VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

DETEKCE ZAHALENÝCH TVÁŘÍ V OBRAZE
MASKED FACE DETECTION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Ondřej Malý**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Jiří Přinosil, Ph.D.**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Telekomunikační a informační technika**

Ústav telekomunikací

Student: Bc. Ondřej Malý ***ID:*** 174228

Ročník: 2 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Detekce zahalených tváří v obraze

POKYNY PRO VYPRACOVÁNÍ:

V rámci práce prostudujte stávající algoritmy pro detekci tváří osob v obrazovém signálu a stanovte jejich úspěšnosti při detekci plně i částečně zahalených osob (šátky, masky, kapuce apod.). Na základě získaných znalostí navrhněte pro vybraný algoritmus takové úpravy, aby byla úspěšná detekce zahalených osob výrazně vyšší. Dané úpravy do algoritmu zaimplementujte a ověřte na reálných snímcích. Algoritmus poté dále upravte, aby dokázal určit, zda je tvář zahalená či nikoli.

DOPORUČENÁ LITERATURA:

[1] ZAFEIRIOU, Stefanos; ZHANG, Cha; ZHANG, Zhengyou. A survey on face detection in the wild: past, present and future. Computer Vision and Image Understanding, 2015, 138: 1-24.

[2] MA, Mei; WANG, Jianji. Multi-View Face Detection and Landmark Localization Based on MTCNN. In: 2018 Chinese Automation Congress (CAC). IEEE, 2018. p. 4200-4205.

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 1.6.2020

Vedoucí práce: Ing. Jiří Přinosil, Ph.D.

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

UPOZORNĚNÍ:
Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

Abstrakt

Cílem práce je nastudovat a otestovat současné metody pro detekování tváře na zahalených tvářích a vyhodnotit výsledky. V první kapitole je teoreticky rozebráno 5 zvolených metod a v druhé kapitole jsou jednotlivé metody vyhodnoceny, jak pro soubor Wider Face, tak pro vlastní soubor fotek se zahalenými tvářemi. Následně je metoda Dlib CNN vylepšena pro lepší detekci zahalených tváří a přeprogramována pro detekci míry zahalení z testovaného obrázku

Klíčová slova

Detekce tváře, zahalení, konvoluční sítě, srovnání metod, wider face, Dlib, MTCNN, Haar

Abstract

The aim of this work is to study and test current methods for face detection on veiled faces and evaluate the results. In the first chapter, five selected methods are theoretically analyzed and in the second chapter the individual methods are evaluated, both for the Wider Face file and for the actual set of photos with veiled faces. Subsequently, the Dlib CNN method is improved for better detection of veiled faces and reprogrammed to detect the degree of veil from the tested image

Keywords

Face detection, veiled face, convolutional neural networks, method comparasion, wider face, Dlib, MTCNN, Haar

Bibliografická citace:

MALÝ, Ondřej. Detekce zahalených tváří v obraze. Brno, 2020. Dostupné také z: https://www.vutbr.cz/studenti/zav-prace/detail/126040. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce Jiří Přinosil.

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma Detekce zahalených tváří v obraze jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: **1.6.2020** ……

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce Ing. Jiřímu Přinosilovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne: 1. 6. 2020 ……

podpis autora

Obsah

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

S příchodem neuronových sítí a strojového učení se rozmohlo spoustu algoritmů, které mají za cíl detekovat v obrazové sekvenci určité objekty, ať už jde o různé předměty nebo lidi a jejich chování. Tato diplomové práce pojednává o metodách detekce lidské tváře, které zažívají obrovský rozmach jak v oblasti obecné detekce člověka z obrázku nebo videa, tak v oblasti zabezpečení různých druhů elektronických přístrojů nebo objektů. Hlavním cílem této práce je navrhnout změnu již existující metody a následně ji vylepšit tak, aby dosahovala co nejlepších výsledků při detekci zahalených tváři, ale zároveň nebyla příliš časově náročná při detekci. Současné metody detekce dosahují v případě nezahalených tváří velmi dobrých výsledků, ale v případě zahalených je výsledek nejistý. Jedním z cílů diplomové práce je ověřit, jak jsou na tom současně používané metody detekce tváří právě v oblasti detekce zahalených tváří.

V první kapitole jsou teoreticky rozebrány jednotlivé detekční metody a následně srovnány jednotlivé metody na souboru Wider Face pomocí benchmarku dostupném k otestování těchto metod.

V druhé kapitole jsou následně popsané metody vyzkoušeny na vlastním souboru fotek se zahalenými tvářemi a výsledky vyhodnoceny. Jsou také rozděleny do skupin, dle podobnosti jednotlivých fotek a vyhodnoceny výsledky metod pro jednotlivé skupiny a na konci vybrána a rozebrána metoda, která bude vylepšena v rámci diplomové práce.

Třetí část teoreticky rozebírá samotné zlepšení detekce zahalených tváří vybranou metodou, kdy byl detektor přetrénován s větším počtem fotek se zahalenými tvářemi, následně vylepšení samotného algoritmu.

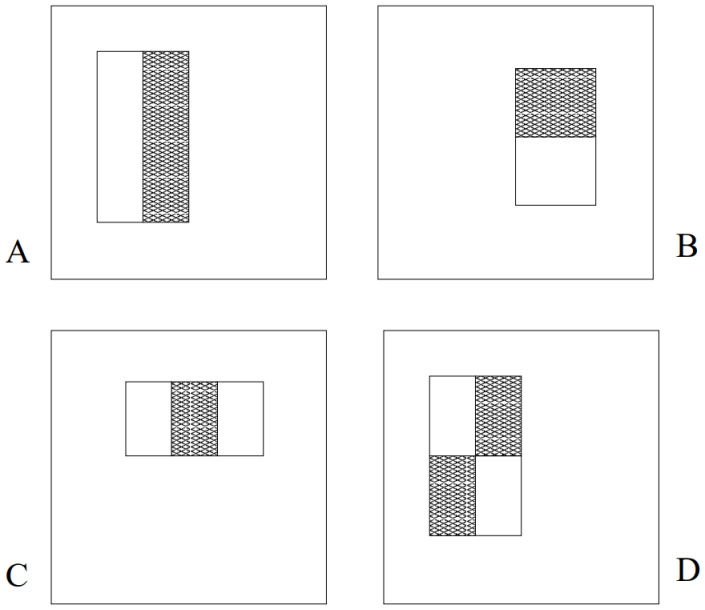
Čtvrtá část porovnává jednotlivé výsledky, a to jak samotného detektoru, tak schopnosti rozeznat míru zahalení. V rámci detekce zahalené tváře byla nejprve srovnána samotná detekce s ostatními metodami a následně srovnány jednotlivé skupiny zahalení a srovnáno v rámci těchto skupin. U rozpoznání míry zahalení byla vypracována matice záměn.

1. METODY DETEKOVÁNÍ TVÁŘE

Metod na detekování tváře je poměrně velké množství. Cílem této kapitoly je teoreticky srovnat vybrané metody, které budou následně prakticky vyzkoušeny na detekci zahalených tváří. Jako základní metoda byla vybrána Haarova kaskáda, která je brána jako referenční, jelikož šlo o první metodu z roku 2001, která je používána do dnes i v různých modifikacích. Dále byla vybrána populární MTCNN, Dlib, OpenFace a poměrně nová metoda TinyFace. Metoda MTCNN je založena na multi kaskádní konvoluční síti, jedná se o metodu, která dosahuje velmi vysoké rychlosti v případě detekce tváří. Metoda Dlib obsahuje dvě možnosti detekce, jedna je založena na principu histogramu orientovaných gradientů. Tento druh dlib detektoru dosahuje velmi dobrých výsledků v oblasti rychlosti detekce při výpočtu na procesoru počítače, celková přesnost ale není dle předpokladů nejlepší, naopak Dlib CNN detekuje ještě rychleji, ale poue v případě, že běží na grafické kartě od firmy Nvidia a je nainstalována knihovna Nvidia CUDA. Metoda OpenFace je známá především jako metoda pro rozpoznávání zahalených tváří, ale i pro samotnou detekci používá vlastní upravený detektor a dle testů dosahuje velmi dobrých výsledků v oblasti detekce nezahalených tváří. Metoda TinyFace podává výsledky především v oblasti detekce malých tváří v obraze. Velmi dobře dokáže rozpoznat i tváře o velikosti několika pixelů.V následujících podkapitolách budou jednotlivé metody teoreticky rozebrány.

1.1 Haarova kaskáda

Haarova kaskáda je algoritmus detekce objektů, který navrhli Paul Viola a Michael Jones ve své práci nazvané „Rapid Object Detection using Boosted Cascade of Simple Features“ v roce 2001. Úspěšnost této metody pro detekování tváří popsaná v práci je 93,9 %. V této práci byl algoritmus použit k detekování tváří z fotografií, je založený na strojovém učení, obraz je popisován pomocí Haarových příznaků. Existuje několik druhů těchto příznaků, jejich volba záleží na druhu vstupních dat. Nejpoužívanější druhy Haarových příznaků jsou znázorněny na obrázku 1.1.



Obrázek 1.1: Druhy Haarových příznaků

Typy A a B jsou základní příznak skládající se z dvou stejně velkých obdélníků, příznak je definován jako rozdíl součtu pixelů, které pokrývá černá plocha od součtu pixelů, které pokrývá bílá plocha. U typu příznaku C jsou opět sečteny pixely pod bílou plochou a odečteny od černého obdélníku, s tím rozdílem, že bílé obdélníky pokrývají dvakrát větší plochu a jsou okolními obdélníky černého obdélníku. V případě typu D počítá rozdíl mezi diagonálním párem obdélníků. Základní velikost vyhledávací oblasti je 24x24 pixelů, díky tomu je soubor poměrně rozsáhlý (okolo 180 tisíc příznaků).

Princip je takový, že čím větší je výsledek (rozdíl mezi černými a bílými obdélníky), tím větší je pravděpodobnost, že dané okno bude relevantním prvkem. [1] [14]

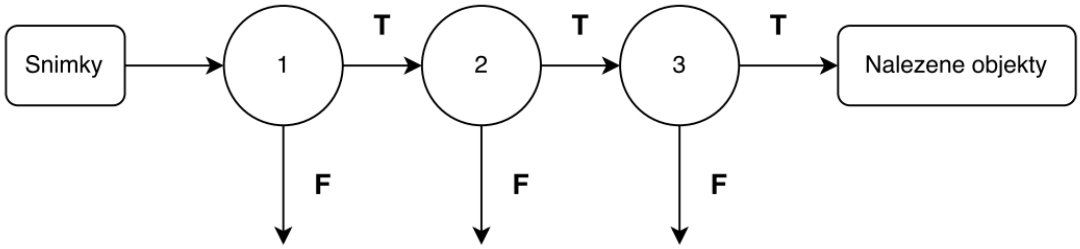
1.1.1 Integrální obraz

Integrální obraz je způsob digitální reprezentace obrazu tak, že každý bod x představuje součet hodnot předchozích pixelů doleva a nahoru. Jedná se o častý způsob výpočtu obdélníkových příznaků. [1]

1.1.2 Kaskádový klasifikátor

Klasifikaci lze definovat jako rozdělení obrázků a obrazových částí do tříd, v tomto případě dvou - pozitivní a negativní, podle definovaných pravidel. Kaskádový klasifikátor je tvořen souborem slabých klasifikátorů, kde každý slabý identifikátor je tvořen malým počtem příznaků. Větší množství slabých klasifikátorů je sloučeno do jednoho silného klasifikátoru. Velká část slabých klasifikátorů je také v každé úrovni kaskády zahozena. Tvorba kaskádového klasifikátoru je znázorněna na obrázku 1.2. Metoda využívající lineární kombinaci slabých lineárních klasifikátorů se nazývá Adaptive Boosting (AdaBoost). Hlavním důvodem zavedení kaskádového klasifikátoru je optimalizace výkonu programu a rychlejší nalezení objektů v obraze. Následuje

proces trénování, kdy jsou odstraněny negativní oblasti a pozitivní pokračují k dalšímu stupni klasifikátoru. [1] [14]



