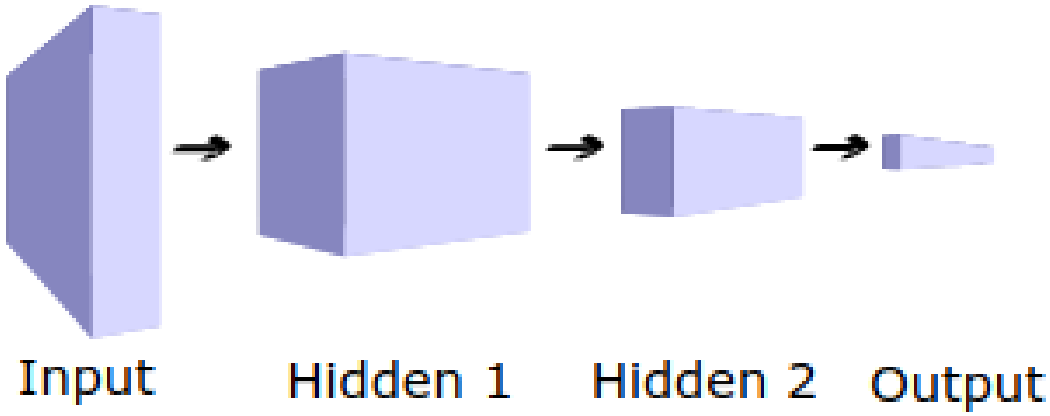
Primjena tenzorskih dekompozicija u ubrzanju konvolucijskih neuronskih mreža

Jan Corazza, Mislav Stojanović

Svibanj, 2023.

Zadatak

Umjetne neuronske mreže s konvolucijskim slojevima dominiraju u prepoznavanju slika [6] [7] i osim toga koriste se u analizi videa, obradi prirodnog jezika, detekciji anomalija, farmakologiji, igranju igara, predviđanju vremenskih nizova...
Konvolucijski slojevi, koji predstavljaju matematičku operaciju kros-korelacije, primarno služe za obradu pikseliziranih podataka. Jedan takav sloj odgovara tenzoru koji ima četiri osi. U ovom radu prikazat ćemo kako se tenzorske dekompozicije mogu primijeniti za redukciju dimenzionalnosti tako da originalni tenzor zamijenimo nizom manjim preslikavanja. Cilj je ubrzati proces treniranja i smanjiti veličinu mreže u memoriji.

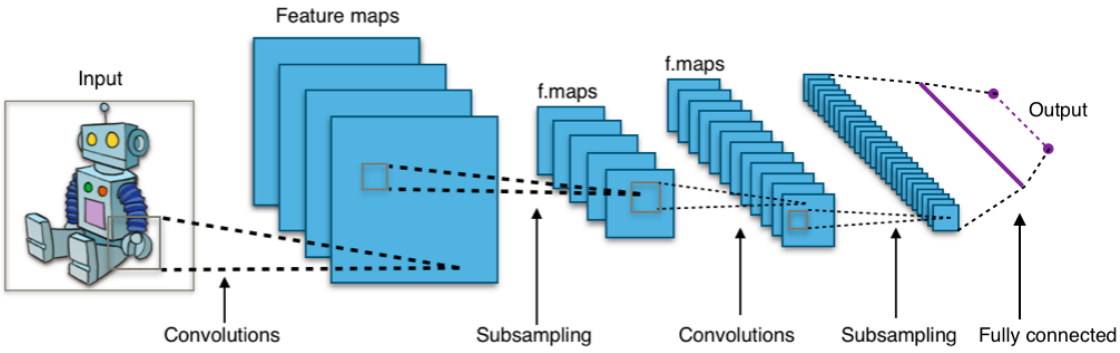


Slika 1: Ilustracija konvolucijske mreže

Konvolucijske neuronske mreže

Osnovno

Konvolucijske neuronske mreže inspirirane su strukturom vizaulnog korteksa u mozgu. Uzimamo djeliće slike i preslikamo ih u nove značajke koristeći konvolucijsku jezgru. Najčešće je to popraćeno slojem koji stapa više značajki u jednu npr. uzimanjem prosjeka ili najveće vrijednosti unutar nekog podskupa (takva operacija se zove *pooling*).



