CSE 847 (Spring 2022): Machine Learning — Project Paper Study Summary

Wei-Chien Liao (liaowei2@msu.edu)

Abstract

This paper is a summary of basic concepts of tensors, Tucker decomposition and higher order singular value decomposition (HOSVD), and variants of randomized algorithms for computing these decompositions.

Keywords: higher order singular value decomposition (HOSVD), Tucker decomposition, randomized HOSVD

1. Notation and Preliminaries

The contents in this section are mainly based on [2].

Definition 1. The **order** of a tensor is the number of dimensions, also called **ways** or **modes**.

In this paper,

- **vectors** (tensors of order 1) are denoeted by boldface lowercase letters, e.g. $\mathbf{a}$.
- **matrices** (tensors of order 2) are denoeted by boldface capital letters, e.g. $\mathbf{A}$.
- **tensors** (order ≥ 3) are denoeted by boldface Euler script letters, e.g. $\boldsymbol{\mathcal{X}}$.
- the i-th entry of a vector $\mathbf{a}$ is denoeted by a_i.
- the (i, j)-th element of a matrix $\mathbf{A}$ is denoeted by A_{ij}.
- the (i, j, k)-th element of a third-order tensor $\boldsymbol{\mathcal{X}}$ is denoeted by x_{ijk}.
- a colon ":" is used to indicate all elements of a mode. e.g. for a matrix $\mathbf{A}$,
  - $\mathbf{a}_{i:} = i$-th row of $\mathbf{A}$.
  - $\mathbf{a}_{:j} = j$-th column of $\mathbf{A}$.

Definition 2. A **fiber** is defined by fixing every index but one.

For a third-order tensor $\boldsymbol{\mathcal{X}}$,

- $\mathbf{x}_{:jk} =$ **column fibers** or **mode-1 fibers** of $\boldsymbol{\mathcal{X}}$.
- $\mathbf{x}_{i:k} =$ **row fibers** or **mode-2 fibers** of $\boldsymbol{\mathcal{X}}$.

- $\mathbf{x}_{ij:}$ = **tube fibers** or **mode-3 fibers** of $\mathcal{X}$.

Definition 3. **Slices** are two-dimensional sections of a tensor defined by fixing all but two indices.

For a third-order tensor $\mathcal{X}$,

- $\mathbf{X}_{i::}$ = **horizontal slices** of $\mathcal{X}$.
- $\mathbf{X}_{:j:}$ = **lateral slices** of $\mathcal{X}$.
- $\mathbf{X}_{::k}$ = **frontal slices** of $\mathcal{X}$.

Definition 4 (Norm of a Tensor). The **norm** of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, denoeted by $||\mathcal{X}||$, is defined as

$$||\mathcal{X}|| = \sqrt{\sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} x^2_{i_1 i_2 \cdots i_N}}. \tag{1}$$

Definition 5 (Inner Product of Tensors). The **inner product** of two same-sized tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, denoeted by $\langle \mathcal{X}, \mathcal{Y} \rangle$, is defined as

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \sqrt{\sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} x_{i_1 i_2 \cdots i_N} y_{i_1 i_2 \cdots i_N}}. \tag{2}$$

Thus, by the definition of norm and inner product, $\langle \mathcal{X}, \mathcal{X} \rangle = ||\mathcal{X}||^2$.

Definition 6 (Rank-one Tensors). A N-way tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is **rank one** if it can be written as the outer product of N vectors,

$$\mathcal{X} = \mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \circ \cdots \circ \mathbf{a}^{(N)}, \tag{3}$$

for some vectors $\mathbf{a}^{(1)}, \mathbf{a}^{(2)}, \cdots, \mathbf{a}^{(N)}$ and "$\circ$" denotes the vector outer product.

Definition 7. A tensor is called **cubical** if every mode is the same size. A cubical tensor is called **supersymmetric** (some literatures call this "symmetric") if its elements remain constant under any permutation of the indices.

For a 3-way tensor $\mathcal{X} \in \mathbb{R}^{I \times I \times I}$, it is supersymmetric if

$$x_{ijk} = x_{ikj} = x_{jik} = x_{jki} = x_{kij} = x_{kji} \quad \forall i, j, k = 1, \cdots I.$$

Definition 8 (Diagnoal Tensor). A tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is **diagonal** if $x_{i_1 i_2 \cdots i_N} \neq 0$ only if $i_1 = i_2 = \cdots = i_N$.

