Prepoznavanje lica koristeći tenzor-tenzor dekompoziciju

Tin Kranželić, Dora Parmać

5. svibnja 2021.

Sadržaj

1. Uvod
2. Tradicionalne dekompozicije
   - CP dekompozicija
   - Tuckerova dekompozicija
3. Tradicionalni algoritmi
   - PCA algoritam
   - TensorFaces
4. Novi pristup - Tensor Framework
5. T-SVD dekompozicija
6. Novi pristup baziran na TSVD
7. Alternativa - tenzor OR pivotiranje

Uvod

- Želimo spremiti podatke poput slika različitih osoba, pod različitim osvjetljenjima, različitih ekspresija i sl u tenzore.
- Nakon što iskoristimo tenzore višeg reda za pohranu multidimenzionalnih podataka, važan korak je kompresija podataka.
- Najčešće korištene tenzor dekompozicije su CANDECOMP/PARAFAC (CP) dekompozicija i Tucker dekompozicija.

Tradicionalne dekompozicije

Neka je $\mathcal{A}$ tenzor reda $I \times J \times K$. Tada se njegova CP dekompozicija može zapisati kao suma vanjskog produkta vektora:

$$\mathcal{A} = \sum_{l=1}^{R} u^{(l)} \circ v^{(l)} \circ w^{(l)} \quad \Longrightarrow \quad (\mathcal{A})_{i,j,k} = \sum_{l=1}^{R} u_i^{(l)} v_j^{(l)} w_k^{(l)} \qquad (1)$$

gdje su

$$u^{(l)} \in \mathbb{R}^I, v^{(l)} \in \mathbb{R}^J, w^{(l)} \in \mathbb{R}^K, zak = 1, 2, \ldots R.$$

Tuckerova dekompozicija

$$\mathcal{A} = \mathcal{G} \times_1 \boldsymbol{U} \times_2 \boldsymbol{V} \times_3 \boldsymbol{W} = \sum_{i=1}^{R_1} \sum_{j=1}^{R_2} \sum_{k=1}^{R_3} = \sigma_{ijk}(u^{(i)} \circ v^{(j)} \circ w^{(k)}) \quad (2)$$

gdje je: $R_1 \leq I, R_2 \leq J, R_3 \leq K$ i za $i, j, k, i^{(i)} \in \mathbb{R}^I, v^{(l)} \in \mathbb{R}^J, w^{(l)} \in \mathbb{R}^K, \mathcal{G} = \sigma_{ijk} \in \mathbb{R}^{R_1 \times R_2 \times R_3}, \boldsymbol{U} = [u^{(1)}, u^{(2)} \dots u^{(R_1)}], \boldsymbol{V} = [v^{(1)}, v^{(2)} \dots v^{(R_2)}],$ $\boldsymbol{W} = [w^{(1)}, w^{(2)} \dots w^{(R_3)}]$.

Tradicionalni algoritmi - PCA algoritam

- Klasična tehnika koja se koristi u prepoznavanju slika i kompresiji
- Koristi kako bi se transformirao veliki broj moguće koreliranih varijabli u manji broj nekoreliranih varijabli poznatije kao glavne komponente
- Prvih nekoliko glavnih komponenata objašnjava većinu varijacija originalnih podataka, dok preostali predstavljaju zanemariv doprinos. Stoga podatka možemo ekonomičnije opisati pomoću prvih nekoliko glavnih komponenata.

PCA algoritam

Neka je $\boldsymbol{X}_1 \dots X_m$ kolekcija $l \times n$ slika. Definiramo matricu $\boldsymbol{A}$ tako da je j-ti stupac vec$(\boldsymbol{X}_j)$. Srednja vrijednost se tada računa kao $\boldsymbol{M} = \frac{1}{m} \sum_{j=1}^{m} \boldsymbol{A}(:, j)$. Ako tada ažuriramo svaki stupac matrice $\boldsymbol{A}$ prema $\boldsymbol{A}(:, j) \leftarrow \boldsymbol{A}(:, j) - \boldsymbol{M}$, tada kažemo da je $\boldsymbol{A}$ u devijacijskoj formi srednje vrijednosti. Kovarijanca matrice $\boldsymbol{A}$ je dana s: $\boldsymbol{C} = \frac{1}{m-1} \boldsymbol{A}\boldsymbol{A}^T$.

