VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

AUTOMATICKÁ KATEGORIZACE FOTOGRAFIÍ PODLE OBSAHU

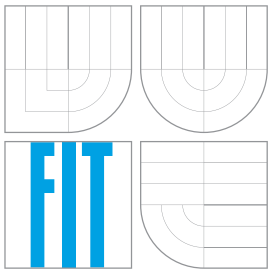
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

LADISLAV NĚMEC

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

AUTOMATICKÁ KATEGORIZACE FOTOGRAFIÍ PODLE OBSAHU

AUTOMATIC CONTENT-BASED IMAGE CATEGORIZATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — LADISLAV NĚMEC
AUTHOR

VEDOUCÍ PRÁCE — Ing. MARTIN VEĽAS
SUPERVISOR

BRNO 2015

Abstrakt

Tato práce se zabývá problematikou klasifikace fotografií podle obsahu. Hlavním cílem práce je implementace aplikace, která je schopná tuto kategorizaci provádět. Řešení se sestává z variabilního systému využívajícího extrakce lokálních příznaků v obraze a vytvoření vizuálního slovníku metodou k-means. Aplikace využívá Bag of Words reprezentace jako globální funkce pro popis každé fotografe. Poslední složkou tohoto systému je klasifikace prováděná na základě Support Vector Machines. V poslední kapitole jsou představeny výsledky experimentování s tímto systémem.

Abstract

This thesis deals with automatic content-based image classification. The main goal of this work is implementation of application which is able to perform this task automatically. The solution consists of variable system using local image features extraction and visual vocabulary built by k-means method. Bag Of Words representation is used as a global feature describing each image. Support Vector Machines - the final component of this system - perform the classification based on this representation. In the last chapter, the results of this experimental system are presented.

Klíčová slova

Automatická kategorizace fotografií, lokální příznaky, význačné body, SURF, k-means, visuální slovník, visuální slova, Bag Of Words, Support Vector Machines, knihovna OpenCV.

Keywords

automatic content-based image categorization, local features, interesting points, SURF, k-means, visual codebook, visual word, Bag Of Words, Support Vector Machines, OpenCV library.

Citace

Ladislav Němec: Automatická kategorizace fotografií podle obsahu, bakalářská práce, Brno, FIT VUT v Brně, 2015

Automatická kategorizace fotografií podle obsahu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Martina Veľase. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Ladislav Němec
19. května 2015

Poděkování

Chtěl bych na tomto místě poděkovat za příkladné vedení a podnětné rady vedoucímu práce Ing. Martinu Veľasovi

Obsah

Kapitola 1

Úvod

V dnešní době je stále jednodušší informace, takřka jakéhokoli druhu, získávat, ale také dát je i dispozici druhým. Celý den se střetáváme s různými druhy informací, které získáváme z velkého množství zdrojů v různých podobách, pracujeme s nimi a snažíme se je co nejvíce využít. Čím dál více se začíná využívat i počítačového vidění, což je odvětví počítačové techniky a vývoje softwaru zabývající se vytvářením softwaru schopného získat informaci ze zachyceného obrazu.

Tato práce se zabývá především kategorizací fotografií podle obsahu. To má uplatnění například v automatickém zpracovávání obrazu z průmyslových kamer, při průmyslové výrobě či v medicíně. Součástí práce je zkoumání efektivnosti použitých metod a hledání ideálního přístupu pro získání maximální přesnosti při kategorizaci.

Praktickým výsledkem této práce by měl být program pro zpracování sady fotografií, který by dokázal fotografie třídit do předem určených skupin podle obsahu. Pro zpracování obrazu je použito existujících knihoven. Dalším cílem práce je zjistit. jak změna parametrů či druhu použité metody ovlivňuje konečný výsledek kategorizace.

Následující kapitola 2 - *Principy metod použitých při kategorizaci fotografií* se zabývá především použitými technikami a vysvětlením toho, jak fungují. Kapitola 3 - *Současné aplikace, inovativní řešení a soutěže* se věnuje současným aplikacím využívajících vyhledávání objektů v obraze a uvedeny jsou také některé inovativní principy. O návrhu jednotlivých částí programu a návrhu testování pojednává kapitola 4 - *Návrh řešení*. Kapitola 5 - *Implementace* informuje o implementaci mé aplikace, která je nakonec zhodnocena v kapitole 6 - *Dosažené výsledky práce*, kde je popsán také vliv použití jednotlivých metod na výsledek klasifikace. V poslední kapitole 7 - *Závěr* je posouzeno zda-li jsou splněny stanovené cíle, a navrženo možné rozšíření programu.

Kapitola 2

Principy metod použitých při kategorizaci fotografií

Tato kapitola věnuje použitým technikám při kategorizaci. Celý proces se sestává ze dvou hlavních částí. První fáze učení, při níž se využívají trénovací data. Z každé fotografie se vyextrahují obrazové příznaky a na jejich základě se vytvoří vizuální slovník a Bag of Words. Poté následuje vytvoření klasifikátoru pro každou ze tříd pomocí Support Vector Machines. Ve fázi druhé dochází už k samotné kategorizaci, při té se využívá k přiřazení fotografie do podskupiny komponenty z fáze předešlé.[24][7]

2.1 Vyhledání lokálních příznaků

Při kategorizaci fotografií je důležité určit, co bude v našem případě předmětem pro porovnávání. Samotný obrazový záznam obsahuje spoustu informací, které mohou být pro třídění do podskupin důležité. Může jím být barva objektů, jejich tvar, rozmístění hran, rohů a další podobné aspekty.

V mé práci je použito hledání podobnosti u objektů patřících do stejné kategorie, a proto pro porovnávání je využito lokálních příznaků, které se poté použijí pro tvorbu vizuálního slovníku. Proces vyhledání příznaků se dá rozdělit do tří hlavních částí. V první fázi se hledají klíčové body fotografie, kterými mohou být rohy, bloby nebo T-spoje. Pro jejich nalezení se používají různé algoritmy (SIFT, SURF, BRISK, ORB ...), které využívají okolí těchto bodů. Druhou fází je výpočet deskriptoru z okolí klíčového bodu. To znamená, že se tento výřez obrazu převede na vektor. Nakonec jsou deskriptory jednotlivých obrazů srovnávány. Porovnávání je založené na vzdálenosti mezi vektory. [4][9]

2.2 Algoritmus SURF

SURF (Speeded-Up Robust Features) je metoda sloužící jako detektor klíčových bodů, ale i k popisu fotografie pomocí deskriptorů. Jedná se o novější obdobu metody SIFT[18]. Mezi ostatními detektory vyniká především svou rychlostí a zároveň se blíží nebo dokonce překonává dříve navržené prostředky. Je optimalizována pro srovnávání dvou objektů, které jsou deformovány. Může se jednat například o otočení, změnu velikosti, rotaci a podobně.

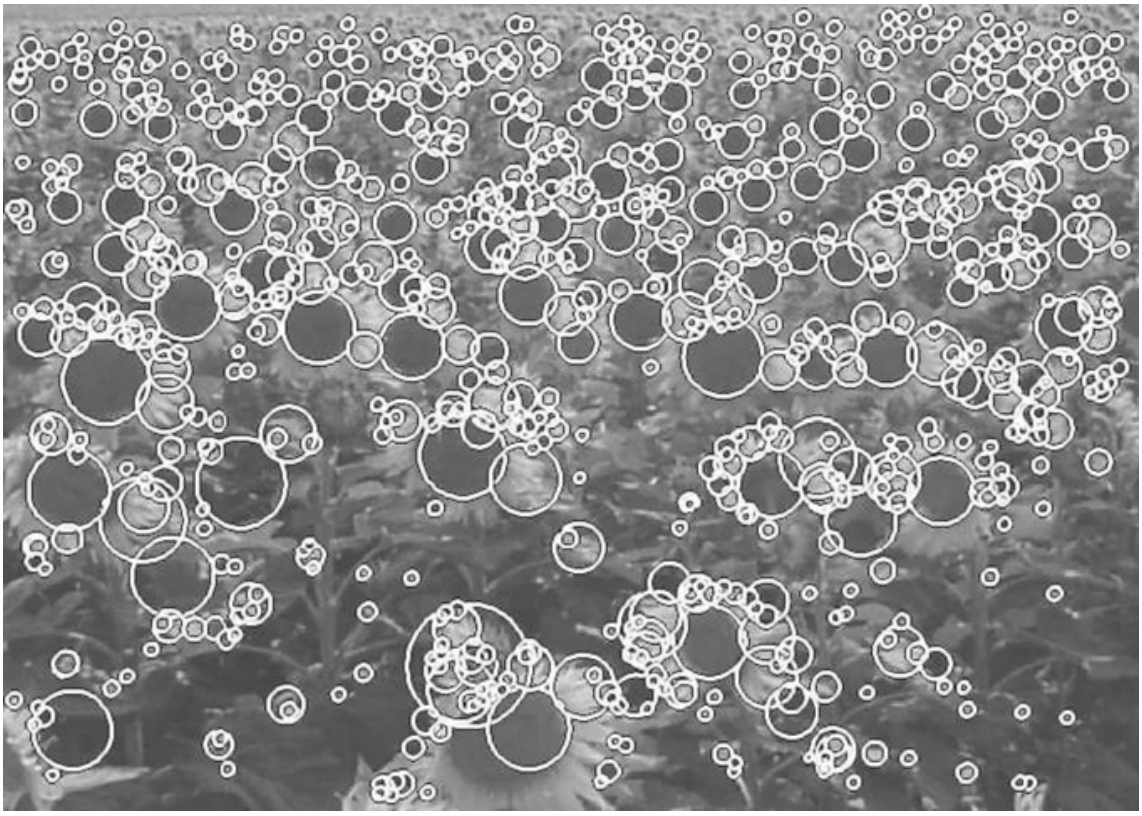
SURF je nezávislý na barvách daného obrazu, využívá pouze intenzity bodů ve formě integrálního obrazu. Je možné ho rychle vypočítat ze vstupního obrazu a použít k urychlení výpočtu v jakékoli svislé obdélníkové oblasti. Integrální obraz $I_{\sum}(x)$ v bodě $x = (x, y)^{\top}$ lze

definovat jako součet intenzit všech obrazových bodů ve vstupním obraze I mezi počátkem a bodem x. Pro výpočet lze formálně použít následující vzorec 2.1. [9]

$$I_{\sum}(x) = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq x} I(i,j) \tag{2.1}$$

Detektor SURF je založen na determinantu Hessovy matice. Tato matice se pro daný účel projevila jako výkonný prostředek. Na rozdíl od Hesson-Laplace detektoru využíváme determinant Hessovy matice i pro výběr rozsahu. Pokud máme bod $x = (x, y)$ v obraze I, Hessova matice $H(x, \sigma)$ v bodě x pří měřítku σ je potom dána vzorcem 2.2. Kde $L_{xx}(x, \sigma)$ je konvolucí Gaussovy derivace druhého řádu $\sigma^2/(\sigma x^2)g(\sigma)$ s obrazem I v bodě x. Podobně pro $L_{xy}(x, \sigma)$ a $L_{yy}(x, \sigma)$. Příklad použití Hassovy matice pro detekci klíčových bodů je na obrázku. 2.1.[4][9]

$$H(x, \sigma) = \begin{bmatrix} L_{xx}(x, \sigma) & L_{xy}(x, \sigma) \\ L_{xx}(x, \sigma) & L_{yy}(x, \sigma) \end{bmatrix} \tag{2.2}$$

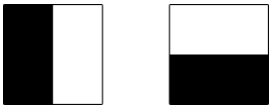


Obrázek 2.1: Detekované klíčové body ve fotografii pole slunečnic. Body byli získány pomocí Hassovy matice. Obrázek je převzat z [4].

