SPARTan: Scalable PARAFAC2 for Large & Sparse Data

Ioakeim Perros
Georgia Institute of Technology
perros@gatech.edu

Evangelos E. Papalexakis
University of California, Riverside
epapalex@cs.ucr.edu

Fei Wang
Weill Cornell Medicine
feiwang03@gmail.com

Richard Vuduc
Georgia Institute of Technology
richie@cc.gatech.edu

Elizabeth Searles
Children's Healthcare Of Atlanta
elizabeth.searles@choa.org

Michael Thompson
Children's Healthcare Of Atlanta
michael.thompson@choa.org

Jimeng Sun
Georgia Institute of Technology
jsun@cc.gatech.edu

ABSTRACT

In exploratory tensor mining, a common problem is how to analyze a set of variables across a set of subjects whose observations do not align naturally. For example, when modeling medical features across a set of patients, the number and duration of treatments may vary widely in time, meaning there is no meaningful way to align their clinical records across time points for analysis purposes. To handle such data, the state-of-the-art tensor model is the so-called PARAFAC2, which yields interpretable and robust output and can naturally handle sparse data. However, its main limitation up to now has been the lack of efficient algorithms that can handle large-scale datasets.

In this work, we fill this gap by developing a scalable method to compute the PARAFAC2 decomposition of large and sparse datasets, called SPARTan. Our method exploits special structure within PARAFAC2, leading to a novel algorithmic reformulation that is both faster (in absolute time) and more memory-efficient than prior work. We evaluate SPARTan on both synthetic and real datasets, showing 22× performance gains over the best previous implementation and also handling larger problem instances for which the baseline fails. Furthermore, we are able to apply SPARTan to the mining of temporally-evolving phenotypes on data taken from real and medically complex pediatric patients. The clinical meaningfulness of the phenotypes identified in this process, as well as their temporal evolution over time for several patients, have been endorsed by clinical experts.

KEYWORDS

Sparse Tensor Factorization; PARAFAC2; Phenotyping; Unsupervised learning

KDD '17, August 13-17, 2017, Halifax, NS, Canada

ACM ISBN 978-1-4503-4887-4/17/08…$15.00
https://doi.org/10.1145/3097983.3098014

ACM Reference format:
Ioakeim Perros, Evangelos E. Papalexakis, Fei Wang, Richard Vuduc, Elizabeth Searles, Michael Thompson, and Jimeng Sun. 2017. SPARTan: Scalable PARAFAC2 for Large & Sparse Data. In *Proceedings of KDD '17, Halifax, NS, Canada, August 13-17, 2017,* 10 pages.
https://doi.org/10.1145/3097983.3098014

1 INTRODUCTION

This paper concerns tensor-based analysis and mining of multimodal data where observations are difficult or impossible to align naturally along one of its modes. A concrete example of such data is electronic health records (EHR), our primary motivating application. An EHR dataset contains longitudinal patient information, represented as an event sequence of multiple modalities such as diagnoses, medications, procedures, and lab results. An important characteristic of such event sequences is that there is no simple way to align observations in time across patients. For instance, different patients may have varying length records between the first admission and the most recent hospital discharge; or, two patients whose records' have the same length may still not have a sensible chronological alignment as disease stages and patient progress vary.

For tensor methods, such data poses a significant challenge. Consider the most popular tensor analysis method in data mining, the canonical polyadic (CP) decomposition (also known as PARAFAC or CANDECOMP) [10, 16, 20]. A dataset with three modes might be stored as an $I \times J \times K$ tensor $\mathcal{X}$, which CP then decomposes into a sum of multi-way outer (rank-one) products, $\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{u}_r \circ \mathbf{v}_r \circ \mathbf{w}_r$, where $\mathbf{u}_r, \mathbf{v}_r, \mathbf{w}_r$ are column vectors of size I, J, K, respectively, that effectively represents latent data concepts. Its popularity owes to its *intuitive output structure* and *uniqueness* property that makes the model reliable to interpret [25, 26, 29, 31, 32], as well as the existence of scalable algorithms and software [4, 5, 13]. However, to make, say, the irregular time points in EHR one of the input tensor modes would require finding some way to align time. This fact is an inherent limitation of applying the CP model: any preprocessing to aggregate across time may lose temporal patterns [21, 22, 41], while more sophisticated temporal feature extraction methods typically need continuous and sufficiently long temporal measures to work [36]. Other proposed methods specific to healthcare applications may give some good results [39, 40, 42] but lack the *uniqueness*

Target Rank	10				40			
#nnz(Millions)	63	125	250	500	63	125	250	500
SPARTan	7.4	8.9	11.5	15.4	14	18.4	61	114
Sparse PARAFAC2	24.4	60.1	72.3	194.5	275.2	408.1	**OoM**	**OoM**

Table 1: Running time comparison: Time in minutes of one iteration for increasingly larger datasets (63m to 500m) and fixed target rank (two cases considered: $R = \{10, 40\}$). The mode sizes for the datasets constructed are: 1Mil. subjects, 5K variables and a maximum of 100 observations per subject. OoM (Out of Memory) denotes that the execution failed due to the excessive amount of memory requested. Experiments are conducted on a server with 1TB of RAM.

guarantee; thus, it becomes harder to reliably extract the actual latent concepts as an equivalent arbitrary rotation of them will provide the same fit. All of these weaknesses apply in the EHR scenario outlined above.

In fact, the type of data in the motivating example are quite general: consider that we have K subjects, for which we record J variables and we permit each k-th subject to have I_k observations, which are not necessarily comparable among the different subjects. For this type of data, Harshman proposed the *PARAFAC2 model* [17]. It is a more flexible version of CP: while CP applies the same factors across a collection of matrices, PARAFAC2 instead applies the same factor along one mode and allows the other factor matrix to vary [25]. At the same time, it preserves the desirable properties of CP, such as uniqueness [18, 24, 35, 37]. As shown in Figure 1, PARAFAC2 approximates each one of the input matrices as: $\mathbf{X}_k \approx \mathbf{U}_k\, \mathbf{S}_k\, \mathbf{V}^T$, where $\mathbf{U}_k$ is of size $I_k \times R$, $\mathbf{S}_k$ is a diagonal R-by-R, $\mathbf{V}$ is of size $J \times R$ and R is the target rank of the decomposition.

Despite its applicability, the lack of efficient PARAFAC2 decomposition algorithms has been cited as a reason for its limited popularity [6, 12]. Overall, PARAFAC2 has been mostly used for dense data (e.g., [24]) or sparse data with a small number of subjects [12]. To our knowledge, no work has assessed PARAFAC2 for large-scale sparse data, as well as the challenges arising by doing so.

In this paper, we propose SPARTan (abbreviated from Scalable PARafac Two) to fill this gap, with a focus on achieving scalability on large and sparse datasets. Our methodological advance is a new algorithm for scaling up the core computational kernel arising in the PARAFAC2 fitting algorithm. SPARTan achieves the best of both worlds in terms of speed and memory efficiency: *a)* it is *faster* than a highly-optimized baseline in all cases considered for both real (Figures 5, 6, 7) and synthetic (Table 1) datasets, achieving up to 22× performance gain; *b)* at the same time, SPARTan is *more scalable*, in that it can execute in reasonable time for large problem instances when the baseline fails due to excessive memory consumption (Table 1). We summarize our contributions as:

- **Scalable PARAFAC2 method:** We propose SPARTan, a scalable algorithm fitting the PARAFAC2 model on large and sparse data.
- **Evaluation on various datasets:** We evaluate the scalability of our approach using datasets originating from two different application domains, namely a longitudinal EHR and a time-evolving movie ratings' dataset, which is also publicly available. Additionally, we perform synthetic data experiments.
- **Real-world case study**: We performed a case study of applying SPARTan on temporal phenotyping over medically complex pediatric patients in collaboration with Children's Healthcare of Atlanta (CHOA). The phenotypes and temporal trends discovered were endorsed by a clinical expert from CHOA.

To promote reproducibility, our code is open-sourced and publicly available at: https://github.com/kperros/SPARTan.

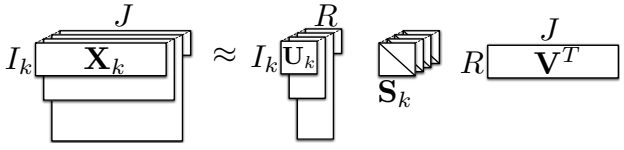


Figure 1: Illustration of the PARAFAC2 model.

2 BACKGROUND

Next we describe the necessary terminology and operations regarding tensors. Then, we provide an overview of the CP model and relevant fitting algorithm. In Table 2, we summarize the notations used throughout the paper.

