of many-way arrays with linear constraints on parameters. Psychometrika 45(1):3–24
52. Bader BW, Kolda TG (2015) MATLAB Tensor Toolbox Version 2.6. http://www.sandia.gov/~tgkolda/TensorToolbox/index-2.6.html. Accessed 21 June 2017
53. Cortez P, Silva A (2008) Using data mining to predict secondary school student performance. In: 5th Annual Future Business Technology Conference, vol 2003, no 2000, pp 5–12
54. Microsoft Azure (2016) Microsoft Azure Pricing calculator. Microsoft. https://azure.microsoft.com/en-us/pricing/calculator/