PCA algoritam

Glavna ideja je $\boldsymbol{A}(:,j) \approx \boldsymbol{U}_r \boldsymbol{U}_r^T \boldsymbol{A}(:,j)$ gdje je $\boldsymbol{U}_r = \boldsymbol{U}(:,1:r)$ za neki $r \ll m$. Matrica $\boldsymbol{U}_r \boldsymbol{U}_r^T$ je ortogonalni projektor na $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r$, prostor razapet s prvih r glavnih komponenata. Također, primijetimo:

$$\boldsymbol{A}(:,j) \approx \boldsymbol{U}_r \boldsymbol{U}_r^T \boldsymbol{A}(:,j) = \sum_{i=1}^{r} c_i \boldsymbol{u}_i \qquad c_i = \boldsymbol{u}_i^T \boldsymbol{A}(:,j) \tag{3}$$

PCA algoritam

Algorithm 1: Traditional Matrix PCA Method

Input: Training: $\mathbf{I}_i$, $i = 1, 2, ..., N$, Testing: $\mathbf{J}$, Truncation index k
Output: Comparison of Test Image with Compressed Training Set

for $i = 1$ **to** N **do**
  $\mathbf{L}(:, i) \leftarrow$ vectorized $\mathbf{I}_i$
end
$\mathbf{M} \leftarrow$ mean image
$\mathbf{A} \leftarrow$ mean-deviation form of $\mathbf{L}$
$\mathbf{U} \leftarrow$ left singular vectors of $\mathbf{A}$
$\mathbf{G} \leftarrow \mathbf{U}(:, 1\!:\!k)^T \mathbf{A}$
$\mathbf{T} \leftarrow \mathbf{J} - \mathbf{M}$
$\mathbf{t} \leftarrow$ vectorized form of $\mathbf{T}$
$\mathbf{c} \leftarrow \mathbf{U}(:, 1\!:\!k)^T \mathbf{t}$
for $j = 1$ **to** N **do**
  Calculate $||c - \mathbf{G}(:, j)||_F$
end
Claim the training image whose coefficient is closest to that of the test image is the match.

Tradicionalni algoritmi - TensorFaces

- Prvi algoritam koji se koristio manipulacijom tenzora za prepoznavanje lica
- Podatci su predstavljeni tenzorom s različitim modovima za različite faktore
- Aproksimacija tenzora obavlja se multidimenzionalnim SVD višeg reda - HOSVD-om, koji je Tuckerova reprezentacija

TensorFaces

Koristimo tenzor $\mathcal{L}$ da bi specificirali ljude, točke gledišta, osvijetljenje, ekspresije i piksele, redom, a P, V, I, E su brojevi ljudi, točaka gledišta, osvijetljenja, ekspresije i piksela u podatcima za treniranje, dok su P', V', I', E' isti ti koeficijenti za podatke za testiranje. *Pix* je duljina svake vektorizirane slike. Tada možemo izračunati jezgreni tenzor:

$$\mathcal{Z} = \mathcal{L} \times_1 \boldsymbol{U}_p^T \times_2 \boldsymbol{U}_v^T \times_3 \boldsymbol{U}_i^T \times_4 \boldsymbol{U}_e^T \times_5 \boldsymbol{U}_{pix}^T \tag{4}$$

TensorFaces

$\mathcal{B} = \mathcal{Z} \times_2 \boldsymbol{U}_v \times_3 \boldsymbol{U}_i \times_4 \boldsymbol{U}_e \times_5 \boldsymbol{U}_{pix}$ definira $V \times I \times E$ različite baze za svaku kombinaciju točaka gledišta, osvijetljenja i ekspresija, a sub tenzor $\mathcal{B}_{v,i,e}$ veličine $P \times 1 \times 1 \times 1 \times Pix$ za bilo koji dani v, i, e. Za svaku danu sliku, koristi se spljoštena matrica $\boldsymbol{B}^{\dagger}_{v,i,e}$ tenzora $\mathcal{B}_{v,i,e}$ za projiciranje u skup vektora koeficijenata $c_{v,i,e}$ za svaku v, i, e kombinaciju. Zatim onaj koji ima najmanji $c_{v,i,e} - \boldsymbol{U}_p(p,:)^T$ identificira nepoznatu sliku kao osobu p jer su vektor retci matrice $\boldsymbol{U}_p$ koeficijenti za svaku osobu p.