2.1 Tensors and Tensor Operations

The *order* of a tensor denotes the number of its dimensions, also known as ways or modes (e.g., matrices are 2-order tensors). A *fiber* is a vector extracted from a tensor by fixing all modes but one. For example, a matrix column is a mode-1 fiber. A *slice* is a matrix extracted from a tensor by fixing all modes but two. In particular, the $\mathbf{X}(:, :, k)$ slices of a third-order tensor $\mathcal{X}$ are called the frontal ones and we succinctly denote them as $\mathbf{X}_k$ [25]. *Matricization*, also called *reshaping* or *unfolding*, logically reorganizes tensors into other forms without changing the values themselves. The mode-n matricization of a N-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is denoted by

Symbol	Definition
$\mathcal{X}$, $\mathbf{X}$, $\mathbf{x}$, x	Tensor, matrix, vector, scalar
$\mathbf{X}^\dagger$	Moore-Penrose pseudoinverse
$\mathbf{X}(:, i)$	Spans the entire i-th column of $\mathbf{X}$ (same for tensors)
$\mathbf{X}(i, :)$	Spans the entire i-th row of $\mathbf{X}$ (same for tensors)
$diag(\mathbf{x})$	Diagonal matrix with vector $\mathbf{x}$ on the diagonal
$diag(\mathbf{X})$	Extract diagonal of matrix $\mathbf{X}$
$\mathbf{X}_k$	shorthand for $\mathbf{X}(:, :, k)$ (k-th frontal slice of tensor $\mathcal{X}$)
$\{\mathbf{X}_k\}$	the collection of $\mathbf{X}_k$ matrices, for all valid k
$\mathbf{X}_{(n)}$	mode-n matricization of tensor $\mathcal{X}$
$\circ$	Outer product
$\otimes$	Kronecker product
$\odot$	Khatri-Rao product
$*$	Hadamard (element-wise) product

Table 2: Notations used throughout the paper.

$\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times I_1 I_2 \ldots I_{n-1} I_{n+1} \ldots I_N}$ and arranges the mode-n fibers of the tensor as columns of the resulting matrix.

2.2 CP Decomposition

The CP decomposition [10, 16, 20] of a third-order tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ is its approximation by a sum of three-way outer products:

$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{u}_r \circ \mathbf{v}_r \circ \mathbf{w}_r \tag{1}$$

where $\mathbf{u}_r \in \mathbb{R}^I, \mathbf{v}_r \in \mathbb{R}^J$ and $\mathbf{w}_r \in \mathbb{R}^K$ are column vectors. If we assemble the column vectors $\mathbf{u}_r, \mathbf{v}_r, \mathbf{w}_r$ as: $\mathbf{U} = [\mathbf{u}_1\ \mathbf{u}_2\ \ldots\ \mathbf{u}_R] \in \mathbb{R}^{I \times R}, \mathbf{V} = [\mathbf{v}_1\ \mathbf{v}_2\ \ldots\ \mathbf{v}_R] \in \mathbb{R}^{J \times R}, \mathbf{W} = [\mathbf{w}_1\ \mathbf{w}_2\ \ldots\ \mathbf{w}_R] \in \mathbb{R}^{K \times R}$, then $\mathbf{U}, \mathbf{V}, \mathbf{W}$ are called the *factor matrices*. Interpretation of CP is very intuitive: we consider that the input tensor can be summarized as R latent concepts. Then, for each r-th concept, the vectors $(\mathbf{u}_r, \mathbf{v}_r, \mathbf{w}_r)$ are considered as soft-clustering membership indicators, for the corresponding I, J and K elements of each mode. An equivalent formulation of Relation (1) w.r.t. the frontal slices $\mathbf{X}_k$ of the input tensor $\mathcal{X}$ is [7]:

$$\mathbf{X}_k \approx \mathbf{U}\, \mathbf{S}_k\, \mathbf{V}^T \tag{2}$$

where $k = 1, 2, \ldots, K$ and $\mathcal{S} \in \mathbb{R}^{R \times R \times K}$ is an auxiliary tensor. Each frontal slice $\mathbf{S}_k$ of $\mathcal{S}$ contains the row vector $\mathbf{W}(k,:)$ along its diagonal: $\mathbf{S}_k = diag(\mathbf{W}(k,:))$. Relation (2) provides another viewpoint of interpreting the CP model, through its correspondence to the Singular Value Decomposition (SVD): each slice $\mathbf{X}_k$ is decomposed to a set of factor matrices $\mathbf{U}, \mathbf{V}$ (similar to the singular vectors) which are common for all the slices, and a diagonal middle matrix (similar to the singular values) which varies for each k-th slice. Note, however, that no orthogonality constraints are imposed on $\mathbf{U}, \mathbf{V}$ of the CP model, as in the SVD [12].

Uniqueness. A fundamental property of CP is uniqueness [26, 31]. The issue with non-uniqueness can be exemplified via matrix factorization as follows [25, 29]: If a matrix $\mathbf{X}$ is approximated by the product of $\mathbf{A}\mathbf{B}^T$, then it can also be approximated with the same error by $\mathbf{A}\mathbf{Q}\mathbf{Q}^{-1}\mathbf{B}^T = \tilde{\mathbf{A}}\tilde{\mathbf{B}}^{\mathbf{T}}$, for any invertible $\mathbf{Q}$. Thus, we can easily construct two completely different sets of rank-one factors that sum to the original matrix. Inevitably, this hurts interpretability, since we cannot know whether our solution is an arbitrarily rotated version of the actual latent factors. In contrast to matrix factorization or Tucker decomposition [25], Kruskal [26] proved that CP is unique, under the condition: $k_{\mathbf{U}} + k_{\mathbf{V}} + k_{\mathbf{W}} \geq 2R + 2$, where $k_{\mathbf{U}}$ is the k-rank of $\mathbf{U}$, defined as the maximum value k such that any k columns are linearly independent. The only exception is related to elementary indeterminacies of scaling and permutation of the component vectors [25, 32]. In sum, the CP decomposition is pursuing the true underlying latent information of the input tensor and provides reliable interpretation for unsupervised approaches.

Fitting the CP model. Perhaps the most popular algorithm for fitting the CP model is the CP-Alternating Least Squares (CP-ALS) [10, 16], listed in Algorithm 1. The main idea is to solve for one factor matrix at a time, by fixing the others. In that way, each subproblem is reduced to a linear least-squares problem. In case the input tensor contains non-negative values, a non-negative least-squares solver (e.g., [9]) can be used instead of an unconstrained one, to further improve the factors' interpretability [6].

Due to the ever increasing need for CP decompositions in data mining, the parallel CP-ALS for sparse tensors has been extensively studied in the recent literature for both single-node and distributed settings (e.g., [4, 11, 14, 23, 28, 34]). A pioneering work in addressing scalability issues for sparse tensors was provided by Bader and Kolda [4] [1]. The authors identified and scaled up the algorithm's bottleneck, which is the materialization of the Matricized-Tensor-Times-Khatri-Rao-Product (MTTKRP). For example, in Algorithm 1, the MTTKRP corresponds to the computation of $\mathbf{X}_{(1)}(\mathbf{W} \odot \mathbf{V})$ when solving for $\mathbf{U}$. For large and sparse tensors, a naive construction of the MTTKRP requires huge storage and computational cost and has to be avoided.

Algorithm 1 CP-ALS

Require: $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ and target rank R
Ensure: $\lambda \in \mathbb{R}^R$, $\mathbf{U} \in \mathbb{R}^{I \times R}$, $\mathbf{V} \in \mathbb{R}^{J \times R}$, $\mathbf{W} \in \mathbb{R}^{K \times R}$
1: Initialize $\mathbf{V}$, $\mathbf{W}$
2: **while** convergence criterion is not met **do**
3: $\quad \mathbf{U} \leftarrow \mathbf{X}_{(1)}(\mathbf{W} \odot \mathbf{V})(\mathbf{W}^T\mathbf{W} * \mathbf{V}^T\mathbf{V})^\dagger$
4: $\quad$ Normalize columns of $\mathbf{U}$
5: $\quad \mathbf{V} \leftarrow \mathbf{X}_{(2)}(\mathbf{W} \odot \mathbf{U})(\mathbf{W}^T\mathbf{W} * \mathbf{U}^T\mathbf{U})^\dagger$
6: $\quad$ Normalize columns of $\mathbf{V}$
7: $\quad \mathbf{W} \leftarrow \mathbf{X}_{(3)}(\mathbf{V} \odot \mathbf{U})(\mathbf{V}^T\mathbf{V} * \mathbf{U}^T\mathbf{U})^\dagger$
8: $\quad$ Normalize columns of $\mathbf{W}$ and store norm in λ
9: **end while**

