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

POKROČILÁ DETEKCE LIDSKÉ TVÁŘE

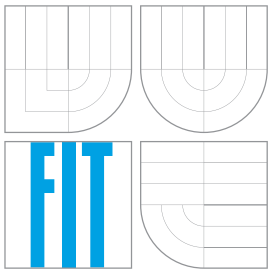
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

IGOR KONÍČEK

BRNO 2013

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

POKROČILÁ DETEKCE LIDSKÉ TVÁŘE

ADVANCED DETECTION OF HUMAN FACE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — IGOR KONÍČEK
AUTHOR

VEDOUCÍ PRÁCE — Doc. Ing. ADAM HEROUT, Ph.D.
SUPERVISOR

BRNO 2013

Abstrakt

K detekci, lokalizaci a určení úhlů natočení tváře bylo dříve přistupováno k jako třem rozlišným úkolům. Tato práce se zabývá algoritmy, které tyto problémy sjednocuje do jednoho algoritmu. K tomu využívá technologie histogramů orientovaných gradientů, které jsou spojené do stromové struktury. První část práce stručně popisuje používané metody k detekci tváře. Cílem práce je experimentovat s danými algoritmy a zjistit jejich kvalitu pro rozpoznávání obličejů a následně brýlí.

Abstract

With detection, localization and pose estimation of face was used to deal as three different tasks. This thesis works with algorithms which unify this tasks into one algorithm. To solve this problem histograms of oriented gradients connected in tree based structure are used. First part of work briefly informs about methods used for face detection. Aim of this work is to experiment with given algorithms and discover their quality for face detection and glasses detection.

Klíčová slova

Detekce tváře, Lokalizace tváře, Určení úhlu natočení tváře, Detekce brýlí, Histogramy orientovaných gradientů, Haarovy příznaky, Integrální obraz

Keywords

Face detection, Face localization, Pose estimation of face, Glasses detection, Histograms of oriented gradients, Haar features, Integral Image

Citace

Igor Koníček: Pokročilá detekce lidské tváře, bakalářská práce, Brno, FIT VUT v Brně, 2013

Pokročilá detekce lidské tváře

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana doc. Ing. Adama Herouta, Ph.D. a uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Igor Koníček
13. května 2013

Poděkování

Chtěl bych poděkovat mému vedoucím doc Ing. Adamovi Heroutovi Ph.D za cenné připomínky při řešení této práce

© Igor Koníček, 2013.
Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

Kapitola 1

Úvod

V posledních letech jsme čím dál více obklopeni digitální technikou, mobily s přístupem na internet a hlavně sociálními sítěmi. Tyto webové aplikace se postupně staly nedílnou součástí lidského života. Člověk si od nepaměti snaží svůj život co nejvíce zjednodušovat a v tom nám pomáhají i tyto sítě. Pokud jim poskytneme své fotky pro zveřejnění, můžou nám být nabídnuty fotky s označenými obličeji, ke kterým poté připíšeme jméno. Takovéto sociální sítě mají k dispozici miliony gigabitů fotek, na kterých trénují své modely a klasifikátory. Tato práce se zabývá algoritmem týmu tvořený X. Zhu a D. Ramanan, jejichž práce byla zveřejněna společně se zdrojovými kódy v červenci 2012 [8].

Detekce a lokalizace obličeje je stále komplikovanou oblastí počítačového vidění. Ke zhoršení rozpoznávání obličeje také přispívá jeho natočení v dané fotografii. K jednotlivým úkolů - detekce, lokalizace a určení úhlu natočení obličeje - bylo dříve přistupováno jednotlivě. Výše zmiňovaný tým tyto rozdílné problémy sjednocuje do jediného algoritmu, který umí nejen rozpoznat obličej a určit jeho polohu, ale navíc umí určit i jeho úhel natočení. K dosažení požadovaného klasifikátoru jsou využity technologie elastických grafů a histogramu orientovaných gradientů.

Cílem této práce je dané algoritmy klasifikátoru získat a prostudovat. Dalším bodem je získání učícího algoritmu, který bude spojen s klasifikátorem, aby jeho výstupem byly modely pro klasifikaci obličejů. Jádrem práce poté bude experimentovat s tvorbou modelů a sledování následných vlivů na rozpoznávání. Důraz bude kladen na sledování změny počtu význačných bodů obličeje, dále na počtu nesprávně anotovaných vstupních dat. Následně bude sledován vliv, který přinese postupné zmenšování obličejů trénovací sady a na závěr bude vyzkoušeno, zda je možné vytvořit použitelný model pro rozpoznávání brýlí.

Čtenář je obeznámen se základní technikou extrakcí příznaků a jejich následnou klasifikací a zpracováním v kapitole 2. Ve 3. kapitole jsou popsány teoretické informace o funkci algoritmu pro současnou detekci obličeje, jeho lokalizaci a zjištění úhlu natočení. Popisu vytváření a zaměření jednotlivých modelů použitých pro jednotlivé experimenty se věnuje kapitola 4. Provádění a výsledné zhodnocení výsledků daných experimentů s modely pro rozpoznávání obličejů a brýlí je zachyceno v kapitole 5. V 6. kapitole jsou shrnuty veškeré dosažené výsledky a také je zde nastíněn další možný vývoj práce.

Kapitola 2

Metody užívané pro rozpoznávání obličejů

Důležitým úkolem v rámci počítačového vidění je přeměna vstupní obrazové informace na data. Přeměnou takovéhoto vstupu se zabývá extrakce příznaků, které spolehlivě definují daný objekt - obličej, motocykl, automobil, brýle, a pod. Spolehlivá definice je taková, že hodnota příznaků na určité třídě objektů dává přibližně stejné číselné hodnoty. Pokud ovšem této třídě předložíme jiný objekt (např. detektoru obličejů je předložen strom), poté by výsledkem klasifikátoru v takovémto případě měla být naprosto odlišná číselná hodnota - lze tedy jednoznačně určit do jaké třídy objekt spadá.

Všeobecné požadavky na příznaky lze charakterizovat následujícími body:[6]

- Spolehlivost - číselné hodnoty klasifikátorů jsou pro stejné třídy velmi podobné
- Rozlišitelnost - tento parametr souvisí se spolehlivostí. Číselné hodnoty klasifikátorů pro rozdílné třídy jsou velmi rozdílné
- Invariantnost - udává požadavek na příznaky, aby byly nezávislé na různé změny okolností (kontrast, posuv, natočení, jas, změny velikosti měřítka)
- Rychlost počítání - klade důraz, aby výpočet jednotlivých příznaků probíhal co nejrychleji. Zde ovšem může docházet k problémům s mírou (ne)spolehlivosti rychle vypočtených příznaků

Příznaky můžeme dále dělit do několika kategorií podle následujících kritérií. Pokud dochází k výpočtu jednotlivých příznaků nad celým vstupním obrazem, poté hovoříme o globálních příznacích. V kontrastu s těmito stojí příznaky lokální, tyto jsou počítány pro každou oblast v daném obraze zvlášť. Mezi další dělení patří tzv. příznaky řídké (příznak je počítán pouze pro určité body) a proti nim stojí tzv. příznaky husté (příznak je počítán pro všechny pozice v obraze). Závěrem z toho plyne, že různé typy příznaků jsou vhodné pro různé typy oblasti počítačového vidění a správná volba může ovlivnit úspěšnost celého algoritmu.

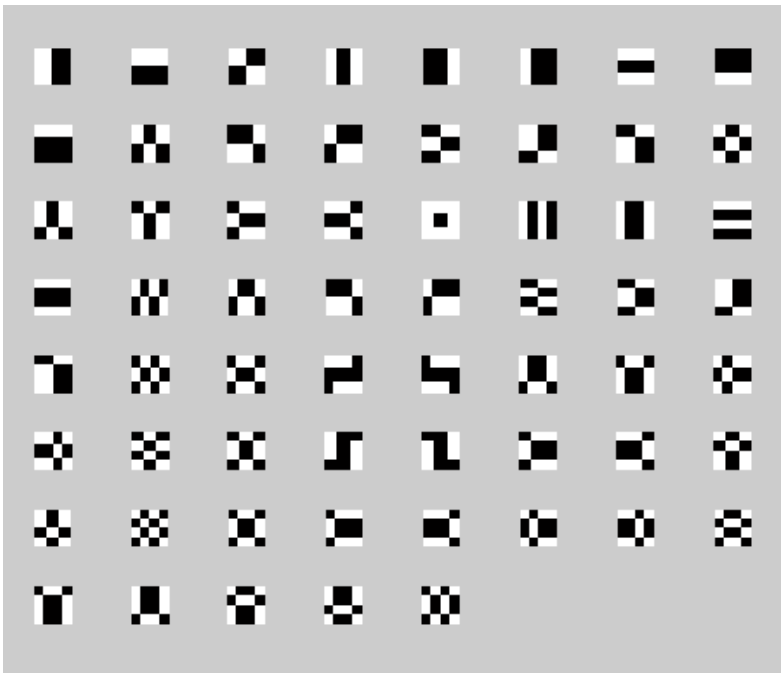
2.1 Metody extrakcí příznaků

2.1.1 Haarovy příznaky

Tyto příznaky jsou založeny na počítání rozdílného jasu v obdélníkových skenovacích oknech. Příznak je tvořen obdélníkem o dvou částech, jedna je bílá a druhá černá. Výpočet

jednotlivé hodnoty příznaku spočívá ve substrakci jednotlivých hodnot intenzit obrazu. Číselnou hodnotu lze vyjádřit pomocí rovnice 2.1, kde P_{bila} představuje sumu intenzity pixelů v bílém obdélníku. Suma intenzit černého obdélníku je reprezentována P_{cerna}

$$I = P_{bila} - P_{cerna} \tag{2.1}$$



Obrázek 2.1: Ukázka Haarových příznaků

Základem těchto příznaků je Haarova vlnka (anglicky Haar wavelet), kterou lze využít k počítání diskrétní vlnové transformace. Značnou výhodou je její rychlost, kdežto hlavní nevýhoda tkví v její nespojitosti. [2] Tato vlnka je definována následujícím předpisem:

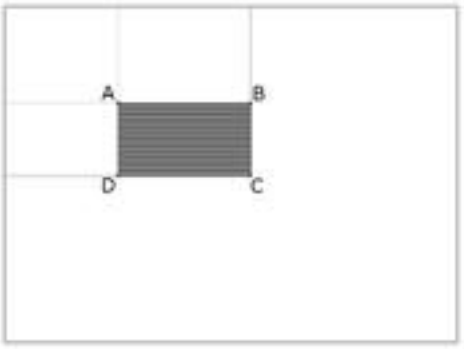
$$\psi(t) = \begin{cases} 0 & t < 0 \\ 1 & 0 \leq t < 1/2 \\ -1 & 1/2 \leq t < 1 \\ 0 & 1 \leq t \end{cases} \tag{2.2}$$

Haarovy příznaky lze dělit do kategorií dle účelu použití - například k detekci hran, či linek. Tyto příznaky lze počítat i pro obdélníková okna natočená o 45° vůči své předešlé poloze.

2.1.2 Integrální obraz

Výpočet integrálního obrazu je metoda, kdy takovýto obraz má v každém svém pixelu hodnotu součtu hodnot obrazových pixelů od začátku obrazu až do tohoto bodu. Máme-li obraz O, poté jeho hodnota integrálního obrazu I lze vyjádřit jako:

$$O(x, y) = \sum_{i=1}^{x} \sum_{j=1}^{y} I(i, j) \tag{2.3}$$



Obrázek 2.2: Ilustrace integrálního obrazu

Pokud máme k dispozici takto vypočítaný integrální obraz, výpočet libovolně velké sumy přes hodnoty původního obrazu zabere vždy přesně čtyři reference do paměti. Pro ilustraci: pokud budeme chtít vypočítat sumu hodnot pixelů celého tmavého obdélníku z obrázku 2.2 nám bude pouze stačit hodnota pixelů A, B, C a D daného integrálního obrazu. Vyčíslení sumy S_1 dosáhneme následujícím předpisem:

$$S_1 = O(A) + O(D) - O(B) - O(C)$$

Tento předpis je využitelný pro získání sumy, která je pravoúhlá - rovnoběžná s osou x. Pokud ovšem je zapotřebí získávat sumy z oblastí, které jsou oproti ose x v natočení, je třeba vypočítat nový integrální obraz. Zde dochází ke zvýšení složitosti výpočtu takového obrazu, který se kalkulován dvojím průchodem. Kde první průchod je ve směru zleva doprava a shora dolů a druhý průchod poté směrem opačným - zprava doleva a zdola nahoru. V praxi se nejčastěji užívá natočení 0° a nebo 45°. V obou případech po vzniku integrálního obrazu opět stačí pouze čtyři reference do paměti.