TensorFaces algoritam

Algorithm 2: TensorFaces Method

Input: Training images $\mathbf{I}_{p,v,i,e}$, $p = 1, ..., P$, $v = 1, ..., V$, $i = 1, ..., I$, $e = 1, ..., E$;
Testing Image $\mathbf{J}$
Output: Coefficients of testing images with some tensor basis and comparison

for $p = 1, \ldots, P, v = 1, \ldots, V, i = 1, \ldots, I, e = 1, \ldots, E$ **do**
  $\mathcal{L}(p, v, i, e, :) \leftarrow$ vectorized $\mathbf{I}_{p,v,i,e}$
end
Do the HOSVD to get the core tensor $\mathcal{Z}$ and orthogonal factor matrices $\mathbf{U}_i$:
$\mathcal{L} \leftarrow \mathcal{Z} \times_1 \mathbf{U}_p \times_2 \mathbf{U}_v \times_3 \mathbf{U}_i \times_4 \mathbf{U}_e \times_5 \mathbf{U}_{pix}$
$\mathcal{B} \leftarrow \mathcal{Z} \times_2 \mathbf{U}_v \times_3 \mathbf{U}_i \times_4 \mathbf{U}_e \times_5 \mathbf{U}_{pix}$
$\mathcal{T}(1, 1, 1, 1, :) \leftarrow$ vectorized testing image
for $v = 1, \ldots, V, i = 1, \ldots, I, e = 1, \ldots, E$; **do**
  $c_{v,i,e} \leftarrow (\mathcal{B}^{\dagger}_{v,i,e})^T \mathcal{T}(1, 1, 1, 1, :)$
  for $p = 1, \ldots, P$ **do**
    Calculate $||c_{v,i,e} - \mathbf{U}_p(p, :)^T||_F$
  end
end
Claim the training image whose coefficient is closest to that of the test image is the match.

Novi pristup - Tensor Framework

Definition

Neka je $\mathcal{A}$ tenzor veličine $l \times p \times n$ i $\mathcal{B}$ tenzor veličine $p \times m \times n$. Tada je tenzor-produkt $\mathcal{A} * \mathcal{B}$ $l \times m \times n$ tenzor:

$$\mathcal{A} * \mathcal{B} = fold(bcirc(\mathcal{A}) \cdot unfold(\mathcal{B})) \tag{5}$$

Neka je $\mathcal{A} \in \mathbb{R}^{l \times m \times n}$ u $\boldsymbol{F_n}$ $n \times n$ DFT matrica. Sada je birc($\mathcal{A}$) blok matrica $n \times n$ koja ima $l \times m$ blokova. Neka $\boldsymbol{P}_1 i \boldsymbol{P}_2$ označavaju permutacije po koracima tako da je:

$$\boldsymbol{P}_1 \cdot birc(\mathcal{A}) \cdot \boldsymbol{P}_2 = \begin{bmatrix} N_{11} & N_{12} & \dots N_{1m} \\ N_{21} & N_{22} & \dots N_{2m} \\ \vdots & \vdots & \ddots \\ N_{l1} & N_{l2} & \dots N_{lm} \end{bmatrix} \tag{6}$$

Vrijedi:

$$\boldsymbol{P}_2^T \cdot (\boldsymbol{F_n^*} \otimes \boldsymbol{I_n}) \cdot \boldsymbol{P}_2 = \boldsymbol{I_m} \otimes \boldsymbol{F_n^*} \tag{7}$$