3 PARAFAC2 OVERVIEW & CHALLENGES

3.1 Model

As we introduced in Section 1, the PARAFAC2 model [17] can successfully deal with an incomparable mode of each slice $\mathbf{X}_k$ [24]. It does so, by introducing a set of $\mathbf{U}_k$ matrices replacing the $\mathbf{U}$ matrix of the CP model in Relation (2). Thus, each slice $\mathbf{X}_k$ is decomposed as shown in Figure 1:

$$\mathbf{X}_k \approx \mathbf{U}_k\, \mathbf{S}_k\, \mathbf{V}^T \tag{3}$$

where $k = 1, \ldots, K, \mathbf{U}_k \in \mathbb{R}^{I_k \times R}, \mathbf{S}_k \in \mathbb{R}^{R \times R}$ is diagonal and $\mathbf{V} \in \mathbb{R}^{J \times R}$. To preserve uniqueness, Harshman [17] imposed the constraint that the cross product $\mathbf{U}_k^T\mathbf{U}_k$ is invariant regardless which subject k is involved [12, 25]. In that way, the CP model's invariance of the factor $\mathbf{U}_k$ itself (or $\mathbf{U}$ given its invariance to k), is relaxed [2]. For the above constraint to hold, each $\mathbf{U}_k$ factor is decomposed as:

$$\mathbf{U}_k = \mathbf{Q}_k H \tag{4}$$

where $\mathbf{Q}_k$ is of size $I_k \times R$ and has orthonormal columns, and $\mathbf{H}$ is an $R \times R$ matrix, which does not vary by k [25]. Then, the constraint that $\mathbf{U}_k^T\mathbf{U}_k$ is constant over k is implicitly enforced, as follows: $\mathbf{U}_k^T\mathbf{U}_k = \mathbf{H}^T\mathbf{Q}_k^T\mathbf{Q}_k\mathbf{H} = \mathbf{H}^T\mathbf{H} = \mathbf{\Phi}$.

There have been several results regarding the uniqueness property of PARAFAC2 [18, 24, 37]. The most relevant [18] towards our large-scale data scenario (i.e., the number of K subjects can easily reach the order of hundreds of thousands) is that a rank-R PARAFAC2 model is unique if $\mathbf{\Phi}$ and $\mathbf{V}$ have rank R, $\mathbf{\Phi}$ has no zero entries and the number of K subjects is at least: $R\,(R+1)\,(R+2)\,(R+3)\ /\ 24$ [19]. Note that this bound on the number of K subjects is

[1] The contributions of [4], among others, are summarized as the Tensor Toolbox [5], which is widely acclaimed as the state-of-the-art package for single-node sparse tensor operations and algorithms.

sufficient but not necessary to achieve uniqueness; it is conjectured (as evaluated through simulation studies) that PARAFAC2 provides unique solutions as long as $K \geq 4$ [19, 24, 35].

3.2 Classical Algorithm for PARAFAC2

Below, we overview the classical algorithm for fitting PARAFAC2, proposed by Kiers et al. [24]. Their original algorithm expects dense data as input. Its objective function is as follows:

$$\min_{\{\mathbf{U}_k\},\{\mathbf{S}_k\},\mathbf{V}} \sum_{k=1}^{K} ||\mathbf{X}_k - \mathbf{U}_k\mathbf{S}_k\mathbf{V}^T||_F^2$$

subject to: $\mathbf{U}_k = \mathbf{Q}_k\mathbf{H}$, $\mathbf{Q}_k^T\mathbf{Q}_k = \mathbf{I}$ and $\mathbf{S}_k$ to be diagonal. Algorithm 2 follows an Alternating Least Squares (ALS) approach, divided in two distinct steps: first (lines 3-6), the set of column-orthonormal matrices $\{\mathbf{Q}_k\}$ is computed by fixing $\mathbf{H}, \mathbf{V}, \{\mathbf{S}_k\}$. This step can be derived by examining the pursuit of each $\mathbf{Q}_k$ as an individual Orthogonal Procrustes Problem [15] of the form:

$$\min_{\mathbf{Q}_k} ||\mathbf{X}_k - \mathbf{Q}_k\mathbf{H}\mathbf{S}_k\mathbf{V}^T||_F^2 \quad (5)$$

subject to $\mathbf{Q}_k^T\mathbf{Q}_k = \mathbf{I}$. Given the SVD of $\mathbf{H}\mathbf{S}_k\mathbf{V}^T\mathbf{X}_k^T$ as $\mathbf{P}_k\Sigma_k\mathbf{Z}_k$, the minimum of objective (5) over column-orthonormal $\mathbf{Q}_k$ is given by $\mathbf{Q}_k = \mathbf{Z}_k\mathbf{P}_k^T$ [15, 24].

Second, after solving for and fixing $\{\mathbf{Q}_k\}$, we find a solution for the rest of the factors as:

$$\min_{\mathbf{H},\{\mathbf{S}_k\},\mathbf{V}} \sum_{k=1}^{K} ||\mathbf{Q}_k^T\mathbf{X}_k - \mathbf{H}\mathbf{S}_k\mathbf{V}^T||_F^2 \quad (6)$$

where $\mathbf{S}_k$ is diagonal. Note the equivalence of the above objective with the CP "slice-wise" formulation of Relation (2). This equivalence implies that minimizing the objective (6) is achieved by executing the CP decomposition on a tensor $\mathcal{Y} \in \mathbb{R}^{R\times J\times K}$ with frontal slices $\mathbf{Y}_k = \mathbf{Q}_k^T\mathbf{X}_k$ (lines 7-10). In order to avoid executing all the costly CP iterations, Kiers et al. [24] propose to run a single CP-ALS iteration, since this suffices to decrease the objective.

The PARAFAC2 model can be extended so that non-negative constraints are imposed on $\{\mathbf{S}_k\}, \mathbf{V}$ factors [7]. This is a property inherited by the CP-ALS iteration, where we can constrain the factors $\mathbf{V}$ and $\mathbf{W}$ to be non-negative, as discussed in Section 2.2. Note that constraining the $\{\mathbf{U}_k\}$ factors to be non-negative as well is not as simple, and a naive approach would violate the model properties [8].

3.3 Challenges of PARAFAC2 on sparse data

Next, we summarize a set of crucial observations regarding the computational challenges, when executing Algorithm 2 on large, sparse data:

Bottleneck of Algorithm 2. Regarding the 1st step (lines 3-6), in practice for sparse $\mathbf{X}_k$, each one of the K sub-problems scales as $O(min(RI^2, R^2I))$, due to the SVD involved [38], where I is an upper bound for I_k. Note that this computation can be trivially parallelized for all K subjects. On the other hand, the 2nd step (lines 7-10) is dominated by the MTTKRP computation (which as we discussed in Section 2 is the bottleneck of sparse CP-ALS). Thus, it scales as $3R\ nnz(\mathcal{Y})$ [34], using state-of-the-art sparse tensor libraries for single-node [4]. Given that none of the input matrices $\mathbf{X}_k$ is completely zero, then: $3R\ nnz(\mathcal{Y}) \geq 3KR^2$. As a result, this step becomes the bottleneck of Algorithm 2 for large and sparse "irregular" tensors, since it cannot be parallelized w.r.t. the K subjects, as trivially happens with the 1st one.

Algorithm 2 PARAFAC2-ALS [24]

Require: $\{\mathbf{X}_k \in \mathbb{R}^{I_k\times J}\}$ for $k = 1, \ldots, K$ and target rank R
Ensure: $\{\mathbf{U}_k \in \mathbb{R}^{I_k\times R}\}$, $\{\mathbf{S}_k \in \mathbb{R}^{R\times R}\}$ for $k = 1, \ldots, K$, $\mathbf{V} \in \mathbb{R}^{J\times R}$
1: Initialize $\mathbf{H} \in \mathbb{R}^{R\times R}$, $\mathbf{V}$, $\{\mathbf{S}_k\}$ for $k = 1, \ldots, K$
2: **while** convergence criterion is not met **do**
3: **for** $k = 1, \ldots, K$ **do**
4: $[\mathbf{P}_k, \Sigma_k, \mathbf{Z}_k] \leftarrow$ truncated SVD of $\mathbf{H}\mathbf{S}_k\mathbf{V}^T\mathbf{X}_k^T$ at rank R
5: $\mathbf{Q}_k \leftarrow \mathbf{Z}_k\mathbf{P}_k^T$
6: **end for**
7: **for** k=1, …, K **do**
8: $\mathbf{Y}_k \leftarrow \mathbf{Q}_k^T\mathbf{X}_k$ // *construct slice* $\mathbf{Y}_k$ *of tensor* $\mathcal{Y}$
9: **end for**
10: Run a single iteration of CP-ALS on $\mathcal{Y}$ to compute $\mathbf{H}$, $\mathbf{V}$, $\mathbf{W}$
11: **for** $k = 1, \ldots, K$ **do**
12: $\mathbf{S}_k \leftarrow diag(\mathbf{W}(k, :))$
13: **end for**
14: **end while**
15: **for** $k = 1, \ldots, K$ **do**
16: $\mathbf{U}_k \leftarrow \mathbf{Q}_k\mathbf{H}$ // *Assemble* $\mathbf{U}_k$
17: **end for**

Imbalance of mode sizes of $\mathcal{Y}$. The size of the intermediate tensor $\mathcal{Y}$ formed is $R \times J \times K$. For large-scale data, we expect that $R << K, J$ since R corresponds to the target rank of the overall PARAFAC2 decomposition. Note that this property of "size imbalance" may not hold for a general large tensor, thus generic CP-ALS solvers (e.g., [4]) cannot exploit it.

