

KNN Projekt Report

Kompresia obrazu / videa

Sebastián Chupáč(xchupa03)
Denis Pojezdal(xpojez00)
Ivan Rachler(xrachl00)
tím: xchupa03

Marec 2024

GIT : <https://github.com/Pojezdal/KNNProject>

1 Problematika

Kompresia je proces, ktorý transformuje vstupný tok dát na výstupný za účelom jeho zmenšenia. To je možné dosiahnuť znížením redundancie dát, bud za pomocí štatistických vlastností dát (pravdepodobnosti symbolov), faktom, že konzumentom dát je človek, ktorý nie je schopný postrehnúť zmenu v niektorých aspektoch prijímaných dát, alebo s využitím kontextu komprimovaného symbolu (okolné symboly).

Dva základné typy kompresie dát sú bezstratová, pri ktorej sú po dekompresii rekonštruované dátá identické s pôvodnými, a stratová, kde sú rekonštruované dátá len aproximáciou pôvodných.

Štatistické metódy založené na teórii informácie a entropie sú väčšinou bezstratové, a ich hlavným princípom je mapovanie symbolov, ktoré sa v dátach vyskytujú často, na kratšie kódové slová. K správnemu fungovaniu potrebujú poznať pravdepodobnostné rozloženie dát vstupného toku. Medzi tieto metódy patrí napríklad Huffmanovo alebo aritmetické kódovanie.

Pre tieto metódy vieme vypočítať ideálnu dĺžku slova s využitím Shannonovho teorému ako $-\log_2(P(A))$, kde $P(A)$ je pravdepodobnosť výskytu symbolu A, ktorú ide dosiahnuť entropickými kodérmi (napríklad aritmetické kodéry).

Stratové metódy, ktoré odstraňujú určitú časť informácie, sa využívajú prevažne pre kompresiu zvuku a videa, kde je možné zohľadniť schopnosti vnímania človeka. Vzniká distortion-rate problém, čiže pomer medzi mierou skreslenia výsledných dát a mierou redukcie, ktorý dáva priestor pre rôzne, často špecifické metódy kompresie.

Jednou z najpoužívanejších a najjednoduchších metód je kvantizácia, ktorá transformuje dátá z jednej domény (často spojitej) do inej, menšej (často diskrétnej) domény. Táto transformácia je nevratná, keďže mapuje viac hodnôt zo vstupnej domény na jednu vo výstupnej a je zdrojom stratovosti.

2 State-of-the-art

Kompresia obrazu za využitia neurónových sietí sa začala objavovať v 90. rokoch minulého storočia za využitia Multi-Layer Perceptronov. Tento prístup kombinoval priestorové transformácie, binárne kódovanie a kvantizáciu. Neskôr prišiel pokrok spolu s rozvojom konvolučných neurónových sietí. Tie umožňujú rozpoznať korelácie medzi susednými pixelmi na obrázku. Prvé implementácie kompresie obrazu pomocou konvolučných neurónových sietí sú z roku 2016. O rok neskôr sa objavili implementácie využívajúce Generative Adversarial Networks. Ich hlavnou výhodou oproti CNN je adversarial loss, ktorá pomáha zlepšiť kvalitu výsledného obrázku. [6]

Najmodernejšie prístupy využívajú napríklad variačné autoenkodéry ako napríklad Ballé et al. [1], modely Gaussovských zmesí - Cheng et al. [3], transformery a CNN [4], či fully-transformer based modely [5].

3 Metriky na učenie a vyhodnotenie

BPP (Bits per pixel)

Táto hodnota vyjadruje silu kompresie. Jedná sa o počet bitov vo výstupe kompresoru na 1 pixel vstupného obrázku.

Pri trénovaní sa táto hodnota pripočítava k loss funkcií, ktorá udáva podobnosť, a nastavenie pomeru medzi týmto dvoma časťami ovplyvňuje výslednú silu kompresie po trénovaní.

PSNR (Peak signal-to-noise ratio)

PSNR je miera, ktorá udáva množstvo šumu oproti pôvodnému signálu. V tomto prípade, signálom je pôvodný obrázok a šum je chyba v rekonštrukcii po dekompresii. Na jej výpočet sa používa MSE (Mean squared error):

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{\text{MAX}^2}{\text{MSE}} \right)$$

$$\text{MSE} = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (O(i,j) - R(i,j))^2$$

Kde:

- MAX je maximálna možná hodnota pixelu v obrázku
- O(i,j) je hodnota pixelu (i,j) v pôvodnom obrázku
- R(i,j) je hodnota pixelu (i,j) v rekonštruovanom obrázku
- m a n sú rozmery obrázku

PSNR bolo využité na porovnanie výsledkov medzi metódami/modelmi.