Primijetimo da je:

$$\boldsymbol{P}_1 \cdot (\boldsymbol{F_n} \otimes \boldsymbol{I_l}) \cdot {\boldsymbol{P}_1}^T \cdot \boldsymbol{P}_1 \cdot bcirc(\mathcal{A}) \cdot \boldsymbol{P}_2 \cdot {\boldsymbol{P}_2}^T \cdot (\boldsymbol{F_n^*} \otimes \boldsymbol{I_m}) \cdot \boldsymbol{P}_2 \tag{8}$$

jednako

$$(\boldsymbol{I_l} \otimes \boldsymbol{F_n}) \begin{bmatrix} N_{11} & N_{12} & \dots N_{1m} \\ N_{21} & N_{22} & \dots N_{2m} \\ \vdots & \vdots & \ddots \\ N_{l1} & N_{l2} & \dots N_{lm} \end{bmatrix} (\boldsymbol{I_m} \otimes {\boldsymbol{F_n}}^*) = \begin{bmatrix} D_{11} & D_{12} & \dots D_{1m} \\ D_{21} & D_{22} & \dots D_{2m} \\ \vdots & \vdots & \ddots \\ D_{l1} & D_{l2} & \dots D_{lm} \end{bmatrix} \tag{9}$$

$$(\boldsymbol{F}_{\boldsymbol{n}}^* \otimes \boldsymbol{I}_{\boldsymbol{l}}) \cdot birc(\mathcal{A}) \cdot (\boldsymbol{F}_{\boldsymbol{n}}^* \otimes \boldsymbol{I}_{\boldsymbol{m}}) =$$

$$\boldsymbol{P}_1^T \begin{bmatrix} D_{11} & D_{12} & \dots D_{1m} \\ D_{21} & D_{22} & \dots D_{2m} \\ \vdots & \vdots & \ddots \\ D_{l1} & D_{l2} & \dots D_{lm} \end{bmatrix} \boldsymbol{P}_2^T = \begin{bmatrix} A^{(1)} & & \\ & \ddots & \\ & & A^{(n)} \end{bmatrix} \quad (10)$$

Neka je $\hat{\mathcal{A}} = fft(\mathcal{A}, [], 3)$, te za svaki i $\hat{\boldsymbol{A}} = \hat{\mathcal{A}}(:,:,i)$. Kada su tenzori gusti, mogli bismo izračunati $\mathcal{A} * \mathcal{B}$ računanjem FFT-a duž svake niti $\mathcal{A} i \mathcal{B}$ kako bi dobili $\hat{\mathcal{A}} i \hat{\mathcal{B}}$, pomnožiti svaki par lica $\hat{\mathcal{A}} i \hat{\mathcal{B}}$ kako bi dobili $\hat{\mathcal{C}}$, te uzeti inverzni FFT duž niti $=> O(lpmn + (lp + pm)nlog(n)) \approx O(lpmn)$.

Tenzor-tenzor produkt, koristeći FFT

Algorithm 3: Tensor-Tensor Product, using Fourier Domain Computation

Input: $\mathcal{A} \in \mathbb{R}^{\ell \times p \times n}, \mathcal{B} \in \mathbb{R}^{p \times m \times n}$
Output: $\mathcal{C} := \mathcal{A} * \mathcal{B}$

$\hat{\mathcal{A}} \leftarrow \text{fft}(\mathcal{A}, [\,], 3)$; $\hat{\mathcal{B}} \leftarrow \text{fft}(\mathcal{B}, [\,], 3)$
for $i = 1$ **to** n **do**
    $\hat{\mathcal{C}}(:,:,i) \leftarrow \hat{\mathcal{A}}(:,:,i)\hat{\mathcal{B}}(:,:,i)$
end
$\mathcal{C} \leftarrow \text{ifft}(\hat{\mathcal{C}}, [\,], 3)$

Lemma

$\mathcal{A} * (\mathcal{B} * \mathcal{C}) = (\mathcal{A} * \mathcal{B}) * \mathcal{C}$

Definition

Ako je $\mathcal{A}$ $l \times m \times n$, tada je $\mathcal{A}^T$ $m \times l \times n$ tenzor nastao transponiranjem svakom frontalnog slicea, te mijenjanjem poretka transponiranih frontalnih sliceova od 2. do n-tog.

Definition

Tenzor je f-dijagonalan ako je svaki frontalni slice dijagonalna matrica.

