Prepoznavanje ljudskih kretnji korištenjem analize tenzora

12. prosinca 2019.

Ana Parać
Sara Pužar
Petra Rožić
Paula Vujasić

Sadržaj

Sažetak

U novije vrijeme automatsko prepoznavanje ljudskih kretnji postalo je predmet velikog zanimanja. Ljudske se kretnje prirodno mogu prikazati pomoću tenzora trećeg reda. U ovom radu opisujemo jednu od metoda temeljenih na obilježjima oblika, gdje se analizira prostorni i vremenski odnos silueta. Koristimo MPCA algoritam kako bismo reducirali dimenziju prostora i izvukli bitne značajke videa.

1 Opis problema

1.1 Uvod

U današnje vrijeme većina je javnih i privatnih objekata pod videonadzorom. Iza kamera se uvijek nalaze osobe koje su zadužene za nadzor objekata i uočavanje opasnosti i prijetnji. Ručna analiza videozapisa zamoran je posao često sklon pogreškama. Kako bi se ublažile ljudske pogreške u tumačenju videosnimki, automatska klasifikacija ljudskog djelovanja postala je jednom od najistraživanijih tema u područjima obrade slike i prepoznavanja uzoraka. Predlažu se različite metode za konstruiranje klasifikacije ljudskog djelovanja. Uglavnom se mogu podijeliti u dvije grupe:

- metode temeljene na obilježjima pokreta – temelje se na informacijama o karakteristikama ljudskih pokreta, kao što su smjer kretanja i brzina
- metode temeljene na obilježjima oblika – temelje se na geometrijskim informacijama o ljudskoj posturi, kao što su siluete i profili

Tenzori se sve češće koriste u području prepoznavanja, primjerice za prepoznavanje lica i prepoznavanje ljudskog djelovanja.

Problem analize ljudskih kretnji rješavao se pomoću raznih algoritama. Neki od njih su:

- Discriminant Tensor Subspace Analysis (DTSA)
- Multilinear Principal Component Analysis (MPCA)

U ovom radu koristit ćemo MPCA algoritam. Svaki video koji prikazuje neku kretnju (mahanje, skakanje, hodanje,...) prikazujemo kao niz frame-ova, odnosno kao tenzor trećeg reda, gdje su prva i druga dimenzija visina i širina slike, a treća dimenzija predstavlja komponentu vremena. Cilj je video projicirati u potprostor tenzora manjih dimenzija, pri čemu ćemo koristiti MPCA algoritam kako bismo reducirali dimenziju prostora i izvukli one značajke videa koje su bitne i karakteristične za video.

1.2 Matematičko modeliranje problema

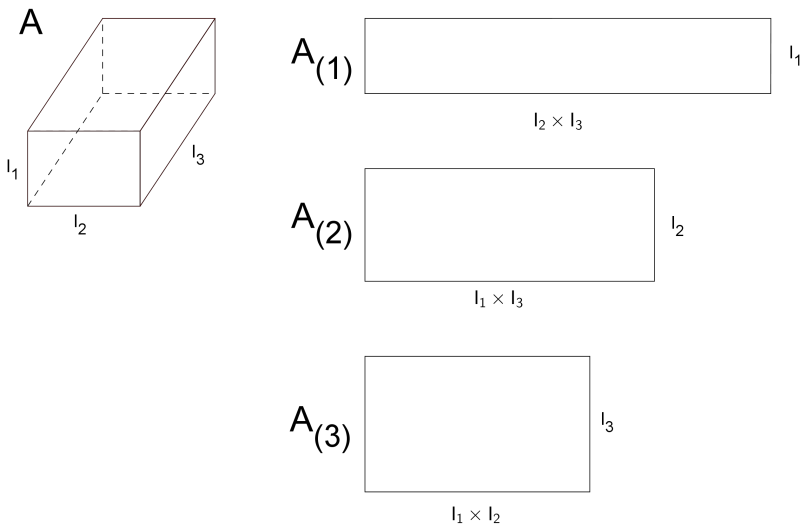
Matematičko modeliranje teme započinjemo definicijom tenzora.

