Prepoznavanje rukom pisanih znamenki koristeći tenzorski SVD

Marina Matešić, Tomislav Novak, Timotej Repak

2. ožujka 2025.

Sažetak

U ovom radu prikazat ćemo 2 algoritma koji s visokom razinom uspješnosti klasificiraju rukom pisane znamenke, što je jedan od osnovnih problema klasifikacije čije primjene sežu puno dalje od same klasifikacije znamenki, prateći rad naveden pod literaturom. Oba algoritma koriste HOSVD - dekompoziciju tenzora.

Sadržaj

1 Osnovno o tenzorima i potrebni rezultati

U algoritmima koji se koriste u ovom radu, bit će potrebni samo tenzori reda 3 pa ćemo sve općenitije rezultate prikazati u specijalnom slučaju za takve tenzore. U ostatku ovog rada tenzore reda 3 nazivat ćemo samo tenzorima.

Tenzor možemo shvatiti kao trodimenzionalno polje realnih brojeva, tj. tenzor $\mathcal{A} = (a_{ijk}) \in \mathbb{R}^{I\times J\times K}$, gdje su I, J i K prirodni brojevi, a dimenzija vektorskog prostora $\mathbb{R}^{I\times J\times K}$ je IJK.

Za dva tenzora $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{I\times J\times K}$ definiramo njihov skalarni produkt kao

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} a_{ijk}b_{ijk}. \tag{1}$$

Kažemo da su $\mathcal{A}$ i $\mathcal{B}$ ortogonalni ako vrijedi $\langle \mathcal{A}, \mathcal{B} \rangle = 0$.

Analogno vektorskom slučaju, normu tenzora $\mathcal{A}$ definiramo kao $\|\mathcal{A}\| = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}$.

Dijelove tenzora $\mathcal{A}(:, j, k)$, $\mathcal{A}(i, :, k)$ i $\mathcal{A}(i, j, :)$ nazivamo niti u modu 1, 2 i 3, redom.

Matricizacija u n-tom modu tenzora $\mathcal{A}$ je matrica $A_{(n)}$ koju dobijemo tako da niti od $\mathcal{A}$ stavimo kao retke matrice $A_{(n)}$.

n-rang tenzora $\mathcal{A}$ definiramo kao $\text{rank}_n(\mathcal{A}) = \text{rank}(A_{(n)})$. Lako se uoči da općenito ne vrijedi $\text{rank}_1(\mathcal{A}) = \text{rank}_2(\mathcal{A}) = \text{rank}_3(\mathcal{A})$.

Množenjem tenzora $\mathcal{A} \in \mathbb{R}^{I\times J\times K}$ matricom $F \in \mathbb{R}^{L\times I}$ u modu 1 dobije se tenzor $\mathbb{R}^{L\times J\times K} \ni \mathcal{B} = \mathcal{A} \times_1 F, \quad \mathcal{B}(l, j, k) = \sum_{i=1}^{I} \mathcal{A}(i, j, k)F(l, i)$.

Množenjem tenzora $\mathcal{A} \in \mathbb{R}^{I\times J\times K}$ matricom $F \in \mathbb{R}^{L\times J}$ u modu 2 dobije se tenzor $\mathbb{R}^{I\times L\times K} \ni \mathcal{B} = \mathcal{A} \times_2 F, \quad \mathcal{B}(i, l, k) = \sum_{j=1}^{J} \mathcal{A}(i, j, k)F(l, j)$.

Množenjem tenzora $\mathcal{A} \in \mathbb{R}^{I\times J\times K}$ matricom $F \in \mathbb{R}^{L\times K}$ u modu 3 dobije se tenzor $\mathbb{R}^{I\times J\times L} \ni \mathcal{B} = \mathcal{A} \times_3 F, \quad \mathcal{B}(i, j, l) = \sum_{k=1}^{K} \mathcal{A}(i, j, k)F(l, k)$.

Za tenzor $\mathcal{A} \in \mathbb{R}^{I\times J\times K}$ i matrice $F \in \mathbb{R}^{L\times I}$, $G \in \mathbb{R}^{M\times J}$ i $H \in \mathbb{R}^{N\times IL}$ vrijedi $(\mathcal{A} \times_1 F) \times_2 G = (\mathcal{A} \times_2 G) \times_1 F = \mathcal{A} \times_1 F \times_2 G \in \mathbb{R}^{L\times M\times K}$ te $(\mathcal{A} \times_1 F) \times_1 H = \mathcal{A} \times_1 (HF) \in \mathbb{R}^{N\times J\times K}$.

