Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Automatická anotace obrázků

Plzeň 2017 Kateřina Kratochvílová

Místo této strany bude zadání práce.

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracovala samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 7. června 2017

Kateřina Kratochvílová

Poděkování

Ráda bych poděkovala Ing. Ladislavu Lencovi, Ph.D. za cenné rady, věcné připomínky, trpělivost a ochotu, kterou mi v průběhu zpracování této práce věnoval.

Abstract

The text of the abstract (in English). It contains the English translation of the thesis title and a short description of the thesis.

Abstrakt

Text abstraktu (česky). Obsahuje krátkou anotaci (cca 10 řádek) v češtině. Budete ji potřebovat i při vyplňování údajů o bakalářské práci ve STAGu. Český i anglický abstrakt by měly být na stejné stránce a měly by si obsahem co možná nejvíce odpovídat (samozřejmě není možný doslovný překlad!).

Obsah

1 Úvod

V dnešní době, kdy je svět přesycen obrázky v digitální podobě, není vůbec snadné nalézt obrázek zobrazující požadovaný obsah. Naneštěstí počítače nedokáží vnímat obraz jako lidé, vnímají totiž obrazy jako sérii binárních informací. Přitom počítače a jejich práce s obrazy by se dala využít v mnoha oborech jako je lékařství nebo doprava. Na základě toho vyplouvá na povrch problém jak spravovat digitální obrázky a efektivně mezi nimi vyhledávat. Prostřednictvím klíčových slov přiřazených k obrázkům se dá problém vyhledávání zjednodušit. Přiřazení klíčových slov probíhá pomocí procesu automatické anotace obrázků. Klíčová slova přiřazená k obrázku by měla vyjadřovat jeho obsah (například les, strom). Při reálném použití můžeme ovšem narazit na problém při zadávání abstraktních slov, například šťastná rodina.

Pro automatickou anotaci obrázků se používá strojové učení. Můžeme ji rozdělit na dvě části. V první části získáme klíčové příznaky ve druhé už je samotná anotace, tedy přidělení klíčových slov. Abychom tento postup mohli provést v praxi, musíme nejdřív klasifikátor natrénovat pomocí trénovací množiny. Trénovací množina je množina obrázků, která již má ke každému obrázku přidána metadata s klíčovými slovy připravenými od lidí. Vybrané obrázky v trénovací množině musí být různorodé, aby anotace probíhala správně. Pojem automatická anotace obrázků je jednoduše řečeno proces, při kterém jsou k obrázku automaticky přiřazena metada, která obsahují klíčová slova.

Cílem práce je navrhnout a implementovat software umožňující automatickou anotaci obrázků. Konkrétně se bude zabývat metodou JEC. Práce bude využívat nízkoúrovňové příznaky, konkrétně barvu a texturu. Metodu budeme zkoušet na standardních datech, následně se budeme snažit výsledky vylepšít, a porovnat s další metodou a literaturou.

# 2 JEC Joint Equal Contribution

Tato metoda je založena na hypotéze, že podobné obrázky mají podobná klíčová slova. Pomocí metody hledání nejbližších sousedů (dále jen KNN) je nalezeno K nejpodobnějších obrázků. Přičemž klíčová slova od jednotlivých sousedů jsou posuzována odlišně a to právě na základě toho o kolik se s testovaným obrázkem liší. Metoda je postavena na dvou typech příznaků - barevných a texturových. [1]

2.1 Příznaky

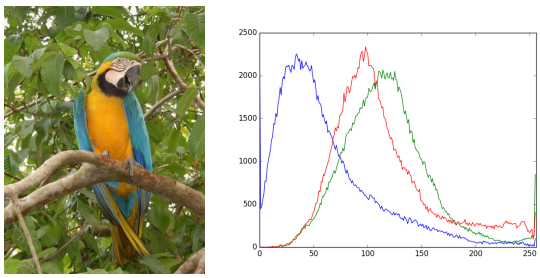
Barva a textura jsou považovány za dva nejdůležitější nízkoúrovňové příznaky pro obrázkovou reprezentaci. Nejběžnější barevné deskriptory jsou barevné histogramy, které jsou často využívány pro porovnávání a indexování obrázků, zejména z důvodu jejich efektivnosti a snadného výpočtu. K vytvoření texturových příznaků se používají Haarovy a Gaborovy wavelety a to především z důvodu, že jsou efektivní při vytváření řídkých a zároveň diskriminačních obrázkových rysů. Je-li žádoucí omezit vliv a předpoklady jednotlivých funkci a maximilizovat množství získaných informací, že využijeme několik jednoduchých a snadných výpočetních funkcí.

