CSE 392: Matrix and Tensor Algorithms for Data	Spring 2024

Lecture 16 — 03/18/2024

Instructor: Shashanka Ubaru — *Scribe: Stefan Rutledge*

1 Canonical Polyadic (CP) Decomposition

1.1 Tensor Decomposition

Tensor Decomposition is analogous to Matrix Decomposition in several ways. Like matrices, tensors are used to represent datasets and are generally very large in size. To combat this, tensor decomposition can be used to compress the size for performance and storage benefits. Furthermore, tensor decomposition can be used to de-noise the tensor and reduce it down to a more efficient representation. Furthermore, tensor decomposition uses and reveals "hidden" correlations between values and patterns that are typically difficult to see in multi-dimensional data. As with matrix decomposition, there are different types of tensor decompositions that all have their associated pros and cons and thus the ideal decomposition to use is always application dependent. Lastly, recall in matrix factorization there are two paradigms of decomposing the matrix: one is to represent it as a sum of rank-1 matrices and the other is the reduce it to representative subspaces. Similarly, tensor factorization has two paradigms of decompositions. As shown in Figure 1, a tensor can be represented as a sum of rank-1 tensors or as a set of representative subspaces.

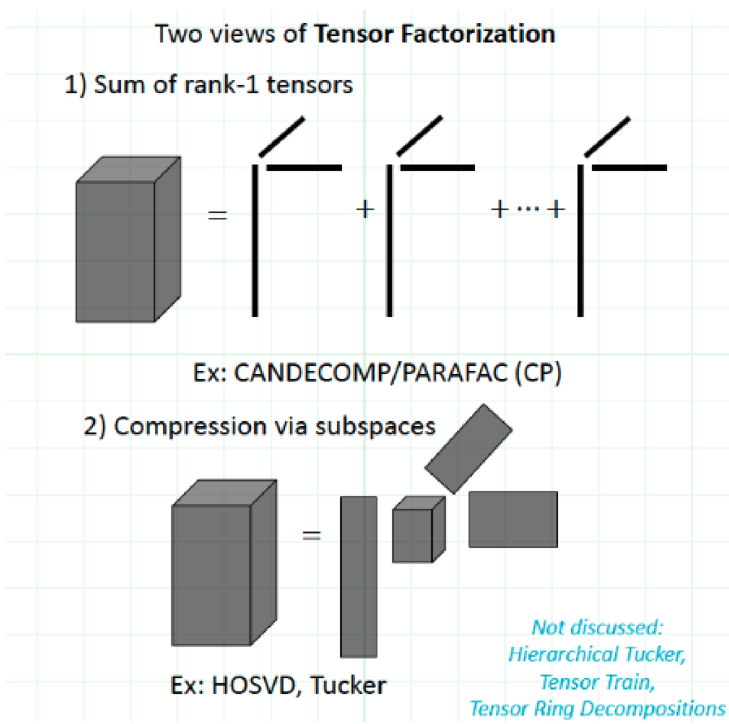
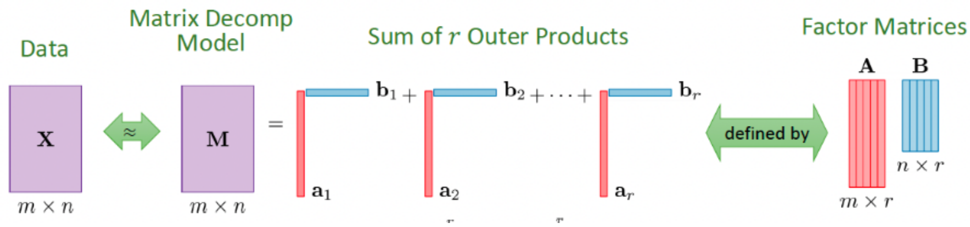


Figure 1: Tensor Factorization

1.2 Matrix Factorization

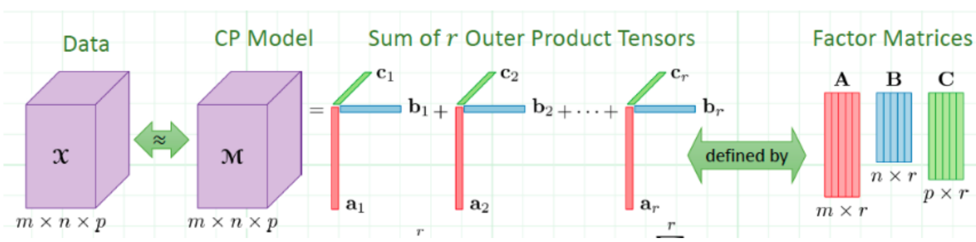
This section will give more background on decomposing a matrix into a sum of rank-1 matrices to build the basis for tensor factorization. To reduce a matrix X (m x n) into a rank-r approximation, M (m x n), then factor matrices, A and B, are found, which are of sizes A = m x r and B = n x r. Then M is composed of the sum of r outer products of each column of A and B, such that M = $AB^T = \sum_{l=1}^{r} a_l b_l^T = \sum_{l=1}^{r} a_l \circ b_l$.



