VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

IDENTIFIKACE OSOB POMOCÍ OBRAZU DUHOVKY

IDENTIFICATION OF INDIVIDUALS USING IRIS IMAGE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MAREK ŽÁKOVIC
AUTHOR

VEDOUCÍ PRÁCE — Ing. MAREK VAŠKO
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

148690

Ústav: Ústav počítačové grafiky a multimédií (UPGM)
Student: **Žákovic Marek**
Program: Informační technologie
Specializace: Informační technologie
Název: **Identifikace osob pomocí obrazu duhovky**
Kategorie: Zpracování obrazu
Akademický rok: 2022/23

Zadání:

1. Prozkoumejte oblast identifikace osob pomocí obrazu duhovky.
2. Vyhledejte a popište moderní metody rozpoznávání osob založené na strojovém učení.
3. Navrhněte systém pro identifikaci osob pomocí duhovky.
4. Najděte si vhodné datové sady pro experimenty.
5. Iterativně vylepšujte návrh identifikačního systému a experimentujte.
6. Diskutujte o možném budoucím vývoji.
7. Vytvořte krátké video a plakát k dosaženým výsledkům.

Literatura:

- Goodfellow, I., Bengio, Y. a Courville A. Deep Learning. MIT Press. 2016.
- He, K., Zhang, X., Ren, S. a Sun, J. Deep residual learning for image recognition. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. December 2016.
- Deng, J., Guo, J., Yang, J., Xue, N., Cotsia, I. et al. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2021.
- Jia, D. a Shen, W. Is normalized iris optimal for iris recognition based on deep learning?. Journal of Electronic Imaging. 2021.
- Alinia Lat, R., Danishvar, S., Heravi, H. a Danishvar, M. Boosting Iris Recognition by Margin-Based Loss Functions. Algorithms. 2022.
- Automation (CASIA), C. A. of Sciences' Institute of. CASIA Iris Image Database. http://biometrics.idealtest.org/.

Při obhajobě semestrální části projektu je požadováno:
Body zadání 1, 2, značné rozpracování bodů 3, 4 a 5.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Vaško Marek, Ing.**
Vedoucí ústavu: Černocký Jan, prof. Dr. Ing.
Datum zadání: 1.11.2022
Termín pro odevzdání: 10.5.2023
Datum schválení: 9.5.2023

Abstrakt

Cieľom tejto bakalárskej práce bolo vytvoriť systém pre identifikáciu osôb pomocou obrazu dúhovky. V práci sú popísané už existujúce metódy a postupy rozpoznávania osôb pomocou dúhovky. Mnou navrhnutá metóda využíva konvolučnú neurónovú sieť natrénovanú za účelom extrakcie príznakov. Tie následne slúžia na porovnanie, či ide o tú istú osobu alebo nie. Experimenty zahŕňajú trénovanie a vyhodnocovanie neurónovej siete. Pre účely tejto práce boli použité voľne dostupné dátové sady, ktoré boli v rámci práce upravené pre konkrétne použitie.

Abstract

The goal of this bachelor's thesis was to create a system for person identification using iris images. The thesis describes existing methods and procedures for iris recognition. The proposed method utilizes a convolutional neural network trained to extract features, which are then used to compare whether the image belongs to the same person or not. The experiments involve training and evaluating the neural network. For the purposes of this thesis, freely available datasets were used, which were modified for specific use.

Kľúčové slová

Identifikácia pomocou obrazu dúhovky, konvolučné neurónové siete, počítačové videnie, extrakcia príznakov, klasifikácia, kontrastívne učenie.

Keywords

Iris recognition, convolutional neural networks, computer vision, feature extraction, classification, contrastive learning.

Citácia

ŽÁKOVIC, Marek. *Identifikace osob pomocí obrazu duhovky.* Brno, 2023. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Marek Vaško

Identifikace osob pomocí obrazu duhovky

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Ing. Marek Vaška. Uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

. . . . . .
Marek Žákovic
9. mája 2023

Poďakovanie

Rád by som poďakoval môjmu vedúcemu práce Ing. Marekovi Vaškovi za jeho ochotu, cenné rady a čas, ktorý mi venoval pri vedení tejto práce.

Obsah

1 Úvod **2**

2 Tradičné postupy pre identifikáciu osôb pomocou oka **3**
2.1 Základné princípy identifikácie pomocou obrazu dúhovky 3
2.2 Limitácie a výzvy pri identifikácii pomocou dúhovky 4
2.3 Existujúce systémy 4

3 Neurónové siete a ich použitie pri extrakcii príznakov **6**
3.1 Konvolučné neurónové siete 6
3.2 Architektúry neurónových sietí na klasifikáciu obrazu 8
3.3 Možnosti extrakcie príznakov 11
3.4 Metódy vyhodnocovania presnosti klasifikátora 17

4 Návrh a implementácia systému **20**
4.1 Dátové sady a príprava vstupných dát 20
4.2 Návrh riešenia 23
4.3 Pracovné prostredie a použité systémy 25

5 Priebeh a výsledky experimentov **26**
5.1 Prvý experiment: výber trénovacieho procesu 26
5.2 Druhý experiment: výber hĺbky siete 28
5.3 Tretí experiment: overenie nutnosti segmentácie a normalizácie vstupných dát 30

6 Záver **32**

Literatúra **33**

A Obsah priloženého pamäťového média **37**

Kapitola 1

Úvod

Už od počiatku ľudstva sa ľudia začleňovali do rôznych spoločenských celkov pomocou tetovaní, oblečenia alebo rôznych zvykov. S vývojom spoločnosti sa postupne zvyšovala potreba kvalitnej identifikácie. V dnešnej dobe, kedy je cestovanie po celom svete otázka pár hodín, v dobe mobilných sietí a internetu, je kvalitná identifikácia osôb extrémne náročná no zato nevyhnutná. Firma potrebuje mať prehľad o svojich zamestnancoch a návštevníkoch, polícia potrebuje mať prehľad o podozrivých, poprípade hľadaných osobách. Na základe totožnosti sú ľuďom udelované rôzne privilégiá, poprípade obmedzenia.

Keďže je problém identifikácie taký dôležitý, existuje niekoľko rôznych prístupov, ako overiť totožnosť človeka. Najbežnejšie overovacie a identifikačné systémy sú dnes založené na používaní rôznych hesiel a osobných identifikačných čísel, alebo aj dokumentov ako napríklad občiansky preukaz alebo pas. Takéto systémy sú ale veľmi zraniteľné. Často trpia rôznymi krádežmi alebo falšovaním. Preto čoraz väčší záujem vzbudzujú metódy biometrickej identifikácie. Tie umožňujú overiť totožnosť človeka podľa jeho fyziologických charakteristík, ktoré sú veľmi obtiažne napodobniteľné.

Moja práca sa zaoberá identifikáciou osôb na základe očnej dúhovky. Samotná identifikácia prebieha pomocou dopredu natrénovanej neurónovej siete. Cieľom tejto práce bolo čo najlepšie natrénovanie neurónovej siete na extrakciu príznakov z obrazu dúhovky. Tieto príznaky boli následne použité za účelom identifikácie. V rámci práce boli prevedené experimenty s viacerými trénovacími métódami na viacerých dátových sadách. Celá práca je členená do jednotlivých kapitol, ktoré popisujú potrebné informácie na vypracovanie práce.

V druhej kapitole sú popísané základné princípy identifikácie osôb pomocou dúhovky, limitácie ktoré takýto prístup má a už existujúce systémy, ktoré sa touto metódou zaoberajú. Tretia kapitola sa venuje umelým neurónovým sieťam a ich využitiu v rámci identifikácie osôb. V štvrtej kapitole je predstavený mnou navrhnutý systém pre riešenie otázky identifikácie pomocou dúhovky. V kapitole číslo päť sú popísané experimenty, ktoré boli vykonané v rámci riešenia tejto práce. Výsledky sú následne zhrnuté v závere práce.

Kapitola 2

Tradičné postupy pre identifikáciu osôb pomocou oka

Identifikácia osôb pomocou oka patrí medzi najspoľahlivejšie metódy biometrického rozpoznávania. Pri identifikácii osôb pomocou oka sa využíva fakt, že ľudské oko je u každej osoby unikátne. Jedným z najdôležitejších prvkov oka slúžiaceho na identifikáciu je dúhovka, ktorá v sebe obsahuje jedinečnú kombináciu farieb a vzorov. Ďalej je dúhovka relatívne stabilná a nemení sa behom života osoby, a vďaka tomu môže byť použitá ako spoľahlivá biometrická metóda pre identifikáciu[28]. Táto metóda sa často využíva ako alternatíva pre identifikáciu pomocou odtlačkou prstov, ktoré sa môžu ľahko mechanicky poškodiť, poprípade so zvyšujúcou sa dĺžkou života zmeniť. Veľkou výhodou použitia tejto metódy je, že skenovacie prístroje a zariadenia použité pri získavaní dát sú rýchle, spoľahlivé a neinvazívne.

2.1 Základné princípy identifikácie pomocou obrazu dúhovky

Pri identifikácii osôb pomocou obrazu dúhovky sa zvyčajne postupuje podľa nasledujúcich krokov:

1. Získanie snímky oka: Existujú viaceré možnosti ako získať snímku oka danej osoby. Snímka môže byť napríklad extrahovaná z bežnej fotografie človeka. Vtedy musí byť brané do úvahy, že finálny produkt nemusí byť dostačujúci. Oveľa spolahlivejšia metóda je zhotovenie snímky oka pomocou špeciálnej kamery, ktorá zabezpečuje dostatočnú kvalitu na ďalšie spracovanie.

2. Spracovanie snímky oka: Po získaní snímky oka sa musí takáto snímka ďalej spracovať. To zahŕňa rôzne úkony, ktoré závisia od spôsobu získania snímky, ako aj od následného použitia. Medzi ne patrí napríklad odstránenie šumu, odstránenie nežiadúceho odrazu svetla zo zrenice, normalizácia obrázku a segmentácia dúhovky. Táto segmentácia je kľúčovým krokom, pretože umožňuje extrahovať unikátne vlastnosti z dúhovky, ktoré sa neskôr použijú pri identifikácii osoby.

3. Extrakcia biometrických dát: Po segmentácii dúhovky sa extrahujú rozsiahle biometrické dáta zo získanej dúhovky. Tie sú následne použité na vytvorenie biometrického profilu osoby.

4. Porovnávanie profilov: Biometrický profil osoby sa následne porovnáva s databázou uložených biometrických profilov, aby sa určila identita danej osoby.

5. Identifikácia osoby: V prípade ak sa nájde zhoda medzi biometrickymi profilmi, identifikácia je považovaná za úspešnú, daná osoba je identifikovaná a jej identita overená. V prípade že sa zhoda nenájde, je osoba považovaná za neznámu a identifikácia prebehla neúspešne.

Existuje mnoho rôznych metód na extrakciu a následné porovnávanie biometrických dát dúhovky. Medzi najpoužívanejšie patrí takzvané Iris kódovanie (angl. *IrisCode* [5]): táto metóda sa zameriava na extrakciu špecifických vlastností dúhovky a vytvorenie unikátneho kódu pre každého jednotlivca. Počas procesu vytvárania je využitá metóda *2D Gabor Wavelets* na prevod obrazu extrahovanej dúhovky do binárnej podoby - vektora. Tento vektor je následne porovnaný s uloženými šablónami pri identifikácii. Ďalšou metódou je tzv. hlboké učenie (angl. *Deep learning*), kedy sa na získavanie a spracúvanie biometrických dát využíva neurónová sieť a strojové učenie[52]. Touto metódou sa budem ďalej zaoberať v tejto práci.

2.2 Limitácie a výzvy pri identifikácii pomocou dúhovky

Aj napriek tomu že je identifikácia pomocou dúhovky považovaná za spoľahlivú a kvalitnú, existujú určité limitácie a výzvy, ktoré môžu ovplyvniť konečnú presnosť výsledkov.

Jednou z najčastejších limitácií je vplyv okolitého prostredia na kvalitu získaných snímok oka. Napríklad príliš jasné alebo príliš tmavé prostredie, nedostatočné osvetlenie alebo rôzne odlesky na zrenici oka môžu viesť k chybným výsledkom[30]. Naviac existuje možnosť, že niektoré ochorenia, poprípade lieky[12], môžu ovplyvniť štruktúru dúhovky, čo môže mať za následok chybný výsledok.

Ďalšiu limitáciu predstavuje rôznorodosť biometrických dát získaných z rôznych zdrojov. Každý snímok oka je unikátny a môže sa líšiť v parametroch ako je napríklad veľkosť, rozlíšenie alebo uhol záberu[30]. Tieto rozdiely môžu mať negatívny dopad na kvalitu výsledku. Ďalej existuje riziko útoku, kedy sa útočník pokúsi o identifikáciu pomocou napodobeniny oka alebo manipuláciou obrazu dúhovky[32].

Je nutné pri návrhu systému na identifikáciu dané limitácie zohľadniť.