2.1.1 Barva

U digitálního obrazu je barva reprezentovaná n-rozměrným vektorem. Jeho velikost a význam jednotlivých složek (tzv. barevných kanálů) zavisí na příslušném barevném prostoru. Počet bitů použitých k uložení buď celého vektoru nebo jeho jednotlivých složek se nazývá barevná hloubka (totožně bitová hloubka). Obvykle se můžeme setkat s hodnotami 8, 12, 14 a 16 bitů na kanál.
V použité metodě jsou získány vlastnosti z obrázků ve třech rozdílných barevných prostorech: RGB, HSV a LAB. RGB (Red, Green, Blue) je nejpoužívanější barevný prostor pro zachycení obrázu nebo jeho zobrazení. Oproti tomu HSV (Hue, Saturation and Value) se snaží zachytit barevný model tak jak ho vnímá lidské oko, ale zároveň se snaží zůstat jednoduchý na výpočet. Hue znamená odstín barvy (měří se jako poloha na standartním barevném kole $0° - 360°$), saturation je systost barvy (množství šedi v poměru k od-

stínu 0% šedá barva - 100% plně sytá barva) a value je hodnota jasu nebo také množství bílého světla (relativní světlost nebo tmavost barvy). Některé kombinace hodnot H, S a V mohou dávat nesmyslné výsledky. RGB je závislý na konkrétním zařízení, nemůže dosáhnout celého rozsahu barev, které vidí lidské oko, zatímco barevný model LAB je shopen obsáhnout celé viditelné spektrum a navíc je nezávislý na zařízení. L (ve zkratce LAB) značí Luminanci (jas dosahuje hodnot 0 - 100, kde 0 je černá a 100 je bílá). Zbylé A a B jsou dvě barvonosné složky, kdy A je ve směru červeno/zeleném a B se pohybuje ve směru modro/žlutém.

Pro RGB, HSV i LAB je použita barevná hloubka 16 bitů na kanál histogramu v jejich příslušném barevném prostoru.



Obrázek 2.1: RGB histogram - zastoupení jednotlivých složek v obrázku

Obrázek 2.2: Barevný prostor RGB a jeho jednotlivé složky v pořadí R, G, B

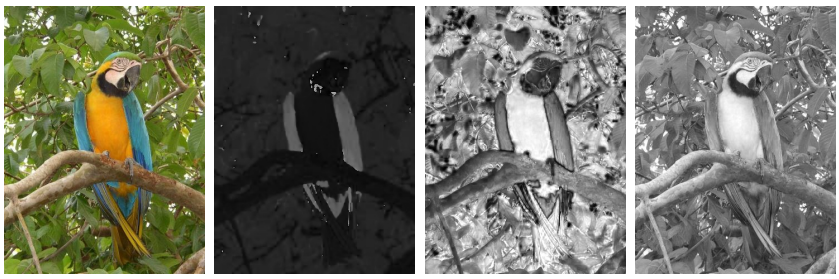
2.1.2 Textura

Jako reprezentace textur a detekci hran budou použity Gabor a Haar wavelety.

Gabor

Gabor filtr je lineární filtr používaný pro analýzu textury, což znamená, že v podstatě analyzuje, zda existuje nějaký specifický frekvenční obsah v

Obrázek 2.3: Barevný prostor LAB a jeho jednotlivé složky v pořadí L, A, B



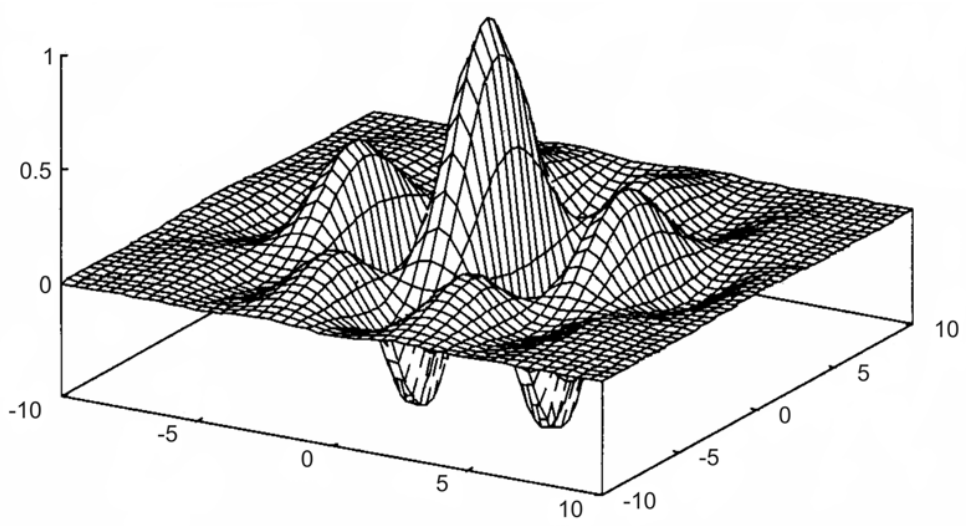
Obrázek 2.4: Barevný prostor HSV a jeho jednotlivé složky v pořadí H, S, V