Structured sparsity of $\{\mathbf{X}_k\}$. First, we observe that even if the input slices $\{\mathbf{X}_k \in \mathbb{R}^{I_k\times J}\}$ are very sparse, all their I_k rows will contain at least one non-zero element. If this is not the case, we can simply filter the zero ones, without affecting the result. However, this does not hold for the J columns of each $\mathbf{X}_k$, which have to be aligned across all K subjects. A direct consequence of that in real datasets is that very few variables (relative to their total number) are typically recorded for each subject. For example, in the EHR data use case introduced in Section 1, very few medical features are recorded for each patient.

Driven by this observation, we are motivated to computationally exploit any column sparsity (i.e., cases where many columns will be completely zero) of each one of the input matrices $\mathbf{X}_k$.

4 THE SPARTAN APPROACH

4.1 Overview

Motivated by the challenges presented in Section 3, we propose a specialized version of the Matricized-Tensor-Times-Khatri-Rao-Product (MTTKRP) kernel, specifically targeting the intermediate tensor $\mathcal{Y} \in \mathbb{R}^{R\times J\times K}$ formed within the PARAFAC2-ALS algorithm. We first provide an overview of the properties that our approach exhibits:

- It is *fully parallelizable w.r.t. the K subjects*. This property is crucial towards scaling up for large-scale "irregular" tensors.
- It *exploits the structured sparsity* of the input frontal slices $\{\mathbf{X}_k\}$. This is possible due to the observation that the k-th frontal slice $\mathbf{Y}_k = \mathbf{Q}_k^T\mathbf{X}_k$ of $\mathcal{Y}$ follows precisely the column sparsity pattern of $\mathbf{X}_k$. In particular, if c_k is the number of columns of $\mathbf{X}_k$ containing

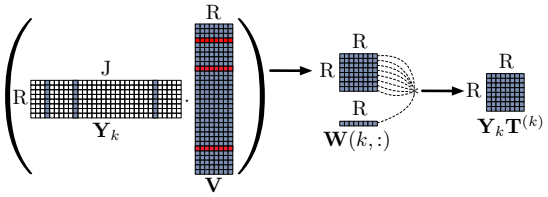


Figure 2: SPARTan computations for the MTTKRP w.r.t. the 1st mode. For each k-th partial result of Equation (8), we only use the rows of V factor matrix corresponding to the non-zero columns of $\mathbf{Y}_k$. For each of the R rows of the resulting matrix, we compute the Hadamard product with $\mathbf{W}(k,:)$, which is the k-th row of the factor matrix W. The described computations fulfill all of the desirable properties presented in Section 4.1.

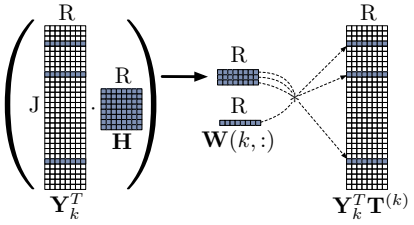


Figure 3: SPARTan computations for the MTTKRP w.r.t. the 2nd mode. For each k-th partial result of Equation (12), we perform the vector-matrix multiplications for each non-zero row of $\mathbf{Y}_k^T$. Then, for each intermediate vector, the Hadamard product with $\mathbf{W}(k,:)$ is computed. Finally, we distribute the vectors to their corresponding positions in $\mathbf{Y}_k^T\mathbf{T}^{(k)}$. As in the case w.r.t. the 1st mode, we limit computations to the necessary ones corresponding to the non-zero columns of $\mathbf{Y}_k$ and all the properties presented in Section 4.1 are preserved.

at least one non-zero element, then $\mathbf{Y}_k$ will contain R c_k non-zero elements located in the positions of the non-zero columns of X_k. Exploiting structured sparsity is indispensable towards minimizing intermediate data and computations to the absolutely necessary ones.

- As a by-product of the above, SPARTan *avoids unnecessary data re-organization* (tensor reshaping/permutations), since all operations are formulated w.r.t. the frontal slices $\mathbf{Y}_k$ of tensor $\mathcal{Y}$. In fact, our approach never forms the tensor $\mathcal{Y}$ explicitly and directly utilizes the available collection of matrices $\{\mathbf{Y}_k\}$ instead.

4.2 Methodology

In the following, we describe the design of our MTTKRP kernel for each one of the tensor modes. We use the notation $\mathbf{M}^{(i)}$ to denote the MTTKRP corresponding to the i-th tensor mode. Note that our factor matrices are: $\mathbf{H} \in \mathbb{R}^{R\times R}, \mathbf{V} \in \mathbb{R}^{J\times R}$ and $\mathbf{W} \in \mathbb{R}^{K\times R}$ as in Line 10 of Algorithm 2.

Mode-1 MTTKRP. First, we re-visit the MTTKRP equation:

$$\mathbf{M}^{(1)} = \mathbf{Y}_{(1)}\,(\mathbf{W} \odot \mathbf{V}), \tag{7}$$

where $\mathbf{M}^{(1)} \in \mathbb{R}^{R\times R}, \mathbf{Y}_{(1)} \in \mathbb{R}^{R\times KJ}$. In order to attempt to parallelize the above computation w.r.t. the K subjects, we define the matrix $\mathbf{T}^{(k)} \in \mathbb{R}^{J\times R}$ to denote the k-th vertical block of the Khatri-Rao Product $\mathbf{W} \odot \mathbf{V} \in \mathbb{R}^{KJ\times R}$:

$$\mathbf{W} \odot \mathbf{V} = \begin{bmatrix} \mathbf{T}^{(1)} \\ \mathbf{T}^{(2)} \\ \vdots \\ \mathbf{T}^{(K)} \end{bmatrix}$$

We then remark that $\mathbf{Y}_{(1)}$ (i.e., mode-1 matricization of $\mathcal{Y}$) consists of an horizontal concatenation of the tensor's frontal slices $\mathbf{Y}_k$. Thus, we exploit the fact that the matrix multiplication in Equation (7) can be expressed as the sum of outer products or more generally, as a sum of block-by-block matrix multiplications:

$$\mathbf{M}^{(1)} = \sum_{k=1}^{K} \mathbf{Y}_k\ \mathbf{T}^{(k)} \tag{8}$$

Through Equation (8), the computation can be easily parallelized over K independent sub-problems and then sum the partial results. This directly utilizes the frontal slices $\mathbf{Y}_k$ without further tensor organization. However, it constructs the whole Khatri-Rao Product (in the form of blocks $\mathbf{T}^{(k)}$). In order to avoid that, we first state an expression for each i-th row of $\mathbf{T}^{(k)}$, which is a direct consequence of the Khatri-Rao Product definition:

$$\mathbf{T}^{(k)}(i,:) = \mathbf{V}(i,:) * \mathbf{W}(k,:), \tag{9}$$

where $*$ stands for the Hadamard (element-wise) product. Then, we express the j-th row of each partial result of Equation (8) as follows:

$$\begin{aligned} [\mathbf{Y}_k\ \mathbf{T}^{(k)}]_{j,:} &= \mathbf{Y}_k(j,:)\ \mathbf{T}^{(k)} \\ &\overset{Eq.(9)}{=} \sum_i \mathbf{Y}_k(j,i) * (\mathbf{V}(i,:) * \mathbf{W}(k,:)) \\ &\overset{(a)}{=} \left(\sum_i \mathbf{Y}_k(j,i) * \mathbf{V}(i,:)\right) * \mathbf{W}(k,:) \\ &\overset{(b)}{=} (\mathbf{Y}_k(j,:)\ \mathbf{V}) * \mathbf{W}(k,:), \end{aligned} \tag{10}$$

where (a) stems from the associative property of the Hadamard product and the fact that $\mathbf{W}(k,:)$ is independent of the summation and (b) from the calculation of matrix multiplication as a sum of outer-products (in particular, we encounter the sub-case of vector-matrix product).

Equation (10) suggests an efficient way to compute the partial results of Equation (8), which we illustrate in Figure 2. First, we compute the matrix product $\mathbf{Y}_k\mathbf{V}$ and for each row of the intermediate result of size $R \times R$, we compute the Hadamard product with $\mathbf{W}(k,:)$. Note that, as we discussed in Section 3, $\mathbf{Y}_k$ is expected to be column-sparse in practice, thus multiplying by $\mathbf{V}$ uses only those rows of $\mathbf{V}$ corresponding to the non-zero columns of $\mathbf{Y}_k$. Thus, we avoid the redundant and expensive computation of the full Khatri-Rao Product. Overall, the methodology described above enjoys all of the properties described in Section 4.1.