Definition

$l \times l \times n$ tenzor $\mathcal{I}_{lln}$ je tenzor identitete čiji je prvi frontalni slice $l \times l$ matrica identiteta, te su ostali frontalni sliceovi 0. Slično, tenzor je f-gornjetrokutast ako je svaki frontalni slice gornjetrokutast.

Definition

$l \times l \times n$ $\mathcal{Q}$ realni tenzor je ortogonalan ako vrijedi:

$$\mathcal{Q}^T * \mathcal{Q} = \mathcal{Q} * \mathcal{Q}^T = \mathcal{I} \tag{11}$$

T-SVD

T-SVD je tenzor dekompozicija bazirana na već spomenutom tenzor-tenzor produktu, što je zapravo analogon matričnom SVD-u.

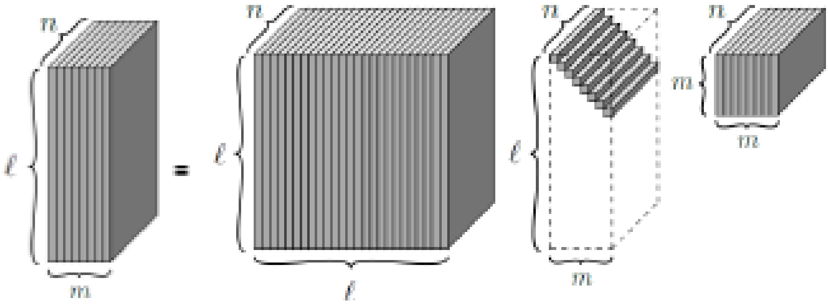
Theorem

Neka je $\mathcal{A}$ realni $l \times m \times n$ tenzor. Tada se $\mathcal{A}$ može faktorizirati kao:

$$\mathcal{A} = \mathcal{U} * \mathcal{S} * \mathcal{V}^T \tag{12}$$

gdje je $\mathcal{U}$ ortogonalni $l \times l \times n$ tenzor, $\mathcal{V}$ ortogonalni $m \times m \times n$ tenzor, a $\mathcal{S}$ f-dijagonalni $l \times m \times n$ tenzor.

T-SVD



Theorem

Neka je $T-SVD$ *tenzora* $\mathcal{A} \in \mathbb{R}^{l \times m \times n}$ *dan s* $\mathcal{A} = \mathcal{U} * \mathcal{S} * \mathcal{V}^T$ *i za*

$k < min(l, m)$ *definiramo:*

$$\mathcal{A}_k = \sum_{i=1}^{k} \mathcal{U}(:, i, :) * \mathcal{S}(i, i, :) * \mathcal{V}(:, i, :)^T \tag{13}$$

Tada je $\mathcal{A}_k = arg \min_{\bar{\mathcal{A}} \in M} ||\mathcal{A} - \bar{\mathcal{A}}||_F$ *gdje je*
$M = \{\mathcal{C} = \mathcal{X} * \mathcal{Y} | \mathcal{X} \in \mathbb{R}^{l \times k \times n}, \mathcal{Y} \in \mathbb{R}^{k \times m \times n}\}$.

Novi pristup baziran na TSVD

Budući ćemo koristiti matrice orijentirane 3. dimenziji, uvodimo novu operaciju: "squeeze" operacija tenzora $\mathcal{X} \in \mathbb{R}^{l \times 1 \times n}$ proizvodi matricu:

$$\boldsymbol{X} := squeeze(\mathcal{X}) \longrightarrow \boldsymbol{X}(i,j) := \mathcal{X}(i,1,j) \tag{14}$$

dok je twist operacije inverz squeeze operacije:

$$twist(squeeze(\mathcal{X})) = \mathcal{X} \tag{15}$$

Neka je $\boldsymbol{M}$ srednja vrijednost slike, $\mathcal{A}(:,j,:) = twist(\boldsymbol{X}_j - \boldsymbol{M}), j = 1, \ldots, m$.

