Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Sledovanie osoby vo viacerých videozáznamoch

Diplomová práca

2019 Bc. Andrej Zbín

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Sledovanie osoby vo viacerých videozáznamoch

Diplomová práca

Študijný program:	Informatika
Študijný odbor:	2508 Informatika
Školiace pracovisko:	Katedra informatiky
Školiteľ:	RNDr. Zuzana Černeková, PhD.

Bratislava, 2019
Bc. Andrej Zbín

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Andrej Zbín
Študijný program: informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Sledovanie osoby vo viacerých videozáznamoch
Person tracking in multiple video sequences

Anotácia: Naštudovať problematiku detekcie a sledovania ľudských tvárí. Analyzovať existujúce riešenia publikované v dostupnej odbornej literatúre. Vytvoriť databázu videozáznamov a hľadaných tvárí pre testovacie účely. Navrhnúť a implementovať metódu, ktorá bude schopná vyhľadať osobu vo viacerých videozáznamoch podľa zadaného vizuálneho vzoru t.j. tváre človeka. Vyhodnotiť dosiahnuté výsledky.

Cieľ: Naštudovať problematiku detekcie a sledovania ľudských tvárí. Analyzovať existujúce riešenia publikované v dostupnej odbornej literatúre. Vytvoriť databázu videozáznamov a hľadaných tvárí pre testovacie účely. Navrhnúť a implementovať metódu, ktorá bude schopná vyhľadať osobu vo viacerých videozáznamoch podľa zadaného vizuálneho vzoru t.j. tváre človeka. Vyhodnotiť dosiahnuté výsledky.

Vedúci: RNDr. Zuzana Černeková, PhD.
Katedra: FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry: prof. Ing. Igor Farkaš, Dr.

Dátum zadania: 08.10.2017

Dátum schválenia: 06.12.2017 prof. RNDr. Rastislav Kráľovič, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie: Ďakujem svojej školiteľke RNDr. Zuzane Černekovej, PhD. za cenné rady, nápady a pripomienky pri písaní tejto práce.

Abstrakt

Cieľom práce je navrhnúť a implementovať systém na sledovanie a vyhľadávanie osôb medzi viacerými videozáznamami. V práci predstavíme niekoľko techník na detekciu objektov na záznamoch a porovnáme ich úspešnosť pri detekcii osôb. Ďalej popíšeme techniky rozpoznávania osôb. Prvá technika vykoná extrakciu príznakov zo snímok a ich následnú klasifikáciu pomocou metódy najbližších susedov a metódy podporných vektorov. Dosiahnuté výsledky porovnáme s inými publikovanými riešeniami. Druhá technika využíva siamskú neurónovú sieť na riešenie problému rozpoznávania osôb. Techniky medzi sebou porovnáme a na základe získaných výsledkov vytvoríme systém na sledovanie a vyhľadávanie osôb, ktorého implementáciu popíšeme a úspešnosť vyhodnotíme v poslednej časti práce.

Kľúčové slová: neurónová sieť, počítačové videnie, detekcia osôb, klasifikácia osôb, extrakcia príznakov, python, opencv

Abstract

The aim of this thesis is to design and implement a system for tracking and searching people among multiple videos. We present several techniques for object detection on records and compare their success in detecting a person. We will also introduce person recognition techniques. The first technique uses the feature extraction from images and their subsequent classification using the method of nearest neighbors and support vector machines method. We compare the results with other published solutions. The second technique uses a siamese neural network to solve the problem of person recognition. We will compare the techniques with each other and, based on the results obtained, we will create a system for tracking and searching for people which implementation and accuracy will be described in the last part of the thesis.

Keywords: artificial neural network, computer vision, person detection, person classification, feature extraction, python, opencv

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

Detekcia, rozpoznávanie a sledovanie osôb je v dnešnej dobe zaujímavá téma v oblasti počítačového videnia. Vďaka širokej možnosti využitia v komerčných, bezpečnostných, ale aj iných oblastiach neustále rastú požiadavky na presnosť, spoľahlivosť a rýchlosť systémov na rozpoznávanie osôb vo videozáznamoch. Bezpečnostné zložky vedia technológiu využiť na rozpoznanie potenciálne nebezpečnej osoby v dave ľudí alebo zistenie, či sa daná osoba nachádzala na určitom mieste. V komerčnej oblasti technológie umožňuje automatizáciu niektorých procesov. Príkladom môže byť automatický dochádzkový systém, ktorý sleduje príchody a odchody zamestnancov.

Cieľom našej práce je vytvoriť systém, ktorý dokáže efektívne rozpoznať, sledovať a vyhľadať osoby na videozáznamoch z viacerých kamier. Takýto systém musí riešiť viacero problémov súvisiacich s rozdielmi medzi jednotlivými kamerami, ako je ich umiestnenie a kalibrácia, zmena osvetlenia na záberoch a podobne. Medzi zábermi kamier taktiež môže nastať zmena výzoru hľadanej osoby, či sa už jedná o oblečenie, zmeny na tvári, čiastočné zakrytie tváre okuliarmi alebo iný účes. Všetky spomenuté problémy výrazne komplikujú rozpoznávanie osôb.

V prvej kapitole práce sme popísali možnosti detekcie objektov na záberoch, ktoré sme uplatnili na detekciu postavy človeka a následne medzi sebou porovnali. V ďalšej kapitole sme skúsili využiť riešenie využívajúce extrakciu príznakov z obrázkov a následnú klasifikáciu osôb pomocou týchto príznakov s využitím metód najbližších susedov a podporných vektorov. V tretej kapitole je možné nájsť iné riešenie problému klasifikácie osôb a to pomocou siamskej neurónovej siete. Popis implementácie nášho systému na sledovaniu osôb vo viacerých videozáznamoch a vyhodnotenie jeho úspešnosti sa nachádza v poslednej kapitole práce.

Kapitola 1

Detekcia osôb

Detekciu definujeme ako identifikáciu regiónu na vstupnej snímke, v ktorom sa nachádza hľadaný objekt. Predtým, ako osobu na zázname môžeme rozpoznávať a sledovať, je potrebné ju detegovať. Jedná sa o dôležitý problém, ktorý je možné riešiť viacerými spôsobmi.

1.1 Haarové príznaky

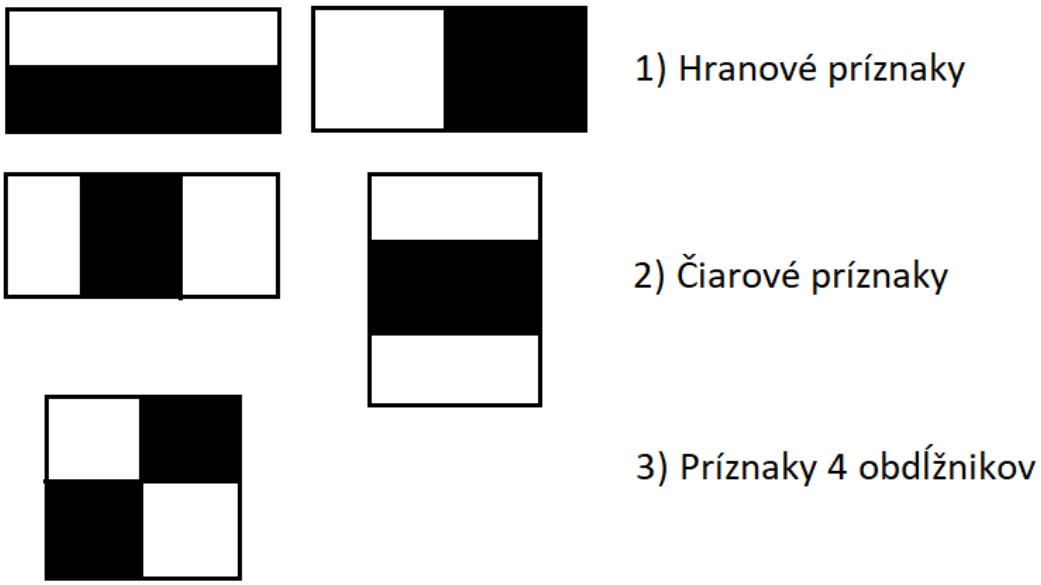
Isté časti objektov, ktoré chceme detegovať, sú tmavšie, ako iné časti. Túto skutočnosť zachytávajú haarové príznaky.

Haarové príznaky si vieme predstaviť ako obdĺžniky, ktoré obsahujú dva typy regiónov. V oboch regiónoch sa sčítajú hodnoty pixelov a výsledný príznak je rozdiel týchto súčtov. Pri detekcii pomocou haarových príznakov sa zoberú všetky možné veľkosti a pozície takýchto obdĺžnikov.

Pre urýchlenie výpočtu súm sa využije metóda integrálneho obrazu. V integrálnom obraze I predstavuje hodnota pixelu súčet všetkých pixelov nachádzajúcich sa vľavo a nad od tohto pixelu v pôvodnom obraze O.

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

Ak chceme vypočítať súčet pixelov obdĺžnikového regiónu. stačia nám tri operácie sčítania a odčítania. Od spodného pravého pixelu odpočítame hodnoty ľavého dolného a pravého horného pixelu od ľavého dolného pixelu a následne ešte pričítame hodnotu ľavého horného pixelu.



Obr. 1.1: Ukážka troch typov haarových príznakov, ktoré sa používajú na detekciu tváre algoritmom Viola-Jones [20]. Súčet hodnôt pixelov v bielych regiónov je odčítaný od súčtu hodnôt pixelov v čiernych regiónov.

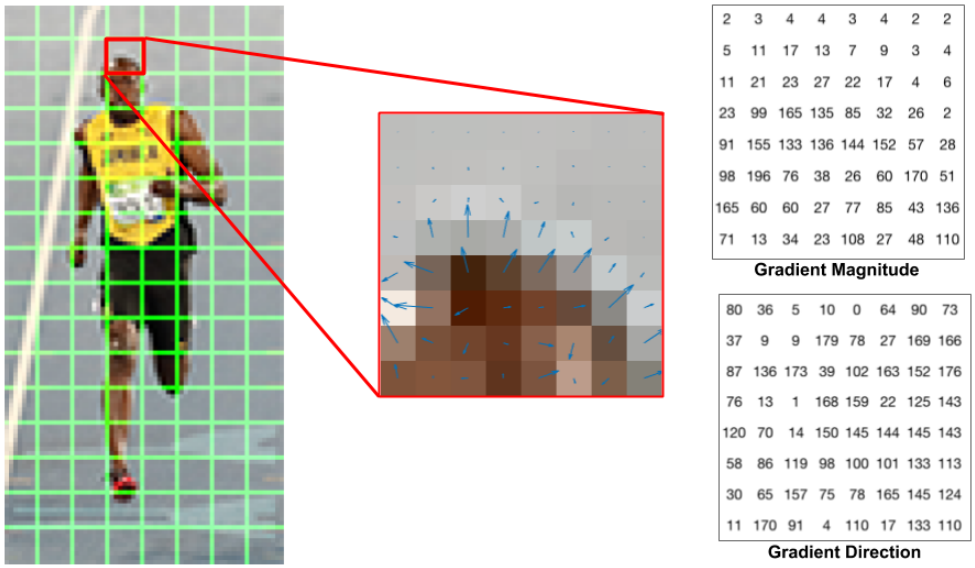
Na vstupnom obraze veľkú časť zaberajú regióny, v ktorých sa hľadaný objekt nenachádza a preto by bolo testovanie všetkých príznakov zbytočné. Je preto zavedený kaskádový systém, kde sa v prvej fáze otestuje len niekoľko príznakov a pokiaľ sa určí, že na tomto regióne sa hľadaný objekt nenachádza, prejdeme na ďalší región a hľadáme ďalej. V opačnom prípade kaskádovo pokračujeme testovaním presnejších príznakov [20].

1.2 Histogram orientovaných gradientov

Hlavnou myšlienkou je, že zachytenie distribúcie orientácie a intenzity gradientov v snímke dokáže popísať tvar hľadaného objektu.

Vstupná snímka je rozdelená na regióny rovnakej veľkosti. Následne sa v každom regióne vypočítajú orientácie gradientov a tie sa zaznamenajú v histograme. Intenzita určuje váhu orientácie v gradiente, čiže gradienty s malou intenzitou vytvorené napríklad šumom nie sú pri výpočte podstatné.

Jednotlivé regióny sú navyše spájané do skupín, ktoré sa môžu prelínať, čo znižuje vplyv zmien osvetlenia na snímke. Prelínanie redukuje rozdiely medzi jednotlivými skupinami a v praxi sa ukazuje, že takéto riešenie dosahuje lepšie výsledky [11]. Spojením výsledných histogramov jednotlivých skupín vznikne výstupný vektor príznakov - histogram orientovaných gradientov [3] [13].



Obr. 1.2: Ukažka výpočtu histogramu orientovaných gradientov. Pre každý región sú vypočítané gradienty reprezentované orientáciou a intenzitou [11].

1.3 Konvolučné neurónové siete

Neurónové siete vyžadujú na ich natrénovanie veľký počet označených dát.

