Počítačové videnie - Úloha - Bag of Visual Words

Ing. Viktor Kocur
viktor.kocur@fmph.uniba.sk

DAI FMFI UK

28.4.2020

Úloha

Cieľ

Cieľom tejto úlohy bude vytvoriť klasifikátor na obrázky pomocou metódy bag of visual words.

Dataset

Pracovať budete s datasetom, ktorý obsahuje dve triedy: dog a cat. Stiahnuť si ho môžete spomedzi materiálov na githube `https://github.com/kocurvik/edu/blob/master/PV/materialy/du03.zip`. Tento dataset je podmnožina dogscats datasetu `https://www.kaggle.com/c/dogs-vs-cats/data`.

BoW

Princíp

Bag of words funguje tak, že najprv počas tréningu vytvoríme slovník (codebook, bag) vizuálnych slov. Pre každý obrázok potom vytvoríme normalizovaný histogram, ktorý reprezentuje aký podiel v obrázku majú jednotlivé vizuálne slová. Tento normalizovaný histogram použijeme ako príznakový vektor pre klasifikátor (napr. SVM, kNN).

Vizuálne slová

Lokálne príznaky

Základom nášho slovníka bude extrakcia deskriptorov lokálnych príznakov z obrázkov trénovacej množiny. Dostaneme tak veľa vektorov rovnakej dĺžky.

Slovník

Vektory z predchádzajúceho kroku zklastrujeme pomocou k-means algoritmu. Dostaneme tak k stredov pre klastre. Každý stred bude jedno slovo v slovníku.

Vizuálne slová

Histogramy

Ak máme slovník tak histogram pre obrázok vytvoríme tak, že pre obrázok exrahujeme deskriptory lokálnych príznakov. Pre každý deskriptor nájdeme slovo zo slovníka ktorému zodpovedá, teda stred klastra pre ktorý je najbližší. Dostaneme takto zoznam slov (môžu sa opakovať). Z tohto zoznamu potom spravíme histogram a normalizujeme ho (podelíme súčtom).

Príznakový vektor

Keďže všetky histogramy sú vektory dĺžky k (podľa počtu klastrov/slov v slovníku), tak ich môžeme používať ako príznakové vektory pre všetky obrázky.

Klasifikácia

Klasifikátor

Keďže každému obrázku vieme priradiť príznakový vektor, tak to vieme spraviť pre každý obrázok z trénovacej množiny. Pre tieto príznaky potom natrénujeme klasifikátor (napr. SVM).

Validácia

Tu je samozrejme vhodné nájsť vhodný klasifikátor a jeho nastavenia a overiť jeho presnosť na validačnej množine. Takisto je vhodné overiť, či sme vybrali vhodné k pri generácii slovníka.

Testovanie

Na konci si náš klasifikátor otestujeme na testovacích dátach.

Matlabovské funkcie

Zakázané príkazy

Matlab má pre BoW hotovú implementáciu pomocou príkazov bagOfFeatures a trainImageCategoryClassifier. Keďže úloha by s nimi bola triviálna, tak nieje dovolené ich používať.

ImageDataStore

Keďže budeme potrebovať pracovať s viacerými obrázkami, tak je vhodné využiť Matlabovskú štruktúru ImageDataStore. V nasledujúcom bloku je príklad ako s nimi pracovať. Ak ho použijete, tak budete mať prístup aj k imds.Labels, kde budú triedy pre dané obrázky.

Načítanie obrázkov

```
imds = imageDatastore('dataset/train', ...
'IncludeSubfolders',true, 'LabelSource', 'foldernames');
n = size(imds.Files, 1)
for i = 1:n
    img = imds.readimage(i);
    if size(img, 3)==3
        img = rgb2gray(img);
    end
    % do something with img
end
```

Slovník

SURF

V tejto úlohe budete používať SURF príznaky. Pred tvorbou slovníka tak vytvorte maticu $n \times 64$ v ktorej bude n deskriptorov z obrázkov (n by malo byť viacnásobne väčšie ako počet obrázkov)

kmeans

Na túto maticu zavoláte k-means s nejakým k. Pri úvodnom programovaní skúšajte menšie k a nastavte si prorerty 'MaxIter' na nejaké menšie číslo (defaultne je to 100). k-means zavolajte tak, aby ste dostali na výstupe aj stredy klastrov.

Histogramy

pdist2

Ak máte obrázok a z neho extrahované deskriptory, tak histogram získate v dvoch krokoch. Najskôr pre každý deskriptor nájdete najbližší stred klastra. Na toto sa vám zíde príkaz pdist2 v kobinácii s minimom.

histcount

V druhom kroku potom musíte vygenerovať histogram. Na to sa vám zíde príkaz histcount. Nezabudnite na to, že histogramy musia mať dĺžku k a jednotlivé prepážky musia byť rovnaké pre každý obrázok. Histogram potom normalizujte, tak že ho vydelíte jeho súčtom.

Klasifikátor

Klasifikácia

Na klasifikáciu použijete SVM a kNN klasifikátor. Hyperparametre môžete nájsť ručne, alebo použite property 'OptimizeHyperparameters'. Pri SVM je dôležité mať správne nastavený kernel a jeho škálu.

Vyhodnotenie

Súčasťou odovzdanej úlohy bude aj pdf súbor kde vyhodnotíte presnosť oboch klasifikátorov pre aspoň 5 rôznych hodnôt k (veľkosť slovníka - rozsah 100-5000, podľa toho čo upočíta váš počítač) na validačnej množine. Nakoniec uvediete aj presnosť na testovacej množine pre najlepšie k.

Tipy

Štruktúra

Veľa z podúloh pri tvorbe BoW klasifikátora trvá dlho, ale napr. ak chceme meniť parametre klasifikátora, tak nepotrebujem odznova načítať obrázky a generovať slovník. Oplatí sa preto jednotlivé kroky mať uložené v skriptoch, keďže tie ukladajú medzivýsledky do workspace.

Očakávaná presnosť

Pri SVM by ste mali dosiahnuť aspoň 70% presnosť a pre kNN aspoň 60%.

Rekapitulácia

Rekapitulácia

Na datasete z Githubu natrénujte BoW klasifikátor využívajúci SURF príznaky. Na validačných dátach si overte vhodnú veľkosť slovníka (z aspoň 5 hodnôt) pre klasifikátor využívajúci SVM a separátne aj kNN (pre oba takisto nájdite pomocou validácie vhodné hyperparametre). Pre najlepšiu možnosť pre oba klasifikátory zistite aj presnosť na testovacích dátach. Do pdfka napíšte výsledky z hľadania vhodného k a takisto výsledky na testovacej množine.

Zakázané príkazy

Nezabudnite, že nesmiete používať matlabovské príkazy bagOfFeatures a trainImageCategoryClassifier.

Odovzdávanie

Odovzdávanie

Odovzdajte vami vytvorené .m súbory (ak máte viacero skriptov, tak vytvorte aj jeden skript ktorý sa bude dať spustiť naraz) a pdf súbor s výsledkami. Súbory zazipujte a pošlite na adresu kocurvik@gmail.com s predmetom PV - DU3. Deadline bude oznámený mailom. Úloha je na 10 bodov. Za každý deň meškania -2 body.