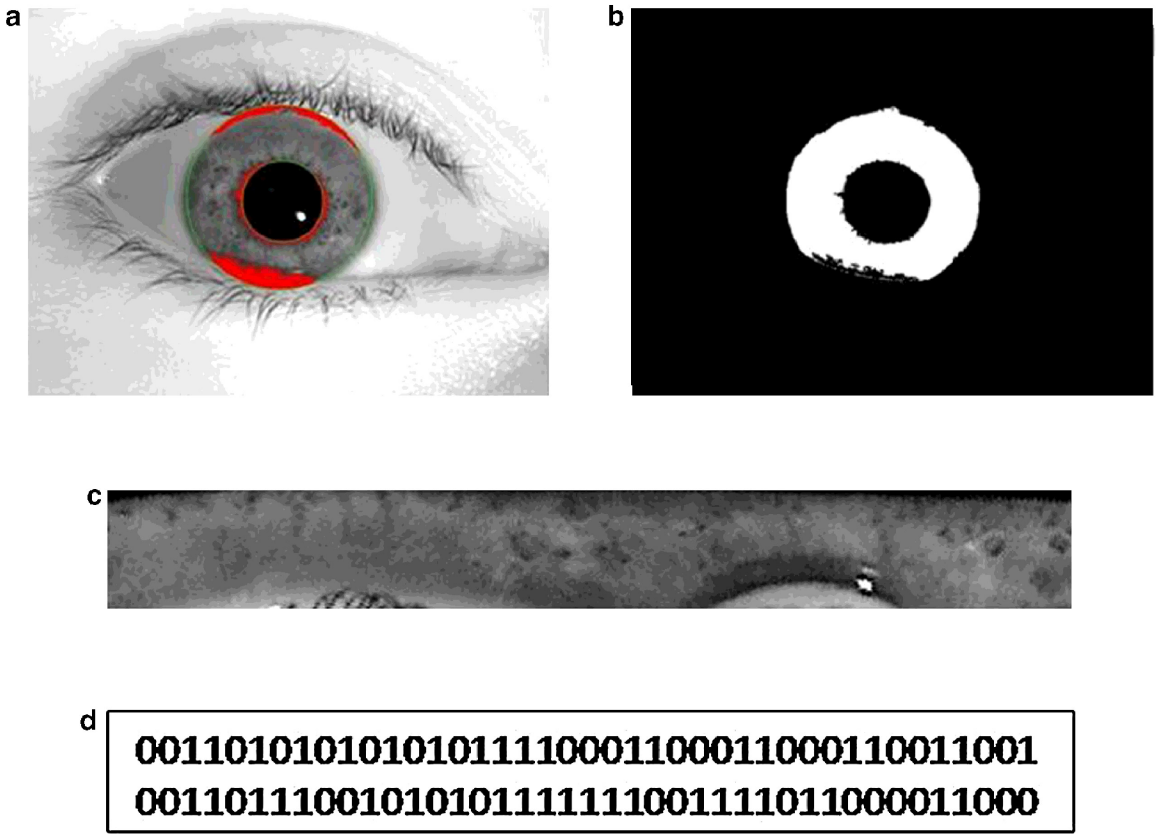
2.3 Existujúce systémy

OSIRIS [27] je open-source nástroj na rozpoznávanie pomocou obrazu dúhovky. Tento softvér sa skladá predovšetkým zo štyroch kľúčových modulov (viď obrázok 2.1, a to segmentácie, normalizácie, extrakcie príznakov a porovnávania vzorov (angl. *template matching*). Proces spracovania a porovnania obrázkov je následovný:

- Segmentácia: slúži na detekciu a extrakciu obrazu dúhovky. Má za úlohu nájsť a izolovať textúru dúhovky od okolitých elementov, ako napríklad viečka alebo mihalnice. Extrakcia príznakov je tým pádom limitovaná len na oblasť dúhovky. Ďalej tento modul generuje binárnu masku, ktorá udáva, ktoré pixely obrázka patria dúhovke. Táto maska je neskôr použitá pri porovnávaní vzorov.

- Normalizácia: slúži na mapovanie extrahovaného obrazu dúhovky na jednotnú šírku a výšku - normalizovaný obraz dúhovky. Proces normalizácie dovoluje porovnanie akýchkoľvek dvoch dúhoviek, nezávisle na veľkosti pôvodného obrázku. Tento proces využíva *Daugmanov rubber-sheet model* [31], pomocou ktorého je prevedený kruhový obraz dúhovky na obdĺžnikový blok pevnej šírky a výšky.

- Extrakcia príznakov: má za úlohu extrahovať charakteristiky textúry danej dúhovky. Tieto príznaky sú základným prvkom pre porovnávanie akýchkoľvek dvoch obrázkov. Výsledok tohto modulu je IrisCode pre danú dúhovku, opísaný vyššie. Ten následne predstavuje vstup do porovnávacieho modulu.
- Porovnávanie vzorov: má za úlohu je rozhodnúť, či dané dva vzorky IrisCode patria rovnakej dúhovke alebo nie. Pre tieto účely je vypočítané skóre podobnosti, ktoré je následne porovnané s prahom (anglicky *treshold*).



Obr. 2.1: Ukážka jednotlivých častí systému OSIRIS: a) segmentácia dúhovky a odstránenie nežiadúcich kontúr, b) maska dúhovky, c) normalizovaný obraz dúhovky, d) IrisCode (prebrané z [27])

Tento nástroj budem ďalej využívať v rámci svojej práce, a to konkrétne za účelom segmentácie a normalizácie dúhovky. Keďže systém OSIRIS podľa autorov [27] dosiahol state-of-the-art výsledky, využijem ho aj ako východiskový systém (angl. *baseline system*) pre porovnávanie výsledkov.

Kapitola 3

Neurónové siete a ich použitie pri extrakcii príznakov

Strojové učenie je oblasť umelej inteligencie, ktorá umožňuje počítačovým systémom získavať isté "skúsenosti" z dát a následne tieto "skúsenosti" využiť na riešenie rôznych problémov. Práve tento fakt nám umožňuje vytvárať programy a systémy, ktoré sú schopné zlepšovať svoje vlastné výkony bez ďalšieho explicitného zásahu programátora. V rámci strojového učenia sú vyvíjané algoritmy, ktoré umožňujú strojom zlepšovať svoje výkony na základe vstupných dát.

Vstupom do strojového učenia sú trénovacie dáta ktoré môžu byť v rôznych formátoch, napríklad číselné, textové, zvukové alebo obrazové. Cieľom strojového učenia je vytvoriť model, ktorý je schopný čo najlepšie predikovať výstup na základe vstupu. Tento model môže byť neskôr použitý na riešenie konkrétnych úloh, ako napríklad detekciu objektov na obrázku, rôzne klasifikačné úlohy alebo ako generátor textu.

V strojovom učení sa často používa rozlišovanie medzi učením s učiteľom (angl. *supervised learning*) a učením bez učiteľa (angl. *unsupervised learning*). Pri učení s učiteľom sa model učí na trénovacích dátach, ktoré majú označenie (angl. *label* alebo *target*) s očakávanými výstupmi. Cieľom je naučiť model čo najpresnejšie predikovať tieto výstupy na základe vstupných dát. Pri učení bez učiteľa sú využité dáta bez označenia. Pri takomto prístupe sa model učí identifikovať rôzne vzory a štruktúry v dátach bez očakávaných výstupov.

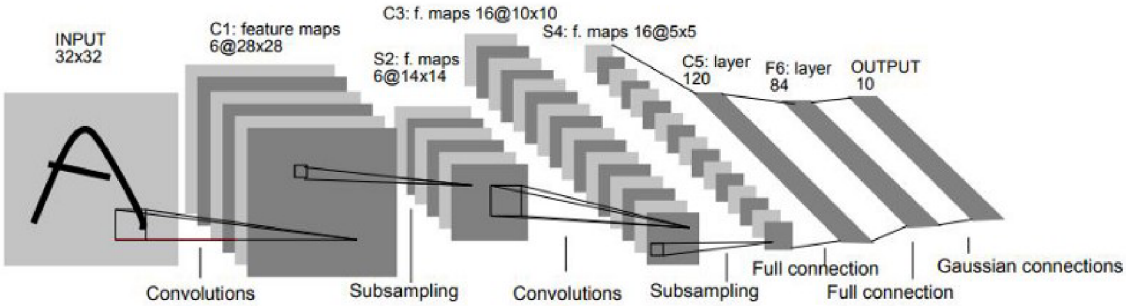
Model neurónovej siete môže byť použitý aj na získavanie vektorov príznakov (anglicky *feature extraction*) zo vstupných dát. Tento proces sa zvyčajne používa pri úlohách, kde sú vstupné dáta vysoho dimenzionálne. Výsledkom extrakcie príznakov je tzv. vektor príznakov, ktorý obsahuje dôležité informácie o vstupných dátach. Extrahovanie príznakov sa často používa v oblastiach so zložitými vstupnými dátami ako napríklad klasifikácia snímok mozgovej aktivity[35] alebo rozpoznávanie hovorenej reči[9].

3.1 Konvolučné neurónové siete

Konvolučná neurónová sieť (anglická skratka *CNN* - ďalej sa bude používať táto skratka) je typ umelej neurónovej siete, ktorá obsahuje aspoň jednu konvolučnú vrstvu (anglicky *convolution layer*)(viď obrázok 3.1). Konvolúcia je matematická operácia, ktorá slúži na spracovanie vstupných dát. Pri konvolúcii sa berie matica vstupných dát - napríklad matica pixelov z obrázka - a aplikuje na na ňu filtrovacia matica, ktorá slúži na detekciu určitých funkčných čŕt. Medzi charakteristické črty radíme napríklad hranu alebo celistvý geomet-

rický tvar. Výstupom konvolučnej operácie je nová matica, ktorá v sebe obsahuje informácie o prítomnosti daných čŕt v jednotlivých častiach vstupného obrázka (viď obrázok 3.2). To má za následok, že sieť je schopná automatickej extrakcie rôznych vlastností vstupných dát, ktoré sú ďalej použité pri klasifikácii alebo regresii.

Hlavným rozdielom medzi CNN a umelou neurónovou sieťou (anglická skratka *ANN*) je ten, že CNN sú primárne využívané v oblasti rozpoznávania vlastností v rámci obrázku. To dovoľuje použitie špecifických vlastností obrázku v rámci architektúry, čo má za následok, že finálna sieť je vhodnejšia na prácu s obrázkami[26].

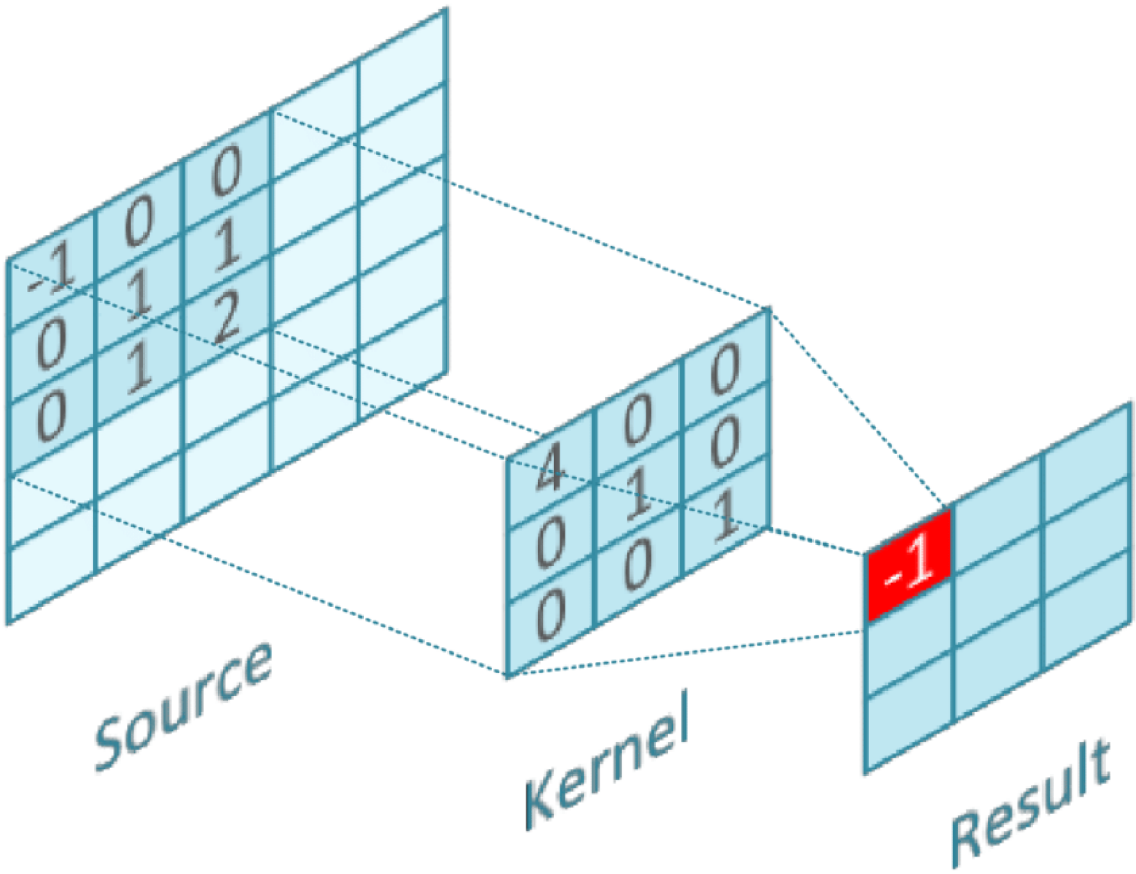


Obr. 3.1: Jedna z prvých konvolučných neurónových sietí s názvom LeNet-5, určená na rozpoznávanie ručne písaných a strojovo tlačených charakterov (prebrané z [21])

Rôzne vrstvy v CNN plnia rôzne úlohy. Medzi najviac využívané vrstvy patrí:

- Vstupná vrstva - táto vrstva prijíma vstupné dáta a prenáša ich do siete.
- Konvolučná vrstva - táto vrstva určuje typ výstupu neurónov. Na to využíva sady konvolučných filtrov (anglicky *kernel*) ktoré majú definované sady váh a veľkosť samotného filtra. Typicky je veľkosť filtra 3 x 3 (tri pixely na výšku aj na šírku), zriedkavejšie sa používa filter o veľkosti 5 x 5. Vstupné dáta, reprezentované ako n-dimenzionálny vektor, sú konvolvované s týmito filtrami a generujú výstupnú mapu príznakov[2]. Konvolučná vrstva pracuje v jeden moment len s malou časťou obrázka, túto časť nazývame recepčné pole[22].
- Poolingová vrstva - táto vrstva slúži na zjednodušenie mapy príznakov generovanej konvolučnou vrstvou tým že ju zredukuje. Poznáme niekoľko poolingových metód, medzi najpoužívanejšie patrí tzv. max pooling, ktorý vyberá z recepčného pola maximálnu hodnotu. Druhá najpoužívanejšia metóda, tzv. average pooling, vytvorí priemernú hodnotu v recepčnom poli.
- Plne prepojená vrstva - táto vrstva obecne slúži na klasifikáciu. Zvyčajne sa nachádza na konci neurónovej siete. V prípade klasifikačnej úlohy má posledná plne prepojená vrstva nastavený počet výstupných kanálov na počet tried v datasete, vďaka čomu vie sieť určiť, pod akú triedu spadá daný obrázok. Ďalej môže plne prepojená vrstva fungovať na generáciu vektorov príznakov. V takom prípade je počet výstupných kanálov upravený na žiadaný počet dimenzií konečného vektora.
- Dropout vrstva - táto vrstva počas trénovania náhodne vynásobí niektoré hodnoty predchádzajúcej vrstvy nulou, čím zabraňuje pretrénovaniu a zvyšuje schopnosť generalizácie siete.

- Batch normalizácia - technika založená na normalizácii vstupných dát. Dáta sú pred vstupom normalizované s cieľom odstránenia extrémnych hodnôt aktivácií. Dôsledkom normalizácie je zrýchlenie trénovania a zlepšenie výkonu siete.



Obr. 3.2: Konvolúcia obrázku o veľkosti 5x5 s filtrom o veľkosti 3x3. Filter bude aplikovaný na každú možnú pozíciu obrázka (prebrané z [49]).