Definition 9 (Matricization). The process of reordering the elements of an N-way array into a matrix is called **matricization** This is also called **unfolding** or **flattening**.

The mode-n matricization of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is denoeted by $\mathbf{X}_{(n)}$ and arranges the mode-n fibers to be the columns of the resulting matrix.

Remark 1. It is also possible to vectorize a tensor. This process is called vectorization.

1.1. Tensor Mulitiplication

The n**-mode prodcut** of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_n}$ is defined as

$$\text{eletmenwise:} \quad (\mathcal{X} \times_n \mathbf{U})_{i_1 \cdots i_{n-1} j i_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 i_2 \cdots i_N} u_{j i_n} \tag{4a}$$

$$\text{unfold tensors:} \quad \mathcal{Y} = \mathcal{X} \times_n \mathbf{U} \Leftrightarrow \mathbf{Y}_{(n)} = \mathbf{U}\mathbf{X}_{(n)} \tag{4b}$$

The result is a tensor of size $I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$. Each mode-$n$ fiber is multiplied by the matrix $\mathbf{U}$. The following are some properties of the n-mode prodcut:

(1) For distinct modes in a series of multiplications, the order of the multiplication is irrelevant.

$$\mathcal{X} \times_m \mathbf{A} \times_n \mathbf{B} = \mathcal{X} \times_n \mathbf{B} \times_m \mathbf{A}, \quad \text{for } n \neq m. \tag{5}$$

(2) If the modes are the same

$$\mathcal{X} \times_n \mathbf{A} \times_n \mathbf{B} = \mathcal{X} \times_n (\mathbf{B}\mathbf{A}). \tag{6}$$

1.2. Some Matrix Products

- **Kronecker Product**

  The **Kronecker product** of two matrices $\mathbf{A} \in \mathbb{R}^{I \times J}$ and $\mathbf{B} \in \mathbb{R}^{K \times L}$, denoeted by $\mathbf{A} \otimes \mathbf{B}$, is defined as

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \cdots & a_{1J}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \cdots & a_{2J}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}\mathbf{B} & a_{I2}\mathbf{B} & \cdots & a_{IJ}\mathbf{B} \end{bmatrix} \tag{7}$$

- **Khatri-Rao Product**

  The **Khatri-Rao product** of two matrices $\mathbf{A} \in \mathbb{R}^{I \times K}$ and $\mathbf{B} \in \mathbb{R}^{J \times K}$, denoeted by $\mathbf{A} \odot \mathbf{B}$, is defined as

$$\mathbf{A} \odot \mathbf{B} = \begin{bmatrix} \mathbf{a}_1 \otimes \mathbf{b}_1 & \mathbf{a}_2 \otimes \mathbf{b}_2 & \cdots & \mathbf{a}_K \otimes \mathbf{b}_K \end{bmatrix} \tag{8}$$

- **Hadamard Product**

  The **Hadamard product** of two matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{I \times J}$, denoeted by $\mathbf{A} * \mathbf{B}$, is defined as

$$\mathbf{A} * \mathbf{B} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} & \cdots & a_{1J}b_{1J} \\ a_{21}b_{21} & a_{22}b_{22} & \cdots & a_{2J}b_{2J} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}b_{I1} & a_{I2}b_{I2} & \cdots & a_{IJ}b_{IJ} \end{bmatrix} \tag{9}$$

Some properties of these matrix prodcuts:

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD} \tag{10a}$$

$$(\mathbf{A} \otimes \mathbf{B})^\dagger = \mathbf{A}^\dagger \otimes \mathbf{B}^\dagger \tag{10b}$$

$$\mathbf{A} \odot \mathbf{B} \odot \mathbf{C} = (\mathbf{A} \odot \mathbf{B}) \odot \mathbf{C} = \mathbf{A} \odot (\mathbf{B} \otimes \mathbf{C}) \tag{10c}$$

$$(\mathbf{A} \odot \mathbf{B})^T (\mathbf{A} \odot \mathbf{B}) = (\mathbf{A}^T\mathbf{A}) * (\mathbf{B}^T\mathbf{B}) \tag{10d}$$

$$(\mathbf{A} \odot \mathbf{B})^\dagger = (\mathbf{A}^T\mathbf{A}) * (\mathbf{B}^T\mathbf{B})^\dagger (\mathbf{A} \odot \mathbf{B})^T \tag{10e}$$