Slika 2: Ilustracija konvolucijske mreže

Linearna algebra

Konvoluciju u jednoj dimenziji možemo zapisati pomoću Toeplitzove matrice. Isto tako, u dvije dimenzije možemo to zapisati pomoću dvostruko-Toeplitzove matrice tj. blok-Toeplitzove matrice kojoj su blokovi Toeplitzove matrice.

$$T = \begin{pmatrix} \mathrm{T}_0 & T_{-1} & T_{-2} & \dots & T_{1-n} \\ T_1 & T_0 & T_{-1} & \dots & T_{2-n} \\ T_2 & T_1 & T_0 & \dots & T_{3-n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ T_{n-1} & T_{n-2} & T_{n-3} & \dots & T_0 \end{pmatrix}, \; T_i \text{ su Toeplitzove matrice.}$$

Slika u boji je zapravo niz triju matrica, gdje pojedina predstavlja R odnosno G ili B komponentu. Tada nam konvolucija kao preslikavanje odgovara tenzoru koji ima četiri osi. Tako možemo zapisati:

$$V(x, y, t) = \sum_i \sum_j \sum_s K(i, j, s, t) X(x - i, y - j, s),$$

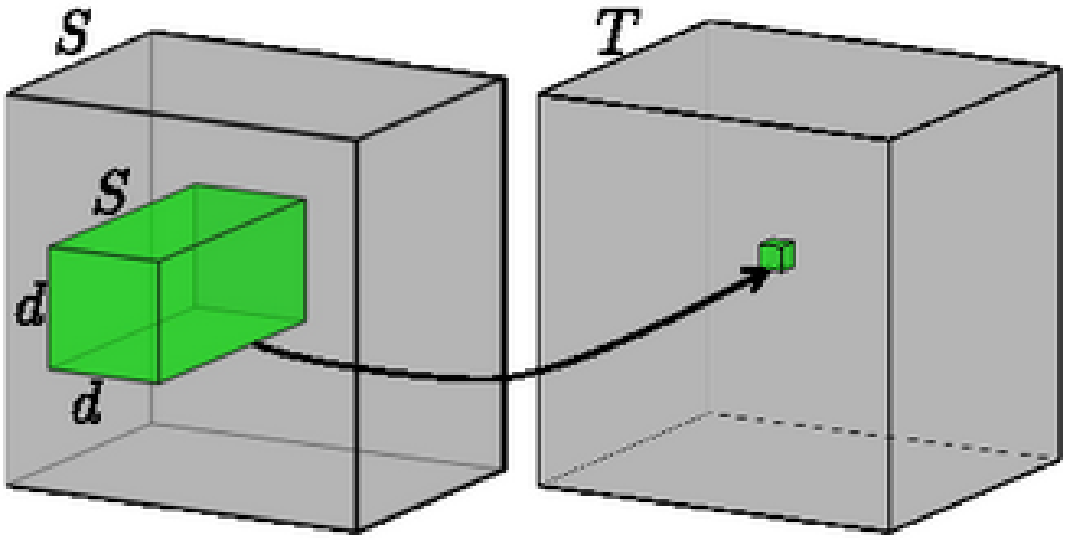
gdje je K tenzor koji predstavlja konvoluciju i X ulazna slika.