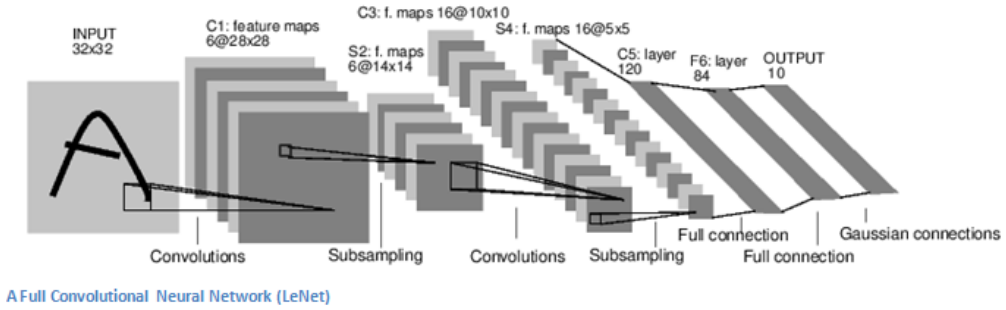
Ak chceme natrénovať neurónovú sieť, ktorá hľadá na snímkach postavy ľudí, potrebujeme snímky s označenými obdĺžnikmi ohraničujúcimi časti snímok, na ktorých sa postavy ľudí nachádzajú. Tomuto postupu hovoríme učenie s učiteľom. Trénovanie prebieha pokusom a omylom. V každej iterácii sieť vyskúša spraviť predikciu na trénovacích príkladoch a vypočíta rozdiel medzi predikciou a očakávaným výstupom. Cieľom je v nasledujúcej iterácii túto chybu znížiť. Na meranie úspešnosti predikcii sa používa stratová funkcia [1].

Neurónová sieť sa skladá zo vstupnej a výstupnej vrstvy, medzi ktorými je niekoľko skrytých vrstiev. V konvolučných neurónových sieťach sa vyskytujú vrstvy, ktorých úlohou je detegovať hrany, tvary, farby a podobne. Tieto vrstvy nazývame konvolučné vrstvy.

V konvolučných vrstvách je dôležitý pojem filter. Filter je matica váh, ktorou sa prenásobujú podregióny vstupu do vrstvy, ktorých veľkosť je rovná veľkosti filtra. Podregióny vznikajú metódou pohybujúceho sa okna, začneme v ľavom hornom roku a postupne sa posúvame o jeden prvok. Tento proces sa nazýva konvolúcia [4]. Pre každú konvolučnú vrstvu máme definovaný počet a veľkosť filtrov.

S využitím väčšieho množstva konvolučných vrstiev zostrojíme sieť, ktorá dokáže rozpoznávať aj komplikovanejšie vzory. Medzi konvolučnými vrstvami zvyčajne máme ďalšie vrstvy, ako sú ReLU vrstvy a vrstvy redukujúce dimenziu.

Na konci siete sa nachádza plne-prepojená vrstva, ktorá spája výstupy filtrov z predchádzajúcej konvolučnej vrstvy, prípadne výstupov ďalších vrstiev, ktoré sa môžu nachádzať po tejto konvolučnej vrstve.



Obr. 1.3: Ukážka architektúry jednoduchej konvolučnej neurónovej siete [4].

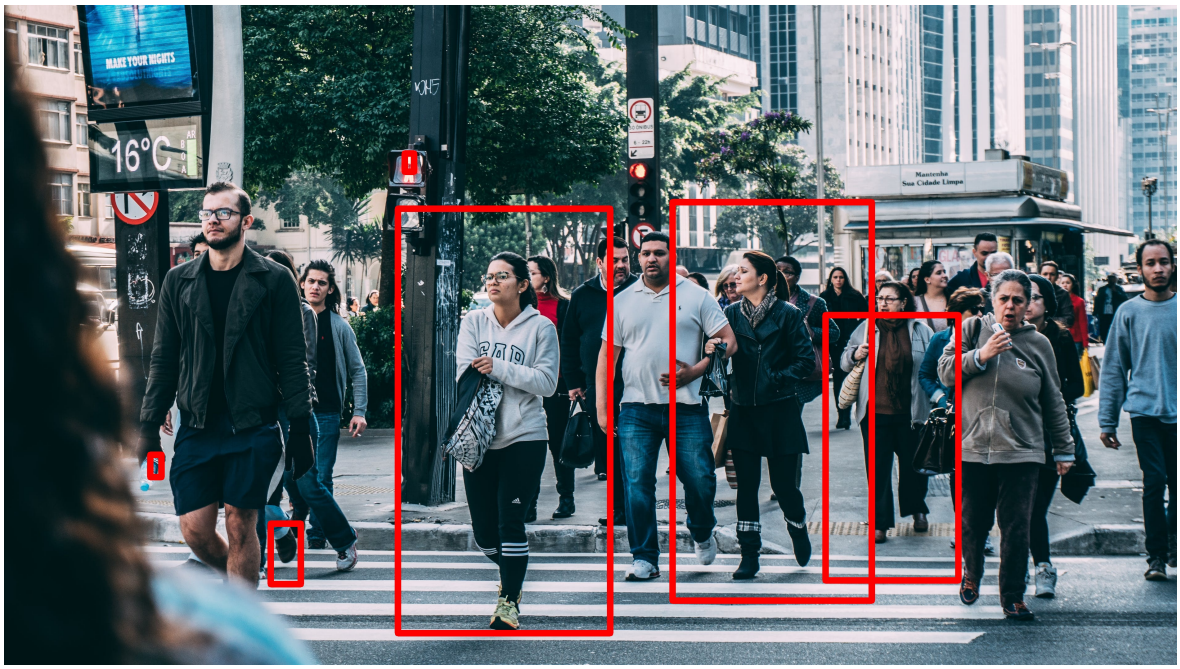
1.4 Porovnanie

Tri popísané typy detekcie sme otestovali pri detekcii postáv na obrázku s rozmerom 3780x2127 pixelov. Pri haarových príznakoch a histograme orientovaných gradientov sme využili knižnicu OpenCV a jej zabudované funkcie. Na detekciu neurónovou sieťou sme využili predtrénovanú neurónovú sieť MobileNet-SSD.

Z nižšie uvedenej tabuľky a obrázkov jednoznačne vidieť lepšie výsledky pri detekcii pomocou neurónovej siete.

Tabuľka 1.1: Namerané časy potrebné na detekciu.

Metóda	Čas potrebný na detekciu
Haarové príznaky	168ms
Histogram orientovaných gradientov	8 188ms
Neurónová sieť	4 901ms



Obr. 1.4: Na obrázku vidieť osoby detegované pomocou haarových príznakov.

Obr. 1.5: Na obrázku vidieť osoby detegované pomocou histogramu orientovaných gradientov.

Obr. 1.6: Na obrázku vidieť osoby detegované pomocou neurónovej siete.

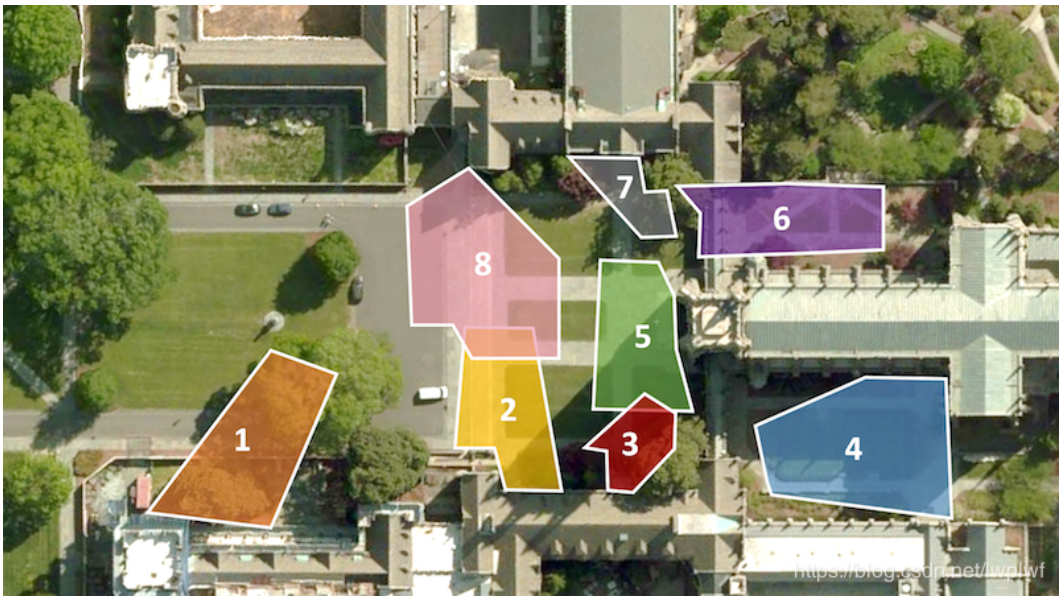
Kapitola 2

Extrakcia príznakov a ich klasifikácia

V kapitole ukážeme riešenie problému klasifikácie osôb na základne extrahovaných príznakov zo snímok celej postavy týchto osôb. Naše riešenie preberá niektoré myšlienky z práce Person Re-identification by Local Maximal OccurrenceRepresentation and Metric Learning [9], detaily riešenia sa však odlišujú. Ďalej porovnáme dve metódy klasifikácie: pomocou najbližších susedov a metódou podporných vektorov.

2.1 Dataset

Využili sme dataset DukeMTMC-reID [15] [23]. Jedná sa o upravenú verziu datasetu DukeMTMC. Pôvodný dataset obsahuje videozáznamy z ôsmich rôznych kamier. V upravenej verzii sú zábery jednotlivých osôb vyrezané z každej stodvadsiatej snímky pôvodného videozáznamu. Takto je získaných 1 404 identít, ktoré sa vyskytujú aspoň na dvoch kamerách a 408 identít vyskytujúcich sa práve na jednej kamere. Náhodne vybraná polovica osôb, ktoré sa vyskytujú na viacerých záberoch, je zvolená ako trénovacia množina, zvyšné zábery patria do testovacej množiny.



Obr. 2.1: Oblasti, ktoré zachytávajú jednotlivé kamery v datasete DukeMTMC

Obr. 2.2: Zábery jednej osoby z rôznych kamier v datasete DukeMTMC-reID

2.2 Extrakcia príznakov

Extrakcia príznakov je metóda redukcie dimenzie elimináciu nepodstatných detailov. Cieľom je získať množinu príznakov, ktorá pôvodný vstup dobre reprezentuje s menším počtom dát.

2.2.1 Lokálny binárny vzor

Príznaky získané pomocou lokálneho binárneho vzoru sa veľmi často využívajú pri rôznych problémoch klasifikácie. Jedná sa o dobre overenú metódu, ktorá má navyše

nízku výpočtovú náročnosť [12]. Lokálny binárny vzor porovnáva hodnoty každého pixelu s jeho susednými pixelmi. Má dva parametre.

Počet susedov P Určuje, s koľkými susednými pixelmi obrázku budeme porovnávať každý pixel.

Polomer R Polomer kružnice, na ktorej ležia susedia, pričom určíme, že pixel má tvar štvorca s dĺžkou strany 1.

Pokiaľ chceme susedov, s ktorými ma pixel spoločnú hranu, nastavíme P=4 a R=1. Ak by sme chceli susedov so spoločným vrcholom pomyselného štvorca, nastavíme P=8 a R=1.

Lokálny binárny vzor spočíta pre každý pixel I s hodnotou V_I a so susedmi $I_0, ..., I_{P-1}$ čiernobieleho obrázku hodnotu $LBP(I)$,

$$LBP(I) = \sum_{n=0}^{P-1} s(V_I - V_{I_n}) * 2^n$$

$$s(k) = \begin{cases} 0 & \text{ak } k < 0 \\ 1 & \text{ak } k \geq 0 \end{cases}$$

Môžeme so všimnúť, že takto získané číslo bude mať hodnotu z intervalu prirodzených čísel $< 0, 2^P)$.

2.2.2 Lokálny ternárny vzor

Funguje podobne ako lokálny binárny vzor [19]. Ak už názov napovedá, pri porovnávaní namiesto dvoch hodnôt vyhodnocuje na tri hodnoty. Má navyše ešte jeden parameter.

Threshold T Podľa thresholdu (hranice) sa určí výsledná hodnota pri porovnávaní

$$LTP(I) = \sum_{n=0}^{P-1} s(V_I, V_{I_n}) * 3^n$$

$$s(c, p) = \begin{cases} 2 & \text{ak } c + T < p \\ 1 & \text{ak } c + T > p \\ 0 & \text{inak} \end{cases}$$

Môžeme so všimnúť, výsledok bude z intervalu prirodzených čísel $< 0, 3^P)$. Väčšie rozpätie intervalu môžu v praxi znamenať lepšie výsledky [19].

Škálovo invariantný lokálny ternárny vzor je rozšírením lokálneho ternárneho vzoru. Jeho výhodou je, že sa dokáže vysporiadať s jemnými zmenami osvetlenia na snímkach. Tieto zmeny môžu byť na celej snímke alebo aj len na jej častiach [10].

$$SILTP(I) = \sum_{n=0}^{P-1} s(V_I, V_{I_n}) * 3^n$$

$$s(c,p) = \begin{cases} 2 & \text{ak } (1+T) * c < p \\ 1 & \text{ak } (1-T) * c > p \\ 0 & \text{inak} \end{cases}$$

2.2.3 Využitie viacrozmerného HSV histogramu pri extrakcii príznakov

Pri hľadaní osoby, ktorú sme stratili z dohľadu, môže byť dobrým príznakom, podľa ktorého sa nám podarí tú istú osobu opätovne nájsť, farba. Veľkú plochu človeka pokrýva oblečenie, ktoré ma vo veľa prípadoch rovnakú farbu z každej strany. To znamená, ak budeme vyhľadávať osobu podľa farby, nezáleží nám, z akého uhlu osobu zachytíme.

Rozpoznávať len podľa farby však rozhodne nie je dobré riešenie. Človek môže mimo záberu kamery oblečenie zmeniť, hlavne ak sú zábery zachytené s väčším časovým rozdielom. Takisto problém môžu spôsobiť dve rôzne osoby s rovnakou farbou oblečenia.