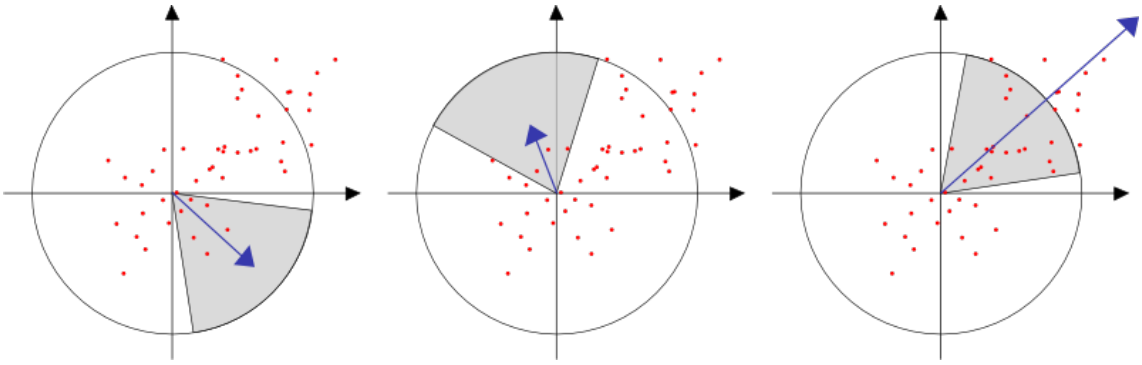
Deskriptor slouží k popisu rozložení intenzity v okolí klíčového bodu. K jeho určení je použita odezva tzv. Haarovy vlnky prvního řádu. Proces převodu se sestává ze dvou fází.

První fáze je přiřazení orientací. Zde se používá Haarovy vlnky pro nalezení gradientu obrazu ve směru x i y (filtry jsou zobrazeny na obrázku 2.2). Po té se sečtou horizontální

i vertikální odezvy v každém intervalu $\pi/3$ kolem souřadné osy zobrazené na obrázku 2.3. Pro každý interval je dále vypočítána velikost vektoru, který je daný počátkem a sumami odezev. Největší takový vektor je následně prohlášen dominantní orientací.



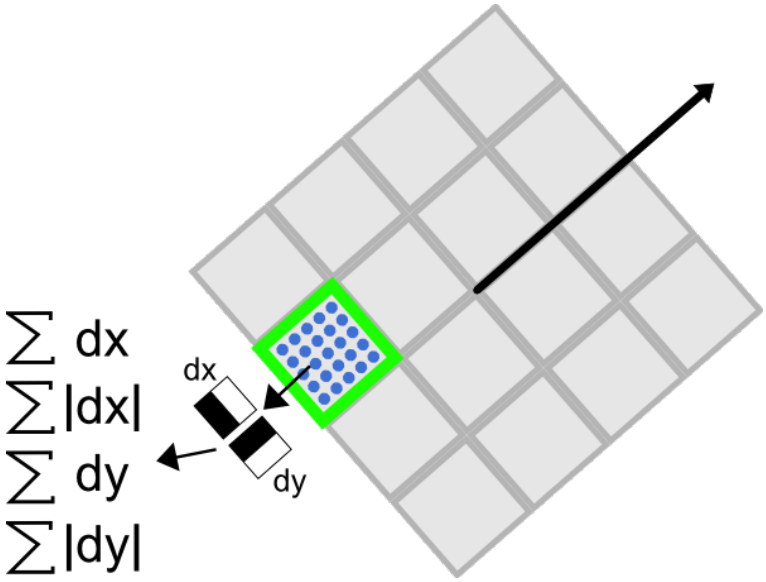
Obrázek 2.2: Haarovy vlnky. Levý filtr vypočítá odezvu ve směru osy x a pravý ve směru osy y. Hodnoty jsou 1 pro černé oblasti a -1 pro bílé. Obrázek je převzat z [9].



Obrázek 2.3: Přiřazování orientací: Kruhová výseč rotuje kolem středu. Pro každou pozici se se hodnoty odezev, nacházející se uvnitř, sečtou a vytvoří se samostatný vektor. Na základě směru největšího vektoru je určena dominantní orientace. Obrázek převzat z [9].

Konečná fáze je extrahováni deskriptoru. Je sestrojeno čtvercové okno o velikosti 20 s (s je velikost měřítka). Orientace čtverce je získána z předchozího kroku. Toto okno je dále rozděleno na 16 stejných čtvercových podoblastí. V každé z těchto podoblastí jsou vypočteny odezvy Haarovy vlnky o velikosti 2 s ve směru x (dx) a y (dy) pro 25 rovnoměrně rozmístěných vzorkových bodů. Následně je každá podoblast popsána vektorem v, jehož výpočet je ve vzorci 2.3 [4]. Následným zřetězením těchto vektorů vznikne finální deskriptor klíčového bodu o délce 4 x 16 = 64. Komponenty deskriptoru jsou znázorněny na obrázku 2.4.

$$v = \left[\sum dx, \sum dy, \sum |dx|, \sum |dy| \right] \tag{2.3}$$



Obrázek 2.4: Komponenty deskriptoru: Zelená čtvercová hranice zobrazuje jednu z 16 podoblastí. Modré tečky znázorňují 25 vzorkových bodů, na nichž jsou vypočítat vlnkové odezvy. Výsledky jsou vypočteny vzhledem k dominantní orientaci. Obrázek převzat z [9].

2.3 Vizuální slovník

Poté, co jsou v obraze nalezeny klíčové body a převedeny na deskriptory, je potřeba vytvořit reprezentující datovou strukturu deskriptorů napříč všemi trénovacími fotografiemi. Právě takovou strukturou je vizuální slovník, který obsahuje soubor vizuálních slov, jenž jsou pro dané skupiny extrahovaných deskriptorů charakteristické.

Jde o analogii pro zpracování přirozeného jazyka, kdy se snažíme zjistit, o čem hovoří dané texty jednotlivých dokumentů. To provádíme na základě porovnání četnosti výskytu jednotlivých slov.

Vizuální slovo je shluk deskriptorů. Tyto slova se získají prostřednictvím algoritmu pro třídění dat do shluků. Může jím být například algoritmu k-means++ popsaný níže (viz 2.4). Pro výkonný vizuální slovník je potřeba určit dobrý poměr velikostí jednotlivých vizuálních slov a jejich počtem ve slovníku. V praxi bylo zjištěno, že nejlepší kompromis přesnosti a výpočetní účinnosti je získán při středně velké velikosti shluků. [7][25]

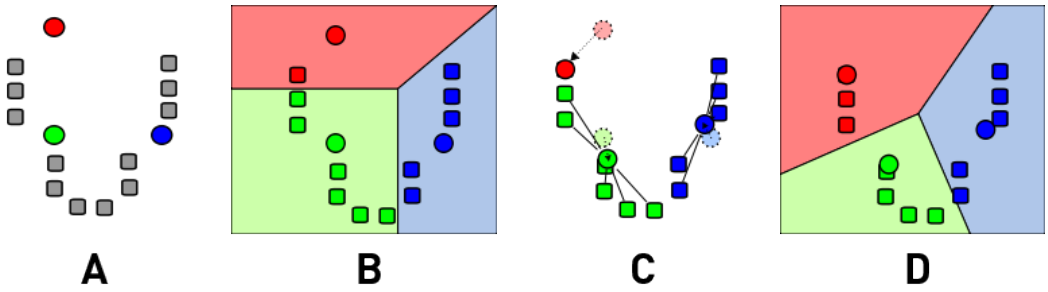
2.4 Algoritmus K-means++

K-means je široce používaná metoda sloužící pro třídění dat do shluků tzv. clusterů. Snaží se minimalizovat průměrnou vzdálenost mezi body ve stejném shluku. Toto minimum vypočítáme vzorcem 2.4, kde k je počet výsledných množin bodů S_i, x_j je d-dimenzionální vektor, který reprezentuje i-tý bod, a μ_i je středem shluku v S_i. Na počátku je určen počet shluků a náhodně se ustanoví jejich středy. Poté se provádí samotný algoritmus, který je iterativní. [2]

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} (x_j - \mu_i) \tag{2.4}$$

Algoritmus k-means popsaný v krocích:

1. Náhodně je vybráno k datových bodů, které jsou považovány za středy clusterů.
2. Každá datový bod je poté přiřazen k nejbližšímu středu shluku.
3. Následně se přepočtou středy shluků.
4. Algoritmus je u konce, pokud jsou všechny datové body přiřazeny do stejných shluků, jako tomu bylo v předešlém kroku iterace. Pokud tomu tak nyní opakují se znovu body 2. a 3. Může být i jiná podmínka ukončení, jako je dosáhnutí maximálního počtu iterací nebo dosažení určité přesnosti.



Obrázek 2.5: Algoritmus v k-means popsaný po krocích. Pořadí obrázků zleva doprava je shodný s kroky algoritmu 1-4 popsanými výše. Obrázky jsou převzaté z [12].

I když k-means nenabízí žádné záruky přesnosti, jeho jednoduchost a rychlost jsou atraktivní pro to, aby se často používali v praxi.

Nicméně algoritmus k-means má alespoň dva hlavní nedostatky: [3]

- Zaprvé bylo prokázáno, že nejhorší případ pro dobu běhu je super-polynom ve vstupní velikosti.
- Zadruhé, nalezení aproximace může být libovolně chybné, pokud je cílová funkce srovnatelná s optimálním shlukováním.

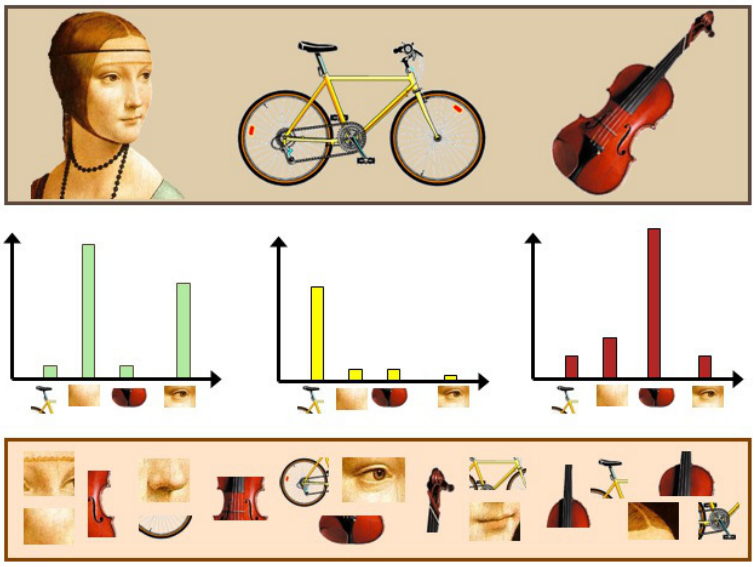
Algoritmus k-means++ [3] řeší druhý z těchto problémů určením postupu pro inicializaci center shluků, před tím než pokračuje klasickou metodou k-means. S k-means++ inicializací je zaručeno, že nalezneme řešení, které je $O(\log k)$, konkurenční k optimálnímu k-means řešení.

Přesný algoritmus inicializace je následovný:

1. Vyberme si jako střed prvního shluku náhodně jeden ze svých datových bodů.
2. Pro každý datový bod x vypočítáme $D(x)$, což je vzdálenost mezi x a nejbližším centrem, které již bylo určeno.
3. Vybereme náhodně jeden nový datový bod jako nové váhové centrum s použitím váženého rozdělení pravděpodobnosti, kde je bod x vybrán s váhou pravděpodobnosti $D(x)^2$.
4. Budeme opakovat body 2. a 3. dokud nebudou vybrány všechny K centra.
5. Nyní, když byla zvoleny počáteční centra, pokračujeme s použitím klasické metody k-means.