CP dekompozicija

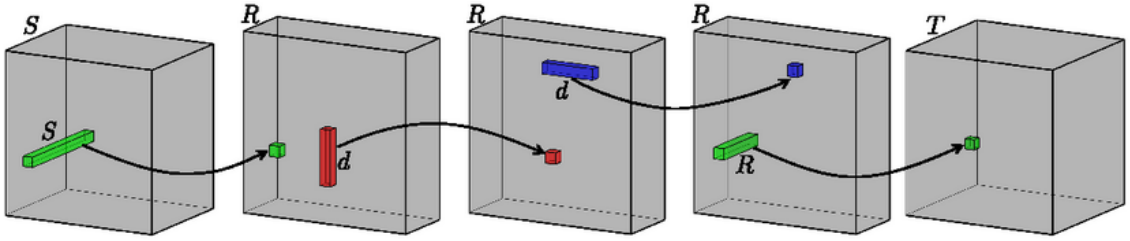
U terminima CP dekompozicije imamo [5] sljedeće

$$
\begin{aligned}
V(x,y,t) &= \sum_i \sum_j \sum_s K(i,j,s,t) X(x-i, y-j, s) \\
&= \sum_r \sum_i \sum_j \sum_s K_r^x(i) K_r^y(j) K_r^s(s) K_r^t(t) X(x-i, y-j, s) \\
&= \sum_r K_r^t(t) \sum_i \sum_j K_r^x(i) K_r^y(j) \sum_s K_r^s(s) X(x-i, y-j, s).
\end{aligned}
$$

Dakle prvo imamo jednu $1 \times 1 \times S$ konvoluciju $K_r^t(s)$ koja smanjuje broj ulaznih kanala sa S na R.
Zatim imamo dubinski separabilne konvolucije s K_r^x i K_r^y.
Naposljetku, s $K_r^s(s)$ vratimo broj kanala s R na originalni T.



Slika 3: Prije dekompozicije



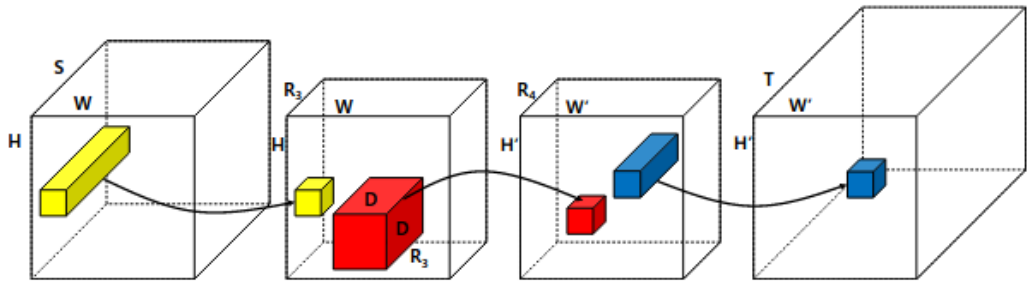
Slika 4: Nakon dekompozicije

Tuckerova dekompozicija

Slično [3] kao ranije

$$V(x,y,t) = \sum_i \sum_j \sum_s K(i,j,s,t) X(x-i, y-j, s)$$
$$= \sum_i \sum_j \sum_s \sum_{r3=1}^{R3} \sum_{r4=1}^{R4} \sigma_{(i)(j)r_3r_4} K^s_{r_3}(s) K^t_{r_4}(t) X(x-i, y-j, s)$$
$$= \sum_i \sum_j \sum_{r3=1}^{R3} \sum_{r4=1}^{R4} K^t_{r_4}(t) \sigma_{(i)(j)r_3r_4} \sum_s K^s_{r_3}(s) X(x-i, y-j, s).$$

Redom, smanjujemo broj ulaznih kanala s S na R_3.
Zatim, konvolucija s $\sigma_{(i)(j)r_3r_4}$ koja ima R_3 ulaznih i R_4 izlaznih kanala.
Na kraju, konvolucija s $K^t_{r_4}$ nas vraća na T izlaznih kanala.