Definicija 1 (Tenzor). Tenzor reda N je produkt od N vektorskih prostora

$$A \in \mathbf{R}^{I_1 \times I_2 \ldots \times I_N},$$

element tenzora A označavamo sa $A_{i_1 i_2 \ldots i_N}$, pri čemu je $0 < i_n \leq I_n, n = 1, 2, \ldots N$

U ovome radu koristit ćemo tenzore trećega reda te niti i odsječke u modu $1, 2, 3$ tenzora A.



Tenzor reda 3 i odsječci u modu 1, 2 i 3

Za tenzore trećeg reda vrijede sljedeće definicije:

Definicija 2 (Niti u modu n). Niti u modu n tenzora $A \in \mathbf{R}^{I_1 \times I_2 \times I_3}$ dio su tenzora dobiven fiksiranjem svih indeksa osim jednog, n-tog indeksa.

Predstavljene su vektorom

$$V_n \in \mathbf{R}^{I_n}.$$

Proces razvijanja niti iz tenzora naziva se vektorizacija tenzora.

Definicija 3 (Odsječci u modu n). Odsječci u modu n tenzora $A \in \mathbf{R}^{I_1 \times I_2 \times I_3}$ dio su tenzora dobiven fiksiranjem n-tog indeksa. Ovisno o modu, odsječke nazivamo redom horizontalni, lateralni ili frontalni odsječak.

Predstavljeni su matricama

$$A_{(n)} \in \mathbf{R}^{I_j \times I_k},$$

gdje $j, k \neq n$. Proces prikazivanja tenzora pomoću matrica naziva se matricizacija tenzora.

Množenje matrice tenzorom provodimo također u određenom modu te dimenzije tenzora i matrice moraju biti odgovarajuće. Neka je $A \in \mathbf{R}^{J_1 \times J_2}$ matrica te $T \in \mathbf{R}^{I_1, I_2, I_3}$ tenzor. Množenje u n-tom modu tada defniramo kao :

$$(T \times_n A)_{j_1, i, j_3} = \sum_{k=1}^{I_n} T_{j_1, k, j_3} \cdot a_{ik}$$

te mora vrijediti $J_2 = I_n$.

Koristeći dekompoziciju tenzora, svaki tenzor se može prikazati pomoću produkta:

$$A = S \times_1 U_1 \times_2 U_2 ... \times_N U_N$$

pri čemu je $U_n, n = 1, 2, ...N$ ortogonalna matrica koja sadrži niti u n-tom modu tenzora A. Matricu $S = A \times_1 U_1^T \times_2 U_2^T ... \times_N U_N^T$ nazivamo jezgrena matrica. Za matricu S kažemo da ima svojstvo potpune ortogonalnosti, što znači da su svaka dva odsječka međusobno okomita.

2 Metode rješavanja

Na početku nalazimo skup tenzora koji najviše pridonose optimizacijskom cilju, odnosno pomoću kojih ćemo najbolje moći konstruirati tenzorski potprostor. Nakon što u *offline* fazi odredimo projekcijske matrice i pomoću njih projiciramo tenzore u nađeni potprostor, u *online* fazi klasificiramo nove tenzore (koje također projiciramo u tenzorski potprostor) na temelju udaljenosti tenzora. Kako bismo odredile udaljenost između tenzora u nađenom potprostoru, koristile smo Frobeniusovu normu. Naposljetku pomoću najmanje udaljenosti određujemo o kojoj se akciji radi.