Tenzorski SVD - HOSVD: Tenzor $\mathcal{A} \in \mathbb{R}^{I\times J\times K}$ se može zapisati kao produkt

$$\mathcal{A} = \mathscr{S} \times_1 U \times_2 V \times_3 W, \tag{2}$$

gdje su:

(1) $U \in \mathbb{R}^{I\times I}$, $V \in \mathbb{R}^{J\times J}$ i $W \in \mathbb{R}^{K\times K}$ ortogonalne matrice,

(2) $\mathscr{S}$ je realni tenzor istih dimenzija kao i $\mathcal{A}$ sa sljedećim svojstvima:

(a) (potpuna ortogonalnost) svaka dva odsječka istog tipa su okomiti, tj.

$$\langle \mathscr{S}(v,:,:), \mathscr{S}(\lambda,:,:)\rangle = 0 \quad \text{za } v \neq \lambda,$$
$$\langle \mathscr{S}(:,v,:), \mathscr{S}(:,\lambda,:)\rangle = 0 \quad \text{za } v \neq \lambda,$$
$$\langle \mathscr{S}(:,:,v), \mathscr{S}(:,:,\lambda)\rangle = 0 \quad \text{za } v \neq \lambda.$$

(b) (*uređenost*) norme odsječaka u svakom modu su uređene (poredane), npr. za mod 1 vrijedi $\|\mathscr{S}(1,:,:)\| \geq \|\mathscr{S}(2,:,:)\| \geq \cdots \geq 0$. Te norme u modu n su upravo singularne vrijednosti matriciziranog tenzora $A_{(n)}$.

Zbog svojstva uređenosti, tenzor koji je prikazan u HOSVD formatu aproksimiramo tenzorom manjih dimenzija tako da ga prikažemo kao produkt gore navedenog tenzora i matrica, ali sa "odsječenim" dijelovima koji su dalje od pozicije $(1,1,1)$, odnosno $(1,1)$.

Svaki se tenzor može zapisati u obliku $\mathcal{A} = \sum_{v=1}^{K} A_v \times_3 w_v$, gdje je $A_v = \mathcal{S}(:,:,v) \times_1 U \times_2 V$ te vrijedi $\langle A_v, A_\mu\rangle = 0$.

2 Algoritmi

2.1 Klasifikacija pomoću HOSVD-a

Neka su u bazi rukom pisanih znamenaka sve znamenke prikazane kao matrice tipa $I \times J$ te neka svakih znamenki ima K (ne nužno, ali za trenutnu jednostavnost zapisa). Sve matrice koje predstavljaju istu znamenku onda posložimo u tenzor čiji su one frontalni slice-ovi.

Na primjer, tenzor koji predstavlja broj 7 neka je $\mathcal{A}^7 \in \mathbb{R}^{I\times J\times K}$ te neka imamo njegovu HOSVD dekompoziciju. Po posljednjoj napomeni iz prethodnog poglavlja, taj tenzor zapišemo kao $\mathcal{A}^7 = \sum_{v=1}^{K} A_v^7 \times_3 w_v^7$ što znači da je svaka sedmica na jedinstven način prikazana kao kombinacija matrica A_v^7 koje čine bazu.

Za neki k uzmimo dominantni k-dimenzionalni potprostor za svaki tenzor koji predstavlja znamenku i označimo $T^\mu = (A_v^\mu)_{v=1}^k, \quad \mu = 0, 1, \ldots, 9$. To možemo zato što su norme od matrica te baze redom padajuće, pa ćemo dobiti na vremenu/složenosti uzimajući manju bazu (potprostor manje dimenzije) kao aproksimaciju a pritom odbacujući 'najnebitniji' dio (onaj s najmanje informacija).

Neka je D neka znamenka koju treba klasificirati, normalizirana. Naš je cilj odrediti T^μ koji najbolje opisuje D pa ćemo onda klasificirati D kao matricu koja odgovara znamenki μ.

Pod "najbolje opisuje" smatrat ćemo da najbolje odgovara u smislu najmanjih kvadrata, odn. vidjet ćemo kojim potprostorom možemo opisati matricu D s najmanjom greškom.