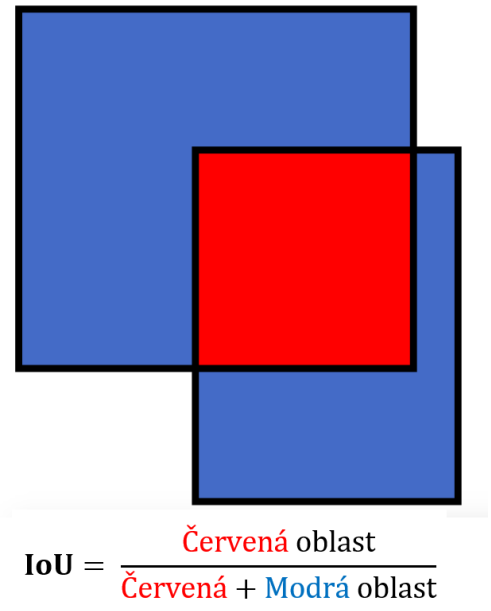
Obrázek 1.2: Tvorba kaskádového klasifikátoru

1.2 MTCNN

MTCNN je víceúlohová kaskádová konvoluční síť, která se skládá ze tří fází (konvolučních podsítí) – P-Net, R-Net a O-Net. Každá z těchto je trénována zvlášť, není možné je trénovat dohromady. [2]

1.2.1 Trénování

P-Net bere jako vstup obrázek o velikosti 12x12 pixelů, z něj následně počítá pomocí konvoluce maticový výsledek, z kterého je vyhodnoceno, zda se jedná o obličej či nikoliv. R-Net a O-Net fungují oproti P-Net, s tím rozdílem, že R-Net pracuje se vstupním formátem 24x24 a O-Net 48x48 pixelů. Výstupem P-Net a R-Net je soubor s umístěním ohraničujících boxů a pravděpodobnost. Výstupem O-Net je navíc soubor bodů určující tvář, jako jsou oči, ústa nos. Jelikož míra chybovosti první fáze P-Net je poměrně velká, vstupují pozitivně vyhodnocené části do další fáze R-Net a stejně tak pozitivní snímky z fáze R-Net vstupují do O-Net. Tato metoda se nazývá hard sample mining. Jako trénovací soubor obrázků pro tento algoritmus byl použit „WIDER-FACE“ datová sada k trénování ohraničení obličeje, který obsahuje přibližně 32 tisíc obrázků lidí v různých situacích. Následně je také použit datový soubor CelebA k trénování orientačních bodů definující obličej. Součástí trénovací množiny je i spousta situací, kdy lidé mají zahalené tváře. Algoritmus by tedy měl být schopen podávat dobré výsledky v této oblasti. K vygenerování dat byl použit volně dostupný algoritmus na GitHubu, který z této databáze obrázků dokáže vygenerovat potřebná data pro trénování. Funguje na principu Intersection over Union (IoU) a prediktivních ohraničujících boxech. IoU je vypočítáno mezi právě zkoušenou částí a částí obrázku predikovanou, kde průnik těchto částí je dělen jejich sjednocením, jak znázorňuje obrázek 1.3. Ideální hodnota je co nejblíže 1, ale za vyhovující je považován výsledek větší, než 0,5. [2]



$$\mathbf{IoU} = \frac{\text{Červená oblast}}{\text{Červená} + \text{Modrá oblast}}$$

Obrázek 1.3: Výpočet IoU

1.2.2 První fáze P-Net

Po předání vstupního obrázku programu dojde k vytvoření pyramidy obrázků, kdy je vstupní obrázek zmenšen do několika velikostí. Každou z těchto kopií prochází P-Net v oblasti 12x12 pixelů a hledá tvář. Začíná v levém horním rohu v pozici (0,0) do pozice (12,12) a postupně prochází celý obrázek. Další krok je dán vztahem (0+2a, 0+2b) do (12+2a, 12+2b), což znamená, že se vyhledávácí oblast postupně posouvá o 2 pixely v určitém směru. Krok posouvání o 2 pixely je zvolen pro redukci výpočetní náročnosti algoritmu, kdy s tímto krokem není snížena přesnost tohoto algoritmu, na druhou stranu je snížen počet výpočetních operací na čtvrtinu, což má za následek menší využití paměti a rychlejší chod programu. Pyramida obrázků je zvolena z důvodu stále stejné velikosti vyhledávací oblasti. Pokud je tvář v dáli, vyhledávací oblast 12x12 ho najde spíše ve středně velkém obrázku. Pokud je tvář ve velké části obrázku (člověk zobrazený na fotce je blízko), bude jeho tvář nalezena až v jednom z nejmenších obrázků. Poté je nutné výsledky z P-Net analyzovat, abychom získali seznam pravděpodobností pro každý ohraničující rámeček. Je nutné se zbavit rámečků s nízkou pravděpodobností a ponechat pouze ty s vysokou. I po odstranění rámečků s nízkou pravděpodobností jich však zbývá velké množství a mnoho z nich se překrývá. Nelze ovšem vzít rámeček s největší pravděpodobností a všechny ostatní smazat, jelikož se v obrázku může vyskytovat více než jeden obličej. Proto se k odstranění redundantních rámečků používá metoda NMS (Non-Maximum Suppression). Jde o techniku vymazání překrývajících se rámečků a ponechání těch, které se nepřekrývají s rámečkem s největší pravděpodobností. Dále je také nutné převést souřadnice ohraničujícího rámečku na

reálné souřadnice pro velikost obrázku, který byl do programu nahrán, protože byly nalezeny ve zmenšeném obrázku. Nakonec uložíme souřadnice ohraničovacích rámečků a předáme je do fáze 2 (R-Net). [2] [11]

1.2.3 Druhá fáze R-Net

Pixely, které jsou ohraničeny pomocí rámečků v originálním obrázku ve fázi P-Net jsou zkopírovány do nového pole. Může se stát, že obličej na fotce není kompletní, je ho vidět například polovina na hraně obrázku a byl detekován první fází, pak zasahuje ohraničující rámeček mimo obrázek. V takovém případě je zkopírována viditelná část a je doplněna nulami v rámci ohraničujícího rámečku. Současné hodnoty pixelů jsou v rozsahu (0, 255), což odpovídá hodnotám RGB, pro potřeby R-Net se ale převádí do rozsahu (-1,1), čehož je docíleno pomocí vztahu . Hodnota 127,5 je zvolena jako polovina z RGB rozsahu.

$$P_{(-1,1)} = \frac{P_{(0,255)} - 127{,}5}{127{,}5} \tag{1.1}$$

Dále je každý rámeček převeden na velikost 24x24 a dojde k procesu vyhledávání. Poté, jako v první fází, dochází k třídění rámečků pomocí NMS, nalezené rámečky v této fázi převedeny do původního rozlišení a dále jsou tyto data předány do další fáze O-net. [2] [11]

1.2.4 Třetí fáze O-Net

Třetí fáze není příliš odlišná od těch předchozích s tím rozdílem, že vstupní data musí být převedeny na velikost 48x48. Další odlišností je výstup z O-Net, zde není výstupem pouze rámeček ohraničující tvář, ale také souřadnice obličejových orientačních bodů (oči, nos, ústa) a pravděpodobnost správnosti každého boxu. Dále je postup stejný, dochází k redukci rámečků pomocí NMS, kdy v této fázi by měl existovat pouze jeden pro každý obličej. Následuje přepočítání rámečku na originální rozlišení a zobrazení do výsledného souboru. [2] [11]

1.3 TinyFaces

Většina detektorů nemění měřítko vyhledávácí oblasti, které používá k vyhledávání, což má za následek špatnou detekci vzdálenějších objektů v obrázku. Z toho důvodu vznikl algoritmus TinyFaces, který byl napsán v prostředí Matlab a dosahuje dobrých výsledků při detekci tváří vzdálených stovky metrů od objektivu. Byl testován na fotce s 1000 tváří v různých vzdálenostech, kdy jich bylo úspěšně detekováno 800. Z důvodu odlišnosti detekce tváře, která je v dáli a výška takového obličeje je například 10 pixelů, od tváře vysoké 400 pixelů, je v tomto algoritmu použito více detektorů se specializací na určitou velikost. Vzhledem k přesnějšímu a podrobnějšímu vyhledávání by mohla být úspěšnost tohoto algoritmu větší i pro zahalené tváře. V rámci detekce tváře jsou

zkoumány tři aspekty: měřítko hledané tváře (z toho pramenící druh detektoru), rozlišení obrázku a kontext obrázku. Kontext obrázku určuje za jakých okolností byla fotka focena. Princip měřítka a kontextu obrázku je zobrazeno na obrázku 1.4. [3]

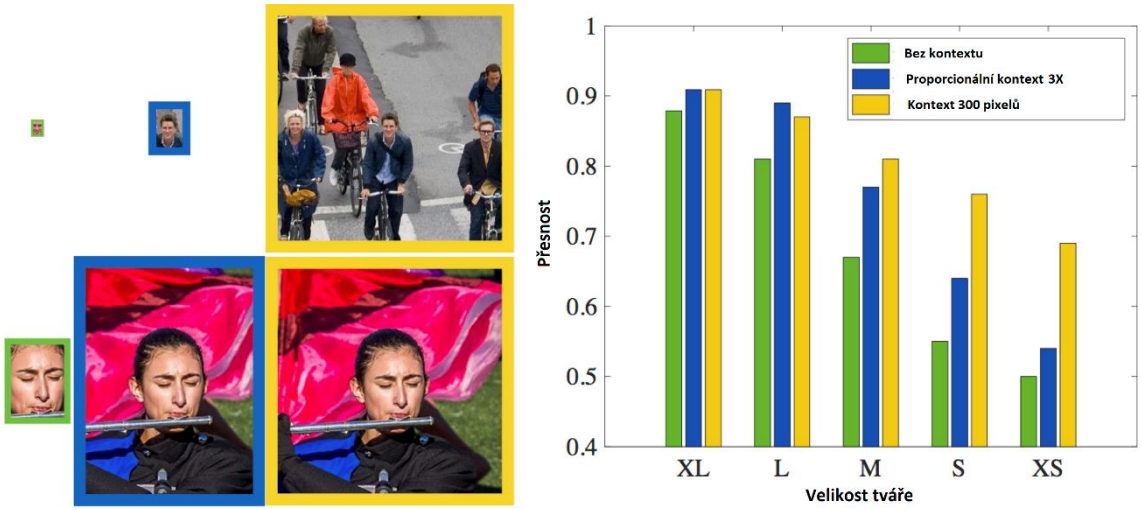
Obrázek 1.4: Určení měřítka a kontextu [3]

1.3.1 Využití vytrénovaných množin

TinyFaces používá vytrénovanou množinu ImageNet. Problém tohoto modelu je, že většina vytrénovaných dat je čtverec v rozmezí 40-140 pixelů, všechny obrázky jsou velikosti přibližně 250x250 pixelů, což znamená, že model je vytrénován pro tento rozsah. Pro lepší rozpoznání větších i menších tváří byly vytrénované data rozšířena o měřítko 0.5x pro nalezení tváří větších, než 140 pixelů a měřítko 2x pro nalezení těch menších, než 40 pixelů. Standartní měřítko bylo zachováno. [3]