Mode-2 MTTKRP. The methodology followed for the Mode-2 case is similar to the one described for the 1st case. We state the corresponding MTTKRP equation:

$$\mathbf{M}^{(2)} = \mathbf{Y}_{(2)}\,(\mathbf{W} \odot \mathbf{H}) \tag{11}$$

where $\mathbf{M}^{(2)} \in \mathbb{R}^{J\times R}, \mathbf{Y}_{(2)} \in \mathbb{R}^{J\times RK}$. The main remark is that $\mathbf{Y}_{(2)}$ consists of an horizontal concatenation of the *transposed* frontal slices $\{\mathbf{Y}_k\}$ of the intermediate tensor $\mathcal{Y}$. Thus, if we denote as $\mathbf{T}^{(k)}$ the k-th vertical block of the Khatri-Rao Product $\mathbf{W} \odot \mathbf{H}$, we can

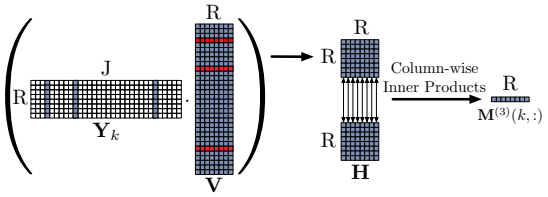


Figure 4: SPARTan computations for the MTTKRP w.r.t. the 3rd mode. We compute each row of the result $\mathbf{M}^{(3)}(k,:)$ independently of others, enabling parallelization w.r.t. the K subjects. As in mode-1, mode-2 cases, we exploit the column sparsity of $\mathbf{Y}_k$. In this case, we also leverage that H is a small R-by-R matrix in practice (due to the "size imbalance" of the intermediate tensor $\mathcal{Y}$). Thus, it is efficient to delay any computations on H until the R-by-R product of $\mathbf{Y}_k\mathbf{V}$ is formed, and then take column-wise inner products between those two matrices. The described operations fulfill all the properties outlined in Section 4.1.

formulate the problem as:

$$\mathbf{M}^{(2)} = \sum_{k=1}^{K} \mathbf{Y}_k^T \, \mathbf{T}^{(k)} \tag{12}$$

Given the above, it is easy to extend Equation (10) for this case, so as to compute a single row of each partial result of Equation (12):

$$[\mathbf{Y}_k^T \, \mathbf{T}^{(k)}]_{j,:} = \left(\mathbf{Y}_k(:,j)^T \, \mathbf{H}\right) * \mathbf{W}(k,:) \tag{13}$$

The corresponding operations are illustrated in Figure 3. A crucial remark is that we can focus on computing the relevant intermediate results only for the non-zero rows of $\mathbf{Y_k}^T$, since the rest of the rows of the result $\mathbf{Y_k}^T\mathbf{T}^{(k)}$ will be zero. In sum, we again avoid redundant computations of the full Khatri-Rao Product and preserve all of the properties described in Section 4.1.

Mode-3 MTTKRP. First, we state the equation regarding the Mode-3 case:

$$\mathbf{M}^{(3)} = \mathbf{Y}_{(3)} \, (\mathbf{V} \odot \mathbf{H}) \tag{14}$$

where $\mathbf{M}^{(3)} \in \mathbb{R}^{K \times R}$ and $\mathbf{Y}_{(3)} \in \mathbb{R}^{K \times JR}$. Note that in this case, we are pursuing the MTTKRP of the mode corresponding to the K subjects. Thus, an entirely different approach than the Mode-1, Mode-2 cases is needed so that we construct efficient independent sub-problems for each one of them. In particular, we need to design each k-th subproblem so that it computes the k-th row of $\mathbf{M}^{(3)}$. In addition, we want to operate only on $\{\mathbf{Y}_k\}$ without forming and reshaping the tensor $\mathcal{Y}$, as well as to exploit the frontal slices' sparsity. To tackle the challenges above, we leverage the fact that [14]:

$$\mathbf{M}^{(3)}(:,r) = \begin{bmatrix} \mathbf{H}(:,r)^T \, \mathbf{Y}_1 \, \mathbf{V}(:,r) \\ \vdots \\ \mathbf{H}(:,r)^T \, \mathbf{Y}_K \, \mathbf{V}(:,r) \end{bmatrix} \tag{15}$$

Then, we remark that in order to retrieve a certain element of the matrix $\mathbf{M}^{(3)}$, we have:

$$\begin{aligned} \mathbf{M}^{(3)}(k,r) &= \mathbf{H}(:,r)^T \, \mathbf{Y}_k \, \mathbf{V}(:,r) \\ &= \mathbf{H}(:,r)^T \, [\mathbf{Y}_k\mathbf{V}](:,r) \end{aligned}$$

The last line above reflects the inner product between the corresponding r-th columns of $\mathbf{H}$ and $[\mathbf{Y}_k \, \mathbf{V}]$, respectively. Thus, in order to retrieve a row $\mathbf{M}(k,:)$, we can simply operate as:

$$\mathbf{M}^{(3)}(k,:) = dot\,(\mathbf{H}, \mathbf{Y}_k\mathbf{V}) \tag{16}$$

Dataset	K	J	$max(I_k)$	#nnz
CHOA	464,900	1,328	166	12.3 Mil.
MovieLens	25,249	26,096	19	8.9 Mil.

Table 3: Summary statistics for the real datasets of our experiments. K is the number of subjects, J is the number of variables, I_k is the number of observations for the k-th subject and #nnz corresponds to the total number of non-zeros.

where the $dot()$ function extracts the inner product of the corresponding columns of its two matrix arguments. We illustrate this operation in Figure 4. Since $\mathbf{H}$ is a small R-by-R matrix (due to the tensor's "size imbalance"), it is very efficient to delay any computations on $\mathbf{H}$ until the R-by-R intermediate matrix is formed as a product of $\mathbf{Y}_k$ $\mathbf{V}$. Then, we simply take the column-wise inner products between those two R-by-R matrices. In that way, all the desirable properties we mentioned in Section 4.1 are also fulfilled.

In Algorithm 3, we list the pseudocode corresponding to the methodology proposed. Note that in lines 8,16, we can accumulate over the partial results in parallel, since the summation is independent of the iteration order.

Algorithm 3 MTTKRP for SPARTan

Require: $\{\mathbf{Y}_k \in \mathbb{R}^{R \times J}\}$ for $k = 1, \ldots, K$, $\mathbf{H} \in \mathbb{R}^{R \times R}$, $\mathbf{V} \in \mathbb{R}^{J \times R}$, $\mathbf{W} \in \mathbb{R}^{K \times R}$, the target rank R and the mode n for which we are computing the MTTKRP
Ensure: $\mathbf{M}^{(n)}$
1: Initialize $\mathbf{M}^{(n)}$ with zeros
2: **if** $n == 1$ **then**
3: **for** $k = 1, \ldots, K$ **do**
4: $temp \leftarrow \mathbf{Y}_k V$
5: **for** $r = 1, \ldots, R$ **do**
6: $temp(r,:) \leftarrow temp(r,:) * \mathbf{W}(k,:)$
7: **end for**
8: $\mathbf{M}^{(1)} \leftarrow \mathbf{M}^{(1)} + temp$ // *sum in parallel* $\forall k = 1, \ldots, K$
9: **end for**
10: **else if** $n == 2$ **then**
11: **for** $k = 1, \ldots, K$ **do**
12: Initialize $temp \in \mathbb{R}^{J \times R}$ with zeros
13: **for** each j-th non-zero column of $\mathbf{Y}_k$ **do**
14: $temp(j,:) \leftarrow (\mathbf{Y}_k(:,j)^T \mathbf{H}) * \mathbf{W}(k,:)$
15: **end for**
16: $\mathbf{M}^{(2)} \leftarrow \mathbf{M}^{(2)} + temp$ // *sum in parallel* $\forall k = 1, \ldots, K$
17: **end for**
18: **else if** $n == 3$ **then**
19: **for** $k = 1, \ldots, K$ **do**
20: $\mathbf{M}^{(3)}(k,:) \leftarrow dot\,(\mathbf{H}, \mathbf{Y}_k\mathbf{V})$ // *in parallel* $\forall k = 1, \ldots, K$
21: **end for**
22: **end if**

5 EXPERIMENTS

5.1 Setup

Real Data Description. Table 3 provides summary statistics regarding the real datasets used.