2.5 Bag of words

Ve vizuálním slovníku máme data reprezentovaná množinou vizuálních slov. Pro popis každé fotografie potřebujeme způsob jak vyjádřit četnost těchto slov v obraze. Pro tento účel nám slouží Bag of Words (BoW) [11]. Jde o metodu původně určenou pro popis obsahu textových dokumentů. BoW je v podstatě histogram popisující četnost výskytu vizuálních slov ze slovníku ve fotografii. Je málo pravděpodobné, že by vizuální slova ve slovníku a vizuální slova nalezená ve fotografii, u které se tvoří BoW reprezentace, byla totožná, proto je nutné při jejich srovnávání počítat s mírnou aproximací.



Obrázek 2.6: Reprezentace obrazu pomocí Bag of Words. Dolní obdélník reprezentuje vizuální slovník, ve kterém jsou vizuální slova. Histogramy reprezentují Bag of Words jednotlivých objektů znázorněných v horním obdélníku. Obrázek převzat z [17].

2.6 Support Vector Machines

Support Vector Machines (SVM) je populární technika pro klasifikaci. SVM hledá nadrovinu, která optimálně rozděluje trénovací data v prostoru příznaků. Nadrovina je optimální, jestliže body leží v opačných poloprostorech a hodnota minima vzdáleností bodů je co největší. Kolem nadroviny je na obě strany co nejširší pruh bez bodů tzv. maximální odstup (angl. maximal margin). Pomocí nejbližších bodů, kterých obvykle není mnoho, vzniká popis nadroviny. Tyto body se nazývají podpůrné vektory (angl. support vectors). Podle těchto vektorů je odvozen i název metody. Metoda rozděluje data do dvou tříd. Rozdělující nadrovina je lineární funkce v prostoru příznaků. Viz obrázek 2.7.

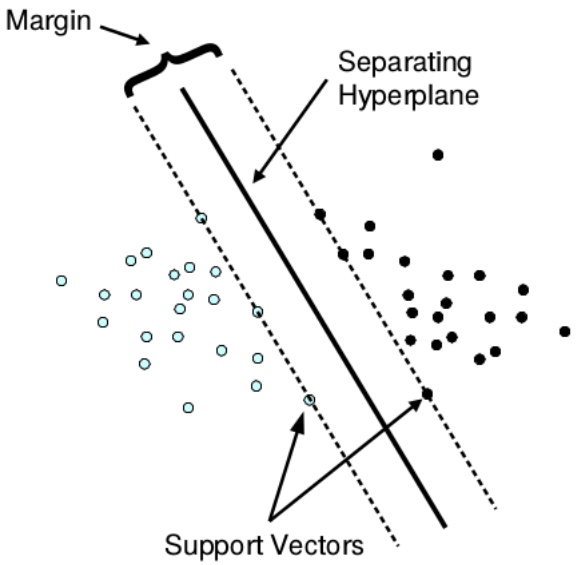
Trénovací sada dat jsou dvojice (x_i, y_i), kde $i = 1, 2, ..., l$, $x \in R^n$ a $y \in 1, -1$, což značí příslušnost do třídy. Support Vector Machines techniky využívají jádrové transformace (angl. kernel transformation) prostoru příznaků dat do prostoru transformovaných příznaků typicky vyšší dimenze. V situaci, kdy je úloha původně lineárně neseparovatelná, dokáží tyto jádra převést úlohu na lineárně separovatelnou. Poté lze na úlohu aplikovat optimalizační algoritmus pro nalezení nadroviny. U lineárního jádra je nadrovina definovaná pomocí $K(x_i, x_j) = {x_i}^T x_j$, tj. hodnota jádrové funkce K na datech v původním prostoru

parametrů. U ostatních jader je skalární součin zaměněn za jádrové funkce způsobující výše zmíněnou transformaci prostoru dat. [13][19]

Druhy jádrových transformací:

- Lineární (bez transformace) - $K(x_i, x_j) = {x_i}^T x_j$
- Polynomické jádro - $K(x_i, x_j) = exp(\gamma\ {x_i}^T x_j + r)^d;\ \gamma > 0$
- Sigmoida - $K(x_i, x_j) = tanh(\gamma\ {x_i}^T x_j + r)$
- RBF (Radial Base Functions) - $K(x_i, x_j) = exp(-\gamma \|x_i - x_j\|)$

Parametry γ a d zadává uživatel, r se počítá. Parametr γ má význam strmosti jádra, vyšší hodnoty γ vedou k podrobnějším, někdy i méně stabilním modelům.



Obrázek 2.7: Klasifikace (lineárně oddělená data). Obrázek převzat z [19].

Kapitola 3

Současné aplikace, inovativní řešení a soutěže

Žijeme v době, kdy množství dat zaznamenaných v podobě digitální fotografie anebo videa roste. Vzniká potřeba s těmi daty efektivně pracovat. Může jít o vyhledávání, kategorizaci, porovnávání a spoustu dalších disciplín, při kterých se využívá vyhledání objektů v obraze a jejich následné komparace. V této kapitole uvedu některé příklady aplikace zmíněných principů a rozpoznávaní obrazové předlohy obecně. Aplikací tohoto typu je velmi velké množství, proto uvedu jen některé. Dále také uvedu některé z inovativních přístupů v této vědní disciplíně a nakonec se budu věnovat soutěži *The Pascal Visual Object Classes Challenge*, která je zaměřena přímo na kategorizaci fotografií a srovnávání existujících metod.

3.1 Aplikace využívající kategorizaci fotografií

V dnešní době se stává stále větším standardem, že existující aplikace dokáží kategorizovat fotografie podle obsahu, vyhledávat objekty ve scéně, rozpoznávat psaný text a spoustu dalších úkonů týkajících se rozpoznávání obrazové předlohy. V této kapitole uvedu pár příkladů těchto aplikací.

- *Google images* - Již v červenci roku 2001 bylo spuštěno v prohlížeči Google vyhledávání obrázků. Jedná se o službu, která v této době nabízela přístup k 250 milionům obrázků (nyní jsou to již miliardy). V dubnu roku 2009 je zavedena experimentální funkce vyhledávání podobných obrázků. Jedná se o funkcionalitu zabudovanou přímo do vyhledávače, která dokáže nalézt obrázky vizuálně podobné a nebo stejné jako námi předložený vzor. Obrázky jsou programu předkládány jako URL odkaz nebo můžou být nahrány přímo z počítače. Obraz se analyzuje na základě barev, klíčových bodů, linek a textur.[27]

- *Adobe Photoshop Elements* - Jedná se o placený, domácky cílený správce a editor fotografií. Tento program také od verze 8 dokáže jako jeden z prvních rozpoznat tváře v obraz [30]. Od verze 10 dokáže vyhledat obrázky s duplicitním nebo podobným obsahem [21]. Od této verze je také k dispozici tzv. Object Search, nástroj pro vizuální vyhledávání, který umí hledat objekty uvnitř fotografií. Uživatel rámečkem vybere objekt v obraze, který chce vyhledat, a program nalezne fotografie, na kterých se tento objekt nachází. [26]

- *Pittpatt* - Jde o společnost, kterou v červenci roku 2011 odkoupila společnost Google. Vyvinuli software sloužící k detekci tváří a jejich následné rozpoznání. Google ho integroval do svých produktů (např. prohlížení fotografií na sociální síti Google+, Picasa a pod.). Je možné ve fotografiích uživatele rozpoznat tváře a pak k nim následně automaticky přiřadit jméno. [6]

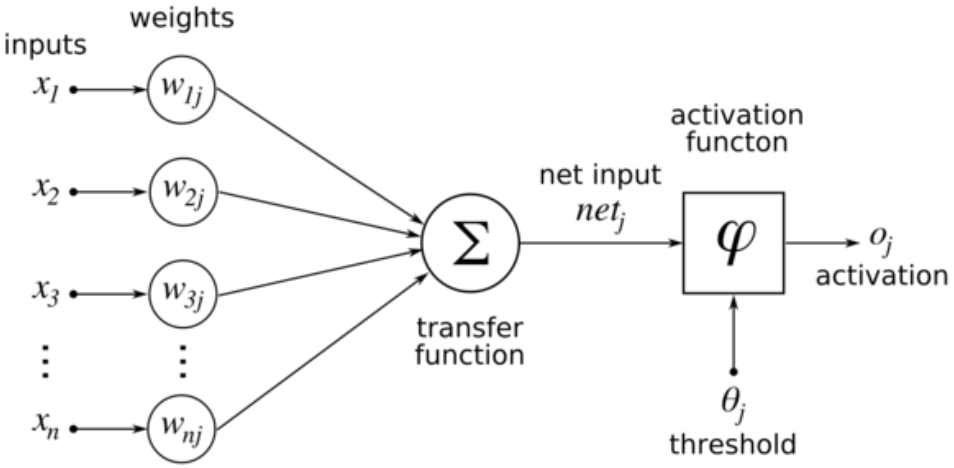
- *ABBYY FineReader* - ABBYY FineReader umožňuje jednotlivcům převádět naskenované dokumenty, soubory PDF a digitální fotografie na editovatelné dokumenty s možností vyhledávání. Verze programu, FineReader 12, rozpozná tištěný text ve 190 jazycích. [1]

- *Zpracování medicínských dat* - Použití automatické metody klasifikace dokáže přispět k zlepšení diagnostiky pomocí lékařských snímků. Využitím obrazových dat od pacientů, u kterých se potvrdila diagnóza, je možné rozpoznat některé poruchy na základě jejich podobnosti. Toto se děje většinou ručně. Je ovšem možné použít automatickou kategorizaci obrazu, která dokáže velký obsah dat zpracovat ve velmi krátkém čase. [29]

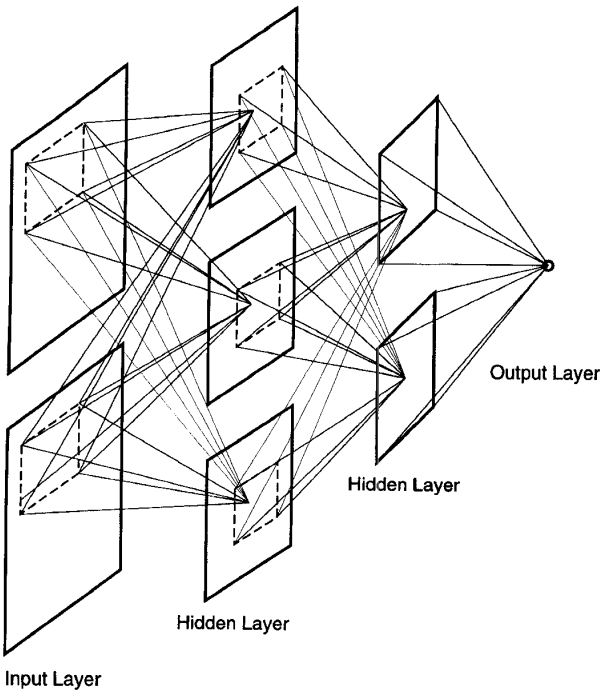
3.2 Inovativní přístupy

Jak již bylo několikrát zmíněno, rozpoznávání objektů v obraze má široké uplatnění a to v mnoha oblastech. Proto je přirozené, že se toto odvětví informatiky stále vyvíjí a je i mnoho inovativních přístupů. V mé práci byl použit jeden ze základních modelů. Jsou i jiné možnosti jak kategorizaci obrazu řešit. Pro představu zde některé jiné přístupy či inovace uvedu.