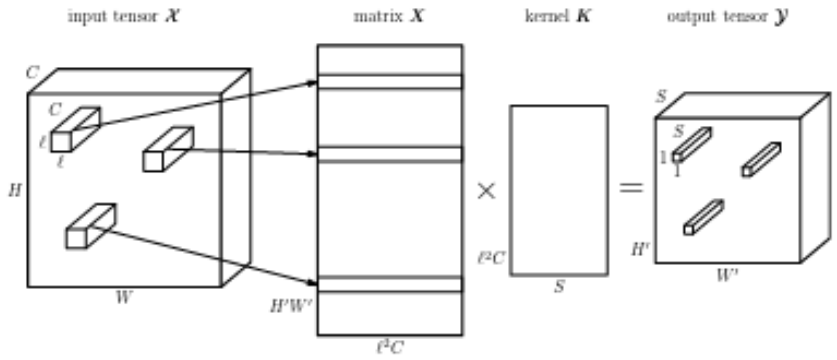


Slika 5: Primjer nakon Tucker-2 dekompozicije

Tensor Train

U ovom [1] slučaju možemo bolje iskoristiti svojstva multilinearnih preslikavanja kako bismo preciznije očuvali rezultate. Konvolucijski tenzor preoblikujemo u matricu. Zatim dimenzije te matrice faktoriziramo tako da dobijemo tenzor vrlo visokog broja osi. Zatim će naš tenzorski vlakić biti dugačak niz malih kockastih tenzora.
Recimo da ulaznu sliku s C kanala preoblikujemo iz $X \times Y \times C$ u $X \times Y \times C_1 \times C_2 \times C_3$. Isto tako, izlazna slika sa S kanala postaje $X \times Y \times S_1 \times S_2 \times S_3$. Vrijede $\prod_i^d c_i = c$ i $\prod_i^d s_i = s$.
Tenzor K koji je originalno $X \times Y \times C \times S$ postaje $G_0[x, y]G_1[c_1, s_1]G_2[c_2, s_2]G_3[c_3, s_3]$. [9]
Konačno

$$V(x, y, s_1, s_2, s_3) = \sum_{i=1}^{l} \sum_{j=1}^{l} \sum_{c_1,c_2,c_3} X(x + i, y + j, c_1, c_2, c_3) \cdot G_0[x, y]G_1[c_1, s_1]G_2[c_2, s_2]G_3[c_3, s_3].$$



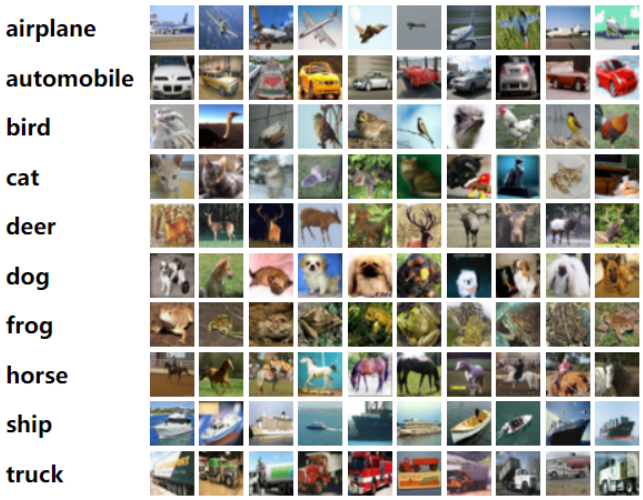
Slika 6: Primjer preoblikovanja prije i nakon primjene tenzorskog vlakića

Rang dekompozicije

Najjednostavnije je uzeti neki proizvoljan broj, recimo trećinu broja kanala. No u radu [8] pokažu kako varijacijskom Bayesovskom aproksimacijom riješiti probleme u kojima se posterior ne može lako ili uopće izračunati. Kada želimo odabrati broj dimenzija za analizu glavnih komponenti tj. matričnu faktorizaciju, imamo nekonveksan optimizacijski problem. Koristili smo gotovu implementaciju gdje pomoću generalizirane (težinske) dekompozicije singularnih vrijednosti, one u kojoj postoje uvjeti na lijeve i desne singularne vektore, egzaktno dobijemo globalno optimalno rješenje.