Snímky najskôr prekonvertujeme do farebného modelu HSV. Model HSV sa v oblasti počítačového videnia využíva z dôvodu, že v praxi dosahuje lepšie výsledky ako model RGB [6]. Na zachytenie farby na snímkach využijeme viacrozmerný histogram. Histogram má rozmer 8x8x8. Nech pixel obrázku má HSV hodnoty (h, s, v), ktoré patria do intervalu $< 0, 255 >$ Potom výskyt týchto hodnôt daného pixelu bude v histograme zaznamenaný na indexe $\lfloor \frac{8h}{256} \rfloor$ prvej dimenzie, $\lfloor \frac{8s}{256} \rfloor$ druhej dimenzie a $\lfloor \frac{8v}{256} \rfloor$ tretej dimenzie.

Obrázok, z ktorého extrahuje príznaky, rozdelíme na regióny veľkosti 10x10 pixelov, pričom sa tieto regióny prekrývajú na piatich pixeloch. V každom regióne vytvoríme viacrozmerný HSV histogram. Pre redukciu dimenzie výstupného vektora príznakov spojíme histogramy regiónov jedného riadku tak, že výsledný histogram bude mať jednotlivé výskyty hodnôt maximalizované (výskyt hodnoty výsledného histogramu daného riadku regiónov je rovný maximálnemu výskytu tej istej hodnoty v danom riadku). Dôležité však je, že ak sa osoba nachádzala na rôznych snímkach na iných častiach týchto snímok (napríklad na jednej snímke mohla byť viac vpravo ako na druhej snímke), výsledný histogram bude pre obe snímky rovnaký. Presnejšie povedané, výskyty hodnôt, ktoré sú získane z časti obrázku, na ktorom sa osoba nachádza, budú v oboch histogramoch približne rovnaké.

2.2.4 Využitie škálovo invariantného lokálneho ternárneho vzor pri extrakcii príznakov

Rovnako, ako pri využití viacrozmerného HSV histogramu, tentokrát čiernobiely obrázok rozdelíme na regióny. Na každom regióne aplikujeme škálovo invariantný lokálny ternárny vzor najskôr s parametrami $P = 4$, $R = 3$, $T = 0,3$ a následne aj $P = 4$,

$R = 5$, $T = 0,3$. Tieto hodnoty sa ukázali pri testovaní ako najlepšie a sú rovnaké, aké boli použité už v spomenutej práci alebo aj iných podobných prácach [9]. Následne zostrojíme (jednorozmerný) histogram zo získaných hodnôt jednotlivých pixelov, ktorý po riadkoch maximalizujeme rovnako, ako v predchádzajúcej podkapitole. Keďže hodnoty sú z intervalu $< 0,80 >$, v histograme zaznamenáme výskyt každej hodnoty samostatne. Hodnota i bude zaznamenaná na indexe i.

2.2.5 Algoritmus retinex

Aj keď škálovo invariantný lokálny ternárny vzor a farebný model HSV čiastočne riešia zmeny osvetlenia medzi zábermi rovnakej, ale aj rôznych kamier, rozhodli sme sa na predspracovanie snímok použiť algoritmus retinex. Jeho úlohou je tieto zmeny odstrániť. Algoritmus je inšpirovaný biologickými mechanizmami očí a ich schopnosti adaptovať sa týmto zmenám [18].

2.2.6 Analýza hlavných komponentov

Ak majú snímky, podľa ktorých chceme klasifikovať, rozmery 65x155 pixelov, extrakciou príznakov ako sme popísali dostaneme vektor príznakov veľkosti až 20 220 $((8^3 + 3^4 + 3^4) * (\frac{155-5}{5}))$. Dimenziu tohto vektora môžeme redukovať pomocou analýzy hlavných komponentov (PCA).

Snahou PCA je znížiť dimenziu množiny vektorov s čo najmenšou stratou informácie. PCA vykoná transfomáciu prvkov vektora, ktoré sú pravdepodobne korelované, na nižší počet lineárne nezávyslých prvkov, ktoré sa nazývajú hlavné komponenty [2]. Vďaka tomu budeme vedieť vykonávať klasifikáciu oveľa rýchlejšie.

2.3 Klasifikácia

Úlohou klasifikácie je identifikovať, do akej množiny kategórií (tried) patrí nové pozorovanie na základne informácií získaných z trénovacej množiny pozorovaní.

2.3.1 Metóda k-najbližších susedov

Metóda k-najbližších susedov je relatívne jednoduchá, no aj tak v mnohých prípadoch účinná klasifikačná metóda [7]. Je založená na výpočte vzdialeností medzi vektormi príznakov trénovacích dát a vektorom príznakov pozorovania, ktoré chceme klasifikovať. Po vypočítaní všetkých vzdialeností pridelíme danému pozorovaniu triedu, ktorá sa vyskytuje najčastejšie v k najbližších (s najmenšou vypočítanou vzdialenosťou) pozorovaniach z trénovacej množiny dát.

Na výpočet vzdialeností sa väčšinou používa euklidovská vzdialenosť. Hodnota k sa volí relatívne malá a líši sa v závislosti od počtu rôznych tried, počtu pozorovaní v trénovacej množine a podobne. Vhodnú hodnotu je možné nájsť experimentovaním [7].

Hlavnou nevýhodou je jeho rýchlosť. Pri veľa dátach v trénovacej množine je metóda pomalá, keďže je pri každej klasifikácii nutné počítať vzdialenosť pre každé pozorovanie v tejto množine.

2.3.2 Metóda podporných vektorov

Pomocou metódy podporných vektorov (SVM) vieme klasifikovať do dvoch tried. Príznaky, podľa ktorých chceme klasifikovať, sú reprezentované ako vrcholy v priestore. Úlohou SVM je nájsť nadrovinu, ktorá tento priestor rozdelí na dva podpriestory, pričom v každom podpriestore sa nachádzajú len príznaky patriace do jednej triedy. Vzdialenosť všetkých príznakov by mala byť od rozdeľovacej nadroviny čo najväčšia.

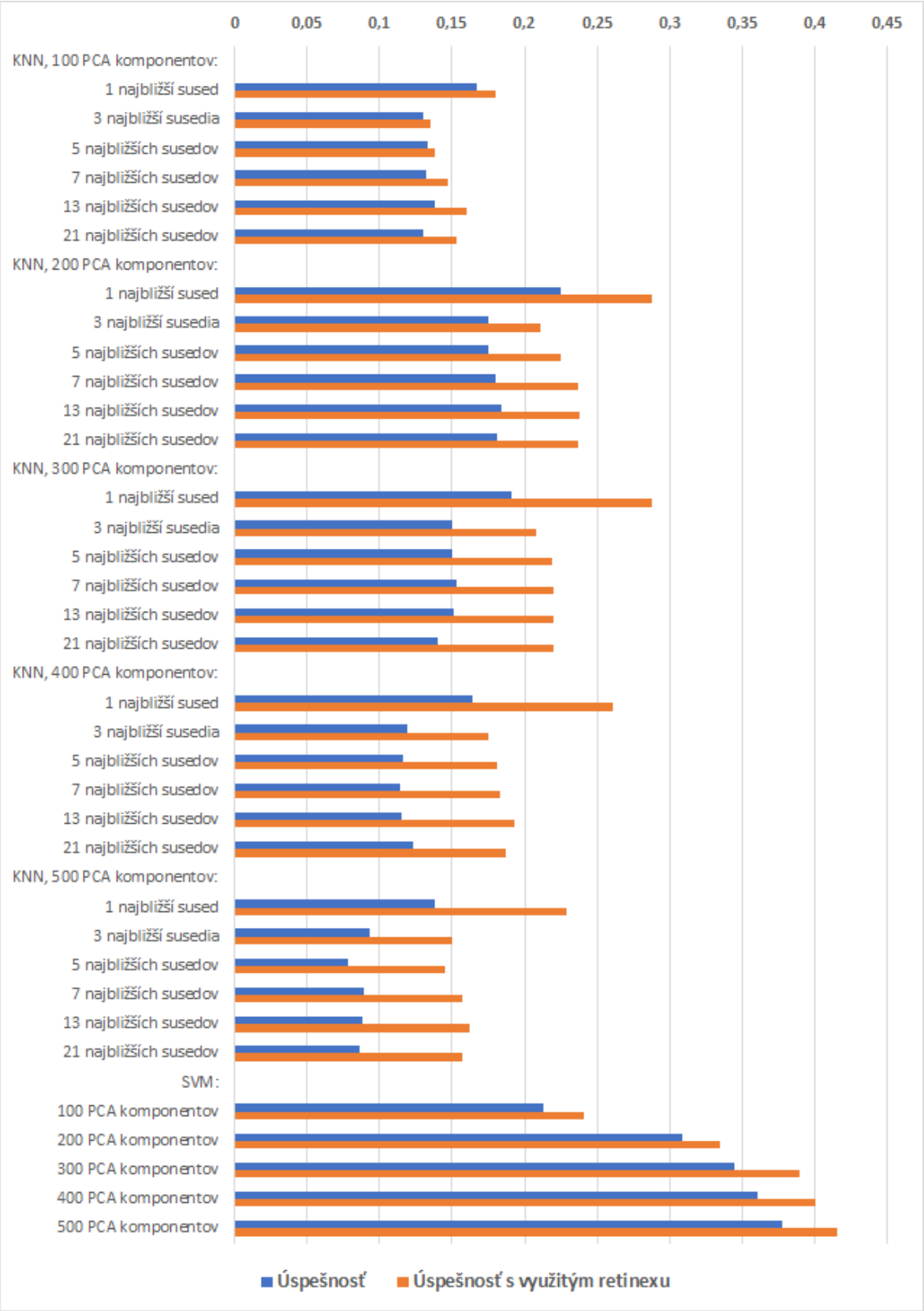
V prípade, že takáto nadrovina neexistuje, je možné uplatniť transformáciu, ktorá priestor príznakov prevedie do vyššej dimenzie, v ktorej už existovať bude. V tomto prípade hovoríme o nelineárnom SVM.

Klasifikácia je oproti metóde najbližších susedov rýchlejšia. Pokiaľ chceme klasifikovať nové pozorovanie, stačí zistiť, do ktorého podpriestoru patrí nový vektor príznakov.

Klasifikácia do viacerých tried je možná zostrojením SVM pre každú triedu a následnom binárnom rozhodnutí, či príklad patrí do tejto triedy alebo nie [14]. Preto je pri väčšom počte tried SVM pomalšie.

2.3.3 Výsledky

Z výsledkov testovania vidieť, že využitie retinexu pri klasifikácii pomáha vo všetkých prípadoch. Metóda najbližších susedov dosahuje najlepšie výsledky pri 200 PCA komponentoch a hľadaní len jedného najbližšieho suseda. Metóda podporných vektorov dosahuje výrazne lepšie výsledky ako metóda najbližších susedov. Najlepšie výsledky dosahuje pri 500 PCA komponentoch.

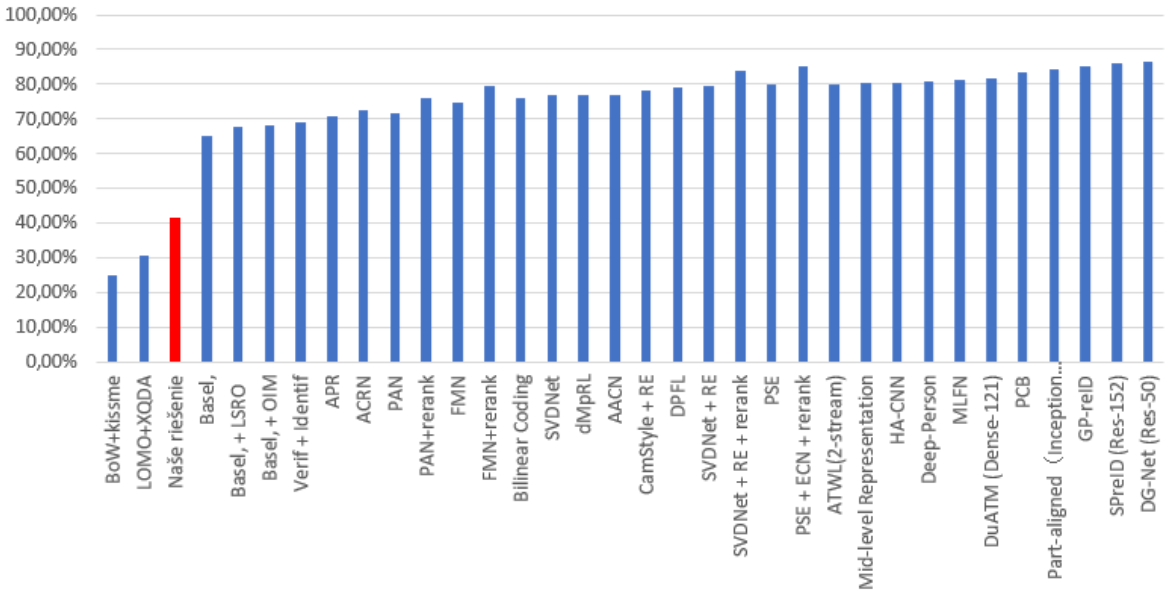


Obr. 2.3: Graf porovnáva výsledky popísaných riešení s rôznymi parametrami a s využitím a taktiež bez využitia algoritmu retinex.