1.3.2 Určení kontextu

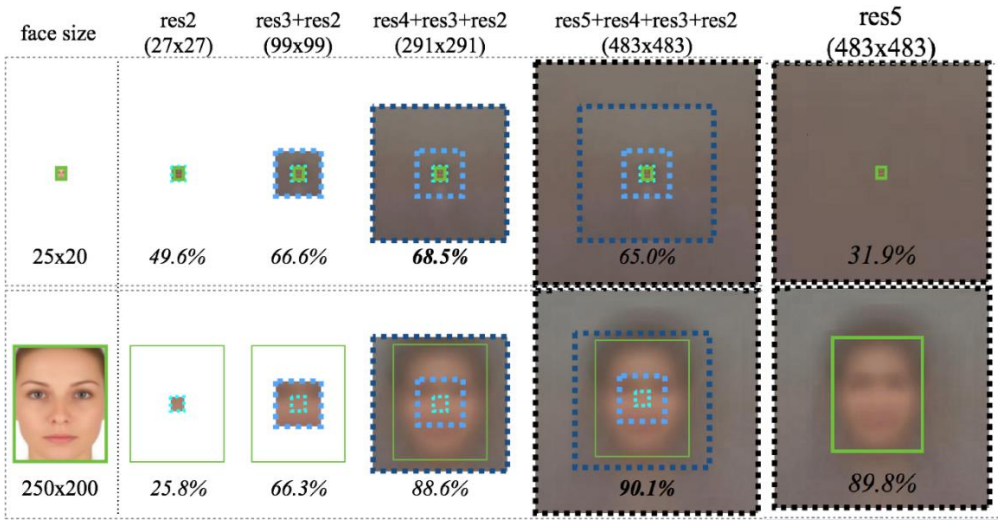
Kontext výrazně pomáhá s detekcí menších tváří jak pro algoritmus, tak pro lidské vidění. Určit lidskou tvář o malé velikosti je pro lidské vidění výrazně snadnější, pokud lze vidět okolí tváře a zasadit ji do prostředí nebo činnosti. Autoři algoritmu provedli výzkum, kde vybraní testující uživatelé určovali, zda navržený detektor určil dobře nebo špatně vybrané pozitivní snímky. Z průzkumu je patrné, že přidání kontextu se pozitivně projevilo na správném vyhodnocení menších obrázků, kdy přidání pevného kontextu redukovalo chybovost u malých obrázků (S) o 20 %. Na velkých obrázcích je vliv kontextu minimální. Více z tohoto průzkumu lze vyčíst na obrázku 1.5.



Obrázek 1.5: Vliv kontextu na správnost detekce [3]

TinyFaces používá tzv. *Foveal descriptors*, které jsou podstatné pro detekci malých tváří a fungují na podobném principu, jako lidské vidění, kdy jsou poskytovány ostré detaily pro malou část obrázku kolem obličeje a zbytek je rozostřený. To poskytuje dostatečnou kapacitu dat pro zasazení obrázku do kontextu, ale neklade příliš vysoké výpočetní nároky na počítač.

Princip těchto deskriptorů je znázorněn na obrázku 1.6. Zelené pole reprezentuje aktuální velikost tváře a tečkované čtverce znázorňují pole spojené s jinou vrstvou. Jednotlivé barevné čtverce reprezentují určitou vrstvu, res2 - tyrkysová, res3 - světle modrá, res4 – tmavě modrá, res5 – černá. Z obrázku je patrné, že při velké tváři je odstranění těchto deskriptorů skoro neznatelné, dojde k zhoršení přesnosti o 1,3 %, ovšem při malé tváři je dojde při odstranění deskriptorů o 33 %, což je znatelný rozdíl. [3]



Obrázek 1.6: Využití foveal descriptors [3]

1.4 Dlib

Dlib je multiplatformní softwarová knihovna napsaná v jazyce C++. Pro převod do Pythonu se používá knihovna Cmake. Dlib obsahuje předpřipravený natrénovaný model a další nástroje pro detekci obličeje v obrázku. Používá dvou principů, jeden je založen na principu HOG (histogram orientovaných gradientů) a SVM (support vector machine), druhý na CNN (konvoluční neuronová síť). [4]

1.4.1 Dlib HOG + SVM

HOG je detektor založený na příznacích, kterými jsou histogramy orientovaných gradientů, tedy histogramy orientace hran. Při detekci detektor na vstupní obrázek aplikuje okno, které postupně prochází celý obrázek. Pokud se okno z velké části shoduje s některým z těch natrénovaných, prohlásí se za kandidáta. Poté se seskupí podobná místa a z nich se vypočtou poloha a velikost detekovaného objektu. SVM je metoda pro naučení detektoru. Při poskytnutí trénovacích vstupů je SVM schopný vytvořit model, který po přijmutí nového vstupu určí jeho příslušnost do jedné ze dvou natrénovaných tříd.

Trénovací data, použitá k natrénování tohoto modelu, obsahují 2825 obrázků, které byly získány z LFW databáze. Tato metoda detekce umožňuje detekci v reálném čase a je schopná najít i mírně natočené tváře, ale není vhodný pro detekci malých tváří. [4]

1.4.2 Dlib CNN

Architektura neuronové sítě z této knihovny umožňuje procházet obrázek a pro každou jeho část zjišťuje, zda se na ní nachází hledaný objekt. Dlib obsahuje model pro detekci lidských obličejů založený na CNN. Nepoužívá pohyblivé okno, takže není nutné vstupní obrázek rozdělovat na menší okna. Jako návratovou hodnotu vrací mmod_rectangle objekt, který obsahuje dvě proměnné – dlib.rectangle a pravděpodobnostní skóre. Výstupem knihovny je pravděpodobnost, že se právě na tomto místě nachází shoda. Je přesnější proti HOG metodě, ale vyžaduje více výkonu a výpočet trvá déle. Model byl vytrénován na datasetu čítajícího 7220 obrázků a je velmi robustní. Detekuje tváře natočené do různých směrů i tváře s okluzí. Tento model běží velmi rychle na grafické kartě, ale je velmi pomalý při výpočtu na procesoru. [4]

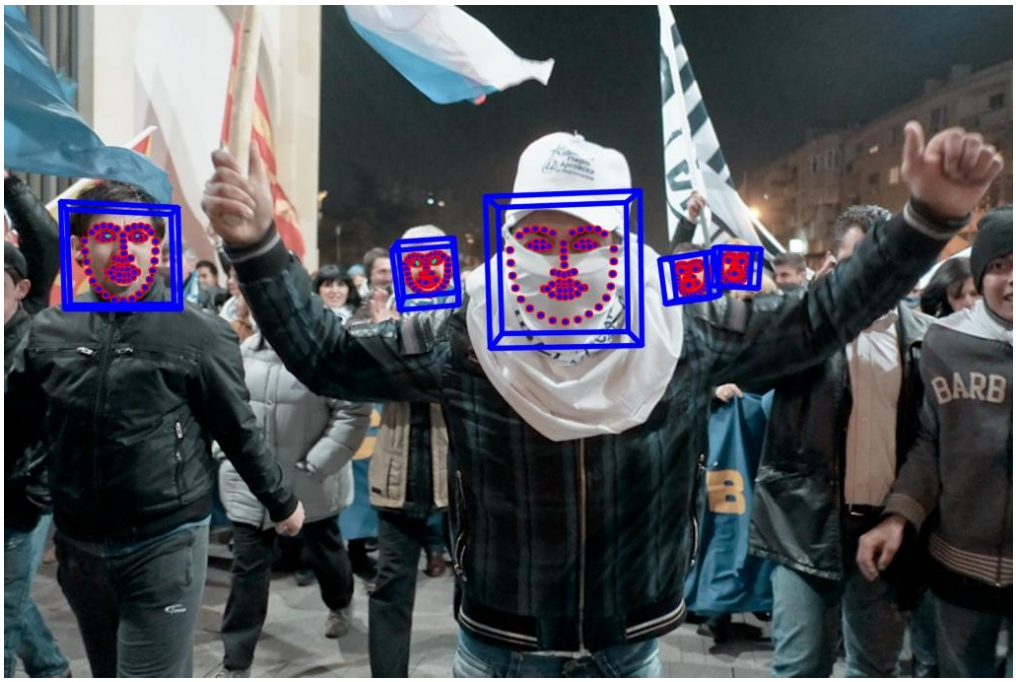
1.5 OpenFace

OpenFace je *opensource* nástroj pro detekci tváře a analýzu jejího chování, který je založen na metodě FaceNet. Využívá matematickou knihovnu pro Python Numpy a knihovnu OpenCV, které slouží pro zpracování a transformaci obrazu. Schopnost tohoto nástroje je mimo detekce tváře také detekovat klíčové body obličeje (landmarky), odhadnout náklon a natočení hlavy ve všech osách, směr pohledu. Tento nástroj dosahuje velmi dobrých výsledků ve všech těchto odvětvích a může být schopným

nástrojem i pro detekci zahalených tváří. Poskytuje výsledky prakticky v reálném čase, je možné jej použít i na detekci z videa nebo kamery v reálném čase.

Sledování pohybu hlavy probíhá pomocí systému Watson, který je součástí modelu založeném na AdaptiveView kamerách. Pro přesné sledování pohybu je nutné OpenFace předat parametry kamery. Pokud nejsou k dispozici, je výsledkem pouze hrubý odhad. Na detekci pohledu očí je použit CLNF *(Conditional Local Neural Fields)*, který se používá na detekci známých orientačních bodů, jako jsou oční víčka, zornice a duhovka. Mimo detekce orientačních bodů, sledování hlavy a pohledu očí dochází ještě k detekci akční jednotky. [12]

Na obrázku 1.7 je možné vidět, jak vypadá vykreslení jednotlivých detekovaných částí ve vstupním obrázku. Modrá krychle určuje trojrozměrný směr naklonění a natočení hlavy na fotce. Tečkovaně je zobrazen odhad obličeje a zeleně směr pohledu, kam se člověk na fotce dívá. [5] [13]



Obrázek 1.7: Detekce OpenFace

2. TESTOVÁNÍ METOD

Popsané metody v kapitole 1 byly otestovány na souboru 300 fotek se zakrytou nebo zahalenou tváří z různých odvětví, například různé zakrytí šátkem, šálou, helmou na lyžích nebo při hokeji, vojáci, hasiči s přilbou, nevěsty se závojem, fotografové se zakrytou části tváře fotoaparátem a další. Pokrytí různých odvětví, kdy obličej není kompletní nebo je něčím překrytý, je poměrně velké. Každá z metod vyhodnotila jednotlivé fotky a u každé fotky mohlo nastat tři stavy, dle kterých byla detekce klasifikována:

- *True positive (TP)* – metoda správně detekovala tvář v obrázku
- *False positive (FN)* – metoda špatně detekovala tvář, která ve vyznačeném místě není
- *False negative (FN)* – metoda nedetekovala tvář v obrázku, která se v něm nachází

Vzhledem k tomu, že soubor fotek obsahuje pouze fotky s obličejem, není nutné uvažovat *True negative* (tedy správně nedetekované tváře na fotkách, které žádnou tvář neobsahují). Po klasifikaci každé z fotek je možné vypočítat následující parametry jednotlivých metod:

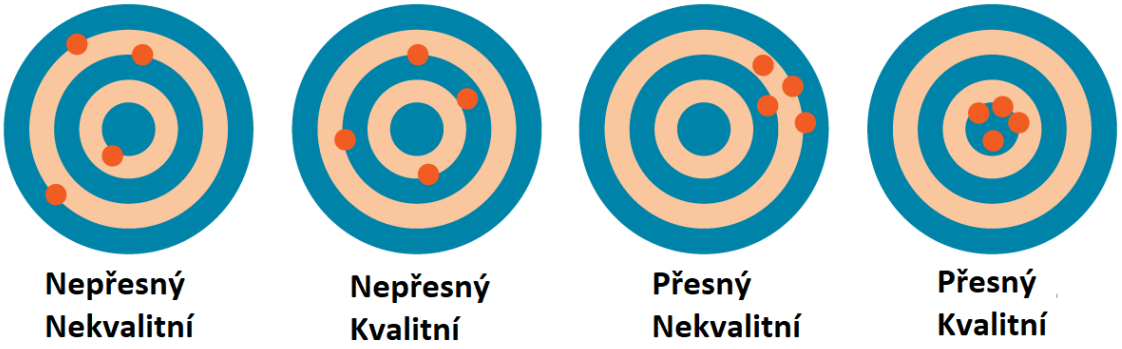
- Přesnost (Accuracy)

$$A = \frac{TP}{TP+FP+FN} \tag{2.1}$$

- Preciznost (Precision)

$$P = \frac{TP}{TP+FP} \tag{2.2}$$

Ze vztahů lze vyčíst, že zatímco kvalita je závislá i na nevyhodnocených tvářích v obraze, tak přesnost na nich nezávisí. To značí, že zatímco kvalita definuje celkové fungování metody na detekci tváří, přesnost nehodnotí neschopnost rozpoznání tváře, ale pouze správnost těch rozpoznaných. Obecný rozdíl mezi kvalitou a přesností je znázorněn na obrázku 2.1.