MS-SSIM (Multi-scale structural similarity index metric)

Táto metrika zohľadňuje charakteristiky obrázkov na viacerých úrovniach.

Obrázky sa dekomponujú pomocou vhodných transformácií a počíta sa SSIM na každej úrovni. SSIM obsahuje 3 rôzne podobnosti: podobnosť jasu, kontrastu a štruktúry. Hodnoty SSIM z každej úrovne sa spájajú váženým priemerovaním. Výsledkom má byť vyhodnotenie, ktoré sa viac približuje ľudskému posudku.

MS-SSIM bolo použité ako jedna z loss funkcií.

LPIPS (Learned perceptual image patch similarity)

LPIPS je neurónová sieť, ktorá slúži ako loss funkcia. Je natrénovaná na datasete, v ktorom ľudia anotovali dvojice rekonštruovaných obrázkov podľa toho, ktorý z nich sa viac podobal pôvodnému. Použitie tejto loss funkcie má docieľiť, že rekonštruované obrázky budú na človeka pôsobiť viac kvalitne než keby bola použitá jedna z bežných loss funkcií.

Vo väčšine experimentov bola využitá kombinácia LPIPS a MS-SSIM ako loss funkcia.

4 Datasetsy

Trénovacie/validačné datasetsy:

- STL-10¹ - 100 000 96x96 RGB obrázkov, 80 000 trénovanie, 20 000 validácia

Vizuálne porovnanie výsledkov:

- Kodak² - 24 768x512 alebo 512x768 RGB obrázkov

¹<https://cs.stanford.edu/~acoates/stl10/>

²<https://www.kaggle.com/datasets/sherylmehta/kodak-dataset/data?select=kodim23.png>

5 Baseline riešenie

Ako základné riešenie využívame architektúry z modulu CompressAI[2], konkrétnie model `bmshj2018_factorized5`.

Tento model má základ ako väčšina autoenkovodér architektúr. Vstupný obrázok x je najprv redukovaný niekoľkými konvolučnými vrstvami do menšieho, latent priestoru y za účelom prvotnej kompresie a táto reprezentácia je následne kvantizovaná pomocou zaokrúhľovania do diskrétnej domény \hat{y} .

Pri trénovaní je z dôvodu potreby výpočtu gradientov klasická aproximácia nahradená pripočítaním náhodného šumu z intervalu $[-0,5; 0,5]$ a z tejto reprezentácie je počítaná ďalšia informácia, a to pravdepodobnosti jednotlivých symbolov za pomocí ďalšej súrady neurónových vrstiev (entropy bottleneck).

Táto naučená pravdepodobnosť má viacero využití. Používa sa ako dodatočná informácia pre aritmetický kodér, a zároveň sa z nej počíta zložka loss funkcie ako $-\log_2(\text{likelihood})$, ktorá zaručuje, že sa siet snaží vytvárať reprezentáciu, ktorá má čo najlepšie vlastnosti pre entropické kódovanie.

Zároveň sa siet snaží naučiť kvantily, typicky medián a hodnoty, ktoré ohraničujú väčšinu z hodnôt generovaných naučeným rozložením pravdepodobnosti. Tie sú optimalizované separatne za použitia auxiliary loss funkcie počítajúcej rozdiel od požadovaných kvantilov.

Kvantizovaná reprezentácia je s využitím naučených pravdepodobností a kvantilov zakódovaná nejakou verziou aritmetického kodéru do finálnej binárnej reprezentácie.

Dekompresia je iba inverziou celého postupu.

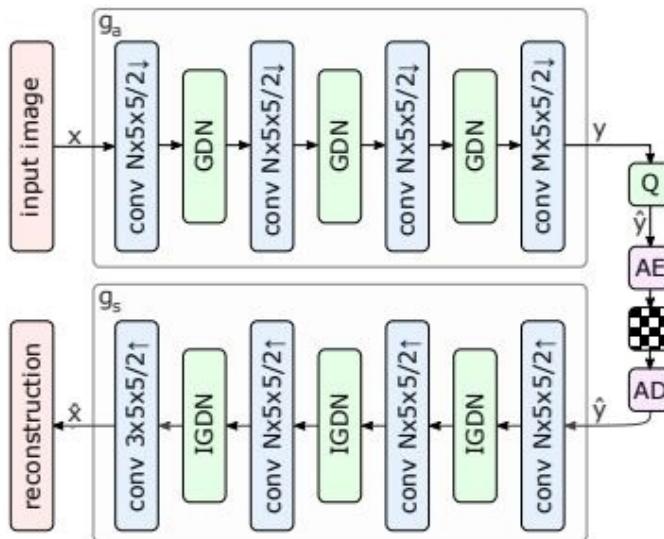


Figure 1: bmshj2018_factorized architektúra. Prebraté z: [1]