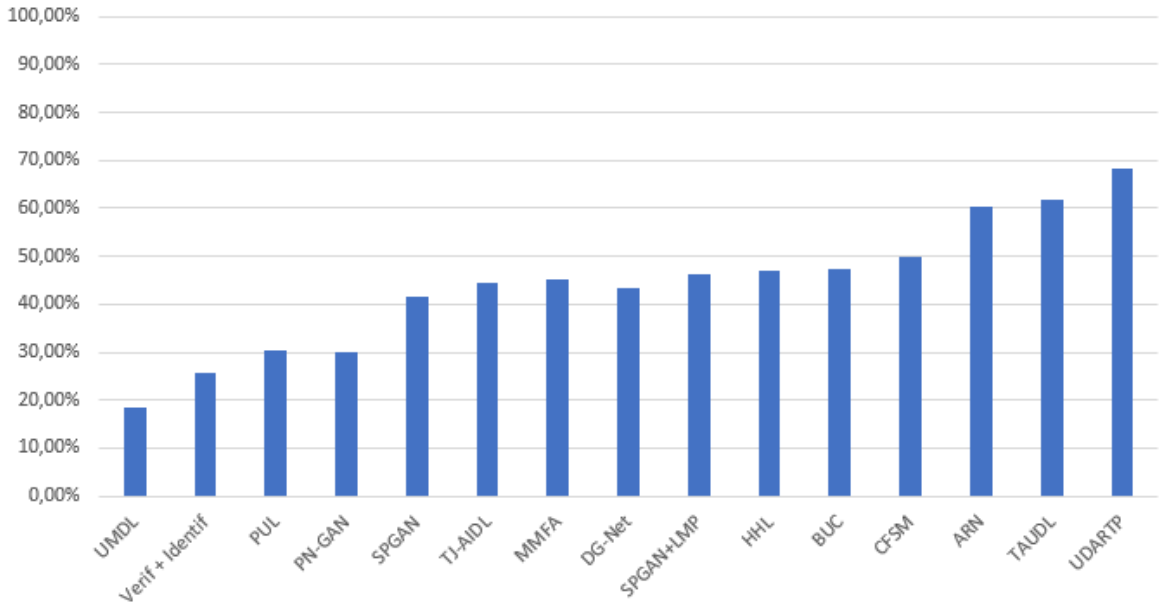
2.4 Porovnanie a diskusia

V porovnaní s publikovanými riešeniami pre tento dataset dosahujeme slabšie výsledky. Avšak ani najlepšie riešenia nedosahujú dostatočnú úspešnosť klasifikácie, pokiaľ sa riešenie aplikuje na odlišný dataset [22]. Pokiaľ chceme vytvoriť systém, ktorý by sa dal využiť na viacerých miestach, bude potrebné iné riešenie.

Ďalšou zjavnou nevýhodou je klasifikácia do dopredu určeného počtu tried, čo prináša problémy bližšie popísané v nasledujúcej kapitole 3.2. Ak by sme chceli riešenie využiť na sledovanie osôb v reálnom čase, toto riešenie je navyše pomalé. Extrakcia príznakov a ich následná klasifikácia zaberie na jednej snímke približne sekundu.



Obr. 2.4: Graf porovnáva úspešnosť klasifikácie nášho najlepšieho riešenie (zobrazeného červenou farbou) s ostatnými zverejnenými riešeniami [22].



Obr. 2.5: Graf ukazuje úspešnosť klasifikácie riešení natrénovaných na datasete Market-1501 a testovaných na DukeMTMC-reID [22].

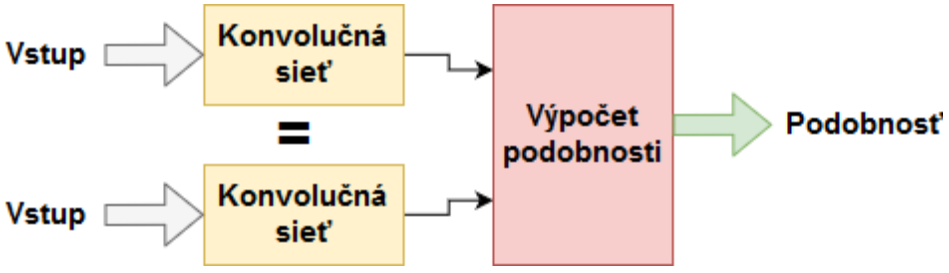
Kapitola 3

Klasifikácia pomocou siamskej neurónovej siete

V tejto kapitole popíšeme možnosti využitia siamskej neurónovej siete na problém rozpoznávania osôb. Ukážeme dve riešenia tohto problému: rozpoznávanie podľa záberov tvárí a záberov celej postavy.

3.1 Siamská neurónová sieť

Siamská neurónová sieť je typ neurónovej siete, ktorá je zložená z dvoch rovnakých podsietí. Tieto podsiete majú okrem architektúry aj rovnaké váhy v jednotlivých vrstvách. Úlohou siamskej architektúry nie je priamo klasifikovať vstup. Ich hlavné využitie je nájdenie podobnosti medzi dvomi vstupmi. Hlavná myšlienka je založená na tom, že ak obe podsiete majú podobné vstupy, ich výstupné vektory budú blízko seba. Naopak, ak sú vstupy málo podobné, výstupné vektory budú od seba ďalej. Inak povedané, vstupy sú si podobné práve vtedy, keď je vzdialenosť medzi výstupnými vektormi malá.



Obr. 3.1: Jednoduchá architektúra siamskej neurónovej siete s konvolučnými podsieťami.

Vzdialenosť d medzi dvoma vektormi $p = (p_1, p_2, ..., p_n)$ a $q = (q_1, q_2, ..., q_n)$ môžeme merať viacerými spôsobmi. Uvádzame dve najpopulárnejšie metriky.

Manhattanovská vzdialenosť Vzdialenosť dvoch vektorov je súčet absolútnych hodnôt rozdielov ich kartézkych súradníc.

$$d(p,q) = \sum_{i=1}^{n} \|p_i - q_i\|$$

Euklidovská vzdialenosť Vzdialenosť dvoch vektorov je dĺžka úsečky spájajúcej dva body reprezentované danými vektormi v n-rozmernom priestore.

$$d(p,q) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

Obe uvedené metriky majú niekoľko dobrých vlastností.

Symetria Dva vstupy sú si rovnako podobné bez ohľadu na to, do ktorej podsiete vstupujú.

$$d(p,q) = d(q,p)$$

Trojuholníková nerovnosť Vysoká podobnosť vstupu s ďalšími dvomi vstupmi zaručí, že aj tieto vstupy si budú dostatočne podobné.

$$d(p,q) \leq d(p,z) + d(z,q)$$

Nezápornosť a nulová vzdialenosť pre rovnaké vektory Navzájom rovnaké vstupy dosahujú maximálnu podobnosť.

$$d(p,q) \geq 0$$

$$d(p,q) = 0 \Rightarrow p = q$$

3.2 Klasifikácia

Pri bežnej neurónovej sieti určenej priamo na klasifikáciu získame pre každý vstup triedu, do ktorej patrí. V prípade klasifikácie osôb budú triedami napríklad ich mená. Takýto prístup má niekoľko nevýhod.

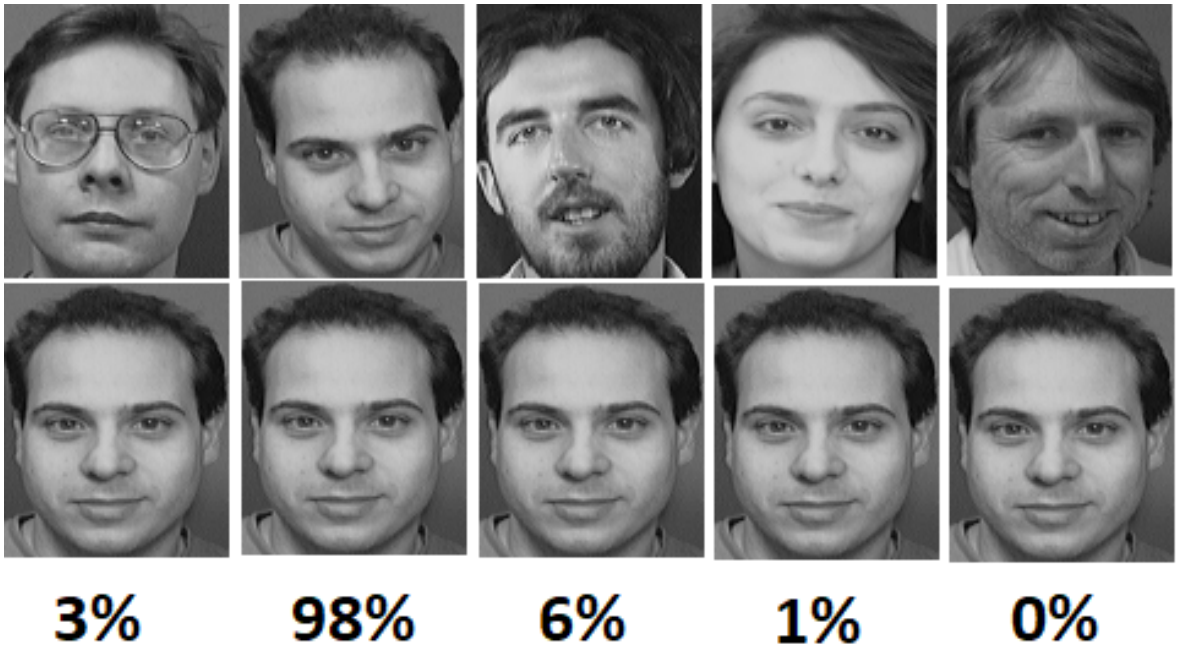
Hlavným problémom je klasifikácia nových alebo neznámych osôb. Môžeme vytvoriť špeciálnu triedu, do ktorej budú patriť všetky neznáme osoby. Ak však chceme klasifikovať viac ako jednu novú osobu a vyžadujeme, aby každá osoba bola zaradená do inej triedy, takáto trieda nám nepomôže. Bude potrebné znova natrénovať sieť aj s novými osobami. Takáto situácia môže nastať napríklad v podniku, ktorý sleduje prítomnosť svojich zamestnancov. Pokiaľ firma zamestná nové osoby, bez opätovného trénovania

siete ich nebude možné správne klasifikovať. Existujú však situácie, kedy dopredu nevieme, aké a koľko osôb vlastne klasifikovať budeme a opätovné trénovanie siete teda nie je možné.

Ďalšou nevýhodou je, že aj pokiaľ sieť bude znova natrénovaná, na jej trénovanie je potrebný veľký počet záberov nových osôb, ktoré chceme klasifikovať. Tieto zábery musia zachytávať nové osoby z rôznych uhlov a v rôznych polohách, čo v niektorých situáciách nemusí byť realizovateľné.

Klasifikácia pomocou siamskej neurónovej siete rieši všetky spomenuté problémy. Po natrénovaní siete nám pre každú triedu stačí uchovávať len jeden vstup, ktorý do tejto triedy patrí a danú triedu reprezentuje. Ak chceme vytvoriť novú triedu, opätovné trénovanie nie je potrebné, stačí len vytvoriť vstup, ktorý do tejto triedy patrí.

Klasifikácia prebieha jednoducho: ak je vstup, ktorý chceme klasifikovať, dostatočne podobný vstupu, ktorý reprezentuje niektorú triedu, bude aj tento vstup patriť do danej triedy.



Obr. 3.2: Vrchný riadok reprezentuje jednotlivé triedy, do ktorých chceme klasifikovať, stredný riadok obsahuje záber, ktorý chceme klasifikovať a posledný riadok ukazuje percentuálnu zhodu v danom stĺpci.

3.3 Architektúra siete

Oboje siete, na hľadanie podobnosti tvárí a postáv, používajú tú istú architektúru. Architektúra siete bola inšpirovaná sieťou určenou na klasifikáciu znakov v rôznych abecedách [8]. Vstupom do siete pre hľadanie podobnosti tvárí sú čiernobiele obrázky

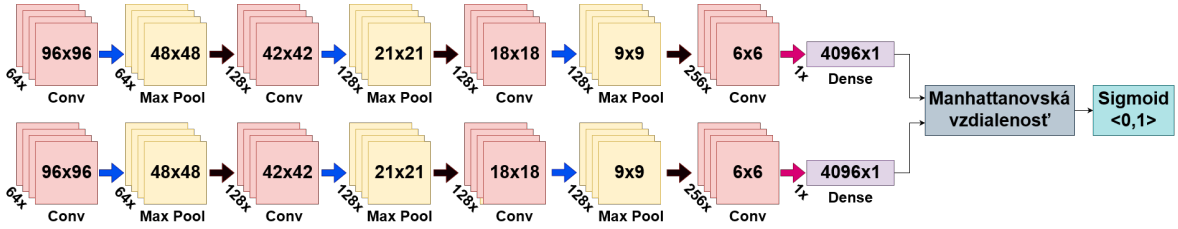
veľkosti 105x105 a do siete pre hľadanie podobnosti postáv sú farebné obrázky o veľkosti 65x155 s troma kanálmi farieb RGB.

Sieť obsahuje štyri konvolučné vrstvy, ktoré majú postupne 64, 128, 128 a 256 filtrov s veľkosťami kernelu 10x10, 7x7, 4x4 a 4x4. Medzi nimi sa nachádzajú max pooling vrstvy, ktoré redukujú každú dimenziu na polovicu.

Max pooling funguje nasledovne: vstupnú maticu rozdelí na regióny rovnakej veľkosti (v našom prípade 2x2). Nová matica bude obsahovať len maximálne hodnoty v každom regióne. Po každej max pooling vrstve nasleduje dropout s pravdepodobnosťou 15% aby sme predišli preučeniu [17].