The **CHOA** (Children Healthcare of Atlanta) dataset corresponds to EHRs of pediatric patients with at least 2 hospital visits. For each patient, we utilize the diagnostic codes and medication categories from their records, as well as the provided age of the patient (in

days) at the visit time. The available International Classification of Diseases (ICD9) [33] codes are summarized to Clinical Classification Software (CCS) [1] categories, which is a standard step in healthcare analysis improving interpretability and clinical meaningfulness. We aggregate the time mode by week and all the medical events over each week are considered as a single observation. The resulting data are of 464,900 subjects by 1,328 features by maximum 166 observations with 12.3m non-zeros.

MovieLens 20M is another real dataset we used, which is *publicly available* [2]. We are motivated to use this dataset, because of the importance of the evolution of user preferences over time, as highlighted in recent literature [27]. For this dataset, we consider that each year of ratings corresponds to a certain observation; thus, for each user, we have a year-by-movie matrix to describe her rating activity. We consider only the users having at least 2 years of ratings.

Implementation details. We used MatlabR2015b for our implementations, along with functionalities for sparse tensors from the Tensor Toolbox [5] and the Non-Negative Least Squares (NNLS) approach [9] from the N-way Toolbox [3] [3]. In both the SPARTan and the baseline implementations, we adjust the CP-ALS iteration arising in the PARAFAC2-ALS, so that non-negative constraints are imposed on the $\{\mathbf{S}_k\}, \mathbf{V}$ factors, as discussed in Section 3.2.

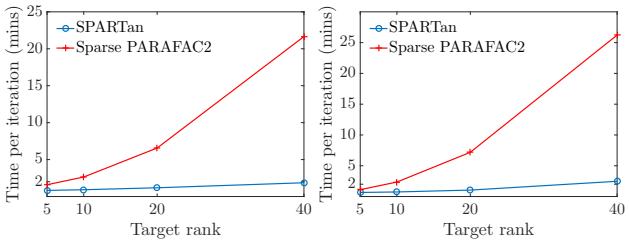
The baseline method corresponds to the standard fitting algorithm for the PARAFAC2 model [24] adjusted for sparse tensors as in [12]. We utilized the implementation from the most recent version of the Tensor Toolbox [5] regarding both the manipulation of sparse tensors, as well as the CP-ALS iteration arising in the PARAFAC2-ALS.

Parallelism. We exploit the capabilities of the Parallel Computing Toolbox of Matlab, by utilizing its parallel pool in both SPARTan and the baseline approach, whenever this is appropriate. Regarding the size of the parallel pool, the number of workers of all the experiments regarding a certain dataset is fixed. For the movie-rating dataset we used the default of 12 workers. For the synthetic and the CHOA datasets, we increased the number of workers to 20 because of the data size increase.

Hardware. We conducted our experiments on a server running Ubuntu 14.04 with 1TB of RAM and four Intel E5-4620 v4 CPU's with a maximum clock frequency of 2.10GHz. Each one of the processors contains 10 cores, and each one of the cores can exploit 2 threads with hyper-threading enabled.

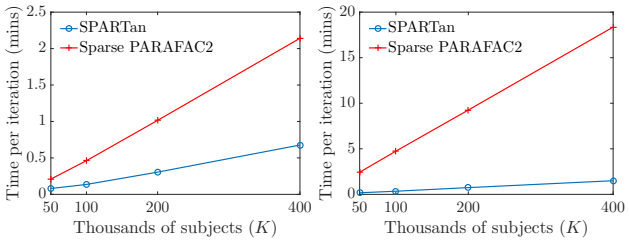
5.2 SPARTan is fast and memory-efficient

Synthetic Data. We assess the scalability of the approaches under comparison for sparse synthetic data. We considered a setup with $1,000,000$ subjects, $5,000$ variables and a maximum of 100 observations for each subject. The number of observations I_k for each subject is dependent on the number of rows of $\mathbf{X}_k$ containing non-zero elements; thus, I_k increases with the dataset density. Indicatively, the mean number of observations I_k for the sparsest dataset created ($\approx$ 63 mil.) is 46.9 and for the densest ($\approx$ 500 mil.) dataset, the mean I_k is 99.3. We randomly construct the factors of



(a) CHOA dataset (b) MovieLens dataset

Figure 5: Time in minutes for one iteration (as an average over 10) for varying target rank for both the real datasets used. SPARTan achieves up to 12× and 11× speedup over the baseline approach for the CHOA and the MovieLens datasets respectively.



(a) Target Rank $R = 10$ (b) Target Rank $R = 40$

Figure 6: CHOA dataset: Time in minutes for one iteration (as an average over 10) for varying number of subjects (K) included and fixed target rank (two cases considered: $R = \{10, 40\}$).

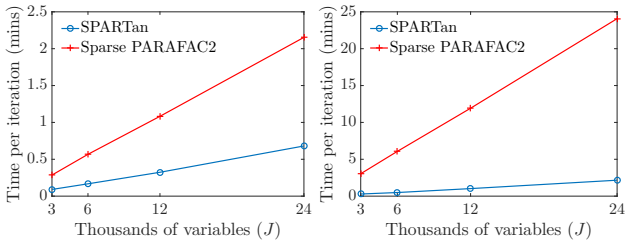
a rank-40 (which is the maximum target rank used in our experiments) PARAFAC2 model. Based on this model, we construct the input slices $\{\mathbf{X}_k\}$, which we then sparsify uniformly at random, for each sparsity level. The density of the sparsification governs the number of non-zeros of the collection of input matrices.

We provide the results in Table 1. First, we remark that SPARTan is *both more scalable* and *faster* than the baseline. In particular, the baseline approach fails to execute in the two largest problem instances for target rank $R = 40$, due to out of memory problems, during the creation of the intermediate sparse tensor $\mathcal{Y}$. Note that as we discussed in Section 4, SPARTan avoids the additional overhead of explicitly constructing a sparse tensor structure, since it only operates directly on the tensor's frontal slices $\{\mathbf{Y}_k\}$. Regarding the baseline's memory issue, since the density of $\mathcal{Y}$ may grow (e.g., $\approx$ 10% in the densest case), we also attempted to store the intermediate tensor $\mathcal{Y}$ as a dense one. However, this also failed, since the memory requested for a dense tensor of size 40-by-5K-by-1Mil. exceeded the available RAM of our system (1TB). Overall, it is clear that the baseline approach cannot fully exploit the input sparsity. On the contrary, SPARTan properly executes for all the problem instances considered in a reasonable amount of time. In particular, for $R = 40$, SPARTan is up to 22× faster than the baseline. Even for a lower target rank of $R = 10$, SPARTan achieves up to 13× faster computation.

Real Data. We evaluate the scalability of the proposed SPARTan approach against the baseline method for the real datasets as well.

[2] https://grouplens.org/datasets/movielens/

[3] We also accredit the dense PARAFAC2 implementation by Rasmus Bro, from where we have adapted many functionalities.



(a) Target Rank $R = 10$ (b) Target Rank $R = 40$

Figure 7: MovieLens dataset: Time in minutes for one iteration (as an average over 10) for varying number of variables (J) included and fixed target rank (two cases considered: $R = \{10, 40\}$).

In Figures 5, 6, 7, we present the results of the corresponding experiments. First, we target the full datasets and vary the pursued target rank (Figure 5). Note that for both datasets considered, the time per iteration of the baseline approach increases dramatically as we increase the target rank. On the contrary, the time required by SPARTan increases only slightly. Overall, our approach achieves up to over an order of magnitude gain regarding the time required per epoch for both datasets.

We also evaluate the scalability of the methods under comparison as we vary the subjects and the variables considered. Since the CHOA dataset (Figure 6) contains more subjects than variables, we vary the number of subjects for this dataset for two fixed target ranks (10, 40). In both cases, SPARTan scales better than the baseline. As concerns the MovieLens dataset (Figure 7), since it contains more variables than subjects, we examine the scalability w.r.t. increasing subsets of variables considered. In this case as well, we remark the favorable scalability properties of SPARTan, rendering it practical to use for large and sparse "irregular" tensors.

5.3 Phenotype discovery on CHOA EHR Data

Motivation. Next we demonstrate the usefulness of PARAFAC2 towards temporal phenotyping of EHRs. Phenotyping refers to the process of extracting meaningful patient clusters (i.e., phenotypes) out of raw, noisy Electronic Health Records [30]. An open challenge in phenotyping is to capture temporal trends or patterns regarding the evolution of those phenotypes for each patient over time. Below, we illustrate how SPARTan can be used to successfully tackle this challenge.

Model Interpretation: We propose the following model interpretation towards the target challenge:

- The common factor matrix $\mathbf{V}$ reflects the *phenotypes' definition* and the non-zero values of each r-th column indicate the membership of the corresponding medical feature to the r-th phenotype.
- The diagonal $\mathbf{S}_k$ provides the *importance membership indicators* of the k-th subject to each one of the R phenotypes/clusters. Thus, we can sort the R phenotypes based on the values of vector $diag(\mathbf{S}_k)$ and identify the most relevant phenotypes for the k-th subject.
- Each $\mathbf{U}_k$ factor matrix provides the *temporal signature* of each patient: each r-th column of $\mathbf{U}_k$ reflects the evolution of the expression of the r-th phenotype for all the I_k weeks of her medical history. Note that since all $\mathbf{X}_k, \mathbf{S}_k, \mathbf{V}$ matrices are non-negative, we only consider the non-negative elements of the temporal signatures in our interpretation.