6 Prostredie

Projekt implementujeme v Pythone, za využitia knižnice Pytorch. Baseline architektúry, loss funkcie a ďalšie komponenty získame importovaním modulu CompressAI³. Trénovanie a experimenty sú realizované na platforme Google Colab, menej náročný vývoj a ladenie na vlastných zariadeniach.

7 Implementácia

S implementáciou sme začali veľmi naivne, pomocou konvolučných a plne prepojených vrstiev, pričom sme sa snažili obrázky zmenšiť na 20 – 70% pôvodnej veľkosti. Implementovaná bola aj augmentácia obrázkov, ktorá ale neskôr už nebola potrebná. Potom sme skúsili

³<https://github.com/InterDigitalInc/CompressAI>

implementovať vlastný variačný autoenkóder, no bez použitia kvantizácie a aritmetického kódovania sme nedosiahli žiadne uspokojivé výsledky.

Rozhodli sme sa pre architektúru z modulu CompressAI⁴, `bmshj2018_factorized5`, ktorý je založený na variačných autoenkodéroch. Vzhľadom na zložitosť implementácie tejto architektúry sme ju neimplementovali, iba importovali zo spomenutého modulu. Vytvorili sme triedu pre tento model, kde voláme jednotlivé vrstvy a komponenty z CompressAI, a je možné ich tu meniť. Vykonané experimenty sú popísané neskôr, no zahrňovali zmienu loss funkcie, počtu konvolúcii, kanálov, a iných parametrov. Na výstup modelu sme pridali sigmoid funkciu ktorá v pôvodnom modeli nie je, aby sa obrázky správne vykreslovali a aby výstupy mohli byt spracované MS-SSIM funkciou.

8 Experimenty

Podľa CompressAI⁴ boli modely trénované aspoň 1 týždeň na obrázkoch 256x256, 4-5M krokov s loss funkciami MSE alebo MS-SSIM, aby dosiahli state-of-the-art výsledky. My sme sa rozhodli upraviť model a trénovanie rôznymi spôsobmi. Ako loss funkciu sme používali vyváženú kombináciu MS-SSIM a LPIPS. Naše výsledky budeme porovnávať s predtrénovanou variantou modelu trénovanou na MS-SSIM loss funkciu a s jpeg kompresiou.

Naše trénovanie bežalo väčšinou 50-70 epoch s veľkosťou batch 64 a 1250 iteráciami na epochu. Použili sme optimizátor Adam, multiplikatívny scheduler a počiatočný learning rate 0,001 alebo 0,0001. V modeli `bmshj2018_factorized` sme upravovali počet konvolučných vrstiev a počet ich kanálov, dimenzionalitu skrytej reprezentácie obrázku a pridali sigmoid aktiváciu na výstup dekodéru. V loss funkciu sme experimentovali s pomerom metrík podobnosti medzi sebou a pomerom medzi spomenutými metrikami a počtom bitov na pixel. Zopár pokusov sa vykonal aj s nastavením veľkosti kernelu vo výpočte MS-SSIM na bežne používanú hodnotu (11), čo vyžadovalo zväčšenie obrázkov počas trénovania.

Zniženie počtu kanálov a dimenzionality skrytej reprezentácie bolo hlavné pre prispôsobenie menšiemu datasetu a zrýchleniu trénovania. Okrem toho, cielmi týchto experimentov bolo dosiahnuť rozličné hodnoty BPP pre porovnanie kvality rekonštruovaných obrázkov na rôznych úrovniach kompresie a pokúsiť sa zachovať kvalitu. Zvyšovaním váhy pre metriky podobnosti v pomere ku BPP sa model trénoval robíť slabšiu kompresiu. Zmena počtu konvolučných vrstiev zmenila rozlíšenie obrázkov pred vstupom do entropického kodéru, čo sa prejavilo na rýchlosťi trénovania aj hodnote BPP.