obraze ve specifických směrech v lokalizované oblasti kolem oblasti analýzy. Frekvence a orientace reprezentující Gabor filtr je podobná lidskému vnímání a jsou zvláště vhodné pro reprezentaci textury a detekci hran. V prostoru je 2D Gaborův filtr funkcí gausova jádra modulovaného sinusouvou rovinou vlnou.

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = exp\left(- \frac{x^2 + \gamma^2 y^2}{2\sigma^2} \right) exp\left(i\left(2\pi \frac{x}{\lambda} + \psi \right) \right) \tag{2.1}$$

Gabor filtry jsou aplikovány na obrázky stejnou cestou jako běžné filtry. Základ tvoří maska (přesnější termín je konvoluční jádro), která reprezentuje filtr. Maskou je myšleno pole (obyvykle 2D protože se jedná o 2D obrázky) pixelů ve kterém každý pixel má přiřazenou hodnotu (váhu). Toto pole je přesunuto na každý pixel obrazu a je provedena konvoluční operace. Když je na obrázek aplikován gabor filtr, poskytuje nejvyšší odezvu na hranach a místech, kde se textura mění.

Gabor filtr reaguje na hrany a změny textury. Když se řekne, že filtr odpovídá na konkrétní funkci, myslí se tím že filtr má rozlišovací hodnotu v prostorové poloze této funkce (když se bude zabývat aplikací konolučních jader v prostoru - směru. Stejně platí i pro jinou oblast, jako frequence)



Obrázek 2.5: Gábor vlnka je tvořena kombinací dvou cosinových funkcí, s rozdílnou frekvencí pro každou osu, a následně jsou vynásobeny dvourozměrnou Gaussovou funkcí [2].

U Gabor filtru máme několik parametrů, které ho ovlivňují.

ksize určuje velikost Gabor jádra. Když je ksize (a, b) je získáno jádro velikostí $a \times b$ pixelů. Jako u mnoha jiných konvolučních jader je preferován rozměr čtverce o lichých hranách (jen kvůli jednotnosti). Pří různých ksize se velikost konvolučního jádra mění. To také znamená, že konvoluční jádro je měřítko invariantní, protože zmenšení velikosti jádra je analogické k zmenšení velikosti obrazu.

sigma označuje standartní odchylka Gaussovi funkce použita v gaborově filtru. Tento parametr kontroluje šířku šířku Gausovi obalu použité v gabor jádře.

theta je orientace normálu na paralelní pruhy Gaborovy funkce. Představuje možná jeden z nejdůležitějších parametrů gabor filtru. Theta rozhoduje jakého druhu funkce (na jaký typ funkce filtr reaguje). Například při nulév thetě bude filtr reagovat pouze na vodorovné příznaky. Proto abychom získali vlastnosti v různých úhlech obrazu, rozdělíme interval mezi 0-180 na několik stejných částí a vypočítáme Gaborovo jádro pro každou takto získanou hodnotu theta.

lambda udává vlnovou délku sinusovky ve výše uvedené rovnici.

gamma určuje prostorový poměr stran. Kontroluju elipsicitu gausovy funkce. Když je gamma = 1 je Gauss do kruhu (obalen kruhem)

psi je fázový posun (určuje jestli nám vrátí reálnou nebo imaginární část).

Podle [1] bude každý obrázek filtrován na třech vlnových délkách a čtyřech orientacích. Z každého z dvanácti obrázků bude histogram postaven skrze získané magnitudy. Vzniklé magnitudy zřetězíme a označíme jako příznak Gabor. Druhý příznak zachycuje Gabor faze. Příznak je označen jako GaborQ

Gabor s knihovnou scikit na datech iaprtc12			
Parametry	Přesnost (%)	Úplnost (%)	Počet nenulových slov
lambda 0.25, 0.5, 1.0 sigma 1 theta 0, $\frac{\pi}{4}$, $\frac{\pi}{2}$, $\frac{3}{4}\pi$	9.9	6.8	151
lambda 2, $2\sqrt{2}$, 4 sigma 1 theta 0, $\frac{\pi}{4}$, $\frac{\pi}{2}$, $\frac{3}{4}\pi$	8.5	5.7	143

2.1.3 Haar

Haar wavelet generují konvoluční blok s Haar filtry na třech rozdílných orientacích (horizontální, diagonální a veritkální). Použité na obrázky různých velikostí.