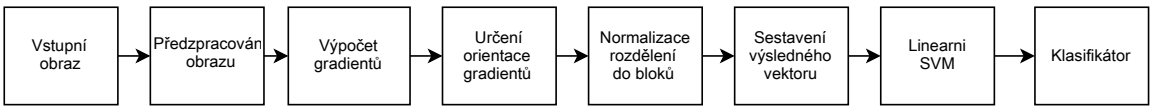
2.1.3 Histogram orientovaných gradientů

Poprvé se s histogramem orientovaných gradientů (HOG) setkáváme ve výzkumné práci týmu Dalal a Triggs [1]. Tato práce byla zaměřená na tvorbu algoritmu pro detekci chodců ve statických obrázcích a postupem času jejich práci rozšířili na detekci lidí ve videu. Dalším rozšířením byl detektor zvířat a dopravních prostředků v obrázcích.

Základní myšlenkou HOG příznaků je, že výskyt hledaného objektu a jeho tvaru může být popsán intenzitou gradientů nebo směrnicemi hran. Vytvořením těchto příznaků lze docílit rozdělením vstupního obrázku na malé, spojené oblasti - zvané buňky. A pro každý pixel uvnitř této buňky vypočítáme směrnici hrany, gradient. Pro zvýšení přesnosti lze provést optimalizaci tím, že vypočítáme hodnotu intenzity gradientů ve větší oblasti obrázku - zvané bloky. Následně napříč jednotlivými buňkami uvnitř těchto bloků provedeme normalizaci. Tato normalizace má za následek lepší odolnost příznaků vůči změnám osvětlení či stínování.

HOG příznaky vyznačují několik klíčových výhod v porovnání s jinými metodami pro popis příznaků. Jelikož HOG pracuje s lokálními buňkami v obrázcích, tím metoda podporuje odolnost vůči fotometrickým změnám (osvětlení, šum,...) a také vůči geometrickým změnám (změna měřítka, velikosti, posuv v obraze,...) s výjimkou natočení objektu. Tyto výhody se projeví ve větších prostorových oblastech. Příznaky tvořené pomocí histogramu

orientovaných gradientů je obzvláště vhodný pro detekci obličejů či lidí v obraze. Obrázek 2.3 ilustruje jak pracuje detektor založený na této metodě. [1]



Obrázek 2.3: Detektor založen na využití HOG. Schéma je převzato z [1]

Předzpracování obrazu

V této fázi u některých metod pro výpočet příznaků dochází k převodu vstupního obrázku do stupně šedi (angl. greyscale). Pro tento převod se užívá empirický vzorec:

$$G = 0.299R + 0.587G + 0.114B$$

který reflektuje, že lidské oko je nejvíce citlivé na barvu zelenou a nejméně na barvu modrou. Pokud máme takto předzpracovaný obrázek, v další fázi se používají filtry. Převážně pro odstranění šumu. Z [1] vyplývá, že se tvůrci zabývali otázkou vlivu užití převodu do stupně šedi a také jednotlivých vyhlazovacích filtrů. Autoři poté došli k závěru, že tyto úpravy mají nepatrný vliv na výkon systému. Z toho vyplývá, že pro tvorbu takového systému lze využít i barevné obrázky (např. v modelu RGB).

Výpočet gradientu

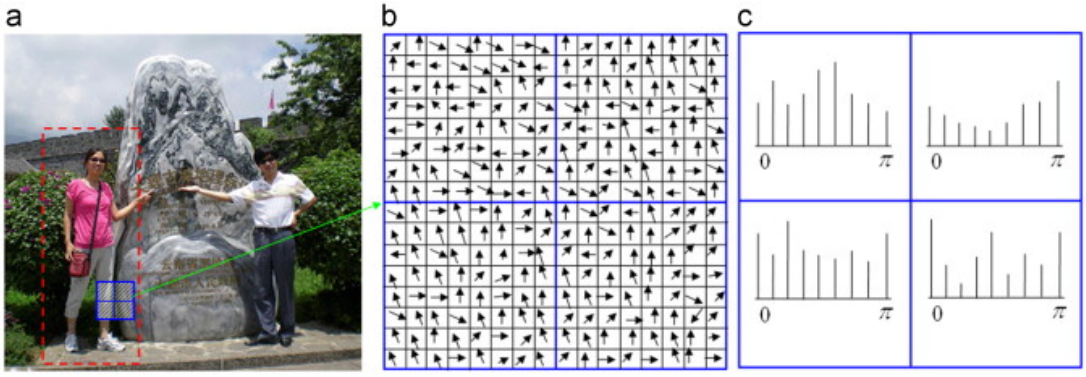
Této metodě je jako vstup dán bitmapový obrázek, tedy pouze shluk různobarevných pixelů. Pokud by na vstupu byl obrázek vektorový, celé počítání by mělo jiný směr. Z toho vyplývá, že je nutno pro každý bod v obrázku spočítat velikost a směr jeho gradientu. Ten je závislý na okolních hodnotách pixelů. Pro výpočet nejdříve autoři využili Gaussova vyhlazení a poté experimentovali s různými tvary a velikostmi derivačních masek. Jako nejlepší masku tyto testy vyhodnotili $[-1, 0, 1]$ pro dx a pro vertikální směr dy masku $[-1, 0, 1]^T$. Použití složitějších masek vedlo k degradaci celkového výkonu a užití Gaussova vyhlazení značně snížil výkon.

Určení orientace gradientů

Jakmile máme derivovaný obraz v obou směrech, aplikujeme vzorec

$$G = arctan\left(\frac{df}{dx}/\frac{df}{dy}\right)$$

výsledná velikost gradientů se poté pohybuje v rozmezí 0°-180° (lze úpravami docílit rozmezí až 0°-360°, ovšem takovéto zvětšení nepřináší znatelnější zvýšení výkonu). Takovýto rozsah je poté seskupen do N tříd. Autoři algoritmu volili pro rozsah 180° dělení po dvaceti stupních. Z toho vyplývá že vznikne 9 tříd, které se značí jako *bin*. Tímto postupem získáme charakteristiku buňky, která je znázorněna histogramem devíti hodnot. Zde jsou uloženy data o zastoupení jednotlivých gradientů pixelů. Ilustrovaný příklad výpočtu směrů gradientu je zobrazen v obrázku 2.4



Obrázek 2.4: Zde je ilustrován princip vzniku histogramu, v a) je červeně vyšrafováno skenovací okno, v němž je modře znázorněna zpracovávaná oblast. V této oblasti je pro každý pixel vypočítán jeho gradient viz b). V těchto oblastech se dělají z pravidla buňky 8×8 pixelů veliké, z nichž je udělán histogram orientovaných gradientů

Normalizace a rozdělení bloků

Seskupením jednotlivých buněk do bloků lze provést kalkulaci normalizační konstanty bloku. Touto hodnotou lze hodnoty v bloku vydělit, čímž se blok normalizuje. Cílem normalizace je zvýšit odolnost příznaků vůči šumu (změna osvětlení, kontrastu, ...). Pro tyto buňky byla zvolena optimální velikost 8×8 pixelů. Další výhodou tohoto algoritmu je částečné překrývání jednotlivých bloků, díky optimalizaci se není třeba obávat o markantní zvýšení výpočetního výkonu. Pro normalizaci lze užít $L1$ nebo $L2$ normalizaci

$$L1 \quad : \quad v \rightarrow v/\sqrt{||v||_1 + \epsilon} \tag{2.4}$$

$$L2 \quad : \quad v \rightarrow v/\sqrt{||v||_2^2 + \epsilon^2} \tag{2.5}$$

Autoři algoritmu prováděli parametrizované experimenty, kde velikost bloku byla dána předpisem $x \times x$ buněk, kde každá buňka sestává z $y \times y$ pixelů a každá buňka obsahuje z binů. Výsledkem těchto experimentů bylo zjištění, že optimální velikost bloků je 3×3 a pro velikost buňky se jeví optimální velikost osm pixelů.

Detekční okno a klasifikátor

Jako klasifikátor příznaků slouží Support Vector Machine (SVM), který z trénovací sady vytvoří model klasifikační. Jako detekční okno se použije okno o 64×128 px. Toto okno je postupně posouváno o určitý krok L a postupně je v tomto výřezu extrahován HOG příznakový vektor (descriptor). Výsledný binární klasifikátor SVM rozhodne, zda se jedná o hledaný předmět (chodec, obličej, automobil, ...) či nikoliv.

2.2 Klasifikace

Cílem klasifikačních úloh je rozřazování vstupů do tříd. Dochází zde k srovnávání vstupu s naučenou databází vzorů.

2.2.1 Support vector machine

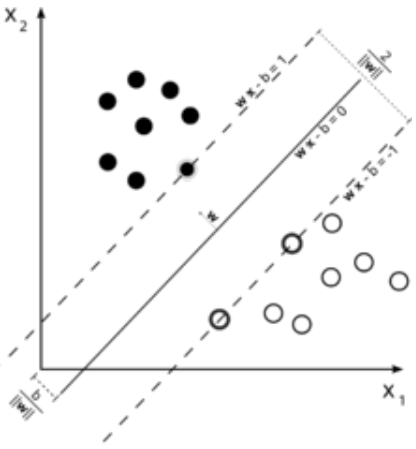
Ve strojovém učení support vector machine (SVM) se jedná o algoritmus, který spadá do kategorie učení s učitelem. Cílem metody je optimálně rozdělit trénovací data. Optimální rozdělení (přímka, rovina, nadrovina, ...) je definováno tak, že body reprezentující příznaky obou tříd leží v opačných poloprostorech a hodnota minima vzdálenosti těchto bodů od dělící roviny je maximální. Snahou je tedy maximalizovat vzdálenost dělící roviny od nejbližších bodů - situaci ilustruje obrázek 2.5. Tato metoda patří mezi binární klasifikátory - rozděluje vstupní data do dvou tříd. Dělící rovina je funkcí lineární. [10]

Významnou vlastností SVM je tzv. jádrová transformace (angl. kernel transformation). Této techniky se využívá pokud vstupní body příznaků nelze separovat lineárně ve stejné dimenzi. Výsledkem této transformace je převod příznaků do vyšší dimenze, kde už je větší šance na úspěšné rozdělení tříd.

Jak již bylo zmíněno výše, SVM je binární klasifikátor. Pokud ovšem je třeba řešit problém o více než dvou třídách, lze použít hierarchickou posloupnost takovýchto klasifikátorů, následující typy vychází z [5]:

- **Jeden proti všem** (angl. One-against-all) předpokládá problém sestávající z N tříd, kde $N > 2$. Je vytvořeno N binárních klasifikátorů. Nechť i popisuje jednu třídu, kde $i = 0..N$. Poté tato třída i je označena jako pozitivní sada a ostatní třídy jsou označeny negativně. Poté probíhá rozpoznávací fáze, kde testovací sada je předložena každému z N SVM klasifikátoru. Ta s nejlepším ohodnocením je vybrána.

- **Jeden proti jednomu** (angl. One-against-one) - základem této metody je vytvoření klasifikátoru, který je natrénován pouze na dvou třídách. Vznikne tedy $N(N-1)/2$ nových klasifikátorů - každá kombinace dvou tříd má svůj klasifikátor. Výhodou je menší nárok na vstupní data. Nevýhodou je velký počet takto vzniklých klasifikátorů a s tím související početní náročnost.