Tada su uzorci reprezentirani vektor stupcima, duljine l, a ulazi su "tube"

- Analogno kovarijacijskoj matrici u tradicionalnom PCA, $\mathcal{K} := \frac{1}{m-1}\mathcal{A} * \mathcal{A}^T$
- Vrijedi: $\mathcal{K} = \mathcal{U} * \mathcal{D} * \mathcal{U}^T$
- Ne računamo svojstveno dekompoziciju od $\mathcal{K}$, već koristimo tenzor verziju SVDa: $\mathcal{A} = \mathcal{U} * \mathcal{S} * \mathcal{V}^T$.
- Iz $\mathcal{K} := \frac{1}{m-1}\mathcal{A} * \mathcal{A}^T = \frac{1}{m-1}\mathcal{U} * \mathcal{S} * \mathcal{S}^T * \mathcal{U}^T$ slijedi da su ne-nul svojstvene vrijednosti od $\mathcal{K}$ kvadrati pripadnih singularnih vrijednosti od $\mathcal{A}$, a lijeva ortogonalna matrica $\mathcal{U}$ sadrži glavne komponente $\mathcal{K}$.

$\mathcal{U}(:, 1 : k, :)$ je ortogonalni skup lateralnih sliceova $\mathcal{U}(:, j, :) \in \mathbb{R}^{l \times 1 \times n}$ i $\mathcal{U}(:, 1 : k, :) * \mathcal{U}(:, 1 : k, :)^T$ definira ortogonalni projektor koji projicira centriranu sliku na prostor manje dimenzije. Ukoliko koristimo prvih nekoliko lijevih singularnih lateralnih sliceova od $\mathcal{U}$ kao novu bazu, tada:

$$\mathcal{A}(:, j, :) \approx \mathcal{U}(:, 1 : k, :) * \mathcal{U}(:, 1 : k, :)^T * \mathcal{A}(:, j, :) = \sum_{t=1}^{k} \mathcal{U}(:, t, :) * \mathcal{C}(t, j, :) \tag{16}$$

T-SVD algoritam 1

Algorithm 4: T-SVD Method I

Input: Training: $\mathbf{I}_i$, $i = 1, 2, ..., N$; Test image $\mathbf{J}$, Truncation index k
Output: Match of Test image against Compressed Representation of Training Set

for $i = 1$ **to** N **do**
  $\mathcal{L}(:, i, :) \leftarrow \mathbf{I}_i$
end
$\mathcal{M} \leftarrow$ mean image
$\mathcal{A} \leftarrow$ mean-deviation form of $\mathcal{L}$
$\mathcal{U} \leftarrow$ left singular vectors of tensor $\mathcal{A}$
$\mathcal{C} \leftarrow \mathcal{U}(:, 1:k, :)^T * \mathcal{A}$
$\mathcal{T}(:, 1, :) \leftarrow \text{twist}(\mathbf{J} - \mathcal{M})$
$\mathcal{B} \leftarrow \mathcal{U}(:, 1:k, :)^T * \mathcal{T}$
for $j = 1$ **to** N **do**
  Calculate $||\mathcal{B} - \mathcal{C}(:, j, :)||_F$
end
Claim the training image whose coefficient is closest to that of the test image is the match.

Lemma

Kada su slike jednodimenzionalni signali ($n = 1$), tenzorski SVD je tada zapravo PCA.

Neka je $\boldsymbol{Y}_j := squeeze(\mathcal{A}(:, j, :))$. Neka je $circ(v)$ cirkularna matrica čiji je prvi stupac v. Iz 16, definicije $*$ i malo manipulacije, vrijedi:

$$\boldsymbol{Y}_j \approx \sum_{t=1}^{k} \boldsymbol{W}_t circ(\boldsymbol{c_{t,j}}) \tag{17}$$

Theorem

Neka je $\boldsymbol{F}$ $n \times n$ *DFT matrica. S* $\hat{\boldsymbol{Y}}_j := \boldsymbol{Y}_j\boldsymbol{F}$, $\hat{\boldsymbol{W}}_j := \boldsymbol{W}_j\boldsymbol{F}$ *i* $\hat{\boldsymbol{c}_{t,j}}$