3.2 Architektúry neurónových sietí na klasifikáciu obrazu

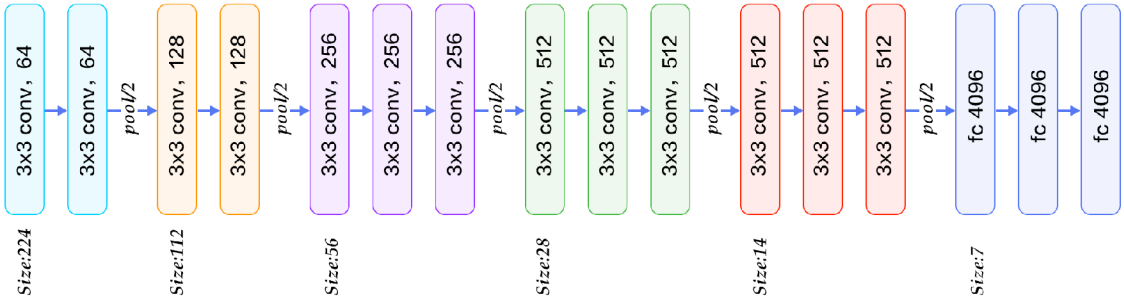
Architektúr neurónových sietí vyvinutých za účelom klasifikácie obrázkov je mnoho. Preto sa každoročne koná medzinárodná súťaž ILSVRC (skratka pre *ImageNet Large Scale Visual Recognition Challenge*). V rámci tejto súťaže sú hodnotené architektúry neurónových sietí v presnosti klasifikácie nad dátovou sadou ImageNet. ImageNet obsahuje viac ako 14 miliónov obrázkov rozdelených do viac ako 20 000 kategórií[6]. Ako metrika pre určovanie presnosti, alebo aj chybovosti, sa využíva Top-1 a Top-5 presnosť. Top-1 udáva presnosť, kedy sú predikovaná trieda a skutočná trieda zhodné. Top-5 udáva presnosť, pri ktorej sa skutočná trieda nachádza medzi piatimi najpravdepodobnejšími triedami, ktoré boli predpovedané neurónovou sieťou[17].

AlexNet

V roku 2012 bola vyvinutá architektúra AlexNet. Táto architektúra obsahuje 5 konvolučných a 3 plne prepojené vrstvy s aktivačnou funkciou ReLU. Výhodou AlexNetu je možnosť trénovať ju na viacerých GPU, čo umožňuje trénovať väčší model a skracuje čas učenia. V súťaži ILSVRC sa objavila dvakrát, a to v rokoch 2010 a 2012. V roku 2010 dosiahla Top-5 chybovosť 17%, o dva roky neskôr dosiahla chybovosť len 15.3%. Pre porovnanie, v roku 2012 bola druhá najlepšia Top-5 chybovosť dosiahnutá v rámci súťaže až 26.2%[20].

VGG

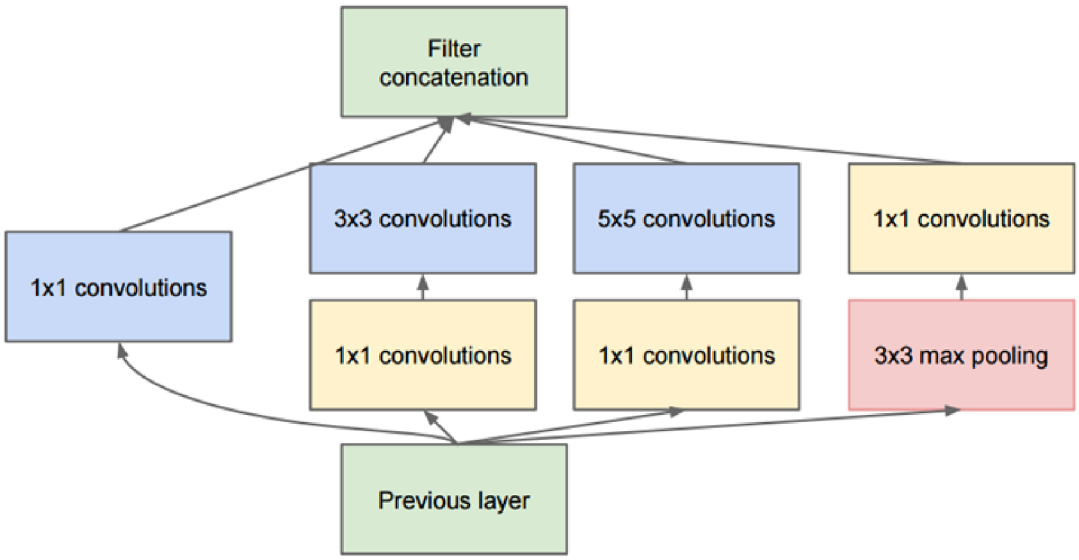
Architektúra VGG bola predstavená v roku 2014 v rámci súťaže ILSVRC. Aj keď táto architektúra bola založená na architektúre AlexNet, jej najväčšou odlišnosťou je veľkosť použitých filtrov v konvolučných vrstvách (AlexNet používa filtre o veľkosti 11 x 11, VGG len 3 x 3)[47]. Architektúra VGG ukázala, že konvolučné neurónové siete nemusia byť zložité ale mali by byť viac hlboké[10]. Existujú dve verzie architektúry VGG - VGG16 (viď obrázok 3.3) a VGG19. Číslovka v názve uvádza počet vnútorných vrstiev siete (súčet konvolučných a plne prepojených vrstiev). V roku 2014 sa architektúra VGG umiestnila na druhom mieste súťaže ILSVRC s Top-5 chybovosťou na úrovni 7.3%. Neskôr sa túto chybovosť podarilo znížiť na 6,8%[34].



Obr. 3.3: Ukážka architektúry VGG16 (prebrané z [3]).

GoogLeNet

Architektúra GoogLeNet bola taktiež predstavená v roku 2014 na súťaži ILSVRC, kde sa s Top-5 chybovosťou 6,67% umiestnila na prvom mieste[39]. Architektúra využíva tzv. Inception moduly, ktoré umožňujú sieti paralelne vykonávať viacero konvolučných operácií s rôznymi veľkosťami filtrov (viď obrázok 3.4). Od dovtedajších architektúr sa odlišovala tým, že veľkosť konvolúcie pre každú vrstvu nebola pevne daná. Neskôr boli vydané ďalšie verzie - Inception-v2, Inception-v3[40], Inception-v4 a Inception-ResNet[38] - ktoré sa snažili o zvýšenie presnosti a zníženie výpočtovej zložitosti.



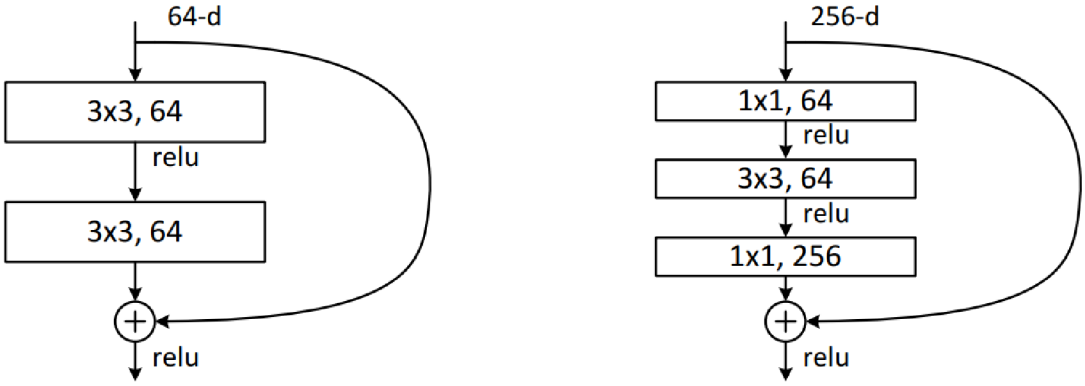
Obr. 3.4: Ukážka modulu Inception. Tieto moduly využívajú konvolúciu o veľkosti 1 x 1 na zníženie počtu operácií. Konvolučná vrstva 1 x 1 sa pridáva na zmenšenie vstupov pred konvolučné vrstvy o veľkosti 3 x 3 a 5 x 5. Zároveň sa táto vrstva pridáva aj za vrstvu max pooling. (prebrané z [3])

ResNet

Najčastejšou úlohou neurónovej siete je mapovanie vstupu na výstup pomocou niekoľko za sebou postavených vrstiev. Teoreticky by sa malo dať toto mapovanie škálovať a to pridaním ďalších vrstiev, čím by sa umožnila aproximácia stále komplikovanejších mapovaní. Z toho v konečnom dôsledku vyplýva, že čím zložitejší problém, tým hlbšia sieť. Avšak v praxi sa tréning hlbších sietí ukázal ako veľmi problematický kvôli tzv. explodujúcim gradientom a degradačnému problému[14].

Architektúra ResNet[14] rieši tieto problémy zavedením nového typu základného bloku nazvaného reziduálny blok, ktorý je znázornený na obrázku 3.5.Vďaka nemu je možné úspešne trénovať siete pozostávajúce z veľkého množstva vrstiev. Autormi článku [14] bolo dokázané, že implementácia tzv. reziduálneho mapovania $F(x) = H(x) - x$ je jednoduchšia ako hľadanie aproximácie priameho mapovania $H(x)$. Z toho vyplývajúca funkcia $F(x) + x$ tiež dokáže aproximovať hľadané mapovanie vstupu na výstup, avšak učenie je výrazne jednoduchšie. V doprednných neurónových sieťach je možné potom túto formuláciu funkcie implementovať pomocou tzv. skratiek (anglicky *shortcut*), vďaka ktorým môže byť jedna alebo viac vrstiev preskočených. Výstup týchto skratiek je následne pričítaný k výstupu preskočených vrstiev. Okrem klasického reziduálneho bloku sa v rámci architektúry ResNet používa aj tzv. *Bottleneck Residual Block*. Tento blok sa používa na zmenšenie počtu parametrov a zvýšenie rýchlosti učenia siete. Skladá sa z troch konvolučných vrstiev (viď obrázok 3.5). Tento typ bloku sa používa v hlbších verziách architektúry ResNet ako napríklad ResNet50 alebo ResNet101. Číslovka na konci názvu oznčuje počet vnútorných vrstiev. Napríklad ResNet18 obsahuje 18 vrstiev, z toho 16 konvolučných vrstiev a 2 plne prepojené vrstvy, zatiaľ čo najhlbšia verzia má až 1202 vrstiev. Aj napriek extrémnej hĺbke táto verzia nepreukázala takmer žiadne optimalizačné ťažkosti[14].

ResNet18 obsahuje 18 vrstiev, z toho 16 konvolučných vrstiev a 2 plne prepojené vrstvy, zatiaľ čo ResNet50 obsahuje dokopy 50 vrstiev, z toho 48 konvolučných vrstiev a 2 plne prepojené vrstvy.



Obr. 3.5: Ukážka reziduálneho bloku spolu so skratkou (vľavo) a reziduálneho bottleneck bloku (vpravo).(prebrané z [14])

3.3 Možnosti extrakcie príznakov

Extrakciu príznakov možno využiť mnoho spôsobmi, napríklad pri identifikácii objektov v obraze[46], identifikácii ľudí pomocou tváre[33] alebo v oblastiach spracovania hovoreného slova[44]. Za týmto účelom je sieť možné natrénovať viacerými spôsobmi. Ako prvá a asi najpriamejšia možnosť je trénovať sieť ako klasifikátor. Touto technikou sa zaoberá mnoho prác ktoré ukázali efektívnosť tohto prístupu[37, 41]. Hlavnou nevýhodou je potreba veľkého množstva označených dát na trénovanie.

Ako druhá možnosť je využitie tzv. kontrastívneho učenia (anglicky *contrastive learning*). Kontrastívne učenie je typ učenia bez učiteľa, ktorý sa používa na učenie reprezentácií dát. Jeho cieľom je naučiť sa vektorovú reprezentáciu dát tak, aby podobné dáta mali blízke vektorové reprezentácie v priestore a odlišné dáta vzdialené v priestore[48]. V kontrastívnom učení sa využíva pár dát, ktorý predstavuje "podobné" alebo "odlišné" vstupy, pomocou ktorých sa učí neurónová sieť tak, aby pre podobné vstupy produkovala blízke hodnoty a pre odlišné vstupy vzdialené hodnoty. Jedným z príkladov kontrastívneho učenia je kontrastívne učenie na obrázkoch. Tu sa každý pár vstupov skladá z dvoch obrázkov, ktoré sú si buď podobné (hovoríme aj že pochádzajú z rovnakej triedy) alebo rozdielne (pochádzajú z rozličných tried). Sieť sa trénuje tak, aby pre podobné obrázky generovala podobné vektory a naopak. V porovnaní s tradičným učením s učiteľom, kontrastívne učenie nevyžaduje označené dáta[43].

Klasifikátor

Využitie klasifikátora na extrakciu príznakov je relatívne jednoduchá záležitosť. Najprv je sieť natrénovaná na klasifikáciu do predom určených tried. Po natrénovaní je následne odstránená posledná plne prepojená vrstva, ktorá v sieti zabezpečovala finálne priradenie pravdepodobností pre všetky triedy. Tým je docielené, že výsledná štruktúra je vektor príznakov pre daný vstupný obrázok[51].

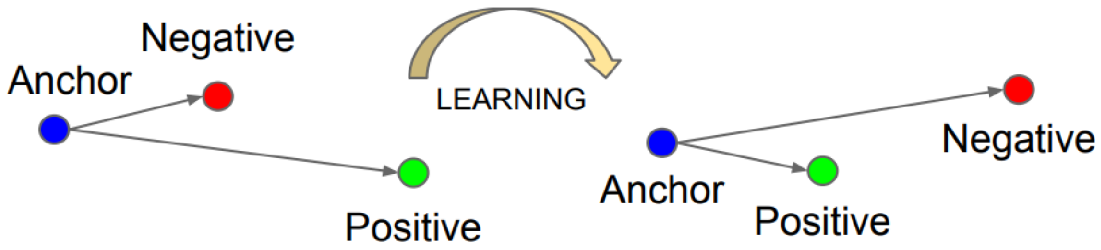
Pre trénovanie klasifikátoru sa najčastejšie používa chybová funkcia *Cross entropy loss*. Cross entropy loss je matematicky definovaná ako:

$$L_{CE} = -\sum_{i=1}^{n} t_i log(p_i) \tag{3.1}$$

pre n tried, kde ti predstavuje očakávanú hodnotu a pi je pravdepodobnosť funkcie softmax pre triedu i[19].

Kontrastívne učenie

Kontrastívne učenie je technika ktorá sa zaoberá trénovaním neurónovej siete na získanie všeobecných príznakov z dát tým, že sa sieť učí, ktoré dáta sú si podobné a ktoré sú odlišné[48]. Kontrastívne učenie bolo pôvodne zamýšľané ako technika pre učenie bez učiteľa, ktorá využíva na učenie rôzne augmentácie vstupných obrázkov, ako napríklad otočenie, orezanie alebo zavedenie šumu. Tieto augmentované obrázky sú následne považované za podobné, zatiaľ čo všetky ostatné pôvodné obrázky sú považované za rozdielne[50]. Existuje aj možnosť použiť namiesto párov rovno trojicu vstupných obrázkov, z ktorých sú si dva podobné a jeden je odlišný. Tréningový algoritmus sa snaží minimalizovať vzdialenosť medzi podobnými bodmi a zároveň maximalizovať vzdialenosť medzi rozdielnymi[33] (viď obrázok 3.6). Ukázalo sa, že kontrastívne učenie sa dá použiť aj pri označených dátach, kedy sa ako podobné obrázky považujú všetky obrázky z jednej triedy a ako rozdielne všetky obrázky z iných tried[18].