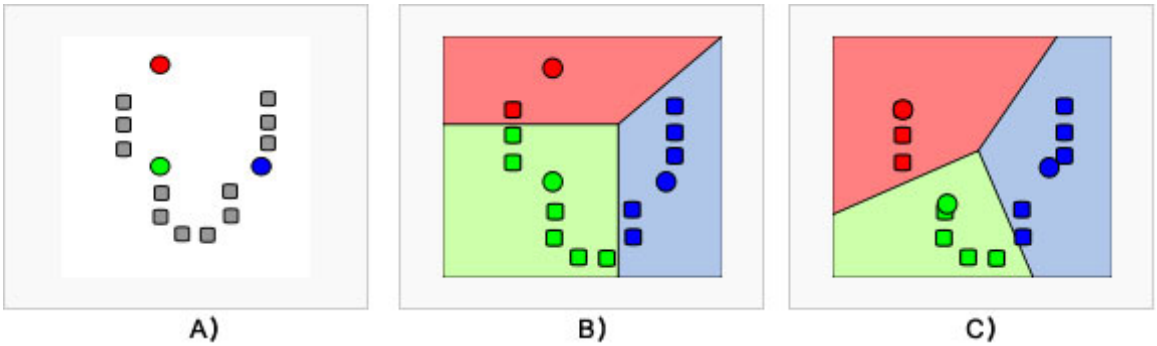
Obrázek 2.5: Obrázek popisuje dvě třídy - bílá a černá. Šrafované přímky značí dělící oblast, která popisuje možné rozdělení těchto dvou tříd. Pouze plná čára dělí tyto dvě množiny v optimální - maximalizované vzdálenosti od obou tříd

2.2.2 K-means

Oproti SVM tato metoda patří do kategorie učení bez učitele. Základní myšlenkou tohoto algoritmu je rozdělit n vstupních bodů do k shluků (angl. clusters). V prvním kroku je třeba zvolit středy těchto shluků. Již tato volba může ovlivnit úspěšnost celého shlukování. V dalším kroku se iterativně berou jednotlivé vstupní body a přiřazují se nejbližším středům. Po dokončení tohoto přiřazování následuje přepočítání souřadnic jednotlivých středů. Následně se opět opakuje krok přiřazení bodů k nejbližším středům a následné přepočítání souřadnic těchto středů. Tím vzniká smyčka, kde se při každé iteraci posouvají jednotlivé středy. Cyklus končí když v následující iteraci nedojde ke změně, nebo nebyl překročen předem stanovený počet kroků. Cílem algoritmu je dosáhnout co možná nejmenších rozdílů uvnitř shluků. Tuto minimalizaci popisuje

$$V = \sum_{j=1}^{k} \sum_{i=1}^{n} ||x_i^j - c_j||^2$$

, kde $||x_i^j - c_j||^2$ je vzdálenost mezi bodem x_i^j a středem shluku c_j.



Obrázek 2.6: Obrázek popisuje princip metody K-means. V kroku A) dochází k rozmístění tří středů shluků - červený, zelený, modrý. V kroku B) jsou body přiřazeny k nejbližším shlukům. Poté jsou přepočítány středy shluků a následně probíhá přiřazení jednotlivých bodů, toto ilustruje krok C)

2.2.3 Neuronové sítě

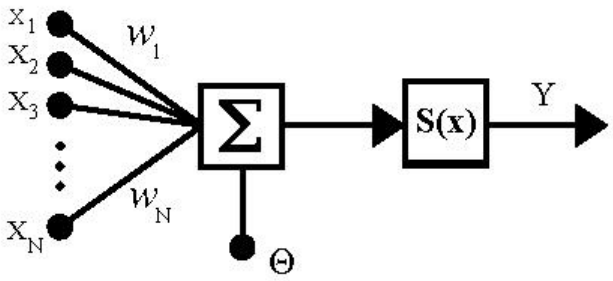
Inspirace pro vznik matematického popisu těchto sítí vychází z biologického základu. Neuronová síť sestává z dílčích částí - neurony. Tyto neurony jsou vzájemně propojeny dendrity a axony, které přenášejí elektrické vzruchy z jednoho neuronu do druhého [9]. Jeden neuron může mít takovýchto vstupů vícero.

Matematický model neuronu je zobrazen na obrázku 2.7. Vstupem neuronu je vektor reálných hodnot, který má n hodnot $x_1, ..., x_n$. Tyto jednotlivé vstupy jsou následně ohodnoceny váhami $w_1, ..., w_N$, které udávají jakou mírou může daný vstup ovlivnit výsledek. Vnitřní potenciál tohoto neuronu je vypočítán jako suma všech vážených vstupů (tím je reprezentován elektrický potenciál u biologického neuronu). Cílem učení neuronových sítí je úprava koeficientu vah w, aby příslušné vstupní hodnoty vytvářely požadovanou odezvu. Matematická rovnice neuronu má následující tvar:

$$Y = S(x) = S(\Sigma + \Theta) = S(w^T * x + \Theta) \tag{2.6}$$

Pokud provedeme následující úpravu: $w_0 = \Theta$, $x_0 = 1$, lze rovnici 2.6 zjednodušit do následujícího tvaru:

$$Y = S(w^T * x) \tag{2.7}$$



Obrázek 2.7: Vstupy neuronu jsou reprezentovány $X_1, ..., X_N$, jejich váhy jsou $W_1, ..., W_N$. Vnitřní potenciál neuronu reprezentuje Σ s aktivačním prahem Θ. Výstupem neuronu je Y, který je výstupem aktivační funkce $S(x)$

Učení může být s učitelem (angl. supervised learning), kdy síť má k dispozici vstupní údaje, které se snaží s co nejvyšší přesností namapovat na předem známou výstupní sadu tím, že zkoriguje jednotlivé váhy vstupních dendritů neuronu. Druhým způsobem učení je bez učitele (angl. unsupervised learning), kdy jsou známy pouze vstupní údaje a síť si snaží váhy sama přizpůsobit. Nefunguje tu žádná zpětná vazba.

2.2.4 AdaBoost

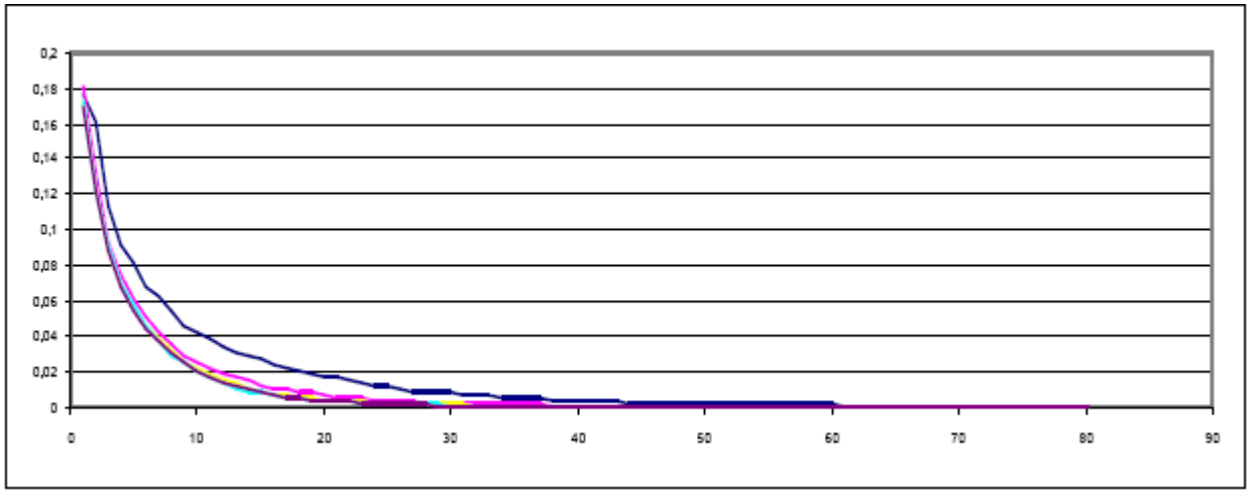
AdaBoost, v nezkrácené podobě Adaptive Boosting, byla formulována týmem Freund a Schapire [3]. Boosting je odvětví strojového učení, které se snaží kombinovat sadu slabých klasifikátorů do jednoho silného klasifikátoru, který je úspěšnější než jakýkoliv ze slabých klasifikátorů. AdaBoost dokáže v přijatelně krátkém trénovacím čase vytvořit klasifikátor s poměrně malou chybou úspěšnosti jen za použití slabších klasifikátorů, které mají například jako vstup pouze jeden příznak (histogram, threshold, rozhodovací strom,...). Výhodou nově vzniklého silného klasifikátoru je jejich přesnost.

Pro slabý klasifikátor platí podmínka, že jeho chyba musí být menší než 0.5 (tedy menší než náhodný generátor). Příkladem takového klasifikátor může být neuronová síť, naivní Bayes či práh jednoho příznaku. Silný klasifikátor je poté lineární kombinací slabých klasifikátorů.[4]

Algoritmus AdaBoost probíhá následovně: Jako vstupními daty je množina x, kde x_i je slabý vážený klasifikátory. Existuje zde distribuční funkce D_t, která uchovává jednotlivé váhy klasifikátorů, $D_t(i)$ je váha pro x_i. Poté v T cyklech probíhá následující postup:[4]

- Vyber či vytvoř slabý klasifikátor minimalizující chybu na datech váženou pomocí $D_t(i)$
- Vypočítej novou váhu klasifikátoru podle velikosti jeho chyby
- Uprav distribuční funkci D_t (pomocí převážení dat). Tím dochází ke snížení váhy u dobře klasifikovaných dat, naopak váha u špatně klasifikovaných dat stoupá. Díky této vlastnosti se v každém dalším kroku vybírá jiný slabý klasifikátor. Váhu vzorku lze převést přímo na výsledek klasifikátoru

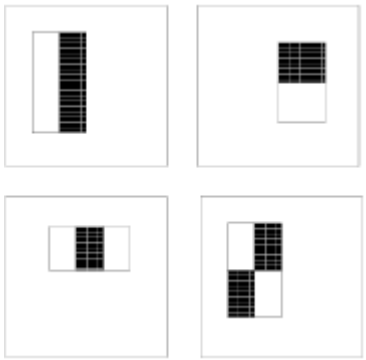
Pokud se algoritmu daří hledat klasifikátory s chybou menší než 50% chyba na trénovacích datech limitně klesne k nule. Toto ilustruje obrázek 2.8



Obrázek 2.8: Graf ilustruje pokles chyby silného klasifikátoru s rostoucím počtem slabých klasifikátorů

2.3 Viola Jones

Široce používaná metoda pro real-time detekci objektů. Byla představena v 2001 [7]. Tato metoda má tři klíčové vlastnosti. První je využití integrálního obrazu, který zrychluje výpočet příznaků vstupního obrazu (viz sekce 2.1.2). Druhou vlastností je učící algoritmus založeny na AdaBoost (viz sekce 2.2.4). Třetím přínosem je zkombinování klasifikátorů do kaskády, která rychle odfiltruje oblasti s pozadím, zatímco oblastí s vyšší pravděpodobností výskytu obličeje v této kaskádě postupují do dalších úrovní, kde dochází k detailnějšímu rozboru vstupu.



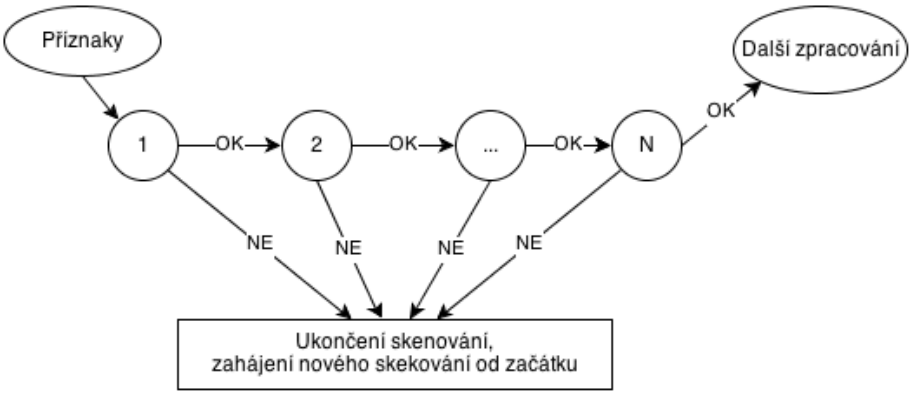
Obrázek 2.9: Typy oblastí pro počítání příznaků metodou Viola Jones

Příznaky jsou počítány jako sumy uvnitř obdélníkové oblasti. Tyto oblasti mají podobu s Haarovými příznaky (sekce 2.1.1), kde hodnoty jsou počítány na základě integrálního obrazu. Obrázek 2.9 ilustruje čtyři rozdílné typy příznaků používané touto metodou. Využity jsou obdélníky v nenatočeném, základním tvaru. Díky využití integrálního obrazu, lze tyto

příznaky počítat v konstantním čase. Jelikož takováto oblast je vždy připojena alespoň k jedné další oblasti, dovoluje to umožnit výpočet pro dva dvou stupňové příznaky pomocí šestí referencí, pro tří stupňové příznaky pomocí osmi referencí a pro čtyř stupňové příznaky pomocí devíti.

Zajímavostí je i počet unikátních obdélníkových oblastí. Například pokud vezmeme standardní $24x24$ pixelové okýnka, můžeme zde nalézt celkem až 45 396 rozdílných příznaků [7]. Samozřejmě pro reálné aplikace se z důvodů vysoké početní náročnosti nevyužívají úplně všechny kombinace příznaků. Pro volbu vhodných příznaků se volí učící algoritmy AdaBoost, který příznaky nejen vybere, ale i natrénuje klasifikátor.