Podaci

Koristili smo [4] CIFAR dataset koji se sastoji od slika u boji veličine 32×32.
Koristili smo [2] ResNet arhitekturu od 18 slojeva.



Slika 7: CIFAR set podataka

Implementacija

Implementacija je rađena u *Pythonu*. Korištene su gotove tenzorske dekompozicije knjižnice *TensorLy*. Za umjetne neuronske mreže korišten je *PyTorch*, a za ostale operacije linearne algebre *NumPy*.

Rezultati

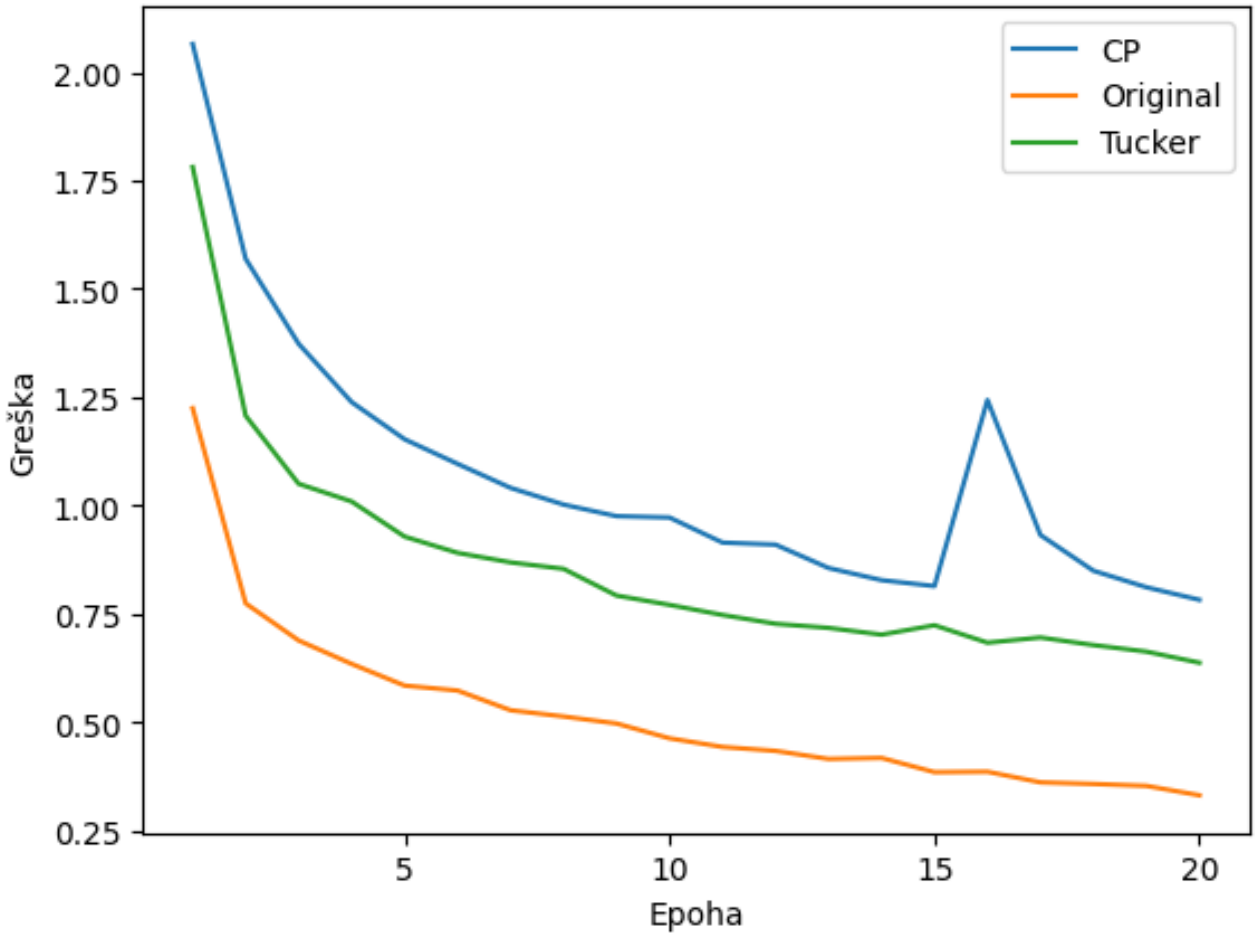
Prošli bismo kroz unaprijed trenirani ResNet i zatim ažurirali težine.