Fourierovi koeficijenti vektor stupca $\boldsymbol{c}_{t,j}$ *duljine* n*,*

$$\hat{\boldsymbol{Y}}_j \approx \left[\sum_{t=1}^{k} \hat{\boldsymbol{c}_{t,j}}^{(1)} \hat{\boldsymbol{U}}^{(1)}(:,t), \ldots, \sum_{t=1}^{k} \hat{\boldsymbol{c}_{t,j}}^{(n)} \hat{\boldsymbol{U}}^{(n)}(:,t) \right] \tag{18}$$

gdje $\hat{\boldsymbol{c}_{t,j}}^{(i)}$ *označava* $i-$ *tu komponentu tog vektora, a* $\hat{\boldsymbol{W}}_t(:,i) = \hat{\boldsymbol{U}^{(i)}}(:,t)$ *gdje je* $\hat{\boldsymbol{U}}^{(i)}$ *unitarna matrica. Slijedi da je Algoritam 2 ekvivalentan somultanoj (i nezavisnoj) PCA metodi na svakom stupcu transformiranje slike, kada oduzmemo srednju vrijednost, u Fourierovom prostoru, uzimajući isto skraćenje indeksa k na svakom stupcu.*

Posljedice teorema

- Nema gubitka u postupku prepoznavanja ako radimo u Fourierovoj domeni
- Usporedba koeficijenata proširenja testne slike $\boldsymbol{J}$ s trening koeficijentom tenzora može se napraviti u Fourierovoj domeni.
- Zbog konjugirane simetrije, samo se polovica produkata matrice mora izvršiti da bi izračunali $\hat{\mathcal{Z}}$
- Spremanje prvih $\frac{n}{2}+1$ frontalnih sliceova zahtijeva istu količinu prostora za pohranu kao i spremanje $\mathcal{U}, \mathcal{C}$

Posljedice teorema

- Mijenjamo 16 sa:

$$\hat{\boldsymbol{Y}}_j \approx \Big[\sum_{t=1}^{k_1} \hat{\boldsymbol{c}_{t,j}}^{(1)} \hat{\boldsymbol{U}}^{(1)}(:, t), \ldots, \sum_{t=1}^{k_n} \hat{\boldsymbol{c}_{t,j}}^{(n)} \hat{\boldsymbol{U}}^{(n)}(:, t)\Big] \tag{19}$$

- Zahtjeve za pohranom možemo opisati na slijedeći način. Moramo pohraniti:

$$K = k_1 + 2k_2 + \ldots + 2k_{\frac{n}{2}} + 2k_{\frac{n}{2}+1} \tag{20}$$

- Iz svake $\hat{\boldsymbol{U}}^i$, $i = 1, 2, \ldots, \frac{n}{2} + 1$ moramo spremiti samo k_i stupaca. Ovo zahtijeva pohranu lK double precision brojeva.
- Za m = broj trening slika, moramo spremiti $lK + mK$ brojeva.

Preostaje nam vidjeti kako identificirati parametre $k_i, i = 1, \ldots, \frac{n}{2} + 1$.

$$||\hat{\mathcal{A}}(:,:,1:\frac{n}{2}+1)||_F^2 = \sum_{i=1}^{\frac{n}{2}+1} ||\hat{\mathcal{A}}(:,:,i)||_F^2 = \sum_{i=1}^{\frac{n}{2}+1} \sum_{j=1}^{min(l,m)} (\hat{\boldsymbol{s}}_j^i)^2 \tag{21}$$

Dakle, da bi optimalno definirali indekse skraćivanja, promotrimo mjerenje relativne energije:

$$\frac{\sum_{i=1}^{\frac{n}{2}+1} \sum_{j=1}^{k_i} (\hat{\boldsymbol{s}}_j^i)^2}{||\hat{\mathcal{A}}(:,:,1:\frac{n}{2}+1)||_F^2} \tag{22}$$

Nova metoda bazirana na ovoj strategiji opisana je Algoritmom 5.