Z důvodu potřeby rozpoznávat obličeje v real-time, je klasifikátor rozdělen na menší, do kaskády seřazené klasifikátory, viz 2.10. Každý takovýto klasifikátor je natrénovaný na datech z klasifikátoru o úroveň výš. Jestliže je v jakékoliv úrovni odmítnuto právě skenované okno jakýmkoliv klasifikátorem, nedochází již k další propagaci, ale k ověřování okna dalšího. V rozpoznávání obličeje, první klasifikátor používá pouze dva příznaky k rozhodování, při účinnosti 40% falešně positivních výsledků [7]. Tím dochází k velmi značnému snížení rozpoznávacího času.



Obrázek 2.10: Grafické znázornění zapojení klasifikátorů do kaskády

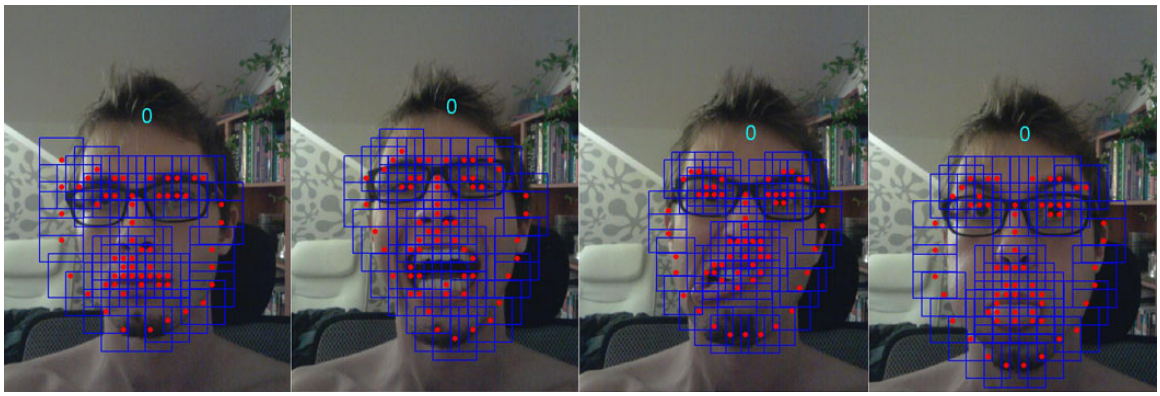
Kapitola 3

Rozpoznání obličeje, úhel natočení a lokalizace obličeje

Problém hledání a analýzy obličejů je základním stavebním kamenem počítačového vidění. V posledních letech byl udělán významný pokrok v rozpoznávání obličejů. Je stále výzvou nalézat spolehlivé algoritmy, které dokáží určit úhel natočení hlavy a rozpoložení rysů obličeje (obočí, oči, nos, ústa, brada, ...) vyskytujících se v přirozeném, nelaboratorním prostředí. Rozpoznávání obličeje je vskutku stále složitá úloha pro extrémní úhly natočení.

Tyto tři úkoly (detekce, úhel natočení a lokalizace obličeje) bývaly tradičně řešeny jako rozdílné problémy s rozličnými sadami technik, jako například: klasifikátory založené na skenovacím okně, eigen-prostorové metody, a nebo *elastic graph models*. Jádrem mé práce jsou algoritmy týmu X. Zhu a D. Ramanan [8]. V jejich práci presentují jediný model, který současně vylepšuje a spojuje tyto tři zmíněné postupy. Tvrdí, že sjednocený přístup může problém zjednodušit; například: vyhledávání polohy obličeje předpokládá obrázky, které jsou před zpracovány rozpoznávačem obličejů.

Tento model je inovativní, přesto využívá jednoduchý princip rozložení 3D struktury; Je využívána směsice stromových struktur a sdílené části histogramů. "Část"je definována na každém význačném bodu obličeje, které jsou poté umístěny do topologického modelu, který reflektuje úhel natočení obličeje. Část může být viditelná pouze v určitém modelu či úhlu natočení. Je dovoleno aby části modelů sdílely jednotlivé části histogramů. Tím je umožněno, aby modely obsahovaly velké počty částí pro úhly natočení při nízké paměťové náročnosti.



Obrázek 3.1: Ilustrace přizpůsobení detekce obličeje v závislosti na tvaru tváře/mimice.

3.1 Související práce s rozpoznáváním obličejů

Pokud je známo, neexistuje žádná předešlá práce, která spojuje problematiku rozpoznávání obličeje, úhlu natočení a lokalizaci obličeje do jednoho algoritmu. Avšak existuje bohatá škála prací, zaměřených na každý z těchto tří problémů zvlášť.

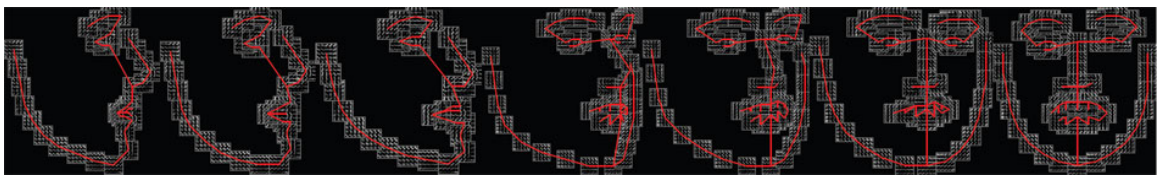
Rozpoznávání obličeje je tvořeno diskrétně trénovaným klasifikátorem skenovacího okna - všudypřítomným detektorem Viola&Jones díky své open-source implementaci v knihovně OpenCV. V porovnání s komerčními systémy, je systém [8] trénován diskrétně s velmi malou trénovací sadou.

Zjišťování úhlu natočení využívá anotovaných dat MultiPIE. Většina metod explicitně používá 3D modely, nebo sadu 2D modelů popisující 3D model. V práci je využita sada 2D modelů, které navzájem sdílejí části histogramů. Tento postup je podobný užití struktur, které popisují topologické změny mezi 2D popisem a samotným objektem.

Lokalizace obličeje se již datuje do dob klasických přístupů pomocí *Active Appearance Models* (AAMs) a elastických grafů. Nedávná práce se zaměřila na globální prostorové modely postavené na detektorech lokálních částí, též známy jako *Constrained Local Models* (CLMs). Nutno podotknout, že tento přístup předpokládá hustě propojené prostorové modely a vyžaduje algoritmus pro ověření přibližné shody. Užitím stromového modelu můžeme efektivně využít dynamického programování a najít globální a optimální řešení.

3.2 Model pro rozpoznávání obličejů

Jak již bylo zmíněno, model pro rozpoznávání je složen ze směsice stromových struktur a sdílené části histogramů V. Každý význačný bod obličeje je modelován jako jedna část a užitím tzv. globálních směsic (angl. global mixtures) jsou zachyceny topologické změny obličeje v závislosti na úhlu pozorování. Tyto směsice jsou ilustrovány v obrázku 3.2. Směsice také mohou být použity k zachycení hrubší deformace obličeje, jako například změna výrazu (otevřená ústa, úsměv, zamračení,...).



Obrázek 3.2: Ilustrace topologických změn modelu v závislosti na úhlu natočení trénovacích dat obličejů. Červené linky znázorňují tzv. pružiny mezi jednotlivými páry částí histogramů. V celém stromu se nenachází jediná smyčka. Obrázek je převzat z [8].

Stromová struktura modelu: Každý strom je lineárně parametrizovaný, obrazově strukturovaný a má následující zápis $T_m = (V_m, E_m)$, kde m indikuje směsici a $V_m \subseteq V$. Nechť I značí obraz a $l_i = (x_i, y_i)$ značí souřadnice pixelu části i. Výpočet skóre konfigurace částí $L = l_i : i \in V$ probíhá jako: [8]

$$S(I, L, m) = App_m(I, L) + Shape_m(L) + \alpha^m \quad (3.1)$$

$$App_m(I, L) = \sum_{i \in Vm} w_i^m \cdot \phi(I, l_i) \quad (3.2)$$

$$Shape_m(L) = \sum_{ij \in E_m} a_{ij}^m dx^2 + b_{ij}^m dx + c_{ij}^m dy^2 + d_{ij}^m dy \quad (3.3)$$

Rovnice 3.2 sčítá počet výskytů šablony w_i^m pro část i, naladěnou pro směsici m na pozici l_i. Zápis $\phi(I, l_i)$ představuje příznakový vektor (HOG) extrahovaný z pixelu na souřadnicích l_i v obraze I. Rovnice 3.3 počítá skóre směsic, které jsou prostorově uspořádány v části L, kde $dx = x_i - x_j$ a $dy = y_i - y_j$ značí posuv i-té části relativně vůči j-té části. Každý výraz této rovnice si lze představit jako pomyslnou pružinu, která představuje prostorové omezení mezi dvěma částmi, kde její parametry (a,b,c,d) udávají její specifickou pozici a tuhost. Poslední výraz α^m reprezentuje skalární bias směsice m.

Sdílené části histogramů modelu: Rovnice 3.1 vyžaduje šablonu w_i^m pro každou směsici/úhel natočení m části i. Může se ovšem stát, že některé části jsou shodné s jinými částmi napříč směsicemi. V extrémních případech by mohl plně sdílený model použít jedinou šablonu pro popis směsic napříč úhly pohledu $w_i^m = w_i$.

Cíl výpočtu modelu

Snahou je maximalizovat $S(I, L, m)$ z rovnice 3.1 nad L a m: [8]

$$S^*(I) = \max_m [\max_L S(I, L, m)] \quad (3.4)$$

Tedy vyčíslení všech směsic a pro každou z nich najít nejlepší konfiguraci pro její části. Jelikož každá směsice $T_m(V_m, E_m)$ je stromová struktura, lze tuto maximalizaci efektivně provést pomocí dynamického programování. Experimentálně bylo zjištěno, že plně sdílený model má menší zhoršení v rychlosti rozpoznávání.

3.3 Popis učení modelu

Pro tvorbu modelu je předpokládáno učení s učitelem. Učícímu algoritmu jsou poskytnuty fotografie obličejů s anotovanými význačnými body, dále jsou poskytnuty negativní fotografie - zde se nenachází žádný obličej. Nejdříve je třeba určit hranovou strukturu E_m jednotlivých směsic. Zatímco stromové struktury jsou vhodné pro modelování lidského těla, pro modelování obličeje to již nejsou tak jednoznačné. Při počítání tvaru se užívá algoritmus, který hledá maximum likelihood, který nejvíce popisuje strukturu obličeje pomocí stromu, při daných směsicích.

Pro positivní příklady I_n, L_n, m_n a pro negativní příklady I_n můžeme definovat predikční funkci $S(I, z)$, kde $z_n = L_n, m_n$. Nutno podotknout, že funkce pro výpočet skóre 3.1 je lineární pro šablonu w, parametry (a, b, c, d) a bias α. Spojením těchto parametrů do jediného vektoru β, můžeme funkci pro výpočet skóre zapsat v následujícím tvaru: [8]

$$S(I, z) = \beta \cdot \Phi(I, z) \quad (3.5)$$

Z výpočtu vyplyne, že pro positivní vstupy by mělo být skóre větší než jedna, zatímco pro negativní vstupy by skóre mělo být menší než mínus jedna.

Kapitola 4

Tvorba modelů pro detekci tváří a brýlí

Autoři detekčního algoritmu dali k dispozici 2 sady algoritmů určené k tvorbě modelů pro rozpoznávání. Při práci s první sadou jsem již v začátku narazil na problém. Tyto kódy totiž vyžadují rozsáhlou sadu fotografií s připojenou anotací. Tato sada je dostupná pouze po zaplacení poplatku v řádu stovek amerických dolarů. Z tohoto důvodu jsem se tedy zaměřil na druhou sadu algoritmů.