Obr. 3.6: Ukážka učiaceho procesu kontrastívneho učenia pri použití troch vstupov. Vzdialenosť medzi podobnými obrázkami (Anchor a Positive) sa zmenšuje, zatiaľ čo vzdialenosť medzi rozdielnymi obrázkami sa zväčšuje (Anchor a Negative).(prebrané z [33])

Siamske siete

Siamske siete využívajú metódu kontrastívneho učenia. Sú špeciálnym typom hlbokých neurónových sietí, ktoré majú za úlohu vyhodnotiť podobnosť medzi dvoma vstupmi. Tieto siete sa skladajú z dvoch identických podsietí ktoré spolu zdieľajú váhy[45](viď obrázok 3.7). Celková architektúra prijíma dva rôzne vstupy, a pre každý zvlášť generuje výstup - vektor príznakov - ktorý popisuje vlastnosti daného vstupu. Následne je možné takto vygenerované vektory príznakov spolu porovnať a určiť podobnosť medzi vstupmi. Pokiaľ sú si vstupy podobné, sú si aj ich vektory príznakov podobné a naopak[43]. Za účelom porovnávania výstupov sa pri kontrastívnom učení často využíva výpočet euklidovskej vzdialenosti alebo kosínusovej podobnosti.

Na samotné trénovanie siamskej siete sa používa chybová funkcia *Contrastive loss*[18].

Contrastive Loss je chybová funkcia, ktorá sa používa na trénovanie siamských sietí, ktoré majú za úlohu naučiť sa extrahovať a porovnávať vzájomnú podobnosť vstupov. Konkrétne sa táto funkcia počíta pre každú dvojicu vstupov v dátovej sade. Ak sú vstupy označené ako "rovnaké" (tj. patria k rovnakej triede), chybová funkcia sa snaží minimalizovať vzdialenosť medzi ich vektormi príznakov. Naopak, ak sú vstupy označené ako "rozdielne" (tj. patria k rôznym triedam), chybová funkcia sa snaží maximalizovať vzdialenosť medzi ich vektormi príznakov.

Nech $\vec{X_1}, \vec{X_2}$ je pár vstupných vektorov a Y je binárne označenie pre tento pár. Y = 0 práve keď sú vektory označené ako podobné, Y = 1 práve keď sú vektory označené ako rozdielne. DW nech označuje Euklidovskú vzdialenosť a GW neurónovú sieť. Potom platí:

$$D_W(\vec{X_1}, \vec{X_2}) = ||G_W(\vec{X_1}) - G_W(\vec{X_2})||_2 \tag{3.2}$$

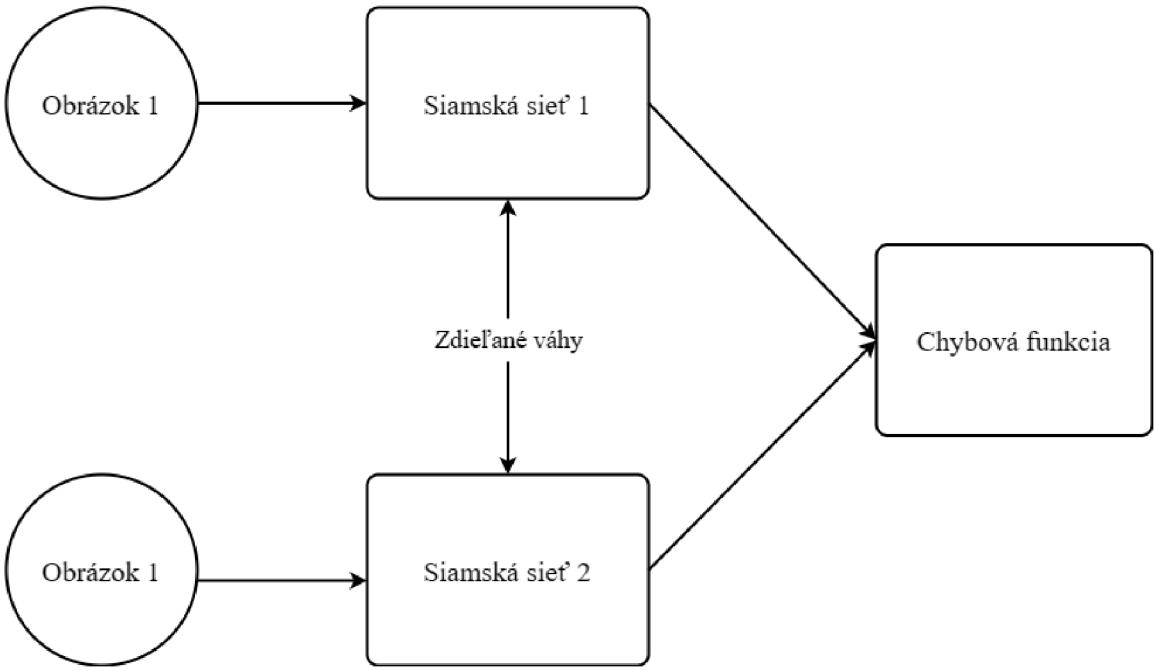
Skrátená notácia DW($\vec{X_1}, \vec{X_2}$) nech je DW. Potom sa dá contrastive loss vyjadriť ako:

$$\mathcal{L}(W) = \sum_{i=1}^{P} L(W, (Y, \vec{X_1}, \vec{X_2})^i) \tag{3.3}$$

pre

$$L(W, (Y, \vec{X_1}, \vec{X_2})^i) = (1 - Y)L_S(D_W^i) + YL_D(D_W^i) \tag{3.4}$$

kde $(Y, \vec{X_1}, \vec{X_2})^i$ je i-ty označený pár, L_S je čiastočná chybová funkcia pre podobné páry, L_D je čiastočná chybová funkcia pre rozdielne páry a P je počet trénovacích párov[18].



Obr. 3.7: Ilustrácia architektúry siamskej siete. Tá je založená na princípe dvoch identických sietí, ktoré navzájom zdieľajú váhy. Na vstupe majú rozdielne obrázky, ich výstupy sú následne vstupom do chybovej funkcie.

Tripletové siete

Tripletové siete priamo nadväzujú na siamske siete a využívajú metódu kontrastívneho učenia. Rozširujú počet vnútorných sietí na tri, a taktiež všetky podsiete zdieľajú spoločné váhy[15](viď obrázok 3.8). Výhodou tripletových sietí je fakt, že sme schopný porovnávať výsledné vektory hneď troch obrázkov. Toho je využité v rámci učenia, a to tak, že na vstup siete sa pošlú tri vstupy- *x,* x^+, x^-, kde obrázok x predstavuje kotvu (anglicky *anchor*), x^+ reprezentuje obrázok patriaci do rovnakej triedy ako x (v literatúre sa zvyčajne označuje ako *positive*) a x^- reprezentuje obrázok patriaci do rozdielnej triedy ako x (v literatúre sa zvyčajne označuje ako *negative*). Architektúra siete dovoľuje, aby bol problém prezentovaný ako klasifikátor dvoch tried, kde cieľom siete je správne klasifikovať ktorý vstup (x^+, x^-) je z rovnakej triedy ako vstup x[15]. Ak ostaneme pri pomenovaní vstupov *x,* $x^+ a x^-$, a reprezentáciu jednej podsiete pomenujeme *Net(X)*, môžeme tripletovú sieť *TripletNet(x,*x^-, x^+*)* zapísať ako:

$$TripletNet(x, x^-, x^+) = \begin{bmatrix} ||Net(x) - Net(x^-)||_2 \\ ||Net(x) - Net(x^+)||_2 \end{bmatrix} \in \mathbb{R}^2_+ \tag{3.5}$$

V prípade obecného zadania, kde cieľom je extrahovať vektor príznakov, nám trieda určuje, ktorý zo vstupov je "bližšie" ku vstupu x.

Trénovanie prebieha pomocou chybovej funkcie *Triplet loss.*

Triplet loss dostáva na vstup trojicu vektorov x, x^+, x^- popísanú vyššie. Cieľom tejto chybovej funkcie je zaistiť, že vzdialenosť vektorov x (anchor) a x^+ (positive) je väčšia ako vzdialenosť vektorov x a x^- (negative). Matematicky sa dá tento vzťah vyjadriť ako

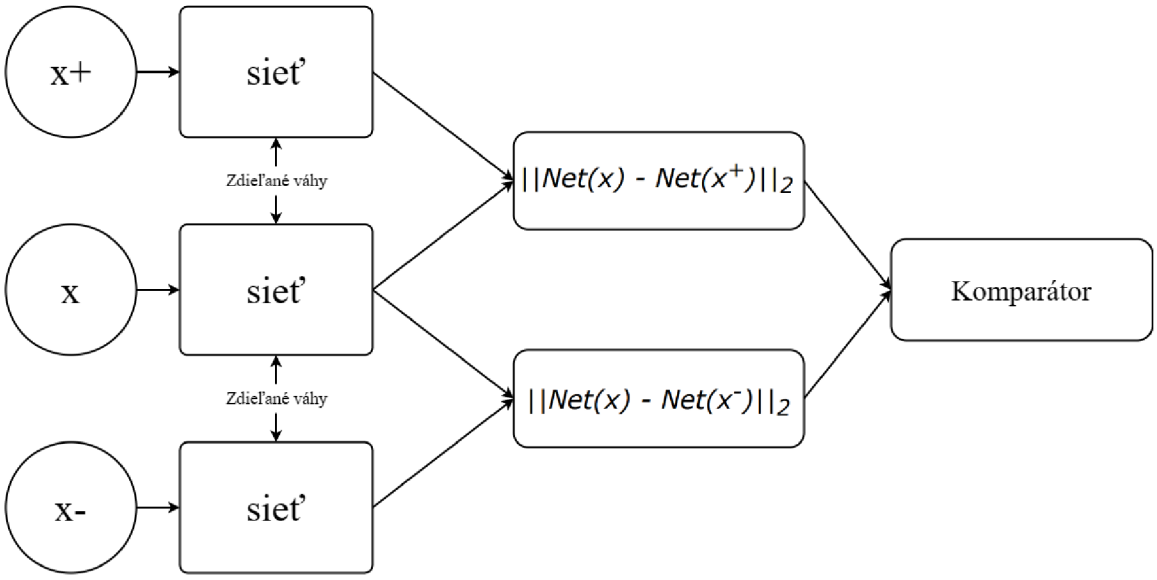
$$\mathcal{F}(x_i, x_i^+, x_i^-) = ||x_i - x_i^+||_2^2 + \alpha < ||x_i - x_i^-||_2^2 \forall (x_i, x_i^+, x_i^-) \in \mathcal{T} \tag{3.6}$$

kde α predstavuje minimálnu medzeru, ktorá je vyžadovaná medzi pozitívnymi a negatívnymi pármi a $\mathcal{T}$ je množina všetkých možných tripletov v trénovacej sade s kardinalitou N. Matematický zápis funkcie triplet loss potom je[15]:

$$\sum_i^N \left[||f(x_i) - f(x_i^+)||_2^2 - ||f(x_i) - f(x_i^-)||_2^2 + \alpha \right]_+ \tag{3.7}$$

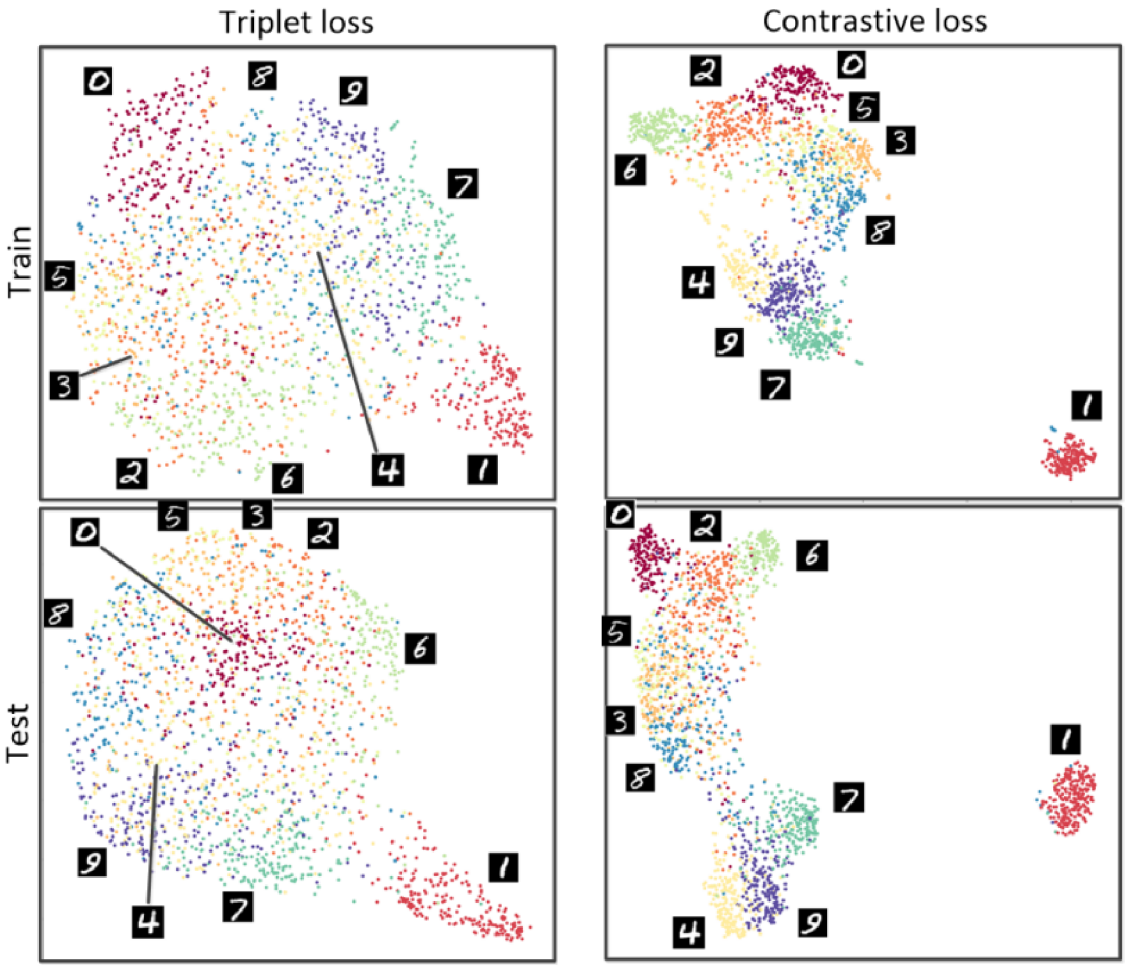
V prípade ak vyjadríme $||f(x_i) - f(x_i^+)||_2^2$ ako $D_{a,p}$ a $||f(x_i) - f(x_i^-)||_2^2$ ako $D_{a,n}$, môžeme predošlý zápis zjednodušiť na:

$$\mathcal{L}_{tri}(\theta) = \sum_{\substack{a,p,n \\ y_a = y_p \neq y_n}} [\alpha + D_{a,p} - D_{a,n}]_+ \tag{3.8}$$



Obr. 3.8: Ilustrácia architektúry tripletovej siete. Tá sa zkladá z troch identických podsietí so zdieľanými váhami. Vstupom do siete sú tri obrázky, cieľom je zistiť, ktoré dva zdieľajú rovnakú triedu a ktorý patrí do inej triedy.