T-SVD algoritam 2

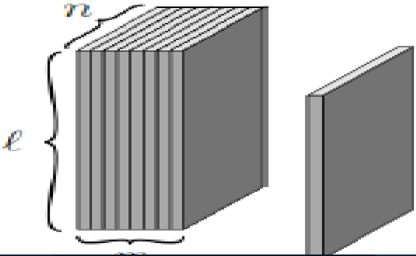
Algorithm 5: T-SVD Method II

Input: Training: $\mathbf{I}_i$, $i = 1, 2, ..., N$; Test image $\mathbf{J}$, Truncation index k
Output: Fourier Domain Comparison of Test image against Compressed Representation of Training Set

for $i = 1$ **to** N **do**
  $\mathcal{L}(:, i, :) \leftarrow \mathbf{I}_i$
end
$\mathcal{M} \leftarrow$ mean image
$\mathcal{A} \leftarrow$ mean-deviation form of $\mathcal{L}$
Frontal face SVDs of $\hat{\mathbf{A}}^{(i)} \leftarrow \hat{\mathcal{A}}(:, :, i)$, $i = 1, \ldots \frac{n}{2} + 1$
Employ drop strategy; keep only $\hat{\mathbf{U}}^{(i)}(:, 1 : k_i)$
$\mathcal{T}(:, 1, :) \leftarrow \text{twist}(\mathbf{J} - \mathcal{M})$
for $i = 1 : \frac{n}{2} + 1$, $t = 1 : k_i$, $j = 1 : N$ **do**
  $\hat{\mathbf{c}}_{t,j}^{(i)} \leftarrow (\hat{\mathbf{U}}^{(i)}(:, t))^H \hat{\mathbf{A}}^{(i)}(:, j)$
end
for $i = 1 : \frac{n}{2} + 1$, $t = 1 : k_i$, **do**
  $\hat{\mathbf{b}_t}^{(i)} \leftarrow (\hat{\mathbf{U}}^{(i)}(:, t))^H \hat{\mathbf{T}}^{(i)}(:, 1)$
end
for $j = 1 : N$ **do**
  Calculate $||\hat{\mathbf{b}_t}^{(i)} - \hat{\mathbf{c}}_{t,j}^{(i)}||_F$
end

Alternativa - tenzor OR pivotiranje

Razlog je potencijalna prednost updatea i downdate baze podataka. Mana ove metode jest da optimizacija neće biti optimalna.



Permutacijski tenzor $\mathcal{P} \in \mathbb{R}^{m \times m \times n}$ je tenzor čiji se ulazi sastoje samo od 0 i 1, i za koji vrijedi $\mathcal{P}^T * \mathcal{P} = \mathcal{P} * \mathcal{P}^T = \mathcal{I}$.

Theorem

Neka je $\mathcal{A}$ $l \times m \times n$ realni tenzor. Tada se $\mathcal{A}$ može faktorizirati na slijedeći način:

$$\mathcal{A} * \mathcal{P} = \mathcal{Q} * \mathcal{R} \tag{23}$$

gdje je $\mathcal{Q}$ $l \times l \times n$ ortogonalni tenzor, $\mathcal{R}$ je $l \times m \times n$ f-gornjetrokutasti tenzor, a $\mathcal{P}$ je permutacijski tenzor.

Tensor QR metoda

Algorithm 6: Tensor QR Method

Input: Training: $\mathbf{I}_i$, $i = 1, 2, ..., N$, Testing: $\mathbf{J}$, Truncation index k
Output: Coefficients of testing images with a basis of reduced dimension

for $i = 1$ **to** N **do**
  $\mathcal{L}(:, i, :) \leftarrow \mathbf{I}_i$
end
$\mathcal{M} \leftarrow$ mean image
$\mathcal{A} \leftarrow$ mean-deviation form of $\mathcal{L}$
$[\mathcal{Q}, \mathcal{R}] \leftarrow$ tensor (pivoted) QR decomposition of $\mathcal{A}$
$\mathcal{C} \leftarrow \mathcal{Q}(:, 1:k, :)^T * \mathcal{A}$
$\mathcal{T}(:, 1, :) \leftarrow \text{twist}(\mathbf{J} - \mathcal{M})$
$\mathcal{B} \leftarrow \mathcal{Q}(:, 1:k, :)^T * \mathcal{T}$
for $j = 1$ **to** N **do**
  Calculate $||\mathcal{B} - \mathcal{C}(:, j, :)||_F$
end
Claim the training image whose coefficient is closest to that of the test image is the match.