4.1 Anotace fotografií

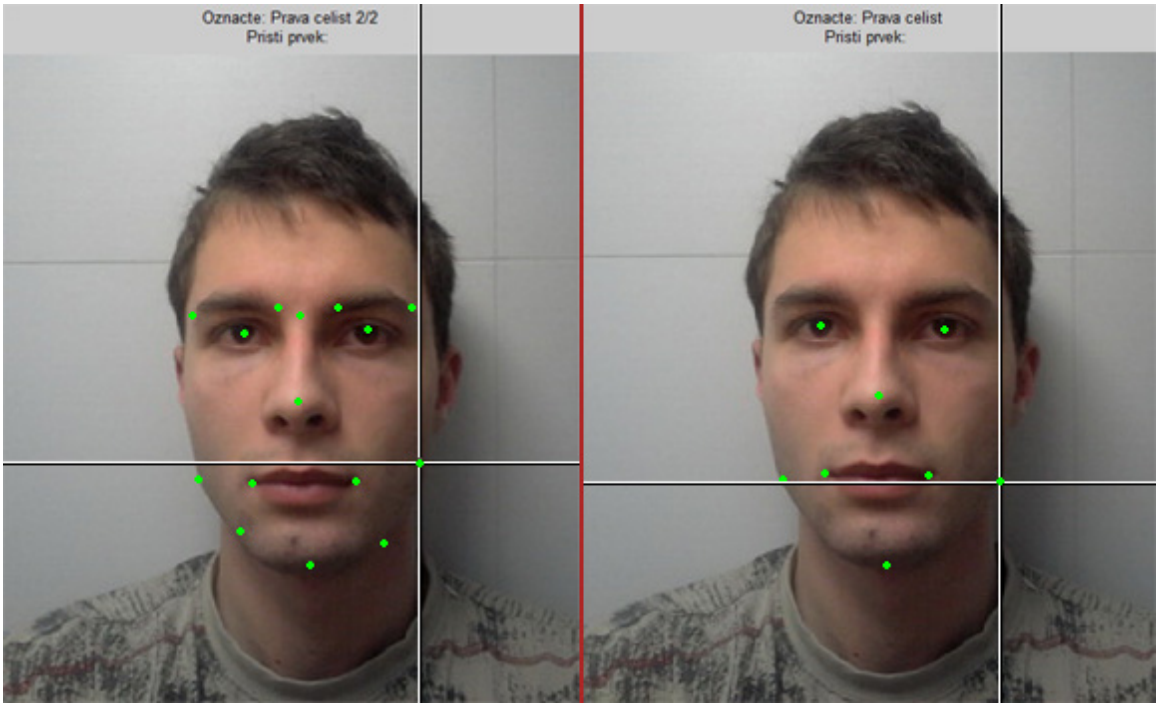
K označení význačných bodů fotografií slouží skript - `learning.m`. Uživatel si zde zvolí o jaký typ anotace půjde "BRÝLE"nebo "OBLIČEJ". Poté zvolí počet význačných bodů (4,6,8,10,15) a skript spustí. Následně je zavolána funkce - `annotateParts(DIRECTORY, REG\_MATCH, replace, part_labels)`. Tato funkce uživatelovi nabídne grafické rozhraní s popisky, kterou část fotografie má vyznačit. Počet positivních fotografií je odvozen podle počtu obrázků v adresáři `scripts\learning\positive`

Po dokončení anotace se ke každé positivní fotce vytvoří stejně pojmenovaný soubor s příponou txt, kde jsou uchovány souřadnice jednotlivých bodů. Tyto dvojice souborů poté tvoří vstupní data pro učící algoritmus tvorby modelu.

4.1.1 Význačné body

Pro obličej jsem definoval následující počet význačných bodů: 4, 6, 8, 10, 15 a 22. Body reprezentují následující části obličeje:

1. 4 body: kořen nosu, brada, levá čelist, pravá čelist
2. 6 bodů: přidává levé a pravé oko
3. 8 bodů: přidává levý a pravý kout úst
4. 10 bodů: přidává levé a pravé obočí
5. 15 bodů: přidává body k obočí a čelisti
6. 22 bodů: přidává body k obočí, nosu, ústům a čelisti



Obrázek 4.1: Skript poskytuje rozhraní pro anotaci fotografií, v horní části jsou informační řádky říkající, který význačný prvek se má označit. Na levé fotce probíhá značení o 15 význačných bodech. Na levé fotce je značeno 8 význačných bodů

4.2 Učící fáze

Druhou částí tvorby modelů je jejich samotný výpočet. Tato sada učících algoritmů nemá žádnou intuitivní podporu pro tvorby modelů obsahující více komponent - úhlů natočení. Po tomto zjištění jsem se tedy zaměřil na tvorbu modelů s jednou komponentou. To znamená, že hlava je natočena přímo do kamery pod nulovým natočením.

Učení probíhá ve dvou fázích - zpracování positivních vstupních dat a zpracování negativních vstupních dat (tyto fotografie neobsahují ani jeden obličej). Výstupem jsou dvě matice obsahující údaje o daných obrázcích. Tyto proměnné jsou poté předány do funkce `trainmodel(NAME,pos,neg,K,pa,sbin)`, kde NAME je výsledné jméno modelu, *pos* a *neg* jsou matice obsahující údaje o positivních a negativních obrázcích, K udává kolik komponent výsledný model bude mít. V mém případě se jedná o jeden úhel natočení, tedy 1; *pa* popisuje stromovou strukturu jednotlivých histogramů, platí vztah, že: *pa(i)* je rodič části *i*. A *sbin* udává šířku a výšku okénka, ze kterého se bude výsledný histogram orientovaných gradientů tvořit. V mých experimentech zachovávám výchozí konstantu 8.

Při trénování je užita metoda Support Vector Machine (SVM), popsána v kapitole 2.2.1. V jednotlivých iteracích jsou vstupní obrázky zpracovány a na základě předem vyznačených bodů jsou postupně vytvářeny histogramy orientovaných gradientů (HOG), které jsou následně spojovány do stromové struktury, kterou udává parametr *pa*. Pro každý vstupní obrázek je vytvořena stromová struktura, kde v každém uzlu se nachází jeden HOG. Učící algoritmus má také schopnost podle vstupních bodů dynamicky měnit velikost tohoto histogramu.

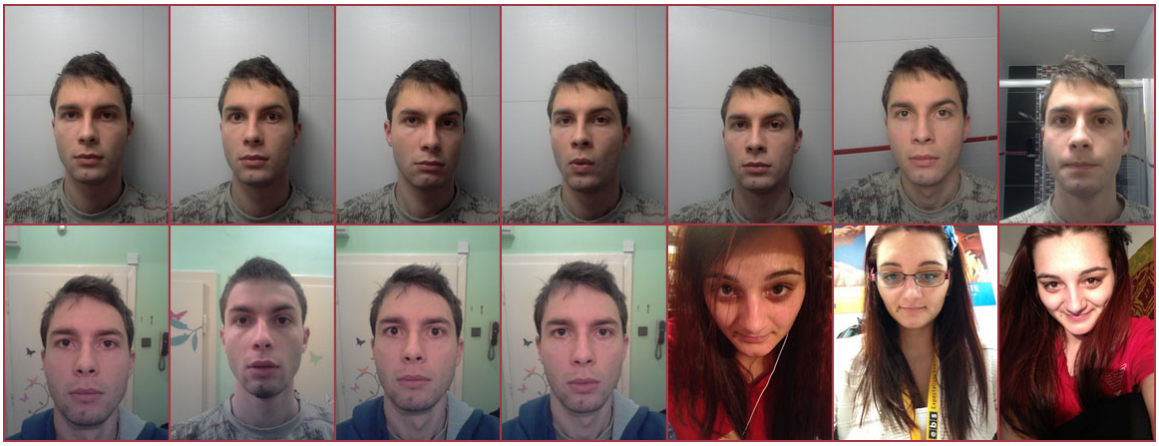
4.3 Vstupní údaje pro tvorbu modelů

Pro tvorbu každého modelu pro rozpoznávání jsou třeba dvě vstupních sady údajů. Positivní údaje = páry fotografií ve formátu jpg a soubor s anotovanými daty ve formátu txt. Dále jsou třeba negativní údaje = fotografie neobsahující hledané elementy positivních údajů (žádné obličeji ani brýle).

Positivní vstupní údaje

Pro anotaci a následný vznik modelů jsem využil 3 sady vstupních obličejů. Jako výchozí složka obsahující tyto fotografie je nastavena `scripts\learning\positive`. Toto nastavení lze změnit ve skriptu `learning.m` - konstanta *DIRECTORY*.

- 1. sada: 7 mých obrázků, vyfotografované frontálně, velikost obličeje: 125*125px
- 2. sada: 4 mých obrázků, vyfotografované frontálně, velikost obličeje: 710*710px
- 3. sada: 4 obrázky třetí osoby - ženy, vyfotografované frontálně, velikost obličeje: 420*420px



Obrázek 4.2: Příklady positivních vstupů, které sloužily k anotaci, první řádek representuje první sadu, první čtyři fotografie ve spodní řadě patří do druhé sady, a poslední tři jsou ze třetí sady

Negativní vstupní údaje

Pro správnou tvorbu modelu je třeba dodat takové fotografie, které neobsahují učený objekt - v mém případě obličej. Pro tyto účely jsem použil rozličné fotografie zástaveb, domů, silnic, řek, přírody, ulic, apod. K disposici jsem měl přibližně 80 fotografií. Některé negativní vstupy jsou zobrazeny v obrázku 4.3.

4.4 Tvorba modelů pro experimenty s počty význačných bodů

Pro experiment budu vytvářet tři druhy modelů: 1. druh bude naučen pouze z první sady fotografií. 2. druh bude naučen z první a druhé sady vstupů. A 3. druh modelů bude naučen ze všech tří sad vstupů. Cílem experimentu bude sledovat úspěšnost rozpoznávacího

Obrázek 4.3: Příklady negativních vstupních fotografií

algoritmu v závislosti na proměnlivém počtu positivních a negativních údajů. Dále chci experimentovat u jednotlivých modelů s jejich rozpoznávacími prahy, kdy se budu snažit docílit co nejvyšší úspěšnosti a sledovat kolik obličejů bude označeno jako false positive (obličej bude označen tam, kde žádný není) a dále opačný extrém - pokusím se minimalizovat false positive a sledovat, kolik bude skutečných obličejů vynecháno. A třetí práh bude nastaven v intervalu těchto dvou krajních možností. V tomto rozpětí leží optimální práh, pro daný model.

Při volbě počtů positivních a negativních vstupních obrázků modelů jsem vycházel z následující tabulky:

Model	1	2	3	4	5	6	7	8	9
POSITIVE	7	7	7	11	11	11	15	15	15
NEGATIVE	1	35	80	1	35	80	1	35	80

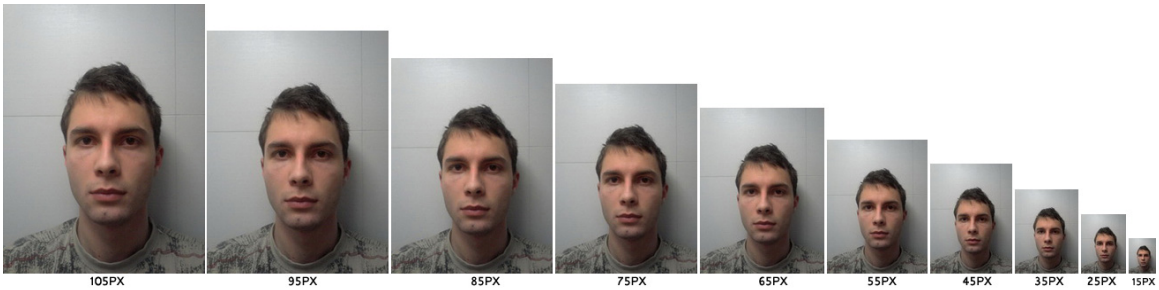
Tabulka 4.1: Kombinace vstupních positivních a negativních obrázků při tvorbě modelů

V tabulce 4.1 znázorňuji způsob výběrů positivních a negativních fotografií. Pro každý význačný bod vytvořím 9 různých modelů, které budou mít nakombinované vstupní podmínky podle tabulky. Cílem experimentu bude také sledovat učící čas jednotlivých modelů a sledovat jaký na něj jednotlivé parametry mají vliv.

4.5 Tvorba modelů pro experimenty s různou velikostí positivních vstupů

První experiment v sobě zahrnuje vstupní positivní fotografie o různé velikosti obličejů. V tomto experimentu se chci zaměřit na variantu, že všechna vstupní anotovaná data měla stejnou velikost obličejů. Každý následující model bude mít velikost obličeje o 10px menší, než model předešlý. Chci tedy zjistit, jak malé obličeje mohou být užity pro trénování, aniž by docházelo k značné degradaci úspěšně nalezených obličejů.

Pokud první experiment bude mít úspěšnost s modely natrénovanými na první datové sadě (obličeje 125px*125px) více než 90%, pak tento experiment začnu s tímto rozlišením a postupně budu ubírat.



Obrázek 4.4: Ilustrace poměrů velikosti vstupních positivních dat.