- *Neuronové sítě* - Umělé neuronové sítě jsou jeden z modelů inspirovaných biologickými strukturami, konkrétně neuronovou sítí v mozku. Základní stavební jednotkou neuronové sítě je neuron 3.1. V biologických neuronových sítích jsou zkušenosti uloženy v dendritech. V umělých neuronových sítích jsou zkušenosti uloženy v jejich matematickém ekvivalentu tzv. váhách. Vstupy neuronu jsou těmito váhami násobeny a následně jsou sečteny. Tento výsledek je předán aktivační funkci. Cílem učení neuronové sítě (tato fáze se nazývá adaptivní) je nadstavit ji tak, aby v klasifikaci byla co nejpřesnější. Díky schopnosti učit se jsou neuronové sítě velmi variabilní a dají se použít na řešení velkého množství problémů. Existuje celá řada typů neuronových sítí a každý typ se hodí pro jinou třídu úloh. V klasifikaci se používají například pro extrakci příznaků ve formě tzv. autoenkoderů. Méně běžný přístup je použití neuronových sítí místo klasifikace SVM. Naopak běžnou metodou je použití hlubokých konvolučních neuronových sítí pro celou úlohu klasifikace (od detekce příznaků až po zařazení fotografie do třídy). Ukázka modelu konvolučních sítí je na obrázku 3.2. [20] [22]



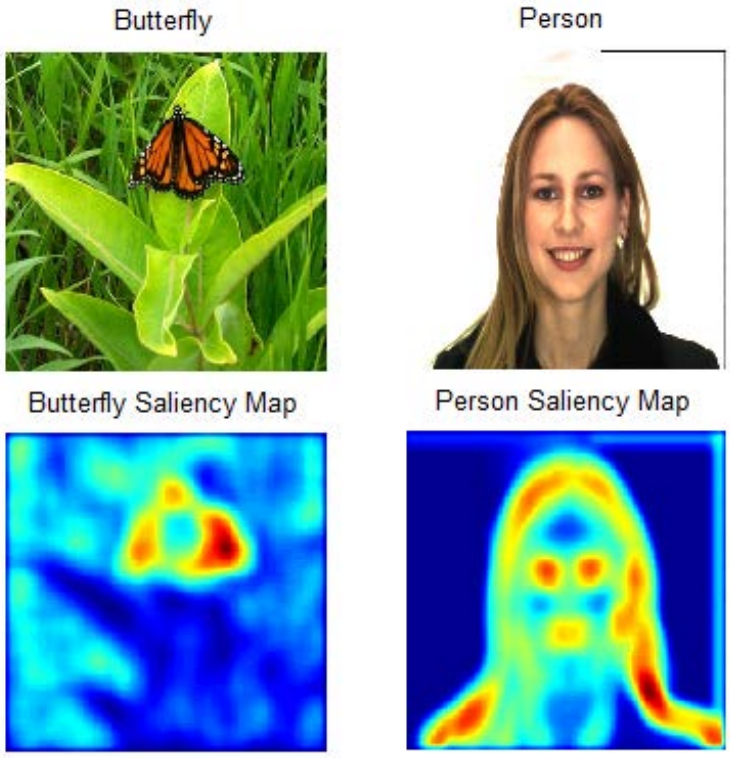
Obrázek 3.1: Neuron - základní stavební jednotka umělých neuronových sítí. Obrázek převzat z [28].



Obrázek 3.2: Struktura čtyřvrstvé konvoluční neuronové sítě. Na vstupu jsou dvě fotografie. Síť má jeden reálný výstup. Obrázek převzat z [22]

- *Barevné korelogramy* - Barevný korelogram je jeden ze způsobu zaznamenání informace o obraze. Jde v podstatě o barevný histogram rozšířený o informaci o prostoru popisující lokální prostorovou korelaci barev v obraze. Barevné korelogramy jsou tedy soubory dvojic barev, které vyjadřují pravděpodobnost, se kterou je pixel mající první barvu umístěn v určité vzdálenosti od pixelu, který má druhou barvu. Autokorelogram je speciální druh korelogramu určující pravděpodobnosti vzdáleností pixelů mající pouze stejnou barvu. [32]

- *Vizuální pozornost (angl. saliency)* - Lidé dokáží přesně rozpoznat tisíce kategorií objektů. Tento fakt motivoval výzkumníky z oblasti počítačového vidění ke studiu zpracování obrazu lidským zrakem za účelem získat provozní principy, které mohou zlepšit techniku rozpoznávání objektů. Zjistili, že mnohé principy jsou již v počítačovém vidění realizovány. Nicméně jeden z principů používaný při rozpoznávání objektů, který byl do té doby ze strany odborníků ignorován, je vizuální pozornost. Vzhledem k tomu, že lidé nemohou zpracovat celou vizuální scénu najednou, fixují svůj pohled na jednotlivé objekty a zajímavé aspekty ve scéně. Oblast scény je analyzována a následně se pozornost zraku přesměruje na další významný prvek. Lidé tento jev, prudké přesunutí očí na objekt zájmu ve scéně a následné vizuální zpracování, provádí více než 170.000 krát za den a asi 3 krát za vteřinu. Toto chování se nazývá vizuální pozornost. Tzv. saliency maps, které jsou znázorněny na obrázku 3.3, jsou úspěšná a biologicky věrohodná technika pro modelování vizuální pozornosti. Těchto map lze využít při vyhledávání a následné kategorizaci objektů v obraze. Více informací je zde [15].

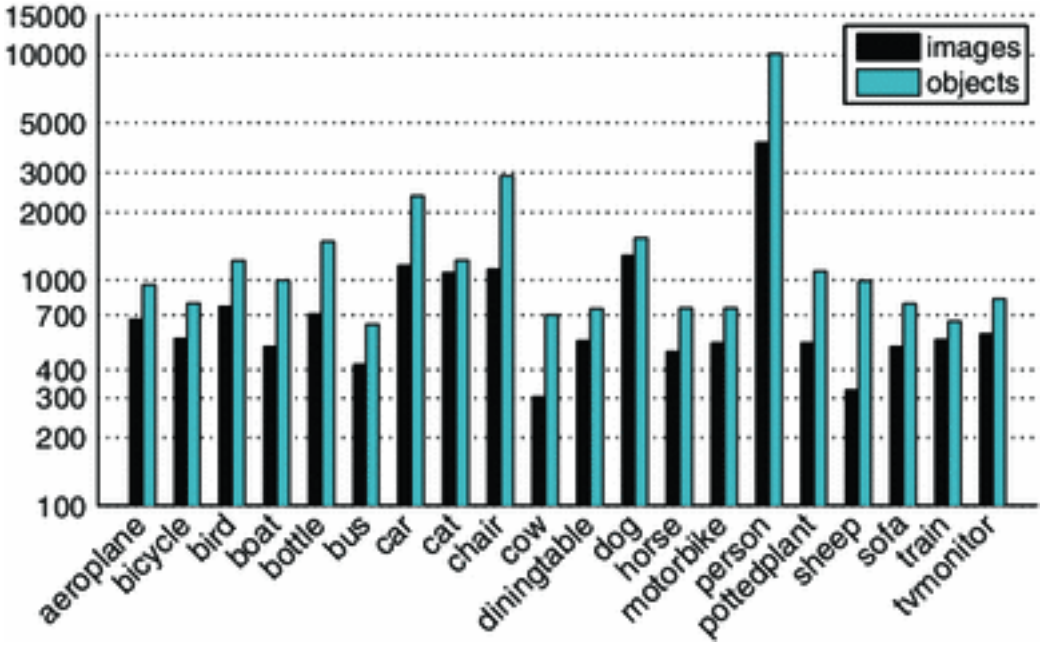


Obrázek 3.3: Dva obrázky a jejich odpovídající saliency maps, které indikují funkce oblastí zájmu v obraze. Hodnoty s vysokou nápadnosti jsou červené a s nízko nápadnosti jsou modré. Obrázek převzat z [15].

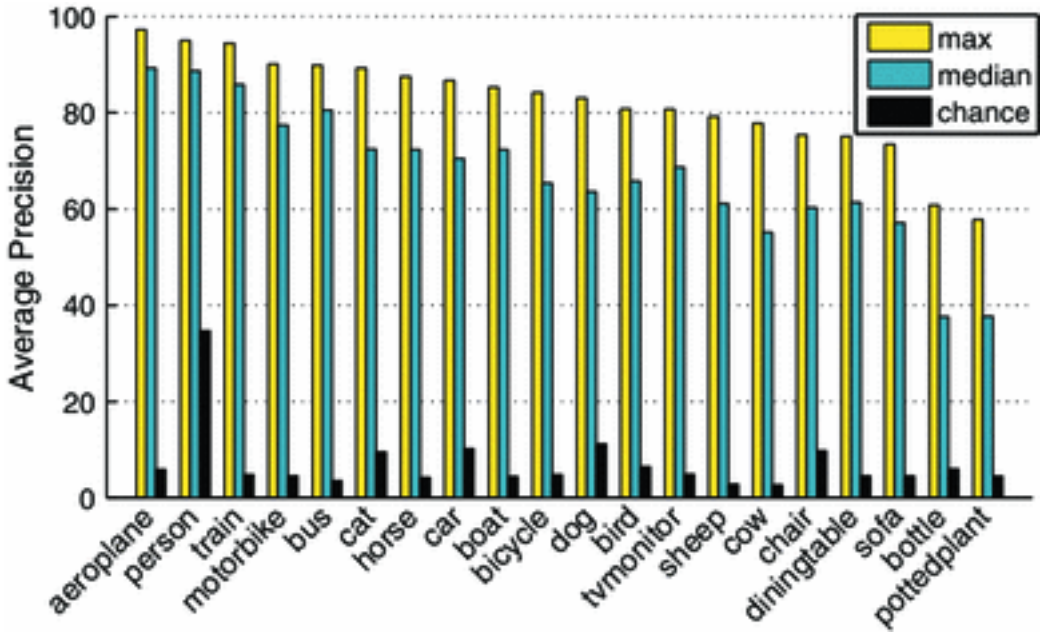
3.3 The Pascal Visual Object Classes Challenge

The Pascal Visual Object Classes (VOC) Challenge [10] se skládá ze dvou částí. Zaprvé jede o veřejně dostupnou sadu snímků získaných z webových stránek Flickr. Tato sada obsahuje