2.1 Učenje tenzorskog potprostora

Kao što je već napomenuto, video je predstavljen kao tenzor trećeg reda $\chi \in \mathbf{R}^{I_1 \times I_2 \times I_3}$, gdje je prva dimenzija visina slike, druga dimenzija širina slike te treća dimenzija broj slika u videu. Za provedbu algoritma koristimo skup od M tenzora, $\{\chi_1, \chi_2, ..., \chi_M\}$. Provodimo MPCA algoritam s ciljem određivanja projekcijskih matrica $U_1 \in \mathbf{R}^{P_1 \times I_1}$, $U_2 \in \mathbf{R}^{P_2 \times I_2}$, $U_3 \in \mathbf{R}^{P_3 \times I_3}$ potprostora manjih dimenzija.
Vrijednosti U_1, U_2, U_3 inicijaliziramo na P_n svojstvenih vektora koji odgovaraju najznačajnijim svojstvenim vrijednostima sljedeće matrice:

$$\sum_{m=1}^{M} \chi_{m(n)} \cdot \chi_{m(n)}^{T} \cdot$$

Nakon incijalizacije ponavljamo sljedeći postupak. Matrice U_1, U_2 i U_3 prilagođavamo tako što im vrijednost postavimo na P_1, P_2 i P_3 svojstvenih vektora koje odgovaraju svojstvenim vrijednostima sljedećih matrica:

$$\Phi^{(1)} = \sum_{m=1}^{M} (\chi_{m(1)} - \overline{\chi_{(1)}})(U_2 \otimes U_3)(U_2 \otimes U_3)^T (\chi_{m(1)} - \overline{\chi_{(1)}})^T \tag{1}$$

$$\Phi^{(2)} = \sum_{m=1}^{M} (\chi_{m(2)} - \overline{\chi_{(2)}})(U_3 \otimes U_1)(U_3 \otimes U_1)^T (\chi_{m(2)} - \overline{\chi_{(2)}})^T \tag{2}$$

$$\Phi^{(3)} = \sum_{m=1}^{M} (\chi_{m(3)} - \overline{\chi_{(3)}})(U_1 \otimes U_2)(U_1 \otimes U_2)^T (\chi_{m(3)} - \overline{\chi_{(3)}})^T \tag{3}$$

Operacija $\otimes$ predstavlja Kroneckerov produkt definiran kao :
$A \in \mathbf{R}^{m \times n}, B \in \mathbf{R}^{p \times q}, A \otimes B \in \mathbf{R}^{mp \times nq}$

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

Postupak prilagođavanja matrica ponavljamo određeni broj iteracija ili dok se ne postigne konvergencija.
Pomoću nađenih projekcijskih matrica U_1, U_2 i U_3 projiciramo tenzore koji predstavljaju akcije u potprostor manjih dimenzija na sljedeći način:

$$Y_i = \chi_i \times_1 U_1 \times_2 U_2 \times_3 U_3$$

2.2 Klasifikacija na temelju udaljenosti tenzora

Nakon obrade svih uzoraka prikupljenih za treniranje, tenzor testnog videa χ_{test} obrađujemo na isti način te određujemo njegovu projekciju Y_{test} množeći ga matricama U_1, U_2 i U_3.

$$Y_{test} = \chi_{test} \times_1 U_1 \times_2 U_2 \times_3 U_3$$

Nakon redukcije dimenzionalnosti nalazimo udaljenost između projiciranog testnog tenzora Y_{test} i tenzora Y_i izračunatih u fazi učenja tenzorskog potprostora.

$$i^* = arg \min_{i \in M} d(Y_i, Y_{test}),$$

gdje je d Frobeniusova norma.

Na kraju određujemo o kojoj se kretnji radi tako da odabiremo onu kojoj odgovara tenzor Y_i s najmanjom udaljenosti u odnosu na Y_{test}.

3 Opis implementacije algoritma

Proces implementacije algoritma za prepoznavanje kretnji podijelile smo u nekoliko faza: pretprocesiranje, inicijalizaciju, lokalnu optimizaciju i klasifikaciju. Slijedi pregled navedenih faza te pseudokod.