2.1.4 Barevný POEM

Výpočet gradientu a magnitudy

Výpočet gradientu probíhá obdobně jako u nebarevného obrázku. Pro každou ze tří složek jsou získány dvě matice filtrované maskami. Celkem bude 3×2 matic. Na matice se dá pohlížet jako na 2 vektory o 3 složkách. Vektory jsou sloučeny pomocí součtu vektoru do jednoho 3 složkového vektoru. Magnituda je opět velikost vektoru tentokrát, ale v prostoru.

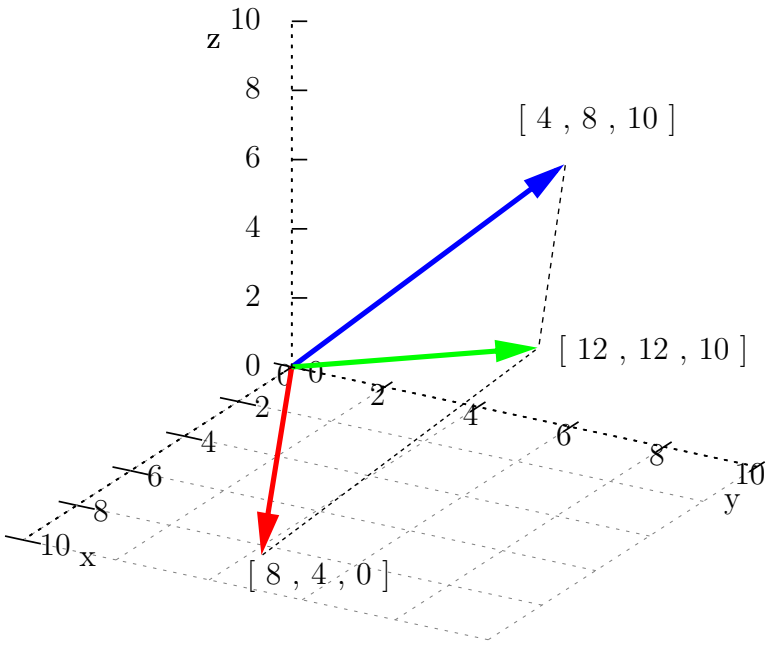
Format vzniklých vektorů

$$u = [blue_x, green_x, red_x] \tag{2.2}$$

$$v = [blue_y, green_y, red_y] \tag{2.3}$$

Pomocí součtu vektorů je získán jeden tříslożkový vektor:

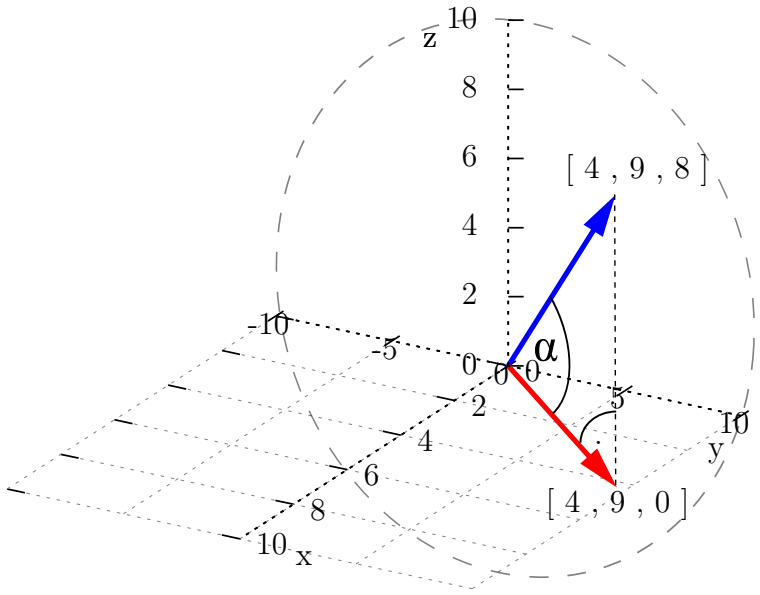
$$\vec{u} + \vec{v} = (u_1 + v_1, u_2 + v_2, u_3 + v_3) \tag{2.4}$$



Obrázek 2.6: Grafické znázornění součtu vektorů. Součet je tvořen z vektorů [4, 8, 10] a [8, 4, 0].