Obrázek 2.1: Obecné srovnání přesnosti a kvality

2.1 Srovnání jednotlivých metod na souboru Wider Face

V následující kapitole budou jednotlivé metody srovnány na souboru fotek Wider Face [8], který je volně dostupný ke stažení. Obsahuje přes 32 tisíc fotek s více než 390 tisíci tvářemi s velkým rozsahem velikostí, situací, za kterých byly fotky pořízeny, s různým natočením hlavy. Soubor Wider Face je rozdělen do 61 složek (kategorií). Autoři tohoto souboru také obsahuje na svých stránkách srovnání velkého množství metod, odkud jsou data čerpány. Výsledky jsou srovnány v tabulce 2.1.

Tabulka 2.1: Srovnání metod na souboru Wider Face

Tabulka srovnání na souboru Wider Face	
Metoda	**Přesnost**
Haarova kaskáda	0,307
MTCNN	0,517
Dlib HOG	0,365
Dlib CNN	0,416
OpenFace	0,45
TinyFaces	0,819

Z tabulky lze vyčíst jednotlivé předpokládané přesnosti pro těchto šest vybraných metod. Nejlépe by dle tohoto testu měla dopadnout metoda TinyFaces, následuje MTCNN a Dlib CNN a OpenFace. Nejhůře by dle předpokladů měla dopadnout Haarova kaskáda a Dlib HOG.

2.2 Vybrané fotky

Do testovací množiny souborů bylo vybráno 300 fotografií z různých žánrů. Přibližně 150 fotek bylo použito z volně dostupné množiny fotek WiderFace, kde jsou fotky roztříděny do různých skupin a je možné zde nalézt několik skupin fotek se zahalenou tváří. Dalších 150 fotek bylo staženo z internetu (převážně Google obrázků).

První skupinu fotek lze charakterizovat jako fotky, kde jsou zakrytá ústa, převážně šátky, burky, kukly, roušky atd. Výběr 4 fotek ze souboru lze vidět na obrázku 2.2.

Obrázek 2.2: Šátky, roušky, kukly

Další skupinou mohou být různé lyžařské, policejní a vojenské přilby, dále hokejové přilby, masky brankářů apod. Čtyři vybrané fotky jsou vidět níže na obrázku 2.3

Obrázek 2.3: Různé druhy helem

Poslední skupinou mohou být všechny ostatní kategorie – například různé karnevalové masky, kostýmy nebo různé předměty, které mohou zakrývat obličej, zde znázorněno jako fotograf, který má zakrytou půlkou obličeje při focení

Obrázek 2.4: Poslední skupina: Kostýmy, masky atd.

2.3 Vyhodnocení dat

Tabulka 2.2: Vyhodnocení detekce pro jednotlivé metody

	Haarova kaskáda	MTCNN	Dlib
Přesnost	0,108	0,406	0,179
Preciznost	0,692	0,922	0,897
	Dlib CNN	**TinyFaces**	**OpenFace**
Přesnost	0,438	0,900	0,579
Preciznost	0,960	0,952	0,957

Z tabulky je patrné, že nejhůře dopadla Haarova kaskáda, která dosahuje zhruba 10,5 % úspěšnosti v detekci zahalených tváří a přesnost této metody také není vysoká. Jedná se však o jeden z prvních algoritmů na detekci tváře z roku 2001, takže výsledky jsou očekávatelné. Podobného výsledku dosáhl Dlib v oblasti kvality (17,6 %), ale tato metoda dosáhla výrazně vyšší přesnosti proti Haarově kaskádě. Výsledky MTCNN, Dlib CNN a OpenFace metod jsou velmi podobné v oblasti přesnosti a s odchylkou 14 % mezi nejhorším Dlib CNN a nejlepším OpenFace, jsou si podobné i co se kvality týče. Nejlépe dopadla metoda TinyFaces, která dosáhla nejlepších výsledků ze všech testovaných metod s kvalitou rovnou přibližně 90 %.

Zajímavým faktem může být také roztřídění podobných fotek do skupin dle situace, v které byla vyfocena. To je znázorněno v tabulkách 2.2, 2.3 a 2.4, kde jsou fotky roztříděny do tří skupin a následně je pro každou metodu porovnáno, jak dokázala detekovat tváře v této skupině fotek. Z tabulky lze vyčíst, že Dlib a Haarova kaskáda mají velký problém při detekci brýlí a různých druhů přileb a helem. Dlib CNN také podává u této kategorie horší výsledky, než u ostatních skupin, ale rozdíl není tak velký, jako u dvou předchozích. Ostatní metody předvádí konstantní výsledky ve všech třech skupinách. Co se týče nejlepší detekce v jednotlivých skupinách, pořadí je stejné jako v tabulce 2.1. V případě Haarovy kaskády podává stejné výsledky pro zahalení šátky a čepice, v případě helem je detekce prakticky nulová. Metoda MTCNN podává stabilní

výsledky napříč všemi skupinami. V případě metody Dlib HOG je situace podobná, jako v případě Haarovy kaskády, kdy podává stejné výsledky ve skupině Šátky a brýle + čepice, na druhou stranu vykazuje velkou chybovost v případě helem. Metody Dlib CNN, OpenFace a TinyFaces podávají stabilní, stejně kvalitní detekci napříč skupinami, žádná skupina nevykazuje přehnanou chybovost.

Tabulka 2.3: Detekce metod pro šátky, šály atd.

Šátky, burky, šály, roušky, závoje									
	Haar			MTCNN			Dlib		
	TP	FP	FN	TP	FP	FN	TP	FP	FN
Součet	31	8	129	65	2	98	39	3	122
Přesnost		0,185			0,394			0,238	
Preciznost		0,795			0,970			0,929	
	Dlib CNN			TinyFaces			OpenFace		
	TP	FP	FN	TP	FP	FN	TP	FP	FN
Součet	78	3	83	152	5	12	103	3	61
Přesnost		0,476			0,899			0,617	
Preciznost		0,963			0,968			0,972	

Tabulka 2.4: Detekce metod pro přilby a helmy s brýlemi

Přilby+lyžařské brýle, hokejové helmy, policejní/vojenské helmy									
	Haar			MTCNN			Dlib		
	TP	FP	FN	TP	FP	FN	TP	FP	FN
Součet	5	4	133	59	7	78	9	0	129
Přesnost		0,035			0,410			0,065	
Preciznost		0,556			0,894			1,000	
	TP	FP	FN	TP	FP	FN	TP	FP	FN
	Dlib CNN			TinyFaces			OpenFace		
Součet	48	2	92	135	5	6	82	1	56
Přesnost		0,338			0,925			0,590	
Preciznost		0,960			0,964			0,988	

Tabulka 2.5: Detekce metod pro masky, kukly atd.

Plavecké čepice+brýle, masky, zlodějské kukly									
	TP	FP	FN	TP	FP	FN	TP	FP	FN
	Haar			MTCNN			Dlib		
Součet	20	8	115	54	4	79	39	5	93
Přesnost		0,140			0,394			0,285	
Preciznost		0,714			0,931			0,886	
	TP	FP	FN	TP	FP	FN	TP	FP	FN
	Dlib CNN			TinyFaces			OpenFace		
Součet	74	3	60	130	7	10	79	6	57
Přesnost		0,540			0,884			0,556	
Preciznost		0,961			0,949			0,929	

2.4 Výběr metody

Z otestovaných metod je třeba vybrat jednu, která bude následně přetrénována, popřípadě vylepšena a přeprogramována, aby bylo dosáhnuto co nejlepších výsledků při detekci zahalených tváří. Jako první bylo uvažováno MTCNN, které se skládá ze tří vrstev, kde přicházelo v úvahu přetrénování poslední vrstvy O-Net. To by bylo možné za předpokladu, že by správně detekované tváře zahazovala až tato vrstva, například kvůli nízké pravděpodobnosti. V tabulce 2.5 je srovnání dat vstupujících do poslední fáze O-Net a celkovému výstupu z MTCNN pro prvních 100 fotek. Dle testu bylo dokázáno, že se tak ve velké míře děje, ale spočítaná maximální dosažitelná kvalita přibližně 85 % není dostačující. Přetrénování všech tří vrstev je náročné a tudíž je metoda MTCNN nevyhovující.

Tabulka 2.6: Srovnání vstupu do O-Net s celkovým výstupem

MTCNN vstup do O-Net			Výstup z MTCNN		
TP	FP	FN	TP	FP	FN
151	0	29	77	8	103
0,84			0,41		

Při vybírání nebude dbáno pouze na nejlepší výsledky v předchozí části, ale také k času, který si program vezme při detekci. Z tohoto důvodu byl zavrhnut nejlépe vycházející TinyFace. Tuto metodu, společně s Dlib CNN bylo možné spustit na grafické kartě, tudíž lze porovnávat čas, který daná metoda spotřebovala pro detekci testovaných 300 fotek. Jako grafická karta byla použita Nvidia GeForce GTX 1050Ti 4 GB a metody běžely na hardwarové architektuře Nvidia CUDA. Při srovnání časové náročnosti při detekci 300 fotek běžela metoda Dlib CNN 2 minuty a 15 sekund, což je v průměru 1,16 vteřiny na jednu fotku. TinyFace běžel 21 minut a 27 sekund, což je v průměru 4,3 vteřiny na jednu fotku. Je zřejmé, že TinyFace je velmi časově náročná metoda a z tohoto důvodu také nebude vybrána.

Jako vhodná metoda pro další postup se tedy spíše jeví Dlib CNN, která běží také na grafické kartě, je rychlá a v rámci hodnocení dosáhla poměrně dobrých výsledků.

3. TRÉNOVÁNÍ DLIB

Dlib je volně dostupný software, který obsahuje spoustu nástrojů pro detekci a rozpoznání tváře. Testovaný software v kapitole 2 byl provozován v jazyce Python, nicméně pro samotné trénování je potřeba nainstalovat verzi pro C++, jelikož Python neumožňuje nastavování druhů konvoluční sítě, které jsou používány pro vytrénování detektoru tváří a veškerých nástrojů, které budou potřeba. Dlib využívá grafického akcelerátoru Nvidia CUDA ve verzi 9.2. Tyto knihovny jsou volně dostupné na stránkách Nvidia pro vývojáře. Jelikož dlib používá kompilátor z Visual Studio 14, bylo nutné nainstalovat i tento vývojářský nástroj a také program Cmake, který umožňuje samotný překlad programu.