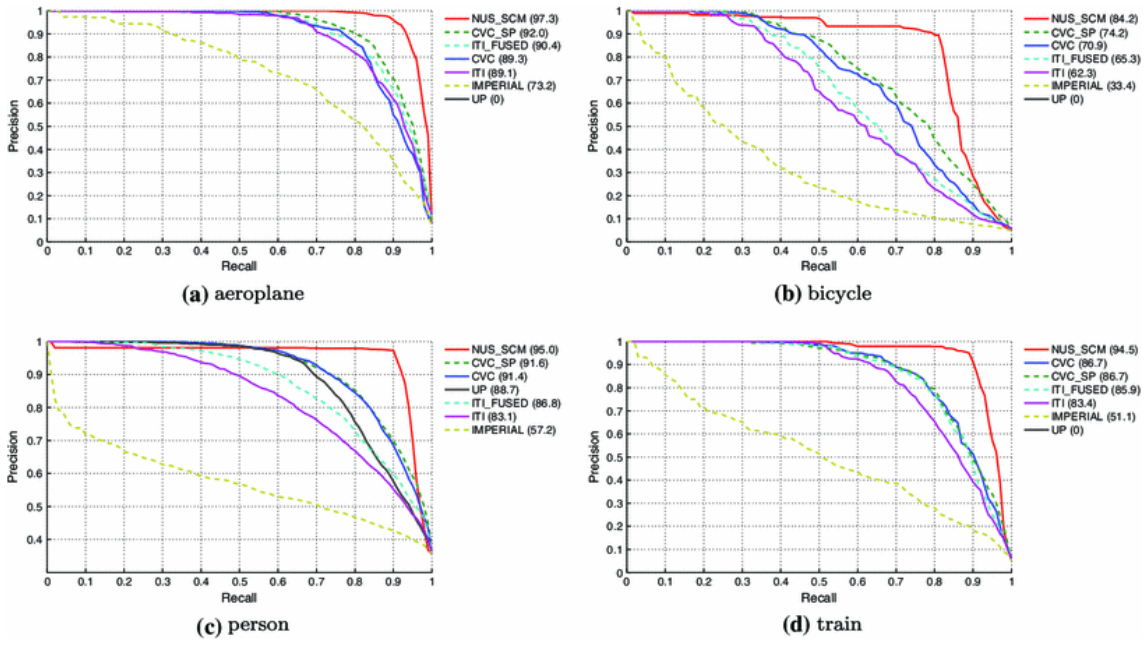
rovněž anotací, která ji popisuje. Zadruhé se jedná o každoroční soutěž, která probíhala v letech 2006 - 2012. Tato soutěž měla pět disciplín: kategorizace podle objektů, detekce objektů, segmentace, klasifikace akcí a rozlišování částí lidského těla. Má práce se zabývá kategorizací obrazu podle objektů na fotografii, tudíž uvedu některá data z této disciplíny. V posledním roce soutěže se sada snímků skládala z 11540 snímků obsahujících 31561 objektů. Histogram rozložení této sady do 20 kategorií můžete vidět zde 3.4. Na obrázku 3.5 jsou výsledky posledního ročníku této soutěže. Výsledky jsou uváděny v průměrné přesnosti (angl. average precision), což je hodnota obsahu plochy pod precision/recall křivkou. Zde můžete vidět konkrétní precision/recall křivky některých z účastníků soutěže u vybraných tříd 3.6.



Obrázek 3.4: Shrnutí datové sady VOC2012. Histogram počtu objektů a obrazů, které obsahují alespoň jeden objekt odpovídající třídy. Obrázek získán z [10].



Obrázek 3.5: Souhrn výsledků klasifikace podle třídy. Pro každou třídu jsou znázorněny tři hodnoty: maximální hodnota AP získaná jakýmkoliv způsobem (max), střední hodnota AP ze všech metod (median) a hodnota AP získaná náhodný tříděním snímků (chance). Obrázek získán z [10].



Obrázek 3.6: Výsledky klasifikace. precision/recall křivky jsou zobrazeny na reprezentativním vzorku tříd. Legenda udává hodnotu AP (%) získané odpovídajícím způsobem u jednotlivých účastníků soutěže. (a) letadlo, (b) jízdní kolo, (c) osoba, (d) vlak. Obrázek získán z [10].

Kapitola 4

Návrh řešení

Jedním z cílů této práce je fungující aplikace schopná kategorizace fotografií do jednotlivých tříd. V této kapitole se věnuji jejímu návrhu. Kapitola popisuje jednotlivé části systému a jejich vzájemné provázání a podrobněji se věnuje některým z důležitějších částí tohoto systému. Dále je také popsán návrh na testování aplikace.

4.1 Obecný návrh architektury

V první fázi vypracování návrhu jsem si vymezil jednotlivé části systému, které výsledná aplikace bude obsahovat. U každého z prvků systému jsem si definoval jeho činnost, vstupy a výstupy. Systém se skládá z dvou hlavních částí:

- *Fáze učení*
  - *Vstup:*
    - soubor trénovacích fotografií
    - data popisující příslušnost fotografií do jednotlivých tříd
  - *Funkce:*
    - extrahování příznaků z trénovacích fotografií
    - vytvoření vizuálního slovníku
    - vytvoření Bag of Words vektoru pro každou z trénovacích fotografií a jejich export
    - vytvoření a export klasifikátoru pro každou ze tříd
  - *Výstup:*
    - vizuální slovník
    - Bag of Words vektor pro každou z trénovacích fotografií
    - klasifikátor pro každou ze tříd
- *Fáze kategorizace*
  - *Vstup:*
    - soubor fotografií ke klasifikaci
    - vizuální slovník
    - klasifikátor pro každou ze tříd

  - *Funkce:*
    * extrahování příznaků z fotografií ke klasifikaci
    * vytvoření Bag of Words vektoru pro každou z fotografií ke klasifikaci a jejich export
    * generování výsledků klasifikace s využitím klasifikátoru
  - *Výstup:*
    * vygenerovaná příslušnost fotografií do tříd určená klasifikátorem

4.2 Návrh jednotlivých částí systému

Má aplikace se skládá ze součástí popsaných v předchozí kapitole. Nyní se budu podrobněji věnovat návrhu jednotlivých komponent systému. Obecný návrh mé aplikace vychází z článku [24]. Vzájemná interakce jednotlivých částí systému je podrobněji znázorněná na obrázku 4.1.

4.2.1 Extrakce obrazových příznaků

Jako první se detekují klíčové body. Pro jejich nalezení je zvolen variabilní detektor. Je možné si vybrat, kterým algoritmem se budou klíčové body vyhledávat. Možnost je zvolit z těchto variant: FAST, STAR, SIFT, SURF, ORB, BRISK, MSER, GFTT, Dense, SimpleBlob.

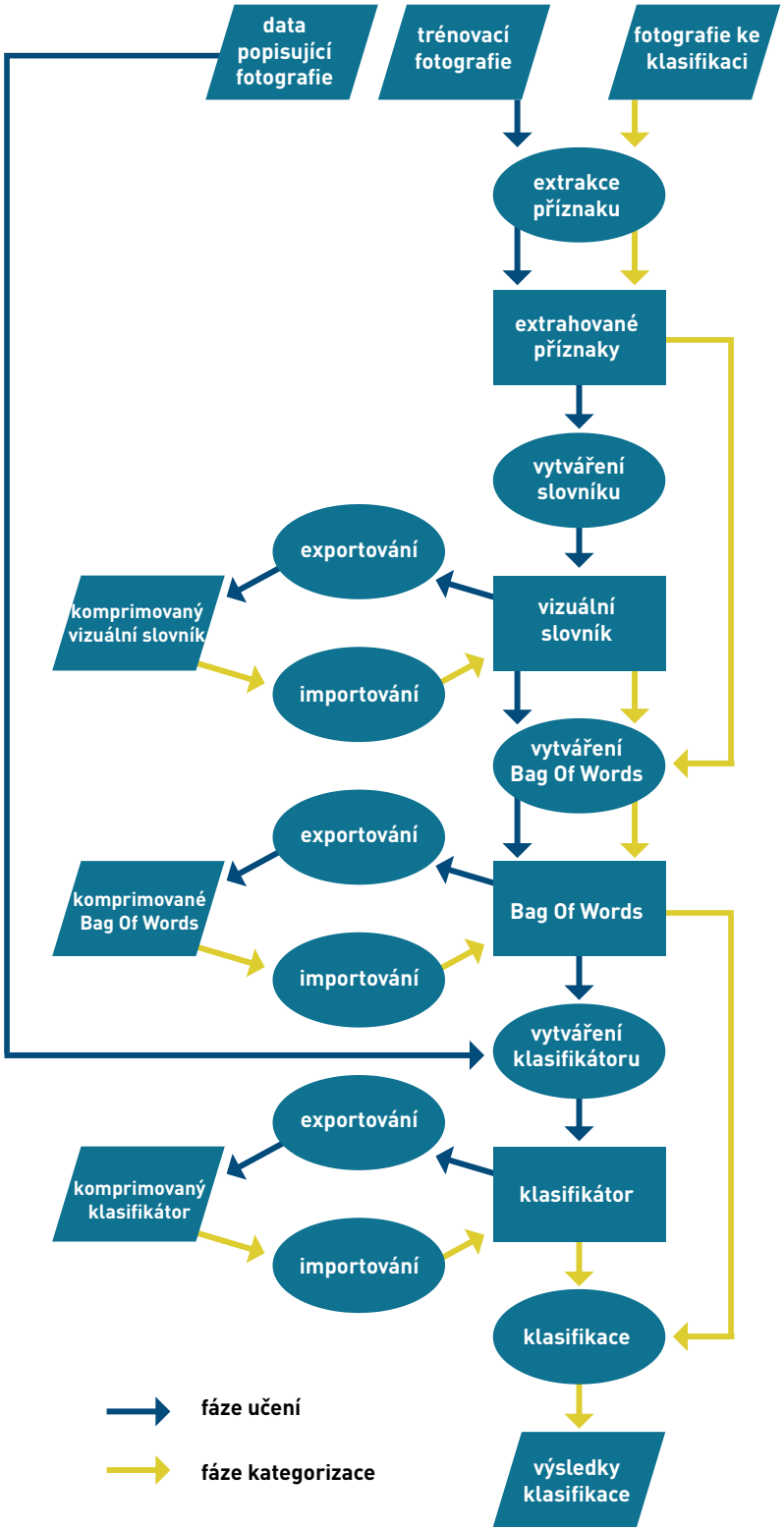
Následuje převedení těchto bodů na deskriptory. I zde je možné vybrat jednu ze škály možných algoritmů: SIFT, OpponentSIFT, SURF, OpponentSURF.

- *Vstup:* soubor trénovacích fotografií
- *Výstup:* soubor deskriptorů

4.2.2 Tvorba vizuálního slovníku

Po extrahovaní deskriptorů ze všech trénovacích fotografií následuje vytváření vizuálního slovníku. Vizuální slovník je soubor vizuálních slov, což jsou k-dimenzionální vektory. Tyto vektory vzniknou shlukováním deskriptorů metodou k-means++. Při vytváření slovníku je mimo použitou metodu shlukování nejdůležitějším parametrem jeho délka. Tato hodnota je volitelná při spuštění aplikace. Po vytvoření vizuálního slovníku bude uložen do souboru, aby bylo možné ho využít při dalším spuštění.

- *Vstup:* soubor deskriptorů
- *Výstup:* vizuální slovník



Obrázek 4.1: Vývojový diagram systému

4.2.3 Tvorba Bag of Words

Tato část systému slouží k výpočtu Bag of Words předložené sadě fotografií. Zpracovává fotografie postupně. U každé extrahuje příznaky a ty poté na základě podobnosti porovná se slovy ve vizuálním slovníku. K tomu lze využít hned několik algoritmů (matcherů), které moje aplikace podporuje: BruteForce, BruteForce-L1, FlannBased. Výsledný Bag of Words popis bude uložen na disk.

- *Vstup:* vizuální slovník
- *Výstup:* popis fotografií pomocí Bag of Words

4.2.4 Vytváření klasifikátoru

Provede se trénování SVM klasifikátoru pro každou ze tříd. Jedním z důležitých parametrů je typ jádra. Ten je volitelný při spuštění aplikace. Možné varianty jsou lineární, polynomické, sigmoid a RBF jádro.

- *Vstup:* informace o příslušnosti trénovacích fotografií do jednotlivých tříd, Bag of Words reprezentaci trénovacích fotografií
- *Výstup:* SVM klasifikátor pro každou ze tříd

4.2.5 Klasifikace

Klasifikátor vygeneruje pro každou fotografii určenou ke klasifikaci hodnotu, která znázorňuje vzdálenost od vymezeného prostoru (margin) kolem nadroviny. Tyto výsledky klasifikace jsou poté uloženy do souboru.

- *Vstup:* klasifikátory jednotlivých tříd, Bag of Words reprezentace fotografií ke kategorizaci.
- *Výstup:* příslušnost jednotlivých fotografií ke třídám stanovená klasifikátorem

4.3 Návrh testování

Tato kapitola popisuje zvolenou datovou sadu a důvody, proč byla zvolena právě tato. Dále se zabývá volitelnými proměnnými, které mají vliv na výsledek klasifikace, a principy na základě kterých jsou výsledky porovnávány.