2. Higher Order Singular Value Decomposition (HOSVD)

This section aims to explain the basic concepts of HOSVD. The HOSVD can be viewed as a form of higher-order principal component analysis (PCA). Different names of HOSVD appear in literatures including Tucker Decomposition, N-mode PCA, or N-mode SVD. The core concept of HOSVD is to decomposes a tensor into a core tensor multiplied (or transformed) by a matrix along each mode. To express this idea in mathematical equation, we first consider the three-way case, then extend it to general N-way tensors.

Let $\mathfrak{X} \in \mathbb{R}^{I\times J\times K}$ be a three-way tensor. The HOSVD of $\mathfrak{X}$ is defined as

$$\mathfrak{X} \approx \mathfrak{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} = \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} g_{pqr}\mathbf{a}_p \circ \mathbf{b}_q \circ \mathbf{c}_r = [\![\mathfrak{G}\ ;\ \mathbf{A},\ \mathbf{B},\ \mathbf{C}]\!]. \tag{11}$$

where $\mathbf{A} \in \mathbb{R}^{I\times P}$, $\mathbf{B} \in \mathbb{R}^{J\times Q}$, and $\mathbf{C} \in \mathbb{R}^{K\times R}$ are called the **factor matrices**, and $\mathfrak{G} \in \mathbb{R}^{P\times Q\times R}$ is called the **core tensor**. The factor matrices can be thought of as the principal components in each mode similar to the case in SVD, and entries of the core tensor show he level of interaction between the different components. Thus, this is similar to the singular value matrix in SVD in some sense. This can also be written in eletmenwise form as follows

$$x_{ijk} \approx \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} g_{pqr}a_{ip}b_{jq}c_{kr} \quad \text{for } i = 1,\cdots,I,\ j = 1,\cdots,J,\ k = 1,\cdots,K. \tag{12}$$

Note that the factor matrices are not assumed to be orthogonal or columnwise orthonormal, but it is possible make factor matrices to have these desired properties. HOSVD can also be written in matricized form,

$$\begin{aligned} \mathbf{X}_{(1)} &\approx \mathbf{A}\mathbf{G}_{(1)}\left(\mathbf{C}\otimes\mathbf{B}\right)^T \\ \mathbf{X}_{(2)} &\approx \mathbf{B}\mathbf{G}_{(2)}\left(\mathbf{C}\otimes\mathbf{A}\right)^T \\ \mathbf{X}_{(3)} &\approx \mathbf{C}\mathbf{G}_{(3)}\left(\mathbf{B}\otimes\mathbf{A}\right)^T \end{aligned} \tag{13}$$

For a general N-way tensor $\mathfrak{X} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, the HOSVD can be generalized as

$$\mathfrak{X} \approx \mathfrak{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \cdots \times_N \mathbf{A}^{(N)} = [\![\mathfrak{G}\ ;\ \mathbf{A}^{(1)},\ \mathbf{A}^{(2)},\cdots,\mathbf{A}^{(N)}]\!]. \tag{14}$$

Expressing this elementwise gives

$$x_{i_1i_2\cdots i_N} \approx \sum_{r_1=1}^{R_1}\sum_{r_2=1}^{R_2}\cdots\sum_{r_N=1}^{R^N} g_{r_1r_2\cdots r_N}a^{(1)}_{i_1r_1}a^{(2)}_{i_2r_2}\cdots a^{(N)}_{i_Nr_N} \quad \text{for } i_n = 1,\cdots,I_n,\ n = 1,\cdots,N. \tag{15}$$

The matricized version is given by

$$\mathbf{X}_{(n)} \approx \mathbf{A}^{(n)}\mathbf{G}_{(n)}\left(\mathbf{A}^{(N)}\otimes\cdots\otimes\mathbf{A}^{(n+1)}\otimes\mathbf{A}^{(n-1)}\otimes\cdots\otimes\mathbf{A}^{(1)}\right)^T, \tag{16}$$

for $n = 1,2,\cdots,N$.

The following definition of n-rank, sometimes called numerical rank or multilinear rank in some literatures, is useful for study algorithms corresponding to HOSVD.