3.1 Rozšíření skupiny fotek

V rámci datasetu bylo použito skupiny fotek od vývojáře tohoto software, který je volně dostupný ke stažení z jeho stránek. Obsahuje přibližně 5 tisíc fotek, které jsou připraveny pro vstup programu. Obsahují asi 6000 tváří z několika skupin, různých ras a v různých pozicích. Tato skupina byla rozšířena o vlastní vybrané fotky se souboru Wider Face. Z tohoto souboru bylo nakonec vybráno přibližně 5000 fotek z různých kategorií se zahalenými tvářemi, které bylo nutné upravit pro vstup do programu, který dělá trénování. K tomu slouží nástroj imglab, který je součástí dlib. Princip nástroje spočívá v tom, že načte veškeré fotky uživatel za pomoci tlačítka shift a levého tlačítka myši označuje ohraničující boxy kolem tváří. Toto bylo potřeba udělat pro celou sadu fotek. Výstupem tohoto nástroje je XML soubor, který se poté předá programu pro trénování. Princip tohoto nástroje znázorňuje obrázek 3.1.

Obrázek 3.1: Nástroj imglab

3.2 Proces trénování detektoru

K vytrénování samotného souboru MMOD objektů, který je poté předán detektoru, je využito konvolučních sítí. Nejdříve je vypočítána 5x5 konvoluční síť con5d, která provádí podvzorkování. Následně je provedena stejná 5x5 síť con5, ale bez podvzorkování. Poté je vytvořen blok podvzorkovaných (8x) con5d sítí. Nakonec je vytvořena celková síť. Funkce input_rgb_image_pyramid zajišťuje, že u vstupního obrázku nezáleží na měřítku a celkově jeho velikosti, jelikož je vytvořena celá pyramida obrázků. Popisovaný kód je zobrazen níže.

```
template <long num_filters, typename SUBNET> using con5d = con<num_filters, 5, 5, 2,
2, SUBNET>;
template <long num_filters, typename SUBNET> using con5 = con<num_filters, 5, 5, 1,
1, SUBNET>;
template <typename SUBNET> using downsampler = relu<bn_con<con5d<32,
relu<bn_con<con5d<32, relu<bn_con<con5d<32, SUBNET>>>>>>>>>;
template <typename SUBNET> using rcon5 = relu<bn_con<con5<45, SUBNET>>>;
using net_type = loss_mmod<con<1, 9, 9, 1, 1,
rcon5<rcon5<rcon5<downsampler<input_rgb_image_pyramid<pyramid_down<6>>>>>>>>;
```

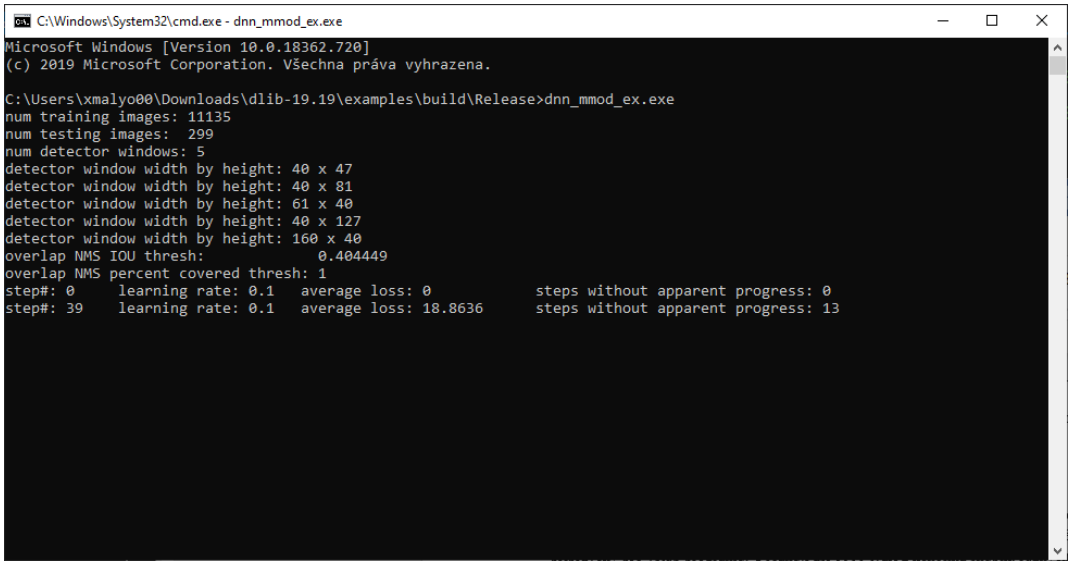
Následně je načtena složka fotek, která byla předána programu jako vstup a poté z XML souboru načteny ohraničující boxy, které byly vytvořeny nástrojem v předchozí kapitole. Poměrně problematické je, že veškeré fotky načítá program do paměti RAM. Na počítači, na kterém bylo provedeno trénování bylo k dispozici 16 GB RAM. Při počtu přibližně 11000 fotek vychází na jednu fotku průměrně velikost 1,77 MB na jednu fotku. Při trénování bylo využito přibližně 15 GB RAM. Je zřejmé, že v případě potřeby rozšíření setu fotografií by bylo nutné buď rozšířit RAM nebo provést kompresi fotografií, čímž by mohla být narušena samotná kvalita vytrénování. Je také možné uvažovat přesunutí fotek z RAM a komunikovat s diskem. Nicméně i v případě rychlých SSD se dle autora prodlužuje čas trénování o desítky hodin, proto je to řešeno tímto způsobem. Poté je definována velikost okna, v tomto případě bylo použito okno o velikosti 40x40.

```
mmod_options options(face_boxes_train, 40, 40);
```

Následuje nastavení prahu pro zastavení trénování. V rámci trénování je po každé skupině kroků spočítáno, kolik iterací bylo vykonáno bez pokroku a v případě překročení nastavené meze ukončí proces trénování a zapíše finální soubor do stejné složky, z jaké byl spouštěn. Dle doporučení byl nastaven na hodnotu 4000.

```
trainer.set_iterations_without_progress_threshold(4000);
```

Samotné trénování probíhá na grafické kartě, která je v tomto případě Nvidia GTX 1050Ti 4 GB a trvalo přibližně 5-6 hodin. Princip trénování je zobrazen na obrázku 3.2. Během trénování lze pozorovat v jakém kroku se právě trénování nachází, *learning rate*, který se postupně zmenšuje, *average loss*, jehož hodnota by po trénování měla být co nejmenší a kroky bez pokroku.



Obrázek 3.2: Proces trénování v příkazovém řádku

Jelikož samotná detekce s touto konvoluční sítí a s rozšířeným souborem fotografií dosahovala lepších výsledků, než vytrénovaná standartně vytrénovaný detektor v dlibu, testovaný v kapitole 2, ale celkový výsledek nebyl stále dostatečný (samotné výsledky budou konzultovány v následující kapitole), bylo možné uvažovat rozšíření souboru, což nebylo možné z důvodu omezení použitého HW, jak bylo popsáno výše.

Z toho důvodu se v rámci zlepšení výsledků detekce uvažovalo o zavedení dalšího stupně detekce s rozdílně nastavenou konvoluční sítí, což se nakonec ukázalo jako výrazné zlepšení samotné detekce. U této konvoluční sítě byla změněna prvotní konvoluční síť, která byla zmenšena na 3x3. Celá konvoluční síť je popsána níže.

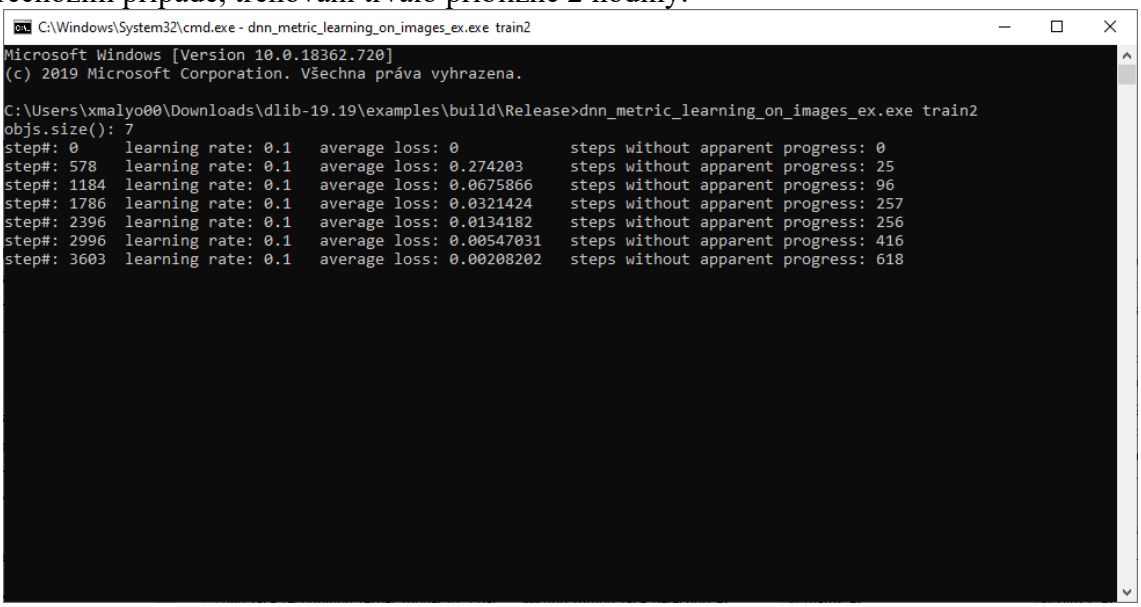
```
template <long num_filters, typename SUBNET> using con5d = con<num_filters,
5, 5, 2, 2, SUBNET>;
template <long num_filters, typename SUBNET> using con3 = con<num_filters, 3,
3, 1, 1, SUBNET>;
template <typename SUBNET> using downsampler = relu<affine<con5d<32,
relu<affine<con5d<32, relu<affine<con5d<16, SUBNET>>>>>>>>>;
template <typename SUBNET> using rcon3 = relu<affine<con3<32, SUBNET>>>;
using net_type1 = loss_mmod<con<1, 6, 6, 1, 1,
rcon3<rcon3<rcon3<downsampler<input_rgb_image_pyramid<pyramid_down<6>>>>>>>;
```

3.3 Proces trénování rozpoznání stupně zahalení

Dalším stádiem v rámci detekce bylo zjistit míru zahalení detekovaného obličeje. Nejdříve bylo třeba připravit si soubor fotek pro trénování. V kořenové složce je nutné seřadit fotky do jednotlivých kategorií, opět pomocí složek. Fotky musí obsahovat tvář patřící do dané skupiny a musí být ve formátu .jpg, velikost 150x150. Způsob seřazení vyjadřuje tato struktura. Rozřazením fotek do složek je realizován binární klasifikátor, kdy každou složkou určuji, které fotky patří k sobě do skupiny a které fotky patří do jiné.

Program nejdříve načte připravený soubor fotek a načte jednotlivou adresářovou strukturu, z které je zřejmé rozřazení jednotlivých fotek. Následuje podobný princip, jako v případě trénování na detekci tváře. Trénovací obrázky prochází konvoluční sítí, která je mírně rozdílná oproti předchozímu případu, kdy se samotná síť skládá z jednotlivých úrovní různých konvolučních sítí.

Následně probíhá samotné trénování. Tento program pracuje na principu K nejbližších sousedů, tedy vytvoření skupin sousedů na základě podobnosti, kdy velmi podobné fotky jsou shlukovány velmi blízko u sebe v prostoru na základě jejich podobnosti, zatímco rozdílné jsou od nich v prostoru vzdáleny. Výsledkem je opět soubor, který je předán později při detekci, na jehož základě je schopen později při detekci určit míru zahalení. Na obrázku 3.3 je znázorněn proces trénování. Princip je podobný, jako v předchozím případě, jsou prováděny jednotlivé kroky, dále *learning rate, average loss* a *steps without progress.* Ty jsou opět nastaveny v programu, jako v předchozím případě, zde nastaven práh ukončení programu na 12000. Doba trénování byl nižší, než v přechozím případě, trénování trvalo přibližně 2 hodiny.