3.1 Pretprocesiranje

Pretprocesiranje je faza učitavanja i obrade videa. Svaki video je učitan, te je potom podijeljen na manje nizove akcija u trajanjima od 20 *frame*-ova, na način da se videi međusobno preklapaju u zadnjih odnosno prvih 10 *frame*-ova. Savki je *frame* obrađen na način da je silueta istaknuta crnom bojom, te potom centrirana i izrezana.

Algorithm 1: Preprocessing

Data: Video snimke ljudskih kretnji
Result: Tenzori spremni za MPCA algoritam
while *ima još video snimki* **do**
  učitaj trenutnu snimku ;
  podijeli je u više manjih tenzora ;
  isticanje i ekstrakcija siluete ;
end

3.2 Inicijalizacija

U fazi inicijalizacije postavljamo matrice U_1, U_2 i U_3 na početne vrijednosti. Izračunavamo sume matricizarnih tenzora u 1., 2. i 3. modu pomnoženih sa svojim transponentom te provodimo dekompoziciju na svojstvene vrijednosti (EVD). Vrijednosti matrica U_1, U_2 i U_3 postavljamo na P_1, P_2 i P_3 svojstvenih vektora dobivenih EVD-om.

Algorithm 2: Initialization

Data: Tenzori - nizovi slika, χ
Result: Inicijalizirane vrijednosti U_1, U_2, U_3
inicijaliziraj $\Theta_1, \Theta_2, \Theta_3$;
for $i \leftarrow 0$ **to** *broj tenzora* **do**
  Θ_1 +=$\chi_{i(1)} \cdot \chi_{i(1)}^T$;
  Θ_2 +=$\chi_{i(2)} \cdot \chi_{i(2)}^T$;
  Θ_3 +=$\chi_{i(3)} \cdot \chi_{i(3)}^T$;
end
U_1 = najznačajnijih P_1 svojstvenih vektora Θ_1 ;
U_2 = najznačajnijih P_2 svojstvenih vektora Θ_2 ;
U_3 = najznačajnijih P_3 svojstvenih vektora Θ_3 ;

3.3 Lokalna optimizacija

U fazi lokalne optimizacije ažuriramo projekcijske matrice U_n. Matrice ažuriramo koristeći formule (1), (2) i (3) za izračun Φ te potom vrijednosti U_1, U_2 i U_3 postavljamo na vrijednosti P_n vektora koji odgovaraju najznačajnijim svojstvenim vrijednostima. Nakon određivanja projekcijskih matrica, sve tenzore koji predstavljaju neku kretnju, χ_i, množimo njima te tako dobivamo projicirane tenzore Y_i.

Algorithm 4: Local optimization

Data: Početne matrice U_1, U_2, U_3
Result: Projekcijske matrice U_1, U_2, U_3
inicijaliziraj $\Theta_1, \Theta_2, \Theta_3$;
for $i \leftarrow 0$ **to** *broj iteracija* **do**
  izračunaj Φ_1, Φ_2 i Φ_3 prema formulama (1), (2) i (3) ;
  ažuriraj U_1, U_2, U_3 na P_n sv. vektora matrica Φ_n ;
end
for $i \leftarrow 0$ **to** *broj tenzora* **do**
  izračunaj $_i = \chi_1 \times_1 U_1 \times_2 U_2 \times_3 U_3$;
end

3.4 Klasifikacija

Posljednja faza je takozvana *online* faza u kojoj primamo video određene kretnje koji potom klasificiramo kao neku od kretnji koje su naučene. Video obrađujemo na isti način kao i videe za treniranje te ga potom množimo maticama U_1, U_2 i U_3. Dobiveni tenzor nazivamo projekcija i označavamo Y_{test}. Potom računamo udaljenosti između Y_{test} i tenzora Y_i izračunatih u prethodnoj fazi, te vraćamo kretnju kojoj odgovara tenzor Y_i s najmanjom udaljenosti u odnosu na Y_{test}. Udaljenost računamo kao Frobeniusovu normu.