Diskretizace směru gradientu

U vektorů získaných v předchozím kroku je určena velikost úhlu mezi vektorem a ekvivalentní vektorem s vynulovanou složkou z. Následně je spočítáno do které části kružnice vektor směřuje. Pro znaménkovou reprezentaci je celkový rozsah $0 - \pi$, pro neznaménkovou reprezentaci $0 - 2\pi$.
Při neznaménkové reprezentaci a počtu směrů $d = 3$, jsou následující intervaly $\left(0 - \frac{\pi}{3}\right)$, $\left(\frac{\pi}{3} - \frac{2\pi}{3}\right)$ a $\left(\frac{2\pi}{3} - \pi\right)$.
Pro výpočet diskretizace směru při neznaménkové reprezentaci je y složka rozdělena na kladnou a zápornou část. To hraje velkou roli, pokud je y složka vektoru záporná. V tom případě je nutné nebrat úhel α, ale jeho doplněk $(180 - \alpha)$.



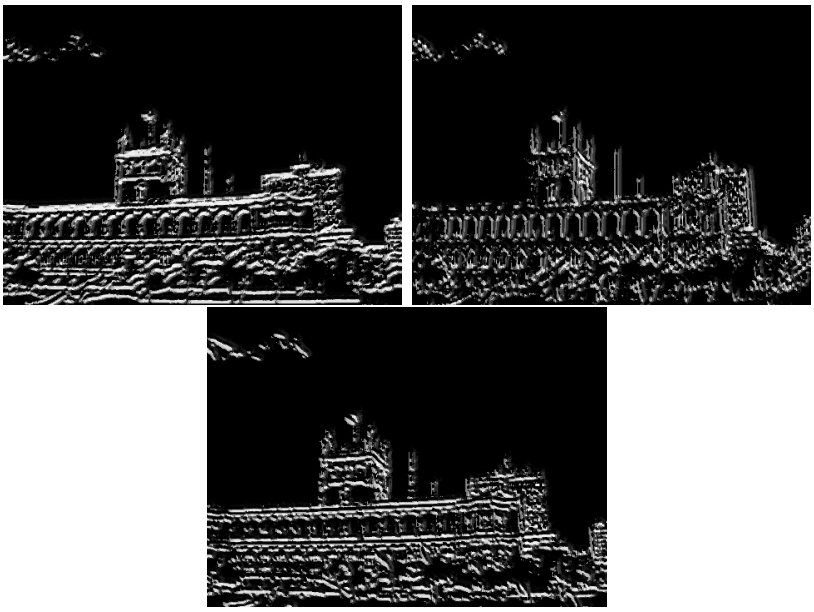
Obrázek 2.7: Grafické znázorněnní součtu vektorů. Součet je tvořen z vektorů [4, 8, 10] a [8 4, 0].

Výpočet lokálního histogramu

U každého směru se vezmou jednotlivé pixely s jejich okolím a zprůměrují se jejich hodnoty. Toto okolí se nazývá cell.

Zakódování příznaků pomocí LBP

LBP operátor je aplikován na okolí každého pixelu o velikost 3×3. Oproti tomu POEM je možné aplikovat na větší okolí. Toto okolí se nazývá block, zpravidla se jedná o kruhové okolí s poloměrem L/2 (L představuje velikost blocku). Pro stanovení intenzit okolních hodnot je možné použít bilineární interpolaci. Pro zvýšení stability v téměř konstantní oblasti lze k centrálnímu pixelu přičítat malou konstantu τ.



Obrázek 2.8: Obrázky po aplikaci LBP s použitím τ. Každý obrázek představuje jeden směr.

Konstrukce globálního histogramu

Obrázky získané z LBP jsou rozdeleny pravidelnou čtvercovou mřížkou. Pro každou vzniklou oblast je vypočten lokální histogram. Vzniklé histogramy jsou zřetězeny. Díky tomu jsou získány tři histogramy pro každý směr jeden, které jsou opět zřetězeny.
Rozdělení obrázků a určování lokálních histogramů se dělá za účelem zachování informace o prostorovém rozložení jednotlivých příznaků.

Jelikož histogram, který bude vytvořen je velmi dlouhý, je možné ho zkrátit vybíráním pouze tz. uniformních vzorů. Některé binární vzory se totiž na běžných obrázcích vyskytují častěji a to až, dle experimentů z 90 %. Jsou to právě výše zmíněné uniformní vzory. Uniformní vzory jsou hodnoty čísla (čísla z pohledu binární reprezentace), kdy dochází k maximálně dvěma přechodům z 0 na 1 a nebo opačně. Například 00011110 je uniformní, oproti tomu 01101111 není. Těchto vzorů je 58, všechny ostatní vzory jsou reprezentovány jediným vzorem. Takto je délka jednoho lokálního histogramu zredukována z 256 na 59 binů.

2.2 Vzdálenosti

vzdálenost vs podobnost

K určení příslušné vzdálenosti se můžeme setkat se čtyřmi měřítky vzdá-

lenosti pro histogramy a rozdělení Kullaback-Leibler divergence KL - divergence, χ^2 statistika, L1 - vzdálenost a L2 - vzdálenost. Na RGB a HSV je nejlépší použít L1 zatímco pro LAB je nejvhodnější KL - divergence.