4.6 Tvorba modelů s nesprávně anotovanými fotografiemi

Cílem tohoto experimentu bude odhalit, jak je učící algoritmus náchylný či odolný na postupné zvyšování nesprávně anotovaných positivních vstupů. V závislosti na předešlých experimentech zvolím počet vstupních dat takový, který dosahuje nejvyšší úspěšnosti. U tohoto počtu poté anotuji třetinu fotografií na nesprávném místě, či vložím obrázek neobsahující obličej. Vytvořeným modelem poté rozpoznám obličeje v datové sadě a vyhodnotím úspěšnost. Následně budu zvyšovat počet takto nesprávně anotovaných vstupů a sledovat účinnost rozpoznávání.

4.7 Tvorba modelů pro experimenty s rozpoznáváním brýlí

Dalším zaměřením experimentu bude model naučený pro rozpoznávání brýlí. Stejně jako obličej, brýle mají svůj charakteristický rys. Jdou popsat těmito body - levý/pravý a horní/dolní roh, pro každou ze dvou půlek. Dohromady tedy osm důležitých bodů. Lze ovšem zvolit středy mezi těmito body, ale domnívám se, že rohy budou mnohem víc výrazněji definovat tvar. Pro následné testování bude vytvořena sada obsahující přibližně 100 fotografií s brýlemi.

Kapitola 5

Experimenty s rozpoznáváním obličeje a brýlí

Tato kapitola popisuje vlastní experimenty prováděné s poskytnutou implementací algoritmů. Cílem této kapitoly je presentovat výsledky jednotlivých experimentů — různý počet význačných bodů na trénovacích datech (zde se zaměřuji také na časy potřebné k učení jednotlivých modelů a na jednotlivé časy modelů pro rozpoznávání, dále na vliv prahů na nalezené obličeje a počet falešně positivních označení), vliv velikosti obličejů trénovacích dat, dále sleduji jakým způsobem ovlivní úspěšnost rozpoznávání počet nesprávně anotovaných vstupů. Na závěr jsou představeny výsledky modelů pro rozpoznávání brýlí.

K účelům testování obličejů jsou použity výhradně fotografie z osobní galerie autora. Tato sada čítá 52 fotografií s celkovým počtem 94 obličejů. Testovací sada pro experimentování s modelem brýlí obsahuje 100 obrázků zahrnujících 110 brýlí.

5.1 Různý počet význačných bodů

5.1.1 8 bodů

Tato sekce zobrazuje výsledky modelů, které byly natrénovány na osmi význačných bodech obličeje a jednotlivé podkapitoly znázorňují výsledky modelů v patřičných tabulkách.

7 positivních vstupů

Následující tabulka zobrazuje výsledky pro 7 positivních vstupů

Negativních			1			35			80
Učící čas			22s			69s			128s
Práh	0,33	0,26	0,22	0,11	0,10	0,06	0,12	0,08	0,07
Nalezeno [%]	67,0	92,6	97,9	75,5	83,0	97,9	74,5	96,8	97,9
False Positive	0	8	59	0	2	91	0	66	153

Tabulka 5.1: Výsledky modelů pro 7 positivních a 1, 35, 80 negativních vstupů

Učící čas jednotlivých modelů je lineárně závislý na počtu negativních vstupů. Průměrný rozpoznávací čas těchto modelů je 253,9 sekund, což je o 0,8s méně než průměr všech devíti modelů. Nejhoršího výsledku dosahuje první model s prahem 0.33, kde bylo snahou získat 0 false positive - jeho úspěšnost je 67%. V tomto případě dochází k zajímavému úkazu

v případech, kdy byla snaha maximalizovat úspěšnost nalezených obličejů. S rostoucím počtem negativních vstupů u modelů také roste počet falešně positivních ohodnocení.

11 positivních vstupů

Následující tabulka zobrazuje výsledky pro 11 positivních vstupů

Negativních			1			20			80
Učící čas			49s			95s			154s
Práh	0,65	0,48	0,45	0,21	0,18	0,15	0,33	0,30	0,27
Nalezeno [%]	75,5	97,9	98,9	86,1	92,6	98,9	90,4	97,9	98,9
False Positive	0	20	60	0	5	34	0	6	29

Tabulka 5.2: Výsledky modelů pro 11 positivních a 1, 20, 80 negativních vstupů

Průměrný čas potřebný k rozpoznávání v téhle variantě je 261,6 sekund, což je o téměř 7 vteřin pomaleji než celkový průměr. Přidáním 4 positivních vstupů, oproti předešlým modelům v 5.1 dochází k zvýšení rozpoznávání jednotlivých modelů. Například v porovnání u 80 negativních vstupů dochází až k 15% zlepšení. Zde stojí za všimnutí, že se zvýšením počtu positivních vstupů došlo ke zvýšení přesnosti vyhledávání a zároveň ke snížení falešně positivních ohodnocení. Další sada testů odhalí, zda k tomuto dochází i s opětovným navýšením positivních vstupů.

15 positivních vstupů

Následující tabulka zobrazuje výsledky pro 15 positivních vstupů

Negativních			1			35			80
Učící čas			přerušeno			přerušeno			přerušeno
Práh	0,74	0,59	0,57	0,36	0,30	0,26	0,45	0,36	0,26
Nalezeno [%]	90,4	95,7	98,9	90,4	95,7	98,9	83,0	93,6	100
False Positive	0	38	54	0	5	16	0	3	29

Tabulka 5.3: Výsledky modelů pro 15 positivních a 1, 35, 80 negativních vstupů

Podíváme-li se do tabulky 5.3 na časy potřebné k tvorbě modelů, zjistíme, že učení došlo k jeho přerušení. Toto vzniklo nedostatkem volné operační paměti. Učící algoritmus je však koncipován tak, že při tvorbě jednotlivých komponent si ukládá informace do adresáře `cache` do souborů ve formátu matlabovských matic. Ty poté v takovém případě při znovu spuštění učení pouze načte a dále pokračuje ve tvorbě dosud nevytvořených komponent. Jelikož je však tento proces rozpůlen, nelze tedy určit přesný čas potřebný k tvorbě modelu.

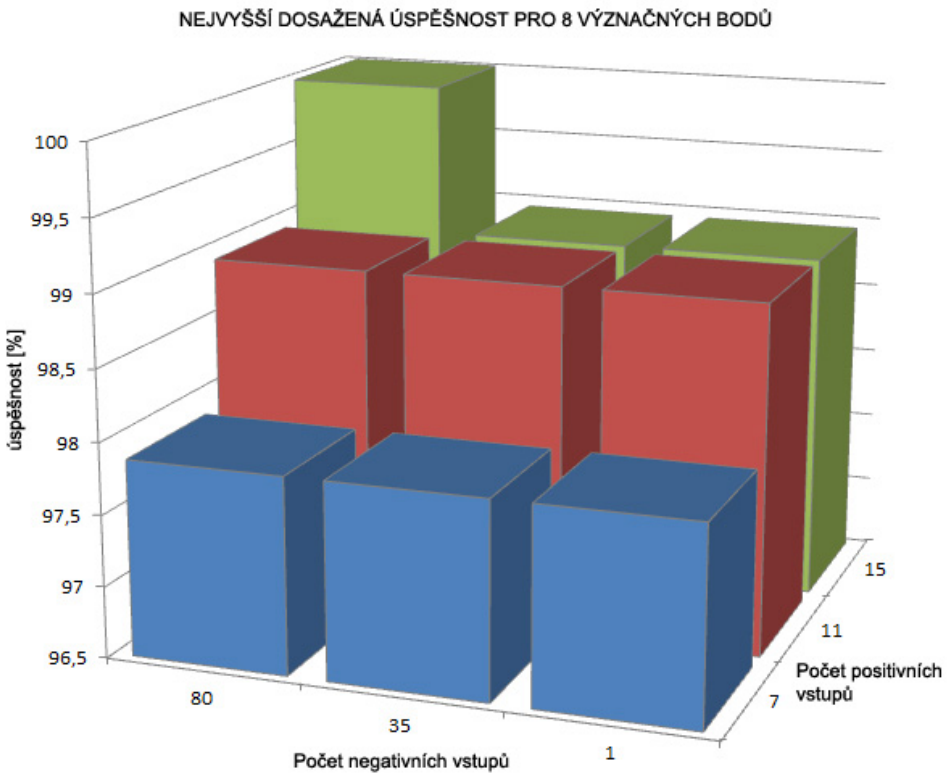
Při užití patnácti vstupních positivních fotografií jednotlivé modely potřebovaly v průměru 248,8 sekund k rozpoznávání. Nejhoršího výsledku v této sadě dosahuje poslední model s prahem 0,45. V tomto experimentu se poprvé setkáváme s modelem, který rozpoznal 100% testovacích vstupů při 29 false positive. V předešlé sekci bylo zjištěno, že s počtem stoupajících positivních a negativních vstupů klesá počet false positive a zároveň stoupá počet nalezených obličejů. Táto vlastnost se projevuje i v tomto případě. Na tuto vlastnost se tedy i zaměřím v experimentech s patnácti význačnými body

Shrnutí

Nyní srovnám všechny údaje z tabulek 5.1, 5.2 a 5.3. 254,7s je průměrný čas modelů potřebných k rozpoznání sady, medián je roven 250,0s. Nejpomaleji byly fotografie rozeznávány sadou druhou - s jedenácti positivními vstup, která je o sedm sekund pomalejší oproti průměru, což odpovídá 2,7% odchylky od průměru. Nejrychleji rozpoznávací sadou je sada třetí - s patnácti positivními vstupy. Od průměrného času se odchyluje o mínus šest vteřin, což je vzhledem k množství vstupních (positivních i negativních) dat překvapivé. Pro další testy s patnácti body budu tedy předpokládat, že kombinace positivních a negativních vstupů nemá vliv na rychlost rozeznávání (očekával jsem, že čím více vstupů bude model mít, tím jeho rozpoznávání bude pomalejší).

V téhle sadě modelů se setkáváme pouze jedenkrát se 100% účinností, kterou dosáhl model poslední s prahem 0,26 při 29 falešně positivních. Čas potřebný pro tvorbu modelu je lineárně závislý na počtu negativních vstupů, kde v průměru přidání jedné negativní fotografie znamenalo navýšení potřebného času o 1,3 sekundy.

Závěrem vyplývá, že model vytvořený již z osmi význačných bodů dává výsledky, které za cenu nízkého počtu falešně positivních označení detekuje přes 90% obličejů ve fotografiích.Úspěšnost jednotlivých modelů je ilustrována v obrázku 5.1.



Obrázek 5.1: Graf ilustruje nejvyšší dosaženou úspěšnosti jednotlivých modelů v závislosti na počtu positivních a nedativních vstupních údajích

5.1.2 15 bodů

Tato sekce zobrazuje výsledky modelů, které byly natrénovány na 15 význačných bodech obličeje a jednotlivé podkapitoly znázorňují výsledky modelů v patřičných tabulkách.

7 positivních vstupů

Následující tabulka zobrazuje výsledky pro 7 positivních vstupů

Negativních			1			35			80
Učící čas			29s			109s			248s
Práh	0,37	0,35	0,33	0,23	0,18	0,16	0,22	0,21	0,19
Nalezeno [%]	91,5	93,6	97,9	93,0	98,9	100	95,7	97,9	98,9
False Positive	2	9	26	0	4	15	1	4	18

Tabulka 5.4: Výsledky modelů pro 7 positivních a 1, 35, 80 negativních vstupů

Z tabulky 5.4 lze pozorovat, že pro učící čas modelů platí přímá úměra závislá na počtu negativních vstupních obrázcích. Průměrný čas rozpoznávání je 292,8 sekund. Nejhoršího výsledku dosahuje první model, kterému byl poskytnut pouze jeden negativní obrázek - úspěšnost 91,5% společně s dvěma false positive. Za povšimnutí stojí rozdíl v úspěšnosti druhého modelu s prahem 0,16, který dosahuje 100% při 15 false positive, kdežto třetí model s prahem 0,19 dosahuje úspěšnosti 98,9% s 18 false positive. Podíváme li se však do dalších tabulek, lze usoudit, že se zřejmě jedná o anomálii.

11 positivních vstupů

Následující tabulka zobrazuje výsledky pro 11 positivních vstupů

Negativních			1			20			80
Učící čas			59s			přerušeno			přerušeno
Práh	0,73	0,70	0.67	0,26	0,22	0,20	0,35	0,33	0,30
Nalezeno [%]	94,7	97,9	100	87,2	95,7	97,9	91,5	95,7	98,9
False Positive	2	11	27	0	4	16	0	2	8

Tabulka 5.5: Výsledky modelů pro 11 positivních a 1, 20, 80 negativních vstupů

Průměrný čas potřebný k rozpoznávání v téhle variantě je 293,2 sekund. Nejhoršího výsledku v tomto případě dosahuje druhý model obsahující 20 negativních obrázků. Pokud srovnáme první a druhý model, lze si všimnout, že první model dosahuje vyššího procenta nalezených obličejů. S výší tohoto procenta ale souvisí také vyšší hodnota false positive.