Za každou konvolučnou vrstvou je vrstva ReLU. Výstupom jej aktivačnej funkcie je jej vstupný vektor, v ktorom sú všetky záporné hodnoty nahradené 0.

Po týchto spomenutých vrstvách nasleduje plne-prepojená vrstva, jej výstupom je vektor o veľkosti 4096. Takéto vektory vstupujú do ďalšej vrstvy dva, jeden z každej podsiete siamskej siete. Vypočítame manhattanovskú vzdialenosť medzi nimi a výslednú vzdialenosť pretransformujeme pomocou sigmoid funkcie $S(x) = \frac{1}{1+e^{-}x}$ na hodnoty z intervalu <0, 1>, kde jednotka značí maximálnu a nula minimálnu podobnosť vstupov do siete.



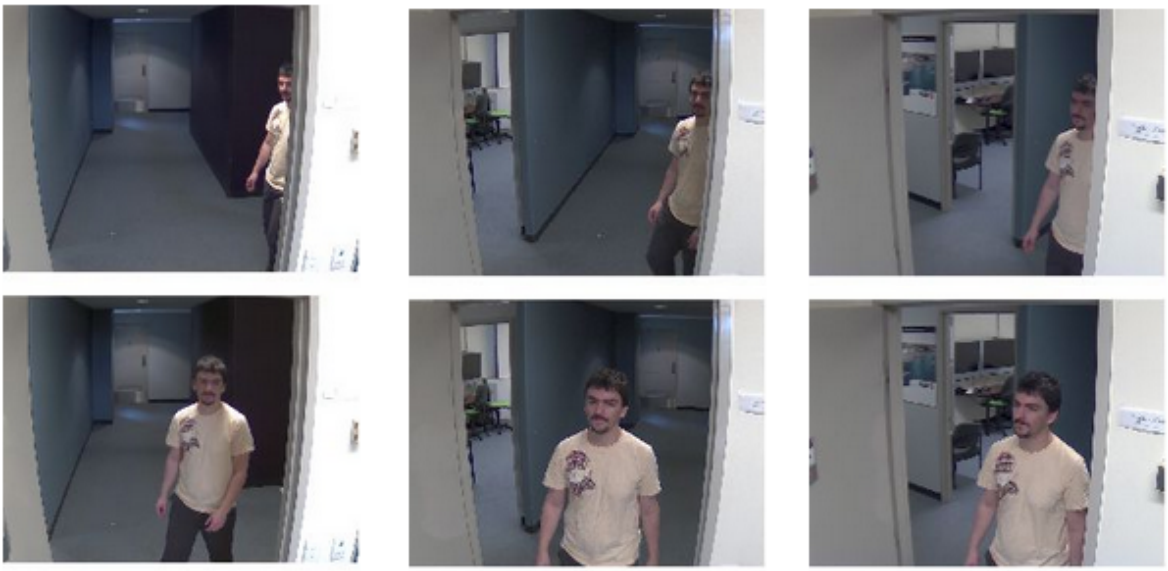
Obr. 3.3: Ukážka jednotlivých vrstiev sieti pre hľadanie podobností tvárí

3.4 Trénovanie siete

3.4.1 Datasety

Na trénovanie siete pre hľadanie podobnosti podľa snímok postáv sme opäť využili dataset DukeMTMC-reID [15] [23] spomenutý v kapitole 2.

Na hľadanie podobnosti tvárí sme zvolili iný dataset, pretože zábery v datasete DukeMTMC-reID nemajú dostatočné rozlíšenie na natrénovanie siete. Vybrali sme dataset ChokePoint [21]. Tento dataset obsahuje zábery nasnímané v rozpätí jedného mesiaca pomocou trojíc kamier umiestnených na viacerých miestach. Celkovo zachytáva 29 rôznych identít na celkovo 64 204 snímkach ich tvárí. Zábery sme rozdelili na trénovaciu a testovaciu množinu. Tri osoby sa vyskytujú len v testovacej množine rovnako ako všetky zábery z niektorých zvolených kamier. Trénovacia množina celkovo obsahuje 11 323 snímok, testovacia 52 247 snímok.



Obr. 3.4: Zábery jednej osoby z jednej trojice kamier v datasete ChokePoint

3.4.2 Stratová funkcia

Ako stratovú funkciu sme zvolili binárnu krížovú entropiu medzi predikciami a očakávanými hodnotami. Formálne, chybová funkcia L je definovaná:

$$L(p,q) = t(p,q) * log(s(p,q)) + (1 - t(p,q)) * log(1 - s(p,q))$$

$$t(p,q) = \begin{cases} 1 & \text{ak p a q patria do rovnakej triedy} \\ 0 & \text{inak} \end{cases}$$

$$s(p,q) = predikcia\ podobnosti$$

Vyskúšali sme použiť aj kontrastnú stratu L‘.

$$L‘(p,q) = \frac{1}{2} * t(p,q) * max(0, m - D(p,q))^2 + \frac{1}{2} * (1 - t(p,q)) * D(p,q)^2$$

$$t(p,q) = \begin{cases} 1 & \text{ak p a q patria do rovnakej triedy} \\ 0 & \text{inak} \end{cases}$$

$$D(p,q) = predikcia\ vzdialenosti$$

$$m = hyperparameter$$

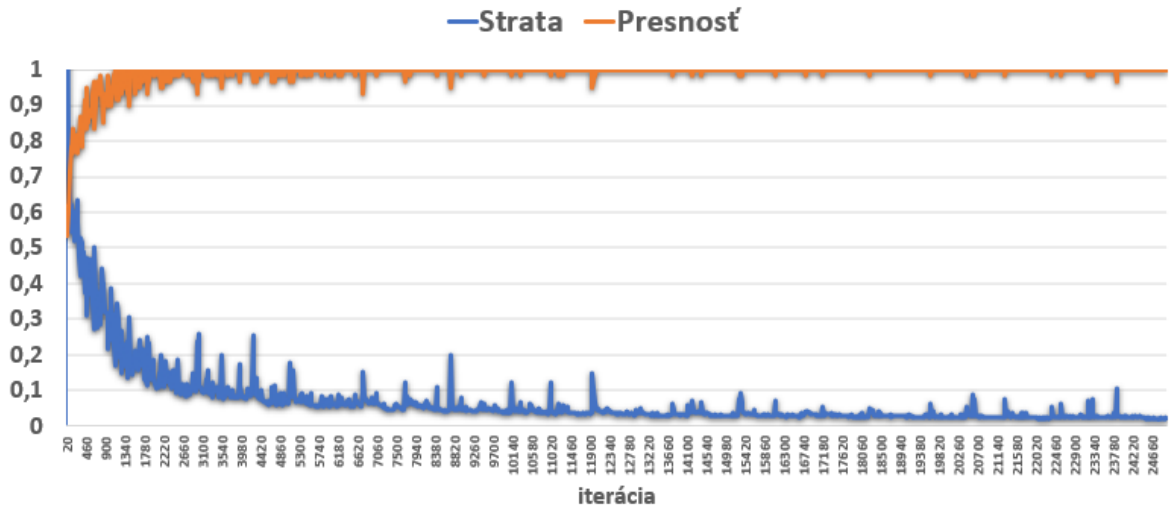
Dosiahnuté výsledky pri učení však boli v porovnaní s krížovou entropiou výrazne horšie po rovnakej časovej dobe učenia. Výhodou krížovej entropie je v tomto prípade aj to, že nemusíme vhodne voliť žiaden hyperparameter.

Na minimalizáciu chyby sme použili optimalizátor Adam s rýchlosťou učenia len 0,001. Vyššie hodnoty spôsobovali príliš veľké kolísanie chyby medzi jednotlivými iteráciami, čo paradoxne spôsobilo pomalšie učenie.

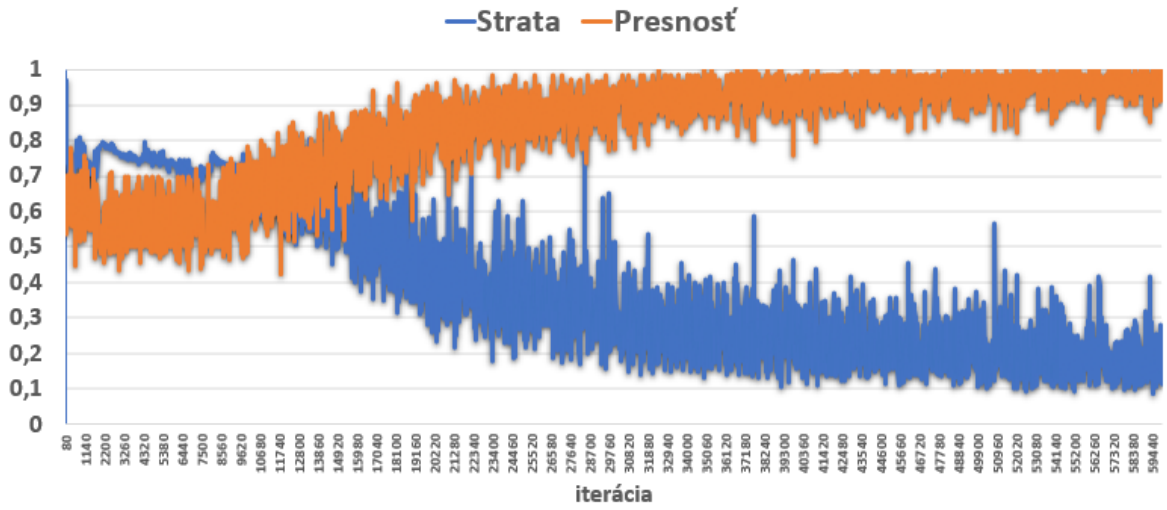
3.4.3 Popis trénovania

Trénovanie prebiehalo v 25 000 iteráciách v prípade hľadania podobnosti tvárí a v 60 000 iteráciách v prípade hľadania podobnosti postáv. V každej iterácii sme zvolili osem osôb. Pre každú osobu bol zvolený jeden náhodný záber. Následne sme z každej kamery sme vybrali jeden záber, ktorý zachytáva tú istú osobu a jeden záber, ktorý zachytáva inú osobu. Pokiaľ sa rovnaká osoba na tejto kamere nenachádzala, nevybrali sme ani záber inej osoby, aby zostal pomer rovnakých a rôznych osôb rovnaký.

Prvý zvolený záber bol porovnávaný s každým vybraným záberom tej istej osoby, pričom očakávaná hodnota podobnosti mala byť 1. Rovnako bol porovnávaný s každým vybraným záberom iných osôb s očakávanou hodnotou podobnosti 0.



Obr. 3.5: Priebeh učenia hľadania podobnosti tvárí. Z grafu vidieť, že na natrénovanie siete stačí aj menší počet iterácií na dosiahnutie dobrej presnosti.



Obr. 3.6: Priebeh učenia hľadania podobnosti postáv. Na natrénovanie je potrebný väčší počet iterácií, ale sieť dosahuje dobrú presnosť na trénovacích príkladoch.

3.5 Výsledky

3.5.1 Meranie úspešnosti

Kvalitu siete ukážeme pomocou chybovej tabuľky, ktorá má dva stĺpce a dva riadky, zobrazujúcej správne zhody, nesprávne zhody, správne nezhody a nesprávne nezhody na náhodne vybraných dvojiciach obrázkov [5].

Správna zhoda (TP) Snímky na vstupe patria rovnakej osobe a sieť rozhodla správne (podobnosť bola vysoká).

Nesprávna zhoda (FP) Sieť rozhodla, že snímky na vstupe patria rovnakej osobe, avšak v skutočnosti zachytávajú rôzne osoby

Správna nezhoda (TN) Snímky na vstupe nepatria rovnakej osobe a sieť rozhodla správne (podobnosť bola nízka).

Nesprávna nezhoda (FN) Sieť rozhodla, že snímky na vstupe zachytávajú rôzne osoby, no v skutočnosti patri tej istej osobe.

Z týchto hodnôt odvodíme ďalšie hodnoty, ktoré dobre ukazujú kvalitu siete.

Presnosť (ACC) Pravdepodobnosť správneho rozhodnutia siete na ľubovoľných príkladoch.

$$ACC = \frac{TP + TN}{TP + FP + TN + FN}$$

Senzitivita (TPR) Pravdepodobnosť správneho rozhodnutia na príkladoch, na ktorých vstupné snímky patrili rovnakej osobe.

$$TPR = \frac{TP}{TP + FN}$$

Špecificita (TNR) Pravdepodobnosť správneho rozhodnutia na príkladoch, na ktorých vstupné snímky nepatrili rovnakej osobe.

$$TNR = \frac{TN}{TN + FP}$$

Precíznosť (PPV) Pomer správnych rozhodnutí na príkladoch, na ktorých vstupné snímky patrili rovnakej osobe, k počtu rozhodnutí, v ktorých sieť určila, že vstupné snímky patrili rovnakej osobe.

$$PPV = \frac{TP}{TP + FP}$$

Negatívna predikčná hodnota (NPV) Pomer správnych rozhodnutí na príkladoch, na ktorých vstupné snímky nepatrili rovnakej osobe, k počtu rozhodnutí, v ktorých sieť určila, že vstupné snímky nepatrili rovnakej osobe

$$NPV = \frac{TN}{TN + FN}$$

Následne vyskúšame priamo klasifikovať osobu. Každá trieda bude reprezentovať inú osobu len jedným záberom. Vyberie jednu z týchto osôb a nejaký iný záber tejto osoby a tento záber porovnáme so zábermi každej triedy. Osobu klasifikujeme do triedy s najväčšou podobnosťou.