Problém s KL - divergencí nastává pouze tehdy, když se histogramy nebudou shodovat v nulách. Jeden předpoklad pro fungování tohoto vzorce je totiž že když je $Q(i) = 0$ tak zároveň musí být i $P(i) = 0$.

Kullaback-Leiber divergence:

$$D_{KL}(P||Q) = \sum_i P(i) \log_e \left(\frac{P(i)}{Q(i)} \right) \tag{2.5}$$

L1 (jinak označováno jako Manhattan):

$$L_1 = \sum_{i=1}^{N} |x_i - y_i| \tag{2.6}$$

L2 (jinak označováno jako Euklidovská vzdálenost)

$$L_2 = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2} \tag{2.7}$$

2.3 Kombinace vzdáleností

Nejrozumnějším přístupem ke zkombinování vzdáleností od různých desktiptorů je aby jednotlivé vzdálenosti přispívali rovnoceně. Z tohoto důvodu je potřeba vzdálenosti přeškálovat na jednotné měřítko.
Označme si I_i jako i-tý obrázek a řekněme, že máme N jeho příznaků $f_i^1, ..., f_i^N$. Nadefinujme si $d_{(i,j)}^k$ jako vzdálenost mezi příznaky f_i^k a f_j^k. Chtěli bychom zkombinovat všechny vzdálenosti příznaků mezi obrázky I_i a I_j tedy $d_{(i,j)}^k$, $k = 1, ..., N$. Vzdálenosti nám ale v praxi nevyjdou tak aby měli stejný poměr na výsledku, proto předtím než vzdálenosti zkombinujeme musíme je normalizovat do jednotné formy. Získáme maximální a minimální hodnotu pro každý příznak a na základě toho hodnotu přeškálujeme na interval od 0 do 1. Jestliže označíme přeškálovánou vzdálenost jako $\tilde{d}_{(i,j)}^k$ následně můžeme označit kompletní vzdálenost mezi obrázky I_i a I_j jako (2.8) Joint Equal Contribution (JEC).

$$JEC = \sum_{k=1}^{N} \frac{\tilde{d}_{(i,j)}^k}{N} \tag{2.8}$$

2.4 Přenesení klíčových slov

Pro přenesení klíčových slov používáme metodu, kdy přeneseme n klíčových slov k dotazovanému obrázku $\tilde{I}$ od K nejbližších sousedů z trénovací sady. Mějme $I_i, i = 1, ..., K$,těchto K nejbližší sousedů seřadíme podle vzrůstající vzdálenosti (tzn. že I_1 je nejvíce podobný obrázek). Počet klíčových slov k danému I_i je označen jako $|I_i|$. Dále jsou popsány jednotlivé kroky alogoritmu na přenesení klíčových slov.

1. Seřadíme klíčová slova z I_1 podle jejich frekvence výskytu v trénovací sadě.

2. Ze všech $|I_1|$ klíčových slov z I_1 přeneseme n nejvýše umístěná klíčová slova do dotazovaného $\tilde{I}$. Když $|I_1| < n$ pokračujte na krok 3.

3. Seřadíme klíčová slova sousedů od I_2 do I_K podle dvou faktorů

   (a) výskytu v trénovací sadě s klíčovými slovy přenesených v kroku 2

   (b) místní frekvence (tj. jak často se vyskytují jako klíčová slova u obrázků I_2 až I_K). Vybereme nejvíce vyskytující $n-|I_1|$ klíčových slov převedených do $\tilde{I}$.

Tento algoritmus pro přenos klíčových slov je poněkud odlišný od algoritmů, které se běžně používají. Jeden z běžně užívaných funguje na principu, že klíčová slova jsou vybrána od všech sousedů (se všemi sousedy je zacházeno stejně bez ohledu na to jak jsou danému obrázku podobní), jiný užívaný algoritmus k sousedům přistupuje váženě (každý soused má jinou váhu)a to na základě jejich vzdálenosti od testovaného obrázku. Při testování se ovšem ukázalo, že tyto přímé přístupy přináší horší výsledky v porovnání s použitým dvoufaktorovým algoritmem pro přenos klíčových slov.
V souhrnu použitá metoda je složenina ze dvou složenin a to obrázkové vzdálenosti (JEC) a výše popsaným algoritmem na přenášení klíčových slov.

3 Testovací databáze

Pro natrénování a následné testování byla použita data z databází ČTK, ESP a IAPRC. Kolekce obrázků na natrénování musí být pečlivě vybrána aby zahrnovala co možná největši okruh z různých témat.