Na obrázku 3.9 je zobrazený rozdiel medzi vektorovými priestormi pre Contrastive loss a Triplet loss. Pre každú jedinečnú úlohu je vhodná iná chybová funkcia. Preto časom začalo vznikať veľa nových chybových funkcií, ktoré vznikli pre potreby jednotlivých úloh. Napríklad *ArcFace loss* je chybová funkcia vyvinutá za účelom trénovania neurónových sietí pre rozpoznávanie ľudí podľa tváre[7], *Lifted structure loss* zase využíva pre učenie neurónovej siete pozitívne a negatívne vzorky z celého batchu[36].



Obr. 3.9: Ukážka priestorov vektorových reprezentácií pre Triplet loss a Contrastive loss na dátovej sade MNIST[8]. Contrastive loss pri učení v jednom kroku buď zväčšuje alebo zmenšuje vzdialenosti medzi vektormi, zatiaľ čo triplet loss v jednom kroku zároveň zväčšuje vzdialenosť s rozdielnym vektorom a zmenšuje vzdialenosť s podobným vektorom. To má za následok, že contrastive loss má všetky skupiny vektorov relatívne blízko seba, zatiaľ čo pri triplet loss sú skupiny vektorov viac rozprestreté v priestore.(prevzaté z [13])

3.4 Metódy vyhodnocovania presnosti klasifikátora

Jednou z najdôležitejších úloh pri vytváraní modelu z oblasti strojového učenia je správne vyhodnotenie výsledkov[1]. Vyhodnocovanie modelov je kritické pre zabezpečenie toho, aby neurónové siete fungovali správne a riešili konkrétne problémy. Aby sme zistili presnosť modelu, je nutné ho podrobiť testom. Na testovanie je využitá testovacia dátová sada. Výsledné predikcie nad touto dátovou sadou sú následne zobrazené prostredníctvom numerickej hodnoty alebo pomocou grafu. Presnosť výsledného hodnotenia závisí aj na použitej testovacej dátovej sade. Konkrétne veľkosť testovacej sady môže mať vplyv na výsledky modelu. V prípade že je testovacia sada príliš malá, môže to viesť k nedostatočne presnému odhadu kvality modelu, pretože dátová sada nebude obsahovať všetky možné situácie, s ktorými sa môže neurónová sieť stretnúť pri bežnom používaní. V takom prípade môže byť výsledná úspešnosť modelu omnoho horšia ako predpovedali výsledky nad testovacou dátovou sadou.

Základná terminológia na určovanie úspešnosti pri vyhodnocovaní kvality:

- *True positive* (skratka TP) znamená správne pozitívne rozhodnutie, tj. model predpovedal pozitívnu triedu a táto odpoveď bola správna.
- *True negative* (skratka TN) znamená správne negatívne rozhodnutie, tj. model predpovedal negatívnu triedu a táto predpoveď bola správna.
- *False positive* (skratka FP) znamená nesprávne pozitívne rozhodnutie, tj. model predpovedal pozitívnu triedu, ale táto predpoveď nebola správna.
- *False negative* (skratka FN) znamená nesprávne negatívne rozhodnutie, tj. model predpovedal negatívnu triedu, ale táto predpoveď nebola správna.
- *True positive rate* (skratka TPR) známy aj ako citlivosť, je pomer počtu správnych pozitívnych rozhodnutí k celkovému počtu pozitívnych príkladov v testovacej množine. Táto metrika ukazuje, ako dobre model identifikuje pozitívne príklady. TPR = TP / (TP + FN)
- *False positive rate* (skratka FPR) je pomer počtu nesprávnych pozitívnych rozhodnutí k celkovému počtu negatívnych príkladov v testovacej množine. Táto metrika ukazuje, ako často model identifikuje negatívne príklady ako pozitívne. FPR = FP / (FP + TN)
- *False negative rate* (skratka FNR) je pomer počtu nesprávne klasifikovaných pozitívnych tried ku všetkým pozitívnym triedam v testovacej sade. Táto metrika ukazuje, aký podiel pozitívnych tried tried model nezachytil a chybne klasifikoval ako negatívne.

Precision

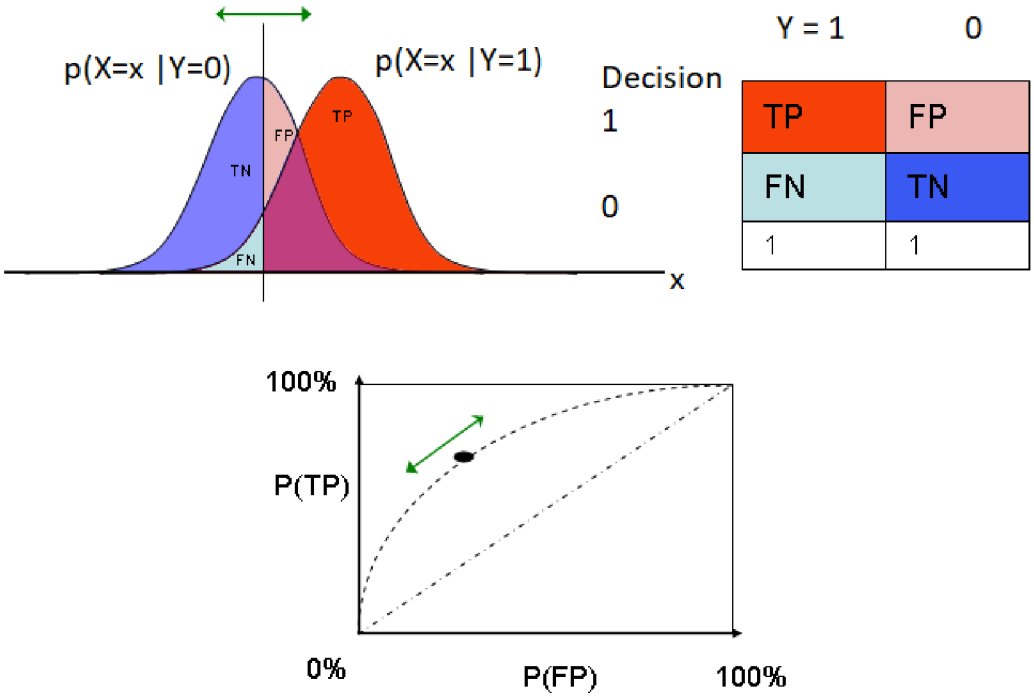
Presnosť pozitívnej triedy (anglicky *Precision*) udáva pomer správne klasifikovaných pozitívnych príkladov ku všetkým klasifikovaným pozitívnym príkladom.

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

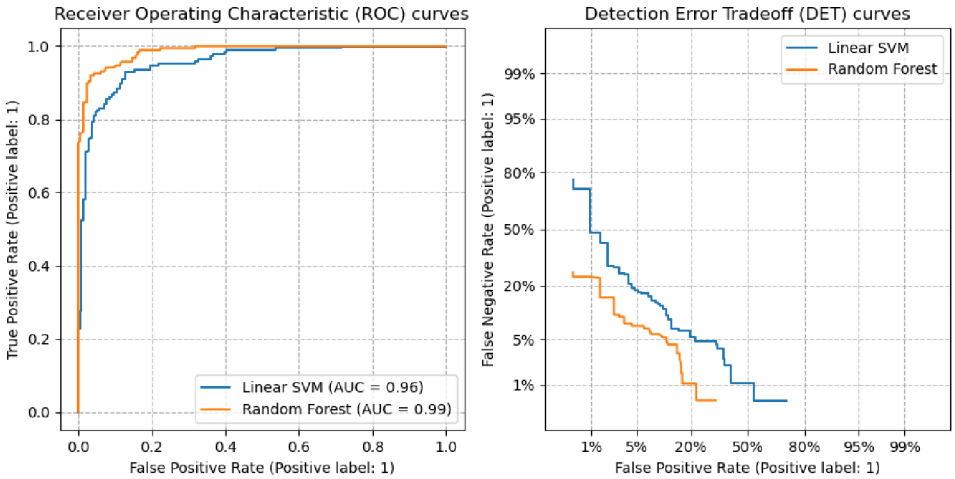
ROC a DET krivky

ROC krivka (skratka pre *Receiver Operating Characteristic*)[11] je nástroj na hodnotenie a optimalizáciu binárneho klasifikačného systému ktorý ukazuje závislosť medzi TPR a FPR. ROC krivka sa vytvára tým, že sa mení hodnota prahu klasifikátora a vypočítavajú sa hodnoty TPR a FPR pre každý prah (viď obrázok 3.10). Ideálne klasifikačné modely majú ROC krivky, ktoré sú čo najviac blízko ľavému hornému rohu grafu, kde je TPR vysoký a FPR nízky. Spolu s ROC krivkou sa zvykne využívať aj tzv. AUC skóre (anglická skratka pre *Area Under the Curve*). To sa vypočítava ako plocha pod ROC krivkou. Vysoké AUC indikuje ,že klasifikátor má vysokú citlivosť a špecifickosť.

DET krivka (skratka pre *Detection Error Tradeoff*)[23] je iný spôsob vizualizácie výkonu klasifikačného modelu, ktorý sa zameriava na hodnotenie detekcie chýb. Ako je ukázané na obrázku 3.11, na rozdiel od ROC krivky, ktorá má lineárne škály pre osi TPR a FPR, DET krivka používa škály normálnej odchýlky pre osi FPR a FNR. Vďaka tomu je na výslednom grafe lepšie viditeľné miesto s nízkym FPR a FNR.



Obr. 3.10: Ukážka ROC krivky a závislosti výsledkov na voľbe prahu. Tabuľka v pravej hornej časti znázorňuje, že súčet TP a FN je rovný počtu všetkých pozitívnych príkladov a súčet FP a TN je rovný počtu všetkých negatívnych príkladov. Graf v ľavej hornej časti znázorňuje podiel TN, FP, TP a FN pri posune prahu. V spodnej časti obrázku je príslušná ROC krivka. (prebrané z [24])



Obr. 3.11: ukážka ROC krivky (vľavo) a DET krivky (vpravo) pre zhodné dáta [29]

Kapitola 4

Návrh a implementácia systému

V tejto kapitole je podrobne opísaná každá časť mnou navrhnutého systému a implementačné podrobnosti. Mnou navrhnutý systém sa skladá z troch hlavných častí:

1. príprava vstupných dát do neurónovej siete
2. využitie neurónovej siete na tvorbu vektorov príznakov
3. porovnávanie a vyhodnocovanie výsledkov

4.1 Dátové sady a príprava vstupných dát

Pre svoje zadanie som sa rozhodol použiť dve verejne dostupné dátové sady namiesto tvorby vlastnej. Hlavným dôvodom bolo to, že získavanie vlastných dát je veľmi časovo aj technologicky náročné. Pre dosiahnutie kvalitných výsledkov je potrebné mať ideálne podmienky alebo použiť veľmi nákladnú a ťažko dostupnú techniku. Zároveň by som musel nájsť dostatočný počet dobrovoľníkov, ktorí by mi umožnili vytvoriť niekoľko fotografií ich očí. To by bolo časovo, finančne a logisticky veľmi náročné. Našťastie existuje niekoľko verejne dostupných dátových sád, ktoré boli vytvorené pre účely rozpoznávania na základe očí. Tieto sady poskytujú dostatočný počet obrázkov očí, ktoré som mohol použiť na trénovanie a testovanie mojej neurónovej siete.

Výhod použitia verejne dostupných dátových sád je hneď niekoľko. Keďže ich vytvorenie bolo založené na vedeckom výskume, môžem veriť ich kvalite a správnosti. To mi umožnilo využiť tieto dáta s dôverou, že sú dostatočné a spoľahlivé pre moje potreby. Ďalej sa naskytuje možnosť porovnať výsledky mojej práce s výsledkami iných prác, ktoré pracovali s rovnakými dátami, poprípade možnosť v budúcnosti porovnať výsledky s mojou prácou.

Ako jedna z možných dátových sád sa naskytovala aj sada *The Hong Kong Polytechnic University Cross-Spectral Iris Images Database*[42]. Tá obsahuje 12 540 obrázkov od 209 osôb, z ktorých polovica je čiernobielych, a teda vhodných pre moju prácu. Avšak pri spracovávaní dátovej sady ,opísanom nižšie, mal systém OSIRIS veľmi časté problémy pri segmentácii dúhovky. Z tohto dôvodu som sa rozhodol dátovú sadu nakoniec nepoužiť.

Databáza IIT Delhi Iris

Prvou použitou dátovou sadou bola IITD Iris [16]. Táto dátová sada obsahuje snímky očí od 224 osôb. Jedná sa o študentov a zamestnancov na IIT Delhi, v New Delhi v Indii. Všetci

jedinci v databáze sú vo vekovej skupine od 14 do 55 rokov a zahŕňajú 176 mužov a 48 žien. Databáza obsahuje dokopy 1120 obrázkov - tj. 5 obrázkov od jednej osoby. Rozlíšenie týchto obrázkov je 320 x 240 pixelov a všetky obrázky boli vytvorené v interiéri.