Dakle, rješavamo minimizacijsku zadaću $\min_{\alpha_v^\mu} \|D - \sum_{v=1}^{k} \alpha_v^\mu A_v^\mu\|$ čije je rješenje zbog ortonormiranosti dano s $\hat{\alpha}_v^\mu = \langle D, A_v^\mu \rangle$. Odnosno, tražimo μ koji minimizira funkciju $R(\mu) = \|D - \sum_{v=1}^{k} \hat{\alpha}_v^\mu A_v^\mu\|^2 = \left\langle D - \sum_{v=1}^{k} \hat{\alpha}_v^\mu A_v^\mu, D - \sum_{v=1}^{k} \hat{\alpha}_v^\mu A_v^\mu \right\rangle = \langle D, D\rangle - \sum_{v=1}^{k} \langle D, A_v^\mu\rangle^2 = 1 - \sum_{v=1}^{k} \langle D, A_v^\mu\rangle^2$.

2.2 Klasifikacija pomoću HOSVD-a s kompresijom

Kako bismo smanjili složenost, u ovom algoritmu koristit ćemo kompresiju podataka prije izgradnje modela klasa.

2.2.1 Trening faza

Neka je $\mathcal{D}$ tenzor koji sadrži sve znamenke iz trening skupa. Budući da su znamenke grupirane po klasama, možemo zapisati HOSVD dekompoziciju cijelog tenzora:

$$\mathcal{D} = \mathscr{S} \times_1 U \times_2 V \times_3 W \approx \mathcal{F} \times_1 U_p \times_2 V_q, \tag{3}$$

gdje je $U_p = U(:, 1:p)$, $V_q = V(:, 1:q)$ i $\mathcal{F} = \mathscr{S}(1:p, 1:q, :) \times_3 W$.

Ovom aproksimacijom smanjujemo dimenzionalnost prikaza znamenaka na $\mathbb{R}^p$ te broj znamenaka u svakoj klasi na q. Reducirani tenzor $\mathcal{F}$ ima oblik:

$$\mathcal{F} = \mathcal{D} \times_1 U_p^T \times_2 V_q^T. \tag{4}$$

Uz pretpostavku da su p i q znatno manji od dimenzija tenzora $\mathcal{D}$, postiže se značajno smanjenje skupa podataka.

Niskodimenzionalna reprezentacija znamenaka dobiva se kao:

$$D_p = D \times_1 U_p^T = \mathcal{F} \times_2 V_q, \tag{5}$$

gdje su stupci D_p reprezentacije znamenaka, a odsječci $\mathcal{F}$ sadrže bazne vektore za svaku klasu.

Nakon kompresije, za svaku znamenku μ formiramo matricu $F^\mu := \mathcal{F}(:, :, \mu)$ koja sadrži samo znamenke klase μ. SVD dekompozicijom dobivamo ortonormirane baze B^μ za svaku klasu:

$$F^\mu = [B^\mu (B^\mu)^\perp] \Sigma^\mu (Q^\mu)^T, \quad \mu = 0, 1, \ldots, 9, \tag{6}$$

2.2.2 Faza testiranja

Za testiranje, ulaznu znamenku D projiciramo u reducirani prostor pomoću U_p:

$$D_p = U_p^T D. \tag{7}$$

Nakon toga rješavamo niz problema najmanjih kvadrata za svaku znamenku μ:

$$\min_{x^\mu} \|D_p - B^\mu x^\mu\|. \tag{8}$$

Budući da su stupci B^μ ortonormalni, rješenje je dano s:

$$\hat{x}^\mu = (B^\mu)^T D_p. \tag{9}$$

Konačno, znamenku svrstamo u klasu μ za koju je rezidual najmanji:

$$R(\mu) = \|D_p - B^\mu (B^\mu)^T D_p\|. \tag{10}$$

3 Tehnička izvedba algoritma i analiza složenosti

3.1 Tehnička izvedba

Algoritam implementiramo u programskom jeziku Python te isječke koda ne prikazujemo u samom radu, već kod prilažemo kao Python bilježnicu, a ovdje navodimo najbitniju logiku i korištene biblioteke.

Za učitavanje podataka koristimo *SciPy* biblioteku, za spremanje i baratanje podacima *NumPy* i *TensorLy* biblioteke. Koristimo upravo *TensorLy* zato što, uz to što sadrži strukture podataka za operacije nad tenzorima (no nije jedina takva biblioteka), ima implementiranu `tucker` funkciju koja je efektivno HOSVD dekompozicija uz uključeno aproksimiranje manjim rangom (argument `rank`) - što je bolje nego da smo mi ručno računali cijeli HOSVD (ili ga izračunali funkcijom iz neke druge biblioteke) pa onda odrezali dio podataka, jer `tucker` ima optimizacije za brže računanje za dani rang aproksimacije.

Prednost *TensorLy-ja* je također ta što nativno radi s *NumPy* objektima što naši podaci i jesu, bez potrebe za konverzijom u neke specifične strukture kao npr. `TensorFlow.Tensor`.