3.1 iaprtc12

Data iaprtc12 obsahují 20 000 obrázků ve formátu jpg s celkovým počtem 291 klíčových slov. Ke každému obrázku jsou přiložena metadata ve formátu XML, která obsahují informace o obrázku v různých jazycích. Kromě angličtiny je tam i například španělština nebo němčina. V metadatatech ovšem nenajdeme klíčová slova tak jak bychom si je představovali, ale v různých tagách nalezneme například titulek obrázku, který může vypadat například The Plaza de Armas, a v tagu description je například a woman and a child are walking over the square. Klíčová slova musela být z těchto dat získána Spolu s databází jsme zíkali klíčová slova byla extrahována
K jednomu obrázku je v průměru přiřazeno 5.7 klíčových slov. Pro trénovaní bylo použito 17664 obrázků, na následné testování jich bylo použito 1960.

Obrázek 3.1: Ukázka obrázku s klíčovými slovy: front lake man mountain rock sky summit

3.2 ČTK

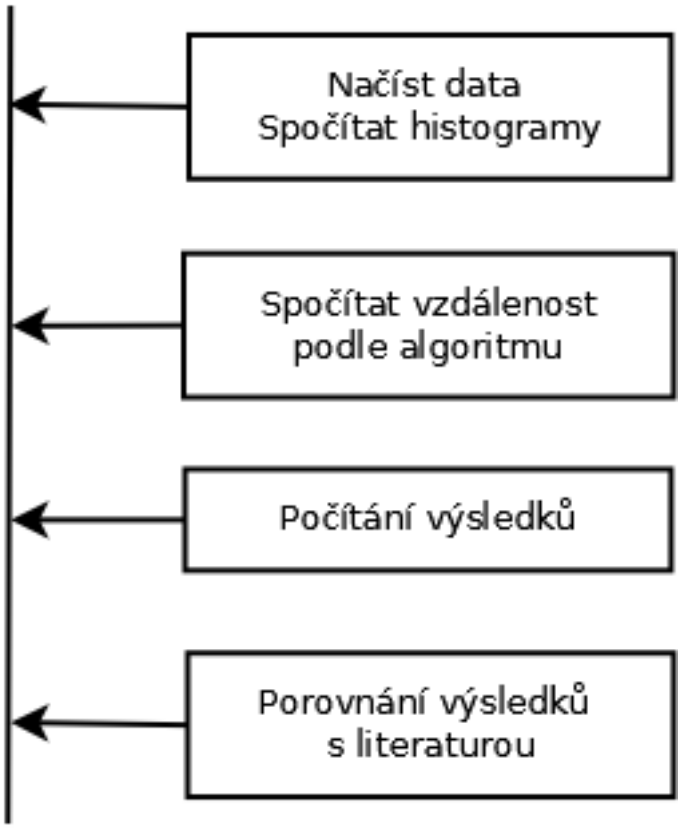
Data od ČTK obsahují 3 383 obrázků. Ke každému obrázku jsou přidána metadata ve formátu XML, která obsahují klíčová slova a další informace o obrázku.

3.3 ESP

Data od ESP obsahují 67 796 obrázků ve formátu jpg. Ke každému obrázku je přiřazen soubor ve formátu desc, který obsahuje anglické anotace.

4 Návrh systému

Systém byl navržen jako modulový a to z důvodu snadné obměny některé z částí, což je výhodné zejména pokud bychom potřebovali například spočítat vzdálenosti vektorů podle jiného algoritmu.



Obrázek 4.1: Návrh systému

5 Použité programové prostředky

Program byl navržen na operační systému Linux. Jako programovací jazyk byl zvolen Python a to z důvodu jeho jednoduchého použití, což je na prototyp, jako je tento velice výhodné na časovou náročnost. Program využívá knihovnu OpenCV 3.1.

5.1 OpenCV

OpenCV (Open source computer vision) je knihovna vydávána pod licencí BSD a je volně k dispozici jak pro akademické účely, tak pro komerční použití. Je vhodná pro použití v C++, C, Python a Javě. Podporuje operační systémy Windows, Linux, Mac OS, iOS a Android.
Knihovna byla navrhnuta pro výpočetní efektivitu v oblasti počítačového vidění a zpracování obrazu se zaměřením na zpracování obrazu v reálném čase. Z důvodu optimalizace byla napsána v C/C++.
Knihovnu OpenCV je možné stáhnout na adrese: http://opencv.org/

5.2 Scikit

Scikit-image je sbírka algoritmů pro zpracování obrazu. Je k dispozici zdarma a bez omezení. [**?**] http://scikit-image.org/docs/dev/install.html

