Lecture2

@sayankotor

based on Skolkovo CDISE Matrix and Tensor Factorization Course, Caltech Tensor&Neural Network Course

23 sent 2020

Tucker Decomposition

Tucker decomposition

- $Y = G \times_1 A \times_2 B \times_3 C = [|G; A, B, C|]$

Represent initial unfolding tensor Y by factors

Note: on the previous lecture we obtained the same formula for CP decomposition using LLS, i.e. minimization the distance between initial and decomposed tensor.

- $Y(1) = AG_{(1)}(C \otimes B)^T \rightarrow A = linalg.solve(...)$
- $Y(2) = BG_{(2)}(C \otimes A)^T$
- $Y(3) = CG_{(3)}(B \otimes A)^T$
- $vec(Y) = (C \odot B \odot A)vec(G)$

Tucker Decomposition

Constrained to factors

- $A^T A = I_{R1}$
- $B^T B = I_{R2}$
- $C^T C = I_{R3}$

Orthogonality properties of core tensor

- All orthogonality. The slices in each mode are mutually orthogonal, e.g., for a 3rd-order tensor and its lateral slices

$$< S_{:k:} S_{:j:} >= 0, forj \neq j \tag{1}$$

- Pseudo-diagonality. The Frobenius norms of slices in each mode are decreasing with the increase in the running index.

$$||S_{:k:}||_F >= ||S_{:j:}||_F = 0, forj >= j \tag{2}$$

These norms play a role similar to singular values in standard matrix SVD.

HOSVD, Sequential Projection

Given tensor Y, we want to obtain its HOSVD in the Tucker format $\hat{Y} = [|S; U_1, U_2, U_3|]$

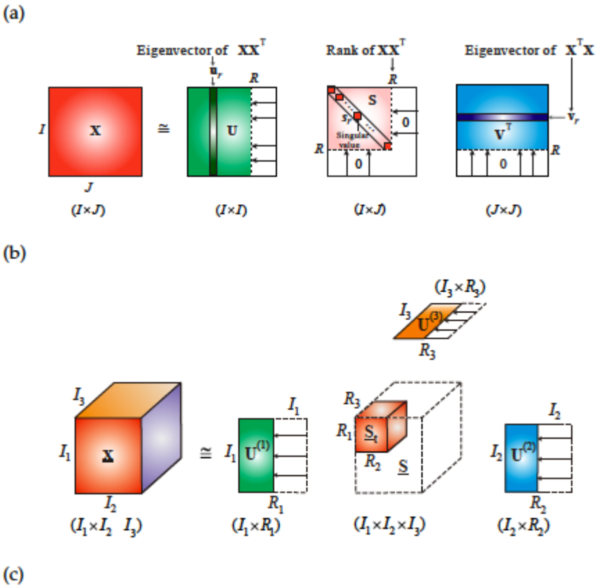
Algorithm

Begin:

- $Y_{(1)} \approx U_1 S_1 (V_1)^T$
- $T_1 = Y \times_1 (U_1)^T$
- $T_{1(2)} \approx U_2 S_2 (V_2)^T$
- $T_2 = T_1 \times_2 (U_2)^T$
- $T_{2(3)} \approx U_3 S_3 (V_3)^T$
- $S = T_2 \times_3 U_3^T$

Return S, U_1, U_2, U_3 End

HOSVD vs. SVD



(a)
Eigenvector of XX^T
u_r
R
Rank of XX^T
Eigenvector of X^T X
I
X
J
≅
U
S
s_r
Singular value
0
V^T
v_r
(I×J)
(I×I)
(I×J)
(J×J)
(b)
(I_3×R_3)
I_3
U^(3)
R_3
I_1
I_2
(I_1×I_2 I_3)
U^(1)
R_1
(I_1×R_1)
S_t
R_2
S
(I_1×I_2×I_3)
U^(2)
(I_2×R_2)
(c)

HOOI

We have tensor $\hat{Y}$ with dimentions $I_1 \times I_2 \times I_3$ and want to find it's Tucker decomposition with ranks [R1, R2, R3] $[|G; A, B, C|]$

Algorithm

Begin:

- Initialize B, C with HOSVD
- repeat:
  - $Z = \hat{Y} \times_2 B^T \times_3 C^T$
  - Update A: A is $R_1 leftleadingsingularvectorsof Z_(0)$
  - $Z = \hat{Y} \times_1 A^T \times_3 C^T$
  - Update B: B is $R_2 leftleadingsingularvectorsof Z_(1)$
  - $Z = \hat{Y} \times_1 B^T \times_2 A^T$
  - Update C: C is $R_3 leftleadingsingularvectorsof Z_(2)$
- until stopping criteria is met number of iterations or accuracy
- $G = Z \times_3 C^T$

Return G, A, B, C End

HT and TT

Hierarchical Tucker

- when you represent decomposition as a tree
- each node correspond to a factor U_k
- factor can be decomposed too by it's own factors

Tensor Train (Matrix Product State)

The special case of Hierarchical Tucker