Obrázek 3.3: Proces trénování rozpoznání stupně zahalení

3.4 Detekce tváře a rozpoznání zahalení

V rámci detekce je nutné programu předat několik souborů. Dva vytrénované soubory na detekci tváře z dvou konvolučních sítí a jeden soubor pro rozpoznání druhu zahalení. Ty to soubory jsou načteny automaticky a není nutné je předávat programu, musí ale být ve stejné složce. Dále je při volání programu nutné předat testovaný obrázek.

Obrázek 3.4: Princip volání programu

V rámci struktury programu, jsou nejprve definovány konvoluční sítě, které byly použity pro vytrénování jak samotného detektoru, tak rozpoznání druhu zahalení. Následuje načtení souborů do vytvořených konvolučních sítí, dále načtení testovaného obrázku, který je zároveň zobrazen v okně a zároveň je vytvořena pyramida obrázků v

různých měřítkách. Následuje provedení detekce (vložit kód), kdy předaný argument konvoluční síti je načtený testovaný obrázek. Detekce je prováděna dvakrát, dle dvou konvolučních sítí. Vzhledem k tomu, že detekce probíhá na jednom obrázku současně, bylo nutné vyřešit detekci stejné tváře oběma detektory, kdy se zobrazovalo ve výsledku více stejných ohraničujících boxů. To zajišťuje část kódu v druhém detektoru, kdy je mezi každým boxem z prvního detektoru a boxy z druhého detektoru spočítán parametr IoU (Intersection over Union). IoU je podíl průniku těchto dvou boxů a jejich sjednocení. Pokud jsou si dva boxy podobné, bude se hodnota IoU blížit k jedničce. V případě, že IoU překročí hodnotu 0,5, je tento ohraničující box z druhého detektoru zavrhnut a není vykreslen.

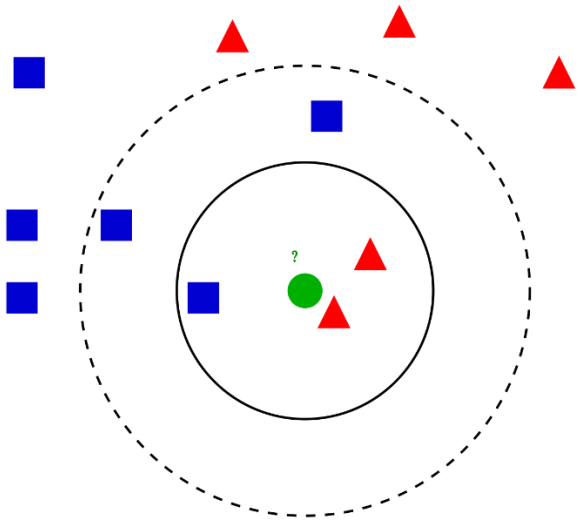
```
for (auto&& d : dets) {
        auto shape = sp(img, d);
        matrix<rgb_pixel> face_chip;
        extract_image_chip(img, get_face_chip_details(shape, 150,
        0.25), face_chip);
        faces.push_back(move(face_chip));
        win.add_overlay(d);
   }



for (auto&& d2 : dets2) {
        double iou2 = 0;
        double iou = 0;
        for (auto&& d : dets) {
              iou = box_intersection_over_union(d, d2);
              if (iou > iou2) {
                    iou2 = iou;
              }
        }
if (iou2 > 0.5) {
              auto shape2 = sp(img, d2);
              matrix<rgb_pixel> face_chip2;
              extract_image_chip(img,get_face_chip_details(
              shape2,150, 0.25), face_chip2);
              faces.push_back(move(face_chip2));
              win.add_overlay(d2);
        }
}
```

Následně je v detektoru vykreslen ohraničující box na celé fotce a v rámci cyklu je připravena nalezená tvář na následné zpracování pomocí detekce míry zahalení. Nejdříve je na detekované tváři proveden výpočet orientačních bodů obličeje, poté je detekovaný obličej vyříznut pomocí funkce extract_image_chip a následně je předán do proměnné faces, což je matice pixelů. Veškerá detekce probíhaly na grafické kartě a jsou poměrně náročné na paměť grafické karty. V případě velkých obrázků (s rozlišením větším jak 2000x2000) se projevil problém, že program skončí s chybovým hlášením o přetečení této paměti. Test byl prováděn na Nvidia GTX1050Ti s 4GB paměti. Dle autora je k bezproblémovému běhu nutná grafická karta s minimálně 5,3 GB paměti, tedy minimálně řada GTX1060 a lepší. Ověření tohoto tvrzení bohužel nebylo možné. Program je schopen

běžet i na procesoru, pak nevzniká žádný problém, ovšem doba běhu programu se výrazně prodlužuje.

```
std::vector<matrix<rgb_pixel>> masky, helmy, bryle, nezahalene, satky;
	matrix<rgb_pixel> img2;
	for (int16_t i = 1; i < 47; ++i) {
		load_image(img2, "train2\\Masky\\obr (" + std::to_string(i) +
		").jpg");
		masky.push_back(move(img2));
	}
	for (int16_t i = 1; i < 202; ++i) {
		load_image(img2, "train2\\Helma\\obr (" + std::to_string(i) +
		").jpg");
		helmy.push_back(move(img2));
	}
	for (int16_t i = 1; i < 50; ++i) {
		load_image(img2, "train2\\Koupani\\obr (" + std::to_string(i) +
		").jpg");
		bryle.push_back(move(img2));
	}
	for (int16_t i = 1; i < 106; ++i) {
		load_image(img2, "train2\\Nezahalene\\obr (" + std::to_string(i)
		+ ").jpg");
		nezahalene.push_back(move(img2));
	}
	for (int16_t i = 1; i < 163; ++i) {
		load_image(img2, "train2\\Satky_saly_rousky\\obr (" +
		std::to_string(i) + ").jpg");
		satky.push_back(move(img2));
	}
```

V rámci samotné detekce míry zahalení je využito metody k-nejbližších sousedů. Tato metoda se v průběhu trénování učí, který druh tváří patří k sobě a které patří do jiné skupiny. Následně je po vytrénování schopná tyto tváře převedené na vektory rozmístit v prostoru takovým způsobem, že podobné tváře jsou velmi blízko sebe a tváře z jiné skupiny jsou velmi vzdáleny. V případě detekce zahalení by tedy testovaná tvář měla být zařazena mezi podobné druhy zahalení. Následně stačí zjistit, kolik se v určité vzdálenosti od právě zařazené tváře, nachází sousedů a jaký jsou druh. Na základě této informace lze určit procentuální zastoupení každé třídy v okolí právě testované tváře. Problémem této metody je určit okruh okolí, v kterém budou sousedi hledáni. To znázorňuje obrázek 3.5. Pokud by oblast prohledávání byla menší, v tomto případě by testovaný objekt byl detekován jako červená skupina. V případě větší (čárkované) oblasti by byl testovaný objekt klasifikován jako modrá skupina. Bylo tedy nutné určit takovou vyhledávací oblast, aby klasifikace byla co nejúspěšnější. [9] [10]



Obrázek 3.5: Princip K-nejbližších sousedů

Jak je uvedeno v kódu na předchozí stránce, nejdříve je nutné načíst trénovací obrázky do matic typu RGB_Pixel tak, aby bylo možné rozlišit jednotlivé trénované skupiny.

```
std::vector<matrix<float, 0, 1>> descriptors_masky = net(masky);
std::vector<matrix<float, 0, 1>> descriptors_helmy = net(helmy);
std::vector<matrix<float, 0, 1>> descriptors_nezahalene =
net(nezahalene);
std::vector<matrix<float, 0, 1>> descriptors_bryle = net(bryle);
std::vector<matrix<float, 0, 1>> descriptors_satky = net(satky);
```

Dále je provedeno rozmístění vytrénovaných dat v prostoru. Každá skupina fotek projde vytrénovanou konvoluční sítí a z každé skupiny jsou vytvořeny face deskriptory, tedy 128D vektory popisující každou skupinu fotek.

```
int16_t pocet, soused_masky, soused_helmy, soused_satky,
soused_nezahalene, soused_bryle = 0;
std::vector<matrix<float,0,1>> face_descriptors = net(faces);
std::vector<image_window> win_clusters(pocetTvari);


for (size_t i = 0; i < face_descriptors.size(); ++i)
    {
        for (size_t j = i; j < descriptors_masky.size(); ++j)
        {
            if (length(face_descriptors[i] - descriptors_masky[j]) <
            0.5)
                soused_masky++;
        }
```

Následně je vytvořeno několik proměnných typu integer, které budou použity později pro inkrementaci. V dalším kroku je provedena konvoluční síť přes vektor face, stejná, která byla použita v části kódu na předchozí straně. Touto konvoluční sítí dojde

k rozmístění každé detekované tváře z právě testovaného obrázku pomocí převedení matice RGB_Pixel na 128D vektor. Vektor faces je naplněn v rámci detekce na straně 30 každou detekovanou tváří. Vektor win_clusters na dalším řádku je typu image_window a zajišťuje vytvoření oken pro vyříznuté tváře, do kterých bude později vložen obrázek s popiskem.

V rámci následujících for cyklů je provedeno srovnání každého 128D vektoru právě testovaného obrázku se 128D vektorem roztříděných trénovacích dat. Kvůli ušetření místa v práci je uvedenou pouze srovnání pro masky, nicméně toto srovnání je provedeno pro všechny skupiny, tedy i pro helmy, šátky (šály, roušky), nezahalené tváře a brýle. Při srovnání jednotlivých vektorů je využito vlastnosti K-nejbližších sousedů, kdy je hledán v definovaném okolí počet sousedů a na základě jejich poměru je možné určit procentuální zastoupení každé třídy v okolí právě testovaného objektu. Hlavním problémem bylo určit „K“, tedy vzdálenost, v jaké budou sousedi hledáni od aktuálního objektu a to tak, aby výsledky byly co nejpřesnější. Dle testů různých hodnot se ukázala nejpřesnější hodnota 0,5. Následně bylo ověřeno v dokumentaci dlib, že ideální hodnota pro hledání sousedů je okolo 0,5-0,6, tudíž výsledky by měly být přesné i dle tohoto ověření. V případě, že je vzdálenost právě testovaných dvou vektorů menší, než tato hodnota, je právě zkoumaný soused vyhodnocen jako blízký právě testovanému a inkrementována proměnná.

```
std::vector<matrix<rgb_pixel>> temp;
temp.push_back(faces[i]);
pocet = soused_bryle + soused_helmy + soused_masky + soused_nezahalene +
soused_satky;

cout << "Celkovy pocet sousedu pro tvar "<< i << " je: " << pocet <<
endl;



soused_bryle = soused_bryle *100 / pocet;
soused_helmy = soused_helmy *100 / pocet;
soused_masky = soused_masky *100 / pocet;
soused_nezahalene = soused_nezahalene *100 / pocet;
soused_satky = soused_satky *100 / pocet;




cout << "Pravdepodobnost, ze jde o nezahalenou tvar je " <<
soused_nezahalene << "%" << endl;
cout << "Pravdepodobnost, ze jde o zahaleni brylemi je " << soused_bryle
<< "%" << endl;
cout << "Pravdepodobnost, ze jde o zahaleni satkem, salou je " <<
soused_satky << "%" << endl;
cout << "Pravdepodobnost, ze jde o zahaleni maskami je " <<
soused_masky<< "%" << endl;
cout << "Pravdepodobnost, ze jde o zahaleni helmou je " << soused_helmy
<< "%" << endl;

win_clusters[i].set_title("Tvar "+ std::to_string(i));
win_clusters[i].set_image(tile_images(temp));
```

Následně po spočítání sousedů ze všech skupin je proveden výpočet celkového počtu sousedů, který je nutný pro následný výpočet procentuálního zastoupení sousedů. Poté je vypočítáno procentuální zastoupení sousedů každé skupiny. Následně je procentuální zastoupení každé třídy vypsáno do příkazového řádku a poté je aktuálně testovaná tvář předána do okna a do popisku okna vypsán identifikátor (např. Tvář 1), kde pod stejným identifikátorem jsou vypsány parametry do příkazového řádku pro případ, že na obrázku je více detekovaných tváří.