4.3.1 Použitá datová sada

Je hned několik volně přístupných datových sad vhodných pro kategorizaci. Pro příklad zde které uvedu.

- *CIFAR-10 a CIFAR-100* - Jde o datovou sadu barevných obrázků o velikosti 32 x 32 pixelů. Obsahuje 10 a v případě CIFAR-100 100 tříd. Tato sada pro mou práci nebyla využita z důvodu malého rozlišení jednotlivých obrázků. Můj program má být zaměřený na zpracování klasických fotografií a proto je vhodné zvolit datovou sadu s větším formátem jednotlivých obrázků. [16]

- *NORB* - Datová sada NORB [14] obsahuje 29160 obrázků 50 objektů, které patří do 5 obecných kategorií. Tyto fotografie byly pořízeny dvěma kamerami za různých světelných podmínek, v různých nadmořských výškách a pod různými úhly. Je to vhodná sada pro experimentování s rozpoznávání 3D objektů, ale pro naše účely je zachyceno na fotografiích příliš málo rozličných objektů.

- *Caltech 101 a Caltech 256* - Jedná se o datové sady fotografií z 101 respektive 256 kategorií. V jedné kategorii je 40 až 900 snímků. Velikost jednotlivých snímků je zhruba 300 x 200 pixelů. Důvod, proč jedna z těchto sad nebyla vybrána, je menší variabilita ve velikostech objektů a jejich natočení. [5]

- *VOC challenge* - Tato datová sada obsahuje pro rok 2012 11540 fotografií s 31561 objektů. Fotografie jsou o rozměrech zhruba 500 x 300 pixelů. Na fotografiích jsou zachycené snímky z velkým množstvím variant objektů, natočení. Snímky také nejsou upravované jako tomu bylo u Caltech 101 a Caltech 256. Obsahuje následujících 20 tříd: letadlo, jízdní kolo, pták, loď, láhev, autobus, auto, kočka, židle, kráva, jídelní stůl, pes, kůň, motocykl, osoba, květina v květináči, ovce, pohovka, vlak, TV/monitor. Spolu s fotografiemi patří k sadě anotace jejich příslušnosti do tříd a označení, která data patří do trénovací skupiny a která jsou určeny ke kategorizaci. Tato sada byla vybrána z následujících důvodů: obsahuje dostatečný počet fotografií a tříd, třídy obsahují fotografie velkou škálu objektů, její fotografie jsou dostatečně velké a obsahují reálné fotografie, má vytvořenou kvalitní anotaci. [10]

4.3.2 Prostředky pro porovnávání výsledků

V popisu návrhu systému je uvedena řada proměnných, které dokáží ovlivnit výsledek kategorizace. Zde je uveden výčet těch, které je možné zvolit při spuštění aplikace, spolu s výčtem možných variant.

- *Typ detektoru* - FAST, STAR, SIFT, SURF, ORB, BRISK, MSER, GFTT, Dense, SimpleBlob.

- *Typ deskriptoru* - SIFT, OpponentSIFT, SURF, OpponentSURF.

- *Typ matcheru* - BruteForce, BruteForce-L1, FlannBased.

- *Délka vizuálního slovníku*

- *Omezení použité paměti při výpočtu deskriptorů pro vytvoření vizuálního slovníku*

- *SVM jádro* - lineární, polynomické, sigmoid a RBF.

Při tolika kombinacích možných nadstavení je potřeba zvolit nějakou normu pro porovnávání výsledků. V mé práci je použito pro porovnávání hodnoty přesnosti (precission) a odezvy (recall). Jejich výpočet je popsán vzorci 4.1 a 4.2, kde *Pravdivě pozitivní výsledky* jsou ty, které klasifikátor označil, že patří do dané třídy, a opravdu tam i patří, *Falešně pozitivní výsledky* jsou ty, které klasifikátor označil, jako patřící do třídy, ale ony tam nepatří. A *Falešně negativní výsledky* jsou ty, které klasifikátor označil, že do třídy nepatří, ale ony tam ve skutečnosti patří. Tyto hodnoty jsou počítány v průběhu klasifikace postupně s každou přidanou fotografií. Následně je jejich závislost vepsána do precision/recall křivky, kde je každý bod sestaven z hodnoty recall na ose x a precision na ose y. Obsah pod touto

křivkou udává průměrnou přesnost (angl. Average precision). Tuto hodnotu používám pro porovnání jednotlivých výsledků klasifikace. [8]

$$precision = \frac{Pravdivě\ pozitivní\ výsledky}{Pravdivě\ pozitivní\ výsledky + Falešně\ pozitivní\ výsledky} \tag{4.1}$$

$$recall = \frac{Pravdivě\ pozitivní\ výsledky}{Pravdivě\ pozitivní\ výsledky + Falešně\ negativní\ výsledky} \tag{4.2}$$

Kapitola 5

Implementace

Implementace aplikace vychází z návrhu popsaného v předchozí kapitole. Pro fázi trénování i klasifikaci slouží jeden program. Pro implementaci jsem si vybral jazyk C++. Tato volba je závislá na druhu použité knihovny OpenCV, které je implementovaná v jazycích C++, C, Python a Java. C++ je podle mého nejhodnější volba s ohledem na rychlost aplikace a výhody objektového programování.

5.1 Použité knihovny

Při vytváření aplikace jsem použil knihovnu OpenCV (Open source Computer Vision). Je to svobodná a otevřená multiplatformní knihovna pro manipulaci s obrazem. Je zaměřena především na počítačové vidění a zpracování obrazu v reálném čase. Původně byla vyvinuta firmou Intel Comporation a nyní je podporována laboratoří Willow Garage a Itseez. Knihovna lze použít na platformách Windows, Android, Maemo, FreeBSD, OpenBSD, iOS, BlackBerry 10, Linux a OS X.

5.2 Implementace jednotlivých částí

V této kapitole je popsána implementace jednotlivých částí. Uvedené třídy a funkce jsou součástí výše uvedené knihovny OpenCV.

5.2.1 Detekce a extrakce příznaků

Na začátku se detekují obrazové příznaky a vytvoří deskriptory. K tomu využívám v mé aplikaci prostředky knihovny OpenCV. Jednotlivé fotografie se načtou do matice (třída `Mat`) funkcí `imread()`.

Detektor reprezentuje třída `FeatureDetector`. Pro získání příznaků je použitá funkce detect patřící do této třídy. Klíčové body jsou reprezentovaná datovou strukturou `KeyPoint`. Deskriptor reprezentuje třída `DescriptorExtractor`. Pro vytvoření deskriptorů využívá aplikace funkce `compute()` z této třídy. Deskriptory z fotografie jsou uloženy do matice. Každý řádek matice reprezentuje jeden deskriptor. Pro detektor i deskriptor je možné použít algoritmy uvedené v návrhu aplikace.

5.2.2 Vizuální slovník a Bag of Words reprezentace

Pokud je již vizuální slovník vytvořen pouze se načte ze souboru. Pokud ne, vytvoří se. Pro trénování vizuálního slovníku je využita třída z OpenCV `BowKMeansTrainer`. Tato třída obsahuje metodu `add()` pro přidání jednotlivých deskriptorů. Jelikož je operační paměť počítače omezená a deskriptorů mnoho, je zavedena proměnná `memoryUse` hlídající, aby velikost deskriptorů nepřesáhla určitou paměťovou kapacitu. Tuto hodnotu může uživatel zadat při startu aplikace. Pokud je při běhu aplikace zažádáno o větší kapacitu operační paměti než je dovoleno operačním systémem, program skončí chybovou hláškou. Pro shlukování deskriptorů do vizuálních slov je k dispozici metoda `cluster()` třídy `BowKMeansTrainer`. Vizuální slovník vzniká metodou k-means++ díky zvolenému parametru `KMEANS_PP_CENTERS`. Dalším volitelným parametrem při definování trenéra vizuálního slovníku je jeho délka. Tu lze také zvolit při startu aplikace. Vizuální slovní se uloží do souboru `vocabulary.xml` a komprimuje.

Pro trénování klasifikátoru je nutné získat Bag of Words reprezentaci jednotlivých trénovacích fotografií. Pokud je to možné, jsou načteny ze souboru. Pokud ne, program využije třídy z knihovny OpenCV `BOWImgDescriptorExtractor`, která reprezentuje extraktor Bag of Words vektorů. Každý Bag of Words vektor je reprezentován maticí a je uložen do speciálního souboru na disk. Jeho získání proběhne metodou z této třídy `compute()`.

5.2.3 Klasifikátor

Stejně jako vizuální slovník nebo Bag of Words vektory je možné načíst za souboru také klasifikátory jednotlivých tříd. Pokud nejsou k dispozici, vytvoří se klasifikátory třídou z knihovny OpenCV. Tato třída se jmenuje `CvSVM`. Klasifikátor vznikne použitím metody `train_auto()` z této třídy. Tato metoda dokáže sama optimalizovat některé parametry. Parametry funkce jsou optimální, když je odhad *cross-validace* zkušební chyby minimální. Jeden z volených parametrů je jádro klasifikátoru. Typy jader, které lze zvolit, byly uvedeny již v návrhu. Klasifikátor pro každou třídu je uložen do souboru na disk.

Pro klasifikaci slouží metoda `predict()`. Pomocí této funkce je stanoven odhad klasifikátoru o příslušnosti dané fotografie do konkrétní třídy. Následně se počítají hodnoty precision a recall v závislosti na dosud proběhlé klasifikaci. Výsledky klasifikace pro konkrétní fotografie jsou uloženy na disk. Pro každou skupinu se dále ukládá soubor obsahující precision/recall křivku zaznamenanou body. V tomto souboru je i hodnota průměrné přesnosti pro danou třídu.

5.3 Přehled tříd a důležitých funkcí

5.3.1 Třídy

- *ObImage* - třída reprezentující fotografii. Obsahuje její identifikační název a cestu, kde je uložena na disku.
- *VocData* - třída pro zpracování dat. Jsou zde například metody pro zpracování výsledků, počítání precision/recall křivky, načítání popisných dat o fotografiích ze souborů.

5.3.2 Důležité funkce

- *trainVocabulary()* - tato funkce provádí detekci a extrakci příznaků. Poté provede trénování vizuálního slovníku, který uloží na disk. Návratová hodnota funkce je matice obsahující slovník.
- *trainSVMClassifier()* - vytvoří Bag of Words reprezentaci trénovacích fotografií. Následně vytrénuje klasifikátor pro konkrétní třídu. Bag of Words vektory i klasifikátor uloží na disk.
- *computeConfidences()* - provádí klasifikaci fotografií pro konkrétní třídu. Před klasifikací vytvoří Bag of Words vektory pro klasifikované fotografie a uloží je na disk. Stejně tak uloží i výsledky klasifikace.
- *computeGnuPlotOutput()* - vypočítá precision/recall křivku a hodnotu průměrné přesnosti. Tyto výsledky uloží na disk.

5.4 Rozhraní aplikace

Pro aplikaci jsem zvolil jednoduché rozhraní příkazové řádky. Jednotlivé kombinace nadstavení klasifikace se zadávají jako parametry při startu aplikace. Aplikace se spouští dle uvedeného vzoru:

```
.\categorization [Přepínače]

Možné přepínače jsou následující:

-i cesta              - cesta ke složce s daty fotografií VOC
-o cesta              - cesta k uložení výstupu aplikace
-v počet slov         - počet slov vizuálního slovníku
-d typ detektoru      - typ zvoleného detektoru klíčových bodů
-s typ deskriptoru    - typ algoritmu pro vytváření deskriptorů
-m typ matcheru       - typ algoritmu pro budování Bag of Words
-l limit paměti       - omezení paměti při tvorbě deskriptorů v MB
-k jádro SVM          - typ jádra SVM klasifikátoru
-h                    - výpis nápovědy
```

Jediný povinný parametr je cesta ke složce s daty fotografií VOC. Zbývající parametry, pokud nejsou nastaveny, použijí následující nastavení: Pro uložení výstupu se použije aktuální adresář. Typ detektoru bude SURF, typ deskriptoru OpponentSURF a typ matcheru BruteForce-L1. Počet vizuálních slov ve slovníku bude 1000. Limit paměti nebude omezen a jádro SVM bude RBF.