Algorithm 5: Classification

Data: video koji klasficiramo, tenzor χ_{test}
Result: oznaka akcije koju video prikazuje
$Y_{test} = \chi_{test} \times_1 U_1 \times_2 U_2 \times_3 U_3$;
for $i \leftarrow 0$ **to** *broj tenzora* **do**
  izračunaj Frobeniusovu udaljenost između Y_{test} i Y_i ;
end
odredi najmanju udaljenost i odgovarajuću kretnju;

4 Rezultati testiranja

4.1 Struktura preuzetih podataka

Skup podataka preuzet je sa stranice http://www.wisdom.weizmann.ac.il/ vision/SpaceTimeActions.html.

Podaci na kojima smo trenirale i testirale algoritam strukturirani su u 10 mapa koje sadrže videe kretnji:

- Wave1 - mahanje jednom rukom
- Wave2 - mahanje dvijema rukama
- Walk - hodanje s jedne strane na drugu
- Run - trčanje s jedne strane na drugu
- Jump - skok u mjestu
- Pjump - skok s jedne strane na drugu
- Skip - skakutanje na jednoj nozi
- Bend - saginjanje
- Side - skakutanje u stranu
- Jack - vježba *jumping jacks*

Za svaku kretnju skup podataka sadrži 9 ili 10 videa različitih duljina.

4.2 Testiranje na preuzetim podacima

Od cjelokupnog skupa podataka izdvojile smo po dva videa za svaku kretnju za testiranje algoritma, dok smo za treniranje koristile sve ostale videe. Kako skup ne sadrži mnogo videa, a i različitih su duljina, svaki video dijelimo na više manjih u trajanju od 20 *frame*-ova. Time dobivamo veći skup podataka na kojem algoritam uči prepoznavati kretnje.

Videi koji su ostavljeni za testiranje su također podijeljeni u manje segmente, kako bi dobili veći skup podataka za testiranje. Algoritam je testiran u više verzija koje se razliku u broju preklapanja *frame*-ova te vrijednostima P_1, P_2,P_3. U sljedećoj tablici prikazani su neki od dobivenih rezultata. Točnost je izračunata kao omjer točno klasificiranih videa u odnosu na ukupan broj videa.

Točnost algoritma na preuzetim podacima			
	$P_1 = 33$, $P_2 = 28$, $P_3 = 6$	$P_1 = 33$, $P_2 = 30$, $P_3 = 6$	$P_1 = 31$, $P_2 = 22$, $P_3 = 6$
Intervali preklapanja = 10	90.32	90.29	89.32
Intervali preklapanja = 5	82.86	85.44	86.41

4.3 Izrada vlastitog skupa podataka i testiranje

Nakon treniranja i testiranja algoritma na preuzetom setu podataka, snimile smo nekoliko videa kako bismo testirale algoritam i na novim podacima, snimljenim u drugačijim uvjetima. Snimile smo po 2 videa za svaku kretnju. Snimljene videe dodatno smo obradile kako bi im smanjile veličinu i prilagodile format. Svaki je video ponovno podijeljen na manje segmenete na isti način kao i kod preuzetih podataka.
Algoritam je uspješno klasifcirao gotovo sve videe.

Literatura

[1] Mingfang Sun, Sujing Wang, Xiaohua Liu, Chengcheng Jia, and Chunguang Zhou: *Human Action Recognition Using Tensor Principal Component Analysis*, 2012.

[2] Haiping Lu, K.N. Plataniotis and A.N. Venetsanopoulos: *MPCA: Multilinear Principal Component Analysis of Tensor Objects* , 2008.

[3] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri: *Actions as space-time shapes*, 2005.

[4] http://www.wisdom.weizmann.ac.il/ vision/SpaceTimeActions.html, Preuzeto : 4.2019.