3.5.2 Výsledky pre hľadanie podobnosti tvárí

Na testovacej databáze záberov tvárí sme dosiahli dobré výsledky vzhľadom na precíznosť a špecificitu. Ak sieť určí, že vstupné snímky patria rovnakej osobe, s pravdepodobnosťou viac ako 92% je to pravda. Rovnako, ak snímky nezachytávajú rovnakú osobu, sieť s pravdepodobnosťou až 95% spraví správne rozhodnutie. Hlavným problémom sa ukazujú falošné nezhody. Ak je na záberoch rovnaká osoba, sieť spraví správne rozhodnutie len v 59% prípadov.

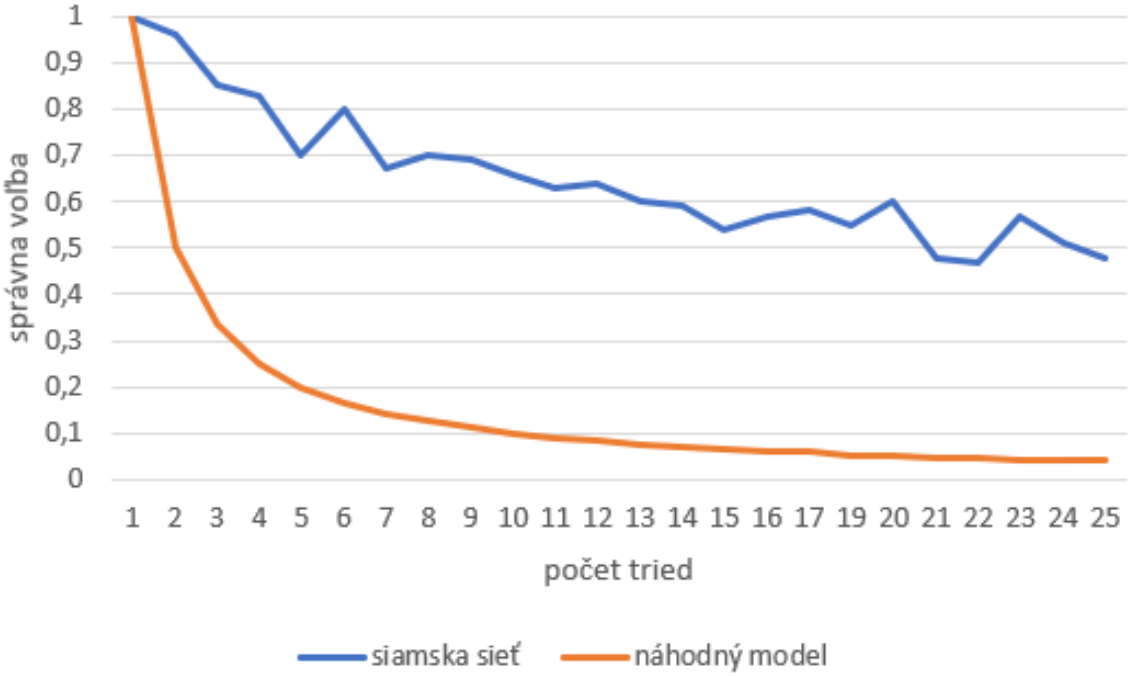
Pri testovaní sme vyhodnotili zhodu, ak výstupná hodnota siete bola väčšia ako 0,5. Ak by sme chceli znížiť počet falošných nezhôd a nevadia nám falošné zhody, je túto hodnotu možné znížiť.

Tabuľka 3.1: Úspešnosť hľadania podobnosti tvárí bez akéhokoľvek zmenšovania.

		Predikcia	
		Zhoda	Nezhodna
Realita	Zhoda	5084	3601
	Nezhoda	420	8265

Presnosť	0,769
Senzitivita	0,585
Špecificita	0,952
Precíznosť	0,924
Negatívna predikčná hodnota	0,697

Výsledky klasifikácie do tried sme porovnali s modelom, ktorý vstup klasifikuje do náhodnej triedy.



Obr. 3.7: Graf porovnáva úspešnosť klasifikácie pomocou siamskej siete s náhodným modelom pre rôzne počty klasifikačných tried.

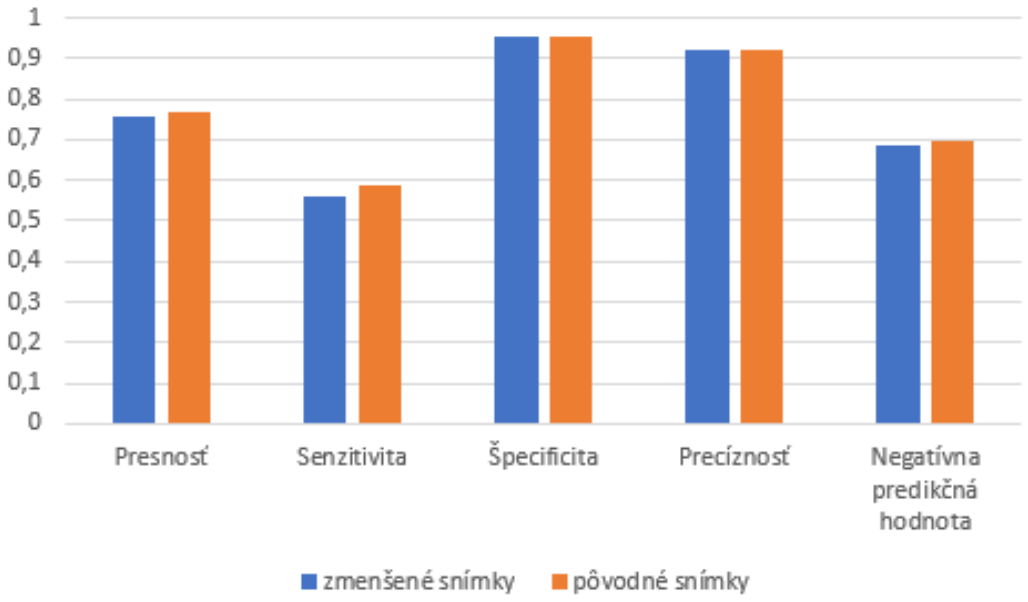
Následne sme vyskúšali hľadanie podobnosti aj na záberoch tvárí s menším rozlíšením. Zábery v testovacom datasete sme zmenšili na tretinu v oboch rozmeroch a sledovali sme, či sa zmení presnosť. Keďže trénovanie prebiehalo len na záberoch v rovnakom rozlíšení, sieť sa mohla preučiť na danom rozlíšení a presnosť sa mohla výrazne znížiť. Zmena v meraných hodnotách však bola zanedbateľná. Ak by však zmena

nastala, jednoduchým riešením by bolo sieť znova natrénovať na rovnakých záberoch, ktoré by sme náhodne zmenšovali.

Tabuľka 3.2: Úspešnosť hľadania podobnosti tvárí na zmenšených snímkach.

		Predikcia	
		Zhoda	Nezhodna
Realita	Zhoda	4892	3823
	Nezhoda	422	8293

Presnosť	0,756
Senzitivita	0,561
Špecificita	0,952
Precíznosť	0,921
Negatívna predikčná hodnota	0,684



Obr. 3.8: Graf porovnáva hľadanie podobnosti tvárí na snímkach pôvodnej a zmenšenej veľkosti.

3.5.3 Výsledky pre hľadania podobnosti postáv

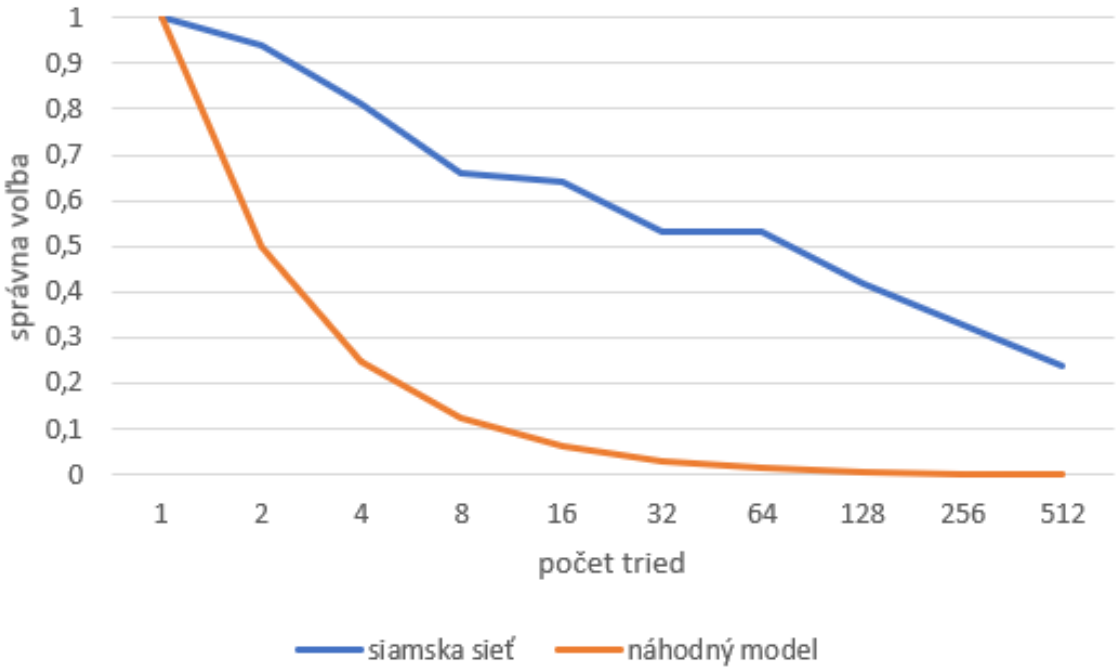
Pri hľadania podobnosti postáv sme dosiahli podobné výsledky ako pri hľadania podobnosti tvárí. Dosahujeme však menší počet falošných nezhôd.

Tabuľka 3.3: Úspešnosť hľadania podobnosti postáv.

		Predikcia	
		Zhoda	Nezhodna
Realita	Zhoda	1661	540
	Nezhoda	158	2043

Presnosť	0,841
Senzitivita	0,755
Špecificita	0,928
Precíznosť	0,913
Negatívna predikčná hodnota	0,791

Výsledky klasifikácie do tried sme taktiež porovnali s modelom, ktorý vstup klasifikuje do náhodnej triedy.



Obr. 3.9: Graf porovnáva úspešnosť klasifikácie pomocou siamskej siete s náhodným modelom pre rôzne počty klasifikačných tried.

3.5.4 Diskusia

Z dosiahnutých výsledkov môžeme povedať, že oboje riešenia sa dajú využiť v systéme na sledovanie osoby vo viacerých záznamoch. Úspešnosť klasifikácie je síce nižšia ako pri riešení z prechádzajúcej kapitoly, avšak je univerzálnejšie. rýchlejšie (približne 50x) a nie je potrebné poznať počet tried, do ktorých chceme klasifikovať.

Problémom, ktorý treba vyriešiť, však zostávajú falošné nezhody, ktoré sú pri prechode osoby medzi zábermi dvoch kamier a jej následnej reidentifikácii dôležité. Ich elimináciou by sme docielili správnu reidentifikáciu osoby na druhom zázname. Riešením by mohla byť kombinácia riešení navrhnutých v tejto kapitole.

Kapitola 4

Implementácia aplikácie

Finálna aplikácia je implementovaná v jazyku Python 3 s využitím knižnice OpenCV. Dokáže súčasne prehrať záznamy uložené na disku vo forme videa alebo sledu obrázkov, záznam z webkamery a taktiež internetový stream. Osoby sú na týchto záznamoch detegované pomocou predtrénovanej neurónovej siete MobileNet-SSD, ich tváre pomocou klasifikátora založeného na Haarových príznakoch implementovaného v knižnici OpenCV. Na klasifikáciu osôb aplikácia využíva siamskú neurónovú sieť. Sieť bola predtrénovaná na datasete ChokePoint [21] pre klasifikáciu na základe tváre a na datasete DukeMTMC [15] [23] pre klasifikáciu na základe celej postavy. Aplikácia umožňuje dotrénovanie sietí na prehrávaných záznamov, čim sa zvýši presnosť klasifikácie. Na vyhľadanie konkrétnej (známej) osoby v zázname stačí len jedna snímka postavy alebo tváre danej osoby. Okrem toho program dokáže identifikovať neznáme osoby, ktoré sa už predtým nachádzali v niektorom z prehrávaných záznamov.

4.1 Detekcia osoby a tváre

Detekciu osôb vykonávame každých 30 snímok. Detekcia osoby je vykonávaná funkciou detect_people v súbore detect.py. Parametrom funkcie je obrázok, na ktorom chceme osoby nájsť. Detekcia môže byť vykonaná pomocou histogramu orientovaných gradientov alebo pomocou neurónovej sieteMobileNet-SSD. Druhá možnosť sa v testovaní ukázala ako lepšia. Okrem lepšej presnosti detekcie bola taktiež rýchlejšia ako prvá spomenutá možnosť. Výstupom funkcie je množina obdĺžnikov, kde každý obdĺžnik ohraničuje jednu osobu na snímke. Obdĺžnik je reprezentovaný 4 bodmi - vrcholmi obdĺžnika. Pred vstupom do siete je obrázok normalizovaný na veľkosť 300x300. Vďaka tomu je detekcia niekoľkonásobne rýchlejšia a pritom stále dostatočne presná.

Detekcia tváre prebieha v aplikácii len na častiach snímok, na ktorých bola detegovaná osoba. Vďaka tomu je detekcia rýchla a vieme jednoducho určiť, ktorá tvár patrí ku ktorej osobe. Detekciu program vykonáva pomocou Haarových príznakov. Táto

metóda sa ukázala ako dostatočne presná na náš účel aj napriek občasným falošným detekciám. Pokiaľ je na snímke detegovaná viac ako jedna tvár, všetky detekcie jednoducho ignorujeme. Síce tak prídeme aj o niekoľko správnych detekcií, ale pokiaľ je osoba na zázname dlhšie, tak táto strata problém nerobí.