Kapitola 6

Dosažené výsledky práce

Jedním z výsledků této práce je aplikace. Cílem nebylo vytvoření aplikace konkurující vynikajícím systémům, které jsou možné v dnešní době vytvořit. Cílem bylo sestavit systém schopný kategorizace, seznámení se základním konceptem kategorizace fotografií a pochopení vlivu jednotlivých faktorů na její výsledek. V této kapitole jsou uvedeny výsledky testů, která byly s výslednou aplikací provedl. Je porovnáváno použití jednotlivých algoritmů na konečný výsledek kategorizace.

6.1 Výsledky testů

V této kapitole jsou uvedeny výsledky testů, které byly s výslednou aplikací provedl. Testy byly provedeny na notebooku Lenovo Z500 s procesorem Intel Core i5-3230M CPU @ 2.60GHz x 4, operační pamětí 8GB s operačním systému Ubuntu 14.04 LTS. Pro porovnání výsledků klasifikace je použita hodnota průměrné přesnosti. Tato hodnota je vypočtena pro každou kategorii a následně je proveden průměr těchto hodnot.

6.1.1 Detekce příznaků

Při získávání příznaků je důležitým parametrem druh detektoru a deskriptoru. V testu jsou jejich druhy zaměňovány. Výsledky jsou uvedeny v tabulce 6.3. Pro tento test jsem zvolil ostatní parametry následovně:

- *Délka vizuálního slovníku* - 1000
- *Omezení paměti (MB)* - 500
- *Matcher* - BruteForce-L1
- *Jádro SVM* - RBF

Detektor	Deskriptor	Průměrná úspěšnost[%]
SIFT	SIFT	18,94
SURF	SURF	20,57
SIFT	OpponentSIFT	19,59
SURF	OpponentSURF	22,93
Dense	SURF	14,58
FAST	OpponentSURF	20,12

Tabulka 6.1: Tabulka výsledků testování detektorů a deskriptorů.

Z tabulky je zřejmé, že jako nejlepší detektor se osvědčil algoritmus SURF. Z výsledku deskriptorů lze odvodit, že OpponentSIFR a OpponentSURF dopadli při klasifikaci lépe než základní metody SIFT a SURF. Je to díky přidané informaci o barvě na základě tzv opponent copor [23]. Nejlépe z testovaných možností se osvědčila kombinace detektoru SURF a deskriptoru OpponentSURF.

6.1.2 Vytváření vizuálního slovníku

Parametry, které jsou v mé aplikaci volitelné a ovlivňují vlastnosti vizuálního slovníku, jsou jeho délka a omezení paměti při vytváření deskriptorů. Omezení paměti se neváže na žádný z algoritmů a technik, které jsou předmětem této práce, a proto jeho vliv na výsledek nejsou zkoumány. Výsledky jsou uvedeny v tabulce 6.2. Pro tento test jsem zvolil ostatní parametry následovně:

- *Detektor* - SURF
- *Deskriptor* - SURF
- *Omezení paměti (MB)* - 500
- *Matcher* - BruteForce
- *Jádro SVM* - RBF

Délka slovníku	Průměrná úspěšnost[%]
500	19,52
1000	21,43
2000	18,21
5000	16,55

Tabulka 6.2: Tabulka výsledků testování vizuálního slovníku.

Nejlepší úspěšnosti kategorizace bylo docíleno při slovníku o 1000 slovech. U tohoto výsledku bylo docíleno nejefektivnější délky slovníku. Z výsledků bylo vypozorováno, že u slovníků s více slovy jsou následně vytrénované Bag of Words jednotlivých fotografií řidší.

6.1.3 Vytváření Bag of Words

Při trénování Bag of Words vektorů se využívá tzv. deskriptor matcher. Jde o metodu sloužící k porovnávání deskriptorů. Jeho vliv na klasifikaci je uveden v tabulce. Výsledky jsou uvedeny v tabulce 6.3. Pro tento test jsem zvolil ostatní parametry následovně:

- *Detektor* - SURF
- *Deskriptor* - OpponentSURF
- *Délka vizuálního slovníku* - 1000
- *Omezení paměti (MB)* - 500
- *Jádro SVM* - RBF

Deskriptor matcher	**Průměrná úspěšnost[%]**
BruteForce	21,33
BruteForce-L1	22,93
FlannBased	21.43

Tabulka 6.3: Tabulka výsledků testování vytváření Bag of Words vektorů.

Z výsledků je zřejmé, že v tomto testu se nejvíce osvědčil BruteForce-L1. Tento algoritmus dokáže nejlépe pro naši datovou sadu a ostatní přednastavení určit, které vizuální slova se ve fotografii nachází.

6.1.4 Klasifikátor

U SVM klasifikátoru budu zkoumat vliv zvoleného jádra a výsledek klasifikace. Výsledky jsou uvedeny v tabulce 6.4. Pro tento test jsem zvolil ostatní parametry následovně:

- *Detektor* - SURF
- *Deskriptor* - OpponentSURF
- *Matcher* - BruteForce-L1
- *Délka vizuálního slovníku* - 1000
- *Omezení paměti (MB)* - 500

Druh jádra	**Průměrná úspěšnost[%]**
Lineární	21,75
Polynomické	20.50
Sigmoida	7.55
RBF	22.93

Tabulka 6.4: Tabulka výsledků testování jednotlivých jader klasifikátoru.

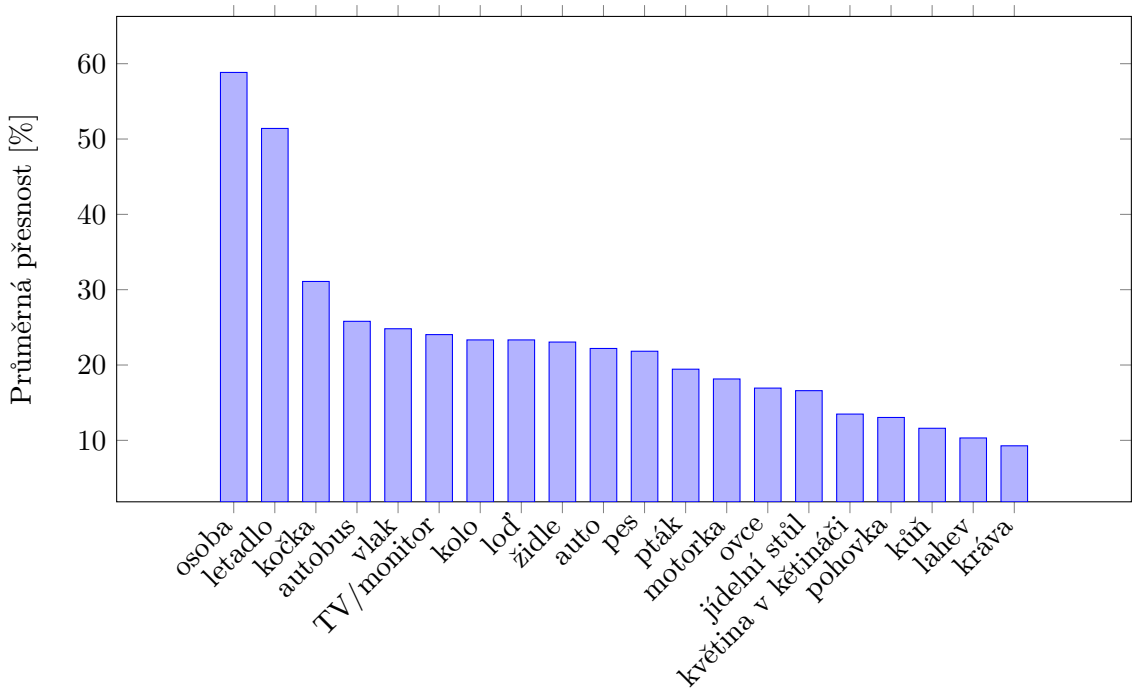
Z tohoto testu vyšlo jako nejlepší jádro SVM pro naši klasifikaci RBF. U tohoto jádra se povedlo nejlépe rozdělit datový prostor nadrovinou.

6.1.5 Nejlepší dosažený výsledek

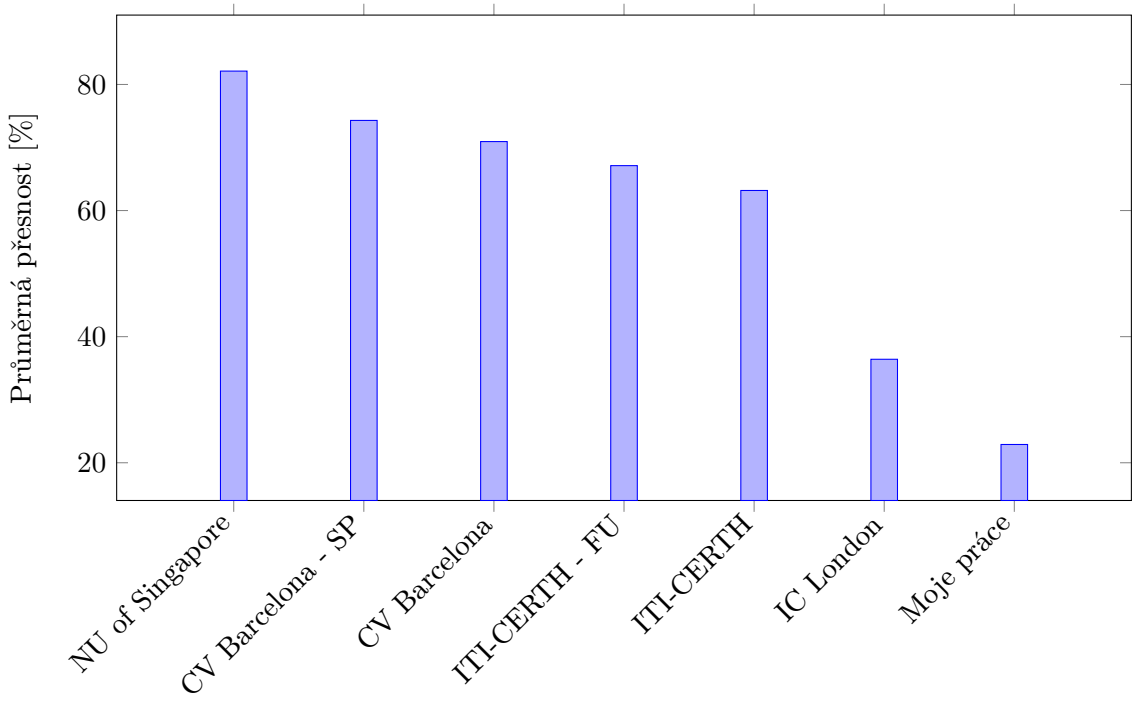
U nejlepšího výsledku bylo dosaženo **22.9 %** průměrné přesnosti. Histogram s výsledky klasifikátorů jednotlivých tříd je uveden o něco níže (6.1). Tohoto výsledku bylo dosaženo při tomto nadstavení:

- *Detektor* - SURF
- *Deskriptor* - OpponentSURF
- *Matcher* - BruteForce-L1
- *Délka vizuálního slovníku* - 1000
- *Omezení paměti (MB)* - 500
- *Jádro SVM* - RBF

Porovnání tohoto výsledku s výsledky soutěže The Pascal VOC Challenge je níže v histogramu 6.2. Výsledek mé práce je ze zúčastněných soutěže nejhorší. To není nijak překvapivé, když uvážíme, že jsem použil jen jeden z klasických systémů kategorizace založený na Bag of Words reprezentaci a SVM klasifikátoru bez moderních pokrokových metod. Nejlepšího výsledku v soutěži dosáhla National University of Singapore. Algoritmus vítězné skupiny funguje na podobné konstrukci jako má práce, ale využívá mimo Bag of Words a SVM a také jiné pokročilejší algoritmy, jako je například reprezentace fotografií za pomocí tzv. Spatial Pyramid. Tato technika je ještě vylepšená díky plovoucímu oknu detekujícího tzv. confidence maps. Více o této metodě najdete zde [31].