	Original	CP	Tucker	TT
Broj parametara	11689512	982985 (8.4091%)	1118478 (9.5682%)	864960 (7.3995%)
Vrijeme	499.6475s	302.5050s	287.0560s	
Preciznost	84.21%	69.08%	75.07%	

Tablica 1: Podaci o memoriji, vremenu i preciznosti

Epoha	Original	CP	Tucker
1	1.2248	2.0655	1.7815
2	0.7742	1.5696	1.2074
3	0.6886	1.3734	1.0502
4	0.6345	1.2388	1.0094
5	0.5844	1.1530	0.9283
6	0.5730	1.0962	0.8908
7	0.5275	1.0408	0.8688
8	0.5131	1.0022	0.8539
9	0.4969	0.9755	0.7922
10	0.4627	0.9724	0.7707
11	0.4428	0.9145	0.7474
12	0.4337	0.9098	0.7271
13	0.4151	0.8559	0.7176
14	0.4174	0.8278	0.7020
15	0.3846	0.8141	0.7237
16	0.3856	1.2437	0.6834
17	0.3610	0.9320	0.6954
18	0.3574	0.8495	0.6779
19	0.3529	0.8114	0.6628
20	0.3312	0.7824	0.6372

Tablica 2: Podaci o konvergenciji - greška po epohama



Slika 8: Graf greške tijekom treniranja

Bibliografija

[1] Timur Garipov, Dmitry Podoprikhin, Alexander Novikov i Dmitry Vetrov. *Ultimate tensorization: compressing convolutional and FC layers alike*. 2016.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren i Jian Sun. *Deep Residual Learning for Image Recognition*. 2015.

[3] Yong-Deok Kim, Eunhyeok Park, Sungjoo Yoo, Taelim Choi, Lu Yang i Dongjun Shin. *Compression of Deep Convolutional Neural Networks for Fast and Low Power Mobile Applications*. 2016.

[4] Alex Krizhevsky. *Learning Multiple Layers of Features from Tiny Images*. 2009.

[5] Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan Oseledets i Victor Lempitsky. *Speeding-up Convolutional Neural Networks Using Fine-tuned CP-Decomposition*. 2015.

[6] Yann LeCun i Yoshua Bengio. *Convolutional Networks for Images, Speech, and Time Series*. The Handbook of Brain Theory and Neural Networks. Cambridge, MA, USA: MIT Press, 1998., str. 255–258.

[7] Yann Lecun, Leon Bottou, Y. Bengio i Patrick Haffner. *Gradient-Based Learning Applied to Document Recognition*. Proceedings of the IEEE 86 (prosinac 1998.), str. 2278–2324.

[8] Shinichi Nakajima, Masashi Sugiyama i S. Babacan. *Global Solution of Fully-Observed Variational Bayesian Matrix Factorization is Column-Wise Independent*. Advances in Neural Information Processing Systems. Ur. J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira i K.Q. Weinberger. Sv. 24. Curran Associates, Inc., 2011.

[9] I. V. Oseledets. *Tensor-Train Decomposition*. SIAM Journal on Scientific Computing 33.5 (2011.), str. 2295–2317.