Table 4: Phenotypes discovered by PARAFAC2. The title annotation for each phenotype is provided by the medical expert. The red color corresponds to diagnoses and the blue color corresponds to medications.

Cancer	**Weight**
Chemotherapy	0.35
Leukemias [39.]	0.27
Immunity disorders [57.]	0.23
HEPARIN AND RELATED PREPARATIONS	0.6
ANTIEMETIC/ANTIVERTIGO AGENTS	0.34
SODIUM/SALINE PREPARATIONS	0.32
TOPICAL LOCAL ANESTHETICS	0.19
ANTIHISTAMINES - 1ST GENERATION	0.16
Sickle Cell Anemia (SCA)	**Weight**
Sickle cell anemia [61.]	0.73
NSAIDS, CYCLOOXYGENASE INHIBITOR - TYPE	0.31
ANALGESICS NARCOTICS	0.26
FOLIC ACID PREPARATIONS	0.2
BETA-ADRENERGIC AGENTS	0.18
SODIUM/SALINE PREPARATIONS	0.16
Neurological System Disorders	**Weight**
Other nervous system symptoms and disorders	0.56
Rehabilitation care; fitting of prostheses; and adjustment of devices [254.]	0.5
Residual codes; unclassified; all E codes [259. and 260.]	0.46
Other connective tissue disease [211.]	0.33
Other and unspecified metabolic; nutritional; and endocrine disorders	0.18
Gastrointestinal Disorders	**Weight**
Residual codes; unclassified; all E codes [259. and 260.]	0.2
Other and unspecified metabolic; nutritional; and endocrine disorders	0.15
Other and unspecified gastrointestinal disorders	0.15
ANALGESIC/ANTIPYRETICS NON-SALICYLATE	0.32
POTASSIUM REPLACEMENT	0.26
BETA-ADRENERGIC AGENTS	0.23
ANALGESICS NARCOTICS	0.23
SODIUM/SALINE PREPARATIONS	0.22
SEDATIVE-HYPNOTICS NON-BARBITURATE	0.21
ANTIEMETIC/ANTIVERTIGO AGENTS	0.19
ANALGESICS NARCOTIC ANESTHETIC ADJUNCT AGENTS	0.18
NSAIDS, CYCLOOXYGENASE INHIBITOR - TYPE	0.16
IRRIGANTS	0.16
LAXATIVES AND CATHARTICS	0.15
GENERAL INHALATION AGENTS	0.15
Liver/Kidney System Disorders	**Weight**
Other aftercare [257.]	0.8
hronic kidney disease [158.]	0.39
Other and unspecified liver disorders	0.3
Immunity disorders [57.]	0.16

Temporal Phenotyping of Medically Complex Patients (MCPs) In order to illustrate the use of PARAFAC2 towards temporal phenotyping, we focus our analysis on a subset of pediatric patients from CHOA, which are classified by them as Medically Complex. These are the patients with high utilization, multiple specialty visits and high severity. Conceptually, those patients suffer from chronic and/or very severe conditions that are hard to treat. As a result, it becomes a very important challenge to accurately phenotype those patients, as well as provide a *temporal signature* for each one of them, which summarizes their phenotypes' evolution.

The number of MCPs in the CHOA cohort is 8,044, their diagnoses and medications sum up to 1,126, and the mean number of weekly observations for those patients is 28. We ran SPARTan for target rank $R = 5$ and the phenotypes discovered are provided in

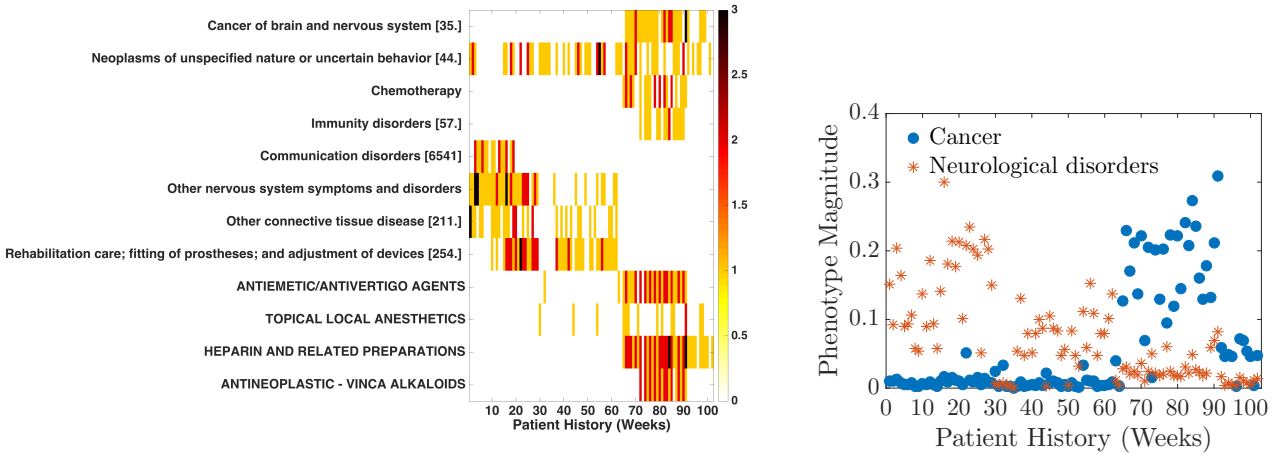


Figure 8: ***Left part:*** **Part of real EHR data of a Medically Complex Patient (MCP). For each week, it contains the occurrences of a diagnosis/medication in the patient's records.** ***Right part:*** **Temporal signature of the patient created by SPARTan. PARAFAC2 captures the stage where cancer treatment is initiated (week 65). At that point, indications of cancer treatment and diagnosis, such as cancer of brain, chemotherapy, heparin and antineoplastic drugs start to get recorded in the patient history. PARAFAC2 also captures the presence of neurological disorders during the first weeks of the patient history. The definition for each phenotype as produced by PARAFAC2 can be found in Table 4.**

Table 4 (*phenotypes' definition* matrix). The labels for each group are the definitions of the phenotypes provided by the medical expert, who endorsed their clinical meaningfulness.

In Figure 8, we provide part of the real EHR, as well as the temporal signature produced by SPARTan, for a certain medically complex patient. Regarding the EHR, we visualize the subset of diagnoses and medications for which the sum of occurrences for the whole patient history is above a certain threshold (e.g., 5 occurrences). This step ensures that the visualized EHR will only contain the conditions exhibiting some form of temporal evolution. For the patient example considered, we identify the top-2 relevant phenotypes through the importance membership indicator matrix S_k as discussed above. For those top-2 phenotypes, we present the resulting *temporal signature*, from which we easily detect intricate temporal trends of the phenotypes involved. Those trends were confirmed by the clinical expert as valuable towards fully understanding the phenotypic behavior of the MCPs.

6 DISCUSSION & CONCLUSIONS

PARAFAC2 has been the state-of-the-art model for mining "irregular" tensors, where the observations along one of its modes do not align naturally. However, it has been highly disregarded by practitioners, as compared to other tensor approaches. Bro [6] has summarized the reason for that as:

> *The PARAFAC2 model has not yet been used very extensively maybe because the implementations so far have been complicated and slow.*

The methodology proposed in this paper renders this statement no longer true for large and sparse data. In particular, as tested over real and synthetic datasets, SPARTan is both fast and memory-efficient, achieving up to 22× performance gains over the best previous implementation and also handling larger problem instances for which the baseline fails due to insufficient memory.

The key insight driving SPARTan's scalability is the pursuit and exploitation of special structure in the data involved in intermediate computations; prior art did not do so, instead treating those computations as a black-box.

The capability to run PARAFAC2 at larger scales is, in our view, an important enabling technology. As shown in our evaluations on EHR data, the clinically meaningful phenotypes and temporal trends identified by PARAFAC2 reflect the ease of the model's interpretation and its potential utility in other application domains.

Future directions include, but are not limited to: *a)* development of PARAFAC2 algorithms for alternative models of computation, such as distributed clusters [23], or supercomputing environments; *b)* extension of the methodology proposed for higher-order "irregular" tensors with more than one mismatched mode.

Finally, to enable reproducibility and promote further popularization of the PARAFAC2 modeling within the area of data mining, we open-source our implementations and make them *publicly available.*

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation, award IIS-#1418511 and CCF-#1533768, Children's Healthcare of Atlanta, Google Faculty Award and UCB. E. Papalexakis was supported by the Bourns College of Engineering at UC Riverside. The work of Fei Wang is partially supported by NSF IIS-#1650723. This work has been funded in part by the Laboratory-Directed Research & Development (LDRD) program at Sandia National Laboratories. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC. , a wholly owned subsidiary of Honeywell International, Inc. , for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

The authors would like to thank Professor Rasmus Bro and Dr. Tamara Kolda for valuable conversations.