Definition 10 (n-rank). $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ be a N-way tensor. Then the **n-rank** of $\mathcal{X}$, denoted by $\text{rank}_n(\mathcal{X})$, is the column rank of $\mathbf{X}_{(n)}$.

Thus, the n-rank of a tensor is the dimension of the vector space spanned by the mode-n fibers. With this terminology, if let $R_n = \text{rank}_n(\mathcal{X})$, then the tensor $\mathcal{X}$ can be called a rank-$(R_1, R_2, \cdots, R_N)$ tensor.

Algorithms for Computing HOSVD

Common algorithms for computing the HOSVD of a given tensor $\mathcal{X}$ include "classical" HOSVD, Sequentially Truncated HOSVD (STHOSVD), and Higher-Order Orthogonal Iteration (HOOI).

The "classical" HOSVD simply perform SVD to compute the left singular vectors of $\mathbf{X}_{(n)}$ for each n, then form the core tensor. Although this is straightforward, it is inefficient for large tensor data since performing SVD is computationally expensive.

The Sequentially Truncated HOSVD (STHOSVD) reduce the comoputation complexity by sequentially shrink the size of the underlying unfolding matrices used as input for SVD in each iteration. The SVD decompositions for $\mathbf{X}_{(n)}$ in "classical" HOSVD is now replaced by truncated SVD. This not only greatly enhance the speed without sacrficing the accuracy, and sometimes even gives better accuracy. However, this method does not guarantee to find the best multilinear rank approximation for a given tensor.

An alternative way to deal with the high computational complexity of "classical" HOSVD is to observe that the problem of finding the left singular vectors of can be reformulated as solving equivalent least squares problems. This type of methods has the advantage that the result output would provide numerically the best multilinear rank approximation for a tensor. The Higher-Order Orthogonal Iteration (HOOI) algorithm is developed based on this observation. If the HOSVD of the tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is given by

$$\mathcal{X} \approx \mathcal{G} \times_1 \mathbf{Q}^{(1)} \times_2 \mathbf{Q}^{(2)} \cdots \times_N \mathbf{Q}^{(N)}$$

Write this in matricized form to obtain for each $n = 1, 2, \cdots, N$,

$$\mathbf{X}_{(n)} \approx \mathbf{Q}^{(n)} \mathbf{G}_{(n)} \left(\mathbf{Q}^{(N)} \otimes \cdots \otimes \mathbf{Q}^{(n+1)} \otimes \mathbf{Q}^{(n-1)} \otimes \cdots \otimes \mathbf{Q}^{(1)} \right)^T .$$

Assume the factor matrix $\mathbf{Q}^{(n)}$ is unknown and other factor matrices are known, then $\mathbf{Q}^{(n)}$ can be computed by solving the following least square problem,

$$\mathbf{Q}^{(n)} = \underset{\mathbf{Q} \in \mathbb{R}^{I_n \times R_n}}{\arg\min} \; ||\mathbf{A}^{(n)} \mathbf{Q}^T - \mathbf{X}_{(n)}^T||_F^2, \tag{17}$$

where

$$\mathbf{A}^{(n)} = \left(\mathbf{Q}^{(N)} \otimes \cdots \otimes \mathbf{Q}^{(n+1)} \otimes \mathbf{Q}^{(n-1)} \otimes \cdots \otimes \mathbf{Q}^{(1)} \right) \mathbf{G}_{(n)}^T .$$

This is equivalent to finding the left singular vectors of the matrix $\mathbf{Z}_{(n)}$, where the corresponding tensor $\mathcal{Z}$ is defined by

$$\mathcal{Z} := \mathcal{X} \times_{k \neq n} \left\{ \mathbf{Q}^{(n)} \right\}^T .$$

Repeat this process for each $n = 1, \cdots, N$. Then compute the core tensor $\mathcal{G}$ by solving the following equivalent least square problem

$$\mathcal{G} = \underset{\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_n}}{\arg\min} ||\mathbf{B}\mathbf{g} - \mathbf{x}||_2^2, \tag{18}$$

where $\mathbf{g}$ and $\mathbf{x}$ are vectorization of the tensors $\mathcal{G}$ and $\mathcal{X}$ respectively.

Remark 2. Unlike the SVD of a matrix, the HOSVD of a tensor is not guaranteed to be unique. A more detailed discussion can be found in section 4.3 of [2].