Druhý model tedy skutečně dosahuje nižšího procenta úspěšnosti, ovšem počet false positive je také nižší. Pokud bychom se měli rozhodnout, který z těchto modelů je lepší, vše by záleželo na dané situaci, či okolnostech. Když do srovnání zahrneme i třetí model, s 80 negativními obrázky, lze si všimnout, že úspěšnost je nepatrně vyšší než u modelu 2, ale počet false positive je přesně poloviční.

15 positivních vstupů

Následující tabulka zobrazuje výsledky pro 15 positivních vstupů

Negativních			1			35			80
Učící čas			přerušeno			přerušeno			přerušeno
Práh	0,72	0,65	0.58	0,41	0,37	0,35	0,42	0,41	0,34
Nalezeno [%]	91,5	97,9	100	97,9	100	100	97,9	100	100
False Positive	0	2	31	0	2	9	0	1	17

Tabulka 5.6: Výsledky modelů pro 15 positivních a 1, 35, 80 negativních vstupů

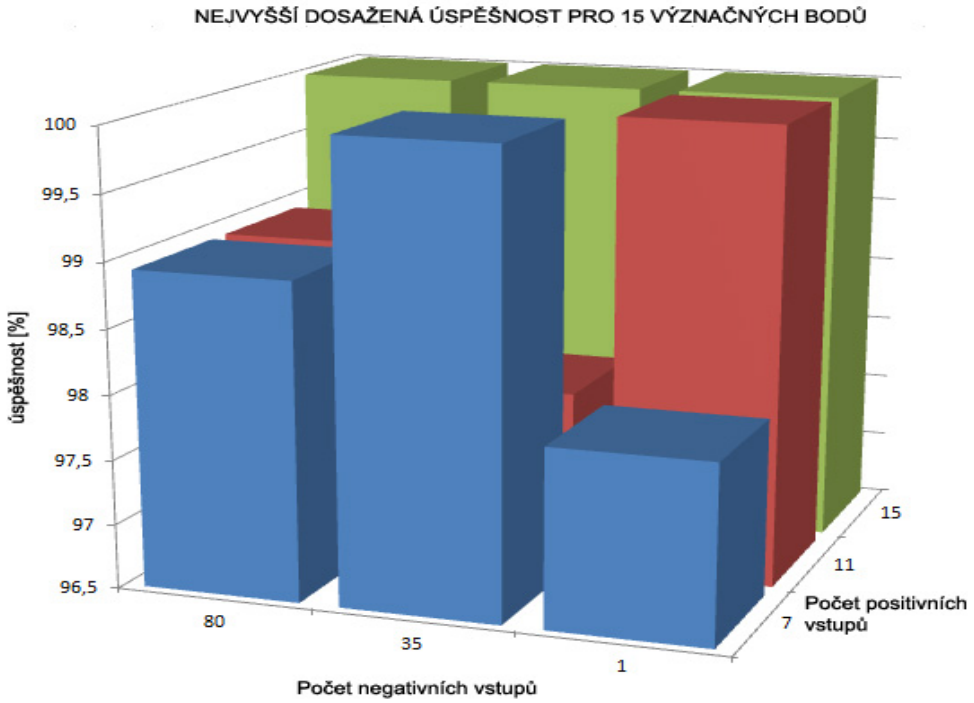
Tabulka 5.6 zobrazuje, že ani u jednoho modelu se nepodařilo uskutečnit jeho tvorbu na poprvé. Opět došlo k problému s nedostatkem paměti. Při užití patnácti vstupních positivních fotografií jednotlivé modely potřebovaly v průměru 293,3 sekund k rozpoznávání. Nejhoršího výsledku opět podle očekávání dosahuje první model s prahem 0,72. V této variantě se setkáváme pětkrát s případy, kdy model nalezl všechny obličeje. Rozdíl je v počtu false positive a také detekčním čase. Z tabulky vyplývá, že s rostoucím počtem negativních vstupů klesá počet false positive. Za povšimnutí stojí druhý a třetí model, které mají dvakrát nalezeno 100%. Podle očekávání s klesajícím prahem stoupá počet false positive. Tedy ze dvou na devět, respektive z jednoho na sedmnáct.

Shrnutí

Nyní srovnám všechny údaje z tabulek 5.4, 5.5 a 5.6. Průměrný čas potřebný na zpracování datové sady je 293,1 sekund a mediánem je 293,1 sekund. Z toho lze vyvodit, že kombinací velkého/středního/malého počtu positivních a negativních fotografií nemá žádný znatelný vliv na rychlost jednotlivých modelů. Nejrychlejším modelem je devátý model - 15 positivních a 80 negativních s prahem 0,34. Nejpomalejšího výsledku dosahuje čtvrtý model - 11 positivních a 1 negativní s prahem 0,67. Jejich rozdíl činí 6,2 sekundy což představuje 2% průměrného času potřebný k detekci.

Nejméně úspěšný, dle očekávání, je první model - 7 positivních a 1 negativní, který dosahuje úspěšnosti 91,5% při dvou false positive. Nejvíce úspěšných, tedy těch co dosáhli 100% nalezení, je přesně sedm. Pokud vezmu v úvahu také počet false positive, docházím k závěru, že nejúspěšnějším modelem je devátý model - 15 positivních a 80 negativních při prahu 0,41 - tento dosahuje pouze 1 false positive. A jeho rychlost je pouze o 0,6s vyšší než rychlost průměrná.

Jelikož při tvorbě složitějších modelů docházelo k zastavení učení z nedostatku paměti, nelze u tohoto případu vyvodit žádný závěr o vlastnostech daného času. Pouze u modelů se sedmi positivními vstupy došlo k bezproblémovému vzniku modelů, lze pouze z těchto údajů vyvodit závěr. Pro tento případ je minimální čas potřebný k naučení při jednom negativním vstupu 29 sekund. Další nárůst je lineární, kde na každou přidanou negativní fotografii připadá navýšení o přibližně 3,1 sekundy. Lineární tendenci lze očekávat i u ostatních přerušených modelů. Úspěšnost jednotlivých modelů je ilustrována v obrázku 5.2.



Obrázek 5.2: Graf ilustruje nejvyšší dosaženou úspěšnosti jednotlivých modelů v závislosti na počtu positivních a nedativních vstupních údajích

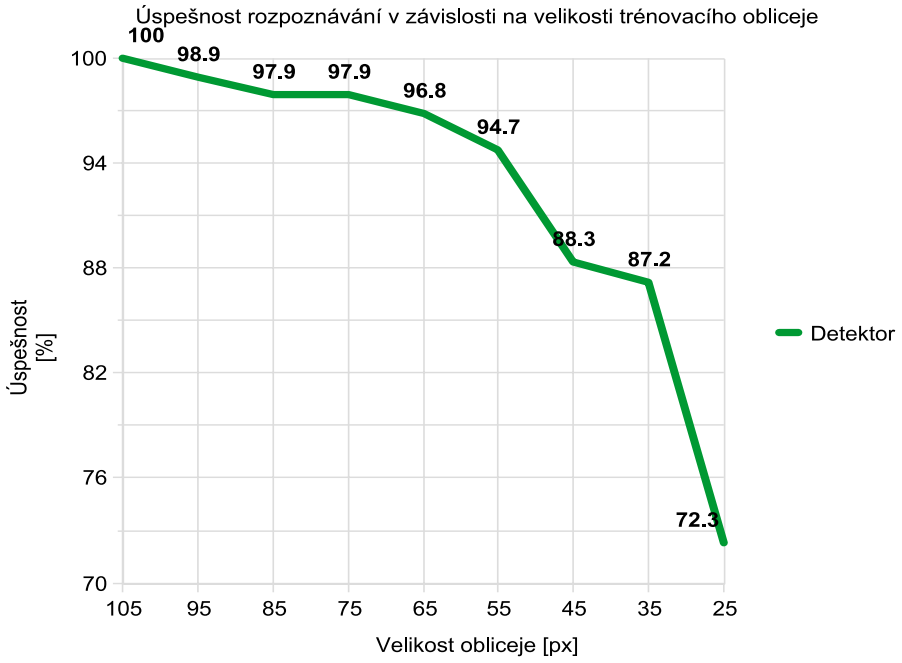
5.2 Různá velikost positivních fotografií

Z předešlého experimentu vyplývá, že již pro první datovou sadu o velikosti 125px*125px je algoritmus schopen nalézt více než 90% obličejů, tak jsem se nyní zaměřil na tvorbu modelů, kde postupně zmenšuji toto rozlišení a sleduji, jaký vliv to má na úspěšnost hledání.

Velikost obličeje	105px	95px	85px	75px	65px	55px	45px	35px	25px	15px
Nalezeno [%]	100	98,9	97,9	97,9	96,8	94,7	88,3	87,2	72,3	∅
False Positive	7	4	1	4	2	5	10	1	18	∅
Velikost buňky	8	8	8	6	6	6	4	3	2	2

Tabulka 5.7: Výsledky modelů pro 15 positivních a 1, 35, 80 negativních vstupů

Cílem tohoto experimentu bylo odhalit, jaké nejmenší rozměry mohou mít trénovací obličeje, aby fungovala detekce. Tabulka 5.7 zobrazuje úspěšnosti modelů vytvořených z daných velikostí obličejů. Význam velikosti buňky je popsán v 2.1.3. Z tabulky vyplývá, že pro model natrénovaný z obličejů velikých 15*15px již nedostáváme žádné výsledky. Při vložení tohoto modelu do detektoru, algoritmus zhavaruje. V tomto experimentu jsem musel měnit i velikost rozpoznávací buňky. Tuto modifikaci jsem provedl, jelikož učící algoritmus nebyl schopen dokončení tvorby modelu. Z tohoto usuzuji, že se zmenšujícím se rozlišením trénovacích vstupů je také zmenšovat rozlišení buňky.



Obrázek 5.3: Úspěšnost modelů v závislosti na velikosti trénovacího obličeje

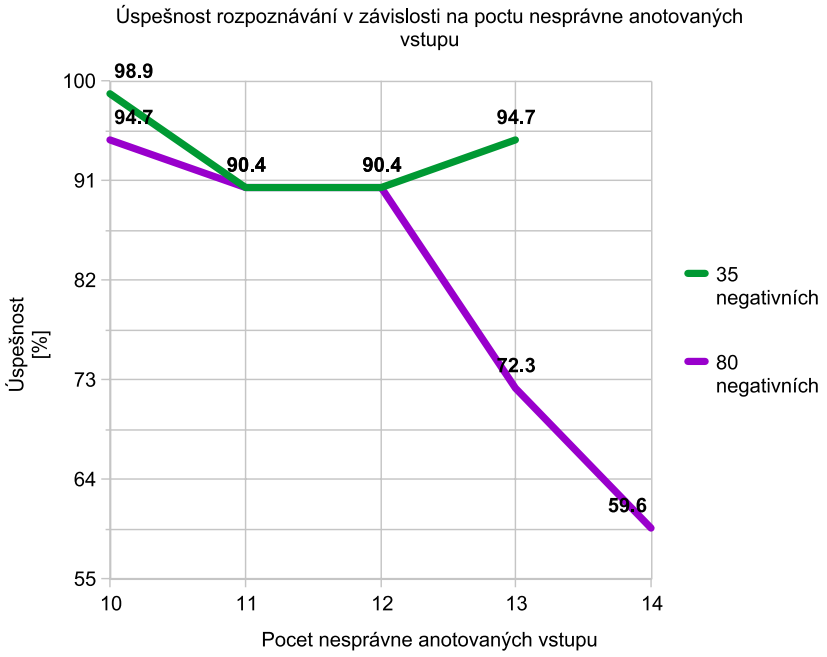
5.3 Vliv počtu nesprávně anotovaných fotografií

Tento experiment má odhalit, jak je učící algoritmus odolný či náchylný na postupné zvyšování množství špatných anotovaných vstupních dat. Z předchozích experimentů plyne, že vhodná startující kombinace je 15 význačných bodů a 15 trénovacích fotografií. Proměnným parametrem zde bude počet negativních vstupů.