4.2 Rozpoznanie osoby na videu

Rozpoznávanie vykonáva funkcia compare_to_detected v súbore recognize.py. Osoby rozpoznávame kombináciou dvoch spôsobov: hľadaním podobnosti medzi snímkami tvárí a hľadaním podobnosti medzi snímkami celých postáv. Oba spôsoby využívajú rovnakú siamskú neurónovú sieť, ako sme popísali v podkapitole 3.3. Pre každú osobu si uchovávame niekoľko záberov jej tváre a postavy. Staré zábery sú nahradené novšími, takže napríklad aj zmena oblečenia v zábere kamery nespôsobí väčší problém. V prípade, že hľadáme konkrétnu osobu, vytvoríme nový záznam a uložíme do neho zábery, podľa ktorých chceme danú osobu hľadať. Ak chceme osobu identifikovať, porovnáme ju so všetkými uloženými osobami podľa tváre a podľa postavy. Jednotlivé výsledky pre tvár a postavu spriemerujeme (je možné nastaviť pomer, ako výsledkom veríme, základný je 1:1), nájdeme najväčšiu podobnosť a pokiaľ je táto podobnosť väčšia ako nastavený parameter, určíme, že sa jedná o tú istú osobu. Podobnosť tváre aj postavy pre dve osoby spočítame tak, že pomocou siamskej siete zistíme podobnosť medzi každými dvomi uloženými zábermi daných osôb. Podobnosť je reálne číslo medzi 0 a 1. Taktiež je možné nastaviť hodnotu threshold a podobnosť pretransformovať len na hodnoty 0 (ak je podobnosť menšia ako threshold) a 1 (inak). Ak nastavíme threshold väčší, vyhneme sa falošným (nesprávnym) zhodám, avšak môžeme prísť o niektoré správne zhody.

Pre zlepšenie presnosti je možné sieť dotrénovať na záberoch z kamier, na ktorých chceme program používať. Pri dotrénovaní sa využívajú predtrénované modely z kapitoly 3. Pri každom rozhodnutí, či sa jedná o rovnakú osobu, sa používateľovi zobrazí možnosť toto rozhodnutie potvrdiť alebo zamietnuť. Pokiaľ používateľ zamietne, môže sa mu zobraziť osoba s ďalšou najbližšou podobnosťou (podľa konfiguráciu). Týmto spôsobom vybudujeme nový dataset, ktorý využijeme na dotrénovanie rovnako, ako v podkapitole 3.4.

Ukázalo sa, že v reálnej prevádzke toto riešenie výrazne zlepšuje presnosť hľadania podobnosti. Je to spôsobené tým, že v novom datasete majú kamery rovnaké pozície ako v reálnej prevádzke, a tým pádom sú osoby zväčša snímané v podobných uhloch v porovnaní s datasetom DukeMTMC. Vďaka predtrénovaniu je potrebný pri trénovaní výrazne menši počet iterácií.

4.3 Sledovanie osoby na jednom videozázname

Po každej detekcii osoby získame nové obdĺžniky ohraničujúce postavu osoby na snímke. Strácame však informáciu o tom, ktorej osobe daný obdĺžnik patrí. Inak povedané, ak je po prvej detekcii osoba ohraničená obdĺžnikom, tak po ďalšej detekcii nevieme povedať, ktorý nový obdĺžnik ohraničuje tú istú osobu. Jednou možnosťou, ako tento problém riešiť, je po každej detekcii osôb znova klasifikovať každú detegovanú osobu na snímke. Hlavným problémom tohto riešenia je jeho časová zložitosť. Podľa počtu hľadaných alebo predtým videných osôb na zázname by porovnanie medzi každým z nich mohlo trvať až niekoľko sekúnd. Takéto riešenie je neakceptovateľné, pokiaľ sledujeme osoby na videozázname v reálnom čase.

Problém sme vyriešili algoritmom centroid tracking [16]. Pre každý obdĺžnik vypočítame jeho stred. Následne každý stred porovnávame so stredmi obdĺžnikov z predchádzajúcej detekcie. Predpokladáme, že páry stredov tvorené zo stredov z aktuálnej a prechádzajúcej detekcie s najmenšou vzdialenosťou patria vždy rovnakej osobe. Páry musia byť samozrejme disjunktné (každý stred patrí len do jedného páru). Pokiaľ niektorý nový stred nepatrí do žiadneho páru, jedná sa o novú osobu, v prípade starého stredu sa naopak táto osoba na zázname už nenachádza. Ak je vzdialenosť medzi niektorými pármi príliš veľká, takýto pár ignorujeme.

Toto riešenie má dva hlavné problémy. Keďže detekcia je vykonávaná len každých tridsať snímok, nové obdĺžniky získame tiež každých tridsať snímok. Preto najkratšia vzdialenosť nemusí nutne znamenať rovnakú osobu, napríklad pokiaľ osoby prejdú oproti sebe. Okrem tohto riešenia sme využili aj sledovanie, ktoré sa vykonáva každú snímku za využitia correlation tracker-u z knižnice dlib. Vďaka tomuto sledovaniu získame nové obdĺžniky každú snímku a tým pádom je centroid tracker oveľa presnejší, keďže vzialenosti medzi správnymi pármi stredov sú oveľa menšie.

Ďalší problém môže nastať, pokiaľ osoba odíde zo záberu kamery a v rovnakom čase a na rovnakom mieste príde do záberu iná osoba. V takomto prípade by algoritmus vyhodnotil, že sa jedná o tú istú osobu. Preto po každej detekcií porovnáme nájdenú osobu s osobou, ktorej identita je jej pridelená popísaným spôsobom. Pokiaľ sa nezhoduje, jedná sa o novú osobu.

4.4 Štruktúra projektu

Uvádzame krátky popis funkcionality súborov v projekte.

centroid_tracker.py Tracker umožňujúci uchovanie ID pre osoby medzi jednotlivými snímkami videa.

config.py Konfiguračné premenné.

detect.py Funkcie pre detekciu postavy a tváre.

download_dataset.py Stiahne všetky nutné datasety a uloží ich do správneho formátu.

feature_extractors.py Funkcie na extrakciu príznakov obrázku.

feature_test.py Testovanie extrahovaných príznakov.

help_functions.py Pomocné funkcie, ako je načítanie obrázkov a podobne.

improve.py Umožňuje vylepšovať model na základe novo-získaných dát.

improve_test.py Testovanie vylepšeného modelu.

person_track.py Trieda obsahujúca informácie, na základne ktorých vieme osobu identifikovať.

play.py Prehrávanie videa a detekcia/rozpoznávanie osôb v ňom.

players.py Triedy umožňujúce rôzne formáty videa (obrázky, video, stream z kamery, video na youtube).

recognize.py Funkcie umožňujúce rozpoznanie osôb na videu.

requirements.txt Nutné python knižnice.

retinex.py Filter na úpravu obrázkov.

siamese_network.py Siamská konvolučná neurónová sieť.

test.py Testovanie natrénovaného modelu siamskej siete.

train.py Trénovanie modelu siamskej siete.

train_help_functions.py Pomocné funkcie na trénovanie (generovanie dát).

ChokePoint Obsahuje snímky datasetu ChokePoint, zložka je automaticky vytvorená.

DukeMTMC-reID Obsahuje snímky datasetu DukeMTMC, zložka je automaticky vytvorená.

known_people Obsahuje snímky osôb, ktoré chceme na videozáznamoch hľadať.

mobilenet_ssd Obsahuje neurónovú sieť MobileNet-SSD.

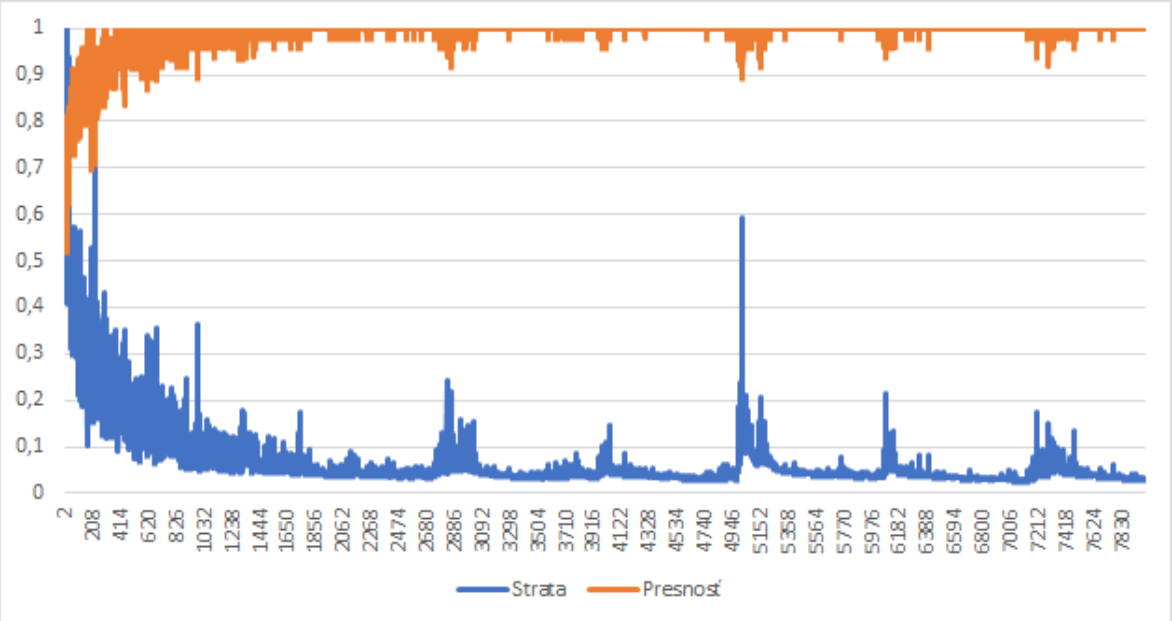
model_history Obsahuje predtrénované modely na rozpoznávanie osôb.

4.5 Výsledky a diskusia

Zo záberov troch kamier z databázy ChokePoint sme vybudovali nový dataset. Na tomto datasete sme siete na rozpoznávanie podľa tváre a postavy dotrénovali. Následne sme zobrali ďalšie zábery z tých istých kamier, na ktorých sa vyskytovali tie isté osoby v inom oblečení a aj nové, predtým nevidené osoby. Z týchto záberov sme opäť vybudovali dataset, na ktorom sme testovali rovnako, ako pri prvotnom trénovaní.

Môžeme si všimnúť, že presnosť siete na rozpoznávanie podľa postavy sa zlepšila, ale siete na rozpoznávanie podľa tváre nie. Je to spôsobené tým, že ak detegujeme osobu, najskôr detegujeme jej postavu a až následne tvár. Avšak tvár nie vždy nájdeme. Preto môže byť vytvorený dataset menší, čo môže spôsobiť preučenie. Po preskúmaní datasetu sme však zistili, že detekcia tváre je niekedy nepresná a program uloží aj zábery, ktoré tvár nezachytávajú. Preto je potrebné dataset manuálne vyčistiť od týchto falošných detekcií pre dosiahnutie lepších výsledkov.

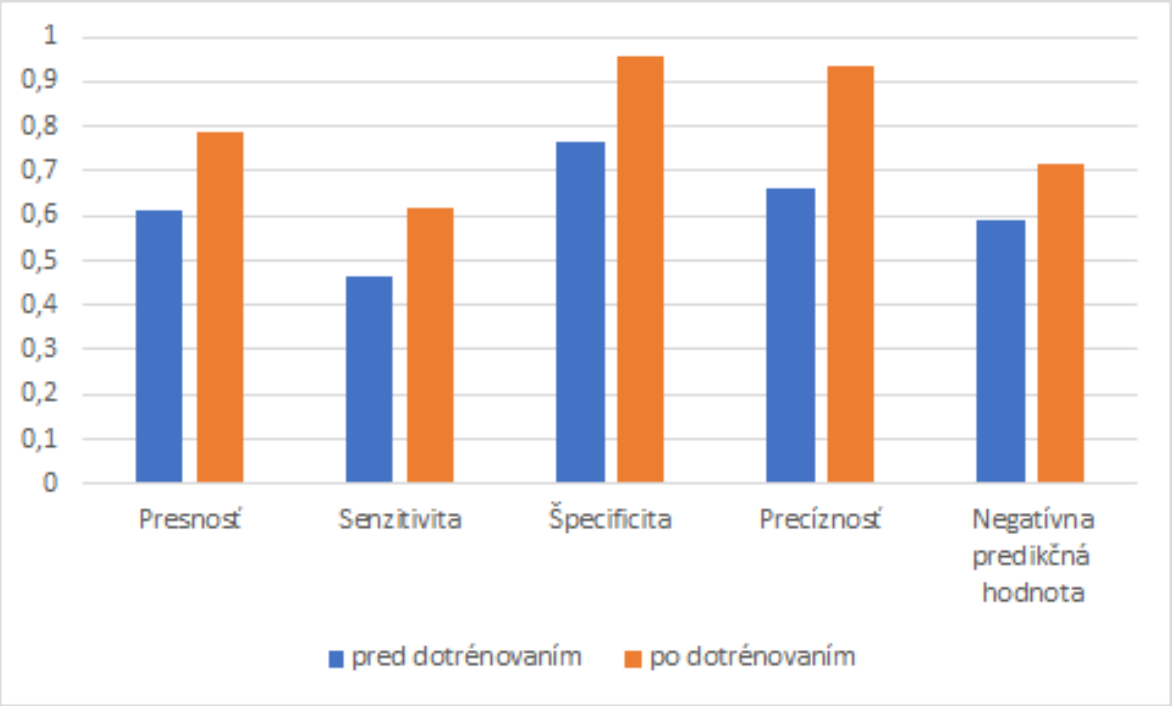
Napriek tomu však sledovanie osôb v zázname a taktiež aj hľadanie známych osôb podľa tváre aj postavy funguje. Tým, že kombinujeme dve riešenia a máme uložených viacero snímok pre každú osobu, problém zaniká. Na záberoch, na ktorých sme vykonali testovanie, program dokázal identifikovať všetky osoby správne. Jedná sa síce o krátky záznam, ale ukazuje, že náš systém je možné využiť na spoľahlivé sledovanie a vyhľadávanie osôb.