3. Randomized Algorithms for Computing HOSVD

This section summarizes some randomized algorithm for computing HOSVD discussed in [1]. The HOSVD algorithms described in the previous section has a common challenge that it is expensive to compute low-rank approximation of the unfolding matrices. In view of this issue, many algorithms are developed based on introducing randomized low-rank matrix algorithms to replace the truncated SVD or economic SVD used in HOSVD algorithms. These randomized algorithms can be mainly classifed as the following four types: random projecttion, randomized sampling, randomized count-sketch, and randomized least-squares. There are also new algorithms, such as the Sketched alternating least-squares (ALS) proposed in [3], which combines count-sketch and random least-squares techniques.

3.1. Random Projection

The basic idea of random projection class algorithms is to approximate the column space of the unfolding matrices of a given data tensor as follows: let $\mathbf{X} \in \mathbb{R}^{m \times n}$ denote the original data matrix.

1. Generate a random matrix $\mathbf{\Omega} \in \mathbb{R}^{n \times p}$ with $p < n$ from some probability distribution such as Gaussian, Bernoulli or uniform distribution.
2. Compute $\mathbf{Y} = \mathbf{X}\mathbf{\Omega} \in \mathbb{R}^{m \times p}$.

Then this new matrix $\mathbf{Y}$ has p column vectors randomly generated from the column space of $\mathbf{X}$ and has fewer columns than the original $\mathbf{X}$.

Based on this concept, the "classical" HOSVD, STHOSVD, and HOOI can be modified by adding this techniques and gives the Random Projection HOSVD (RP-HOSVD) algorithm, the Randomized STHOSVD (R-STHOSVD) algorithm and the Random Projection HOOI (RP-HOOI) algorithm respectively. An issue of the above three randomized algorithms is that they all require passing the input data tensor multiple times. This could produce high communication costs if the data tensors are stored on multiple cores. Hence, a class of algorithms refeered to as Randomized Pass-Efficient algorithms (such as R-PET) are proposed to remedy this issue.

3.2. Randomized Sampling

The main idea of this class of methods is to compute the factor matrices by sampling the columns or rows of the corresponding unfolding matrix of the input data tensor. There are many different sampling techniques that can be applied to randomly select the columns or rows. The sampling can be based on probability distributions like Gaussian distribution or uniform distribution. It is also possbile to consider sampling with or without replacement. There are other techniques like sampling based on leverage score and cross-approximation that can also be applied to the sampling step of this class of algorithms. Some examples of algorithms belonging to this class are Randomized Sampling Tucker Approximation (R-ST) and Randomized Higher Order Interpolatory Decomposition (R-HOID).

3.3. Randomized Least-squares

The motivation of developing this type of algorithms is that RP-HOOI requires several multiplications of random matrices and operations of the core tensors with factor matrices. This becomes expensive if it needs many iterations to converge. This class of methods aims to reduce this cost by replacing these costly computations by some equivalent least squares problems. As shown in the previous section, computing the HOSVD of a tensor can be transformed to an equivalent optimization problem which can then be reformulated as solving some least squares problems.

3.4. Randomized Count-sketch

Similar to the idea of random projection, count-sketch also aims to generate random vectors in the column or row space spanned by the original data matrix. In random project, it requires to perform matrix multiplication of original data matrix and a random matrix. This could be expensive when coping with large data. Count-sketch attempts to cure this problem by generating the desired vectors without comoputing matrix multiplications. The procedure of count-sketch can be summarized into the following three steps:

1. Hashing
2. Distributing vectors with same hash numbers to the same group
3. Signing the columns and summing each group as a representative column

This kind of techniques can be applied to tensors with some proper extensions. This technique is applied in RP-HOOI to solve the underlying least squares problems.

References

[1] Salman Ahmadi-Asl, Stanislav Abukhovich, Maame G. Asante-Mensah, Andrzej Cichocki, Anh Huy Phan, Tohishisa Tanaka, and Ivan Oseledets. Randomized algorithms for computation of tucker decomposition and higher order SVD (HOSVD). IEEE Access, 9:28684–28706, 2021.

[2] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. SIAM Review, 51(3):455–500, 2009.

[3] Linjian Ma and Edgar Solomonik. Fast and accurate randomized algorithms for low-rank tensor decompositions. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, Advances in Neural Information Processing Systems, 2021.