REFERENCES

[1] 2017. Clinical Classifications Software (CCS) for ICD-9-CM. https://www.hcup-us.ahrq.gov/toolssoftware/ccs/ccs.jsp. (2017). Accessed: 2017-02-11.

[2] Evrim Acar and Bülent Yener. 2009. Unsupervised multiway data analysis: A literature survey. *IEEE transactions on knowledge and data engineering* 21, 1 (2009), 6–20.

[3] Claus Andersson and Rasmus Bro. 2000. The N-way toolbox for MATLAB. Available online. (January 2000). http://www.models.life.ku.dk/source/nwaytoolbox/

[4] Brett W Bader and Tamara G Kolda. 2007. Efficient MATLAB computations with sparse and factored tensors. *SIAM Journal on Scientific Computing* 30, 1 (2007), 205–231.

[5] Brett W. Bader, Tamara G. Kolda, and others. 2015. MATLAB Tensor Toolbox Version 2.6. Available online. (February 2015). http://www.sandia.gov/~tgkolda/TensorToolbox/

[6] Rasmus Bro. 1997. PARAFAC. Tutorial and applications. *Chemometrics and intelligent laboratory systems* 38, 2 (1997), 149–171.

[7] R Bro. 1998. Multi-way analysis in the food industry. (1998).

[8] Rasmus Bro, Claus A Andersson, and Henk AL Kiers. 1999. PARAFAC2-Part II. Modeling chromatographic data with retention time shifts. *Journal of Chemometrics* 13, 3-4 (1999), 295–309.

[9] Rasmus Bro and Sijmen De Jong. 1997. A fast non-negativity-constrained least squares algorithm. *Journal of chemometrics* 11, 5 (1997), 393–401.

[10] J Douglas Carroll and Jih-Jie Chang. 1970. Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika* 35, 3 (1970), 283–319.

[11] Dehua Cheng, Richard Peng, Ioakeim Perros, and Yan Liu. 2016. SPALS: Fast Alternating Least Squares via Implicit Leverage Scores Sampling. In *Advances In Neural Information Processing Systems*. 721–729.

[12] Peter A Chew, Brett W Bader, Tamara G Kolda, and Ahmed Abdelali. 2007. Cross-language information retrieval using PARAFAC2. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 143–152.

[13] Eric C Chi and Tamara G Kolda. 2012. On tensors, sparsity, and nonnegative factorizations. *SIAM J. Matrix Anal. Appl.* 33, 4 (2012), 1272–1299.

[14] Joon Hee Choi and S Vishwanathan. 2014. DFacTo: Distributed factorization of tensors. In *Advances in Neural Information Processing Systems*. 1296–1304.

[15] Gene H Golub and Charles F Van Loan. 2013. *Matrix Computations*. Vol. 3. JHU Press.

[16] Richard A Harshman. 1970. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis. (1970).

[17] R. A. Harshman. 1972b. PARAFAC2: Mathematical and technical notes. *UCLA Working Papers in Phonetics* 22 (1972b), 30–44.

[18] Richard A Harshman and Margaret E Lundy. 1996. Uniqueness proof for a family of models sharing features of Tucker's three-mode factor analysis and PARAFAC/CANDECOMP. *Psychometrika* 61, 1 (1996), 133–154.

[19] Nathaniel E Helwig. 2013. The special sign indeterminacy of the direct-fitting Parafac2 model: Some implications, cautions, and recommendations for Simultaneous Component Analysis. *Psychometrika* 78, 4 (2013), 725–739.

[20] Frank L Hitchcock. 1927. The expression of a tensor or a polyadic as a sum of products. *Studies in Applied Mathematics* 6, 1-4 (1927), 164–189.

[21] Joyce C Ho, Joydeep Ghosh, Steve R Steinhubl, Walter F Stewart, Joshua C Denny, Bradley A Malin, and Jimeng Sun. 2014. Limestone: High-throughput candidate phenotype generation via tensor factorization. *Journal of biomedical informatics* 52 (2014), 199–211.

[22] Joyce C Ho, Joydeep Ghosh, and Jimeng Sun. 2014. Marble: high-throughput phenotyping from electronic health records via sparse nonnegative tensor factorization. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 115–124.

[23] U Kang, Evangelos Papalexakis, Abhay Harpale, and Christos Faloutsos. 2012. Gigatensor: scaling tensor analysis up by 100 times-algorithms and discoveries. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 316–324.

[24] Henk AL Kiers, Jos MF Ten Berge, and Rasmus Bro. 1999. PARAFAC2-Part I. A direct fitting algorithm for the PARAFAC2 model. *Journal of Chemometrics* 13, 3-4 (1999), 275–294.

[25] Tamara G Kolda and Brett W Bader. 2009. Tensor decompositions and applications. *SIAM review* 51, 3 (2009), 455–500.

[26] Joseph B Kruskal. 1977. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear algebra and its applications* 18, 2 (1977), 95–138.

[27] Neal Lathia, Stephen Hailes, Licia Capra, and Xavier Amatriain. 2010. Temporal diversity in recommender systems. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 210–217.

[28] Evangelos E Papalexakis, Christos Faloutsos, and Nicholas D Sidiropoulos. 2015. ParCube: Sparse Parallelizable CANDECOMP-PARAFAC Tensor Decomposition. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 10, 1 (2015), 3.

[29] Evangelos E Papalexakis, Christos Faloutsos, and Nicholas D Sidiropoulos. 2016. Tensors for data mining and data fusion: Models, applications, and scalable algorithms. *ACM Transactions on Intelligent Systems and Technology (TIST)* 8, 2 (2016), 16.

[30] Rachel L Richesson, Jimeng Sun, Jyotishman Pathak, Abel N Kho, and Joshua C Denny. 2016. Clinical phenotyping in selected national networks: demonstrating the need for high-throughput, portable, and computational methods. *Artificial Intelligence in Medicine* 71 (2016), 57–61.

[31] Nicholas D Sidiropoulos and Rasmus Bro. 2000. On the uniqueness of multilinear decomposition of N-way arrays. *Journal of chemometrics* 14, 3 (2000), 229–239.

[32] Nicholas D Sidiropoulos, Lieven De Lathauwer, Xiao Fu, Kejun Huang, Evangelos E Papalexakis, and Christos Faloutsos. 2016. Tensor decomposition for signal processing and machine learning. *arXiv preprint arXiv:1607.01668* (2016).

[33] Vergil N Slee. 1978. The International classification of diseases: ninth revision (ICD-9). *Annals of internal medicine* 88, 3 (1978), 424–426.

[34] Shaden Smith, Niranjay Ravindran, Nicholas D Sidiropoulos, and George Karypis. 2015. SPLATT: Efficient and parallel sparse tensor-matrix multiplication. In *Parallel and Distributed Processing Symposium (IPDPS), 2015 IEEE International*. IEEE, 61–70.

[35] Alwin Stegeman and Tam TT Lam. 2015. Multi-set factor analysis by means of Parafac2. *Brit. J. Math. Statist. Psych.* (2015).

[36] Jimeng Sun, Charalampos E Tsourakakis, Evan Hoke, Christos Faloutsos, and Tina Eliassi-Rad. 2008. Two heads better than one: pattern discovery in time-evolving multi-aspect data. *Data Mining and Knowledge Discovery* 17, 1 (2008), 111–128.

[37] Jos MF ten Berge and Henk AL Kiers. 1996. Some uniqueness results for PARAFAC2. *Psychometrika* 61, 1 (1996), 123–132.

[38] Lloyd N Trefethen and David Bau III. 1997. Numerical linear algebra. (1997).

[39] Fei Wang, Noah Lee, Jianying Hu, Jimeng Sun, Shahram Ebadollahi, and Andrew F Laine. 2013. A framework for mining signatures from event sequences and its applications in healthcare data. *IEEE transactions on pattern analysis and machine intelligence* 35, 2 (2013), 272–285.

[40] Fei Wang, Jiayu Zhou, and Jianying Hu. 2014. DensityTransfer: A Data Driven Approach for Imputing Electronic Health Records. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*. IEEE, 2763–2768.

[41] Yichen Wang, Robert Chen, Joydeep Ghosh, Joshua C Denny, Abel Kho, You Chen, Bradley A Malin, and Jimeng Sun. 2015. Rubik: Knowledge guided tensor factorization and completion for health data analytics. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1265–1274.

[42] Jiayu Zhou, Fei Wang, Jianying Hu, and Jieping Ye. 2014. From micro to macro: data driven phenotyping by densification of longitudinal electronic medical records. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 135–144.