Recall the SVD of a matrix is the traditional dimension reducer/feature extractor because it is optimal for both performance and space. The SVD minimizes the Frobenius norm between matrix A and rank-k approximation B which allows it to denoise and reduce the matrix to its principle components and compress the matrix into k(n+m) values, as opposed to mn.

1.3 Tensor Factorization

In attempts to find an analogous of SVD for high-dimensional tensors, mathematicians proposed decomposing a tensor X (a x b x c) into a sum of rank 1 tensors, which is called the Canonical Polyadic (CP) Decomposition. Similar to matrices, to reduce a tensor X (m x n x p) into a rank-r approximation M (m x n x p), then factor matrices A, B, and C, are found, which are of sizes A = m x r, B = n x r, C = p x r. Then M is composed of the sum of r outer products of each column of A, B, and C such that M = $\sum_{l=1}^{r} a_l \circ b_l \circ c_l$. This idea can be extended to a d-way tensor by summing the outer products of the columns for all d factor matrices.



The Kruskal Notation is shown as M = $[[A, B, C]]$ for a three way tensor decomposition. Also for a unit normalized decomposition with diagonal vector λ: M = $[[\lambda; A, B, C]]$

1.4 Matrix Kronecker Product

The Kronecker product, denoted by $\otimes$, is a common operation on matrices or tensors. Given two matrices A and B, the Kronecker product results in a larger block matrix, where each element of A is multiplied by the entire matrix B.

If A is an $m \times n$ matrix and B is a $p \times q$ matrix, then the Kronecker product of A and B is denoted as $A \otimes B$, and it results in an $mp \times nq$ block matrix.

The Kronecker product is defined as follows:

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{bmatrix}$$

Where a_{ij} represents the element in the i-th row and j-th column of matrix A, and $a_{ij}B$ denotes the matrix B scaled by a_{ij}.

The Kronecker Product upholds a number of properties:

1. $(C \otimes B) \otimes A = C \otimes (B \otimes A)$
2. $(B \otimes A)^T = B^T \otimes A^T$
3. $(B \otimes A)(C \otimes D) = (BD) \otimes (AC)$
4. $\mathbf{vec}(AXB^T) = (B \otimes A)\mathbf{vec}(X)$
5. $(B \otimes A)^{-1} = B^{-1} \otimes A^{-1}$

With the Kronecker product, for vectors $a \in \mathbb{R}^m, b \in \mathbb{R}^n, c \in \mathbb{R}^p$, $c \otimes b \otimes a$ produces a vector of length mnp. Thus, the vectorized form of the rank-1 one tensor X is denoted as $\mathbf{vec}(X) = c \otimes b \otimes a$

1.5 KhatriRao Product

The Khatri-Rao product, also known as the column-wise Kronecker product, is an operation on two matrices. Given two matrices A and B, where A is of size $m \times r$ and B is of size $n \times r$, the Khatri-Rao product, denoted $A \odot B$, results in a matrix of size $mn \times r$ obtained by taking the Kronecker product of the corresponding columns of the matrices.

Mathematically, the Khatri-Rao product is defined as follows:

Let $A = [a_1, a_2, \dots, a_n]$ and $B = [b_1, b_2, \dots, b_n]$, where a_i and b_i represent the i-th column vectors of A and B respectively. Then the Khatri-Rao product is given by:

$$A \odot B = [a_1 \otimes b_1, a_2 \otimes b_2, \dots, a_n \otimes b_n]$$

In this product, each column of the resulting matrix is formed by taking the Kronecker product of the corresponding columns of A and B, $(B \odot A)_{*j} = B_{*j} \otimes A_{*j}$.