Experiment jsem zahájil špatným anotováním význačných bodů - všechny jsem umístil na souřadnice [0,0]. Tuto variantu ovšem učící algoritmus nedokázal zpracovat a opakovaně havaroval. Proto jsem souřadnice všech 15 fotografií zanechal, ovšem dané fotografie jsem nahradil obrázkem pouze o bílé barvě, tedy hodnoty gradientů ve všech bodech pixelů jsou rovny nule. Na začátku experimentu jsem předpokládal, že pro tvorbu modelu bude použita třetina nesprávně anotovaných fotografií, ale ve výsledcích jsem nezaznamenal žádnou změnu ve výkonnosti, proto jsem zvýšil počet na deset. Graf 5.4 ilustruje průběh úspěšnosti. První sadu experimentů jsem provedl s 80 negativními vstupy - to ilustruje fialová přímka. Průběh je dle očekávání klesající. Experimenty končí na 14 nesprávných vstupech, protože maximum je 15, bylo tedy zapotřebí zachovat alespoň jednu správně anotovanou fotografii.

Druhá sada byla vytvořena za použití 35 negativních vstupů. Zde je překvapující průběh grafu, protože se ve všech svých bodech vyskytuje nad, nebo splývá s fialovou přímkou. Zde je možným vysvětlením tohoto úkazu tzv. přeučení. Ovšem pro 14 špatných vstupů již tento model nedává žádné přijatelné výsledky (mnohonásobně vyšší počet false positive, než nalezených obličejů).

Pro úplnost experimentu byly vytvořeny modely, které měly k disposici pouze jeden negativní vstup. Zde pouze první model - 10 nesprávných vstupů - dosáhl měřitelných hodnot. Jeho úspěšnost je pouhých 38%. Další varianty (11, 12, 13 a 14) nedávaly žádné měřitelné hodnoty.

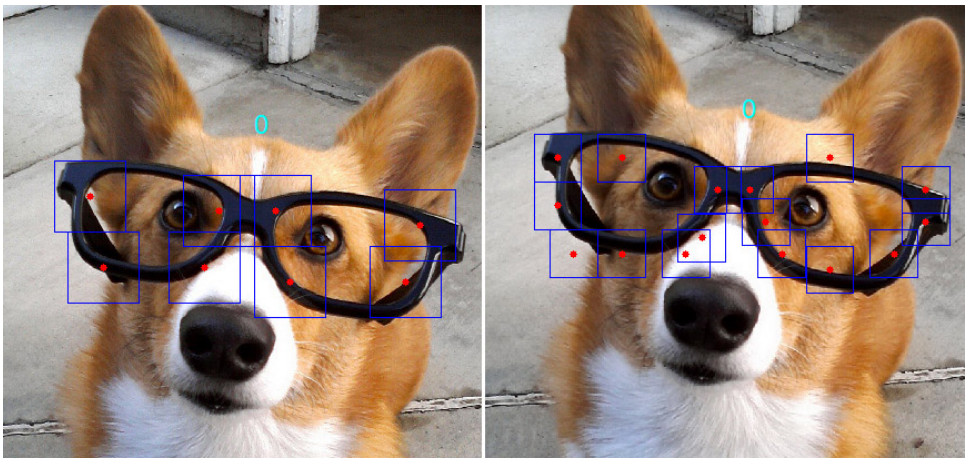


Obrázek 5.4: Vliv počtu nesprávně anotovaných fotografií na úspěšnost rozpoznávání

5.4 Rozpoznávání brýlí

Tento experiment je zaměřený na zjištění, zda je schopné vytvořit použitelný model pro rozpoznávání brýlí v obraze. Testovací sada obsahuje 100 fotografií se 110 brýlemi. Nachází se v ní rozličné typy a umístění brýlí (tlusté/tenké rámy, na lidech/zvířatech, namalované, či dokonce vytetované, apod.). Pro experimenty byly vytvořeny dva modely:

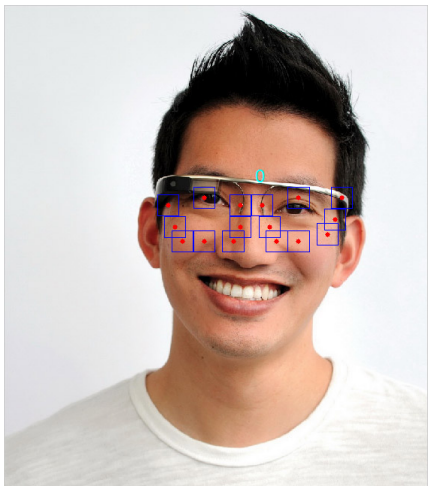
- model A: 8 význačných bodů, 9 positivních vstupů a 20 negativních
- model B: 16 význačných bodů, 9 positivních vstupů a 80 negativních



Obrázek 5.5: Detektor rozeznal brýle na hlavě psa. Levý obrázek je detekován modelem A, pravý obrázek detekován modelem B

Pro danou testovací sadu dosahuje model A nejlepšího výsledku 93% při 22 false positive. Očekával jsem, že model B, který má více vstupních dat dosáhne ještě lepších výsledků. Tento model dosáhl stejné úspěšnosti, ale má méně false positive - 14. Po dokončení testování na sadě, ve které se na všech fotografiích vyskytují alespoň jedny brýle, jsem provedl detekci brýlí na sadě užívanou pro rozpoznávání obličejů. Zde se vyskytuje 75 obličejů bez brýlí a 19 obličejů s brýlemi. Cílem bylo zjistit, jaké bude chování modelů, když jim bude předložen obličej bez brýlí.

Model A na druhé datové sadě objevil pouhých 53% případů obličeje s brýlemi, zatímco Model B nalezl plných 100%. Překvapujícím faktem ovšem je, že modely začaly detekovat oblasti očí, i přestože tam žádné brýle nebyly. Model A rozeznal 28% ze 75 případů. Model B jich nalezl 66%. Tento úkaz je způsoben tvarem obličeje v oblasti očí, které v jistých situacích mohou mít stejné zakřivení, či hodnoty gradientů, stejně jakoby se na stejném místě vyskytovaly brýle. Této vlastnosti by se dalo například využít při aplikacích, které k ovládání využívají pohyby očí. Takto by se nejdříve v obraze našla poloha očí a poté by mohl být použitý jiný algoritmus, který by v tomhle místě detekoval oči a jejich pohyb.



Obrázek 5.6: Model B rozpoznal oblast očí. Muž má na sobě koncept google glass

Kapitola 6

Závěr

Cílem této práce bylo experimentálně vyhodnotit vlastnosti algoritmů, které byly zveřejněny v práci [8]. Jako hlavními cíli experimentů bylo ověřit, zda je možné úspěšně rozpoznávat obličeje s ještě menší datovou sadou než použili autoři a zda je možné pro další rozpoznávání využít i menší počet význačných bodů. Dalším cílem bylo zjistit účinnost učícího algoritmu, jeho odolnost vůči počtu nesprávně anotovaných fotografií, následně bylo experimentováno s velikostí obličejů trénovacích dat. Závěrečný experiment měl ověřit, zda je možné algoritmu použít nejen na rozpoznávání obličejů ale i také brýlí.

Pro správné pochopení jak detektor funguje jsem musel nastudovat teorii metod extrakcí obrazových příznaků. Pří studiu této oblasti jsem se zaměřil na oblast Haarových příznaků, se kterou úzce souvisí algoritmy integrálního obrazu. Jádrem rozpoznávání jsou histogramy orientovaných gradientů. Pro následné zpracování příznaků byla prostudována teorie klasifikátorů. Mezi něž patří: Support Vector Machine, K-means, Neuronové sítě a Ada-boost. Následně jsem nastudoval teorii o Viola-Jones, který využívá klasifikátory v kaskádě, čímž urychluje rozpoznávání.

Pro tvorbu jednotlivých modelů pro rozpoznávání byly získány dva algoritmy. Po prostudování jednoho jsem došel k závěru, že pro správnou funkčnost potřebuje specificky anotovanou datovou sadu, která je ovšem placená v řádu stovek amerických dolarů. Při testování druhé sady učících algoritmů se po dlouhodobém experimentování nepodařilo zprovoznit učení modelů pro více úhlů pohledů. K tomuto došlo ze dvou možných důvodů: učící algoritmus nepodporuje tvorbu modelů pro více úhlů pohledů, a nebo tato možnost je implementována, ovšem není zde žádná přímá viditelnost na tuto funkci. Proto je práce zaměřena na modely, které hledají obličeje přímo natočené do objektivu.

Z experimentů vyplývají následující závěry. Autoři algoritmů dali k disposici naučené modely, které byly vytvořeny z více než 30 význačných bodů. První experiment měl ověřit, zda nižší počet bodů bude stále dostatečný pro nalezení obličeje. Výsledkem je, že při 15 význačných bodech a 15 trénovacích fotografiích naučený model dokázal rozpoznat veškeré obličeje v datové sadě. Druhý experiment ověřoval, jaký může být nejmenší rozměr trénovacího obličeje. Z experimentů vyplývá, že pro rozměry $25 \times 25px$ dokáže algoritmus rozpoznat 72% datové sady a při rozměrech $15 \times 15px$ přestává rozpoznávací algoritmus naprosto fungovat. Třetí experiment ověřuje jaký vliv má počet nesprávně anotovaných trénovacích fotografií. Při 14 nesprávných z 15 dokáže algoritmus rozpoznat téměř 60% datové sady. Poslední experiment měl za úkol ověřit, zda lze modely trénovat pro rozpoznávání brýlí. Zde modely rozpoznaly 92% brýlí na testovací sadě brýlí a poté byl modelu předložena sada obličejů, kde rozpoznal 66% oblastí očí (zde se žádné brýle nenacházely).

Z experimentů vyplývá, že algoritmus založený na histogramu orientovaných gradientů

spojený do stromové struktury podává úctyhodné výsledky. V porovnání s jinými přístupy, které potřebují trénovací sadu o velikosti stovek, či tisíců obličejů, tento mechanismus již při 15 vstupech dává velmi uspokojivé výsledky. Dále je také velmi odolný na různou velikost trénovacích obličejů. Je také odolný vůči nesprávně anotovaným trénovacím datům. Algoritmus lze také použít i pro rozpoznávání jiných objektů, nejenom obličejů.

Jako následné pokračování, či vylepšení práce by bylo vhodné se zaměřit na učící algoritmus, aby byl schopen učit modely pro více úhlů pohledů. Jelikož jsou jednotlivé zdrojové kódy psané v Matlabu (za částečného využití funkcí c++), bylo by zajímavé provést kompletní přepsání do C++ (či jiného vyššího jazyku) a pozorovat, zda to má vliv na urychlení rozpoznávání. Sloučení problémů lokalizace tváře, její detekce a zjištění úhlu natočení do jednoho algoritmu má rozhodně vysoký potenciál v oblasti rozpoznávání obličejů do budoucna.

Literatura

[1] DALLAL, N.; TRIGGS, B.: Histograms of Oriented Gradients for Human Detection. [online], 2005.
URL <http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf>

[2] DOBEŠ, M.: *Zpracování obrazu a algoritmy v C#*. Praha: BEN - technická literatura, 2008, ISBN 978-80-7300-233-6.

[3] FREUND, Y.; SCHAPIRE, R. E.: A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, ročník 55, August 1997: s. 119–139.

[4] HRADIŠ, M.: Klasifikace a rozpoznávání - Boosting. online, 2012.
URL <http://www.fit.vutbr.cz/study/courses/IKR/public/stare_prednasky_2012/07_boosting/IKR-Boosting.pdf>

[5] MADZAROV, G.; GJORGJEVIKJ, D.; CHORBEV, I.: A Multi-class SVM Classifier Utilizing Binary Decision Tree. *Informatica*, ročník 33, č. 2, 2008: s. 223–241, ISSN 1854-3871.

[6] SZELISKI, R.: *Computer Vision, Algorithms and Applications*. Springer, 2010, ISBN 978-1-84882-934-3.

[7] VIOLA, P.; JONES, M.: Rapid object detection using a boosted cascade of simple features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 2001: s. 501–518, doi:10.1109/CVPR.2001.990517.

[8] ZHU, X.; RAMANAN, D.: Face detection, pose estimation, and landmark localization in the wild. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012: s. 2879–2886, doi:10.1109/CVPR.2012.6248014.
URL <http://www.ics.uci.edu/~xzhu/face/>

[9] ŠÍMA, J.; NERUDA, R.: *Teoretické otázky neuronových sítí*. Praha, 1996, ISBN 80-85863-18-9, 390 s.

[10] ŽIŽKA, J.: Vybrané slajdy k předmetu Strojové učení. online, 2006.
URL <http://is.muni.cz/el/1433/podzim2006/PA034/09_SVM.pdf>