6 Vyhodnocení výsledků

Zpracování výsledků probíhá jako porovnání anotací přidělených člověkem s anotacemi přidělenymi klasifikátorem. Označme si w_{auto} jako počet obrázků, kterým bylo dané slovo přiřazeno klasifikátorem, w_{human} počet obrázků, kterým bylo dané slovo přiřazeno člověkem a $w_{corrrectly}$ jako počet obrázků, kterým bylo slovo přiřazené správně. U klasifikátorů se počítá precision (přesnost) a recall (úplnost) pro každé slovo v testovací sadě.
Recall (6.1) je počet obrázků správně anotovaných s daným slovem děleno počtem obrázků, kterým bylo toto slovo přiděleno v anotaci člověkem. Precision (6.2) je počet správně anotovaných obrázků s tímto slovem děleno celkovým počtem anotovaných obrázků s tímto slovem (správně nebo ne). [6]

$$Rec = \frac{w_c}{w_h} \tag{6.1}$$

$$Prec = \frac{w_c}{w_a} \tag{6.2}$$

6.1 Srovnání výsledků

	IAPRTC12			ESP			ČTK		
Metoda	$P_{\%}$	$U_{\%}$	N	$P_{\%}$	$U_{\%}$	N	$P_{\%}$	$U_{\%}$	N
RGB	14.1	9	167	0	0	0	0	0	0
LAB	12.7	7.5	148	0	0	0	0	0	0
HSV	16.7	10.9	181	0	0	0	0	0	0
RGB, LAB, HSV	17.4	11.1	178	0	0	0	0	0	0
POEM	21.5	12.8	189	0	0	0	0	0	0
RGB, LAB, HSV, POEM	21.8	13.8	187	0	0	0	0	0	0
Gabor	9.9	6.8	151	0	0	0	0	0	0
Barevný POEM	21	12.4	184	0	0	0	0	0	0
JEC	0	0	0	0	0	0	0	0	0

Tabulka 6.1: Výsledky získané v rámci práce.

	IAPRTC12			ESP			ČTK		
Metoda	$P_{\%}$	$U_{\%}$	N	$P_{\%}$	$U_{\%}$	N	$P_{\%}$	$U_{\%}$	N
RGB	24	24	233	20	22	212	0	0	0
LAB	24	25	232	20	22	221	0	0	0
HSV	20	20	215	18	20	212	0	0	0
Haar	20	11	176	21	18	205	0	0	0
HaarQ	19	16	189	18	19	207	0	0	0
Gabor	15	15	183	15	16	186	0	0	0
GaborQ	8	9	137	14	15	193	0	0	0
JEC	28	29	250	22	25	224	0	0	0

Tabulka 6.2: Výsledky z literatury.

7 Závěr

V teoretické části byly popsány nízkoúrovňové příznaky barva a textura. Byla rozebrána metoda JEC, která bude v bakalářské práci implementována. Seznámili jsme se s knihovnou OpenCV, prostudovali obrázky a přiložená metadata od ČTK, ESP a iaprtc12.

8 Uživatelská dokumentace

popsani jak vypada zdrojovej soubor kterej to zere, nejdriv cesta k souboru a pak jeho klicovy slova
že je potřeba aspoň těch 16 Gb RAM

Literatura

[1] Ameesh Makadia, S. K. V. P. A new baseline for image annotation. *Commun. ACM*. July 1961, 4, 7, s. 321. ISSN 0001-0782. doi: 10.1145/366622.366644. Dostupné z: http://doi.acm.org/10.1145/366622.366644.

[2] Cruse, H. Neural Networks as Cybernetic Systems. *Commun. ACM*. July 1961, 4, 7, s. 321. ISSN 0001-0782. doi: 10.1145/366622.366644. Dostupné z: http://doi.acm.org/10.1145/366622.366644.

[3] Hoare, C. A. R. Algorithm 64: Quicksort. *Commun. ACM*. July 1961, 4, 7, s. 321. ISSN 0001-0782. doi: 10.1145/366622.366644. Dostupné z: http://doi.acm.org/10.1145/366622.366644.

[4] *Class Graphics2D* [online]. Oracle, 2016. [cit. 2016/03/09]. Java SE Documentation. Dostupné z: https://docs.oracle.com/javase/7/docs/api/java/awt/Graphics2D.html.

[5] Knuth, D. E. *The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms*. Addison-Wesley Longman Publishing Co., Inc., 1997. ISBN 0-201-89684-2.

[6] V. Lavrenko, J. J. R. A model for learning the semantics of pictures. *Commun. ACM*. July 1961, 4, 7, s. 321. ISSN 0001-0782. doi: 10.1145/366622.366644. Dostupné z: http://doi.acm.org/10.1145/366622.366644.