The KRP upholds a number of properties:

1. $C \odot (B \odot A) = (C \odot B) \odot A$

2. $(B \odot A)^T (B \odot A) = B^T B * A^T A$

3. $(B \otimes A)(D \odot C) = (BD) \odot (AC)$

1.6 Matricized Tensor Times KRP (MTTKRP)

With the KRP, a three way tensor, X, can be unfolded along the first dimension, aka matricized, by the following equation:

$$X_{(1)} = \sum_{l=1}^{r} a_l (c_l \otimes b_l)^T = [a_1, ..., a_r][c_1 \otimes b_1, ..., c_r \otimes b_r]^T = A(C \odot B)^T$$

Hence the three factorization matrices can be defined as:

$$A = X_{(1)}(C \odot B), B = X_{(2)}(C \odot A), C = X_{(3)}(B \odot A)$$

And for a d-way tensor the mode-k matricized tensor times KRP is given by:

$$V = X_{(k)}(A_d \odot ... \odot A_{k+1} \odot A_{k-1} \odot ... \odot A_1)$$

1.7 CP Tensor Decomposition

With the above definitions let us return to the CP tensor decomposition as a rank-r approximation of tnesor X. To find the CP decomposition is equivalent to $\min_{a_l, b_l, c_l} \|X - \sum_{l=1}^{r} \sigma_l \cdot a_l \circ b_l \circ c_l\|_F$. If the Frobenius norm is equal to 0 and r is minimized then r is the rank of tensor X. The solution to CP decomposition does not uphold the orthogonal property, unlike SVD, and is not a matrix product based factorization. However, the CP decomposition solution does uphold a uniqueness property if $\mathbf{rank}_k(A) + \mathbf{rank}_k(B) + \mathbf{rank}_k(C) \geq 2r + 2$. Note that $\mathbf{rank}_k(A)$ is the maximum value of k such that any k columns of A are linearly indepedent, NOT the rank of A.

1.8 Alternating Least Squares (CP-ALS)

Finding the rank of tensor X is an NP hard problem and thus an approximation algorithm will be necessary. One potential algorithm is the Alternating Least Squares algorithm which optimizes across all three variables, A, B, and C, one at a time. The general idea is to fix B and C and solve for A, then fix A and C and solve for B, and fix A and B and solve for C, and repeat until convergence. The equation to solve for A is shown below:

$$\min_A \|X_{(1)} - A(C \odot B)^T\|_F^2 =$$

$$\min_A \|(C \odot B)A^T - X_{(1)}^T\|_F^2$$

From normal equations:

$$(C \odot B)^T (C \odot B) A^T = (C \odot B)^T X_{(1)}^T$$

$$(C^T C * B^T B) A^T = (C \odot B)^T X_{(1)}^T$$

$$A^T = (C^T C * B^T B)^{-1}(C \odot B)^T X_{(1)}^T$$

$$A = X_{(1)}(C \odot B)(C^T C * B^T B)^{-1}$$

This can be written for a general d-way tensor as the following:

$$\min_{A_k} \|X_{(k)} - A_{A_k}(A_d \odot ... \odot A_{k+1} \odot A_{k-1} \odot ... \odot A_1)^T\|_F^2 =$$

$$\min_{A_k} \|Z_k A^T - X_{(k)}^T\|_F^2$$

where $Z_k = A_d \odot ... \odot A_{k+1} \odot A_{k-1} \odot ... \odot A_1$

$$Z_k^T Z_k A_k^T = Z_k^T X_{(k)}^T$$

$$(A_d^T A_d * ... * A_{k+1}^T A_{k+1} * A_{k-1}^T A_{k-1} * ... * A_1^T A_1)A_k^T = Z_k^T X_{(k)}^T$$

$$A_k = X_{(k)} Z_k V_k^{-1}$$

where $V_k = A_d^T A_d * ... * A_{k+1}^T A_{k+1} * A_{k-1}^T A_{k-1} * ... * A_1^T A_1$

Finally for a d-way tensor the generalized alternating least squares algorithm is given as follows for a desired rank r, also followed by a MATLAB example:

1. Initialize $A_k \in \mathbb{R}^{n_k \times r}$ for all $k \in [d]$
2. repeat
3. for k = 1,...,d do
4. $Z_k \leftarrow A_d \odot ... \odot A_{k+1} \odot A_{k-1} \odot ... \odot A_1$
5. $A_k \leftarrow \arg\min_B \|Z_k B^T - X_{(k)}^T\|_F^2$
6. end
7. until $\|X - [[A_1, A_2, ..., A_d]]\|_F^2$ under threshold

```
% Tensor decomposition

% Example for CP-ALS using tensor-toolbox

% Tensor Toolbox : https://www.tensortoolbox.org/

% The Excitation Emission Matrix (EEM) tensor data
% has been curated from a series of
% Fluorescence Spectroscopy experiments

% Dataset from https://gitlab.com/tensors/tensor_data_eem

addpath(genpath(pwd))
%% Load data

load eem18

% 18 samples x 251 emissions x 21 excitations.

%% Visualization

viz_slices(X,mixtures,1,'X-slices')

%% CP decomposition

rng('default')
M = cp_als(X,5);
viz_eem_cp(M,mixtures,[],'eem_model');

%% Error

err = norm(X - tensor(M))
```

Figure 2: MATLAB example of CP-ALS