Također koristimo biblioteku *scikit-learn* za odvajanje skupa podataka na trening i testni dio, Pythonovu biblioteku *time* za mjerenje vremena i *matplotlib* i vanjsku *Seaborn* za prikaz grafova rezultata.

3.2 Analiza složenosti

Spomenimo da testove izvršavamo na modernom procesoru (AMD Ryzen 5 3600 – 6 jezgri, 3.6 GHz) u Windows okruženju.

Označimo prvo veličine u našem kodu (i u zagradama dajemo stvarne vrijednosti za MNIST skup podataka koji koristimo).

- N – broj uzoraka za treniranje (56 000)
- d – jedna dimenzija kvadrata slike znamenke (28)
- K – broj uzoraka jedne znamenke (5000)
- C – broj klasa (10)
- k_1, k_2, k_3 – rangovi aproksimacije
- p, q – parametri kompresije

Koristimo fiksni K u drugom algoritmu zato jer moramo sve klase pospremiti u isti tenzor pa se dimenzija, jasno, mora poklopiti.

3.2.1 Algoritam 1

Trening faza

Za svaku klasu μ formiramo tenzor $A_\mu \in \mathbb{R}^{d\times d\times K}$, gdje i ovdje radi jednostavnosti koristimo fiksni K u analizi jer je odstupanje malo jer ima dovoljno blizu uniformnom broju uzoraka po klasama. Dakle, za to nam treba $O(Cd^2K)$ vremena.

Dalje, za svaki A_μ HOSVD dekompozicija uključuje SVD na *unfoldanim* matricama redova $d \times dK$, $d \times dK$ i $K \times d^2$, pa je to, za sve klase, ukupno $O(CK^2d^2)$ (što je oko 10^{11} operacija), no funkcija `tucker` optimalnije računa aproksimacije manjeg ranga pa se to spusti na oko 10^{10} operacija.

Za konstrukciju baze trebamo načiniti k_3 tenzora ranga 1 A_k po klasi, što je ukupno vremenski $O(Ck_3d^2)$.

Dominira, naravno računanje HOSVD-a i dobivamo očekivane rezultate a to je da nam za trening fazu treba oko 40 sekundi.

Testna faza

Za svaki testni uzorak, projekcija na k_3 matrica zahtijeva $O(kd^2)$ operacija po klasi odn. ukupno $O(Ck_3d^2)$. To je red veličine 10^7 operacija pa jasno da nam treba puno manje od sekunde za jednu klasifikaciju.

3.2.2 Algoritam 2

Trening faza

Za spremanje u tenzor dimenzije $d^2 \times K \times C$ trebamo $O(d^2KC)$ vremena (oko 10^7 operacija).

Za HOSVD također teoretski trebamo 3 SVD-a, teoretske složenosti $O(K^2d^2C)$, što je oko 10^{11} operacija - sličan red veličine kao i od trening faze prvog algoritma, no nemamo ubrzanje jer ovaj put radimo s većim rangovima (p i q su osjetno veći od smanjenih rangova prvog algoritma).

Za računanje matrice F trebamo tenzorsko množenje u modovima koje nas košta – prvo $O(d^2pKC)$, a drugo $O(pqKC)$. Dominira prvo s oko 10^9 operacija.

Dakle, cjelokupna trening faza treba oko 10^{11} operacija, što je u skladu s dobivenom brzinom od oko minutu i pol.

Testna faza

Sada za svaki testni uzorak veličine d^2, projekcija na prostor stupaca veličine p košta $O(d^2p)$ što je samo oko 10^4 operacija – manje no u prvom algoritmu pa i brže, dobivamo 2-3 puta manje vrijeme.

4 Rezultati i zaključak

U ovom radu iznijeli smo dva algoritma i usporedili njihove složenosti i performanse. Algoritmi izvode klasifikaciju znamenaka pomoću HOSVD dekompozicije tenzora, s optimizacijama aproksimacije manjim rangom te kompresijom.

Brži rezultat drugog algoritma komentiran je u odjeljku analize složenosti, a u Python bilježnici nakon samog izvođenja ispisani su rezultati preciznosti, iz kojih se vidi da se nakon što se test drugog algoritma izvodi 2-3 puta brže od testa prvog, ne gubi uvelike na preciznosti – prvi daje 95-96%, a drugi samo malo manje – 94-95%.

5 Literatura

B. Savas and L. Eldén, Handwritten digit classification using higher-order singular value decomposition, *Department of Mathematics, Linköping University, Sweden*, 2006