4. VÝSLEDKY EXPERIMENTU DETEKCE ZAHALENÝCH TVÁŘÍ

Po procesu trénování následuje otestování samotné detekce. Jako soubor fotek byl použit ten stejný, jako v případě kapitoly 2, tedy vybraných 300 fotek různých druhů zahalení a profesí. Následně bude srovnáno, jaká je schopnost rozeznat zahalenou tvář.

4.1 Srovnání výsledků detektoru

V tabulce 4.1 je znázorněno srovnání nově přetrénované metody Dlib CNN s předchozím testováním v kapitole 2.

Tabulka 4.1: Srovnání metod s nově vytrénovanou

Metody	Přesnost	Preciznost
Dlib CNN	0,438	0,96
Dlib CNN po přetrénování	**0,73**	**0,96**

Z tabulky je zřejmé, že došlo k výraznému pokroku v oblasti přesnosti této metody, preciznost zůstala stejná. Výsledek odpovídá přesnosti 73 %, což je zlepšení oproti původním hodnotám na skoro dvojnásobnou hodnotu. I v rámci přetrénování nebylo dosaženo výsledků TinyFaces, ale dle pozdějšího hledání bylo zjištěno, že tato metoda používá pro trénování všechny fotky ze souboru WiderFace, v rámci testu tedy došlo k tomu, že velká část fotek, které byly testovány, byly zároveň použity pro vytrénování této metody a jedná se o zjevnou výhodu pro tuto metodu. Její výsledek je tedy nutné brát s rezervou.

V rámci rozpoznání zahalených tváří se tedy jedná o poměrně dobrý dosažený výsledek. Na přibližně 10 % zlepšení na této detekci má vliv přidání druhé konvoluční sítě, která rozezná některé obličeje, které předchozí síť nerozpozná a naopak. Vzájemně se tedy doplňují. To způsobuje, že jedna síť pracuje s menším podvzorkováním a každá síť je tedy schopní detekovat jinak velké tváře v obrázku. Součástí hodnocení kvality detekce je i rozdělení vybraných fotek do skupin a zhodnocení schopnosti detekce v různých skupinách. To je znázorněno tabulkách 4.2-4.5 na další straně, kde jsou fotky rozděleny do čtyř skupin – zahalení brýlemi, ať už koupacími nebo slunečními, potom různé šátky, roušky, šály a podobně. V rámci většího zahalení potom karnevalové masky a v poslední skupině různé druhy helem.

Tabulka 4.2: Srovnání metod pro šátky, roušky...

Šátky, burky, šály, roušky, závoje				
	Dlib CNN			
Přesnost		0,476		
Preciznost		0,960		
	Přetrénovaný Dlib CNN			
Přesnost		**0,690**		
Preciznost		**0,960**		

Tabulka 4.3: Srovnání metod-brýle a koupací čepice

Přilby, hokejové helmy, policejní helmy				
	Dlib CNN			
Přesnost		0,338		
Preciznost		0,960		
	Přetrénovaný Dlib CNN			
Přesnost		**0,730**		
Preciznost		**0,960**		

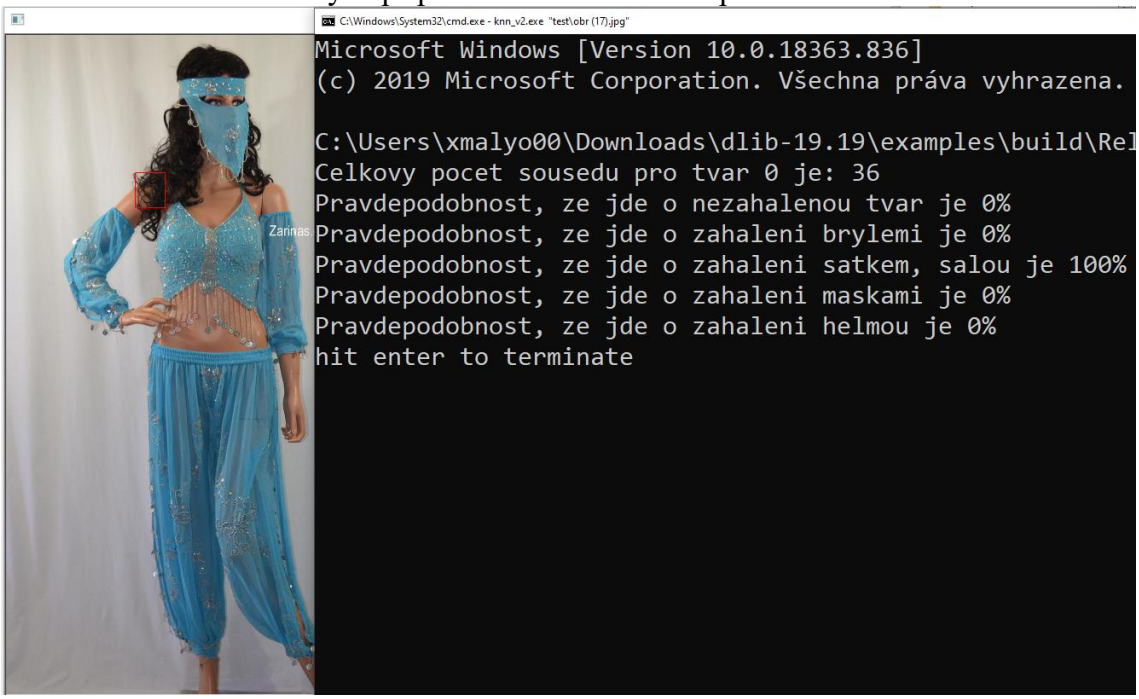
Tabulka 4.4: Srovnání metod pro masky

Plavecké čepice+brýle				
	Dlib CNN			
Přesnost		0,540		
Preciznost		0,960		
	Přetrénovaný Dlib CNN			
Přesnost		**0,690**		
Preciznost		**0,961**		

Tabulka 4.5: Srovnání metod pro helmy, přilby různých profesí

Masky				
	Dlib CNN			
Přesnost		0,500		
Preciznost		0,960		
	Přetrénovaný Dlib CNN			
Přesnost		**0,690**		
Preciznost		**0,961**		

Z tabulek je zřejmé, že nejlépe si vede přetrénovaná metoda při detekci zahalených tváří maskami. Je poměrně zajímavé, že nejlépe je detekována tvář s prakticky plným zahalením, v tomto případě bylo testováno na různých karnevalových maskách. Druhý nejlepší výsledek podává detektor na skupině fotek s různými druhy helem, jako například lyžařské přilby, hokejové brankářské helmy, policejní a vojenské helmy. Naopak hůře si vedou v detekci méně zahalené tváře, které jsou kryté brýlemi nebo třeba potápěčskými brýlemi a koupacími čepicemi a také druhá skupina, kde je tvář krytá šátky, rouškami, šálami. V rámci trénovacích dat bylo vybráno pro detektor stejný počet fotek pro každou trénovací skupinu, takže žádná skupina nebyla zvýhodněna větším počtem tváří pro vytrénování. Co se týče detektoru, správná detekce je provedena tak, že je okolo obličeje vykreslen ohraničující box červené barvy, který je vykreslen okolo tváře. V případě, kdy nebyla tvář detekována, nevykreslí se nic. Jsou zde i v minimální míře případy, kdy dojde k detekci falešně pozitivní tváře. Tato skutečnost je znázorněna na obrázku 4.1. Detektor detekoval tvář tam, kde žádná není a vyhodnotil ji jako tvář zahalenou šátkem. Takových případů nastalo minimum při samotné detekci.



Obrázek 4.1: Falešně pozitivní detekce

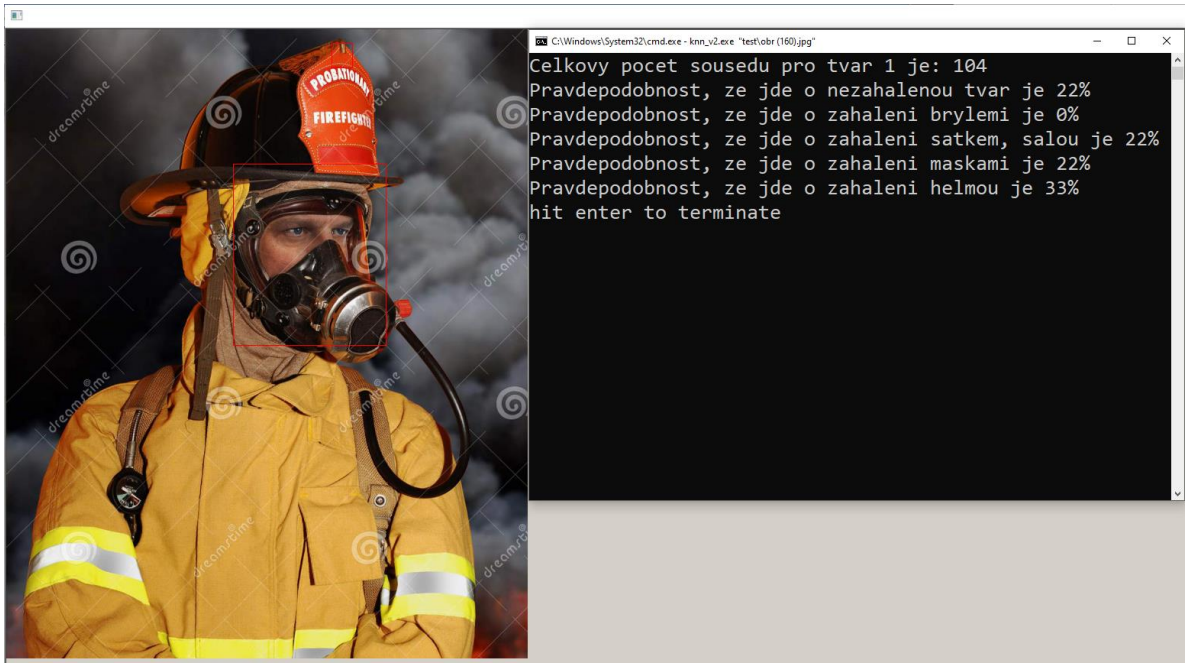
4.2 Zhodnocení schopnosti detekovat míru zahalení

Testování schopnosti detekovat míru zahalení probíhalo na přibližně 200 tvářích tak, aby byly rovnoměrně rozprostřeny do všech testovaných skupin. V tabulce 4.6 je znázorněna matice záměn pro detekci míry zahalení. Matice se používá ke stanovení vlastností a kvality klasifikátoru. Matice záměn v kontextu klasifikačních úloh strojového učení je matice, obsahující ve sloupcích skutečnou hodnotu předpovídaného znaku a v řádcích předpověď klasifikátoru. Jednotlivé buňky matice obsahují četnosti toho, kolikrát došlo na zkoumané datové množině k dané kombinaci skutečné a předpověděné hodnoty. Případy na diagonále matice záměn jsou klasifikovány správně, mimo diagonálu jde o chyby.