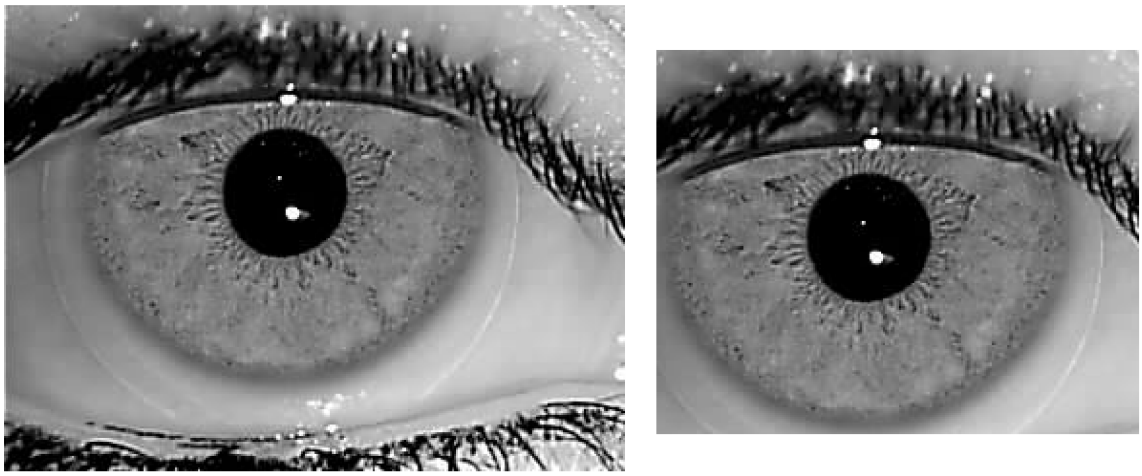
Databáza CASIA-IrisV4

Druhou použitou dátovou sadou bola CASIA-IrisV4[4]. Konkrétne časť dátovej sady s názvom CASIA-Iris-Interval. Táto dátová sada obsahuje dokopy 2 639 obrázkov od 249 osôb. Rozlíšenie obrázkov je 320 x 280 pixelov a všetky boli zachytené v interiéri.

Úprava dát pomocou nástroja OSIRIS

Obe dátové sady obsahujú obrázky očí fotené zo vzdialenosti pár centimetrov, ktoré v sebe zahŕňajú aj zbytočné okolie oka. Pre minimalizáciu tejto zbytočnej informácie som zvolil dva postupy.

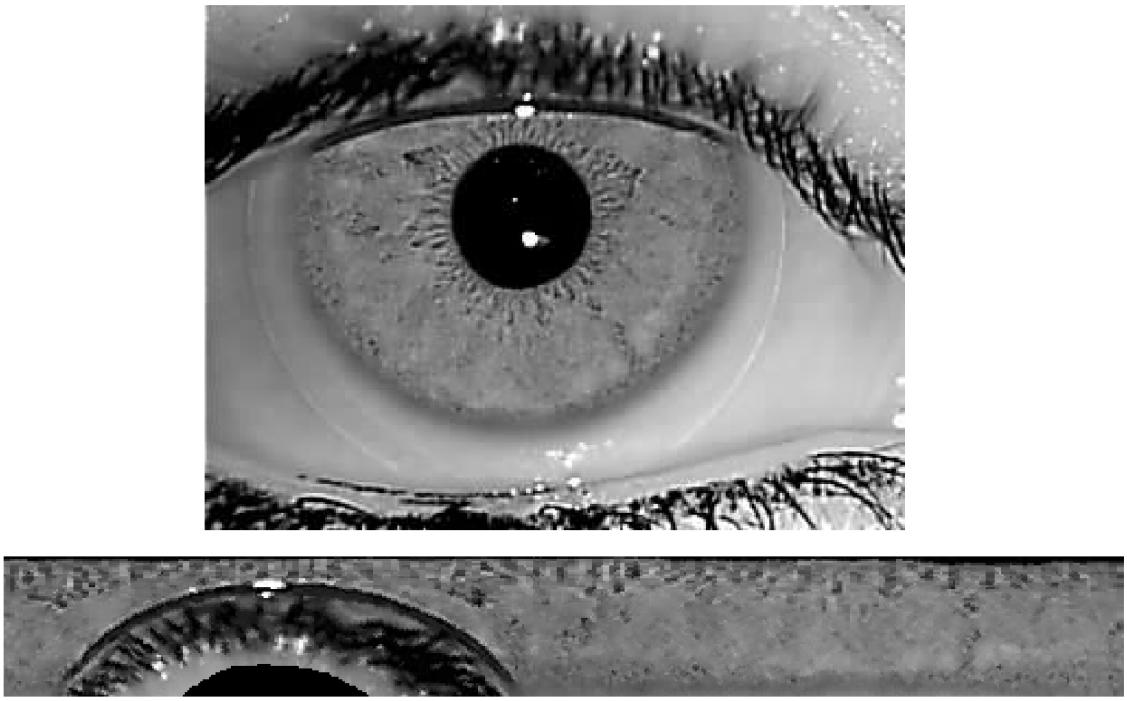
Ako prvé som vyskúšal orezať obrázok do obdĺžnikového tvaru o veľkosti samotnej dúhovky (obrázok 4.1). Tým sa mi podarilo eliminovať časť zbytočnej informácie obrázku, s ktoru neurónová sieť nemala pracovať. Koordinácie pre orezanie, v tvare x1, y1, x2, y2, som získal pomocou systému OSIRIS, ktorý pre každý obrázok pri segmentácii dúhovky vytvorí textový dokument obsahujúci body, pomocou ktorých je následne prevedená samotná extrakcia dúhovky. Následne som pomocou mnou napísaného skriptu prešiel celým textovým dokumentom a našiel najväčšie a najmenšie hodnoty x a y, podľa ktorých som následne samotný obrázok orezal. Týmto spôsobom som upravil len dátovú sadu IITD Iris, ktorá je z dvoch použitých sád jednoduchšia, a preto mala potenciál na lepšie výsledky.



Obr. 4.1: Pôvodný obrázok z dátovej sady (vľavo), mnou orezaný obrázok (vpravo).

Ako druhý postup som zvolil využitie celého systému OSIRIS, vďaka ktorému som z každého obrázka z dátovej sady získal normalizovanú podobu dúhovky (obrázok 4.2). Proces normalizácie v systéme OSIRIS je popísaný v časti 2.3 Týmto spôsobom som spracoval obe dátové sady.

V konečnom dôsledku som mal k dispozícii až tri dátové sady: IITD_cropped - mnou orezaná dátová sada IITD -, IITD_normalized - dátová sada extrahovaná pomocou systému OSIRIS z dátovej sady IITD - a CASIA_IrisV4_normalized - dátová sada extrahovaná



Obr. 4.2: Pôvodný obrázok z dátovej sady (hore), normalizovaný obrázok dúhovky pomocou systému OSIRIS (dole). Obrázok ďalej ukazuje, že ani systém OSIRIS nieje vždy stopercentný, keďže súčasťou normalizovaného obrázku dúhovky je aj časť zreničky a mihalníc, ktoré sa systému nepodarilo správne detekovať.

pomocou systému OSIRIS z dátovej sady CASIA-IrisV4-Interval. Všetky tieto dátové sady boli neskôr použité pri experimentoch.

Rozdelenie vstupných dát

Pre svoju prácu som mal k dispozícii tri dátové sady opísané vyššie. Zároveň som mal ku každej dátovej sade súbor obsahujúci metadáta k danej sade (tabuľka 4.1). V rámci každej jednej dátovej sady boli obrázky rozdelené na obrázok z pravého a obrázok z ľavého oka. Toto rozdelenie som využil a rozdelil každú dátovú sadu na dve časti - trénovaciu a testovaciu. Trénovacia časť obsahovala všetky obrázky z pravého oka, testovacia zase všetky obrázky z ľavého oka. Týmto rozdelením som docielil, že pri testovaní neurónovej siete jej na vstup prídu neznáme dáta. Trénovacia časť bola následne ešte rozdelená v pomere 0.8 : 0.2 na časť skutočne použitú na trénovanie, a časť slúžiacu na validáciu pri trénovaní.

Keďže štruktúry z knižnice PyTorch ktoré sa bežne používajú na vytvorenie dátovej sady (konkrétne trieda *Dataset*) neboli pre moje potreby dostatočné, musel som si na načítanie dát vytvoriť vlastnú triedu. Táto trieda (nazvaná *CustomDataset*) vytvára dátovú sadu pomocou troch parametrov. Prvý z nich je csv súbor obsahujúci metadáta o dátovej sade, druhý je priečinok obsahujúci samotné obrázky a tretí parameter udáva či ide o testovaciu alebo trénovaciu sadu.

Tabuľka 4.1: Ukážka dokumentu obsahujúceho metadáta. V prvom riadku je popísané, čo predstavujú hodnoty nachádzajúce sa na jednom riadku, v druhom riadku je ukážka týchto hodnôt. Pre mňa podstatná hodnota je v ôsmom stĺpci (stĺpec H), v ktorom je strana tváre, z ktorej obrázok pochádza. To mi rozdelilo dátovú sadu na trénovaciu a testovaciu. Druhá pre mňa podstatná hodnota je v piatom stĺpci (stĺpec E). Táto hodnota predstavuje pomyselnú triedu obrázka využitú pri trénovaní a testovaní.

id	dataset	relative_path	image_id
0	casia_iris_v4_interval	/CASIA-Iris-Interval/001/L/S1001L01.jpg	01

subject_id	unique_image_id	unique_subject_id	side
001	casia_iris_v4_interval_001_01_L	casia_iris_v4_interval_001	L

4.2 Návrh riešenia

Výber vhodnej neurónovej siete

Architektúr umelých neurónových sietí poznáme mnoho. Každá bola vytvorená za iným účelom, z čoho vyplýva že každá bude dobrá na niečo iné. Vhodných kandidátov na klasifikáciu obrázkov je hneď niekoľko:

- AlexNet: Táto architektúra bola navrhnutá v roku 2012 pre súťaž ImageNet Large Scale Visual Recognition Challenge, kde zaznamenala lepšie výsledky v oblasti klasifikácie ako predošlé metódy.
- VGGNet: Táto architektúra bola vyvinutá v roku 2014 a skladá sa z hlbokého zásobníka vrstiev. VGGNet sa vyznačuje veľmi malými konvolučnými jadrami, ktoré sú kombinované do hlbokých zásobníkov.
- GoogleNet : Táto architektúra bola vyvinutá v roku 2014 a vyznačuje sa použitím tzv. Inception modulov, ktoré zlepšujú výkonnosť a znižujú počet parametrov siete.
- ResNet: Architektúra vyvinutá v roku 2015 vyznačujúca sa použitím reziduánlych blokov.

Autori článku [25] popisujú vlastný návrh hlbokej neurónovej siete s názvom IRISNet. Táto sieť bola špecificky navrhnutá za účelom klasifikácie osôb na základe obrazu dúhovky, a výsledky práce preukazovali že by mohlo ísť o silného kandidáta pre moju prácu. Avšak po naprogramovaní neurónovej siete podľa špecifikácie uvedenej v danom článku [25] som nebol schopný dosahovať ani zďaleka také výsledky ako boli uvádzané. Už pri skorom testovaní sa preukázalo že sieť nie je vhodná pre splnenie môjho zadania.

Nakoniec som sa rozhodol pre architektúru ResNet pre jej úspechy v klasifikačných úlohách. Konkrétne som počas svojich experimentov využil architektúry ResNet18 a ResNet50. ResNet50 má omnoho väčšiu kapacitu na učenie ako ResNet18 a môže potenciálne dosiahnuť lepšie výsledky, avšak pri veľkosti mojich dátových sád bolo veľké riziko rýchleho pretrénovania.

Spôsob porovnávania výsledkov

Moja úloha spočívala v porovnaní dvoch vektorov príznakov a následnom určení, či sa jedná o tú istú osobu alebo nie. Na porovnanie zhodnosti dvoch vektorov som použil euklidovskú

vzdialenosť. Euklidovská vzdialenosť určuje vzdialenosť dvoch bodov v n-rozmernom priestore. V prípade veľkej podobnosti vstupných vektorov sa výsledok euklidovskej vzdialenosti blíži k nule, v opačnom prípade sa výsledok blíži k jednotke.

Keďže sa moja úloha dala jednoducho premietnuť ako binárna klasifikácia, rozhodol som sa použiť na porovnávanie výsledkov AUC skóre a DET krivky. Pri vyhodnocovaní klasifikátorov s použitím DET kriviek môžeme sledovať výkon modelu v závislosti od rôznych prahov pre rozhodovanie o triede. Vďaka tomu som mohol vyhodnotiť, ktorý prístup trénovania neurónovej siete poskytuje lepšie výsledky a pri ktorých prahoch to dokazuje.

Ako východiskový systém pre porovnávanie som použil systém OSIRIS a jeho funkciu porovnávania, ktorá mi pre jednotlivé páry vstupných obrázkov vyprodukovala pravdepodobnosti, či sú zhodné alebo nie. Tieto pravdepodobnosti som použil na vytvorenie AUC skóre a DET kriviek pre daný dataset.

4.3 Pracovné prostredie a použité systémy

Ako implementačný jazyk pre moje zadanie som si vybral Python3, keďže sa jedná o jeden z najpoužívanejších jazykov v oblasti strojového učenia. Vývojové prostredie mi sprostredkovala služba Google Colab, ktorá poskytuje aj grafické karty, ktoré môžu byť použité na trénovanie neurónových sietí. Táto služba je bezplatná, avšak môže sa stať, že sú všetky dostupné GPU už využívané. Tomu sa dá zabrániť "rezerváciou", ktorá už je ale spoplatnená. Na obrázku 4.3 je znázorné, ktoré python knižnice som použil pri vypracovaní mojej práce.

Jedna z najpodstatnejších z nich je knižnica Pytorch Metrics Learning. Táto knižnica mi poskytla implementáciu chybových funkcií, ako aj triedu sampler. Sampler mi umožnoval počas tréningu meniť dĺžku jednotlivých trénovacích cyklov, ako aj nastavenie, koľko obrázkov z jednej triedy sa bude v batchi nachádzať. Chybová funkcia Triplet margin loss zase ponúka zmeniť množstvo, koľko trojíc sa vytvorí z trénovacích obrázkov v batchi.

Obr. 4.3: Knižnice ktoré som použil pri vypracovaní zadania. PyTorch a Torchvision mi poskytli modely neurónových sietí. NumPy a pandas som použil na spracovanie dát, Matplotlib na vizualizáciu dát a Scikit-learn na vyhodnocovanie. Pytorch metrics learning mi dodala chybové funkcie a sampler. Knižnicu Pickle som využil na serializáciu a deserializáciu dát.

Kapitola 5

Priebeh a výsledky experimentov

Hlavnou úlohou môjho zadania bolo natrénovať neurónovú sieť pre čo najlepšiu extrakciu príznakov. Túto úlohu som rozdelil do troch samostatných experimentov, kde som si vytýčil rôzne ciele.

5.1 Prvý experiment: výber trénovacieho procesu

V rámci môjho prvého experimentu som využil dátovú sadu CASIA_IrisV4_normalized opísanú vyššie. Cieľom experimentu bolo zistiť, aký trénovací postup je pre moje zadanie najvhodnejší. Počas celého experimentu bola použitá sieť ResNet18. Cieľom experimentu bolo porovnanie metód používaných pri učení neurónových sietí.