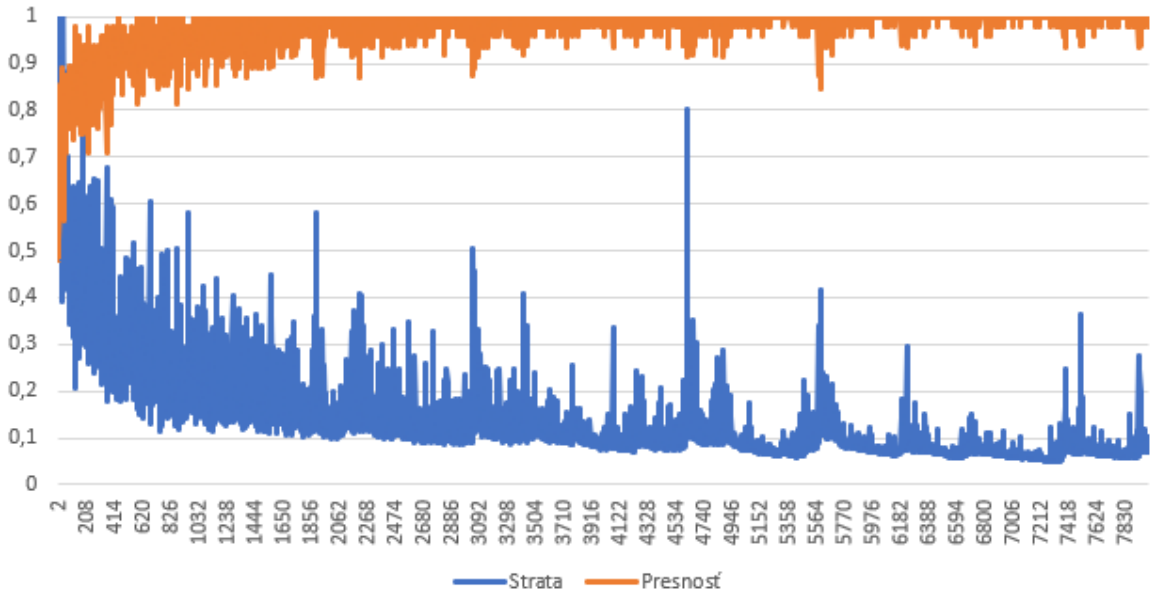
Obr. 4.1: Priebeh doučenia hľadania podobnosti tvárí. Z grafu vidieť, že na dotrénovanie siete stačí malý počet iterácií.

Tabuľka 4.1: Porovnanie úspešnosti zhôd pred a po dotrénovaní rozpoznávania podľa tváre.

		Predikcia pred dotrénovaním		Predikcia po dotrénovaní	
		Zhoda	Nezhodna	Zhoda	Nezhodna
Realita	Zhoda	810	590	624	776
	Nezhoda	103	1297	13	1387



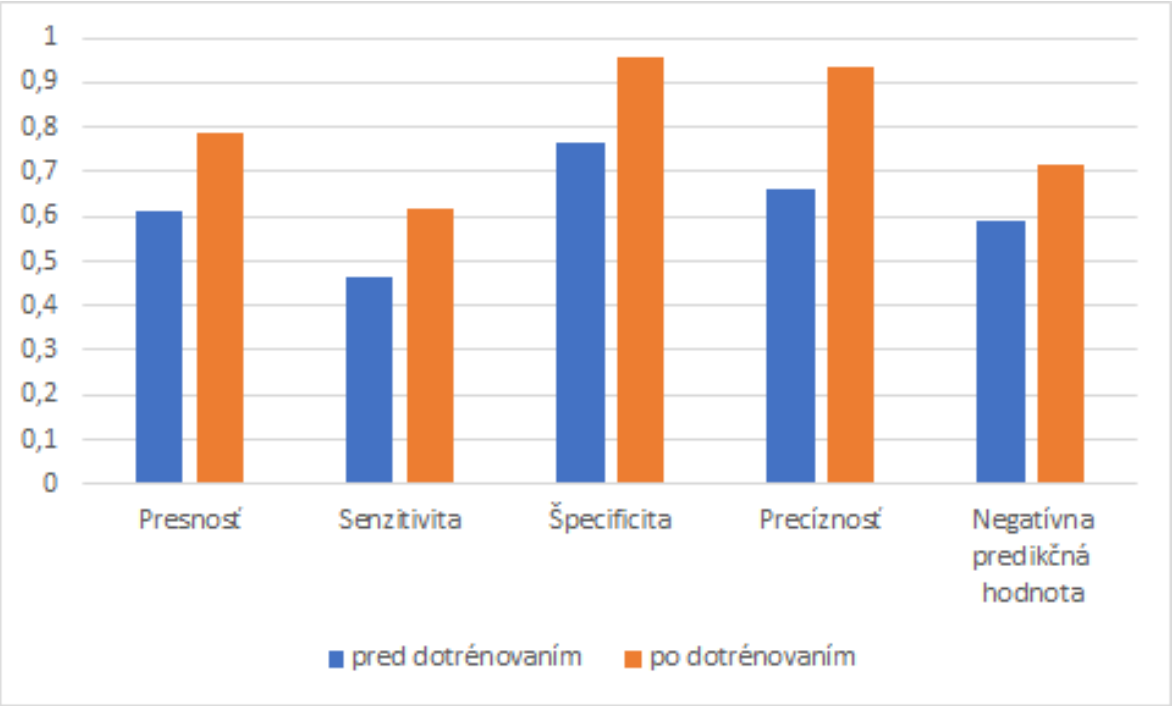
Obr. 4.2: Porovnanie úspešnosti zhôd pred a po dotrénovaní rozpoznávania podľa tváre.



Obr. 4.3: Priebeh doučenia hľadania podobnosti postáv. Na dotrénovanie je potrebný opäť len malý počet iterácií.

Tabuľka 4.2: Porovnanie úspešnosti zhôd pred a po dotrénovaní rozpoznávania podľa postavy.

		Predikcia pred dotrénovaním		Predikcia po dotrénovaní	
		Zhoda	Nezhodna	Zhoda	Nezhodna
Realita	Zhoda	696	804	925	575
	Nezhoda	352	1148	66	1434



Obr. 4.4: Porovnanie úspešnosti zhôd pred a po dotrénovaní rozpoznávania podľa postavy.

Záver

Výsledkom práce je aplikácia umožňujúca sledovanie osôb vo viacerých videozáznamoch a taktiež aj vyhľadávanie osôb podľa zadaného vizuálneho vzoru: buď tváre alebo postavy. Osoby v aplikácii rozpoznávame kombináciou dvoch spôsobov, a to podľa tváre a celého tela, v oboch prípadoch s využitým siamskej neurónovej siete. Táto sieť porovnáva dve snímky a rozhoduje, či sa na nich nachádza rovnaká osoba. Aplikácia dosahuje dobré výsledky pri testovaní na datasete ChokePoint.

Okrem toho sme v práci popísali viacero spôsobov detekcie osôb na videozázname. Dva zo spôsobov, pomocou haarových príznakov a pomocou histogramu orientovaných gradientov, sa ukázali pre našu aplikáciu ako nevhodné. Dosahovali horšie výsledky a navyše boli výrazne pomalšie ako detekcia pomocou neurónovej siete.

Osoby sme vyskúšali klasifikovať aj pomocou extrahovaných príznakov a metód najbližších susedov a podporných vektorov do dopredu určeného počtu tried. V porovnaní s najlepšími zverejnenými riešeniami sme dosiahli slabšie výsledky. Väčším problémom však bolo, že metóda nebola dostatočne univerzálna a v porovnaní so siamskou neurónovou sieťou aj výrazne pomalšia.

Aplikáciu je možné ďalej vylepšovať. Možnosťou ďalšej práce by mohlo byť sledovanie osôb vo väčšom dave ľudí. Klasifikácia v tomto prípade zlyháva, pretože osoby sa prekrývajú a na snímke, podľa ktorej by sme osobu mali klasifikovať, sa nachádza viacero ľudí. Aplikovaním pokročilejších algoritmov sledovania by sme s vysokou pravdepodobnosťou dosahovali ešte lepšie výsledky.

Napriek tomu sme ukázali, že riešenie pomocou siamskej neurónovej siete je vhodné v systémoch na sledovanie osôb vo viacerých záznamoch. Doučenie siete pomocou vybudovania nového datasetu vytvoreného zo záberov kamier, na ktorých by sa aplikácia mala reálne využiť, výrazne zvyšuje jej presnosť a robí je univerzálnym riešením pre rozpoznávanie, sledovanie a vyhľadávanie osôb na videozáznamoch.

Literatúra

[1] Panth Bhavsar. Object detection with convolutional neural networks, Január 2018. [online]. Cit. 2019-04-14. Dostupné na internete: https://medium.com/datadriveninvestor/object-detection-with-convolutional-neural-networks-dde190eb7180.

[2] Matt Brems. A one-stop shop for principal component analysis, Apríl 2017. [online]. Cit. 2019-04-09. Dostupné na internete: https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c.

[3] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *In CVPR*, pages 886–893, 2005.

[4] Adit Deshpande. A beginner's guide to understanding convolutional neural networks, Júl 2016. [online]. Cit. 2019-04-14. Dostupné na internete: https://adeshpande3.github.io/A-Beginner

[5] Tom Fawcett. An introduction to roc analysis. In *Pattern Recognition Letters 27*, pages 861–874. 2005.

[6] Nathan Glover. Hsv vs. rgb, Október 2016. [online]. Cit. 2019-04-06. Dostupné na internete: https://handmap.github.io/hsv-vs-rgb/.

[7] Onel Harrison. Machine learning basics with the k-nearest neighbors algorithm, September 2018. [online]. Cit. 2019-04-10. Dostupné na internete: https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761.

[8] Gregory Koch. Siamese neural networks for one-shot image recognition. Master's thesis, University of Toronto, 2015.

[9] Shengcai Liao, Yang Hu, Xiangyu Zhu, and Stan Li. Person re-identification by local maximal occurrence representation and metric learning. pages 2197–2206, 06 2015.

[10] Shengcai Liao, Guoying Zhao, Vili Kellokumpu, Matti Pietikäinen, and Stan Li. Modeling pixel process with scale invariant local patterns for background subtraction in complex scenes. volume 0, pages 1301–1306, 06 2010.

[11] Satya Mallick. Histogram of oriented gradients, December 2016. [online]. Cit. 2019-04-13. Dostupné na internete: https://www.learnopencv.com/histogram-of-oriented-gradients/.

[12] Md Abdur Rahim, Md Shafiul Azam, Nazmul Hossain, and Md Rashedul Islam. Face recognition using local binary patterns (lbp). *Global Journal of Computer Science and Technology*, 2013.

[13] Mithi. Vehicle detection with hog and linear svm, Marec 2017. [online]. Cit. 2019-04-13. Dostupné na internete: https://medium.com/@mithi/vehicles-tracking-with-hog-and-linear-svm-c9f27eaf521a.

[14] Savan Patel. Svm (support vector machine) — theory, Máj 2017. [online]. Cit. 2019-04-11. Dostupné na internete: https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72.

[15] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European Conference on Computer Vision workshop on Benchmarking Multi-Target Tracking*, 2016.

[16] Adrian Rosebrock. Simple object tracking with opencv, Júl 2018. [online]. Cit. 2019-04-13. Dostupné na internete: https://www.pyimagesearch.com/2018/07/23/simple-object-tracking-with-opencv/.

[17] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[18] David Stutz. Retinex theory and algorithm, Jún 2015. [online]. Cit. 2019-04-16. Dostupné na internete: https://davidstutz.de/retinex-theory-and-algorithm/.

[19] Xiaoyang Tan and Bill Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. In *IEEE Transactions on Image Processing*, pages 1635–1650, 2010.

[20] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, Dec 2001.

[21] Yongkang Wong, Shaokang Chen, Sandra Mau, Conrad Sanderson, and Brian C. Lovell. Patch-based probabilistic image quality assessment for face selection and improved video-based face recognition. In *IEEE Biometrics Workshop, Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 81–88. IEEE, June 2011.

[22] Zhedong Zheng. State-of-the-art. [online]. Cit. 2019-04-22. Dostupné na internete: https://github.com/layumi/DukeMTMC-reID_evaluation/tree/master/State-of-the-art.

[23] Zhedong Zheng, Liang Zheng, and Yi Yang. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

Príloha

Súčasťou práce je DVD príloha obsahujúca všetky zdrojové súbory. Ďalej obsahuje niektoré vypočítané dáta a ukážky využitých datasetov. Zdrojové súbory sa tiež dajú nájsť v repozitári projektu na GitHube na adrese https://github.com/AndrejZbin/Diplomovka