Tabulka 4.6: Matice záměn pro detekci míry zahalení

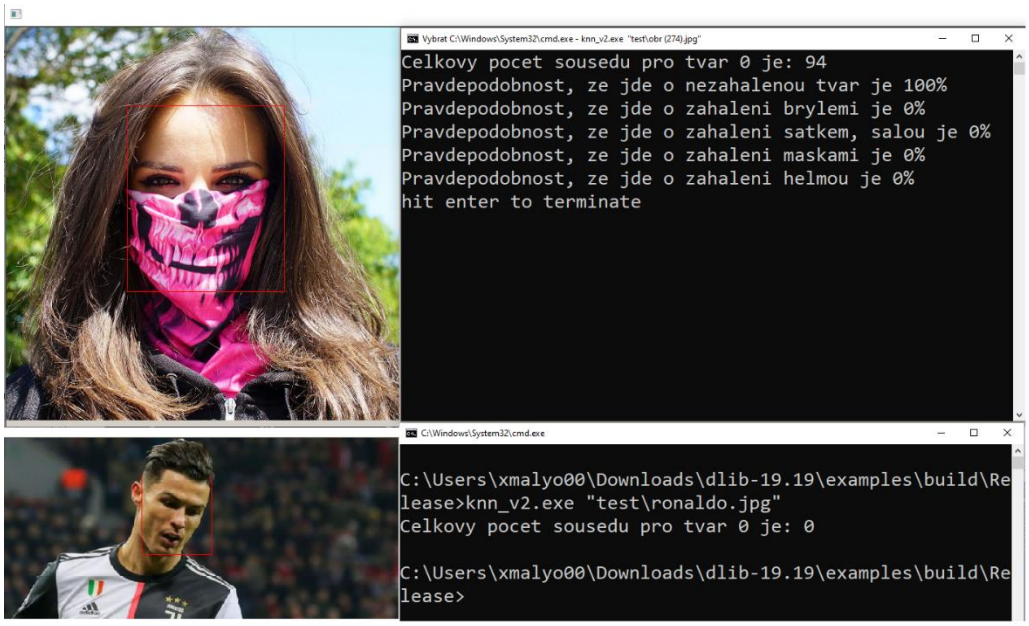
	Realita					
Predikce		Nezahalené	Brýle	Šály, roušky	Masky	Helmy
	Nezahalené	43	0	3	2	0
	Brýle	0	41	0	0	0
	Šály, roušky	0	0	41	0	1
	Masky	0	0	0	37	0
	Helmy	1	2	1	0	54

Z tabulky níže je zřejmé, že z celkových 226 tváří bylo správně klasifikováno 216 tváří a celková správnost detekce míry zahalení je rovna 95 %. Jde o velmi dobrý výsledek, ale tato tabulka nezohledňuje nejistotu klasifikátoru, kdy v několika případech došlo ke správné klasifikaci, ale pravděpodobnost u dané skupiny byla rozdělena mezi více skupin a jistota klasifikátoru nepřesahovala například 30 %. I tak ale počet sousedů této skupiny převyšoval ostatní a jedná se o správnou klasifikaci. Nejistota rozhodnutí o druhu zahalené tváře je znázorněna na obrázku 4.1. V tomto případě došlo ke správné detekci, ale detektor dle vyjádřených procent není příliš vypovídající o jaké zahalení dle detektoru jde.



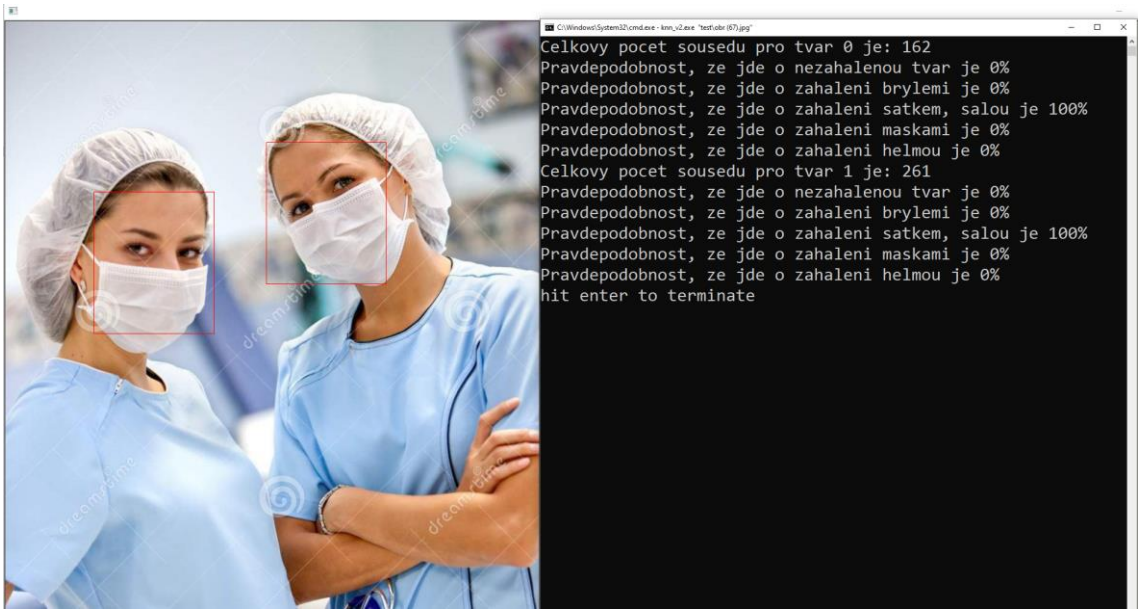
Obrázek 4.2: Nejistota detektoru při detekci

Na dalším obrázku je přibližně znázorněno v jakých případech detektor selhal při určování míry zahalení. V případě první fotky s růžovým šátkem může být matoucí, že na šátku je dokreslena struktura lebky, zubů a celkové obrysy tváře, čímž mohl být zmaten. V případě další fotky na obrázku 4.2 detektor sice provedl správnou detekci tváře, ale při určování druhu zahalení nenalezl žádné sousedy k této detekované tváři. V tomto případě tedy program skončil a tvář nijak neklasifikoval. Takové chování detektor vykazoval pouze v případě dvou fotek, nicméně i takový výsledek je možný a jedná se o možnou chybu samotné detekce míry zahalení tváře.



Obrázek 4.3: Chybné detekce míry zahalení tváře

Na obrázku 4.3 a 4.4 je vidět princip správné detekce. Nalevo je zobrazen vstupní obrázek se zvýrazněnou detekcí tváře v obrázku, napravo pak vyříznutí detekovaných tváří pro jednoznačnou identifikaci výstupu v příkazovém řádku, kam se vypisují jednotlivé pravděpodobnosti skupin. Zde je výsledek správný.



Obrázek 4.4: Detekce dvou tváří kryté rouškami

Obrázek 4.5: Příklad detekce nezahalené tváře

ZÁVĚR

V práci bylo teoreticky shrnuto několik vybraných metod detekce tváře a porovnány jejich možnosti při detekování. Nejprve byly jednotlivé metody teoreticky rozebrány a byla srovnána schopnost jejich detekce na databázi Wider Face, kde byly vyhodnoceny výsledky a srovnány jednotlivé metody. Dále byly metody srovnány na souboru fotek se zahalenými tvářemi, kde byla vyhodnocena celková přesnost a preciznost, následně také sloučeny fotky stejné kategorie a vyhodnoceny výsledky pro tyto jednotlivé kategorie. Následně byly jednotlivé metody okomentovány a vybrána metoda Dlib CNN k dalšímu postupu v diplomové práci. Následně je teoreticky rozebráno samotné zlepšení detekce zahalených tváří touto metodou, kdy byl detektor přetrénován s větším počtem fotek se zahalenými tvářemi, následně vylepšení samotného algoritmu. V rámci zlepšení detekce poté byla přidána další vrstva detektoru s jinou konvoluční sítí, čímž bylo dosaženo zlepšení samotné detekce téměř na dvojnásobek. Následovalo vytrénování a naprogramování samotné míry rozeznání detekce, kde bylo využito metody K-nejbližších sousedů. Poté jsou vyhodnoceny jednotlivé výsledky, a to jak samotného detektoru, tak schopnosti rozeznat míru zahalení. V rámci detekce zahalené tváře byla nejprve srovnána samotná detekce s ostatními metodami a následně srovnány jednotlivé skupiny zahalení a srovnáno v rámci těchto skupin. U rozpoznání míry zahalení byla vypracována matice záměn, kde se ukázalo, že celkově správnost rozpoznání míry zahalení je okolo 95 %.

Cílem této práce bylo experimentálně ověřit schopnost současných detektorů detekovat zahalené tváře. Toto se ukázalo jako reálné, nicméně pro zajištění vyšší přesnosti by bylo potřeba vytvořit rozsáhlejší databázi. V práci byly rovněž uvažovány detektory, které mají dobrý poměr přesnosti ku výpočetnímu času, pokud by byl hlavním kritériem přesnost, dají se použít detektory s vyšší přesností, ale i výrazně vyššími výpočetními nároky. Samotná detekce zahalených tváří může být v současné době psaní práce poměrně aktuální a může být použito ve velkých prostorech či firmách ke kontrole, popřípadě k vedení statistiky, kolik lidí v prostoru nosí roušky a kolik nikoliv.

LITERATURA

[1] VIOLA, Paul, et al. Rapid object detection using a boosted cascade of simple features. CVPR (1), 2001, 1.511-518: 3.

[2] Chi-Feng Wang. *How Does A Face Detection Program Work?* 2018. Dostupné z towardsdatascience: https://towardsdatascience.com/how-does-a-face-detection-program-work-using-neural-networks-17896df8e6ff

[3] ZAFEIRIOU, Stefanos; ZHANG, Cha; ZHANG, Zhengyou. A survey on face detection in the wild: past, present and future. Computer Vision and Image Understanding, 2015, 138: 1-24.

[4] HU, Peiyun; RAMANAN, Deva. Finding tiny faces. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017. p. 951-959.

[5] KING, Davis E. Dlib-ml: A machine learning toolkit. Journal of Machine Learning Research, 2009, 10.Jul: 1755-1758.

[6] AMOS, Brandon, et al. Openface: A general-purpose face recognition library with mobile applications. CMU School of Computer Science, 2016, 6.

[7] MA, Mei; WANG, Jianji. Multi-View Face Detection and Landmark Localization Based on MTCNN. In: 2018 Chinese Automation Congress (CAC). IEEE, 2018. p. 4200-4205

[8] YANG, Shuo, et al. Wider face: A face detection benchmark. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. p. 5525-5533.

[9] GARCIA, Vincent; DEBREUVE, Eric; BARLAUD, Michel. Fast k nearest neighbor search using GPU. In: 2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops. IEEE, 2008. p. 1-6.

[10] DERNIROZ, Giilsen; GIIVENIR, H. Altay. Genetic algorithms to learn feature weights for the nearest neighbor algorithm.

[11] XIANG, Jia; ZHU, Gengming. Joint face detection and facial expression recognition with mtcnn. In: 2017 4th International Conference on Information Science and Control Engineering (ICISCE). IEEE, 2017. p. 424-427.

[12] BRADSKI, Gary; KAEHLER, Adrian. Learning OpenCV: Computer vision with the OpenCV library. " O'Reilly Media, Inc.", 2008.

[13] BALTRUŠAITIS, Tadas; ROBINSON, Peter; MORENCY, Louis-Philippe. Openface: an open source facial behavior analysis toolkit. In: 2016 IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE, 2016. p. 1-10.

[14] PADILLA, R.; COSTA FILHO, C. F. F.; COSTA, M. G. F. Evaluation of haar cascade classifiers designed for face detection. World Academy of Science, Engineering and Technology, 2012, 64: 362-365.