Image Size	Konvolúcie	M, N	α	β	γ	PSNR	BPP
96x96	5	64, 96	4	1	1	22,8	0,24
96x96	5	96, 96	15	1	1	22,70	0,35
96x96	4	64, 96	10	2	1	25,57	0,77
96x96	4	64, 96	50	2	1	26,19	1,39
96x96	4	64, 96	100	2	1	26,20	1,59
96x96	4	64, 96	25	8	1	27,00	1,46
96x96	4	64, 96	100	1	3	28,85	1,57
208x208	4	64, 96	25	1	1	25,54	0,87
208x208	4	64, 96	500	1	1	26,55	2,16
96x96	3	32, 48	4	2	1	25,57	0,56
96x96	4	96, 96	10	2	1	25,02	0,78

Table 1: Prehľad experimentov a ich výsledkov. α je váha metrík podobnosti oproti bpp, β je váha pre LPIPS a γ je váha pre MS-SSIM v spojenej metrike podobnosti.

⁴<https://interdigitalinc.github.io/CompressAI/zoo.html#training>

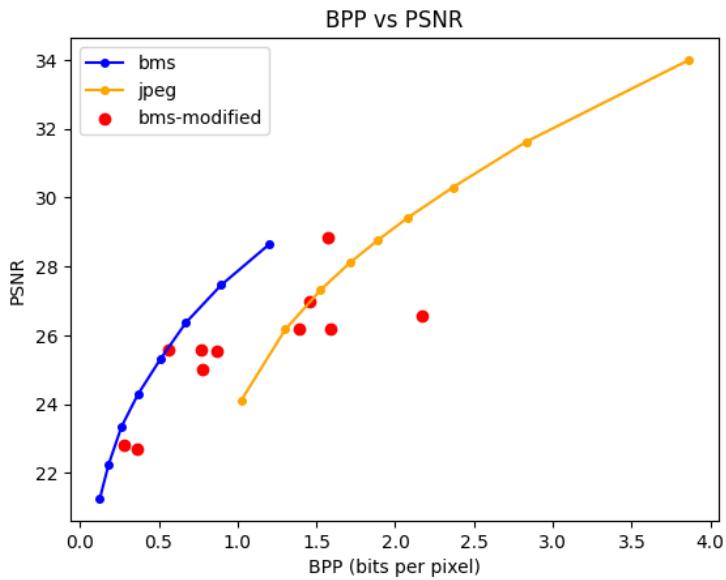


Figure 2: Vyhodnotenie našich zmien v porovnaní s baseline a jpeg.



Figure 3: Porovnanie originálnych obrázku z STL datasetu (hore) s rekonštruovanými obrázkom (dole) pomocou modelu s $\alpha = 100, \beta = 1, \gamma = 3, BPP = 1.57, PSNR = 28, 85$.



Figure 4: Porovnanie originálneho obrázku z KODAK datasetu (hore) s rekonštruovaným obrázkom (dole) pomocou modelu s $\alpha = 100$, $\beta = 1$, $\gamma = 3$, $BPP = 1.57$, $PSNR = 28, 85$.



Figure 5: Porovnanie originálneho obrázku z KODAK datasetu (hore) s rekonštruovaným obrázkom (dole) pomocou modelu s $64, 96, \alpha = 10, \beta = 2, \gamma = 1, BPP = 0.77, PSNR = 25.57$

9 Záver

Z výsledkov našich experimentov vyplýva, že distorčné metriky ako PSNR niesu veľmi spolahlivé na presné určenie kvality obrázkov. Upravené baseline modely sme z časových dôvodov trénovali len na zlomku iterácií a menších obrázkoch ako pôvodní autori.

Rozdelenie práce

Na projekte sa všetci členovia tímu podielali rovnomerne.

- Sebastián Chupáč - prieskum existujúcich riešení, implementácia, trénovanie a experimenty, vyhodnotenie...
- Denis Pojezdál - prieskum existujúcich riešení, implementácia, trénovanie a experimenty, vyhodnotenie...
- Ivan Rachler - prieskum existujúcich riešení, implementácia, trénovanie a experimenty, vyhodnotenie...

References

- [1] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.
- [2] Jean Bégaint, Fabien Racapé, Simon Feltman, and Akshay Pushparaja. Compressai: a pytorch library and evaluation platform for end-to-end compression research. *arXiv preprint arXiv:2011.03029*, 2020.
- [3] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [4] Jinming Liu, Heming Sun, and Jiro Katto. Learned image compression with mixed transformer-cnn architectures, 2023.
- [5] Natacha Luka, Romain Negrel, and David Picard. Image compression using only attention based neural networks, 2023.
- [6] Gilad David Maayan. State of the art in image compression algorithms based on deep learning. <https://towardsdatascience.com/ai-based-image-compression-the-state-of-the-art-fb5aa6042bfa>, 2024. Accessed: 2024-05-09.