V prvej časti som sa zameral na metódu trénovania viactriedneho klasifikátora. Ako trénovaciu chybovú funkciu som použil Cross entropy loss. Keďže základné nastavenie siete ResNet18 v knižnici PyTorch je pre klasifikáciu do 10 tried, musel som upraviť poslednú vrstvu, aby pri učení klasifikovala do mnou požadovaného množstva tried (konkrétne 198). Keďže som nepoužil predtrénovaný model, všetky vnútorné váhy boli nastavené náhodne. Ako optimalizátor som použil *Adam* z knižnice PyTorch s hodnotou learn rate nastavenou na 0,001. Najlepšie výsledky som dosiahol s veľkosťou batchu 128. Trénovanie bolo ukončené v momente, keď sa validačná chyba už 10 epoch nezlepšila. V rámci vyhodnotenia som odstránil poslednú plne prepojenú vrstvu, aby sieť namiesto pravdepodobností produkovala vektory príznakov. Následne som použil celú testovaciu sadu na tvoru vektorov, z ktorých som vytvoril dvojice a tie porovnal pomocou Euklidovskej vzialenosti. Výsledky som porovnal s očakávanými výsledkami.

V druhej časti experimentu som sa zameral na metódu kontrastívneho učenia s využitím siamských sietí. Na trénovanie som použil chybovú funkciu Contrastive loss. Opäť som musel upraviť pôvodnú architektúru ResNet18, a to tak že som odstránil poslednú plne prepojenú vrstvu už na začiatku. Najlepšie výsledky sa mi tentokrát podarilo dosiahnuť s veľkosťou batchu 64 a hodnotou learn rate nastavenou na 0,01. Následne som sieť použil na tvorbu vektorov príznakov z testovacej dátovej sady, vytvoril dvojice a porovnal.

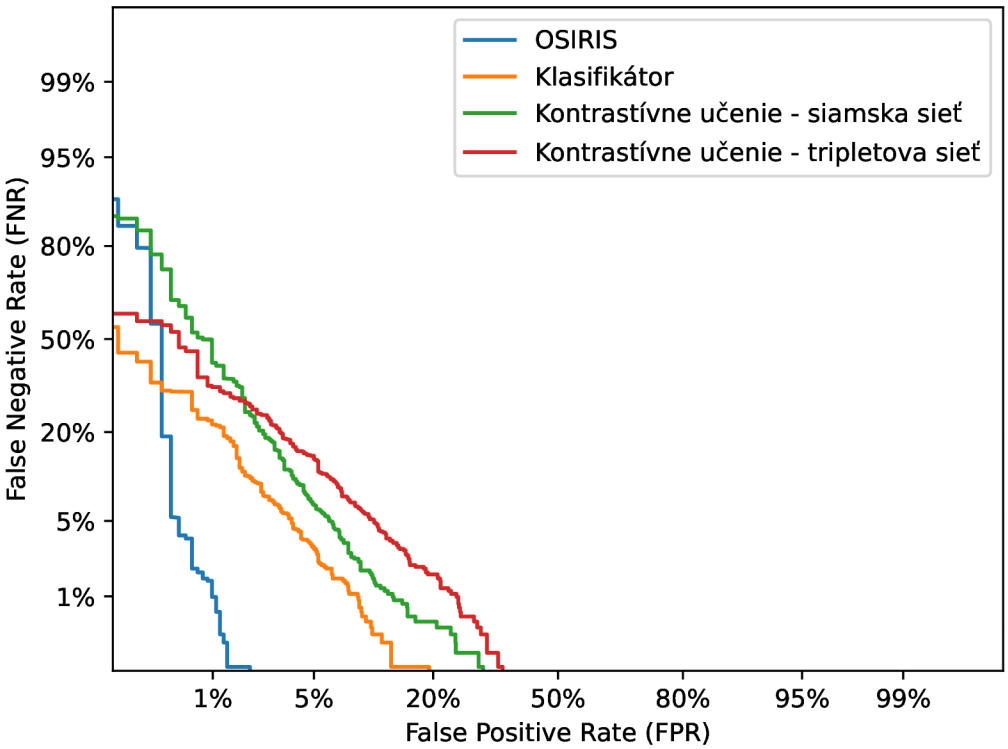
V tretej časti experimentu som sa zameral nametódu kontrastívneho učenia s využitím tripletovej siete. Ako chybovú funkciu som použil Triplet margin loss. Zo začaitku som ponechal trénovacie nastavenia z predošelj časti experimentu, to sa ale ukázalo ako neefek-

tívne. Najlspšie výsledky som bol schopný dosiahnuť pri zmene veľkosti batchu na 128, a pri nastavení chybovej funkcie, kedy sa na tvorbu trojíc použijú všetky obrázky z batchu. Po ukončení trénovania som sieť použil na tvorbu vektorov a porovnanie rovnako ako v minulých častiach experimentu.

Záver prvého experimentu

Po ukončení experimentov som pomocou systému OSIRIS vyrobil dáta na porovnanie mojich výsledkov. Porovnanie výsledných DET kriviek je zobrazené na obrázku 5.1. V tabuľke 5.1 je zobrazené výsledné AUC skóre pre každú sieť natrénovanú počas tohoto experimentu.

Prvý experiment dopadol pozoruhodne. Asi najviac prekvapujúce pre mňa bolo poradie úspešnosti jednotlivých sietí. Aj napriek tomu, že metóda kontrastívneho učenia by mala byť vhodnejšia pre tvorbu vektorov príznakov, najlepšie výsledky dosiahla sieť trénovaná pomocou metódy na trénovanie klasifikátora.



Obr. 5.1: Porovnanie výsledných DET kriviek pre trénovacie metódy sietí z prvého experimentu.

Tabuľka 5.1: Tabuľka AUC skóre pre trénovacie metódy sietí z prvého experimentu

Trénovacia metóda	**AUC skóre**
Klasifikátor	**0.9920**
Kontrastívne učenie - siamska sieť	0.9828
Kontrastívne učenie - tripletova sieť	0.9779

5.2 Druhý experiment: výber hĺbky siete

V druhom experimente som použil novú dátovú sadu, a to konkrétne IITD_normalized. Hlavný dôvod pre použitie tejto sady bol, že pôvodná dátová sada IITD Iris sa mi zdala vizuálne pôsobivejšia, a teda bola vysoká šanca, že vyextrahované obrázky jednotlivých dúhoviek v sebe budú obsahovať lepšiu informáciu. Tým pádom by mohli byť pre neurónovú sieť jednoduchšie na rozpoznanie. Zároveň som chcel v tomto experimente overiť, že voľba siete ResNet18 bola správna, a že hlbšia sieť nieje potrebná.

V prvej časti experimentu som postupoval podobne ako v prvom experimente. Znova som použil architektúru ResNet18 a metódu trénovania viactriedneho klasifikátora s chybovou funkciou Cross entropy loss. Ako najlepšia kombinácia hyperparametrov sa potvrdila veľkosť batchu 128 a learn rate na hodnote 0,001. Oproti minulému experimentu som ale zmenil podmienku ukončenia trénovania tým že som znížil hranicu na 5 epoch.

Po získaní čo najlepšieho klasifikátora som opäť použil metódu kontrastívneho učenia s využitím siamskej siete a chybovej funkcie Contrastive loss. Najlepšie výsledky som dosiahol s veľkosťou batchu 64 a hodnotou learn rate 0,001.

Pri trénovaní metódou kontrastívneho učenia s využitím triplretovej siete som opäť použil Triplet margin loss. Oproti minulému expretimentu som zaviedol jednu zmenu, a to použitie sampleru. Pomocou neho som nastavil dĺžku jedného trénovacie cyklu na 1500 obrázkov čo som neskôr pre dosiahnutie lepšieho výsledku znížil na 1000. Počet obrázkov z jednej triedy som nechal na 5, keďže po zmene tohoto parametru som nepostrehol žiadne badateľné zmeny.

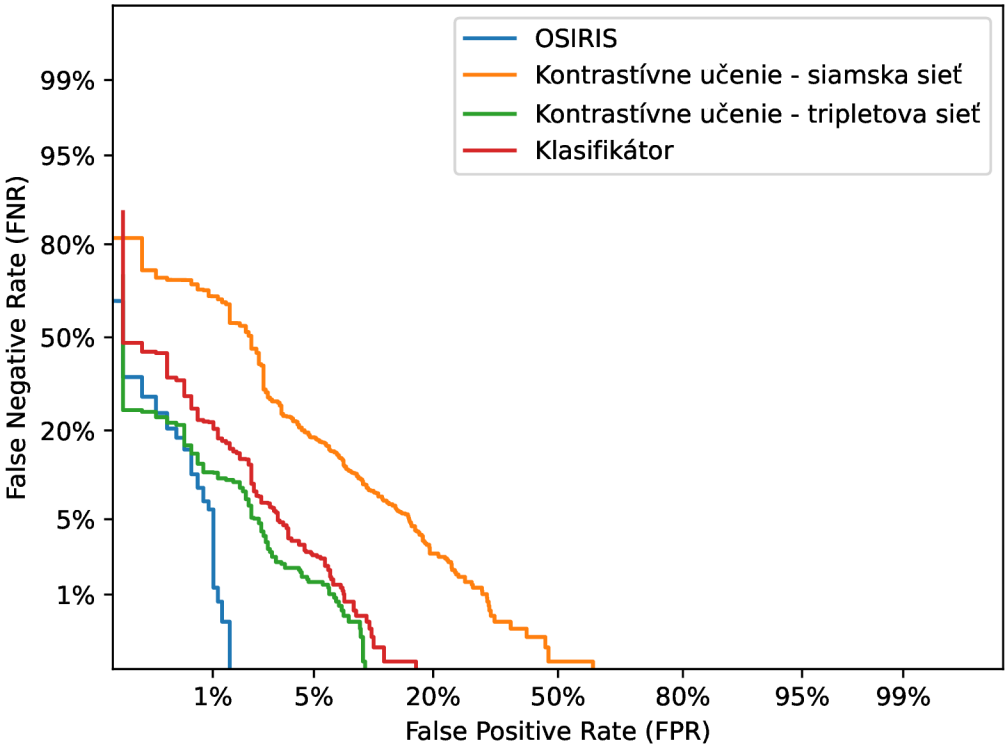
Výsledky neurónových sietí trénovaných na novej dátovej sade sú zobrazené na obrázku 5.2. Príslušné AUC skóre pre každú sieť je v tabuľke5.2.

V tomto experimente sa ukázala ako najpresnejšia sieť tá natrénovaná pomocou kontrastívneho učenia s využitím tripletovej siete. V porovnaní s minulým experimentom, táto sieť dosiahla lepšie AUC skóre ako ktorákoľvek iná. Preto som sa rozhodol pokračovať v rámci experimentu s touto metódou učenia.

Ako hlbšiu sieť som si vybral ResNet50. Pri trénovaní som zvolil menšiu veľkosť batchu - 32. Learn rate som nechal na hodnote 0,001. Dĺžka trénovacieho cyklu ostala na hodnote 1000 obrázkov na cyklus, s 5 obrázkami z jednej triedy. S týmto nastavením som dosiahol najlepšie výsledky.

Tabuľka 5.2: Tabuľka AUC skóre pre trénovacie metódy sietí z druhého experimentu

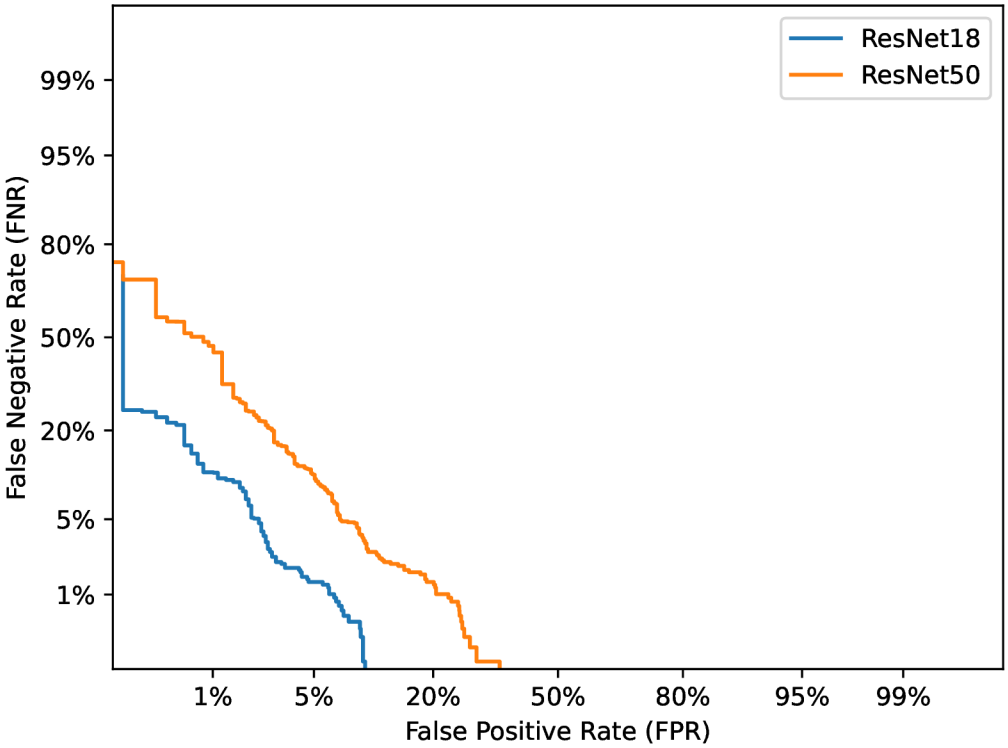
Trénovacia metóda	**AUC skóre**
Klasifikátor	0.9941
Kontrastívne učenie - siamska sieť	0.9644
Kontrastívne učenie - tripletova sieť	**0.9952**



Obr. 5.2: Porovnanie výsledných DET kriviek pre trénovacie metódy sietí z druhého experimentu.

Zhrnutie výsledkov druhého experimentu

Na novej dátovej sade dosahovala najlepšie výsledky neurónová sieť trénovaná metódou kontrastívneho učenia s využitím tripletovej siete. Výsledky porovnania rôznych hĺbok (obrázok 5.3) potvrdili môj predpoklad. Sieť ResNet50 nedokázala dosahovať lepšie výsledky ako sieť ResNet18, napriek veľmi podobným trénovacím podmienkam.



Obr. 5.3: Porovnanie výsledných DET kriviek pre trénovacie metódy sietí z druhého experimentu.

5.3 Tretí experiment: overenie nutnosti segmentácie a normalizácie vstupných dát

V treťom experimente som chcel overiť, či je naozaj segmentácia a následná normalizácia nutná, alebo je možné dosiahnuť podobné alebo lepšie výsledky aj bez nej. Keďže sa v minulých experimentoch osvedčili hlavne metódy trénovania klasifikátora a kontrastívneho učenia s využitím tripletovej siete, rozhodol som sa použiť tieto metódy aj v tomto experimente. V tomto experimente som použil dátovú sadu IITD_cropped.