Obrázek 6.1: Histogram průměrných přesností jednotlivých kategorií u nejlepšího výsledku.



Obrázek 6.2: Histogram porovnání výsledků mého klasifikátoru s výsledky soutěže The Pascal VOC Challenge 2012. Data převzata z [10].

Kapitola 7

Závěr

Cílem této práce bylo seznámit se s principy kategorizace fotografií podle objektů, které obsahují, pochopit, jak fungují jednotlivé využívané principy, na základě nasbíraných informací sestavit program a s jeho pomocí pochopit vliv jednotlivých kombinací zvolených algoritmů a proměnných na výsledek klasifikace.

Při návrhu systému jsem vycházel z klasické koncepce klasifikace s využitím extrakce příznaků, vytvořením vizuálního slovníku, reprezentace fotografií pomocí koncepce Bag of Words a konečnou klasifikací pomocí Support Vector Machines. Pro detekci, vytvoření vektorů deskriptorů a použití deskriptor matchuru jsem využil variability knihovny OpenCV a dal uživateli prostor pro experimentování s vlivem jednotlivých algoritmů na výsledek kategorizace. Stejně tak je možno si zvolit délku vizuálního slovníku a jádro klasifikátoru SVM.

S tímto systémem jsem následně prováděl experimenty a sledoval vliv jednotlivých zvolených algoritmů na výsledek klasifikace. Nejlepšího výsledku jsem docílil použitím detektoru SURF, deskriptoru OpponentSURF, deskriptor matcheru, BruteForce-L1, 1000 řádkovým vizuálním slovníkem a RBF jádrem klasifikátoru. Průměrná přesnost v tomto případě dosáhla 22,9 %.

Tento výsledek je nekonkurenceschopným ostatním výsledkům ze soutěže The Pascal VOC Callenge. Příčinou je nevyužití moderních přístupů, které jsou již dnes standardní.

Rozšíření práce by mohlo obsahovat některý z inovativních přístupů, které v práci zmiňuji. Obzvlášť použití konvolučních neuronových sítí se již vdnešních rozpoznávacích systémech stalo standardem. Dalším možným rozšířením by mohla být například metoda Spatial Pyramid Matching využitá výhercem soutěže The Pascal VOC Challenge 2012.

Literatura

[1] ABBYY COMPANY: ABBYY FineReader. [online], [cit. 2015-13-4].
URL http://www.abbyy.com/finereader/

[2] AGARWAL, S.; YADAV, S.; SINGH, K.: Notice of Violation of IEEE Publication Principles K-means versus k-means clustering technique. In *Engineering and Systems (SCES), 2012 Students Conference on*, March 2012, s. 1–6, doi:10.1109/SCES.2012.6199061.

[3] ARTHUR, D.; VASSILVITSKII, S.: K-means++: The Advantages of Careful Seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007, ISBN 978-0-898716-24-5, s. 1027–1035.
URL http://dl.acm.org/citation.cfm?id=1283383.1283494

[4] BAY, H.; ESS, A.; TUYTELAARS, T.; aj.: Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, ročník 110, č. 3, 2008: s. 346 – 359, ISSN 1077-3142, doi:http://dx.doi.org/10.1016/j.cviu.2007.09.014, similarity Matching in Computer Vision and Multimedia.
URL http://www.sciencedirect.com/science/article/pii/S1077314207001555

[5] COMPUTATIONAL VISION AT CALTECH: Caltech 101. [online], 2006 [cit. 2015-16-4].
URL http://www.vision.caltech.edu/Image_Datasets/Caltech101/

[6] CRUSCH BASE: PittPatt. [online], [cit. 2015-21-4].
URL https://www.crunchbase.com/organization/pittpatt

[7] CSURKA, G.; DANCE, C.; FAN, L.; aj.: Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, ročník 1, Prague, 2004 [cit. 2015-3-2], s. 1–2.

[8] DAVIS, J.; GOADRICH, M.: The Relationship Between Precision-Recall and ROC Curves. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, New York, NY, USA: ACM, 2006, ISBN 1-59593-383-2, s. 233–240, doi:10.1145/1143844.1143874.
URL http://doi.acm.org.ezproxy.lib.vutbr.cz/10.1145/1143844.1143874

[9] EVANS, C.: Notes on the OpenSURF Library. , č. CSTR-09-001, January 2009 [cit. 2015-12-2].
URL http://www.chrisevansdev.com

[10] EVERINGHAM, M.; ESLAMI, S.; VAN GOOL, L.; aj.: The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*, ročník 111, č. 1, 2015: s. 98–136, ISSN 0920-5691, doi:10.1007/s11263-014-0733-5.
URL http://dx.doi.org/10.1007/s11263-014-0733-5

[11] FILLIAT, D.: A visual bag of words method for interactive qualitative localization and mapping. In *Robotics and Automation, 2007 IEEE International Conference on*, April 2007, ISSN 1050-4729, s. 3921–3926, doi:10.1109/ROBOT.2007.364080.

[12] GITI, M.: Implementing The K-Means Clustering Algorithm in C#.NET. [online], 2015 [cit. 2015-26-4].
URL http://www.codeproject.com/Articles/985824/Implementing-The-K-Means-Clustering-Algorithm-in-C

[13] HEARST, M.; DUMAIS, S.; OSMAN, E.; aj.: Support vector machines. Jul 1998, ISSN 1094-7167, s. 18–28, doi:10.1109/5254.708428.

[14] HUANG, F. J.; LECUN, Y.: THE NORB DATASET, V1.0. [online], 2004 [cit. 2015-16-4].
URL www.cs.nyu.edu/~ylclab/data/norb-v1.0/

[15] KANAN, C.; COTTRELL, G.: Robust classification of objects, faces, and flowers using natural image statistics. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, June 2010, ISSN 1063-6919, s. 2472–2479, doi:10.1109/CVPR.2010.5539947.

[16] KRIZHEVSKY, A.: The CIFAR-10 dataset. [online], [cit. 2015-16-4].
URL http://www.cs.toronto.edu/~kriz/cifar.html

[17] LEVI, G.: Bag of Words Models for visual categorization. [online], 2013 [cit. 2015-28-4].
URL https://gilscvblog.wordpress.com/2013/08/23/bag-of-words-models-for-visual-categorization/

[18] LOWE, D. G.: Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, ročník 60, č. 2, 2004: s. 91–110, ISSN 0920-5691, doi:10.1023/B:VISI.0000029664.99615.94.
URL http://dx.doi.org/10.1023/B%3AVISI.0000029664.99615.94

[19] MEYER, D.; VIEN, F. T.: Support vector machines. *Google scholar*, 2014 [cit. 2015-3-2].
URL https://scholar.google.cz/scholar?q=Support+vector+machines+Meyer%2C+David+Wien&btnG=&hl=cs&as_sdt=0%2C5

[20] PARK, S. B.; LEE, J. W.; KIM, S. K.: Content-based image classification using a neural network. *Pattern Recognition Letters*, ročník 25, č. 3, 2004: s. 287 – 300, ISSN 0167-8655, doi:http://dx.doi.org/10.1016/j.patrec.2003.10.015.
URL http://www.sciencedirect.com/science/article/pii/S0167865503002253

[21] POLZER, J.: Adobe Photoshop Elements 10. [online], 11 2011 [cit. 2015-20-4].
URL https://www.maxiorel.cz/adobe-photoshop-elements-10-sam-rozpozna-najde-hledane-objekty-ve-vasi-sbirce-fote

[22] SAHINER, B.; CHAN, H.-p.; PETRICK, N.; aj.: Classification of mass and normal breast tissue: a convolution neural network classifier with spatial domain and texture images. *Medical Imaging, IEEE Transactions on*, ročník 15, č. 5, Oct 1996: s. 598–610, ISSN 0278-0062, doi:10.1109/42.538937.

[23] SCHWARZ, M. W.; COWAN, W. B.; BEATTY, J. C.: An Experimental Comparison of RGB, YIQ, LAB, HSV, and Opponent Color Models. *ACM Trans. Graph.*, ročník 6, č. 2, Duben 1987: s. 123–158, ISSN 0730-0301, doi:10.1145/31336.31338.
URL http://doi.acm.org.ezproxy.lib.vutbr.cz/10.1145/31336.31338

[24] SIVIC, J.; ZISSERMAN, A.: Video Google: a text retrieval approach to object matching in videos. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, Oct 2003 [cit. 2015-3-2], s. 1470–1477 vol.2, doi:10.1109/ICCV.2003.1238663.

[25] SIVIC, J.; ZISSERMAN, A.: Efficient Visual Search of Videos Cast as Text Retrieval. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, ročník 31, č. 4, April 2009: s. 591–606, ISSN 0162-8828, doi:10.1109/TPAMI.2008.111.

[26] ADOBE SYSTEMS SOFTWARE: Adobe Photoshop Elements 13. [online], [cit. 2015-1-5].
URL http://www.adobe.com/cz/products/photoshop-elements.html

[27] GOOGLE COMPANY: Our history in depth. [online], [cit. 2015-30-4].
URL https://www.google.com/intl/en/about/company/history/

[28] TURNER, A.: Artificial Neural Networks. [online], 2015 [cit. 2015-8-4].
URL http://andrewjamesturner.co.uk/ArtificialNeuralNetworks.php

[29] UWIMANA, E.; RUIZ, M. E.: Automatic Classification of Medical Images for Content Based Image Retrieval Systems (CBIR). *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, ročník 52, č. 12, 2008: s. 788–792, doi:10.1177/154193120805201205,
http://pro.sagepub.com/content/52/12/788.full.pdf+html.
URL http://pro.sagepub.com/content/52/12/788.abstract

[30] ČÍŽEK, J.: Photoshop Elements 8. [online], 10 2009 [cit. 2015-20-4].
URL http://www.zive.cz/clanky/photoshop-elements-8-zna-vasi-tvar/sc-3-a-149214/default.aspx

[31] YANG, J.; YU, K.; GONG, Y.; aj.: Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, June 2009, ISSN 1063-6919, s. 1794–1801, doi:10.1109/CVPR.2009.5206757.

[32] ZHAO, Q.; TAO, H.: Object tracking using color correlogram. In *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*, Oct 2005, s. 263–270, doi:10.1109/VSPETS.2005.1570924.

Příloha A

Obsah DVD

Přiložené DVD obsahuje:

- zdrojové kódy této práce
- zdrojové kódy použité použité knihovny OpenCV umožňující překlad
- binární spustitelnou verzi výsledné aplikace spolu se sdílenými knihovnami
- použitá trénovací a testovací sada fotografií
- vygenerované data aplikací při testování
- soubor readme s popisem použití
- skript demonstrující činnost programu
- PDF a LaTeXverzi této práce
- plakát reprezentující použité metody a výsledky

Příloha B

Plakat