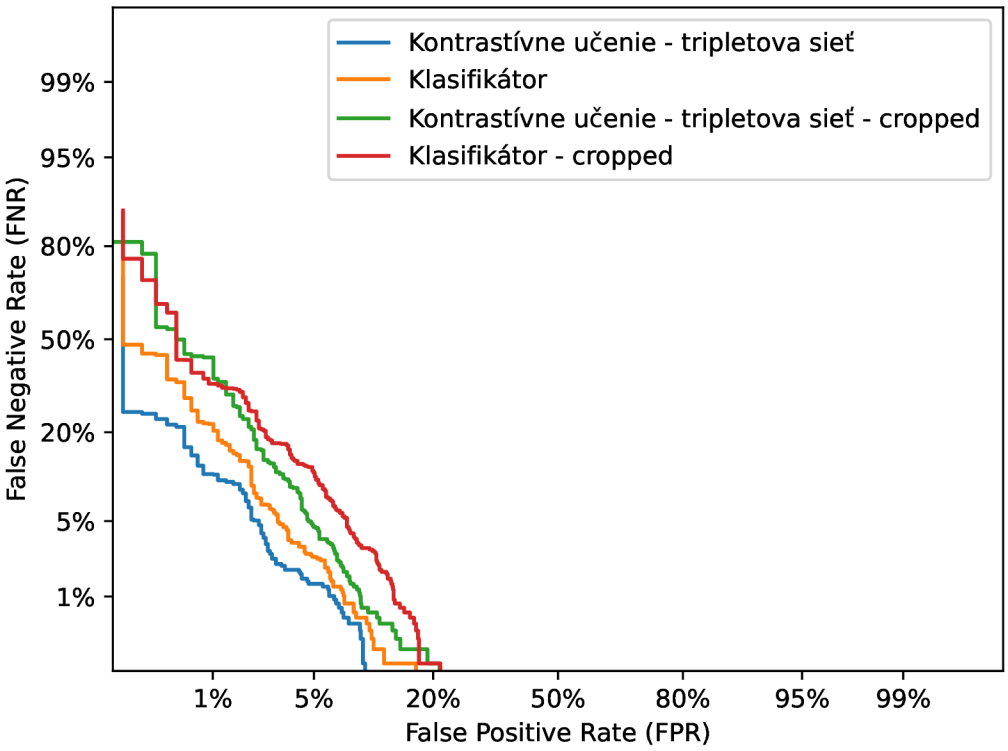
Postup trénovania sa oproti minulým experimentom nezmenil. Najprv som sa zameral na trénovanie trvej siete pomocou metódy trénovania klasifikátora, následne som použil pre trénovanie druhej siete metódu kontrastívneho učenia s využitím tripletovej siete. Pri klasifikácii som použil chybovú funkciu Cross entropy loss, pri kontrastívnom učení chybovú funkciu Triplet margin loss.

Záver tretieho experimentu

Výsledky tretieho experimentu sú zobrazené na obrázku 5.4.

Výsledky jasne dokazujú, že pre dosiahnutie lepších výsledkov je naozaj nutné použiť normalizovaný obrázok dúhovky. Na druhej strane výsledok ukazuje, že v prípade ak by

segmentácia a normalizácia nebola možná, sieť je schopná rozlišovať aj obrázky celých očí, aj keď s menšou presnosťou.



Obr. 5.4: Porovnanie výsledných DET kriviek pre trénovacie metódy sietí z tretieho experimentu. Výsledky ukazujú väčšiu presnosť pri použití normalizovaných obrázkov dúhovky.

Kapitola 6

Záver

Cieľom tejto práce bolo vytvoriť systém umožňujúci identifikáciu osôb pomocou obrazu dúhovky. Tohoto cieľa sa podarilo dosiahnúť využitím neurónovej siete natrénovanej za účelom extrakcie príznakov z obrázka dúhovky.

V práci sú popísané tradičné postupy používané pri identifikácii osôb pomocou dúhovky, ako aj už existujúci systém, ktorý tieto postupy využíva. Ďalej je popísané využitie strojového učenia v tejto oblasti, v rámci ktorého sú opísané najpoužívanejšie architektúry konvolučných neurónovych sietí. Súčasťou práce je aj mnou navrhnutý systém využívajúci konvolučnú neurónovú sieť ResNet. Tá bola natrénovaná na čo najlepšiu extrakciu príznakov, ktoré boli následne porovnané pomocou euklidovskej vzdialenosti. Táto vzdialenosť následne určovala, či ide o tú istú osobu alebo nie. Pri experimentoch v rámci trénovania tejto siete boli použité dve metódy trénovania, a to trénovanie viactriedneho klasifikátora a trénovanie pomocou metódy kontrastívneho učenia. Taktiež bola overená teória, či je segmentácia a následná normalizácia obrazu dúhovky naozaj potrebná. V rámci experimentov boli využívané verejne dostupné dátové sady, ktoré boli upravené pre konkrétne experimenty. Súčasťou záveru každého experimentu je porovnanie výsledkov neurónových sietí pomocou DET kriviek a AUC skóre.

Pri ďalšom vývoji by sa dalo zamerať na ostatné architektúry konvolučných neurónových sietí, poprípade vytvorenie vlastnej neurónovej siete. Ďalej existuje množstvo rôznych postupov v rámci kontrastívneho učenia, ktoré môžu mať pozitívny dopad na presnosť neurónovej siete.

Literatúra

[1] Agrawal, S. K. *Metrics to evaluate your classification model to take the right decisions*. Dec 2022. Dostupné z: https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions.

[2] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al Dujaili, A. Q., Duan, Y. et al. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*. 2021, zv. 8.

[3] Bezdan, T. a Bacanin, N. Convolutional Neural Network Layers and Architectures. In:. Január 2019, s. 445–451.

[4] *CASIA-IrisV4*. Dostupné z: http://www.cbsr.ia.ac.cn/china/Iris%20Databases%20CH.asp.

[5] Daugman, J. Information Theory and the IrisCode. *IEEE Transactions on Information Forensics and Security*. 2016, zv. 11, č. 2, s. 400–409.

[6] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. et al. Imagenet: A large-scale hierarchical image database. In: Ieee. *2009 IEEE conference on computer vision and pattern recognition*. 2009, s. 248–255.

[7] Deng, J., Guo, J., Yang, J., Xue, N., Kotsia, I. et al. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Institute of Electrical and Electronics Engineers (IEEE). oct 2022, zv. 44, č. 10, s. 5962–5979. Dostupné z: https://doi.org/10.1109%2Ftpami.2021.3087709.

[8] Deng, L. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*. IEEE. 2012, zv. 29, č. 6, s. 141–142.

[9] Deshmukh, R. a Ghule, K. Feature Extraction Techniques for Speech Recognition: A Review. *International Journal of Scientific and Engineering Research*. Máj 2015, zv. 6, s. 143–147.

[10] Deshpande, A. *The 9 Deep Learning Papers You Need to know about (understanding CNNS part 3)*. Aug 2016. Dostupné z: https://adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html.

[11] Fawcett, T. Introduction to ROC analysis. *Pattern Recognition Letters*. Jún 2006, zv. 27, s. 861–874.

[12] Francese, R., Frasca, M., Guarino, A., Malandrino, D., Risi, M. et al. On the Limitation of Pathological Iris Recognition: Neural Network Perspectives. In: *2020 24th International Conference Information Visualisation (IV)*. 2020, s. 68–73.

[13] Ghojogh, B., Sikaroudi, M., Shafiei, S., Tizhoosh, H., Karray, F. et al. Fisher Discriminant Triplet and Contrastive Losses for Training Siamese Networks. In: *2020 International Joint Conference on Neural Networks (IJCNN)*. 2020, s. 1–7.

[14] He, K., Zhang, X., Ren, S. a Sun, J. *Deep Residual Learning for Image Recognition*. 2015.

[15] Hoffer, E. a Ailon, N. *Deep metric learning using Triplet network*. 2018.

[16] *IIT Delhi Iris Database*. Dostupné z: https://www4.comp.polyu.edu.hk/~csajaykr/IITD/Database_Iris.htm.

[17] Kaushik, P. *Still using accuracy as a classification metric?* Towards Data Science, Feb 2021. Dostupné z: https://towardsdatascience.com/understanding-top-n-accuracy-metrics-8aa90170b35.

[18] Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y. et al. *Supervised Contrastive Learning*. 2021.

[19] Koech, K. E. *Cross-entropy loss function*. Towards Data Science, Jul 2022. Dostupné z: https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e.

[20] Krizhevsky, A., Sutskever, I. a Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In: Pereira, F., Burges, C., Bottou, L. a Weinberger, K., ed. *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2012, sv. 25. Dostupné z: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[21] Lecun, Y., Bottou, L., Bengio, Y. a Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*. 1998, zv. 86, č. 11, s. 2278–2324.

[22] Luo, W., Li, Y., Urtasun, R. a Zemel, R. *Understanding the Effective Receptive Field in Deep Convolutional Neural Networks*. 2017.

[23] Martin, A., Doddington, G., Kamm, T., Ordowski, M. a Przybocki, M. The det curve in assessment of detection task performance. *The DET Curve in Assessment of Detection Task Performance*. Január 1997, s. 1895–1898.

[24] MEFANET. *Roc Křivka*. 2023. Dostupné z: https://www.wikiskripta.eu/w/ROC_k%C5%99ivka#/media/Soubor:Receiver_Operating_Characteristic.png.

[25] Omran, M. a AlShemmary, E. N. An Iris Recognition System Using Deep convolutional Neural Network. *Journal of Physics: Conference Series*. IOP Publishing. may 2020, zv. 1530, č. 1, s. 012159. Dostupné z: https://dx.doi.org/10.1088/1742-6596/1530/1/012159.

[26] O'Shea, K. a Nash, R. An Introduction to Convolutional Neural Networks. *ArXiv e-prints*. November 2015.

[27] Othman, N., Dorizzi, B. a Garcia Salicetti, S. OSIRIS: An open source iris recognition software. *Pattern Recognition Letters*. 2016, zv. 82, s. 124–131. ISSN 0167-8655. An insight on eye biometrics. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0167865515002986.

[28] Padma Polash, P. a Monwar, M. M. Human iris recognition for biometric identification. In: *2007 10th international conference on computer and information technology*. 2007, s. 1–5.

[29] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B. et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 2011, zv. 12, s. 2825–2830. Dostupné z: https://scikit-learn.org/stable/_images/sphx_glr_plot_det_001.png.

[30] Proença, H. Non-cooperative iris recognition: Issues and trends. In: *2011 19th European Signal Processing Conference*. 2011, s. 2259–2263.

[31] Rana, H., Azam, M., Akhtar, M. R., Quinn, J. a Moni, M. A. A fast iris recognition system through optimum feature extraction. *PeerJ Computer Science*. Apríl 2019, zv. 5, s. e184.

[32] Ruiz Albacete, V., Tome Gonzalez, P., Alonso Fernandez, F., Galbally, J., Fierrez, J. et al. Direct Attacks Using Fake Images in Iris Verification. In: Schouten, B., Juul, N. C., Drygajlo, A. a Tistarelli, M., ed. *Biometrics and Identity Management*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, s. 181–190. ISBN 978-3-540-89991-4.

[33] Schroff, F., Kalenichenko, D. a Philbin, J. FaceNet: A unified embedding for face recognition and clustering. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, s. 815–823.

[34] Simonyan, K. a Zisserman, A. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015.

[35] Sodagudi, S., Manda, S., Smitha, B., Chaitanya, N., Ahmed, M. A. et al. EEG signal processing by feature extraction and classification based on biomedical deep learning architecture with wireless communication. *Optik*. 2022, zv. 270, s. 170037. ISSN 0030-4026. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0030402622012955.

[36] Song, H. O., Xiang, Y., Jegelka, S. a Savarese, S. *Deep Metric Learning via Lifted Structured Feature Embedding*. 2015.

[37] Sun, Y., Wang, X. a Tang, X. Deep Learning Face Representation from Predicting 10,000 Classes. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, s. 1891–1898.

[38] Szegedy, C., Ioffe, S., Vanhoucke, V. a Alemi, A. *Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning*. 2016.

[39] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. et al. Going deeper with convolutions. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, s. 1–9.

[40] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. a Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, s. 2818–2826.

[41] Taigman, Y., Yang, M., Ranzato, M. a Wolf, L. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, s. 1701–1708.

[42] *The Hong Kong Polytechnic University cross-spectral Iris Images Database*. Dostupné z: https://www4.comp.polyu.edu.hk/~csajaykr/polyuiris.htm.

[43] Tiu, E. *Understanding contrastive learning*. Towards Data Science, Jan 2021. Dostupné z: https://towardsdatascience.com/understanding-contrastive-learning-d5b19fd96607.

[44] Wang, D., Su, J. a Yu, H. Feature Extraction and Analysis of Natural Language Processing for Deep Learning English Language. *IEEE Access*. 2020, zv. 8, s. 46335–46345.

[45] Wang, J. a Ko, K. DeepSign: Efficient Siamese Convolutional Neural Networks for Signature Verification. Jún 2019.

[46] Wang, S. A Review of Gradient-Based and Edge-Based Feature Extraction Methods for Object Detection. In: *2011 IEEE 11th International Conference on Computer and Information Technology*. 2011, s. 277–282.

[47] Wei, J. *VGG neural networks: The next step after Alexnet*. Towards Data Science, Jul 2019. Dostupné z: https://towardsdatascience.com/vgg-neural-networks-the-next-step-after-alexnet-3f91fa9ffe2c.

[48] Weng, L. *Contrastive Representation Learning*. May 2021. Dostupné z: https://lilianweng.github.io/posts/2021-05-31-contrastive/.

[49] Wicht, B. *Deep Learning feature Extraction for Image Processing*. 2018. Dizertačná práca. University of Fribourg.

[50] Xie, E., Ding, J., Wang, W., Zhan, X., Xu, H. et al. DetCo: Unsupervised Contrastive Learning for Object Detection. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, s. 8372–8381.

[51] Yamashita, R., Nishio, M., Do, R. a Togashi, K. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*. Jún 2018, zv. 9.

[52] Yin, Y., He, S., Zhang, R., Chang, H., Han, X. et al. *Deep Learning for Iris Recognition: A Review*. Marec 2023.

Príloha A

Obsah priloženého pamäťového média

Adresárová štruktúra priloženého pamäťového média:

- **latex/** - obsahuje zdrojové súbory textovej správy pre LaTeX
- **nets/** - obsahuje predtrénované modely sietí
- **poster/** - obsahuje plagát
- **source/** - obsahuje zdrojové kódy a dokumentáciu
- **video/** - obsahuje video
- **README.pdf** - podrobnejšie informácie o priloženom médiu
- **doc.pdf** - text bakalárskej práce